SYSMAC CS 系列

CS1G/H-CPU@@-EV1
CS1G/H-CPU@@H
CS1D-CPU@@H

SYSMAC CJ 系列

CJ1G-CPU@@
CJ1G/H-CPU@@H
CJ1M-CPU@@

可编程序控制器

指令参考手册

2003 年 7 月

注意：

OMRON 公司生产的产品是由合格的操作员按正常步骤使用，并且仅用于本手册所描述的用途。

本手册中的以下约定用于指明手册中的注意事项和分类。必须始终注意它所规定的情况。疏忽这些注意事项可能引起人身伤害或财产的损坏。

! **危险** 表示一个紧迫的危急情况，如果不避免将导致死亡或严重伤害。

! **警告** 表示一个潜在的危急情况，如果不避免将导致死亡或严重伤害。

! **注意** 表示一个潜在的危急情况，如果不避免将导致中轻度伤害或财产损失。

OMRON 产品附注

在本手册中，所有 OMRON 产品均以大写字母开头。当 “Unit”表示 OMRON 产品名称时，不管它是否是产品的正式名称，也以大写字母表示。

缩写 “Ch”出现在某些显示和某些 OMRN 产品中时，往往表示 “Word”。在这个意义上，文件中缩写为 “Wd”。

缩写 “PLC”表示可编程序控制器。然而，在一些编程装置显示中所用到的缩写 “PC”也表示可编程序控制器。

直观标题

出现在本手册左侧的下列标题帮助你区分不同类型的信息。

注 表示对有效和方便操作产品特别重要的信息。

1,2,3... 1. 表示一种或另一种的列举说明，例如操作步骤，检查表等等。

目录

目录

关于本手册：

本手册描述了用于 CS/CJ 系列可编程序控制器 (PLC)CPU 单元所支持的梯形图编程指令。CS 系列和 CJ 系列由下表分述。

单元	CS 系列	CJ 系列
CPU 单元	CS1-H CPU 单元：CS1H-CPU@@H CS1G-CPU@@H	CJ1-H CPU 单元：CJ1H-CPU@@H CJ1G-CPU@@H
	CS1 CPU 单元：CS1H-CPU@@-EV1 CS1G-CPU@@-EV1 CS1D CPU 单元：CS1D-CPU@@H	CJ1 CPU 单元：CJ1G-CPU@@-EV1 CJ1M CPU 单元：CJ1M-CPU@@
基本 I/O 单元	CS 系列基本 I/O 单元	CJ 系列基本 I/O 单元
特殊 I/O 单元	CS 系列特殊 I/O 单元	CJ 系列特殊 I/O 单元
CPU 总线单元	CS 系列 CPU 总线单元	CJ 系列 CPU 总线单元
电源单元	CS 系列电源单元	CJ 系列电源单元

在可编程序控制器系统中试图使用 CS/CJ 系列 CPU 单元和编程前，请仔细阅读本手册及下页表中所列的相关手册，确保理解所有信息。

第 1 章 介绍 CS/CJ 系列 PLC 所支持的指令集。

第 2 章 提供用于参考的各种指令列表。

第 3 章 分别描述 CS/CJ 系列指令集中的指令。

第 4 章 提供每一条 CS/CJ 系列指令的指令执行时间和步数。

本手册及相关内容手册

名称	书号	内容
SYSMAC CS/CJ 系列 CS1G/H-CPU@@-EV1， CS1G/H-CPU@@H， CS1D-CPU@@H， CJ1M-CPU@@， CJ1G-CPU@@， CJ1G/H-CPU@@H 可编程序控制器操作指令参考手册	W340	描述 CS/CJ 系列 PLC 可编程序控制器所支持的梯形图编程指令 （本手册）。
SYSMAC CS/CJ 系列 CS1G/H-CPU@@-EV1， CS1G/H-CPU@@H， CS1D-CPU@@H， CJ1M-CPU@@， CJ1G-CPU@@， CJ1G/H-CPU@@H 可编程序控制器编程手册	W394	手册描述 CS/CJ 系列 PLC 可编程序控制器的编程和使用该系列可编程序控制器功能的其它方法（本手册）。
SYSMAC CS 系列 CS1G/H-CPU@@-EV1， CS1G/H-CPU@@H 可编程序控制器操作手册	W339	提供 CS 系列 PLC 可编程序控制器概述及其设计、安装维护及其基本操作说明。
SYSMAC CJ 系列 CJ1M-CPU@@， CJ1G-CPU@@， CJ1G/H-CPU@@H 可编程序控制器操作手册	W393	提供 CJ 系列 PLC 可编程序控制器概述及其设计、安装维护及其基本操作说明。
SYSMAC CJ 系列 CJ1M-CPU22/23 内置式 I/O 功能操作手册	W395	介绍 CJ1M CPU 单元内置式 I/O 功能。
SYSMAC CS 系列 CS1D-CPU@@H CPU 系列 CS1D-DPL1 双单元 CS1D-PA207R 电源单元 双系统操作手册	W405	提供基于 CS1D CPU 单元的双系统概述及其设计、安装、维护及其它基本操作说明。
SYSMAC CS/CJ 系列 CQM1H-PRO01-E， C200H-PRO27-E， CQM1-PRO01-E 编程器操作手册	W341	说明如何使用编程器来编程和操作 CS/CJ 系列 PLC 可编程序控制器。
SYSMAC CS/CJ 系列 CS1G/H-CPU@@-EV1， CS1G/H-CPU@@H， CJ1M-CPU@@， CJ1G-CPU@@， CJ1G/H-CPU@@H， CS1W-SCB21/41， CS1W-SCU21， CJ1W-SCU41 通信指令参考手册	W342	介绍 CS/CJ 系列可编程序控制器用的 PLC 系列（上位链接）和 FINS 通信指令。
SYSMAC WS02-CXP@@-E CX-Programmer 用户手册 3.0	W361	介绍 CS/CJ 系列可编程序控制器用的 PLC 系列（上位链接）和 FINS 通信指令。
SYSMAC WS02-CXP@@-E CX-Server 用户手册	W362	
SYSMAC CS/CJ 系列 CS1W-SCB21-V1/41-V1， CS1W-SCU21-V1， CJ1W-SCU41 串行通信卡 / 单元操作手册	W336	说明如何使用 CX-Programmer，支持 CS/CJ 系列可编程序控制器的一个编程设备及在 CX-Programmer 内部的 CX-NET。
SYSMAC WS02-PSTC1-E CX-Protocol 操作手册	W344	介绍生成协议宏作为通信序列与外部设备通信的 CX-Prtocol 的使用。
SYSMAC CS/CJ 系列 CJ1W-ETN01/ENT11， CJ1W-ETN11 以太网操作手册	W343	CJ1W-ETN01， CJ1W-ENT11 和 CJ1W-ETN11 以太网单元的安装和操作说明。

！警告 不阅读或不理解本手册所提供的内容，可能导致人身伤亡，危及产品或使产品发生故障。因此，在着手进行所提供的任何步骤操作前，请全面、仔细阅读每个章节，并确保已理解了本章节所提供的信息及相关章节内容。

注意事项

本章提供了使用 CS/CJ 系列可编程序控制器 (PLC) 和相关设备的一般性注意事项。

本章中所包含的内容对于安全可靠地使用可编程序控制器是非常重要的。用户在着手安装或使用可编程序控制器系统前，务必阅读并理解本章内容。

1 面向的读者

本手册是为下列人员编写的，他必须具有电气系统知识（电气工程师或具有相当水平者）。

- 从事 FA 系统的安装人员。
- 从事 FA 系统的设计人员。
- 从事 FA 系统及设备的管理人员。

2 一般注意事项

用户必须按照操作手册中给出的性能，规格来使用产品。

在将本产品用于本手册中未述及的条件下，或将产品应用与核控制系统，铁路系统，航空系统，车辆，内燃机系统，医疗设备，娱乐机械，安全装置，或若使用不当时可能会对生命和财产造成严重影响的其它系统，机械及装置前，请务必咨询欧姆龙的特约经销商。

请确保本产品的额定值和性能特性满足系统，机械和装置的要求，务必给系统、机械和装置提供双重安全机制。

本手册编有供单元的编程和操作用的资料，在着手使用前务必阅读本手册，并将手册备在身边以供操作时参阅。

! 警告 可编程序控制器和所有可编程序控制器单元用于规定的用途和规定的条件下是十分重要的，特别在会直接或间接地影响到人的生命的应用中。在将可编程序控制器系统应用于上述情况前，请前务必咨询欧姆龙的特约经销商。

3 安全注意事项

! 警告 即使在程序没有运行的情况下（即，即使在编程模式下），CPU 单元也会刷新 I/O。在改变分配给 I/O 单元、特殊 I/O 单元、或 CPU 总线单元的存储器中的任何部分的状态前确认完全安全。对分配给任一单元的数据的改变，可能引起与单元连接的负载发生不可预料的操作。下列操作中的任何一种都可能引起存储器状态的改变。

- 将 I/O 存储器数据从编程设备传送到 CPU 单元。
- 从编程设备改变存储器中的当前值。
- 从编程设备强制置位 / 复位。
- 从存储器卡或 EM 文件存储器传送 I/O 存储器文件到 CPU 单元。
- 从上位计算机或网络上的另一可编程序控制器传送到 I/O 存储器。

! 警告 在带电的情况下不要试图拆卸任何单元，否则会导致电击。

! 警告 在带电的情况下不要触及任何接线端和端子板，否则可能导致电击。

! 警告 不要试图拆卸、修理或改装任何单元，任何这类做法或企图都可能导致故障、火灾或电击。

! 警告 不要在带电的情况下或刚断开电源后立即接触电源，否则可能导致电击。

! 警告 为了在因可编程序控制器或其它影响可编程序控制器操作的外部因素引起异常时确保系统安全，外部电路中 （即不在可编程序控制器内）要设置安全措施，否则可能导致严重事故。

- 在外部控制电路中必须设有紧急停止电路、联锁电路、限位电路以及类似的安全措施。
- 在自诊断功能检测任何错误时或在执行严重故障报警 (FALS) 指令时，可编程序控制器会将所有的输出置 OFF 状态。为了保证系统的安全，必须设有外部安全措施。作为这种故障的防范措施。
- 由于输出继电器的卡死、烧坏或输出晶体管的损坏，可编程序控制器输出可能保持在 ON 或 OFF 状态。作为这个问题的防范措施，必须提供外部安全措施以保证系统安全。
- 在 24V 直流输出 （可编程序控制器的工作电源）过载或短路时，电压可能下降并使各输出变为 OFF。为了保证系统的安全，必须设有外部安全措施，作为这个问题的防范措施。

! 注意 在使用外部工具将存储在文件存储器 （存储器卡或 EM 文件存储器）中的数据文件传送到 CPU 单元的 I/O 区 (CIO) 之前要确认安全。否则，与输出单元连接的设备可能误动作，而不顾 CPU 单 元的操作模式。

! 注意 只有在确认延长循环时间不会引起不利的作用后才执行在线编辑，否则，输入信号可能不能读到。

! 注意 当用户程序和参数数据写入CPU单元时，CS1-H、CJ1-H、CJ1M和CS1D CPU单元自动将用户程序和参数数据后备在闪存器内。然而，I/O存储器（包括DM、EM 和 HR 区）数据不写入闪 存器。在电源中断时， DM、EM 和 HR 区数据由电池保持。如果电池出问题，在电源中断后这些区域中的数据可能出错。每当电池出错标志 (A40204) 为 ON，如果 DM、 EM 和 HR 区域内容用于控制外部输出，应防止不适当的输出发生。

! 注意 在传送程序给其它节点或改变I/O存储器区的内容前要确认目标节点的安全。在没有确认安全的情况下传送可能导致伤害发生。

! 注意 用操作手册中规定的力矩来拧紧AC电源单元端子板上的螺丝，螺丝松动可能引起燃烧或误动作。

! 注意 不要在带电的情况下或刚断开电源后立即接触电源单元，否则，电源单元将会烫伤或烧伤操作者。

4 操作环境注意事项

! 注意 请勿在下列场所操作控制系统：

- 阳光直射处；
- 温度或湿度超出规格中规定范围处；
- 温度急剧变化易引起结露处；
- 有腐蚀性气体和易燃性气体处；
- 有尘埃 （特别是铁屑）或盐雾处；
- 暴露于水、油、或化学品处；
- 易受冲击或振动处。

! 注意 将系统安装在下列场所时需采取适当和有效的预防措施：

- 有静电或其它形式噪音处；
- 有较强电磁场处；
- 可能暴露于射线处；
- 靠近于动力电源处。

! 注意 可编程序控制器的工作环境对系统的可靠和寿命具有很大的影响，非正常的工作环境会导致可编程序控制器系统误动作、故障及其它不可预料的问题出现。系统安装及在寿命期内应确保工作环境在规定的条件内。

5 应用注意事项

使用可编程序控制器系统时要遵循下列注意事项。

- 如果要编制一个以上任务，必须使用 CX-Programmer （在 Windows 上运行的编程软件） 。手持编程器只能用来编制一个循环任务加上几个中断任务程序。然而手持编程器可以用来编辑原先用 CX- Programmer 生成的任务程序。
- 当使用 C200H 特殊 I/O 单元与下列功能结合时，访问 CS 系列 CS1 CPU 单元的 I/O 存储器区域及地址会受到限制。
  - 当内部一个 ASCII 单元用 PLC READ，PLC WRITE 及类似命令编程传送， CPU 单元传送数据会受限制。
  - 在 CPU 单元的数据传递中对分配位和 DM 区分类有限制（源和目标区域和地址的分类）。

- 用现场总线网 (CompoBus/D) 主单元 (CIO 0050 ～ CIO 0099) 的现场总线网 (CompoBus/D) 输出区与 I/O 位区 (CIO 0000 ～ CIO 0319) 重叠。对分配现场总线网络系统会与 I/O 单元分配重叠的任何系统，不要采用自动分配的方式。相反，应采用对现场总线网络设备用编程设备或 CX-Programmer 人工 I/O 配址，确认相同字或位不会分配二次以上后将 I/O 表的结果传送到 CPU 单元。如果当相同位被同时分配到现场总线网络设备和 I/O 单元（这种情况的出现，即使是采用自动分配方式）现场总线网络设备和 I/O 单元可能都将显示出错误操作。
- 用于 PLC 连接单元 (CIO 0247 ～ CIO 0250) 的特殊位和标志与 I/O 位区域 (CIO 0000 ～ CIO0319) 重叠。不要对在 I/O 单元分配将产生与 I/O 单元重叠的任何系统采用 I/O 自动分配。取而代之的是使用编程装置或 CX-Programmer 对 I/O 单元人工 I/O 分配。确信 PLC 连接单元的特殊位标志没被使用后，将 I/O 结果表输送到 CPU 单元。如果 PLC 连接单元的特殊位和标志也被分配到 I/O 单元 （这种情况的出现即使是采用自动分配方式）， PLC 连接单元和 I/O 单元可能都将显示出错操作。

！ 警告 请始终注意这些注意事项。不遵守下列各注意事项可能导致严重伤害，甚至致命伤害。

- 在安装单元时，总是连接到接地电阻不大于 100 Ω 的地线上。连接到大于 100 Ω 的接地电阻地线上，将可能导致电击。
- 在短接电源单元的 GR 和 LG 端子时，必须安装一个不大于 100 Ω 接地电阻的接地。
- 在着手做下列任一项工作前，总是将 PLC 上的电源关断（置 OFF）。否则可能引起误动作或电击。
  - 安装或拆卸电源单元、I/O 单元、CPU 单元、内部板子或其它任何单元。
  - 装配各单元。
  - 设定 DIP 开关或旋转开关。
  - 连接系统电缆或电线。
  - 连接或断开连接器。

！ 注意 不注意下列注意事项可能引起 PLC 或系统的错误操作，或可能危及 PLC 或 PLC 单元。请始终注意这些注意事项。

- 在 CS1-H、CS1D、CJ1-H 和 CJ1M CPU 单元内的用户程序和参数区数据后备在内置式闪存器中。当后备操作进行时，CPU 单元前面的 BKUP 指示灯会点亮。当 BKUP 指示灯点亮时，不要断开 CPU 单元的电源。如果断开的话，则数据将不被后备。

- CS1-H、CS1D、CJ1、CJ1-H 和 CJ1M CPU 单元在出厂时已装有电池，内部时钟已设定时间。它如同 CS 系列 CS1 CPU 单元一样，在使用前不必清除存储器内容或设置时钟。
- 当第一次使用 CS 系列 CPU 单元时，安装随单元提供的 CS1W-BAT1 电池，并且在开始编程前，用编程装置清除所有存储区。使用内部时钟时，将电池安装好后接通电源，并用编程装置或利用 DATE(735) 指令设置时钟。时钟在时间未设置前是不启动的。
- 当 CPU 单元出厂时，PLC 已经设置好。所以 CPU 单元将由手持编程器上的开关设置从操作模式开始。当手持编程器未连接，CS 系列 CS1 CPU 单元将从程序模式开始，但 CS1-H、CS1D、CJ1、CJ1-H 或 CJ1M CPU 单元从运行模式开始，并立即开始操作。在没有确认安全的情况下，不要随意允许操作开始。
- 当用编程装置（手持编程器或 CX-Programmer）生成一个 AUTOEXEC、IOM 文件以在启动期间自动传送数据时，将第一个写地址置在 D20000，并确保所数据的量不超出 DM 区。即使在生成 AUTOEXEC、IOM 文件时设置了另外地址，启动时从存储器卡读数据文件时数据会被写入 CPU 单元 D20000 起的数据区。此外，如果数据量超出 DM（使用 CX-Programmer 时是可能的），则剩余数据会被写到 EM 区。
- 请在接通控制系统电源前总是先接通 PLC 电源。如果 PLC 电源是在接通控制电源后接通的话，则可能导致控制系统信号暂时出错。因为 PLC 电源接通时，在直流输出单元和其它单元的输出端子上会瞬时变为 ON。
- 为了在出现内部电路故障而引起输出单元的输出保持 ON 的情况下保证安全，用户必须采取故障安全措施。这种情况可能发生在继电器、晶体管或其它元件上。
- 为了保证在信号线断开、瞬时电源中断或其它原因引起信号不正确、丢失或异常情况下安全，用户必须采取故障安全措施。
- 用户必须在外部电路中（即不在可编程序控制器内）设置联锁电路、限位电路和类似安全措施。
- 在传送数据时不要断开 PLC 电源，特别是在读或写存储器卡时不要断开电源。此外，在 BUSY 指示灯点亮时也不要取出存储器卡。如要取出存储器卡，请先按存储器卡开关，等 BUSY 指示灯熄灭，然后取出存储器卡。
- 如果I/O保持位变成ON，则PLC从RUN或MONTOR模式切换为PROGRAM模式时，PLC 的输出不会变 OFF，而会保持其原先的状态。请确保这种情况发生时外部负载不会产生危险。（当操作因致命错误停止时，包括 FALS(007) 指令产生的结果，输出单元的所有输出全都变为 OFF，而只有内部输出状态会被保持）。

- CPU 单元中的 DM、 EM 和 HR 区的内容都由电池后备。如果电池电压下降，数据可能丢失。在程序中提供预防措施，如果电池电压下降，使用电池出错标志 (A40204) 再初始化数据或采取其它行动。
- 当 CS 系列 PLC 用 200 ～ 240V 交流电供电时，总是把电源单元上电压选择端子处金属跳接线去除 （除非电源单元具有很大电压范围的规格）如果用 200 ～ 240V 交流电供电，而金属跳接线仍接着，那么将损坏产品。
- 总是使用操作手册中规定的电源电压，不正确的电压可能导致误动作或燃烧。
- 请采取适当措施保证提供具有额定电压和频率的指定电源。请特别注意供电不稳定的地方，不正确的电源可能导致误动作。
- 请安装外部短路器和采取其它安全措施，防止外部接线短路。防短路安全措施不充分可能导致燃烧。
- 切勿将高于额定输入电压的电压施加输入单元，过高的电压可能导致燃烧。
- 切勿将超出最大开关容量的电压或负载接到输出单元，过电压或过载可能导致燃烧。
- 当进行耐压试验时，要断开功能接地端，否则可能导致燃烧。
- 请按操作手册中的规定正确地安装单元。不正确地安装单元，可能导致误动作。
- 对 CS 系列 PLC 要确保所有的单元和底板的固定螺丝按手册的规定力矩拧紧，不正确的拧紧力矩，可能导致误动作。
- 确保端子螺丝和电缆连接器螺丝均按有关手册所规定的力矩拧紧。否则可能导致误动作。
- 在接线时请保留粘在单元上的标签。撕去标签后会使异物落入单元而导致误动作。
- 为确保合适的散热效果，在完成全部接线后撕去标签。保留标签会影响散热而导致误动作。
- 接线时使用压接端子，不要把多股线直接连到端子上，这样的连接可能导致燃烧。
- 正确连接所有接线。
- 通电前，请对所有接线和开关设置进行双重检查。错误的接线会导致燃烧。
- 安装单元应在彻底检查端子板和连接器后进行。
- 确认端子板、内存单元，扩展电缆和其它具有固定装置的设备被正确固定好，否则将导致误动作。

- 在开始操作前，应检查开关设置、DM 区的内容及其它准备工作。不正确的设置与数据在启动操作时可能导致不可预料的动作。
- 用户程序在单元中正式运行前须作检查，否则将导致不可预料的运行。
- 在着手下列任何一项工作前，请确认在系统中是否会发生不利影响。否则可能导致不可预料的动作。
  - 改变 PLC 的操作模式。
  - 对存储器中的某一位强制置位 / 复位。
  - 改变存储器中某一字或设定值的当前值。
- 在把 DM 区、HR 区的内容及其它恢复运行所需的数据传送到新的 CPU 单元后再恢复运行。否则可能会导致不可预料的动作。
- 不要拽拉或弯折电缆超过其允许的限度。其中任何一种行为都可能导致电缆断裂。
- 不要在电缆或其它接线上堆放物品，否则可能导致电缆断裂。
- 不要使用商用个人计算机 RS-232C 电缆。应使用本手册中列出的专用电缆或按手册规格制作电缆。使用商用电缆可能危及外部设备或 CPU 单元。
- 万不要将 CPU 单元上 RS-232C 口上的引脚 6 （5V 电源）与其它设备连接，除非设备上用了 NT-AL001 或 CJ1W-C1F11 适配器。否则会损坏外部设备或 CPU 单元。
- 当更换零件时，务必确认新零件的额定值是否正确。否则可能导致误动作或燃烧。
- 在接触单元前，为使所聚积的静电放电，务必先接触接地金属物。否则可能导致误动作或损害。
- 在运输或存储电路板时，为防止受静电影响，应用抗静电材料将其包上，并注意保持适当的储存温度。
- 不要裸手接触各电路板或安装在电路板上的零件。电路板上有尖刺的引线和其它部件，否则可能引起伤害。
- 不要短接电池端子或将对电池充电、解体、加热或焚烧。不要使电池受到强烈冲击，诸如此类的情况都可能导致电池漏电、绝缘击穿、发热或爆炸。请将掉落在地上或受到过度冲击的电池拿掉。使用受过冲击的电池可能导致漏电。
- UL 标准要求，电池的更换只能由有经验的技术人员操作，不具有资格的人员不得更换电池。
- 对于 CJ 系列 PLC、电源单元、CPU 单元、I/O 单元和 CPU 总线单元顶部的滑扣必须完全卡入扣住 （直到它的卡入部位） 。否则，这些单元将不能正常工作。
- 对于 CJ 系列的 PLC，务必将端盖安装到 PLU 最右的单元上。没有安装端盖的 PLC 将不能正常运行。

- 如果不合适的数据连接表和参数被设置，可能导致不可预料的运行。即使已经设置了合适的数据连接表和参数，在启动或停止数据连接前仍需确认控制系统不会受到有害的影响。
- 当路由表从编程装置传送到 CPU 单元， CPU 总线单元将被重新启动。重新启动这些单元需要读新路由表并使它有效，在允许 CPU 总线单元复位前，确认系统不会受到有害的影响。

6 符合 EC 规程

6-1 适用规程

- EMC 规程
- 低压规程

6-2 规则

EMC 规程

OMRON 公司的所有装置都附合 EC 规程，也符合有关 EMC 标准，所以它们可以很方便地装入其它装置和所有的机械中。为了符合 EMC 标准，对各实际产品都作了检验 （参见下注）。然而各产品是否符合用户所用的系统要求，必须由用户来检验。

符合 EC 规程的 OMRON 装置的 EMC 相关性能，随装有 RMRON 装置的设备的配置、接线和其它条件或控制电板的不同而不同。因此，为了确认各装置和整个机械符合 EMC 标准，用户必须作最终检查。

注 适用 EMC （电磁兼容）标准如下：

EMS （电磁敏感度）：	EN61131-2 （CS 系列） / EN61000-6-2 （CJ 系列）
EMI （电磁干扰）：	EN50081-2 （辐射发射：10m 调整率）

低压规程

始终保证装置工作在交流 50V ～ 1000V 的电压范围，直流 75V ～ 1500V 电压范围内，满足 PLC 所要求的安全标准 (EN61131-2)。

6-3 符合 EC 规程

CS/CJ 系列 PLC 符合 EC 规程。为了保证使用 CS/CJ 系列 PLC 的机械或装置符合 EC 规程， PLC 必须按下列要求安装：

1,2,3...

1. CS/CJ 系列 PLC 必须安装在控制面板内。
2. 通信电源和 I/O 电源用的直流电源必须采用加强绝缘或双重绝缘。
3. CS/CJ 系列 PLC 符合 EC 规程，也符合一般发射标准 (EN50081-2)。辐射发射特性（10m 调整率）可能随所用的控制面板的配置，与控制面板的连接的其它设备、接线和其它条件的不同而不同。因此，用户必须确认整个机械或设备符合 EC 规程。

6-4 继电器输出噪声降低法

CS/CJ 系列 PLC 符合 EMC 规程的一般发射标准 (EN50081-2)。然而，由继电器输出切换产生的噪声可能不满足这些标准。在这种标准下，负载则必须连接一个抗噪声滤波器或在 PLC 外部采用合适的预防措施。

为满足标准而采取的预防措施随负载侧的装置，接线，各机械的配置等的不同而不同。下例是降低产生的噪音用的预防措施实例。

预防措施

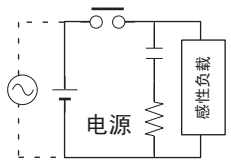
（详情参见 EN50081-2）

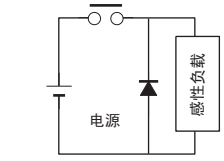
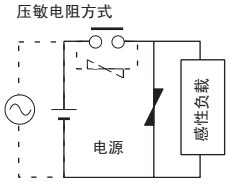
如果包括 PLC 在内的整个系统，其负载开关切换频率小于每分钟 5 次，则不需要采取预防措施。

如果包括 PLC 在内的整个系统，其负载开关切换频率大于每分钟 5 次，则需要采取预防措施。

预防措施实例

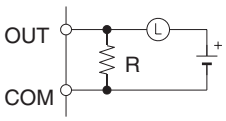
当切换感性负载时，请将浪涌保护器、二极管等与负载或触点并联，连接如下所示。

电路	电流		特性	需要元件
	AC	DC		
CR方法 	是	是	如果负载是一个继电器或螺旋管，则在电路断开的瞬间和负载重新接入的瞬间会有一时间迟滞。 如果电源电压是 24V 或 48V，则浪涌保护器与负载并联接入。如果电源电压是 100V ～ 200V，则在触点之间接入浪涌保护器。	每 1A 的触点电流，电容器的容量必须在 1 ～ 0.5 μ F 间，而每 1V 的触点电压，电阻器的电阻必须在 0.5 ～ 1 Ω 间。然而这些值可能随负载和继电器的特性不同而不同。请根据经验决定这些值，并考虑触点分断时的电容抑制火花放电和电路再次闭合时电阻对流入负载的电流限制。 电容器绝缘强度必须是 200~300V。如果电路是 AC 电路，则应使用无极性的电容。

电路	电流		特性	需要元件
	AC	DC		
二极管方法 	否	是	与负载并联的二极管使线圈积累的能量变为电流，然而流入线圈，因此由于电感负载的电阻，电流会转换成焦耳热。 由这种方法引起的，在电路断开瞬间和负载重新接入瞬间之间，这个时间迟滞比由 CR 方法引起的更长。	二极管的反向耐压至少必须是电路电压值的 10 倍。二极管的正向电流必须等于或大于负载电流。 如果浪涌保护电路是应用于低压电路的电子回路，则二极管的反向绝缘强度可以是大于电源电压的 2 ～ 3 倍。
压敏电阻方式 	是	是	压敏电阻方式是使用恒压特性的压敏电阻来防止触点之间承受高压电。在电路断开的瞬间和负载重新接入瞬间之间有时间迟滞。 如果电源电压是 24V 或 48V，则压敏电阻器与负载并联。如果电源电压是 100 ～ 200V，则压敏电阻与触点并联。	---

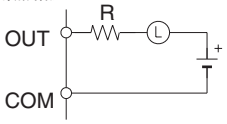
当切换一个具有浪涌电流负载如白炽灯时，抑制浪涌电流方法如下所示。

预防措施1



提供-约为额定值3/1
的暗电流通过白炽灯

预防措施2



提供一限流电阻

第 1 章
简介

本章介绍一般指令特性及指令执行期间可能出现的错误。

1-1 一般指令特性

1-1-1 程序容量

程序容量是指 CPU 单元中用户程序区的大小，并用程序步数来表示。每个 CS/CJ 系列指令在用户程序区所需步数从 1 步～ 7 步不等，取决于指令和所需的操作数。

CS 系列

下表显示了每种 CS 系列 CPU 单元最大的编程步数。

- CS1-H CPU 单元

型号	程序容量	I/O 点数
CS1H-CPU67H	250K 步	5,120
CS1H-CPU66H	120K 步	
CS1H-CPU65H	60K 步	
CS1H-CPU64H	30K 步	
CS1H-CPU63H	20K 步	
CS1G-CPU45H	60K 步	
CS1G-CPU44H	30K 步	1,280
CS1G-CPU43H	20K 步	960
CS1G-CPU42H	10K 步	

- CS1 CPU 单元

型号	程序容量	I/O 点数
CS1H-CPU67-E	250K 步	5,120
CS1H-CPU66-E	120K 步	
CS1H-CPU65-E	60K 步	
CS1H-CPU64-E	30K 步	
CS1H-CPU63-E	20K 步	
CS1G-CPU45-E	60K 步	
CS1G-CPU44-E	30K 步	1,280
CS1G-CPU43-E	20K 步	960
CS1G-CPU42-E	10K 步	

- CS1D CPU 单元

型号	程序容量	I/O 点数
CS1D-CPU67H	250K 步	5,120
CS1D-CPU65H	60K 步	

CJ 系列

下表显示了每种 CJ 系列 CPU 单元最大的编程步数。

- CJ1-H CPU 单元

型号	程序容量	I/O 点数
CJ1H-CPU66H	120K 步	2,560
CJ1H-CPU65H	60K 步	
CJ1G-CPU45H	60K 步	1,280
CJ1G-CPU44H	30K 步	

型号	程序容量	I/O 点数
CJ1G-CPU43H	20K 步	960
CJ1G-CPU42H	10K 步	

- CJ1 CPU 单元

型号	程序容量	I/O 点数
CJ1G-CPU45	60K 步	1,280
CJ1G-CPU44	30K 步	

- CJ1M CPU 单元

型号	程序容量	I/O 点数
CJ1M-CPU23	20K 步	640
CJ1M-CPU22	10K 步	320
CJ1M-CPU13	20K 步	640
CJ1M-CPU12	10K 步	320

注 CS/CJ 系列的 PC 用步计量程序容量，而以前的 OMRON PC（如 C 系列和 CV 系列）则用字来计量程序容量。一般来说，一个步相当于一个字。然而对于 CS/CJ 系列某些指令而言，指令所需的内存量并不相同。如果另一种 PC 的用户程序容量转换到 CS/CJ 系列 PC 中，那么假定一个字为一步就会产生不准确。参照*第 4 章指令执行时间和步*的结尾处从有关以前 OMRON PC 转换程序容量的信息。

在一个程序中步数和指令数是不相同的。例如：LD 和 OUT 每次需要一步，但 MOV(021) 则需要 3 步。另外一些指令甚至需要 7 步。对于双字长操作数的指令，它所需的步数也会增加。例如：MOVL(498) 通常要 3 步，但如果一个常数被指定为源操作数 (S)，那么它需要 4 步。参照*第 4 章指令执行时间和步*中有关每个指令所需的步数。

1-1-2 微分指令

CS/CJ 系列 PC 中的大部分指令都有非微分和上升沿微分变化二种形式，而且有些指令还有下降沿微分变化。

- 非微分指令在每个扫描周期执行。
- 上升沿微分指令仅当执行条件从 OFF → ON 变化时，执行 1 次。

• 下降沿微分指令仅当执行条件 ON → OFF 变化时执行一次。

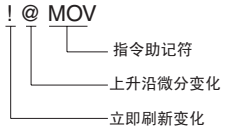
变化	指令类型	操作	格式	举例
非微分型	输出指令 （指令需要一执行条件）	当执行条件为 ON 时，指令在每个循环执行。	每个循环执行输出指令	MOV
	输入指令 （指令用作执行条件）	位处理（如读、比较或测试）在每个循环执行。当结果为 ON 时，执行条件为真。	每个循环执行输入指令	
上升沿微分（带前缀 @）	输出指令	当执行条件从 OFF → ON 变化时，该指令只执行一次。	上升沿微分指令执行一次	@MOV 在CIO 000102每次从OFF→ON跳变时，MOV（021）执行一次。
	输入指令 （指令用作执行条件）	位处理（如读、比较或测试）在每个循环执行。当结果从 OFF → ON 变化时，执行条件在一个循环内为真。	上升沿微分输入指令	在CIO 000103每次从OFF→ON跳变时，ON执行条件建立一个循环。
下降沿微分（带前缀 %）	输出指令	当执行条件从 ON → OFF 变化时，该指令只执行一次。	%下降沿微分指令执行一次	0001 02 %SET 在CIO 000102每次从ON→OFF跳变时，SET执行一次
	输入指令 （指令用作执行条件）	位处理（如读、比较或测试）在每个循环执行。当结果从 ON → OFF 变化时，执行条件在一个循环内为真。	下降沿微分输入指令	0001 03 在CIO 000103每次从ON→OFF跳变时，ON执行条件建立一个循环

注 下降沿微分 (%) 仅对 LD，AND，OR 和 RSET 指令有效。为建立其他指令的下降沿微分变化，可用 DIFD(014) 或 DOWN(522) 控制工作位来控制指令的执行。

1-1-3 指令变化

在变化前缀 (@， %， !) 加到一个指令前可形成一个微分指令或立即刷新。

变化		前缀	操作
微分	上升沿微分	@	建立一个上升沿微分指令。
	下降沿微分	%	建立一个下降沿微分指令。
立即刷新		!	当执行该指令时，在 I/O 区或特殊 I/O 单元区中的指令操作码数据将被刷新。



1-1-4 指令位置和执行条件

下表显示了指令能编程的位置。下表也显示了当一个指令需要一个执行条件和不需要一个执行条件的情况。指定指令的细节参阅*第 2 章指令一览*。

指令类型		位置	执行条件	格式	例
输入	开始逻辑条件的指令	在左母线或一个指令块起始处	不需要		LD， LD TST 和输入比较指令，如 LD>
	连接指令	在一个开始指令和输出指令之间	需要		AND， OR， AND TST，输入比较指令，如：AND> ， UP，DOWN， NOT
输出		在右母线	需要		大多数指令 （如 OUT 和 MOV）
			不需要		如 END， JME， FOR， ILC 指令

除了这些指令， CS/CJ 系列 PC 还配有块程序指令。细节参阅块程序指令的描述。

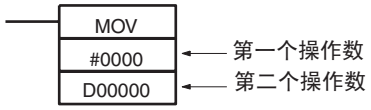
注 如果一个执行条件没有位于一个需要执行条件的指令之前，那么当从外部设备检查程序时，会出现一个程序错误。

1-1-5 操作数中输入数据

操作数是在指令执行时，用 I/O 内存地址或所用的常数预先设置的参数。有三种基本形式的操作数：源操作数，目的操作数和号。

操作数		使用代码	内容	
源	包含数据或数据本身的地址	S	源操作数	除控制数据之外的源数据
		C	控制数据	用单个位或多个位控制指令执行的控制数据
目的	存储数据的地址	D	---	
号	包含一个号如一个跳转号或子程序号	N	---	

注 也可以通过操作数在指令中所处的位置把它们归类 （第一操作数，第二操作数……）。操作数使用的代码根据指定函数的操作数而各不相同。



指定位地址

描述	例	指令举例
为指定一个位地址，直接指定字地址和位地址。 □□□□ □□ 位号 字地址 注 定时器/计数器完成标志或 任务标志不用“字地址+位号”格式	0001 02 位02 字CIO 0001	0001 02

指定字地址

描述	例	指令举例
为指定一个字地址，直接指定字地址。 □□□□ 字地址	0003 字CIO 0003 D00200 字D00200	MOV 0003 D00200

用二进制方式指定 DM/EM 间接地址

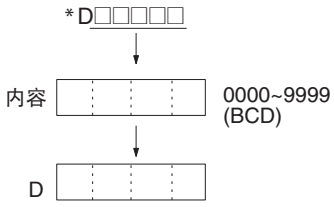
描述	例	指令举例
当在一个DM或EM地址前安放一个@前缀时，该字内容指定另一个作为操作数。内容可以是000～7FFF（0～32767），对应于DM或EM区中所要求的字地址。@ D□□□□ @D□□□□□ 内容 00000～32767 (0000～7FFF) D	---	---
当 @D@@@@@ 的内容在 0000 和 7FFF(00000~32767) 之间时，指定了 D00000 和 D32767 之间的相应字。	@D00300 0 1 0 0 十进制： 256 指定D00256 加@前缀	MOV #0001 @D00300
当 @D@@@@@ 的内容在 8000 和 FFFF(32768~65535) 之间时，指定了当前 EM 区的 E0_00000 和 E0_32767 之间的相应字。	@D00300 8 0 0 1 十进制数：32,769 指定E0_00001	---

描述	例	指令举例
当 @En@_@@@@@ 的内容在 0000 和 7FFF(00000~32767) 之间时，指定了 En@_00000 和 En@_32767 之间的相应字。	@E1_00200 0 1 0 1 十进制数：257 ↓ 指定E1_00257	MOV #0001 @E1_00200
当 @En@_@@@@@ 的内容在 8000 和 FFFF(32768~65535) 之间时，指定了 E(@+1)-00000 和 E(@+1)-32767 （在下一个 EM 区）之间的相应字。	@E1_00200 8 0 0 2 十进制数：32770 ↓ 指定E2_00002	

注 当 PC 设置中选择二进制方式时， DM 区和当前 EM 区地址 （0~C 区）当作连续存储地址处理。如果一个间接地址的 DM 字所含的值大于 32767，那么 EM 零区号中的一个字将被指定。例如：当间接寻址的 DM 字为 8000 的十六进值（十进制为 32768），那么将指定零区号 E00000。

如果一个间接寻址的 EM 字所含值大于 32767，那么在下一个 EM 区的一个字将被指定。例如，当间接寻址区 2 区号 EM 字为 8000 的十六进制值 （十进制为 32768），那么 E3_00000 将被指定。

用 BCD 方式指定间接 DM/EM 地址

方法	描述	例	指令举例
间接 DM/EM 寻址（BCD 码方式）	当在一个DM或EM地址前放置一个前缀*时，该字指定另一个字的BCD内容作为操作数。内容为0000~9999，对应DM或EM区中所要的字地址。 	*D00200 0 1 0 0 ↓ 指定D00100 加前缀*	MOV #0001 *D00200

寻址索引寄存器

方法	描述		例	指令举例
直接寻址索引寄存器	MOVR(560) 将一个字或位的内部 I/O 内存地址传送到一个索引寄存器中 (IR0~IR15)。 （MOVRW(561) 将一个定时器或计数器的 PV 内部 I/O 内存地址传送到一个索引寄存器）。		IR0 IR2	MOVR 0010 IR0 把 CIO 0010 的内部 I/O 内存地址储存在 IR0 中。 MOVR 000102 IR2 把 CIO 00102 内部 I/O 内存地址储存在 IR2 中。
用索引寄存器间接寻址	基本操作（无偏移）	把含在 IR@ 中的 I/O 内存地址中的字或位作为操作数用。在索引寄存器前放一逗号以表示间接寻址。（根据指令或操作数可决定所指定的位 / 字）。	,IR0 ,IR1	LD ,IR0 取包含在 IR0 中的 I/O 内存地址上位的状态。 MOV #0001, IR1 把 #0001 传送到包含在 IR1 中的 I/O 内存地址上的字中。
	常数偏移	把偏移值 (-2,048~+2,047) 加到包含在 IR@ 中的 I/O 内存地址上，并且把结果地址当作操作数用。（指令执行时偏移转换为二进制）	+5 ,IR0 +31 ,IR1	LD +5 ,IR0 把 5 加到包含在 IR0 中的 I/O 内存地址中，并且取该地址的位状态。 MOV #0001 +31 ,IR1 把 31 加到包含在 IR1 中的 I/O 内存地址上，并且把 #0001 传送到该地址的字中。
	DR 偏移	把数据寄存器中的带符号二进制内容加到包含在 IR@ 中的 I/O 内存地址上，并且把结果地址当操作数用。	DR0 ,IR0 DR0 ,IR1	LD DR0 ,IR0 把 DR0 内容加到包含在 IR0 中的 I/O 内存地址上，并且取该地址位的状态。 MOV #0001 DR0 ,IR1 把 DR0 内容加到包含在 IR1 中的 I/O 内存地址上，并且把 #0001 传送到该地址的字中。
	自动递增	在 IR@ 读 I/O 内存地址后，索引寄存器的内容增加 1 或 2。 递增 1：,R@+ 递增 2：,IR@++ 注 当指令执行时即使出现错误，并且错误标志变 ON，索引寄存器将递增。	,IR0 + + ,IR1 +	LD ,IR0 + + 取包含在 IR0 中的 I/O 内存地址上位的状态，并且使寄存器递增 2。 MOV #0001 ,IR1 + 把 #0001 传送到包含在 IR1 中的 I/O 内存地址上的字中，并且使寄存器递增 1。
	自动递减	IR@ 内容减 1 或 2，然后把寄存器中的 I/O 内存地址当操作数用。 递减 1：,– IR@ 递减 2：,– –IR@ 注 当指令执行时即使出现错误，并且错误标志变 ON，索引寄存器将递减。	, – – IR0 , – IR1	LD , – – IR0 使 IR0 的内容减 2，然后取 I/O 内存地址上位的状态。 MOV #0001 , – IR1 使 IR1 的内容减 1，然后把 #0001 传送到该 I/O 内存地址的字中。

注 确保索引寄存器的内容指示有效的 I/O 内存地址。

指定的常数

方式	使用的操作数	数据格式	代码	范围	例
常数（16 位数据）	所有二进制数据和一定范围之内的二进制数据	不带符号的二进制	#	#0000 ～ #FFFF	---
		带符号的十进制	±	–32,768 ～ +32,767	---
		不带符号的十进制	&	&0 ～ &66,535	---
	所有 BCD 数据和一定范围之内的 BCD 数据	BCD	#	#0000 ～ #9999	---
常数（32 位数据）	所有二进制数据和一定范围之内的二进制数据	不带符号的二进制	#	#0000 0000 ～ #FFFF FFFF	---
		带符号的十进制	+ –	–2,147,483,648 ～ +2,147,483,647	---
		不带符号的十进制	&	&0 ～ &4,294,967,295	---
	所有 BCD 数据和一定范围之内的 BCD 数据	BCD	#	#0000 0000 ～ #9999 9999	---

指定文本字符串

方式	描述	代码	例	指令举例
文本字符串	文本以 ASCII 码方式存放（1 字节 / 字符，除了特殊字符）。从范围中最低的低字节开始。 如果字符为奇数个，那么就将 00(NULL) 存放在范围内最后一个字中的高字节中。 如果字符为偶数个，那么就将 0000(NULL) 存放在范围内的最后字的后面。		"ABCDE" "A" "B" "C" "D" "E" NUL = 41 42 43 44 45 00 "ABCD" "A" "B" "C" "D" NUL NUL = 41 42 43 44 00 00	MOV$ D00100 D00200 D00100 41 42 D00101 43 44 D00102 45 00 ↓ D00200 41 42 D00201 43 44 D00202 45 00

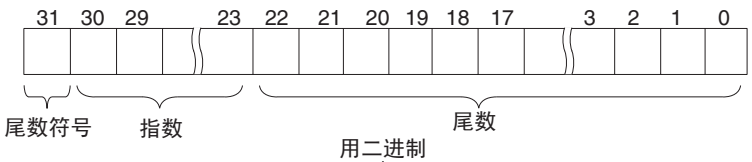
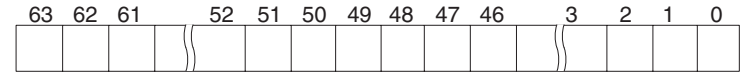
下表为能用 ASCII 码表示的字符。

右位 \ 左位	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0			SP	0	@	P	'	p				―	タ	ミ		
1			!	1	A	Q	a	q			。	ア	チ	ム		
2			”	2	B	R	b	r			「	イ	ツ	メ		
3			#	3	C	S	c	s			」	ウ	テ	モ		
4			$	4	D	T	d	t			、	エ	ト	ヤ		
5			%	5	E	U	e	u			・	オ	ナ	ユ		
6			&	6	F	V	f	v			ヲ	カ	ニ	ヨ		
7			’	7	G	W	g	w			ァ	キ	ヌ	ラ		
8			(	8	H	X	h	x			ィ	ク	ネ	リ		
9			)	9	I	Y	i	y			ゥ	ケ	ノ	ル		
A			*	:	J	Z	j	z			ェ	コ	ハ	レ		
B			+	;	K	[	k	{			ォ	サ	ヒ	ロ		
C			,	<	L	¥	l	|			ャ	シ	フ	ワ		
D			―	=	M	]	m	}			ュ	ス	ヘ	ン		
E			.	>	N	^	n	~			ョ	セ	ホ	゛		
F			/	?	O	_	o				ッ	ソ	マ	°		

1-1-6 数据格式

下表显示了可用在 CS/CJ 系列 PC 中的数据格式。

名称	格式	十进制范围	十六进制数范围
不带符号的二进制数据	15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 二进制 2^{15} 2^{14} 2^{13} 2^{12} 2^{11} 2^{10} 2^{9} 2^{8} 2^{7} 2^{6} 2^{5} 2^{4} 2^{3} 2^{2} 2^{1} 2^{0} 十进制 32768 16384 8192 4096 2048 1024 512 256 128 64 32 16 8 4 2 1 十六进制 2^{3} 2^{2} 2^{1} 2^{0} 2^{3} 2^{2} 2^{1} 2^{0} 2^{3} 2^{2} 2^{1} 2^{0} 2^{3} 2^{2} 2^{1} 2^{0}	0 ～ 65,535	0000 ～ FFFF
带符号的二进制数据	15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 二进制 2^{15} 2^{14} 2^{13} 2^{12} 2^{11} 2^{10} 2^{9} 2^{8} 2^{7} 2^{6} 2^{5} 2^{4} 2^{3} 2^{2} 2^{1} 2^{0} 十进制 32768 16384 8192 4096 2048 1024 512 256 128 64 32 16 8 4 2 1 十六进制 2^{3} 2^{2} 2^{1} 2^{0} 2^{3} 2^{2} 2^{1} 2^{0} 2^{3} 2^{2} 2^{1} 2^{0} 2^{3} 2^{2} 2^{1} 2^{0} 符号位 0:正数 1:负数	–32,768 ～ +32,767	8000 ～ 7FFF
BCD 数据	15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 BCD 2^{3} 2^{2} 2^{1} 2^{0} 2^{3} 2^{2} 2^{1} 2^{0} 2^{3} 2^{2} 2^{1} 2^{0} 2^{3} 2^{2} 2^{1} 2^{0} 十进制 0～9 0～9 0～9 0～9	0 ～ 9,999	0000 ～ 9999

名称	格式	十进制范围	十六进制数范围
浮点数据	 尾数符号 指数 尾数 用二进制 实数=（−1）符号 x 1.[尾数] x 2指数 ● 符号（31位） 当符号位为1时是负数，当符号位为0时是正数。 ● 尾数 尾数包括从位00～位22的23位，并用二进制形式定义这部分数为十进制小数点后的数1.□□□.....。 ● 指数 指数包括从位23～位30的8位，并用二进制定义n+127用于数的2^n部分 注 这是标准IEEE754单精度格式，并且它仅用在浮点转换和运算指令中。在用CX-Programmer（不能用手持编程器）进行设定和监控I/O内存数据时，可使用这数据格式。用户不必了解这个数据格式的细节，只要知道每个数占用二个字。	---	---
双精度浮点数据	 尾数符号 指数 尾数 用二进制 实数=（−1）符号 x 1.[尾数] x 2指数 符号（63位） 当符号位为1时是负数，当符号位为0时是正数。 尾数 尾数包括从位00～位51的52位，并用二进制形式定义这部分数为十进制小数点后的数1.□□□.....。 指数 指数包括从位52～位62的11位，并用二进制定义1023+n用于数的2^n部分 注 这是标准IEEE754双精度格式，并且它仅用在浮点转换和运算指令中。在用CX-Programmer（不能用手持编程器）进行设定和监控I/O内存数据时，可使用这数据格式。用户不必了解这个数据格式的细节，只要知道每个数占用四个字。	---	---

带符号的二进制数

带符号的二进制负数可用十六进制的绝对值的补码来表示。如十进制值-12,345 的绝对值等于 3039 十六进制。它的补码为 10000-3039 或 CFC7。

为把一个带符号的二进制负数转换成十进制数，将这个数的补码 (1000-CFC7=3039) 转换成十进制数（3039 十六进制 =12,345 十进制），然后加一个负号 (-12345)。

1-2 指令执行检查

1-2-1 执行指令时所出现的错误

当从外部设备输入一条指令或由外部设备（除了编程器）完成一个程序检查后，检查一条指令操作数或所处位置，但这些不是最终的检查，指令执行时，会出现 4 种错误。

指令执行错误（ER 标志 ON）

通常，指令执行错误是非致命错误，但是 PC 设置能把它设置为致命错误。如果设定已存在，那么当出现指令执行错误时，指令执行错误标志 (A29508) 将变为 ON，并且程序执行将停止。

访问错误（AER 标志 ON）

通常，访问错误是非致命错误，但是 PC 设置能把它设置为致命错误。如果设定已存在，那么当出现访问错误时，非法访问错误标志 (A29510) 和间接 DM/EM BCD 错误标志 (A29509) 将变为 ON，并且程序执行停止。

非法指令错误

当出现此错误时，非法指令错误标志 (A29514) 变为 ON，并且程序执行停止。

UM（用户程序内存）溢出错误

当出现此错误时，UM 溢出错误标志 (A29515) 将变为 ON，并且程序执行停止。

1-2-2 致命错误（程序错误）

当出现以下程序错误之一时，程序执行将停止。当已经出现程序错误，程序执行被停止期间，正在执行任务的编号被写入 A294，并且程序地址被写入 A298 和 A299。

使用这些字的内容来确定程序错误位置，并按要求改正。

地址	描述
A294	当由于程序错误，程序执行停止时，当前任务编号被写到这个字中。 循环任务有 0000~001F（循环任务：0~31） 中断任务有 8000~80FF 任务号（中断任务：0~255）
A298 和 A299	当由于程序错误，程序执行停止时，当前程序地址被写到这些字中。 A299 包含程序地址左四位， A298 包含程序地址右四位。

错误标志或访问错误标志变 ON 时的所有错误被当作程序错误。以下表格列出了程序错误。当其中之一错误出现时， PC 设置能设置为中止程序执行。

错误类型	描述	相应标志
无结束指令	在程序中无 END(001) 指令	无 END 错误标志（A29511）
任务错误	有三种原因能引起任务错误： 1) 没有一个可执行的循环任务。 2) 没有一个可分配给任务的程序。 3) 中断已产生，但是不存在对应中断的任务。	任务错误标志（A29512）
指令执行错误 *	CPU 试图执行一个指令 , 但是指令操作数所提供的数据不正确。 * 如果 PC 设置被设置为处理指令错误作为致命错误 （程序错误），那么指令执行错误标志 (A29508) 将变为 ON，并且程序执行停止。	错误标志 (ER)，指令执行错误标志 (A29508)
访问错误 *	有五个原因可能引起访问错误： 1) 对参数区的读 / 写。 2) 对未安装的内存进行写操作。 3) 对一个是 EM 文件内存的 EM 区进行读 / 写操作。 4) 对一个只读区进行写操作。 5) 虽然PC设置为BCD码间接寻址，但是DM/EM字的内容不是 BCD 码。 * 如果 PC 设置被设置为处理指令错误作为致命错误 （程序错误），非法访问错误标志 (A29510) 将变为 ON，并且程序执行停止。	访问错误标志 (AER)，非法访问错误标志 (A29510)
BCD 间接 DM/EM 错误 *	虽然 PC 设置为 BCD 码间接寻址，但是 DM/EM 字的内容不是 BCD 码。 * 如果 PC 设置被设置为处理指令错误作为致命错误 （程序错误），那么指令执行错误标志 (A29509) 将变为 ON，并且程序执行停止。	访问错误标志 (AER)，BCD 间接 DM/EM 错误标志 (A29509)
微分溢出错误	在线编程时反复插入和删除微分指令 （超出 31072 次）。	微分溢出错误 (A29513)
UM 溢出错误	超出 UM （用户程序内存）中的末地址。	UM 溢出错误标志 (A29515)
非法指令错误	程序包含一个不能被执行的指令。	非法指令错误标志 (A29514)

第 2 章
指令摘要

本章提供用于 CS/CJ 系列 PLC 中的指令摘要。

2-1 按指令功能分类指令

CS/CJ 系列按功能分类如下表。（指令按*第 3 章 指令*中的功能排序）

* 以一个星号标记的指令或者指令集仅由 CS1-H，CJ1-H，CJ1M 和 CS1D CPU 单元支持。

** 以两个星号标记的指令或者指令集仅由 CJ1M CPU 单元支持。

*** 以三个星号标记的指令或者指令集仅由 CS1D CPU 单元支持。

分类	细类	助记符	指令	助记符	指令	助记符	指令
基本指令	输入	LD	装载	LD NOT	装载非	AND	非
		AND NOT	与非	OR	或	OR NOT	或非
		AND LD	逻辑块与	OR LD	逻辑块或	---	---
	输出	OUT	输出	OUT NOT	输出非	---	---
顺序输入指令	---	NOT	非	UP	条件 ON	DOWN	条件 OFF
	位测试	LD TST	装载位测试	LD TSTN	装载位非测试	AND TST	位测试与
		AND TSTN	位测试与非	OR TST	位测试或	OR TSTN	位测试或非
顺序输出指令	---	KEEP	保持	DIFU	上升沿微分	DIFD	下降沿微分
		OUTB*	单个位输出	---	---	---	---
	置位 / 复位	SET	置位	RSET	复位	SETA	多个位置位
		RSTA	多个位复位	SETB*	单个位置位	RSTB*	单个位复位
顺序控制指令	---	END	结束	NOP	空操作	---	---
	联锁	IL	联锁	ILC	联锁解除	---	---
	跳转	JMP	跳转	JME	跳转结束	CJP	条件跳转
		CJPN	条件跳转	JMP0	多路跳转	JME0	多路跳转结束
	重复	FOR	FOR-NEXT 循环	BREAK	循环中断	NEXT	FOR-NEXT 循环

分类	细类		助记符	指令	助记符	指令	助记符	指令
定时器和计数器指令	BCD	定时器（带定时器符号）	TIM	定时器	TIMH	高速定时器	TMHH	1ms 定时器
			TTIM	累加定时器	---	---	---	---
		定时器（不带定时器号）	TIML	长定时器	MTIM	多输出定时器	---	---
		计数器（带计数器号）	CNT	计数器	CNTR	可逆计数器	CNR	定时器 / 计数器复位
	二进制 *	定时器（带定时器号）	TIMX	定时器	TIMHX	高速定时器	TMHHX	1ms 定时器
			TTIMX	累加定时器	---	---	---	---
		定时器（不带定时器号）	TIMLX	长定时器	MTIMX	多输出定时器	---	---
		计数器（带计数器号）	CNTX	计数器	CNTRX	可逆计数器	CNRX	定时器 / 计数器复位
比较指令	符号比较		LD, AND, OR + =, <>, <, <=, >, >=	符号比较 （无符号）	LD, AND, OR + =, <>, <, <=, >, >= + L	符号比较 （双字，无符号）	LD, AND, OR + =, <>, <, <=, >, >= +S	符号比较 （有符号）
			LD, AND, OR + =, <>, <, <=, >, >= + SL	符号比较 （双字，有符号）	---	---	---	---
	数据比较（条件标志）		CMP	不带符号比较	CMPL	不带符号双字比较	CPS	带符号二进制比较
			CPSL	带符号双字二进制比较	ZCP*	区域比较	ZCPL*	双字区域比较
	表格比较		MCMP	多字比较	TCMP	表格比较	BCMP	不带符号块比较
			BCMP2**	扩展块比较	---	---	---	---
数据传送指令	单 / 双字		MOV	传送	MOVL	双字传送	MVN	取反传送
			MVNL	双字取反传送	---	---	---	---
	位 / 数字		MOVB	位传送	MOVD	数字传送	---	---
	数据交换		XCHG	数据交换	XCGL	双字数据交换	---	---
	块 / 位传送		XFRB	多位传送	XFER	块传送	BSET	块设定
	分配 / 收集		DIST	单字节数据分配	COLL	数据收集	---	---
	索引寄存器		MOVR	传送至寄存器	MOVRW	定时器 / 计数器 PV 传送至寄存器	---	---

分类	细类	助记符	指令	助记符	指令	助记符	指令
数据移位指令	1 位移动	SFT	移位寄存器	SFTR	可逆移位寄存器	ASLL	双字左移
		ASL	算术左移	ASR	算术右移	ASRL	双字右移
	0000 十六进制异步	ASFT	异步移位寄存器	---	---	---	---
	字移位	WSFT	字移位	---	---	---	---
	1 位循环移位	ROL	循环左移	ROLL	双字循环右移	RLNC	无进位循环左移
		RLNL	无进位双字循环左移	ROR	循环右移	RORL	双字循环右移
		RRNC	无进位循环右移	RRNL	无进位双字循环右移	---	---
	1 数字移位	SLD	1 数字左移	SRD	1 数字右移	---	---
	N 位数据移位	NSFL	n 位数据左移	NSFR	n 位数据右移	---	---
	N 位移位	NASL	n 位左移	NSLL	双字 n 位左移	NASR	n 位右移
		NSRL	双字 n 位右移	---	---	---	---
递增 / 递减指令	BCD	++B	BCD 递增	++BL	双字 BCD 递增	– –B	BCD 递减
		– –BL	双字 BCD 递减	---	---	---	---
	二进制	++	二进制递增	++L	双字二进制递减	– –	二进制递减
		– –L	双字二进制递减	---	---	---	---

分类	细类	助记符	指令	助记符	指令	助记符	指令
四则运算指令	二进制加	+	无进位带符号二进制加	+L	无进位带符号双字二进制加	+C	有进位带符号二进制加
		+CL	有进位带符号双字二进制加	---	---	---	---
	BCD 加	+B	无进位 BCD 加	+BL	无进位双字 BCD 加	+BC	有进位 BCD 加
		+BCL	有进位双字 BCD 加	---	---	---	---
	二进制减	–	无进位带符号二进制减	–L	无进位带符号双字二进制减	–C	有进位带符号二进制减
		–CL	有进位带符号双字二进制减	---	---	---	---
	BCD 减	–B	无进位 BCD 减	–BL	无进位双字 BCD 减	–BC	带进位 BCD 减
		–BCL	有进位双字 BCD 减	---	---	---	---
	二进制乘	*	带符号二进制乘	*L	带符号双字二进制乘	*U	不带符号二进制乘
		*UL	不带符号双字二进制乘	---	---	---	---
	BCD 乘	*B	BCD 乘	*BL	双字 BCD 乘	---	---
	二进制除	/	带符号二进制除	/L	带符号双字二进制除	/U	不带符号二进制除
		/UL	不带符号双字二进制除	---	---	---	---
	BCD 除	/B	BCD 除	/BL	双字 BCD 除	---	---

分类	细类	助记符	指令	助记符	指令	助记符	指令
转换指令	BCD/ 二进制转换	BIN	BCD →二进制	BINL	双字 BCD → 双字二进制	BCD	二进制→ BCD
		BCDL	双字二进制→ 双字 BCD	NEG	二进制求补	NEGL	双字二进制求补
		SIGN	16 位→ 32 位带符号二进制	---	---	---	---
	译码 / 编码	MLPX	数据译码	DMPX	数据编码	---	---
	ASCII/ 十六进制转换	ASC	ASCII 转换	HEX	ASCII →十六进制	---	---
	行 / 列转换	LINE	列→行	COLM	行→列	---	---
	带符号二进制 /BCD 转换	BINS	带符号 BCD →二进制	BISL	带符号双字 BCD →二进制	BCDS	带符号二进制→ BCD
		BDSL	带符号双字二进制→ BCD	---	---	---	---
逻辑指令	逻辑与 / 或	ANDW	逻辑与	ANDL	双字逻辑与	ORW	逻辑或
		ORWL	双字逻辑或	XORW	异或	XORL	双字异或
		XNRW	异或非	XNRL	双字异或非	---	---
	求补	COM	求补	COML	双字求补	---	---
特殊算术指令	---	ROTB	二进制平方根	ROOT	BCD 平方根	APR	算术处理
		FDIV	浮点数除	BCNT	位计数器	---	---
浮点数运算指令	浮点 / 二进制转换	FIX	浮点→ 16 位	FIXL	浮点→ 32 位	FLT	16 位→浮点
		FLTL	32 位→浮点	---	---	---	---
	浮点基本运算	+F	浮点数加	–F	浮点数减	/F	浮点数除
		*F	浮点数乘	---	---	---	---
	浮点函数	RAD	度→弧度	DEG	弧度→度	SIN	正弦
		COS	余弦	TAN	正切	ASIN	反正弦
		ACOS	反余弦	ATAN	反正切	---	---
	浮点数学	SQRT	平方根	EXP	指数	LOG	对数
		PWR	指数幂	---	---	---	---
	符号比较与转换 *	LD, AND, OR + =, <>, <, <=, >, >= + F	符号比较（单精度浮点数）	FSTR*	浮点数→ ASCII	FVAL*	ASCII →浮点数

分类	细类	助记符	指令	助记符	指令	助记符	指令
双精度浮点数指令 *	浮点 / 二进制转换	FIXD	双精度浮点数 → 32 位	FIXLD	双精度浮点数 → 32 位	DBL	16 位→双精度浮点数
		DBLL	32 位→双精度浮点数	---	---	---	---
	浮点基本运算	+D	双精度浮点数加	–D	双精度浮点数减	/D	双精度浮点数除
		*D	双精度浮点数乘	---	---	---	---
	浮点三角函数	RADD	双精度度→弧度	DEGD	双精度弧度→度	SIND	双精度正弦
		COSD	双精度余弦	TAND	双精度正切	ASIND	双精度反正弦
		ACOSD	双精度反余弦	ATAND	双精度反正切	---	---
	浮点数学	SQRTD	双精度平方根	EXPD	双精度指数	LOGD	双精度对数
		PWRD	双精度指数幂	---	---	---	---
	符号比较与转换	LD, AND, OR + =, <>, <, <=, >, >= + D	符号比较 （双精度浮点数）	---	---	---	---
表格数据处理指令	堆栈处理	SSET	设置堆栈	PUSH	压入堆栈	LIFO	后进先出
		FIFO	先进先出	SNUM*	堆栈的尺寸读	SREAD*	堆栈中数据读
		SWRIT*	堆栈中数据重写	SINS*	堆栈中数据插入	SDEL*	堆栈中数据删除
	1 记录 / 多字处理	DIM	定维记录表格	SETR	设置记录位置	GETR	取记录号
	记录→字处理	SRCH	数据搜索	MAX	寻找最大值	MIN	寻找最小值
		SUM	求和	FCS	帧校验	---	---
	字节处理	SWAP	交换字节	---	---	---	---
数据控制指令	---	PID	PID 控制	PIDAT*	带自整定的 PID 控制	LMT	极限控制
		BAND	静带控制	ZONE	静域控制	SCL	标度
		SCL2	标度 2	SCL3	标度 3	AVG	平均值
子程序指令	---	SBS	子程序调用	MCRO	宏	SBN	子程序入口
		RET	子程序返回	GSBS*	全局子程序调用	GSBN*	全局子程序入口
		GRET*	全局子程序返回	---	---	---	---
中断控制指令	---	MSKS***	设置中断屏蔽	MSKR***	读中断屏蔽	CLI***	清除中断
		DI	禁止中断	EI	允许中断	---	---

分类	细类	助记符	指令	助记符	指令	助记符	指令
高速计数器 / 脉冲输出指令 **	---	INI	模式控制	PRV	高速计数器 PV 值读	CTBL	比较表载入
		SPED	速度输出	PULS	设置脉冲	PLS2	脉冲输出
		ACC	加速度控制	ORG	起始地址搜索	PWM	可变占空比系数脉冲
步指令	---	STEP	步定义	SNXT	步启动	---	---
基本 I/O 单元指令	---	IORF	I/O 刷新	SDEC	七段译码	IORD	智能 I/O 读
		IOWR	智能 I/O 写	DLNK*	CPU 总线 I/O 单元刷新	---	---
串行通信指令	---	PMCR	协议宏	TXD	发送	RXD	接收
		STUP	修改串口设置	---	---	---	---
网络指令	---	SEND	网络发送	RECV	网络接收	CMND	发布命令
显示指令	---	MSG	显示信息	---	---	---	---
文件存储指令	---	FREAD	读数据文件	FWRIT	写数据文件	---	---
时钟指令	---	CADD	日历加	CSUB	日历减	SEC	小时→秒
		HMS	秒→小时	DATE	时钟调整	---	---
调试指令	---	TRSM	跟踪存储器采样	---	---	---	---
故障诊断指令	---	FAL	故障报警	FALS	严重故障报警	FPD	故障点检测
其它指令	---	STC	置进位	CLC	清除进位	EMBC	选择 EM 区
		WDT	延长最大循环时间	CCS*	保存条件标志	CCL*	载入条件标志
		FRMCV*	把 CV 值转换成地址值	TOCV*	把地址值转换为 CV	IOSP***	禁止外部服务
		IORS***	允许外部设备服务	---	---	---	---

分类	细类		助记符	指令	助记符	指令	助记符	指令
块指令	定义块编程区		BPRG	块程序开始	BEND	块程序结束	---	---
	块程序启动 / 停止		BPPS	块程序暂停	BPRS	块程序重新启动	---	---
	退出		EXIT *位地址*	条件结束	EXIT NOT *位地址*	条件结束非	输入条件 EXIT	条件结束
	IF 分支处理		IF *位地址*	条件块分支	IF NOT *位地址*	条件块分支（非）	ELSE	条件块分支（另）
			IEND	条件块分支结束	---	---	---	---
	等待		WAIT *位地址*	一个循环并等待	WAIT NOT *位地址*	一个循环并等待非	*输入条件* WAIT	一个循环并等待
	定时器 / 计数器	BCD	TIMW	定时器等待	CNTW	计数器等待	TMHW	高速计数器等待
		二进制 *	TIMWX	定时器等待	CNTWX	计数器等待	TMHWX	高速计数器等待
	重复		LOOP	循环块	LEND *位地址*	循环块结束	LEND NOT *位地址*	循环块结束非
			输入条件 LEND	循环块结束	---	---	---	---
文本字符串处理指令	---		MOV$	移动串	+$	连接串	LEFT$	取左串
			RIGHT$	取右串	MID$	取中间串	FIND$	查找串
			LEN$	串长度	RPLC$	替换串	DEL$	删除串
			XCHG$	交换串	CLR$	清除串	INS$	插入串
			LD, AND, OR + =$, <>$, <$, <=$, >$, >=$	串比较	---	---	---	---
任务控制指令	---		TKON	任务 ON	TKOF	任务 OFF	---	---

2-2 指令功能

2-2-1 顺序输入指令

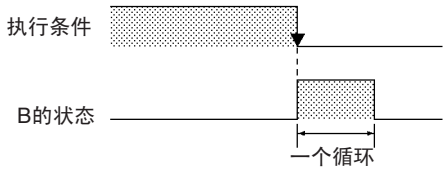
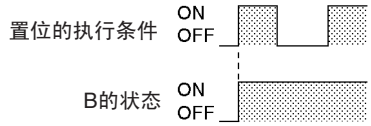
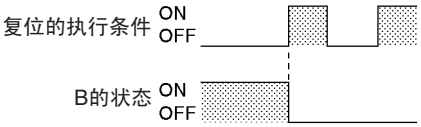
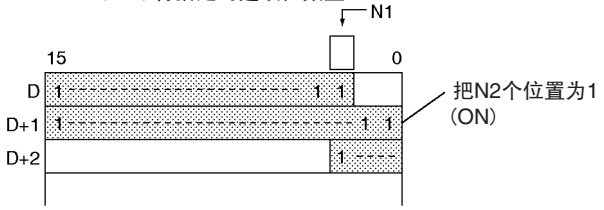
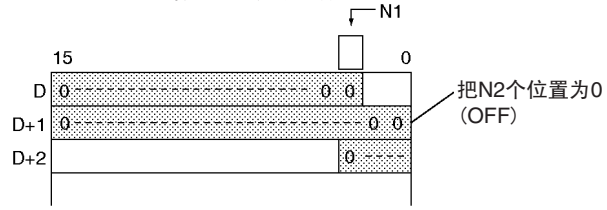
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
装载 LD @LD %LD !LD !@LD !%LD		指定一个逻辑开始，并且用指定操作位的 ON/OFF 状态建立一个 ON/OFF 执行条件	逻辑开始 不需要	142
装载非 LD NOT @LD NOT %LD NOT !LD NOT !@LD NOT !%LD NOT 指令 @LD NOT %LD NOT !@LD NOT !%LD NOT 仅适用于 CS1-H， CJ1-H， CJ1M CPU 单元		指定一个逻辑开始，并且用指定操作位的 ON/OFF 状态取反建立一个 ON/OFF 执行条件。	逻辑开始 不需要	144
与 AND @AND %AND !AND !@AND !%AND		把指定的操作位状态和当前执行条件进行逻辑与操作。	行线上继续 需要	146
与非 AND NOT @AND NOT %AND NOT !AND NOT !@AND NOT !%AND NOT 指令 @AND NOT %AND NOT !@AND NOT !%AND NOT 仅适用于 CS1-H，CJ1-H， CJ1M CPU 单元		把指定操作位的状态取反并和当前执行条件进行逻辑与。	行线上继续 需要	148
或 OR @OR %OR !OR !@OR !%OR		把指定操作位的 ON/OFF 状态和当前执行条件进行逻辑或操作。	行线上继续 需要	150
或非 OR NOT @OR NOT %OR NOT !OR NOT !@OR NOT !%OR NOT 指令 @OR NOT %OR NOT !@OR NOT !%OR NOT 仅适用于 CS1-H， CJ1-H， CJ1M CPU 单元		把指定位状态取反和当前执行条件进行逻辑或操作。	行线上继续 需要	151

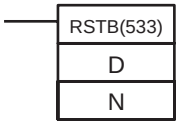
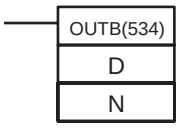
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
与装载 AND LD	逻辑块 - 逻辑块与	在逻辑块之间进行逻辑与。 LD ～ 逻辑块A LD ～ 逻辑块B AND LD ····· 把逻辑块A和逻辑块B串联起来。	行线上继续 需要	153
或装载 OR LD	逻辑块 逻辑块或	在逻辑块之间进行逻辑或。 LD ～ 逻辑块A LD ～ 逻辑块B OR LD ····· 把逻辑块A和逻辑块B并联起来。	行线上继续 需要	155
非 NOT 520	---	执行条件取反。	行线上继续 需要	161
条件 ON UP 521	UP(521)	当执行条件从 OFF → ON 时， UP(521) 把执行条件在一个循环内变 ON。	行线上继续 需要	162
条件 OFF DOWN 522	DOWN(522)	当执行条件从 ON → OFF 时， DOWN(522) 把执行条件在一个循环内变 ON。	行线上继续 需要	162
位测试 LD TST 350	TST(350) S N S:源字 N:位号	LD TST(350)，AND TST(350) 和 OR TST(350) 在程序中的用途类似于 LD， AND 和 OR，当指定字中的指定位为 ON 时，执行条件为 ON，反之，执行条件为 OFF。	行线上继续 不需要	163
位测试 LD TSTN 351	TSTN(351) S N S:源字 N:位号	LD TSTN(351)， AND TSTN(351) 和 OR TSTN(351) 在程序中的用途类似于 LD NOT， AND NOT 和 OR NOT，当指定字中的指定位为 ON 时，执行条件为 OFF，反之，执行条件为 ON。	行线上继续 不需要	163
位测试 AND TST 350	AND TST(350) S N S:源字 N:位号	LD TST(350)，AND TST(350) 和 OR TST(350) 在程序中的用途类似于 LD， AND 和 OR，当指定字中的指定位为 ON 时，执行条件为 ON，反之，执行条件为 OFF。	行线上继续 需要	163
位测试 AND TSTN 351	AND TSTN(351) S N S:源字 N:位号	LD TSTN(351)， AND TSTN(351) 和 OR TSTN(351) 在程序中的用途类似于 LD NOT， AND NOT 和 OR NOT，当指定字中的指定位为 ON 时，执行条件为 OFF，反之，执行条件为 ON。	行线上继续 需要	163

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
位测试 OR TST 350	TST(350) S N S:源字 N:位号	LD TST(350)，AND TST(350) 和 OR TST(350) 在程序中的用途类似于 LD， AND 和 OR，当指定字中的指定位为 ON 时，执行条件为 ON，反之，执行条件为 OFF。	行线上继续 需要	163
位测试 OR TSTN 351	TSTN(351) S N S:源字 N:位号	LD TSTN(351)， AND TSTN(351) 和 OR TSTN(351) 在程序中的用途类似于 LD NOT， AND NOT 和 OR NOT，当指定字中的指定位为 ON 时，执行条件为 OFF，反之，执行条件为 ON。	行线上继续 需要	163

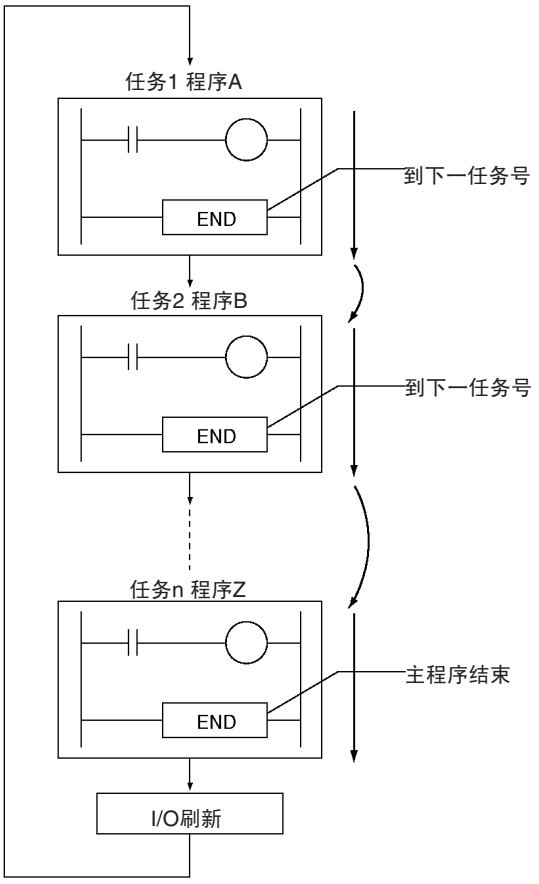
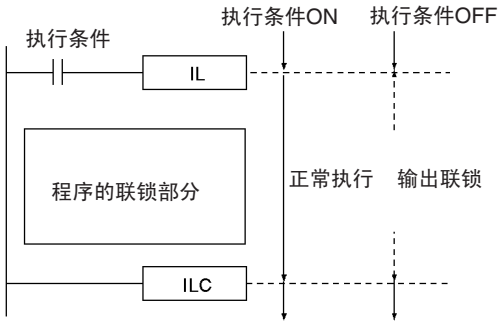
2-2-2 顺序输出指令

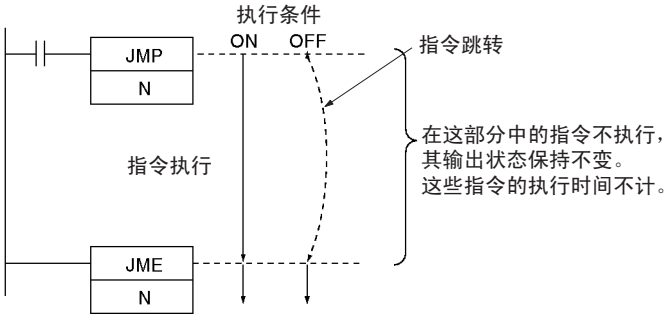
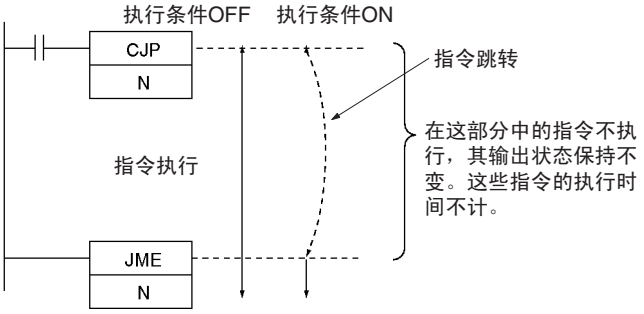
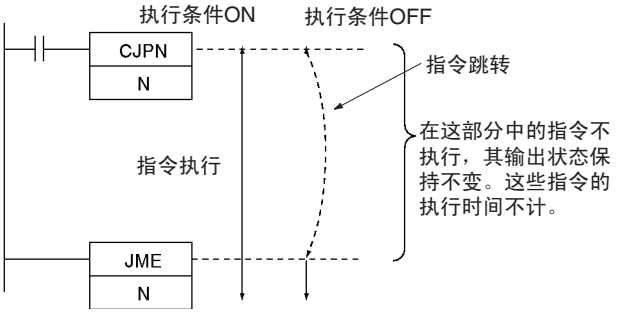
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
输出 OUT !OUT		把逻辑运算的结果 （执行条件）输出到指定位。	输出 需要	166
输出 OUT NOT !OUT NOT		把逻辑运算的结果 （执行条件）取反并输出到指定位。	输出 需要	167
保持 KEEP !KEEP 011	S（置位） KEEP(011) B R（复位） B:位号	置位 A KEEP C 复位 B = A B C / A C 置位执行条件 复位执行条件 B的状态	输出 需要	168
上升沿微分 DIFU !DIFU 013	DIFU(013) B B:位号	当执行条件从OFF→ON变化（上升沿）时， DIFU (013) 将所指定的位在一个循环内变为ON。 执行条件 B的状态 一个循环	输出 需要	173

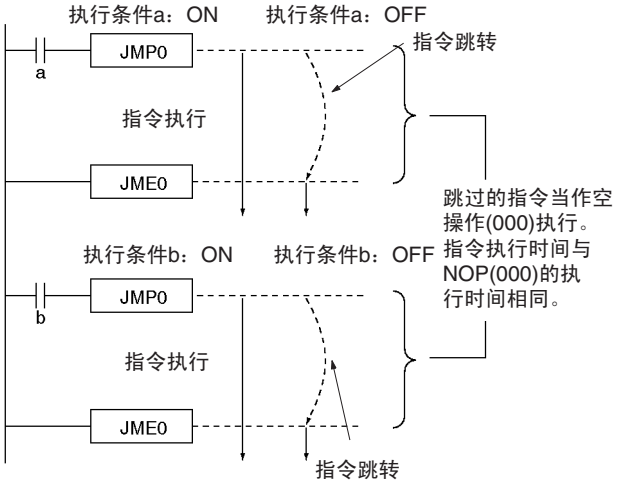
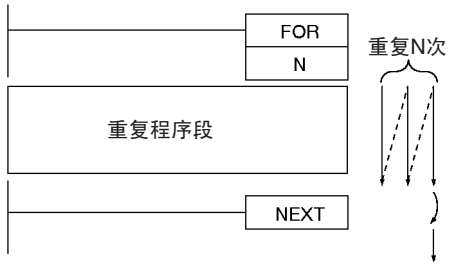
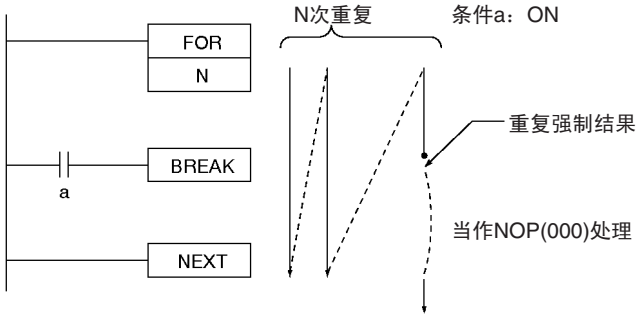
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
下降沿微分 DIFD !DIFD 014	DIFD(014) B B:位号	当执行条件从ON→OFF变化（下降沿）时， DIFD (014) 将所指定的位在一个循环内变为ON。 	输出 需要	173
置位 SET @SET %SET !SET !@SET !%SET	SET B B:位号	SET在执行条件为ON时，把操作位变为ON。 	输出 需要	175
复位 RSET @RSET %RSET !RSET !@RSET !%RSET	RSET B B:位号	RSET在执行条件为ON时，把操作位变为OFF。 	输出 需要	175
多位置位 SETA @SETA 530	SETA(530) D N1 N2 D:起始字 N1:起始位 N2:位数量	SETA (530) 将指定的连续位数置ON。 	输出 需要	177
多位复位 RSTA @RSTA 531	RSTA(531) D N1 N2 D:起始字 N1:起始位 N2:位数量	RSTA (531) 将指定的连续位数置OFF。 	输出 需要	177
单个位置位 （仅适用于 CS1-H, CJ1-H, CJ1M, 单元） SETB @SETB !SETB	SETB(532) D N D:字地址 N:位号	当执行条件为 ON 时， SETB(532) 将指定字中的指定位置 ON。 不象 SET 指令， SETB(532) 可用作在一个 DM 或 EM 字中对某一位作置位操作。	输出 需要	180

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
单个位复位 （仅适用于 CS1-H, CJ1-H, CJ1M 或者 CS1D 单元） RSTB @RSTB !RSTB	 D:字地址 N:位号	当执行条件为 ON 时， RSTB(533) 将指定字中的指定位清为 OFF。 不象 RSET 指令， RSTB(533) 可用作在一个 DM 或 EM 字中对某一位作复位操作。	输出 需要	180
单个位输出 （仅适用于 CS1-H, CJ1-H, CJ1M 或者 CS1D 单元） OUTB @OUTB !OUTB	 D:字地址 N:位号	OUTB(534) 输出逻辑处理的结果 （执行条件）给指定位。 不象 OUT 指令， OUTB(534) 可在一个 DM 或 EM 字中用于控制某一位。	输出 需要	184

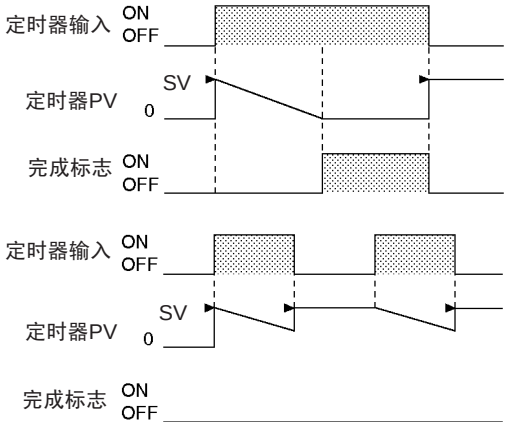
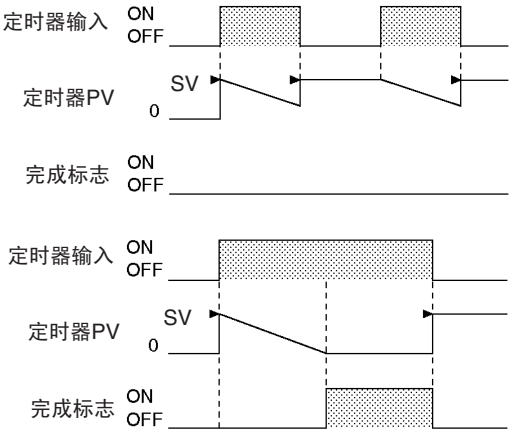
2-2-3 顺序控制指令

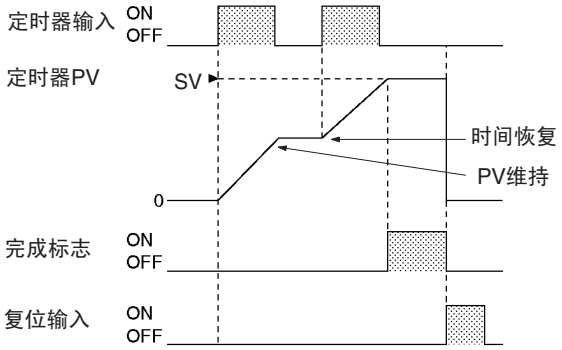
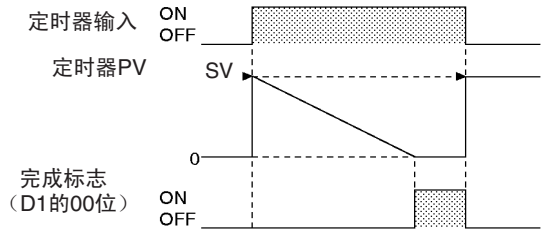
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
结束 END 001	END(001)	表明程序结束。 END(001)作为一个循环内的程序执行的完成。 END(001)指令后面的任何指令都不执行。 执行下一个任务号程序。 当程序执行到最高任务号时，END(001)意味着主程序结束。 	输出 不需要	186
空操作 NOP 000		此指令无任何功能。（NOP(000) 不做任何操作）	输出 不需要	187
联锁 IL 002	IL(002)	当IL(002)的执行条件为OFF时，IL(002)和ILC(003)之间的所有输出都联锁。IL(002)和ILC(003)总是成对使用。 	输出 需要	187

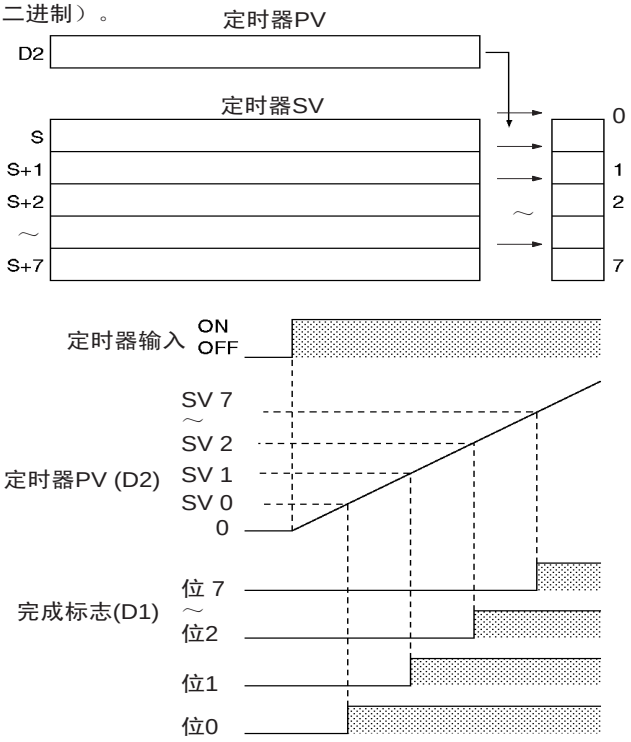
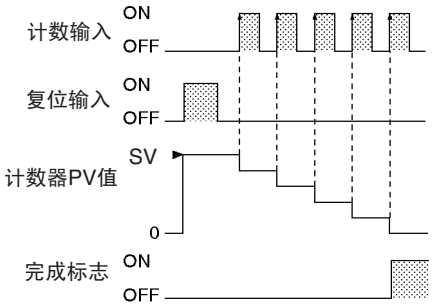
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
联锁解除 ILC 003	ILC(003)	当 IL(002) 执行条件为 OFF 时， IL(002) 和 ILC(003) 之间的所有输出都联锁。 IL(002) 和 ILC(003) 总是成对使用。	输出 不需要	187
跳转 JMP 004	JMP(004) N N:跳转号	当JMP(004)的执行条件为OFF时，程序执行直接跳转至与JMP(004)指令相同编号的第一个JME(005)。 JMP(004)和 JME(005)总是成对使用。 	输出 需要	191
跳转结束 JME 005	JME(005) N N:跳转号	表明 JMP(004) 或 CJP(510) 跳转结束。	输出 不需要	191
条件跳转 CJP 510	CJP(510) N N:跳转号	CJP(510)的用法和JMP(004)相反。 当CJP(510)的执行条件为ON时，程序执行直接跳转至与CJP(510)指令相同编号的第一个JME(005)。CJP(510)和JME(005)总是成对使用。 	输出 需要	195
条件跳转 CJPN 511	CJPN(511) N N:跳转号	CJPN(511)的用法几乎等同于JMP(004)。 当CJP(511)的执行条件为OFF时，程序执行直接跳转至与CJPN(511)指令相同编号的第一个JME(005)。CJPN(511)和JME(005)总是成对使用。 	输出 不需要	195

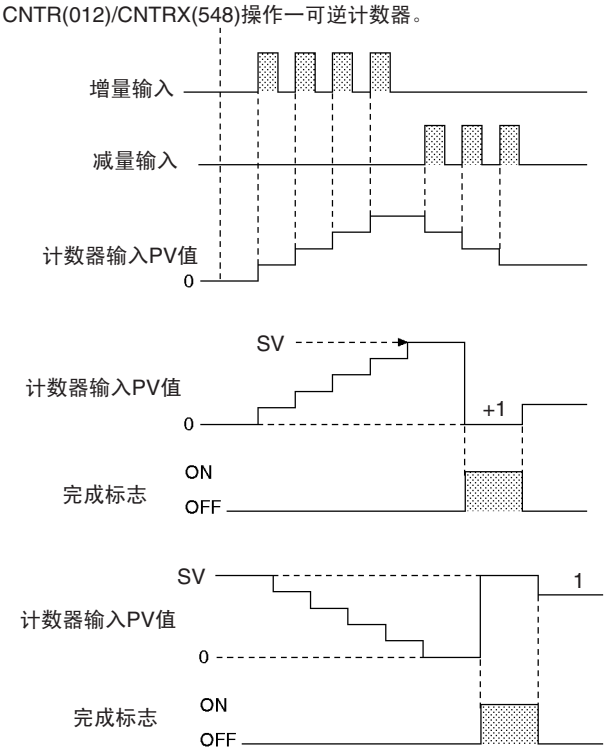
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
多路跳转 JMP0 515	JMP0(515)	当JMP0(515)的执行条件为OFF时，从JMP0(515)至下一JME0(516)的所有指令都被当作空操作(000)。JMP0(515)和JME0(516)成对使用。在程序中能使用的对数无任何限制。 	输出 需要	199
多路跳转结束 JME0 516	JME0(516)	当 JMP0(515) 的执行条件为 OFF 时，从 JMP0(515) 至下一 JME0(516) 的所有指令都被当作 NOP(000)。 JMP0(515) 和 JME0(516) 成对使用。在程序中能使用的对数无任何限制。	输出 不需要	199
FOR-NEXT 循环 FOR 512	FOR(512) N N:循环编号	FOR(512)和NEXT(513)之间的指令重复指定的次数。FOR(512)和NEXT(513)成对使用。 	输出 不需要	201
退出循环 BREAK 514	BREAK(514)	在FOR-NEXT循环中编程，对所给的执行条件取消循环执行。循环中余下的指令作为NOP(000)处理。 	输出 需要	204
FOR-NEXT 循环 NEXT 513	NEXT(513)	FOR(512) 和 NEXT(513) 之间的指令重复指定的次数。 FOR(512) 和 NEXT(513) 成对使用。	输出 不需要	201

2-2-4 定时器和计数器指令

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
定时器 TIM (BCD) TIMX （二进制） （仅适用于 CS1-H, CJ1-H, CJ1M 或者 CS1D）	TIM N S N:定时器编号 S:设定值 TIMX(550) N S N:定时器编号 S:设定值	TIM/ TIMX（550）定时器以0.1S为单位作一减量计时，此设定值（SV）的设定值范围为：0～999.9s（BCD）和0～6553.5s（二进制）。 	输出 需要	207
高速定时器 TIMH 015 (BCD) TIMHX 551 （二进制） （仅适用于 CS1-H, CJ1-H, CJ1M 或者 CS1D）	TIMH(015) N S N:定时器编号 S:设定值 TIMHX(551) N S N:定时器编号 S:设定值	TIMH(015)/TIMHX(551)以10ms为单位一减量计时。此设定值(SV)的设定范围为：0～99.99s(BCD)和655.35s（二进制）。 	输出 需要	211
1 毫秒定时器 TMHH 540 (BCD) TMHHX 552 (BCD) （仅适用于 CS1-H, CJ1-H, CJ1M 或者 CS1D）	TMHH(540) N S N:定时器编号 S:设定值 TMHHX(552) N S N:定时器编号 S:设定值	TMHH(540)/TIMHHX(552) 以 1ms 为单位作一减量计时。此设定值 (SV) 的设定范围为：0~9.999s 和 0~65.535s （二进制）。 TMHH(540) 和 TIMHHX(552) 的定时图表和上面所给的 TMHH(015) 图表相同。	输出 需要	216

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
累加定时器 TTIM 087 (BCD) TTIMX 555 （二进制） （仅适用于 CS1-H, CJ1-H, CJ1M 或者 CS1D）	定时器输入 TTIM(087) N S 复位输入 N:定时器编号 S:设定值 定时器输入 TTIMX(555) N S 复位输入 N:定时器编号 S:设定值	TTIM(087)/ TTIMX(555)定时器以0.1s为单位作一增量定时，此设定值(SV)的设定值范围为：0～999.9s(BCD)和0～6,553.5s（二进制）。 	输出 需要	219
长定时器 TIML 542 (BCD) TIMLX 553 （二进制） （仅适用于 CS1-H，CJ1-H，CJ1M 或者 CS1D）	TIML(542) D1 D2 S D1:完成标志 D2:PV字 S:SV字 TIMLX(553) D1 D2 S D1:完成标志 D2:PV字 S:SV字	TIML(542)/TIMLX(553)以0.1s为单位作一减量计时。 最高可定时到：约为115天(BCD)和49,710天（二进制）。 	输出 需要	222

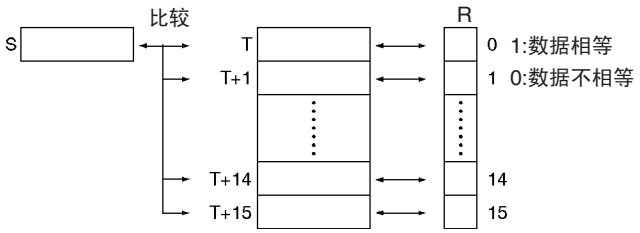
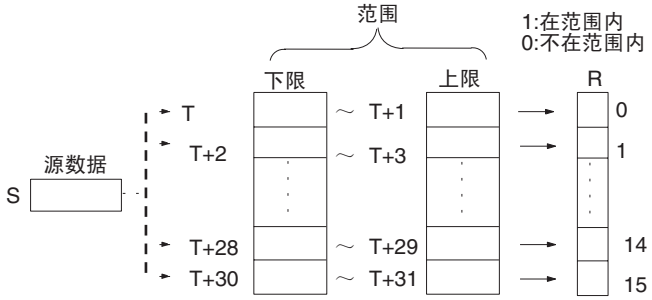
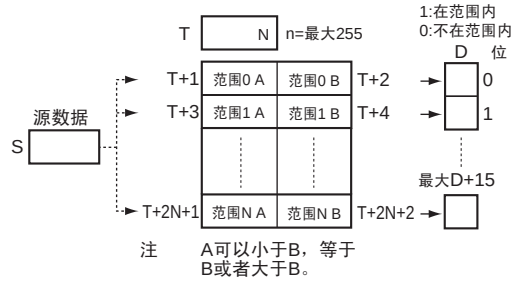
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
多路输出定时器 MTIM 543 (BCD) MTIMX 554 （二进制） （仅适用于 CS1-H, CJ1-H, CJ1M 或者 CS1D）	MTIM(543) D1 D2 S D1:完成标志 D2:PV字 S:SV首字 MTIMX(554) D1 D2 S D1:完成标志 D2:PV字 S:SV首字	MTIM(543)/ MTIMX(554)定时器以0.1s为单位作一增量定时，此定时器有八个独立的SV和完成标志。 此设定值(SV)的设定值范围为0～999.9s(BCD)和0～6,553.5s（二进制）。 	输出 需要	226
计数器 CNT (BCD) CNTX 546 （二进制） （仅适用于 CS1-H， CJ1-H，CJ1M 或者 CS1D）	计数输入 CNT N S 复位输入 N:计数器编号 S:设定值 计数输入 CNTX(546) N S 复位输入 N:计数器编号 S:设定值	CNT/CNTX(546)作一减量计数。此设定值(SV)的设定值范围为:0～9,999(BCD)或0～65,535（二进制）。 	输出 需要	231

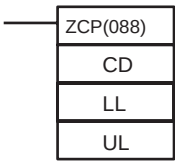
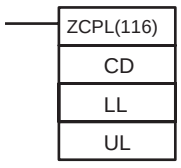
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
可逆计数器 CNTR 012 (BCD) CNTRX 548 （二进制） （仅适用于 CS1-H， CJ1-H，CJ1M 或者 CS1D）	增量输入 减量输入 复位输入 CNTR(012) N S N:计数器编号 S:设定值 增量输入 减量输入 复位输入 CNTRX(548) N S N:计数器编号 S:设定值	CNTR(012)/CNTRX(548)操作一可逆计数器。 	输出 需要	234
复位定时器 / 计数器 CNR @CNR 545 (BCD) CNRX @CNRX 547 （二进制） （仅适用于 CS1-H， CJ1-H，CJ1M 或者 CS1D）	CNR(545) N1 N2 N_1:范围中第一个编号 N_2:范围中最后一个编号 CNRX(547) N1 N2 N_1:范围中第一个编号 N_2:范围中最后一个编号	CNR(545)/CNRX(547) 在一指定定时器或计数器号范围内对定时器或计数器复位。把设定值 (SV) 设至最大，为 9999。	输出 需要	238

2-2-5 比较指令

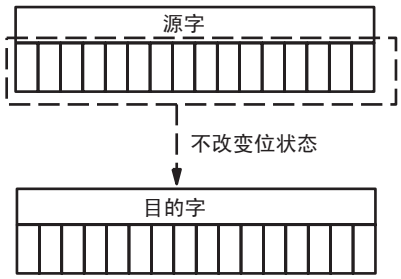
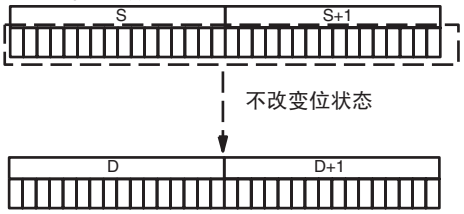
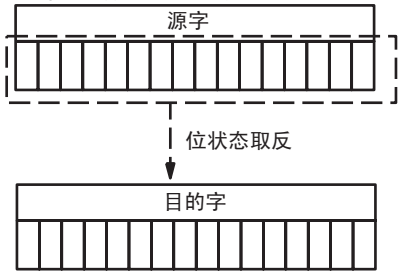
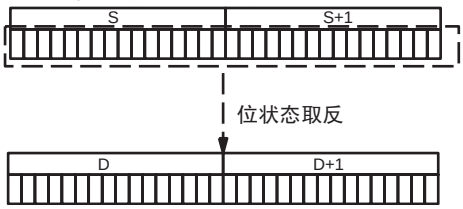
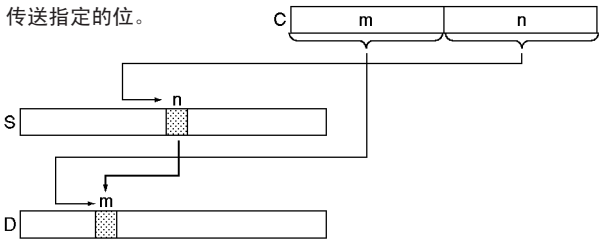
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
符号比较 （不带符号） LD, AND, OR + =, <>, <, <=, >, >= 300 (=) 305 (<>) 310 (<) 315 (<=) 320 (>) 325(>=)	符号和选项 S_1 S_2 S_1:比较数据1 S_2:比较数据2	符号比较指令（不带符号）以16位二进制数据形式对二个数值（常数和/或指定字的内容）进行比较，并且当比较条件是真时，形成一个ON执行条件。有三种类型的符号比较指令：LD(LOAD)，AND和OR。 LD 当比较结果是真时，执行条件为ON。 AND 当比较结果是真时，执行条件为ON。 OR 当比较结果是真时，执行条件为ON。	LD：不需要 AND, OR：需要	246
符号比较 （双字，不带符号） LD, AND, OR + =, <>, <, <=, >, >= + L 301 (=) 306 (<>) 311 (<) 316 (<=) 321 (>) 326 (>=)	S_1: 比较数据 1 S_2: 比较数据 2	符号比较指令 （双字，不带符号）以不带符号的 32 位二进制数据对二个数值 （常数和 / 或指定字的内容）进行比较，并且当比较条件是真时，形成一个 ON 执行条件。有三种类型的符号比较指令：LD(LOAD)，AND 和 OR。	LD：不需要 AND, OR：需要	246
符号比较 （带符号） LD, AND, OR + =, <>, <, <=, >, >= +S 302 (=) 307 (<>) 312 (<) 317 (<=) 322 (>) 327 (>=)	S_1: 比较数据 1 S_2: 比较数据 2	符号比较指令 （带符号的）以带符号的 16 位二进制 （4 位数十六进制）对二个数值 （常数和 / 或指定字的内容）进行比较，并且当比较条件为真时，形成一个 ON 执行条件。有三种类型的符号比较指令：LD(LOAD)，AND 和 OR。	LD：不需要 AND, OR：需要	246

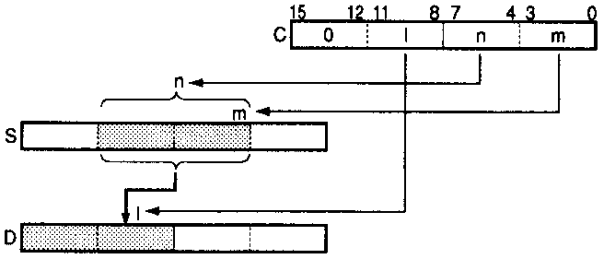
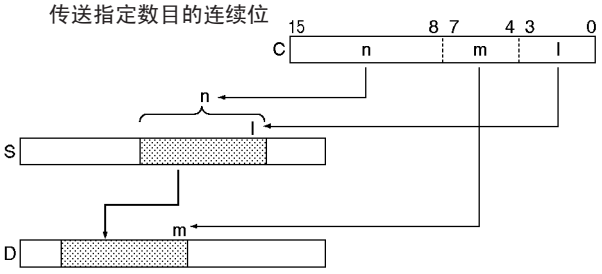
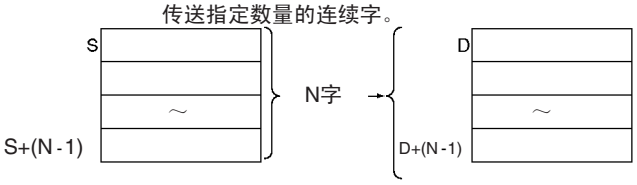
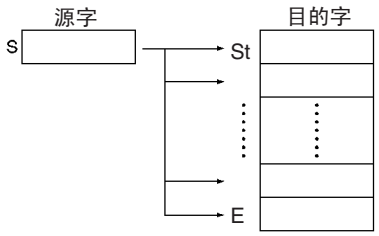
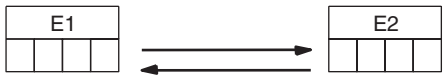
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
符号比较 （带符号双字） LD, AND, OR + =, <>, <, <=, >, >= +SL 303 (=) 308 (<>) 313 (<) 318 (<=) 323 (>) 328 (>=)	S_1: 比较数据 1 S_2: 比较数据 2	符号比较指令 （双字，带符号）以带符号的 32 位二进制数据 （8 位数十六进制）对二个数值 （常数和 / 或指定字的内容）进行比较，并且当比较条件是真时，形成一个 ON 执行条件。有三种类型的符号比较指令：LD(LOAD)， AND 和 OR。	LD：不需要 AND, OR：需要	246
比较 CMP !CMP 020	CMP(020) S_1 S_2 S_1:比较数据1 S_2:比较数据2	比较二个不带符号的二进制数值（常数和/或指定字的内容），并且把结果输出到辅助区的算术标志中。 不带符号二进制数据比较 S1 ↔ S2 算术标志 (>, >=, =, <=, <, <>)	输出 需要	252
双字比较 CMPL 060	CMPL(060) S_1 S_2 S_1:比较数据1 S_2:比较数据2	比较二个双字不带符号的二进制数值（常数和/或指定字的内容），并且把结果输出到辅助区的算术标志中。 不带符号二进制数据比较 S1+1 S1 ↔ S2+1 S2 算术标志 (>, >=, =, <=, <, <>)	输出 需要	254
带符号二进制数比较 CPS !CPS 114	CPS(114) S_1 S_2 S_1:比较数据1 S_2:比较数据2	比较二个带符号的二进制数值（常数和/或指定字的内容），并且把结果输出到辅助区的算术标志中。 带符号二进制数据比较 S1 ↔ S2 算术标志 (>, >=, =, <=, <, <>)	输出 需要	257
带符号双字二进制数比较 CPSL 115	CPSL(115) S_1 S_2 S_1:比较数据1 S_2:比较数据2	比较二个双字带符号的二进制数值（常数和/或指定字的内容），并且把结果输出到辅助区的算术标志中。 带符号二进制数据比较 S1+1 S1 ↔ S2+1 S2 算术标志 (>, >=, =, <=, <, <>)	输出 需要	260
多路比较 MCMP @MCMP 019	MCMP(019) S_1 S_2 R S_1:第1组的首字 S_2:第2组的首字 R:结果字	将16个连续字与另一16个连续字进行比较，并且当结果字内容不相等时，使相应的位变ON。 比较 R S1 ↔ S2 ↔ 0 S1+1 ↔ S2+1 ↔ 1 0:字相等 1:字不相等 S1+14 ↔ S2+14 ↔ 14 S1+15 ↔ S2+15 ↔ 15	输出 需要	263

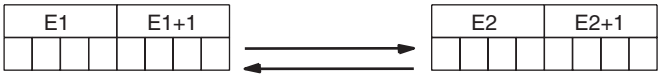
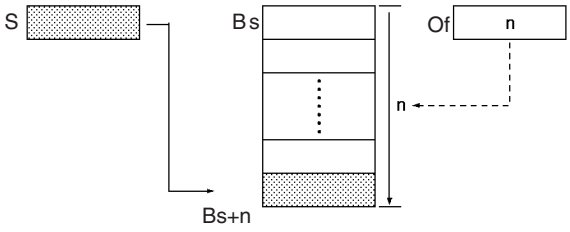
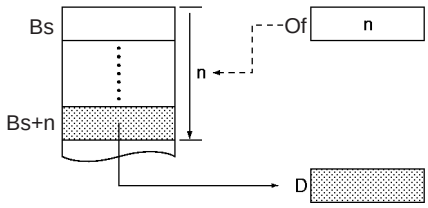
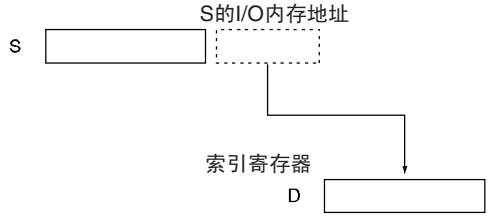
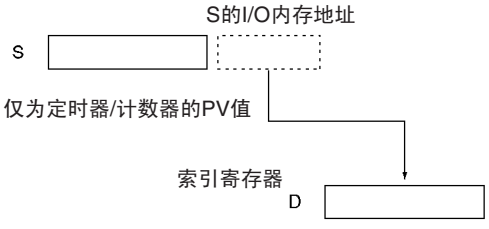
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
表格比较 TCMP @TCMP 085	TCMP(085) S T R S:源数据 T:表格首字 R:结果字	将源数据和16个连续字的内容进行比较， 在字的内容相等时，使相应的位变ON。 	输出 需要	265
不带符号的块比较 BCMP @BCMP 068	BCMP(068) S T R S:源数据 T:表格首字 R:结果字	将源数据在16个范围（以16个下限和16个上限来定义） 内比较，源数据在范围内时，使相应的位变ON。 	输出 需要	268
扩展块比较 BCMP2 @BCMP2 502 （仅适用于 CJ1M）	BCMP2(502) S T R S:源数据 T:表格首字 R:结果字	将源数据在 256 个范围（以上、下限定义）内比较，源数据在范围内时， 使相应的位变 ON。  注 A可以小于B，等于 B或者大于B。	输出 需要	270

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
区域范围比较 （仅适用于 CS1-H，CJ1-H，CJ1M 或 CS1D） ZCP @ZCP 088	 CD:比较数据 （1个字） LL:范围的下限 UL:范围的上限	将 CD 中 （字的内容或常数） 16 位不带符号二进制数和由 LL 和 UL 定义的范围比较，将把输出结果送到辅助区的算术标志中。	输出 需要	274
双字区域比较 （仅适用于 CS1-H，CJ1-H，CJ1M 或 CS1D） ZCPL @ZCPL 116	 CD:比较数据 （2个字） LL:范围的下限 UL:范围的上限	将 CD 和 CD+1 中 （字的内容或常数） 32 位不带符号二进制数和由 LL 和 UL 定义的范围比较，将把输出结果送到辅助区算术标志。	输出 需要	277

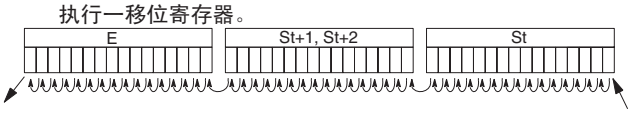
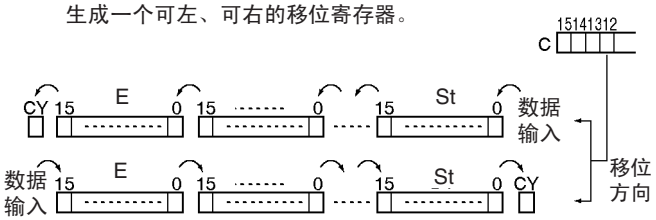
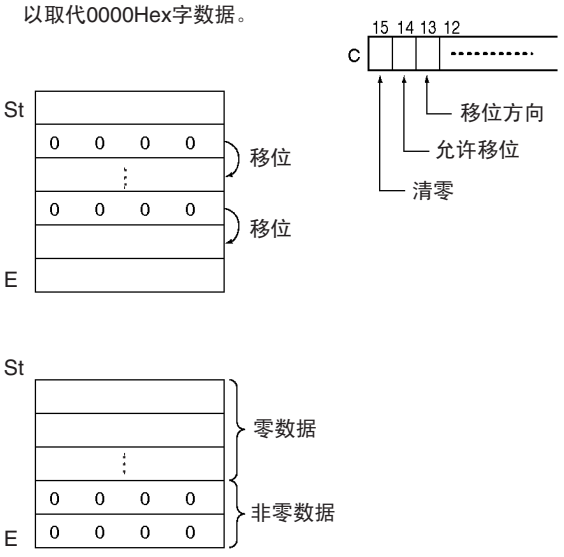
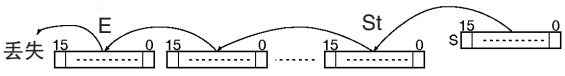
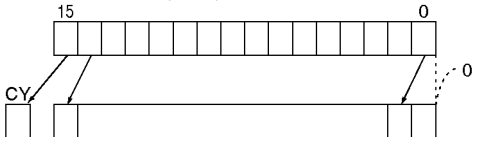
2-2-6 数据传送指令

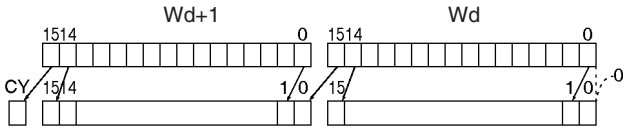
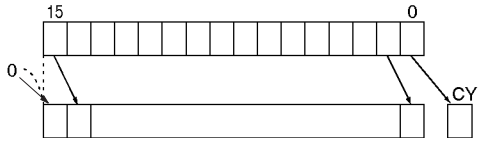
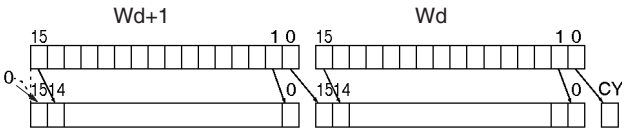
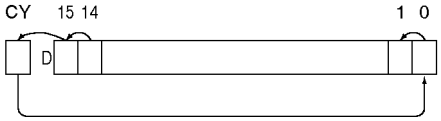
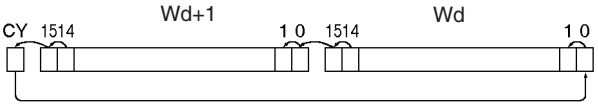
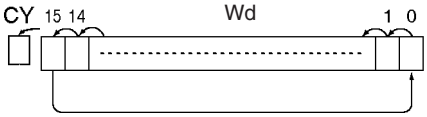
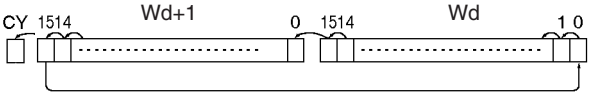
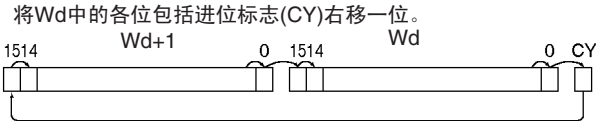
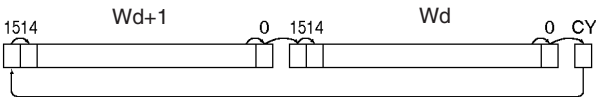
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
传送 MOV @MOV !MOV !@MOV 021	MOV(021) S D S:源 D:目的	把一个字的数据传送到指定字中。 	输出 需要	279
双字传送 MOVL @MOVL 498	MOVL(498) S D S:源起始字 D:目的起始字	把二个字的数据传送到指定的字中。 	输出 需要	282
取反传送 MVN @MVN 022	MVN(022) S D S:源 D:目的	把一个字的数据的反码传送指定字。 	输出 需要	281
双字取反传送 MVNL @MVNL 499	MVNL(499) S D S:源起始字 D:目的首字	把二个字的数据的反码传送到指定的字中。 	输出 需要	284
位传送 MOVB @MOVB 082	MOVB(082) S C D S:源字或数据 C:控制字 D:目的字	传送指定的位。 	输出 需要	285

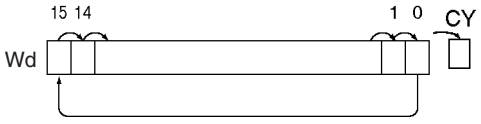
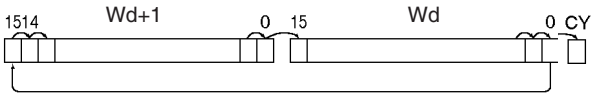
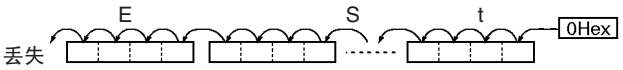
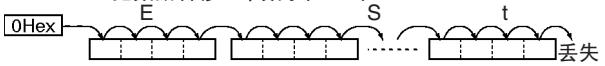
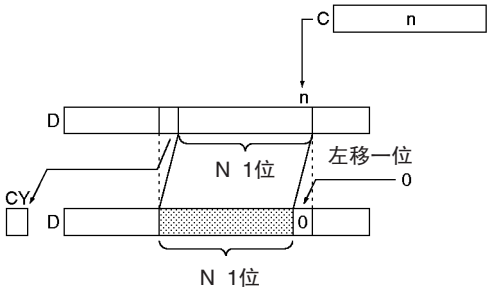
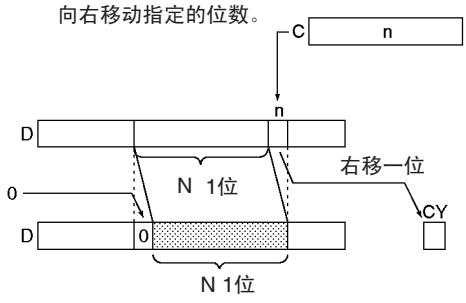
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
数字传送 MOVD @MOVD 083	MOVD(083) S C D S:源字或数据 C:控制字 D:目的字	传送指定的数字或多个数字。（每个数字由四个位组成） 	输出 需要	287
多位传送 XFRB @XFRB 062	XFRB(062) C S D C:控制字 S:源起始字 D:目的首字	传送指定数目的连续位 	输出 需要	290
块传送 XFER @XFER 070	XFER(070) N S D N:字数目 S:源起始字 D:目的首字	传送指定数量的连续字。 	输出 需要	292
块设置 BSET @BSET 071	BSET(071) S St E S:源字 St:开始字 E:结束字	将相同字的内容复制到几个连续字中。 	输出 需要	295
数据交换 XCHG @XCHG 073	XCHG(073) E1 E2 E1:第一个交换字 E2:第二个交换字	把两个指定字的内容进行交换。 	输出 需要	297

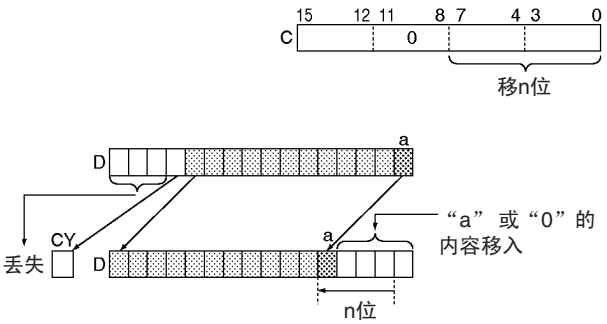
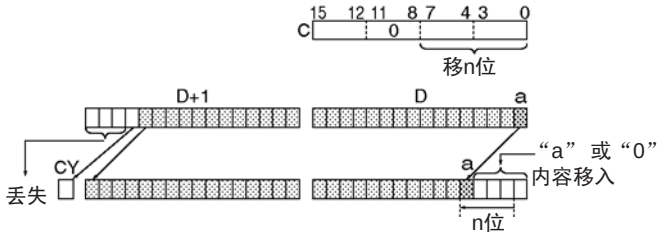
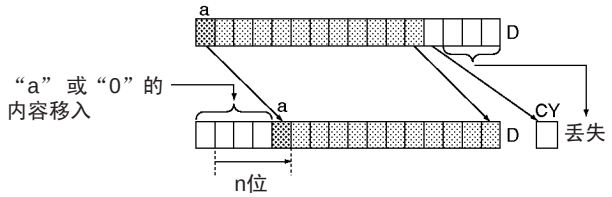
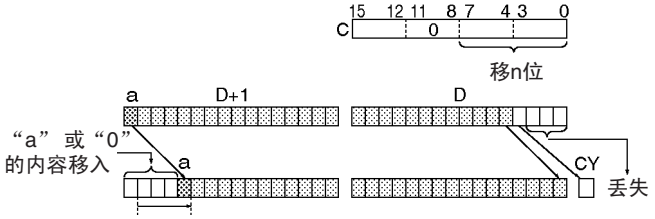
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
双字数据交换 XCGL @XCGL 562	XCGL(562) E1 E2 E1:第一个交换字 E2:第二个交换字	把一对连续字的内容与另一对连续字的内容进行交换。 	输出 需要	298
单字分配 DIST @DIST 080	DIST(080) S Bs Of S:源字 Bs:目的基地址 Of:偏移	把源字传送到目的字中，目的字的地址由目的基地址加上一偏移值。 	输出 需要	300
数据收集 COLL @COLL 081	COLL(081) Bs Of D Bs:源基地址 Of:偏移 D:目的字	把源字（在源基地址上加一偏移值）传送到目的字。 	输出 需要	302
传送至寄存器 MOVR @MOVR 560	MOVR(560) S D S:源 （所需字和位） D:目的 （索引寄存器）	把指定字、位或定时器/计数器完成标志的PC内部I/O内存地址存放到指定的索引寄存器中。（使用MOVRW(561)指令）把一个字、位、或定时器/计时器PV值的内部 I/O内存地址存放到一个索引寄存器中）。 	输出 需要	304
传送定时器 / 计数器 PV 值至寄存器 MOVRW @MOVRW 561	MOVRW(561) S D S：源字 （所需TC号） D：目的 （索引寄存器）	把定时器/计数器的PV值的PC内部I/O内存地址存放到指定的寄存器中。（使用MOVR(560)指令）把一个定时器/计时器完成标志的内部 I/O内存地址存放到一个索引寄存器中。 	输出 需要	306

2-2-7 数据移位指令

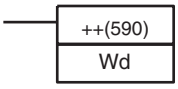
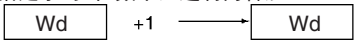
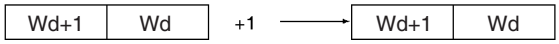
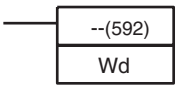
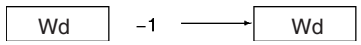
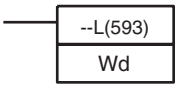
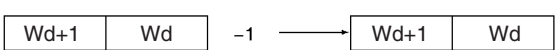
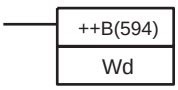
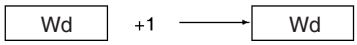
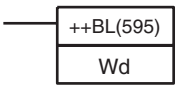
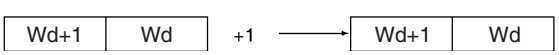
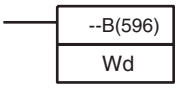
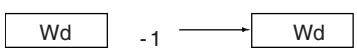
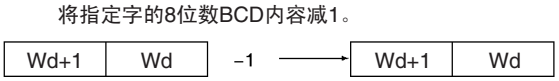
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
移位寄存器 SFT 010	数据输入 移位输入 复位输入 SFT(010) St E St:起始字 E:结束字	执行一移位寄存器。  丢失 每次移位输入时的数据输入的状态。	输出 需要	309
可逆移位寄存器 SFTR @SFTR 084	SFTR(084) C St E C:控制字 St:起始字 E:结束字	生成一个可左、可右的移位寄存器。 	输出 需要	310
异步移位寄存器 ASFT @ASFT 017	ASFT(017) C St E C:控制字 St:起始字 E:结束字	在指定字范围内朝St或E方向移动非零字， 以取代0000Hex字数据。  	输出 需要	313
字移位 WSFT @WSFT 016	WSFT(016) S St E S:源字 St:起始字 E:结束字	将St和E之间的数据按字移位。 	输出 需要	316
算术左移 ASL @ASL 025	ASL(025) Wd Wd:字	把Wd的内容左移一位。 	输出 需要	317

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
双字左移 ASLL @ASLL 570	ASLL(570) Wd Wd:字	把Wd和Wd+1的内容左移一位。 	输出 需要	319
算术右移 ASR @ASR 026	ASR(026) Wd Wd:字	将Wd的内容右移一位。 	输出 需要	321
双字右移 ASRL @ASRL 571	ASRL(571) Wd Wd:字	将Wd和Wd+1的内容右移一位。 	输出 需要	322
循环左移 ROL @ROL 027	ROL(027) Wd Wd:字	将Wd中的各位包括进行标志(CY)左移一位。 	输出 需要	324
双字循环左移 ROLL @ROLL 572	ROLL(572) Wd Wd:字	将Wd和Wd+1中的各位包括进位标志(CY)左移一位。 	输出 需要	326
无进位循环左移 RLNC @RLNC 574	RLNC(574) Wd Wd:字	将Wd中的各位不包括进位标志(CY)左移一位。 	输出 需要	331
无进位双字循环左移 RLNL @RLNL 576	RLNL(576) Wd Wd:字	将Wd和Wd+1中的各位不包括进位标志(CY)左移一位。 	输出 需要	332
循环右移 ROR @ROR 028	ROR(028) Wd Wd:字	将Wd中的各位包括进位标志(CY)右移一位。 	输出 需要	327
双字循环右移 RORL @RORL 573	RORL(573) Wd Wd:字	将Wd和Wd+1中的各位包括进位标志(CY)右移一位。 	输出 需要	329

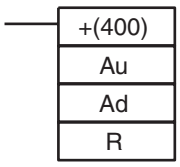
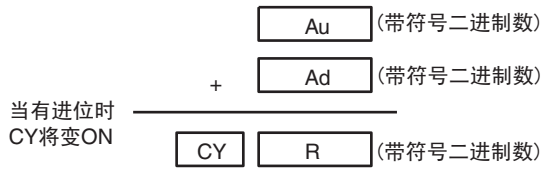
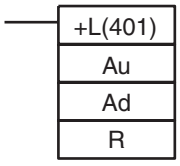
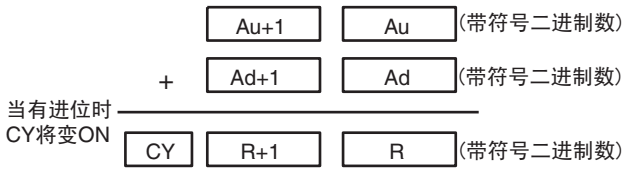
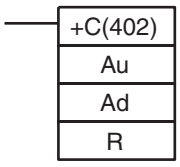
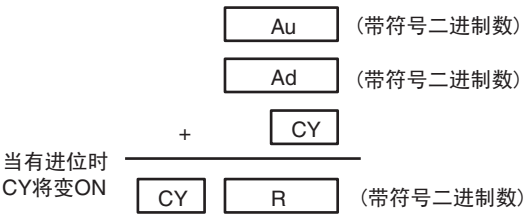
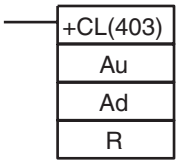
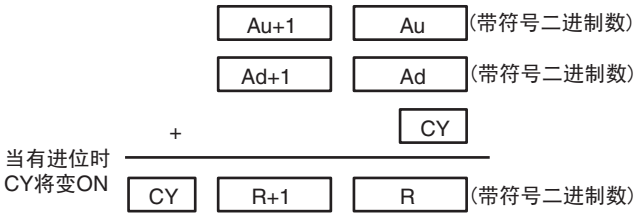
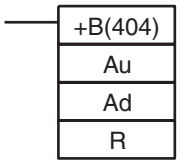
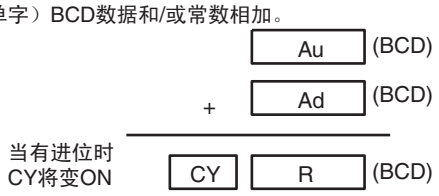
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
无进位循环右移 RRNC @RRNC 575	RRNC(575) Wd Wd:字	把Wd中各位不包括进位标志(CY)右移一位。 Wd的最右位的内容移至最左位和进位标志(CY)。 	输出 需要	334
无进位双字循环右移 RRNL @RRNL 577	RRNL(577) Wd Wd:字	把Wd和Wd+1中的各位不包括进位标志(CY)右移一位。 把Wd的最右位的内容移至Wd+1最左位和进位标志(CY)。 	输出 需要	336
一个数左移 SLD @SLD 074	SLD(074) St E St:起始字 E:结束字	把数据左移一个数字位（4位）。 	输出 需要	338
一个数右移 SRD @SRD 075	SRD(075) St E St:起始字 E:结束字	把数据右移一个数字位（4位）。 	输出 需要	339
N 位数据左移 NSFL @NSFL 578	NSFL(578) D C N D:移位起始字 C:起始位 N:移位数据长度	向左移动指定的位数。 	输出 需要	341
N 位数据右移 NSFR @NSFR 579	NSFR(579) D C N D:移位起始字 C:起始位 N:移位数据长度	向右移动指定的位数。 	输出 需要	343

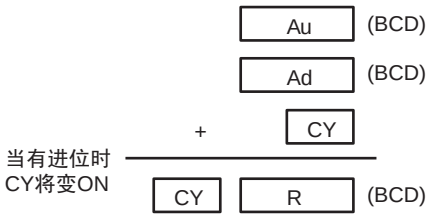
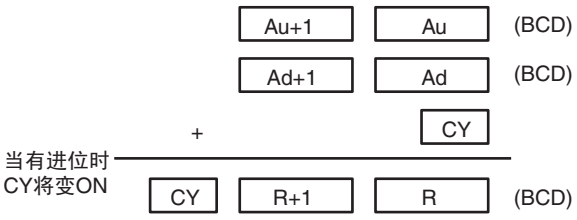
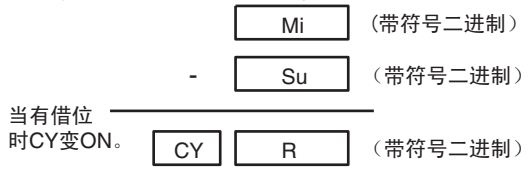
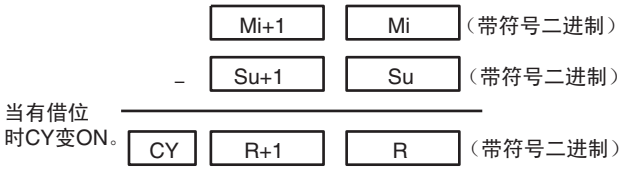
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
左移 N 位 NASL @NASL 580	NASL(580) D C D:移位字 C:控制字	将指定的16位数据左移指定的位数。 	输出 需要	345
双字左移 N 位 NSLL @NSLL 582	NSLL(582) D C D:移位字 C:控制字	将指定的32位数据左移指定的位数。 	输出 需要	348
右移 N 位 NASR @NASR 581	NASR(581) D C D:移位字 C:控制字	将指定的16位数据右移指定的位数。 	输出 需要	350
双字右移 N 位 NSRL @NSRL 583	NSRL(583) D C D:移位字 C:控制字	将指定的32位数据右移指定的位数。 	输出 需要	353

2-2-8 递增 / 递减指令

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
二进制递增 ++ @++ 590	 Wd:字	将指定字的4位数十六进制内容加1。 	输出 需要	356
双字二进制递增 ++L @++L 591	 Wd:字	将指定字的8位数十六进制内容加1。 	输出 需要	358
二进制递减 – – @– – 592	 Wd:字	将指定字的4位数十六进制内容减1。 	输出 需要	360
双字二进制递减 – –L @– –L 593	 Wd:首字	将指定字的8位数十六进制内容减1。 	输出 需要	362
BCD 递增 ++B @++B 594	 Wd:字	将指定字的4位数BCD内容加1。 	输出 需要	364
双字 BCD 递增 ++BL @++BL 595	 Wd:首字	将指定字的8位数BCD内容加1。 	输出 需要	366
BCD 递减 – –B @– –B 596	 Wd:字	将指定字的4位数BCD内容减1。 	输出 需要	368
双字 BCD 递减 – –BL @– –BL 597	 Wd:首字	将指定字的8位数BCD内容减1。 	输出 需要	370

2-2-9 四则运算指令

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
无进位带符号二进制加法 + @+ 400	 Au:被加字 Ad:加字 R:结果字	4位数（单字）十六进制数据和/或常数相加。 	输出 需要	373
无进位带符号双字二进制加法 +L @+L 401	 Au:被加首字 Ad:加首字 R:结果首字	8位数（双字）十六进制数据和/或常数相加。 	输出 需要	375
有进位带符号二进制加法 +C @+C 402	 Au:被加字 Ad:加字 R:结果字	4位数（单字）十六进制数和/或常数相及进位标志(CY)加。 	输出 需要	377
有进位带符号双字二进制加法 +CL @+CL 403	 Au:被加首字 Ad:加首字 R:结果首字	8位数（双字）十六进制数和/或常数相及进位标志(CY)加。 	输出 需要	379
无进位 BCD 加法 +B @+B 404	 Au:被加字 Ad:加字 R:结果字	4位数（单字）BCD数据和/或常数相加。 	输出 需要	381

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
无进位双字 BCD 加法 +BL @+BL 405	+BL(405) Au Ad R Au:被加首字 Ad:加首字 R:结果字	8位数（双字）BCD数据和/或常数相加。 	输出 需要	382
有进位 BCD 加法 +BC @+BC 406	+BC(406) Au Ad R Au:被加字 Ad:加字 R:结果字	4位数（单字）BCD数据和/或常数及进位标志（CY）相加。 	输出 需要	384
有进位双字 BCD 加法 +BCL @+BCL 407	+BCL(407) Au Ad R Au:被加首字 Ad:加首字 R:结果首字	8位数（双字）BCD数据和/或常数及进位标志（CY）相加。 	输出 需要	386
无进位带符号二进制减法 – @– 410	-(410) Mi Su R Mi:被减字 Su:减字 R:结果字	4位数（单字）十六进制数据和/或常数相减。 	输出 需要	387
无进位带符号双字二进制减法 –L @–L 411	-L(411) Mi Su R Mi:被减首字 Su:减首字 R:结果首字	8位数（双字）十六进制数据和/或常数相减。 	输出 需要	389

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
有进位带符号二进制减法 –C @–C 412	-C(412) Mi Su R Mi:被减字 Su:减字 R:结果字	4位数（单字）十六进制数据和/或常数及进位标志（CY）相减。 Mi （带符号二进制） Su （带符号二进制） – CY 当有借位时CY变ON。 CY R （带符号二进制）	输出 需要	393
有进位带符号双字二进制减法 –CL @–CL 413	-CL(413) Mi Su R Mi:被减首字 Su:减首字 R:结果首字	8位数（双字）十六进制数据和/或常数及进位标志（CY）相减。 Mi+1 Mi （带符号二进制） Su+1 Su （带符号二进制） – CY 当有借位时CY变ON。 CY R+1 R （带符号二进制）	输出 需要	395
无进位 BCD 减法 –B @–B 414	-B(414) Mi Su R Mi:被减字 Su:减字 R:结果字	4位数（单字）BCD数据和/或常数相减。 Mi (BCD) - Su (BCD) 当有借位时CY变ON。 CY R (BCD)	输出 需要	398
无进位双字 BCD 减法 –BL @–BL 415	-BL(415) Mi Su R Mi:被减首字 Su:减首字 R:结果首字	8位数（双字）BCD数据和/或常数相减。 Mi +1 Mi (BCD) – Su+1 Su (BCD) 当有借位时CY变ON。 CY R+1 R (BCD)	输出 需要	399
有进位 BCD 减法 –BC @–BC 416	-BC(416) Mi Su R Mi:被减字 Su:减字 R:结果字	4位数（单字）BCD数据和/或常数及进位标志（CY）相减。 Mi (BCD) Su (BCD) – CY 当有借位时CY变ON。 CY R (BCD)	输出 需要	403

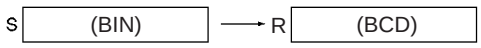
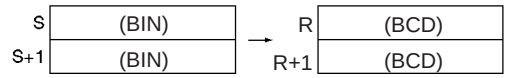
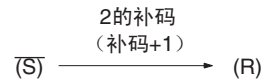
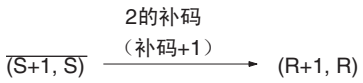
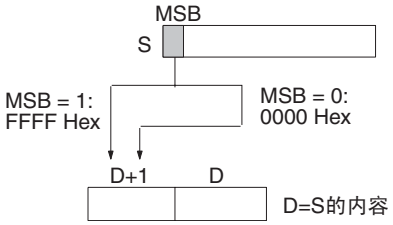
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
有进位双字 BCD 减法 −BCL @−BCL 417	-BCL(417) Mi Su R Mi:起始被减字 Su:起始减字 R:结果起始字	8位数（双字）BCD数据和/或常数及进位标志（CY）相减。 Mi +1 Mi (BCD) Su+1 Su (BCD) − CY 当有借位时CY变ON。 CY R+1 R (BCD)	输出需要	404
带符号二进制乘法 * @* 420	*(420) Md Mr R Md:被乘字 Mr:乘字 R:结果字	4位数带符号十六进制数据和/或常数相乘。 Md （带符号二进制） × Mr （带符号二进制） R +1 R （带符号二进制）	输出需要	406
带符号双字二进制乘法 *L @*L 421	*L(421) Md Mr R Md:被乘数首字 Mr:乘数首字 R:结果首字	8位数带符号十六进制数据和/或常数相乘。 Md + 1 Md （带符号二进制） × Mr + 1 Mr （带符号二进制） R + 3 R + 2 R + 1 R （带符号二进制）	输出需要	408
不带符号二进制乘法 *U @*U 422	*U(422) Md Mr R Md:被乘字 Mr:乘字 R:结果字	4位数不带符号十六进制数据和/或常数相乘。 Md （不带符号二进制） × Mr （不带符号二进制） R +1 R （不带符号二进制）	输出需要	410
带符号双字二进制乘法 *UL @*UL 423	*UL(423) Md Mr R Md:被乘数首字 Mr:乘数首字 R:结果首字	8位数不带符号十六进制数据和/或常数相乘。 Md + 1 Md （不带符号二进制） × Mr + 1 Mr （不带符号二进制） R + 3 R + 2 R + 1 R （不带符号二进制）	输出需要	412

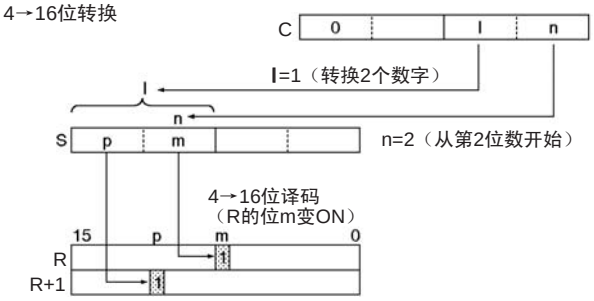
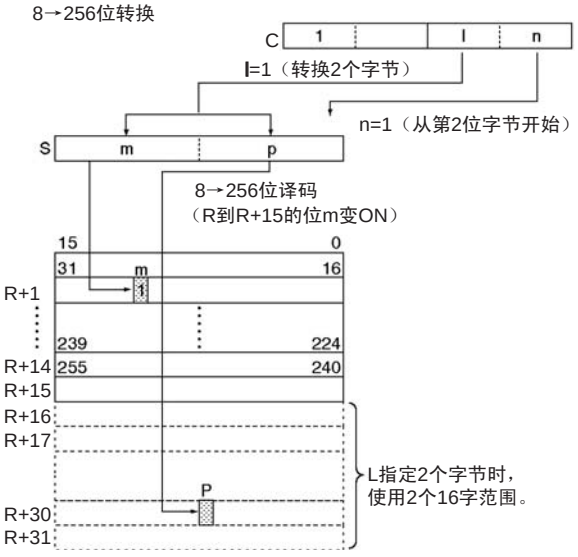
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
BCD 码乘法 *B @*B 424	*B(424) Md Mr R Md:被乘字 Mr:乘字 R:结果字	4位数（单字）BCD数据和/或常数相乘。 Md (BCD) × Mr (BCD) R +1 | R (BCD)	输出 需要	413
双字 BCD 码乘法 *BL @*BL 425	*BL(425) Md Mr R Md:被乘数首字 Mr:乘数首字 R:结果首字	8位数（双字）BCD数据和/或常数相乘。 Md + 1 | Md (BCD) × Mr + 1 | Mr (BCD) R + 3 | R + 2 | R + 1 | R (BCD)	输出 需要	415
带符号二进制除法 / @/ 430	/(430) Dd Dr R Dd:被除字 Dr:除字 R:结果字	4位数（单字）带符号十六进制数据和/或常数相除。 Dd （带符号二进制） ÷ Dr （带符号二进制） R +1 | R （带符号二进制） 余数 商	输出 需要	417
带符号双字二进制除法 /L @/L 431	/L(431) Dd Dr R Dd:被除数首字 Dr:除数首字 R:结果首字	8位数（双字）带符号十六进制数据和/或常数相除。 Dd + 1 | Dd （带符号二进制） ÷ Dr + 1 | Dr （带符号二进制） R + 3 | R + 2 | R + 1 | R （带符号二进制） 余数 商	输出 需要	419
不带符号二进制除法 /U @/U 432	/U(432) Dd Dr R Dd:被除字 Dr:除字 R:结果字	4位数（单字）不带符号十六进制数据和/或常数相除。 Dd （不带符号二进制） ÷ Dr （不带符号二进制） R +1 | R （不带符号二进制） 余数 商	输出 需要	421

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
不带符号双字二进制除法 /UL @/UL 433	/UL(433) Dd Dr R Dd:被除数首字 Dr:除数首字 R:结果首字	8位数（双字）不带符号十六进制数据和/或常数相除。 Dd + 1 | Dd （不带符号二进制） ÷ Dr + 1 | Dr （不带符号二进制） R + 3 | R + 2 | R + 1 | R （不带符号二进制） 余数 商	输出 需要	423
BCD 码除法 /B @/B 434	/B(434) Dd Dr R Dd:被除字 Dr:除字 R:结果字	4位数（单字）BCD数据和/或常数相除。 Dd (BCD) ÷ Dr (BCD) R +1 | R (BCD) 余数 商	输出 需要	425
双字 BCD 码除法 /BL @/BL 435	/BL(435) Dd Dr R Dd:被除数首字 Dr:除数首字 R:结果首字	8位数（双字）BCD数据和/或常数相除。 Dd + 1 | Dd (BCD) ÷ Dr + 1 | Dr (BCD) R + 3 | R + 2 | R + 1 | R (BCD) 余数 商	输出 需要	427

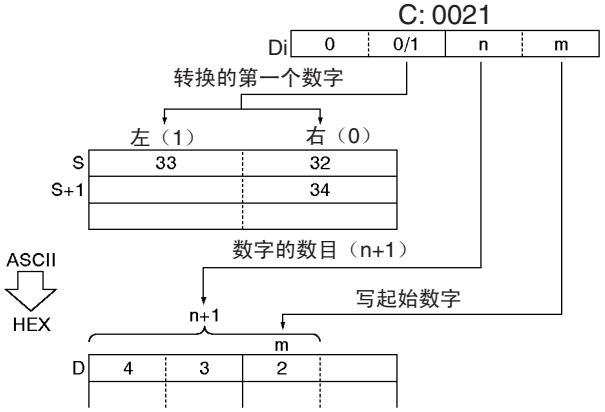
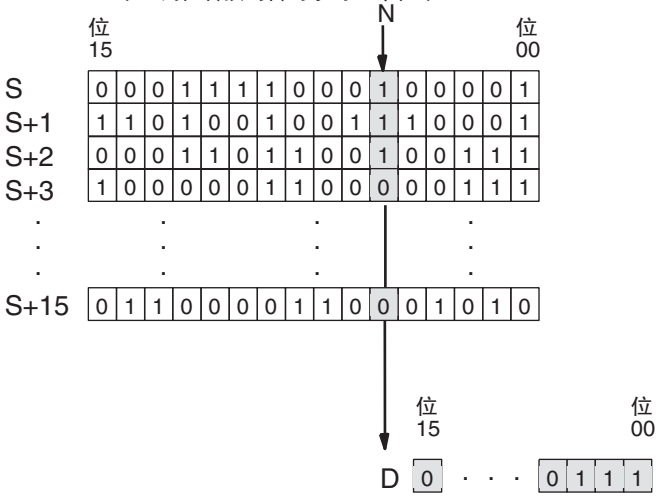
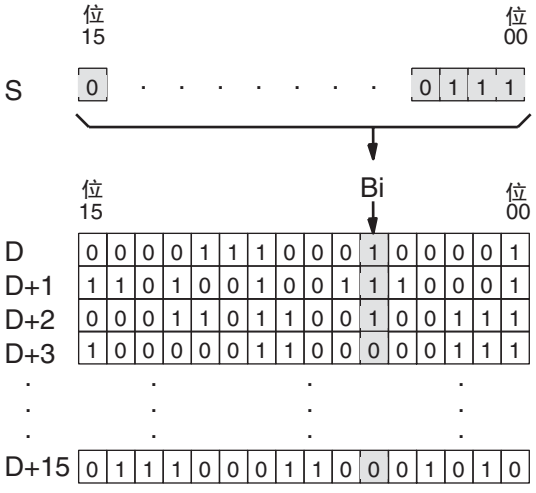
2-2-10 转换指令

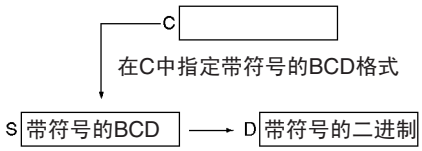
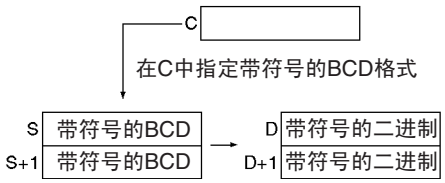
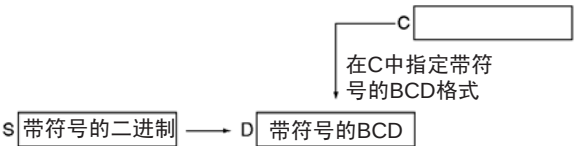
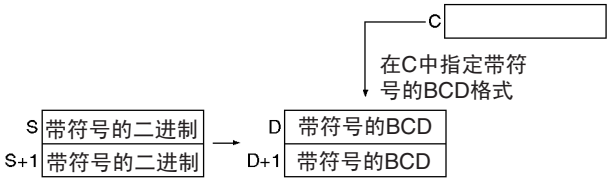
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
BCD →二进制 BIN @BIN 023	BIN(023) S R S:源字 R:结果字	把BCD数据转换成二进制数据。 S (BCD) → R (BIN)	输出 需要	429
双字 BCD →二进制 BINL @BINL 058	BINL(058) S R S:源首字 R:结果首字	8位数BCD数据转换成8位数十六进制数据（32位二进制数据）。 S (BCD) → R (BIN) S+1 (BCD) → R+1 (BIN)	输出 需要	430

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
二进制→ BCD BCD @BCD 024	BCD(024) S R S:源字 R:结果字	把一个字的二进制数据转换成一个字的BCD数据。 	输出 需要	432
双字二进制→ BCD BCDL @BCDL 059	BCDL(059) S R S:源首字 R:结果首字	把8位数十六进制（32位二进制）数据转换成8位数BCD数据。 	输出 需要	433
二进制求补 NEG @NEG 160	NEG(160) S R S:源字 R:结果字	计算一个字的十六进制数据2的补码。 	输出 需要	435
双字二进制求补 NEGL @NEGL 161	NEGL(161) S R S:源首字 R:结果首字	计算两个字的十六进制数据2的补码。 	输出 需要	437
带符号 16 位→ 32 位二进制 SIGN @SIGN 600	SIGN(600) S R S:源字 R:结果起始字	把一个带符号的16位二进制值扩展为等同的32位。 	输出 需要	439

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
数据译码 MLPX @MLPX 076	MLPX(076) S C R S:源字 C:控制字 R:结果首字	读源字中指定数（或字节）的数字值，把结果字（或16字范围）中的相应位变ON，并且把结果字（或16字范围）中的所有其他位变OFF。 4→16位转换  8→256位转换  	输出 需要	440

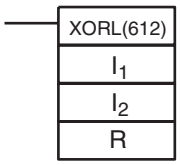
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
数据编码 DMPX @DMPX 077	DMPX(077) S R C S:源首字 R:结果字 C:控制字	在源字（16字范围）中寻找第一个或者最后一个ON位的位置，并将该值写到结果字中指定的数字（字节）。 16→4位转换 C 0 I/0 I n 寻找最左位 （最高位地址） 15 p n 0 S S+1 I=1（转换2个字） 16→4位译码 （最左位m写到R中） 最左位 最右位 n=2（从第2位字节开始） n R p m 256→8位转换 C 0 I/0 I n I=0(转换一个16字范围) 15 0 s 31 16 S+1 239 m 224 s+14 255 240 s+15 最左位 寻找最左位 （最高位地址） 256→8位译码 （最左位m写到R中） n=1（从第1位字节开始） R m	输出 需要	445
ASCII 码转换 ASC @ASC 086	ASC(086) S Di D S:源字 Di:数字指定器 D:目的首字	把源字中的4位十六进制数转换成相应的8位ASCII码。 Di 0 I/0 n m 转换的第一个数字 m S 1 2 3 HEX 数字的数目（n+1） ASCII 左（1） 右（0） D 33 31 32	输出 需要	449

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
ASCII 码转换→十六进制 HEX @HEX 162	HEX(162) S Di D S:源首字 Di:数字指定器 D:目的首字	把源字中的最多4字节的ASCII数据转换成相应的十六进制数，并将这些数字写入指定的目的字中。 	输出 需要	453
列→行 LINE @LINE 063	LINE(063) S N D S:源首字 N:位号 D:目的字	把16个字范围（16个连续字中的相同位号）中一列位转换到目的字的16个位中。 	输出 需要	457
行→列 COLM @COLM 064	COLM(064) S D N S:源字 D:目的字 N:位号	把源字中的16位转换到16个字范围的目的字的某列位中（16个连续字中的相同位号）。 	输出 需要	459

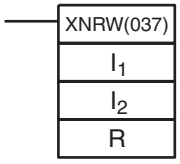
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
带符号的 BCD → 二进制 BINS @BINS 470	BINS(470) C S D C:控制字 S:源字 D:目的字	把一个字的带符号BCD数据转换成一个字的带符号的二进制数据。 	输出 需要	462
带符号的双字 BCD →二进制 BISL @BISL 472	BISL(472) C S D C:控制字 S:源首字 D:目的首字	把双字带符号BCD数据转换成双字带符号的二进制数据。 	输出 需要	465
带符号的二进制→ BCD BCDS @BCDS 471	BCDS(471) C S D C:控制字 S:源字 D:目的首字	把一个字的带符号二进制数据转换成一个字的带符号的BCD数据。 	输出 需要	468
带符号的双字二进制→ BCD BDSL @BDSL 473	BDSL(473) C S D C:控制字 S:源首字 D:目的首字	把双字带符号二进制数据转换成双字带符号的BCD数据。 	输出 需要	470

2-2-11 逻辑指令

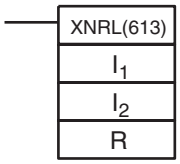
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
逻辑与 ANDW @ANDW 034	ANDW(034) I_1 I_2 R I_1:输入1 I_2:输入2 R:结果字	对一个单字数据和单字/或常数的相应位作一逻辑与运算。 $I_1 . I_2 \rightarrow R$ I_1 I_2 R 1 1 1 1 0 0 0 1 0 0 0 0	输出 需要	474
双字逻辑与 ANDL @ANDL 610	ANDL(610) I_1 I_2 R I_1:输入1 I_2:输入2 R:结果字	对一个双字数据和双字/或常数的相应位作一逻辑与运算。 $(I_1, I_1+1) . (I_2, I_2+1) \rightarrow (R, R+1)$ I_1, I_1+1 I_2, I_2+1 R, R+1 1 1 1 1 0 0 0 1 0 0 0 0	输出 需要	476
逻辑或 ORW @ORW 035	ORW(035) I_1 I_2 R I_1:输入1 I_2:输入2 R:结果字	对一个单字数据和单字/或常数的相应位作一逻辑或运算。 $I_1 + I_2 \rightarrow R$ I_1 I_2 R 1 1 1 1 0 1 0 1 1 0 0 0	输出 需要	477
双字逻辑或 ORWL @ORWL 611	ORWL(611) I_1 I_2 R I_1:输入1 I_2:输入2 R:结果字	对一个双字数据和双字/或常数的相应位作一逻辑或运算。 $(I_1,I_1+1) + (I_2,I_2+1) \rightarrow (R, R+1)$ I_1,I_1+1 I_2,I_2+1 R, R+1 1 1 1 1 0 1 0 1 1 0 0 0	输出 需要	479
异或 XORW @XORW 036	XORW(036) I_1 I_2 R I_1:输入1 I_2:输入2 R:结果字	对一个单字数据和单字/或常数的相应位作一异或逻辑运算。 $I_1.\overline{T_2} + \overline{T_1}.I_2 \rightarrow R$ I_1 I_2 R 1 1 0 1 0 1 0 1 1 0 0 0	输出 需要	481

<table>
<tr><th>指令
助记符
代码</th><th>符号 / 操作数</th><th>功能</th><th>位置
执行条件</th><th>页号</th></tr>
<tr><td>双字异或
XORL
@XORL
612</td><td> 

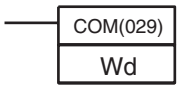
I1:输入1
I2:输入2
R:结果字</td><td>对一个双字数据和双字/或常数的相应位作一逻辑异或运算。

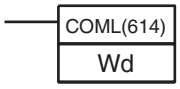
$(I_1.I_1+1).\ (\overline{I_2.I_2+1}) + (\overline{I_1.I_1+1}).\ (I_2.I_2+1) \rightarrow (R, R+1)$
<table>
<tr><th>$I_1.I_1+1$</th><th>$I_2.I_2+1$</th><th>R, R+1</th></tr>
<tr><td>1</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>1</td></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
</table></td><td>输出
需要</td><td>483</td></tr>
<tr><td>异或非
XNRW
@XNRW
037</td><td> 

I1:输入1
I2:输入2
R:结果字</td><td>对一个单字数据和单字/或常数的相应位作一逻辑异或非运算。

$I_1.I_2+\overline{I_1}.\overline{I_2} \rightarrow R$
<table>
<tr><th>I_1</th><th>I_2</th><th>R</th></tr>
<tr><td>1</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>1</td></tr>
</table></td><td>输出
需要</td><td>485</td></tr>
<tr><td>双字异或非
XNRL
@XNRL
613</td><td> 

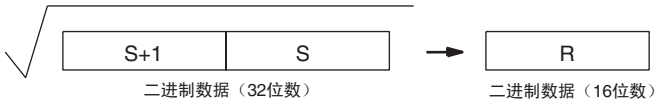
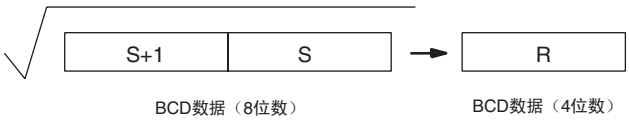
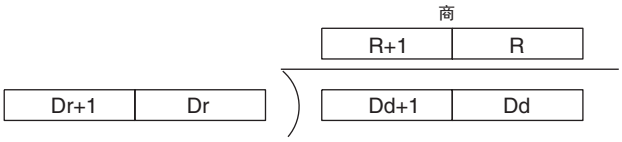
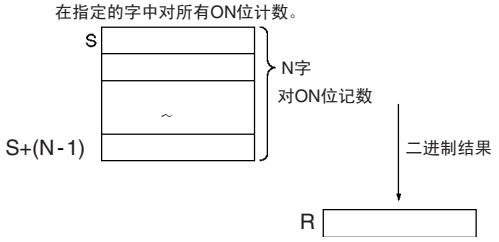
I1:输入1
I2:输入2
R:结果字</td><td>对一个双字数据和双字/或常数的相应位作一逻辑异或非运算。

$(I_1.I_1+1).\ (I_2.I_2+1) + (\overline{I_1.I_1+1}).\ (\overline{I_2.I_2+1}) \rightarrow (R, R+1)$
<table>
<tr><th>$I_1.I_1+1$</th><th>$I_2.I_2+1$</th><th>R, R+1</th></tr>
<tr><td>1</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>1</td></tr>
</table></td><td>输出
需要</td><td>486</td></tr>
<tr><td>求反
COM
@COM
029</td><td> 

Wd:字</td><td>把字中所有ON位变OFF，而把所有OFF位变ON。
$\overline{Wd} \rightarrow$Wd: 1→0和0→1</td><td>输出
需要</td><td>488</td></tr>
<tr><td>双字求反
COML
@COML
614</td><td> 

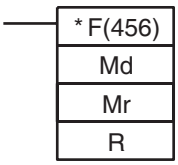
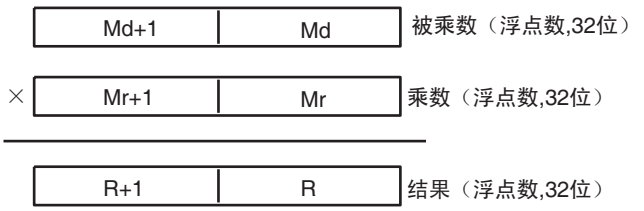
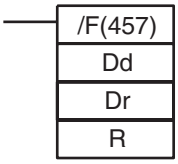
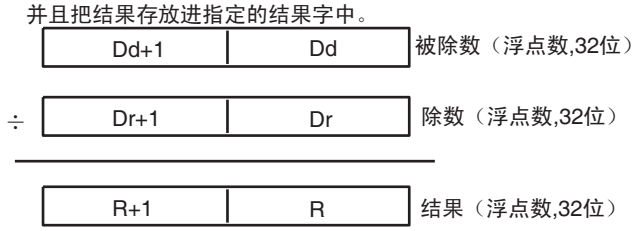
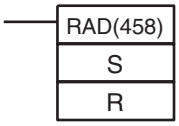
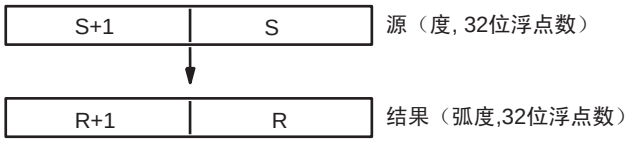
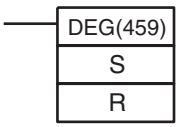
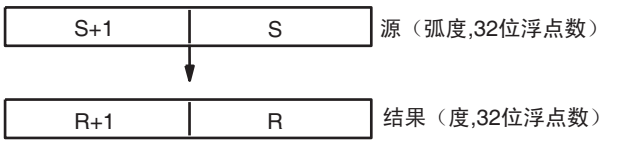
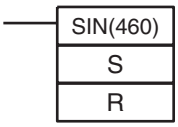
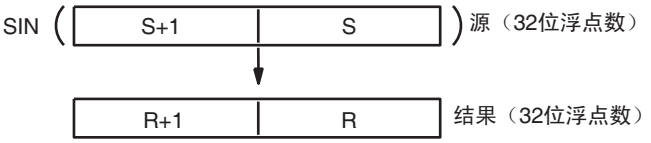
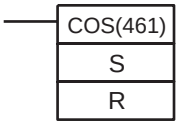
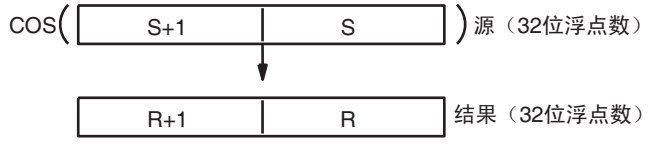
Wd:字</td><td>把字Wd和Wd+1中所有ON位变OFF，而把所有OFF位变ON。
$\overline{(Wd+1, Wd)} \rightarrow (Wd+1, Wd)$</td><td>输出
需要</td><td>490</td></tr>
</table>

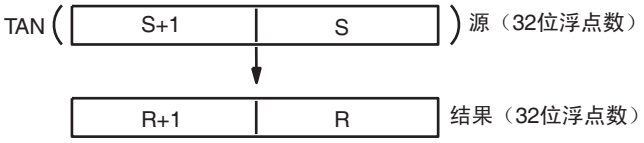
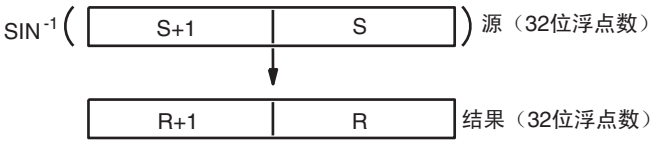
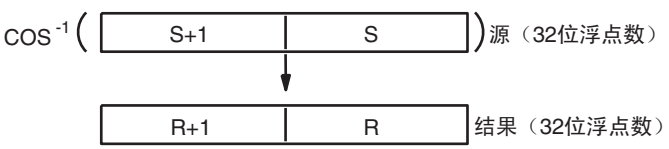
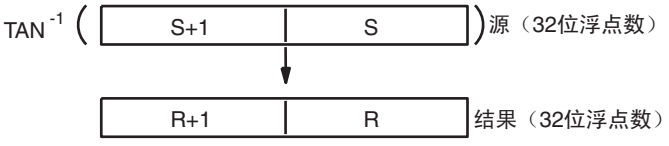
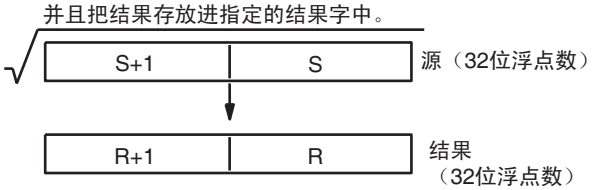
2-2-12 特殊算术指令

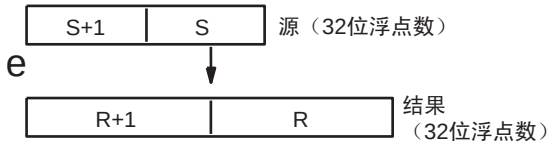
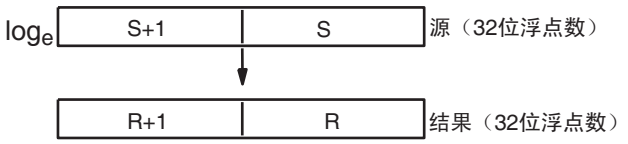
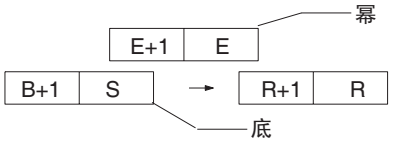
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
二进制平方根 ROTB @ROTB 620	ROTB(620) S R S:源首字 R:结果字	算出指定字的32位二进制数的平方根，并且把结果的整数部分输出到指定结果字中。 	输出 需要	491
BCD 平方根 ROOT @ROOT 072	ROOT(072) S R S:源首字 R:结果字	求出8位数BCD码的平方根，并且把结果的整数部分输出到指定结果字中。 	输出 需要	493
数学处理 APR @APR 069	APR(069) C S R C:控制字 S:源首字 R:结果字	计算源数据的正弦、余弦或线性逼近。 线性逼近功能允许 X， Y 之间的关系用线段来模拟。	输出 需要	497
浮点数除法 FDIV @FDIV 079	FDIV(079) Dd Dr R Dd:被除数首字 S:除数首字 R:结果首字	一个7位浮点数和另一个浮点数相除。 浮点数用科学法表示（7位尾数和1位指数）。 	输出 需要	509
位计数器 BCNT @BCNT 067	BCNT(067) N S R N:字数 S:源首字 R:结果字	在指定的字中对所有ON位计数。 	输出 需要	513

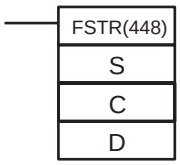
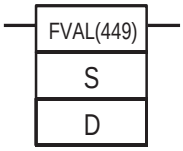
2-2-13 浮点数运算指令

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
浮点数 → 16 位 FIX @FIX 450	FIX(450) S R S:源首字 R:结果字	把一个32位浮点数据转换成带符号的16位二进制数据， 并且把结果存放进指定结果字中。 S+1 | S 浮点数据（32位） ↓ R 带符号二进制数据（16位）	输出 需要	520
浮点数 → 32 位 FIXL @FIXL 451	FIXL(451) S R S:源首字 R:结果字	把一个32位浮点数据转换成带符号的32位二进制数据， 并且把结果存放进指定结果字中。 S+1 | S 浮点数据（32位） ↓ R+1 | R 带符号二进制数据（32位）	输出 需要	522
16 位 →浮点数 FLT @FLT 452	FLT(452) S R S:源字 R:结果首字	把带符号的16位二进制数据转换成一个32位浮点数据， 并且把结果存放进指定结果字中。 S 带符号二进制数据（16位） ↓ R+1 | R 浮点数据（32位）	输出 需要	523
32 位→浮点数 FLTL @FLTL 453	FLTL(453) S R S:源首字 R:结果首字	把带符号的32位二进制数据转换成一个32位浮点数据， 并且把结果存放进指定结果字中。 S+1 | S 带符号二进制数据（32位） ↓ R+1 | R 浮点数据（32位）	输出 需要	525
浮点数加法 +F @+F 454	+F(454) Au Ad R Au:被加数首字 AD:加数首字 R:结果首字	把两个32位浮点数相加，并且把结果存放进指定的结果字中。 Au+1 | Au 被加数（浮点数，32位） + Ad+1 | Ad 加数（浮点数，32位） R+1 | R 结果（浮点数，32位）	输出 需要	527
浮点数减法 –F @–F 455	-F(455) Mi Su R Mi:被减数首字 Su:减数首字 R:结果首字	把两个32位浮点数相减，并且把结果存放进指定的结果字中。 Mi+1 | Mi 被减数（浮点数，32位） - Su+1 | Su 减数（浮点数，32位） R+1 | R 结果（浮点数，32位）	输出 需要	529

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
浮点数乘法 *F @*F 456	 Md:被乘数首字 Mr:乘数首字 R:结果首字	把二个32位浮点数据相乘，并且把结果存放进指定的结果字中。 	输出 需要	531
浮点数除法 /F @/F 457	 Dd:被除数首字 Dr:除数首字 R:结果首字	把一个32位浮点数据去除另一个32位浮点数据， 并且把结果存放进指定的结果字中。 	输出 需要	533
度 →弧度 RAD @RAD 458	 S:源首字 R:结果首字	把一个32位浮点数从度变为弧度， 并且把结果存放进指定的结果字中。 	输出 需要	535
弧度→度 DEG @DEG 459	 S:源首字 R:结果首字	把一个32位浮点数从弧度变为度， 并且把结果存放进指定的结果字中。 	输出 需要	536
正弦 SIN @SIN 460	 S:源首字 R:结果首字	求出一个32位浮点数（用弧度表示）的正弦， 并且把结果存放进指定的结果字中。 	输出 需要	538
余弦 COS @COS 461	 S:源首字 R:结果首字	求出一个32位浮点数（用弧度表示）的余弦， 并且把结果存放进指定的结果字中。 	输出 需要	540

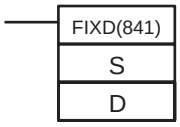
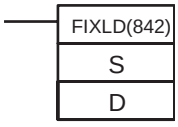
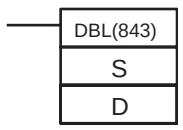
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
正切 TAN @TAN 462	TAN(462) S R S:源首字 R:结果首字	求出一个32位浮点数（用弧度表示）的正切， 并且把结果存放进指定的结果字中。 	输出 需要	542
反正弦 ASIN @ASIN 463	ASIN(463) S R S:源首字 R:结果首字	求出一个32位浮点数的反正弦值，并且把结果存放进指定 的结果字中。（反正弦函数与正弦函数正好相反，它返回 一个角度值，该角度的正弦值在-1和+1之间） 	输出 需要	544
反余弦 ACOS @ACOS 464	ACOS(464) S R S:源首字 R:结果首字	求出一个32位浮点数的反余弦值，并且把结果存放进指定的 结果字中。（反余弦函数与余弦函数正好相反，它返回一个 角度值，该角度的余弦值在-1和+1之间） 	输出 需要	546
反正切 ATAN @ATAN 465	ATAN(465) S R S:源首字 R:结果首字	求出一个32位浮点数的反正切值，并且把结果存放进指定的 结果字中。（反正切函数与正切函数正好相反，它返回一个 角度值，该角度产生正切值） 	输出 需要	548
平方根 SQRT @SQRT 466	SQRT(466) S R S:源首字 R:结果首字	求出一个32位浮点数的平方根， 并且把结果存放进指定的结果字中。 	输出 需要	550

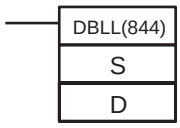
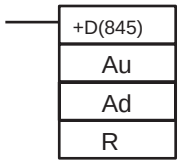
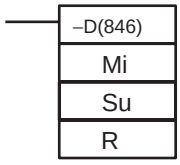
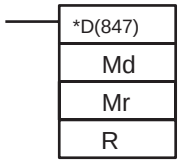
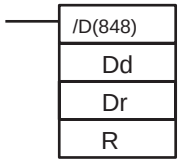
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
指数 EXP @EXP 467	EXP(467) S R S:源首字 R:结果首字	求一个32位浮点数的自然指数（底为e）， 并且把结果存放进指定的结果字中。 	输出 需要	552
对数 LOG @LOG 468	LOG(468) S R S:源首字 R:结果首字	求一个32位浮点数的自然对数（底为e）， 并且把结果存放进指定的结果字中。 	输出 需要	554
指数幂 PWR @PWR 840	PWR(840) B E R B:基础首字 E:指数首字 R:结果首字	自乘一个32位浮点数到另一个32位浮点数的幂。 	输出 需要	556
浮点符号比较 （仅适用于 CS1-H, CJ1-H, CJ1M 或 CS1D） LD, AND 或 OR + =F (329), <>F (330), <F (331), <=F (332), >F (333), 或 >=F (334)	使用LD: 符号，选项 S1 S2 使用AND: 符号，选项 S1 S2 使用OR: 符号，选项 S1 S2 S1:比较数据1 S2:比较数据2	比较指定的单精度数据 （32 位）或常数，并且在比较结果是 ON 时，产生一个 ON 执行条件。 浮点符号比较指令可用：LD （载入）， AND 和 OR。	LD：不需要 AND 或 OR： 需要	557

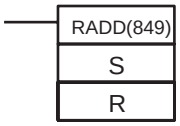
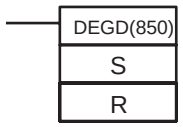
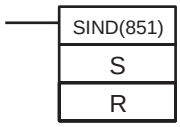
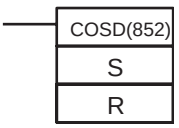
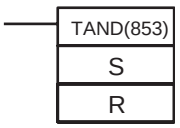
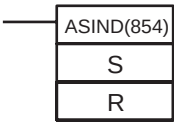
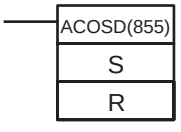
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
浮点数→ ASCII （仅适用于 CS1-H， CJ1-M 或 CS1D） FSTR @FSTR 448	 S:源首字 C:控制字 D:目的字	把指定格式 （32 位小数点或指数格式）的单精度浮点数据转换成字符串数据 (ASCII)，并且把结果存入目的字中。	输出 需要	561
ASCII →浮点数 （仅适用于 CS1-H,CJ1-M 或 CS1D） FVAL @FVAL 449	 S:源字 D:目的首字	把指定格式 （小数点或指数格式）表达的单精度浮点数数据字符串数据 (ASCII) 转换成 32 位单精度浮点数数据，并且把结果存入目的字中。	输出 需要	566

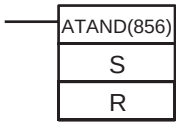
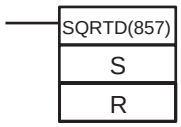
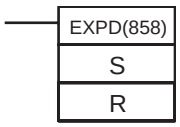
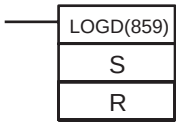
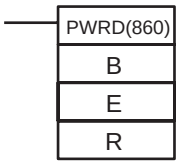
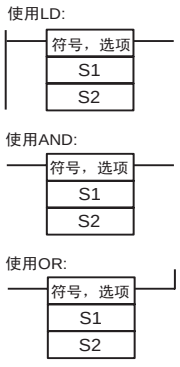
2-2-14 双精度浮点数指令

仅适用于 CS1-H， CJ1-H， CJ1M 或 CS1D CPU 单元。

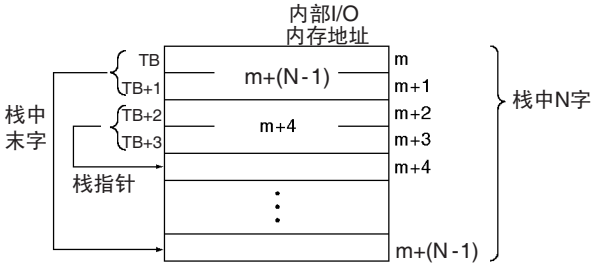
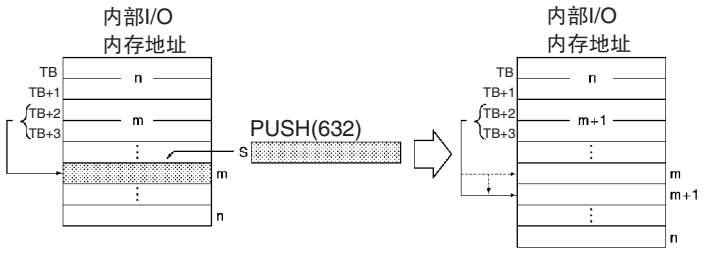
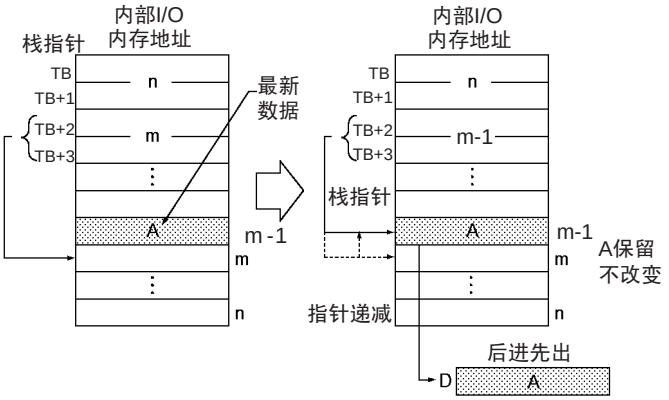
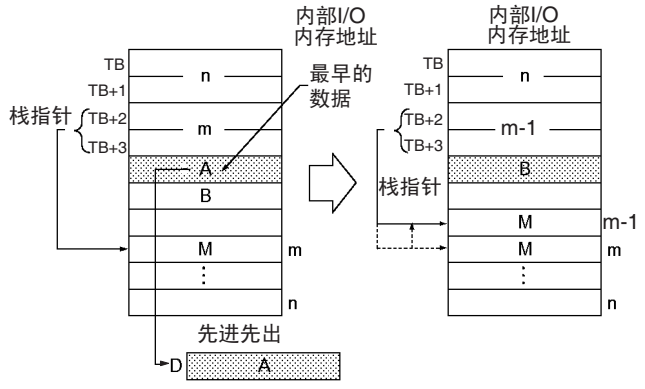
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
双字长浮点数→ 16 位二进制 FIXD @FIXD 841	 S:源首字 D:目的字	把一个指定的双精度浮点数据 （64 位）转换成 16 位二进制数据，并且把结果存入到目的字中。	输出 需要	577
双字长浮点数→ 32 位二进制 FIXLD @FIXLD 842	 S:源首字 D:目的字	把一个指定的双精度浮点数据 （64 位）转换成 32 位带符号二进制数据，并且把结果存入到目的字中。	输出 需要	578
16 位二进制 →双字长浮点数 DBL @DBL 843	 S:源字 D:目的首字	把一个指定的 16 位带符号二进制数据转换成双精度浮点数据 （64 位），并且把结果存入到目的字中。	输出 需要	580

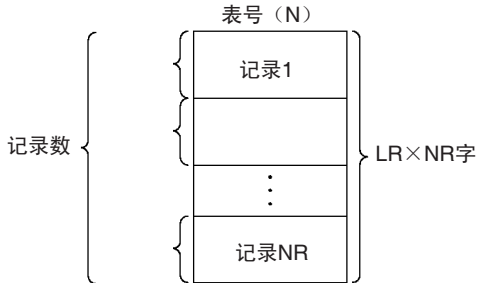
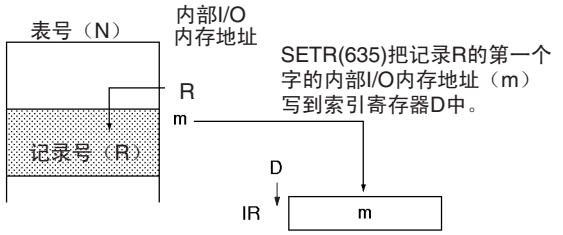
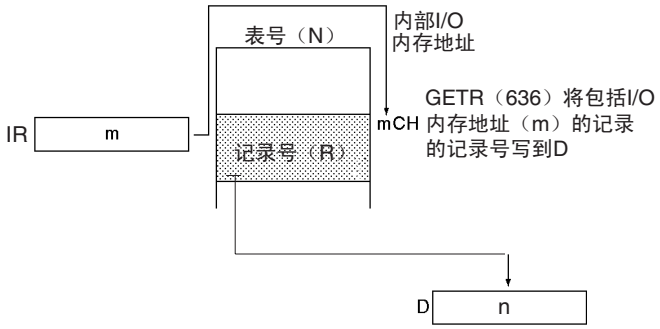
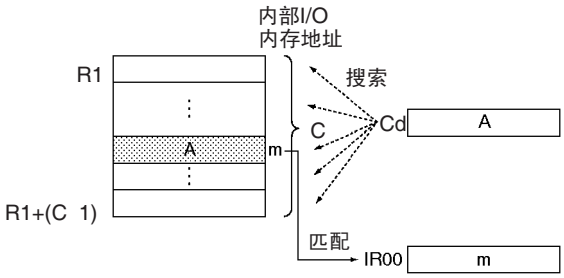
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
32 位二进制→双字长浮点数 DBLL @DBLL 844	 S:源首字 D:目的首字	把一个指定的 32 位带符号二进制数据转换成双精度浮点数据 （64 位），并且把结果存入到目的字中。	输出 需要	581
双字长浮点数加 +D @+D 845	 Au:被加数首字 Ad:加数首字 R:结果首字	把两个指定的双精度浮点数值 （每个数均为 64 位）相加，并且把结果存入到目的字中。	输出 需要	583
双字长浮点数减 –D @–D 846	 Mi:被减数首字 Su:减数首字 R:结果首字	把两个指定的双精度浮点数值 （每个数均为 64 位）相减，并且把结果存入到目的字中。	输出 需要	585
双字长浮点数乘 *D @*D 847	 Md:被乘数首字 Mr:乘数首字 R:结果首字	把两个指定的双精度浮点数值 （每个数均为 64 位）相乘，并且把结果存入到目的字中。	输出 需要	587
双字长浮点数除 /D @/D 848	 Dd:被除数首字 Dr:除数首字 R:结果首字	把两个指定的双精度浮点数值 （每个数均为 64 位）相除，并且把结果存入到目的字中。	输出 需要	589

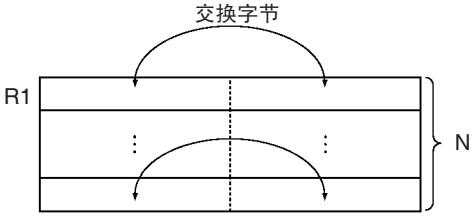
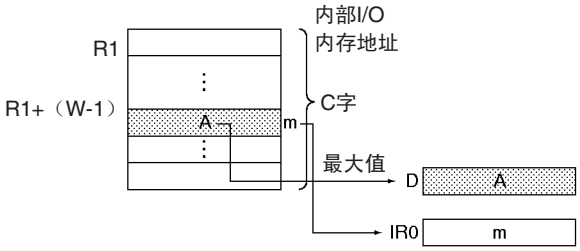
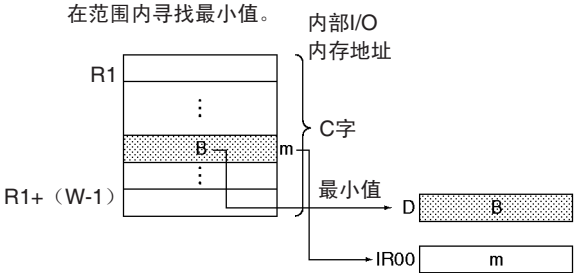
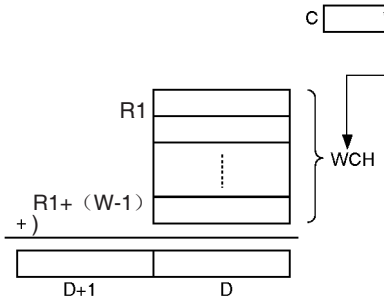
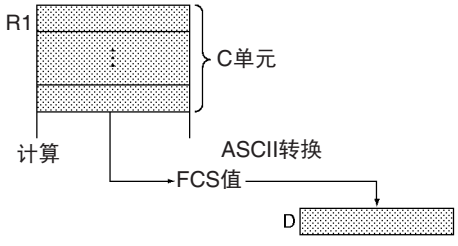
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
双字长度→弧度 RADD @RADD 849	 S:源首字 R:结果首字	把一个指定的双精度浮点数 （64 位）从度变为弧度，并且把结果存入到目的字中。	输出 需要	591
双字长弧度→度 DEGD @DEGD 850	 S:源首字 R:结果首字	把一个指定的双精度浮点数据 （64 位）从弧度变为度，并且把结果存入到目的字中。	输出 需要	593
双字长正弦 SIND @SIND 851	 S:源首字 R:结果首字	求出一个指定的双精度 （64 位）浮点数据 （用弧度表示）的正弦，并且把结果存入到目的字中。	输出 需要	594
双字长余弦 COSD @COSD 852	 S:源首字 R:结果首字	求出一个指定的双精度 （64 位）浮点数据 （用弧度表示）的余弦，并且把结果存入到目的字中。	输出 需要	596
双字长正切 TAND @TAND 853	 S:源首字 R:结果首字	求出一个指定的双精度 （64 位）浮点数据 （用弧度表示）的正切，并且把结果存入到目的字中。	输出 需要	598
双字长反正弦 ASIND @ASIND 854	 S:源首字 R:结果首字	求出一个指定的双精度 （64 位）浮点数据 （用弧度表示）的反正弦值，并且把结果存入到目的字中。（反正弦函数与正弦函数正好相反，它返回一个角度值，该角度的正弦值在 -1 和 +1 之间）。	输出 需要	600
双字长反余弦 ACOSD @ACOSD 855	 S:源首字 R:结果首字	求出一个指定的双精度 （64 位）浮点数据 （用弧度表示）的反余弦值，并且把结果存放到指定的结果字中。（反余弦函数与余弦函数正好相反，它返回一个角度值，该角度的余弦值在 -1 和 +1 之间）。	输出 需要	602

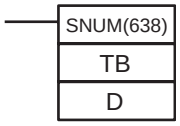
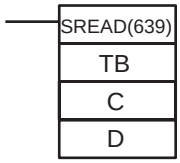
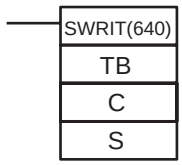
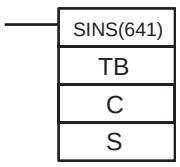
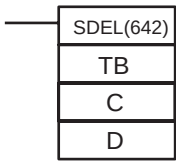
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
双字长反正切 ATAND @ATAND 856	 S:源首字 R:结果首字	求出一个指定的双精度 （64 位）浮点数据 （用弧度表示）的反正切值，并且把结果存入到目的字中。（反正切函数与正切函数正好相反，它返回一个角度值，该角度产生正切值）。	输出 需要	604
双字长平方根 SQRTD @SQRTD 857	 S:源首字 R:结果首字	求出一个指定的双精度 （64 位）浮点数据的平方根值，并且把结果存入到目的字中。	输出 需要	606
双字长指数 EXPD @EXPD 858	 S:源首字 R:结果首字	求出一个指定的双精度 （64 位）浮点数据的自然指数 （底为 e），并且把结果存入到目的字中。	输出 需要	608
双字长对数 LOGD @LOGD 859	 S:源首字 R:结果首字	求出一个指定的双精度 （64 位）浮点数据的自然对数 （底为 e），并且把结果存入到目的字中。	输出 需要	610
双字长指数幂 PWRD @PWRD 860	 B:基础首字 E:指数首字 R:结果首字	自乘一个指定的双精度 （64 位）浮点数到另一个指定的双精度 （64 位）浮点数，并且把结果存入到目的字中。	输出 需要	612
双字长符号比较 LD, AND 或 OR + =D (335), <>D (336), <D (337), <=D (338), >D (339), 或 >=D (340)	 S1:比较数据1 S2:比较数据2	比较指定的双精度数据 （64 位）并且在比较结果是 ON 时产生一个 ON 执行条件。 双精度浮点符号比较指令可用三种符号：LD （载入）， AND 和 OR。	LD：不需要 AND 或 OR： 需要	614

2-2-15 表格数据处理指令

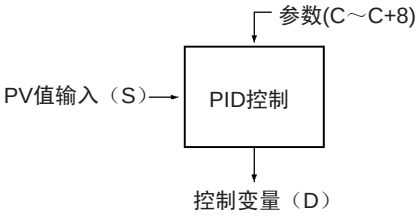
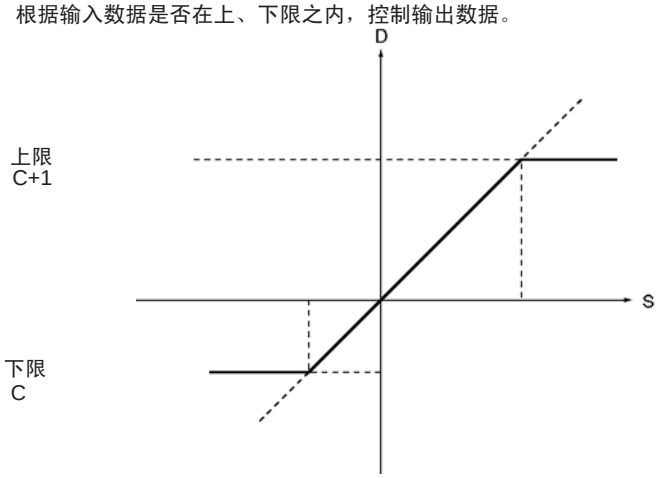
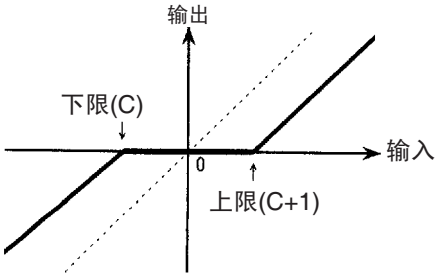
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
堆栈设置 SSET @SSET 630	SSET(630) TB N TB:栈首址 N:编号字	在指定字开始的指定长度中定义一个栈， 并初始化此数据区的字全为零。 	输出 需要	623
压入栈 PUSH @PUSH 632	PUSH(632) TB S TB:栈首址 S:源字	把一个字的数据写入指定栈。 	输出 需要	626
后进后出 LIFO @LIFO 634	LIFO(634) TB D TB:栈首址 D:目的字	读写指定栈中的最后一个数据（栈中最新数据）。 	输出 需要	632
先进先出 FIFO @FIFO 633	FIFO(633) TB D TB:栈首址 D:目的字	读写指定栈中的第一个数据（栈中最早的数据）。 	输出 需要	629

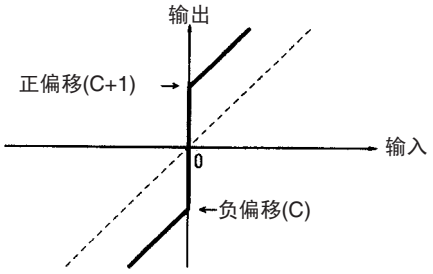
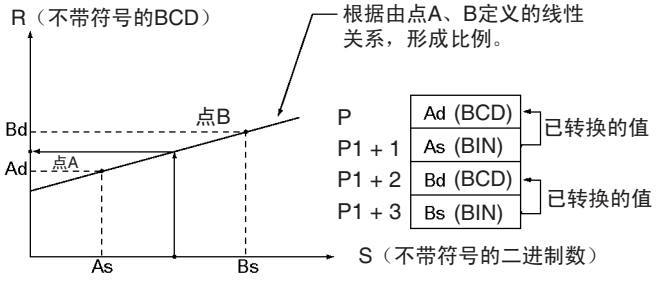
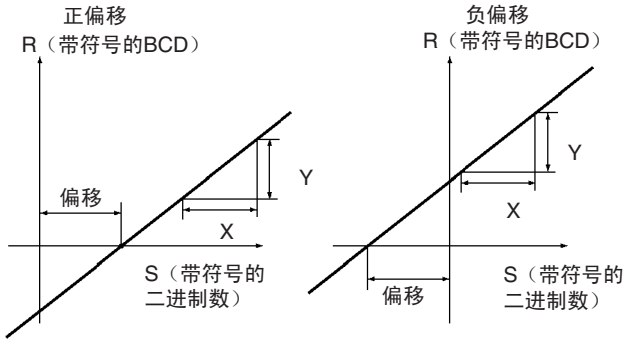
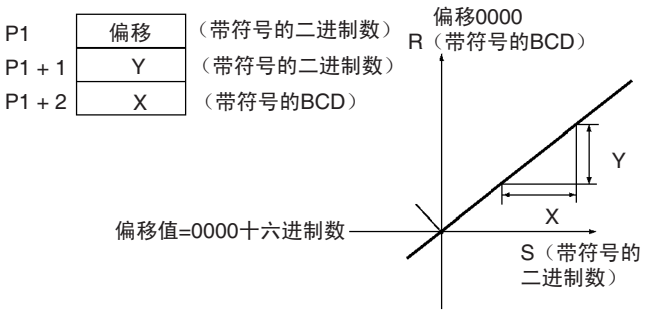
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
定义记录表 DIM @DIM 631	DIM(631) N LR NR TB N:表号 LR:每个记录长度 NR:记录数 TB:栈首址	用规定每个记录长度及记录数来定义记录表。 最多可定义16个记录表。 	输出 需要	635
设置记录位置 SETR @SETR 635	SETR(635) N R D N:表号 R:记录号 D:目的索引寄存器	把指定记录的位置（第一个记录的内部I/O内存地址）写进指定的索引寄存器中。 	输出 需要	638
获得记录号 GETR @GETR 636	GETR(636) N IR D N:表号 IR:索引寄存器 D:目的字	返回包含在指定索引寄存器内的内部I/O内存地址中记录的记录号。 	输出 需要	640
数据搜索 SRCH @SRCH 181	SRCH(181) C R1 Cd C:控制首字 R1:范围首字 Cd:比较数据	在指定字的范围内搜索一个字的数据。 	输出 需要	642

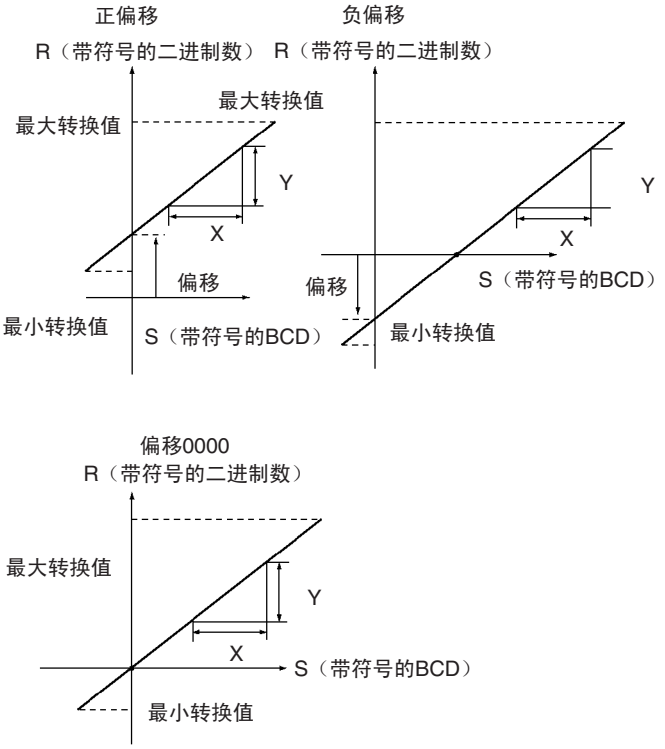
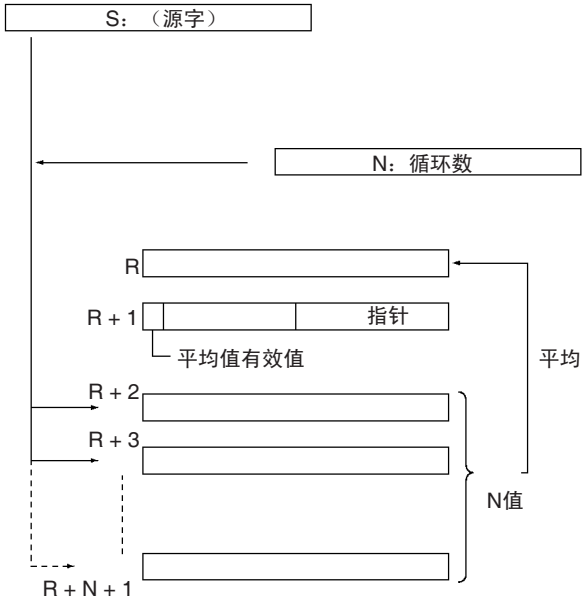
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
交换字节 SWAP @SWAP 637	SWAP(637) N R1 N:字数 R1:范围首字	将范围内的所有字的左字节和右字节交换。 	输出 需要	644
寻找最大值 MAX @MAX 182	MAX(182) C R1 D C:控制首字 R1:范围首字 D:目的字	在范围内寻找最大值。 	输出 需要	646
寻找最小值 MIN @MIN 183	MIN(183) C R1 D C:控制首字 R1:范围首字 D:目的字	在范围内寻找最小值。 	输出 需要	650
求和 SUM @SUM 184	SUM(184) C R1 D C:控制首字 R1:范围首字 D:目的字	把范围内的字节或字相加并把结果输出到2个结果字中。 	输出 需要	653
帧校验和 FCS @FCS 180	FCS(180) C R1 D C:控制首字 R1:范围首字 D:目的字	计算指定范围内的ASCII FCS值。 	输出 需要	656

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
读栈大小 （仅适用于 CS1-H, CJ1-H, CJ1M 或者 CS1D） SNUM @SNUM 638	 TB:栈首址 D:目的字	在指定的栈中计栈数据量（字数）。	输出 需要	659
读栈数据 （仅适用于 CS1-H, CJ1-H, CJ1M 或者 CS1D） SREAD @SREAD 639	 TB:栈首址 C:偏移值 D:目的字	从栈中指定的数据元素读数据。偏移值表明了所需数据元素的位置（在当前指针位置前有多少个数据元素）。	输出 需要	662
覆盖栈数据 （仅适用于 CS1-H, CJ1-H, CJ1M 或者 CS1D） SWRIT @SWRIT 640	 TB:栈首址 C:偏移值 D:目的字	把源数据写入栈中指定的数据元素中（覆盖原来的数据）。偏移值表明所需数据元素的位置（在当前指针位置前有多少个数据元素）。	输出 需要	665
插入栈数据 (仅适用于 CS1-H, CJ1-H, CJ1M 或者 CS1D) SINS @SINS 641	 TB:栈首址 C:偏移值 D:目的字	把源数据插入栈中指定的数据元素中，并将栈中其余的数据向下移。偏移值指明了插入点的位置（在当前指针位置前有多少个数据元素）。	输出 需要	668
删除栈数据 (仅适用于 CS1-H, CJ1-H, CJ1M 或者 CS1D) SDEL @SDEL 642	 TB:栈首址 C:偏移值 D:目的字	把栈中指定位置处的数据元素删除，并将栈中其他的数据向上移。偏移值指明了删除点的位置（在当前指针位置前有多少个数据元素）。	输出 需要	671

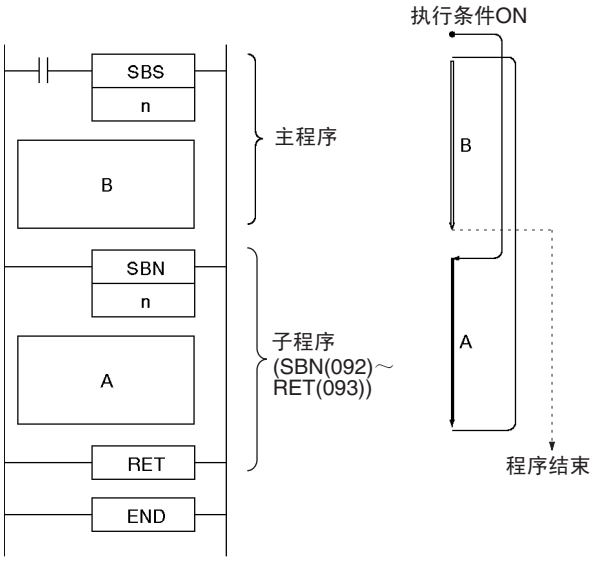
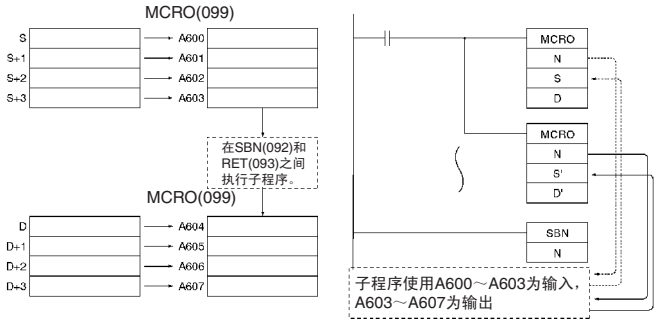
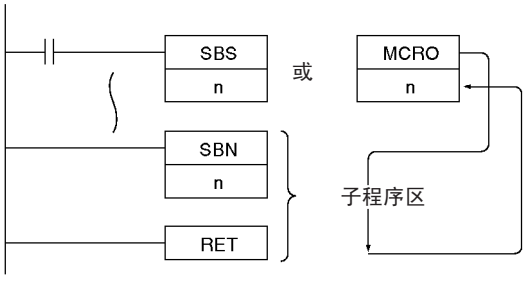
2-2-16 数据控制指令

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
PID 控制 PID 190	PID(190) S C D S:输入字 C:参数首字 D:输出字	根据指定参数进行PID控制。 	输出 需要	675
带自调整的 PID 控制 PIDAT 191 （仅适用于 CS1-H, CJ1-H 或者 CJ1M）	PIDAT(191) S C D S:输入字 C:参数首字 D:输出字	根据指定的参数执行 PID 控制。 PID 常数可用 PIDAT(191) 指令自动调整。	输出 需要	686
限位控制 LMT @LMT 680	LMT(680) S C D S:输入字 C:限位首字 D:输出字	根据输入数据是否在上、下限之内，控制输出数据。 	输出 需要	696
静带控制 BAND @BAND 681	BAND(681) S C D S:输入字 C:限位首字 D:输出字	根据输入数据是否在静带范围内控制输出数据。 	输出 需要	698

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
静域控制 ZONE @ZONE 682	ZONE(682) S C D S:输入字 C:参数首字 D:输出字	把指定的偏移值和输入数据相加并输出结果。 	输出 需要	701
标度 SCL @SCL 194	SCL(194) S P1 R S:源字 P1:参数首字 R:输出结果字	根据指定的线性关系，把不带符号的二进制数据转换成不带符号的BCD数据。 	输出 需要	704
标度 2 SCL2 @SCL2 486	SCL2(486) S P1 R S:源字 P1:参数首字 R:输出结果字	根据指定的线性关系，把带符号的二进制数据转换成带符号的BCD数据。 偏移值可在定义线性函数时输入。  	输出 需要	708

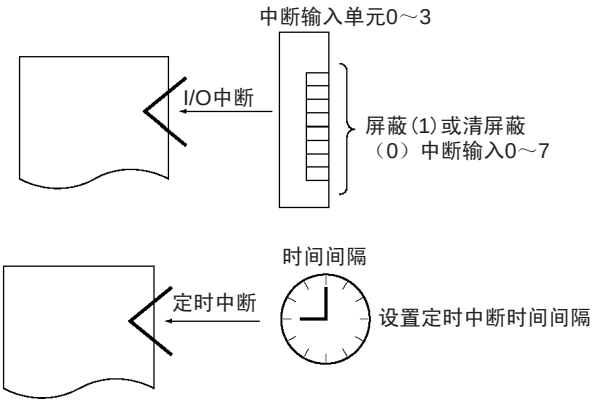
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
标度 3 SCL3 @SCL3 487	SCL3(487) S P1 R S:源字 P1:参数首字 R:结果字	根据指定的线性函数，把带符号的BCD数据转换成带符号的二进制数据。偏移值可在定义线性函数时输入。  	输出 需要	712
平均值 AVG 195	AVG(195) S N R S:源字 N:循环数 R:结果字	求出指定循环数目的输入字的平均值。 	输出 需要	716

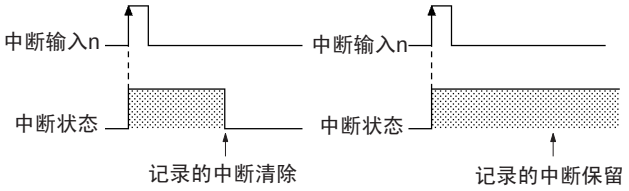
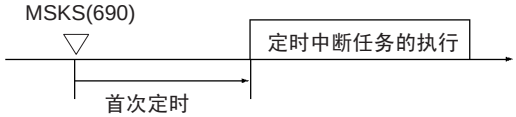
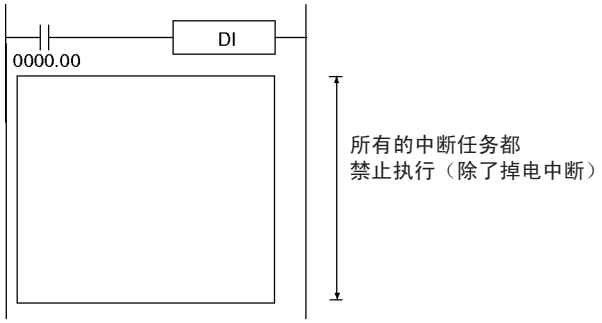
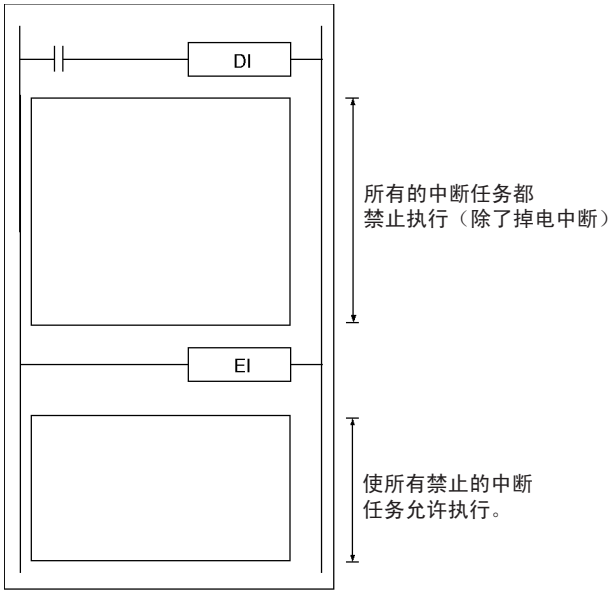
2-2-17 子程序指令

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
子程序调用 SBS @SBS 091	SBS(091) N N:子程序编号	调用指定子程序号的子程序并执行它。 	输出 需要	720
宏 MCRO @MCRO 099	MCRO(099) N S D N:子程序编号 S:输入参数首字 D:输出参数首字	调用指定子程序号的子程序，并使用S～S+3中的输入参数执行子程序，输出参数至D～D+3。 	输出 需要	725
子程序入口 SBN 092	SBN(092) N N:子程序编号	指示指定子程序号的子程序开始。 	输出 需要	729
子程序返回 RET 093	RET(093)	指示子程序结束。	输出 需要	732

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
全局子程序调用 （仅适用于 CS1-H, CJ1-H, CJ1M 或者 CS1D） GSBS 750	GSBS(750) N N:子程序编号	调用指定子程序号的子程序并执行它。	输出 需要	732
子程序入口 （仅适用于 CS1-H, CJ1-H, CJ1M 或者 CS1D） GSBN 751	GSBN(751) N N:子程序编号	指示指定子程序号的子程序开始。	输出 需要	740
全局子程序返回 （仅适用于 CS1-H, CJ1-H, CJ1M 或者 CS1D） GRET 752	GRET(752)	指示子程序结束。	输出 需要	743

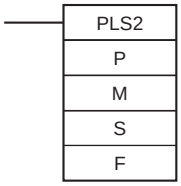
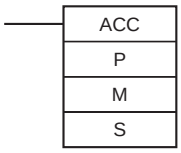
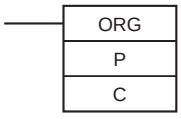
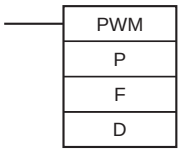
2-2-18 中断控制指令

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
设置中断屏蔽 （CS1D 不支持） MSKS @MSKS 690	MSKS(690) N S N:中断识别字 S:中断数据	对I/O中断或定时中断设置中断处理。在PC刚上电时，I/O中断和定时中断都被屏蔽（禁止）。MSKS(690)可用于清屏蔽或屏蔽I/O中断，及设置定时中断的时间间隔。 	输出 需要	744
读中断屏蔽 （CS1D 不支持） MSKR @MSKR 692	MSKR(692) N D N:中断识别字 D:目的字	读当前由 MSKS(690) 设定的中断处理设定。	输出 需要	750

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
清除中断 （CS1D 不支持） CLI @CLI 691	CLI(691) N S N:中断识别字 S:中断数据	对I/O中断或保留已记录的中断输入，或对定时中断设定首次定时中断的时间。 N = 0～3  N = 4～5 	输出 需要	755
禁止中断 DI @DI 693	DI(693)	除了掉电中断，所有的中断任务都禁止执行。 	输出 需要	760
允许中断 EI 694	EI(694)	允许执行所有由DI（693）禁止的中断任务。 	输出 不需要	762

2-2-19 高速计数器和脉冲输出指令 （仅适用于 CJ1M-CPU22/23）

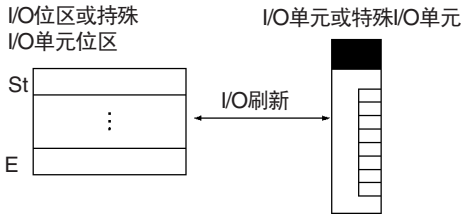
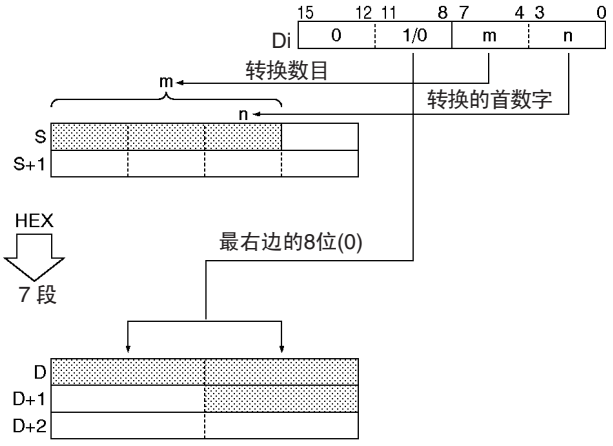
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
控制模式 INI @INI 880	INI P C NV P: 口定义 C: 控制字 NV: 输出最新 PV 值首字	INI(880) 指令用于启动和停止目标值比较，改变一个高速计数器的当前值 (PV)，改变中断输入 （计数器模式）的当前值 (PV)，改变一个脉冲输出的当前值 (PV) 或停止脉冲输出。	输出 需要	769
读高速计数器 PV PRV @PRV 881	PRV P C D P: 口定义 C: 控制数据 D: 目的首字	PRV(881) 指令用于读出一个高速计数器的当前值 (PV) 和脉冲输出或中断输入 （计数器模式）。	输出 需要	773
比较表写入 CTBL @CTBL 882	CTBL P C TB P: 口定义 C: 控制字 TB: 比较表首字	CTBL(882) 指令用于完成一个高速计数器的当前值 (PV) 对目标值或范围比较。	输出 需要	777
速度输出 SPED @SPED 885	SPED P M F P: 口定义 M: 输出模式 F: 脉冲频率首字	SPEED(885) 指令用于完成不带加 / 减速度的脉冲输出和频率设定。	输出 需要	781
脉冲设定 PULS @PULS 886	PULS P T N P: 口定义 T: 脉冲类型 N: 脉冲数	PULS(886) 指令用于设定脉冲输出的脉冲数。	输出 需要	786

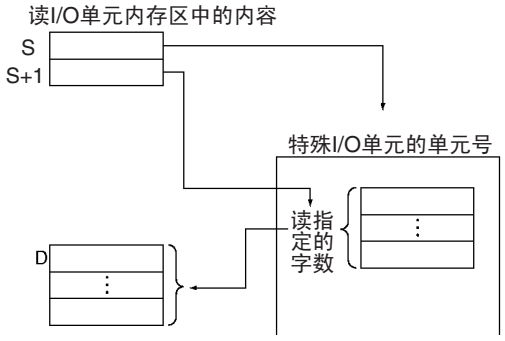
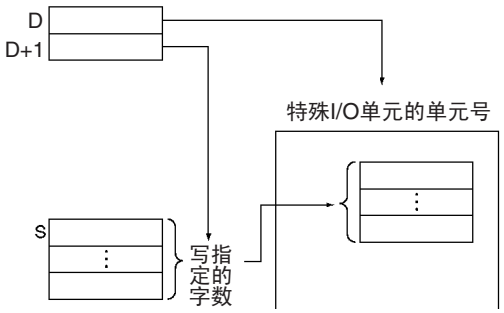
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
脉冲输出 PLS2 @PLS2 887	 P: 口定义 M: 输出模式 S: 设定表首字 F: 脉冲频率首字	PLS2(887) 指令用于设定脉冲频率和加速度 / 减速度，并完成带加速度 / 减速度 （不同加 / 减速度率）的脉冲输出，指令仅适用于定位控制。	输出 需要	789
加速度控制 ACC @ACC 888	 P: 口定义 M: 输出模式 S: 设定表首字	ACC(888) 指令用于设定脉冲频率和加速度 / 减速度，并完成带加速度 / 减速度 （相同加 / 减速度率）的脉冲输出，指令可用于定位控制和速度控制。	输出 需要	795
原点搜索 ORG @ORG 889	 P: 口定义 C: 控制数据	ORG(889) 用于完成原点搜索和返回。	输出 需要	802
带占空比可变的脉冲 PWM @ 891	 P: 口定义 F: 频率 D: 占空比	PWM(891) 用于输出占空比可变的脉冲。	输出 需要	805

2-2-20 步指令

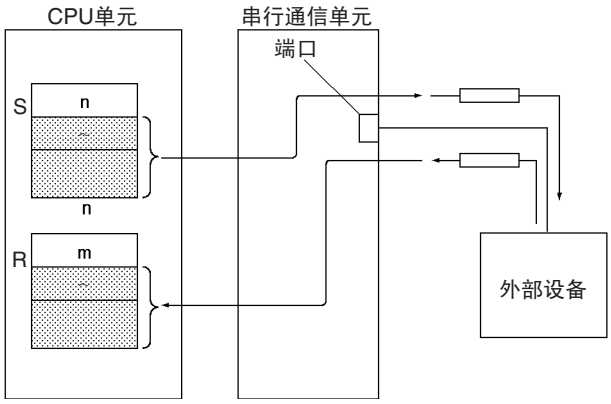
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
步定义 STEP 008	STEP(008) B B:位	STEP(008) 以下列 2 种方式作用 , 这取决于它的位置和控制位是否输出被指定。 (1) 开始一个指定的步。 (2) 结束步程序区 （例 : 步执行）。	输出 需要	808
步启动 SNXT 009	SNXT(009) B B:位	SNXT(0009) 用于下列三种情况 : (1) 开始步程序执行。 (2) 继续到下一个步控制位。 (3) 结束步程序执行。	输出 需要	808

2-2-21 基本 I/O 单元指令

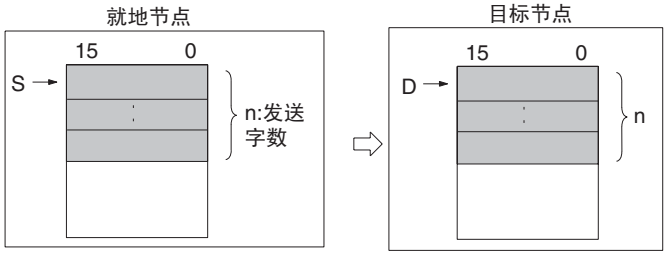
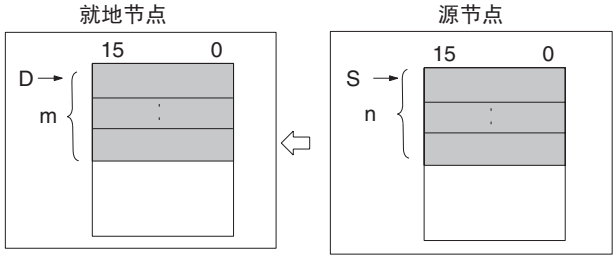
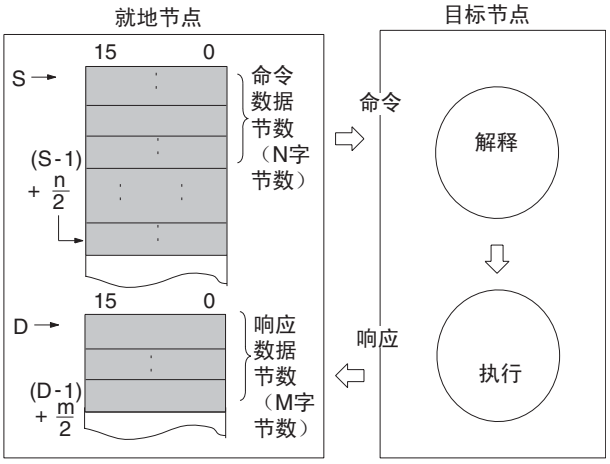
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
I/O 刷新 IORF @IORF 097	IORF(097) St E St:起始字 E:结束字	刷新指定的I/O字。 	输出 需要	825
7 段译码 SDEC @SDEC 078	SDEC(078) S Di D S:源字 Di:数字指定器 D:目标首字	把指定数字中的十六进制数转换成相应的8位、7段显示码, 并把它存入指定目标字中的高或低8位。 	输出 需要	828

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
智能 I/O 读 IORD @IORD 222	IORD(222) C S D C:控制数据 S:传送源和字数 D:传送目标 和字数	读I/O单元内存区中的内容 	输出 需要	831
智能 I/O 写 IOWR @IOWR 223	IOWR(223) C S D C:控制数据 S:传送源和字数 D:传送目标 和字数	把CPU单元I/O内存区的内容输出到特殊I/O单元的单元号 	输出 需要	834
CPU 总线 I/O 刷新 （仅 CS1-H, CJ1-H, CJ1M 或 CS1D） DLNK @DLNK 226	DLNK(226) N N:单元号	在 CPU 总线单元中，用指定单元数立即刷新 I/O。	输出 需要	837

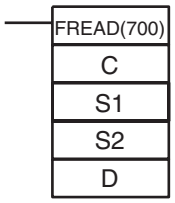
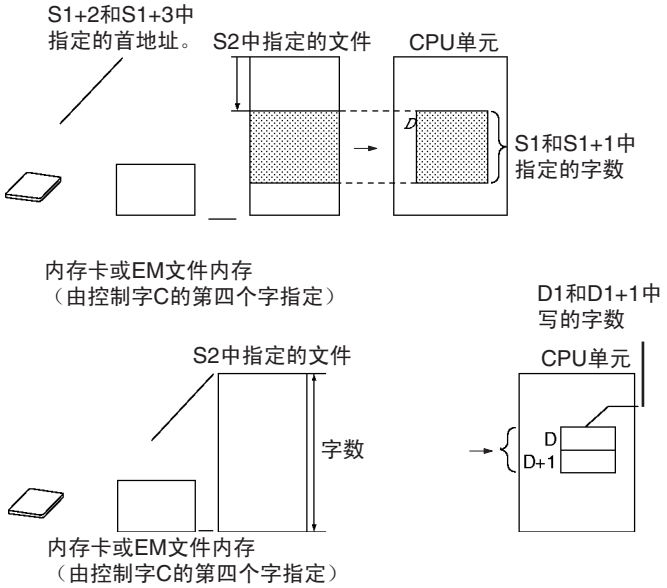
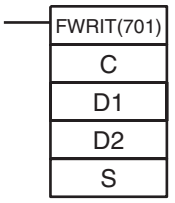
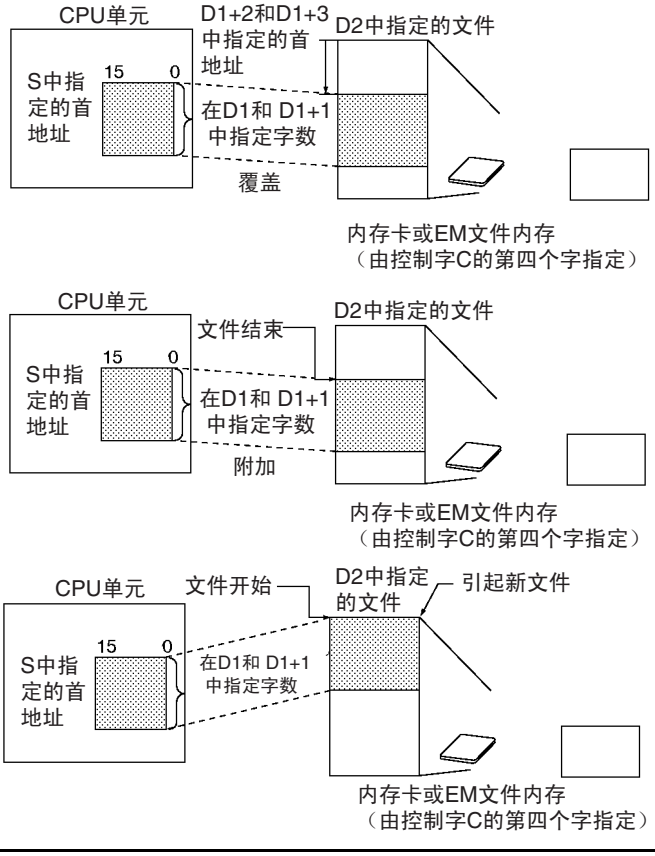
2-2-22 串行通信指令

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
协议宏 PMCR @PMCR 260	PMCR(260) C1 C2 S R C1:控制字1 C2:控制字2 S:发送首字 R:接收首字	调用和执行在串行通信板或串行通信单元中登录过的通信序列。 	输出 需要	844
发送 TXD @TXD 236	TXD(236) S C N S:源首字 C:控制字 N:字节数 0000～0100十六进制数（0～256十进制数）	从 CPU 单元内置的 RS-232C 端口把一指定的字节数的数据输出	输出 需要	853
接收 RXD @RXD 235	RXD(235) D C N D:源目标字 C:控制字 N:字节数 0000～0100十六进制数（0～256十进制数）	从 CPU 单元内置的 RS-232C 端口读指定字节数的数据。	输出 需要	858
修改串行口设置 STUP @STUP 237	STUP(237) C S C:控制字 （端口） S:源首字	修改 CPU 单元，串行通信单元 （CPU 总线单元），或串行通信板中串输出口的通信参数。在 PLC 操作中， STUP(237) 能使协议发生改变。	输出 需要	863

2-2-23 网络指令

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
网络发送 SEND @SEND 090	SEND(090) S D C S:源首字 D:目标首字 C:控制首字	把数据传送到网中的一个节点。 	输出 需要	879
网络接收 RECV @RECV 098	RECV(098) S D C S:源首字 D:目标首字 C:控制首字	要求网中的一个节点传送数据并接收此数据。 	输出 需要	885
发布命令 CMND @CMND 490	CMND(490) S D C S:源首字 D:目标首字 C:控制首字	发送FINS命令并接收响应。 	输出 需要	890

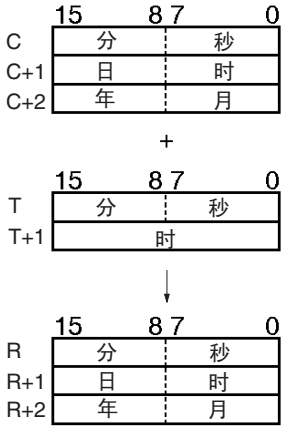
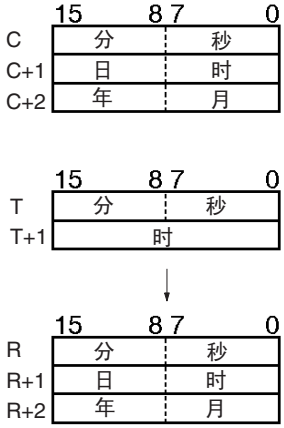
2-2-24 文件存储指令

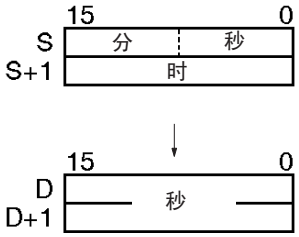
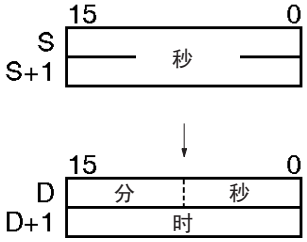
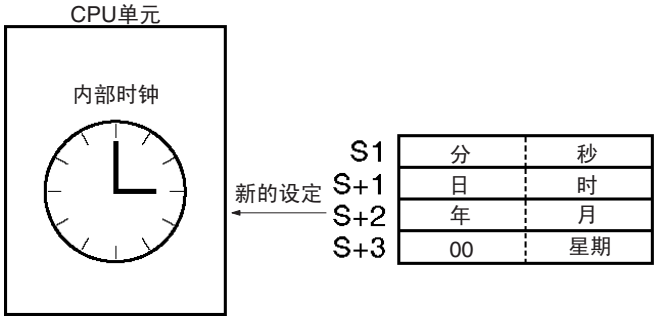
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
读数据文件 FREAD @FREAD 700	 C:控制字 S1:源首字 S2:文件名 D:目标首字	把指定的数据或数据量从文件内存的指定数据文件读到CPU单元的指定数据区中。 	输出 需要	899
写数据文件 FWRIT @FWRIT 701	 C: 控制字 D1:目标首字 D2:文件名 S: 源首字	要求网中的一个节点传送数据并接收此数据。 	输出 需要	906

2-2-25 显示指令

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
显示信息 MSG @MSG 046	MSG(046) N M N:信息号 M:信息首字	读指定的 16 个扩展 ASCII 字 , 并且在外围设备 （如编程器）上显示该输出信息。	输出 需要	913

2-2-26 时钟指令

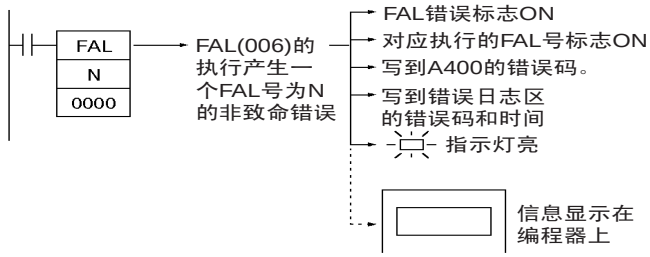
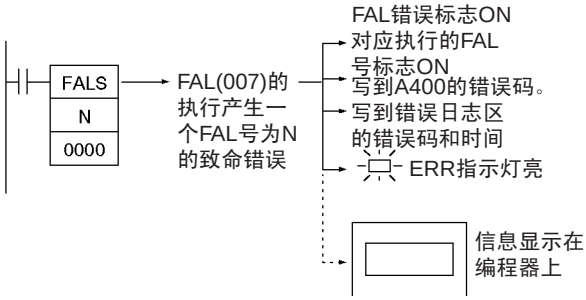
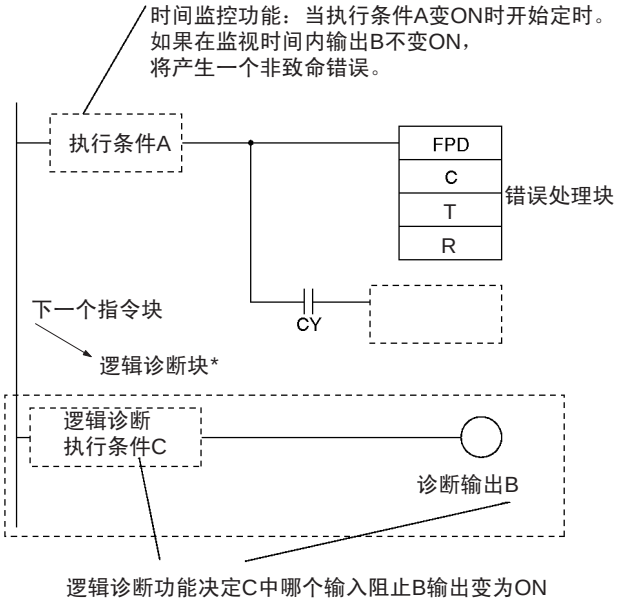
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
日历加法 CADD @CADD 730	CADD(730) C T R C:日历首字 T:时间首字 R:结果首字	把指定字中的日历数据和时间相加。 	输出 需要	916
日历减法 CSUB @CSUB 731	CSUB(731) C T R C:日历首字 T:时间首字 R:结果首字	把指定字中的日历数据和时间相减。 	输出 需要	920

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
小时→秒 SEC @SEC 065	SEC(065) S D S:源首字 D:目标首字	将以小时/分/秒表示的时间转换成仅以秒表示的等值时间 	输出 需要	923
秒→小时 HMS @HMS 066	HMS(066) S D S:源首字 D:目标首字	将以秒表示的时间转换成以 小时/分/秒表示的等值时间 	输出 需要	925
时钟调整 DATE @DATE 735	DATE(735) S S:源首字	改变内部时钟设定为指定源字中的设定 	输出 需要	928

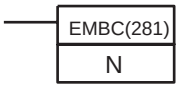
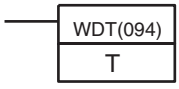
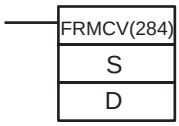
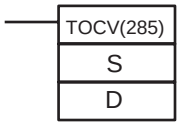
2-2-27 调试指令

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
跟踪内存采样 TRSM 045	TRSM(045)	当执行 TRSM(045) 时，预定位或字的状态被采样并存于跟踪内存中。TRSM(045) 可用于程序中的任何地方，且使用任意次。	输出 不需要	930

2-2-28 故障诊断指令

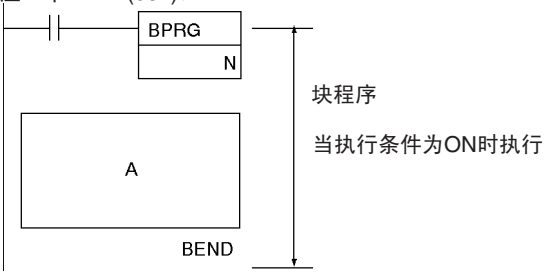
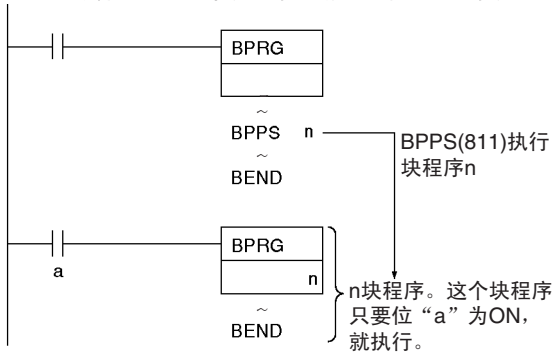
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
故障报警 FAL @FAL 006	FAL(006) N S N:FAL号 S：产生信息首字或错误代码	产生或清除用户定义的非致命错误。此错误不会使PC停止运行。也产生系统非致命错误 	输出 需要	934
严重故障报警 FALS 007	FALS(007) N S N:FAL号 S：产生信息首字或错误代码	产生用户定义的致命错误，此错误使PC机停止运行。 	输出 需要	942
故障点检测 FPD 269	FPD(269) C T R C:控制字 T:监控时间 R:寄存器首字	通过监视在FPD(269)执行和诊断输出执行之间的时间,诊断一个指令块中的错误,并且找出哪个输入阻止输出变ON。 	输出 需要	950

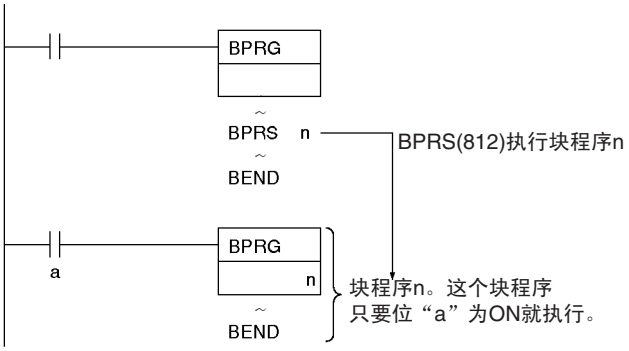
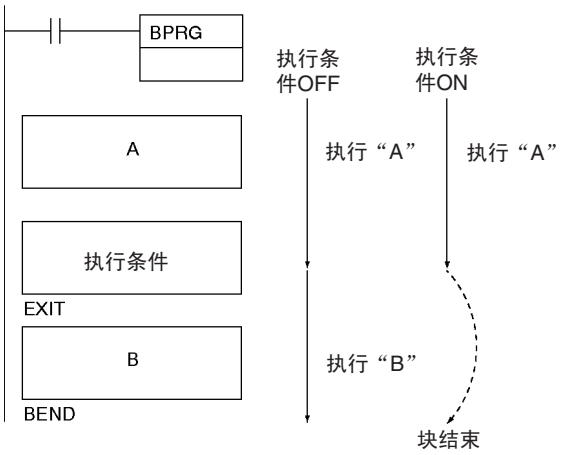
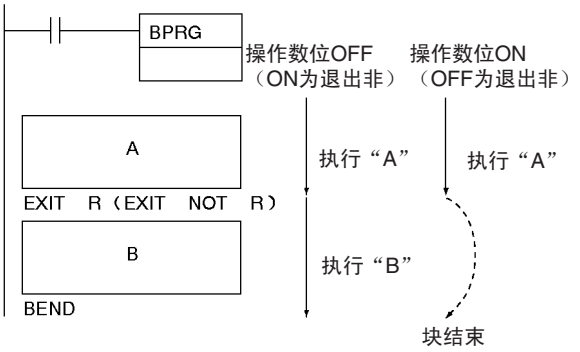
2-2-29 其它指令

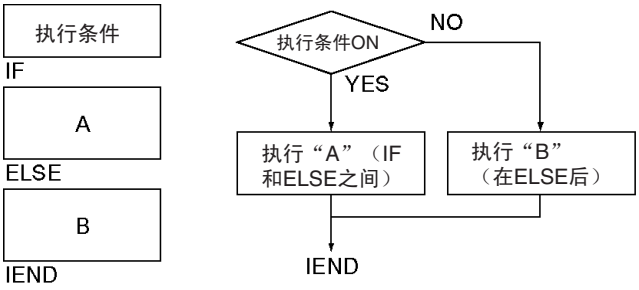
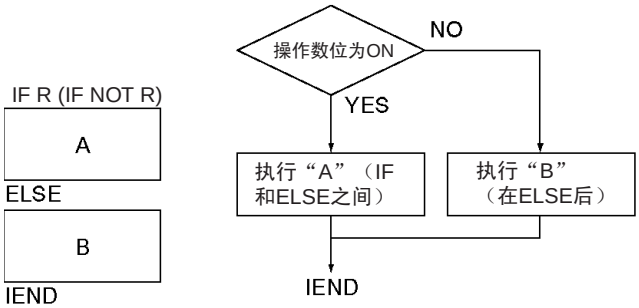
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
置进位 STC @STC 040		置进位标志 (CY)。	输出 需要	959
清进位 CLC @CLC 041		进位标志 (CY) 清零。	输出 需要	960
选择 EM 区 EMBC @EMBC 281	 N:EM区号	改变当前 EM 区。	输出 需要	961
延长最大循环时间 WDT @WDT 094	 T:定时器设定	延长最大循环时间 , 但仅在此指令执行的循环时间。	输出 需要	963
存储条件标志 （仅适用于 CS1-H,CJ1-H,CJ1M 和 CS1D） CCS @CCS 282		存储条件标志状态。	输出 需要	965
载入条件标志 （仅适用于 CS1-H,CJ1-H,CJ1M 和 CS1D） CCL @CCL 283		读取已存的条件标志状态。	输出 需要	967
从 CV 变换地址（仅适用于 CS1-H,CJ1-H,CJ1M 和 CS1D） FRMCV @FRMCV 284	 S:含有CV系列内存地址字 D:目标索引寄存器	把 CV 系列 PLC 内存地址变换到等同的 CS/CJ 系列 PLC 内存地址。	输出 需要	968
把地址变换到 CV（仅适用于 CS1-H,CJ1-H,CJ1M 和 CS1D） TOCV @TOCV 285	 S:含有CS系列内存地址索引寄存器 D:目标字	把 CS/CJ 系列 PLC 内存地址变换到等同的 CV 系列 PLC 内存地址。	输出 需要	972

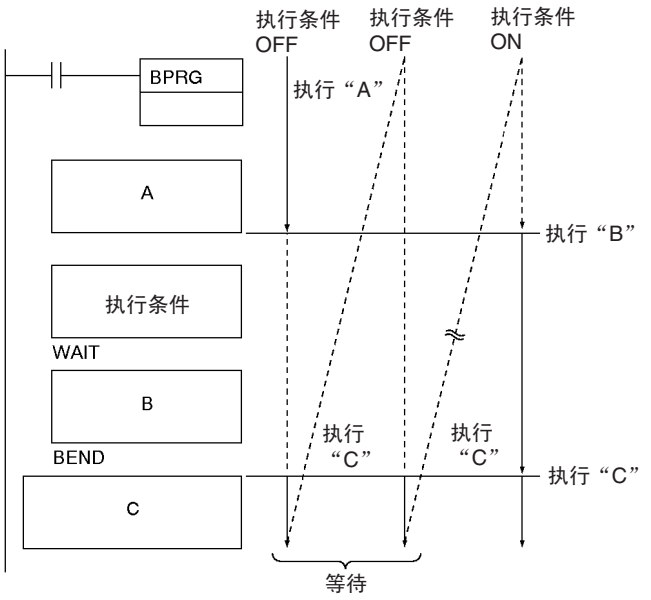
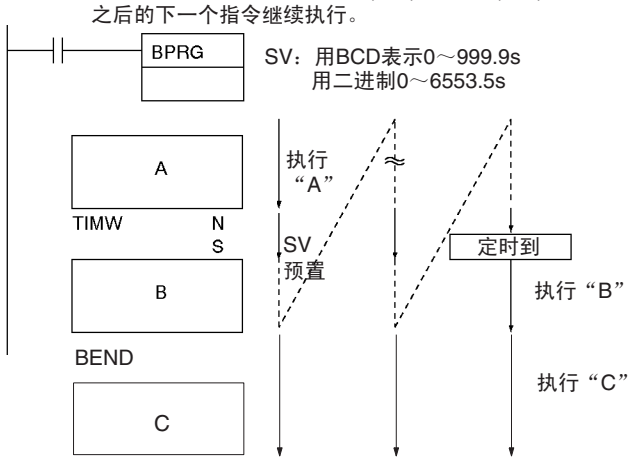
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
禁止外围设备服务（仅适用于 CS1-H,CJ1-H,CJ1M 和 CS1D） IOSP @IOSP 287	IOSP(287)	在运行并行执行模式或外设优先模式运行程序时，禁止外设服务。	输出 需要	976
允许外围设备服务（仅适用于 CS1-H,CJ1-H,CJ1M 和 CS1D） IORS 288	IORS(288)	在运行并行执行模式或外设优先模式运行程序时，允许曾由 IOSP(287) 禁止的外设服务	输出 需要	978

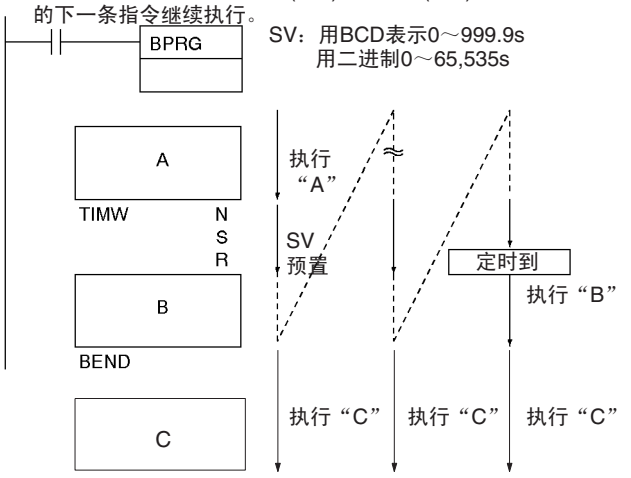
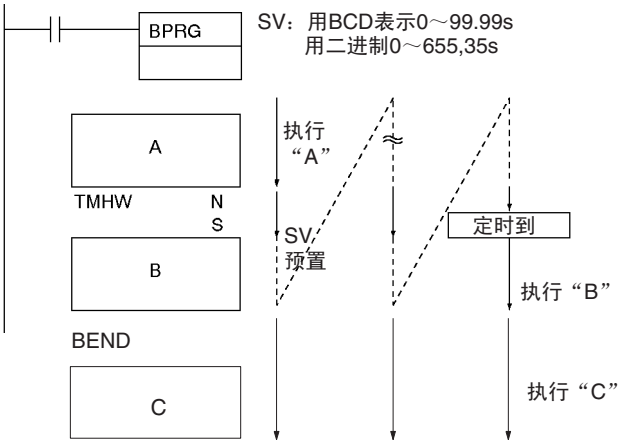
2-2-30 块指令

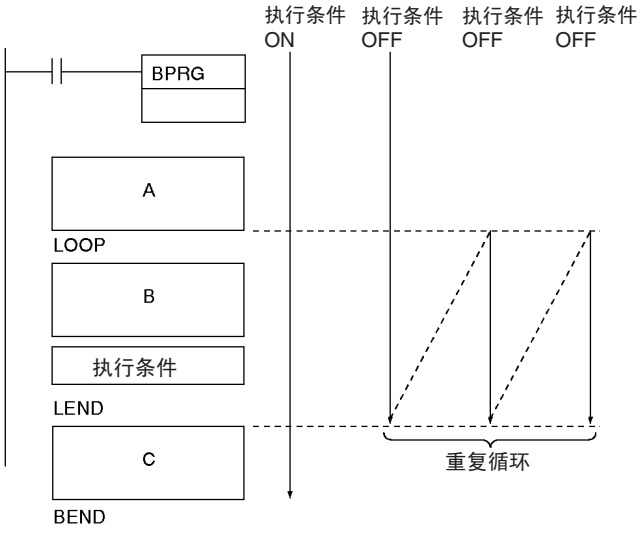
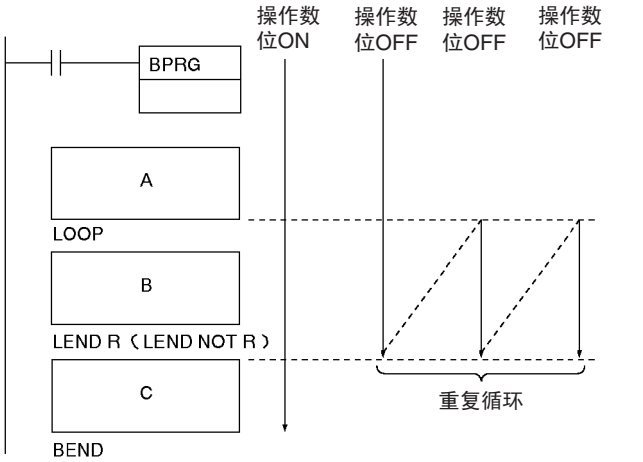
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
块程序开始 BPRG 096	BPRG(096) N N:块程序号	定义一个块程序区。对于每一个BPRG(096)对应一个BEND(081)。 	输出 需要	983
块程序结束 BEND 801		定义一个块程序区。对于每一个 BPRG(096) 对应一个 BEND(081)。	块程序需要	983
块程序暂停 BPPS 811	BPPS (811) N N:块程序号	暂停并且从另一个块程序从新启动指定的程序块。 	块程序需要	985

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
块程序重新开始 BPRS 812	BPRS (812) N N:块程序号	暂停并且从另一个块程序重新启动指定的块程序。 	块程序需要	985
条件块退出 EXIT 806	EXIT(806) B: 位操作数	如果执行条件为ON,没有操作数位的EXIT(806)将退出程序。 	块程序需要	991
条件块退出 EXIT 806	EXIT(806)B B: 位操作数	如果执行条件为ON具有操作数位的EXIT(806)将退出程序。 	块程序需要	991
条件块退出非 EXIT NOT 806	EXIT NOT(806) B B: 位操作数	如果执行条件为 OFF, 具有操作数位的 EXIT(806) 将退出程序。	块程序需要	991

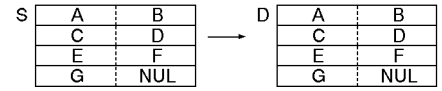
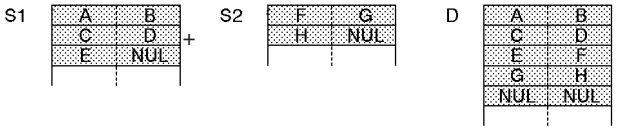
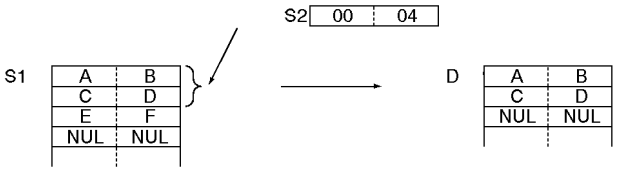
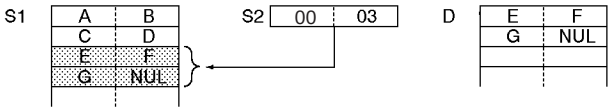
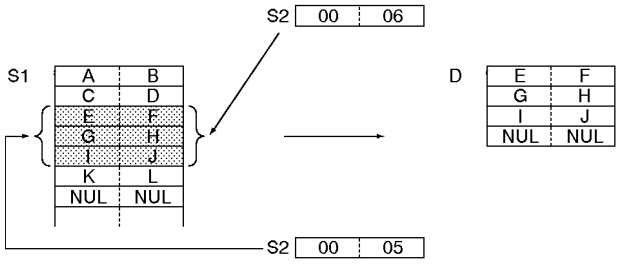
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
条件块分支 IF 802	IF (802)	如果执行条件为ON，那么就执行IF(802)和ELSE(803)之间的指令，如果执行条件为OFF，那么将执行ELSE(803)和IEND(804)之间的指令。 	块程序需要	988
条件块分支 IF 802	IF (802) B B: 位操作数	如果操作数位为ON,那么就执行IF(802和ELSE(803)之间的指令， 如果操作数位为OFF,那么将执行ELSE(803)和END(804)之间的指令。 	块程序需要	988
条件块分支（非） IF NOT 802	IF (802) NOT B B: 位操作数	如果操作数位为 OFF, 那么就执行 IF(802 和 ELSE(803) 之间的指令， 如果操作数位 ON, 那么将执行 ELSE(803) 和 IEND(804) 之间的指令	块程序需要	988
条件块分支 (ELSE) ELSE 803	---	如果忽略 ELSE(803) 指令，且操作数位为 ON, 那么就执行 IF(802 和 IEND(804) 之间的指令。	块程序需要	988
条件块分支 结束 IEND 804	---	如果操作数位为 OFF，仅执行 IEND(804) 后面的指令。	块程序需要	988

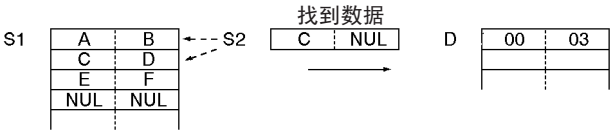
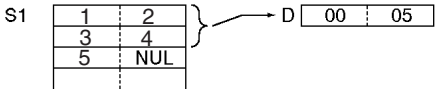
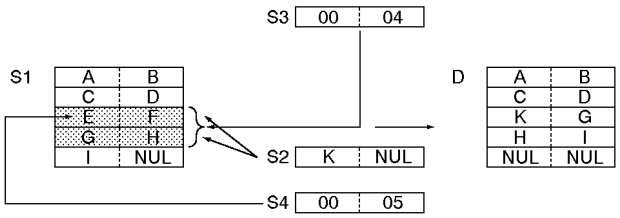
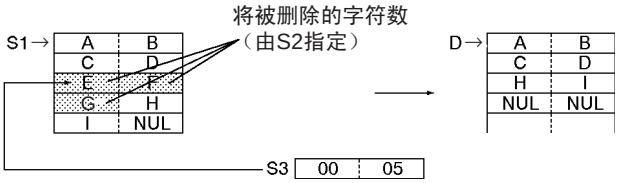
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
一个循环与等待 WAIT 805	WAIT(805)	如果WAIT(805)的执行条件为ON，那么将跳过块程序中的其余指令。  	块程序需要	994
一个循环与等待 WAIT 805	WAIT(805) B B: 位操作数	如果操作数位为 OFF （对 WAIT NOT(805) 而言是 ON），那么将跳过块程序中的其余指令。在下一个循环中，除了 WAIT(805) 或 WAIT(805)NOT 需要外，不执行程序中的其他程序。当执行条件变 ON （WAIT(805)NOT 为 OFF），执行从 WAIT(805) 或 WAIT(805)NOT 到程序结束之间的指定。	块程序需要	994
一个循环与等待（非） WAIT NOT 805	WAIT(805) NOT B B: 位操作数	如果操作数位为 OFF （对 WAIT NOT(805) 而言是 ON），那么将跳过块程序中的其余指令。在下一个循环中，除了 WAIT(805) 或 WAIT(805)NOT 需要外，不执行程序中的其他程序。当执行条件变 ON （WAIT(805)NOT 为 OFF），执行从 WAIT(805) 或 WAIT(805)NOT 到程序结束之间的指定。	块程序需要	994
定时器等待 TIMW 813 (BCD) TIMWX 816 （二进制） （仅适用于 CS1-H,CJ1-H,CJ1M 和 CS1D）	TIMW(813) N SV N: 定时器号 SV: 设定值 TIMWX(816) N SV N: 定时器号 SV: 设定值	延迟执行块程序的剩余指令,直至定时时间到。 当定时器定时到,块程序TIMW(813)/TIMWX(816)之后的下一个指令继续执行。  	块程序需要	998

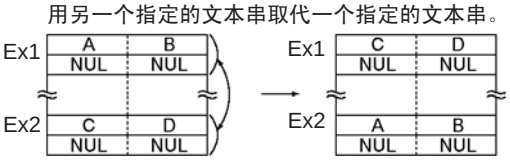
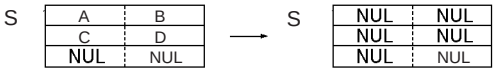
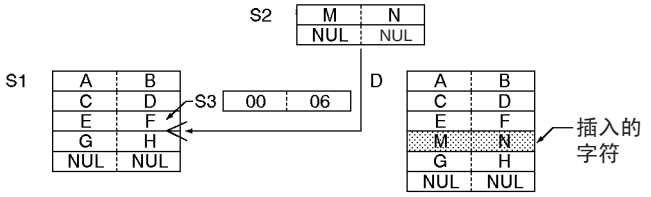
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
计数器等待 CNTW 814 (BCD) CNTWX 817 （二进制） （仅适用于 CS1-H,CJ1-H,CJ1M 和 CS1D）	CNTW(814) N SV N: 计数器号 SV: 设定值 I: 计数输入 CNTWX(817) N SV N: 计数器号 SV: 设定值 I: 计数输入	延迟执行块程序的剩余指令，直至计数器计数值达到。 当计数器完成计数，CNTW(814)/CNTWX(817)之后的下一条指令继续执行。 	块程序需要	1001
高速定时器等待 TMHW 815 (BCD) TMHWX 818 （二进制） （仅适用于 CS1-H,CJ1-H,CJ1M 和 CS1D）	TMHW(815) N SV N: 定时器号 SV: 设定值 TMHW(818) N SV N: 定时器号 SV: 设定值	延迟执行块程序的剩余指令,直至定时时间到。 当定时器定时到,块程序TMHW(815)之后的下一个指令继续执行。 	块程序需要	1004

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
循环 LOOP 809	---	LOOP(809)指定循环程序的开始 	块程序需要	1007
循环结束 LEND 810	LEND (810)	LEND(810) 或 LEND(810)NOT 指定循环的结束。当 LEND(810) 或 LEND(810)NOT 到达时，程序执行将循环回到前面 LOOP(809)，直到 LEND(810) 或 LEND(810)NOT 的操作数位分别变为 ON 或 OFF，或直到 LEND(810) 的执行条件变为 ON。	块程序需要	1007
循环结束 LEND 810	LEND (810) B B: 位操作数	如果LEND(810)操作数位为OFF，（或对LEND(810)NOT为ON），循环执行从LOOP(809)后的下一个指令处重复开始。如果LEND(810)操作数位为ON（或对LEND(810)NOT为OFF），循环将结束，并且到LEND(810)或LEND(810)NOT后的下一个指令处继续执行。  注 对LEND(810)NOT 的操作数位状态必须相反	块程序需要	1007
循环结束非 LEND NOT 810	LEND(810) NOT B: 位操作数	LEND(810) 或 LEND(810)NOT 指定循环的结束。当 LEND(810) 或 LEND(810)NOT 到达时，程序执行将循环回到前面 LOOP(809)，直到 LEND(810) 或 LEND(810)NOT 的操作数位分别变为 ON 或 OFF，或直到 LEND(810) 的执行条件变为 ON。	块程序需要	1007

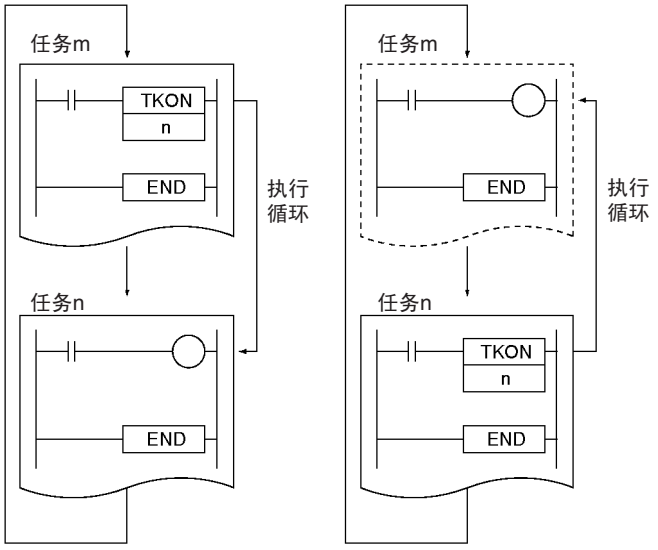
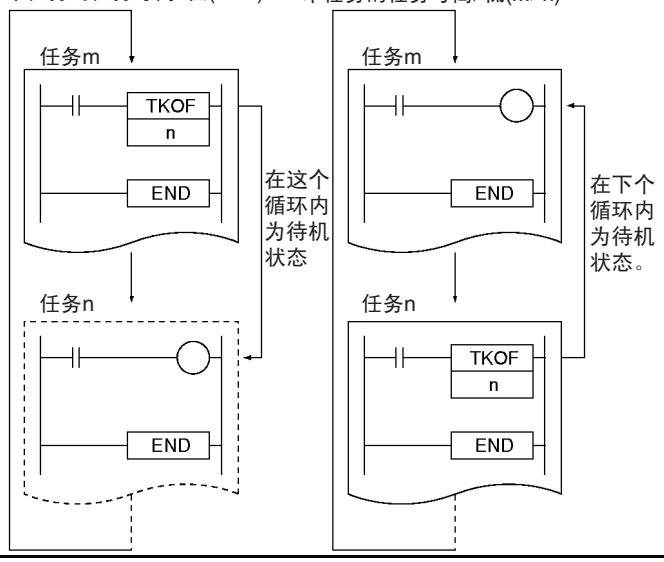
2-2-31 文本字符串处理指令

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
串发送 MOV$ @MOV$ 664	MOV$(664) S D S:源首字 D:目标首字	传送文本串。 	输出 需要	1013
链接串 +$ @+$ 656	+$(656) S1 S2 D S1:文本串1 S2:文本串2 D:目标首字	把一个文本串和另一文本串链接。 	输出 需要	1015
取左串 LEFT$ @LEFT$ 652	LEFT$(652) S1 S2 D S1:文本串1 S2:文本串2 D:目标首字	从一文本串的左边(开始)取一指定的字符数。 	输出 需要	1018
取右串 RGHT$ @RGHT$ 653	RGHT$(653) S1 S2 D S1:文本串1 S2:文本串2 D:目标首字	从一文本串的右边(结束)取一指定的字符数。 	输出 需要	1020
取中间串 MID$ @MID$ 654	MID$(654) S1 S2 S3 D S1:文本串1 S2:文本串2 S3:起始位置 D:目标首字	从一文本串中间任何位置读取一指定的字符数 	输出 需要	1022

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
寻找串 FIND @FIND$ 660	FIND$(660) S1 S2 D S1:文本串源首字 S2:寻找文本串首字 D:目标首字	在一文本串中找寻一指定的文本串。 	输出 需要	1024
计算串长 LEN$ @LEN$ 650	LEN$(650) S D S:文本串首字 D:目标首字	计算文本串的长度。 	输出 需要	1026
取代串 RPLC$ @RPLC$ 661	RPLC$(654) S1 S2 S3 S4 D S1:文本串首字 S2:取代的文本串首字 S3:字符数 S4:首位置 D:目标首字	用一指定的文本串从指定位置取代一个文本串。 	输出 需要	1028
删除串 DEL$ @DEL$ 658	DEL$(658) S1 S2 S3 D S1:文本串首字 S2: 字符数 S3: 首位置 D:目标首字	从文本串的中间删除指定的文本串。 	输出 需要	1031

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
交换串 XCHG$ @XCHG$ 665	XCHG$(665) Ex1 Ex2 Ex1:交换首字1 Ex2:交换首字2	用另一个指定的文本串取代一个指定的文本串。 	输出 需要	1033
清除串 CLR$ @CLR$ 666	CLR$(666) S S:文本串首字	用NUL(00HEX)清除整个文本串。 	输出 需要	1035
插入串 INS$ @INS$ 657	INS$(657) S1 S2 S3 D S1:基于文本串首字 S2:插入文本串首字 S3:起始位置 D:目标首字	从文本串的中间删除一指定的文本串。 	输出 需要	1037
串比较 LD, AND, OR + =$, <>$, <$, <=$, >$, >=$ 670 (=$) 671 (<>$) 672 (<$) 673 (<=$) 674 (>$) 675 (>=$)	LD 符号 S1 S2 AND 符号 S1 S2 OR 符号 S1 S2 S1:文本串1 S2:文本串2	根据 ASCII 代码值 , 串比较指令 (=$,<>$,<$,<=$,>$,>=$,) 从开始比较两个文本串。如果比较结果为真 , 那么就为 LOAD;AND, 或 OR 产生一个 ON 执行条件。	LD: 不需要 AND, OR: 需要	1040

2-2-32 任务控制指令

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件	页号
任务 ON TKON @TKON 820	TKON(820) N N:任务号	使指定的任务可执行。 指定任务的任务号比本任务的任务号高/低(m<n) 指定任务的任务号比本任务的任务号高/低(m>n) 	输出 需要	1045
任务 OFF TKOF @TKOF 821	TKOF(821) N N:任务号	使一指定的任务为待机状态 指定任务的任务号比本任务的任务号高/低(m<n) 指定任务的任务号比本任务的任务号高/低(m>n) 	输出 需要	1049

2-3 助记符按字母顺序排列的指令表

A

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
ACC	加速度控制	888	@ACC	---	---	795
ACOS	反余弦	464	@ACOS	---	---	546
ACOSD	双字反余弦	855	@ACOSD	---	---	602
AND	与	---	@AND	%AND	!AND	146
AND <	与小于	310	---	---	---	246
AND <$	与串小于	672	---	---	---	1040
AND <>	与不等于	305	---	---	---	246
AND <>$	与串不等于	671	---	---	---	1040
AND <>D	与双字浮点不等于	336	---	---	---	614
AND <>F	与浮点不等于	330	---	---	---	557
AND <>L	与双字不等于	306	---	---	---	246
AND <>S	与带符号不等于	307	---	---	---	246
AND <>SL	与双字带符号不等于	308	---	---	---	246
AND <D	与双字浮点小于	337	---	---	---	614
AND <F	与浮点小于	331	---	---	---	557
AND <L	与双字小于	311	---	---	---	246
AND <S	与带符号小于	312	---	---	---	246
AND <SL	与双字带符号小于	313	---	---	---	246
AND =	与等于	300	---	---	---	246
AND =$	与串等于	670	---	---	---	1040
AND =D	与双字浮点等于	335	---	---	---	614
AND =F	与浮点等于	329	---	---	---	557
AND =L	与双字等于	301	---	---	---	246
AND =S	与带符号等于	302	---	---	---	246
AND =SL	与双字带符号等于	303	---	---	---	246
AND >	与大于	320	---	---	---	246
AND >$	与串大于	674	---	---	---	1040
AND >D	与双字浮点大于	339	---	---	---	614
AND >F	与浮点大于	333	---	---	---	557
AND >L	与双字大于	321	---	---	---	246
AND >S	与带符号大于	322	---	---	---	246
AND >SL	与双字带符号大于	323	---	---	---	246
AND LD	逻辑块与	---	---	---	---	153

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
AND NOT	与非	---	---	---	!AND NOT	148
AND TST	与位测试	350	---	---	---	163
AND TSTN	与位测试	351	---	---	---	163
AND <=	与小于等于	315	---	---	---	246
AND <=$	与串小于等于	673	---	---	---	1040
AND <=D	与双字浮点小于等于	338	---	---	---	614
AND <=F	与浮点小于等于	332	---	---	---	557
AND <=L	与双字小于等于	316	---	---	---	246
AND <=S	与带符号小于等于	317	---	---	---	246
AND <=SL	与双字带符号小于等于	318	---	---	---	246
AND >=	与大于等于	325	---	---	---	246
AND >=$	与串大于等于	675	---	---	---	1040
AND >=D	与双字浮点大于等于	340	---	---	---	614
AND >=F	与浮点大于等于	334	---	---	---	557
AND >=L	与双字大于等于	326	---	---	---	246
AND >=S	与带符号大于等于	327	---	---	---	246
AND >=SL	与双字带符号大于等于	328	---	---	---	246
ANDL	双字逻辑与	610	@ANDL	---	---	476
ANDW	逻辑与	034	@ANDW	---	---	474
APR	数学处理	069	@APR	---	---	497
ASC	ASCII 转换	086	@ASC	---	---	449
ASFT	异步移位寄存器	017	@ASFT	---	---	313
ASIN	反正弦	463	@ASIN	---	---	544
ASIND	双字反正弦	854	@ASIND	---	---	600
ASL	算术左移	025	@ASL	---	---	317
ASLL	双字左移	570	@ASLL	---	---	319
ASR	算术右移	026	@ASR	---	---	321
ASRL	双字右移	571	@ASRL	---	---	322
ATAN	反正切	465	@ATAN	---	---	548
ATAND	双字反正切	856	@ATAND	---	---	604
AVG	求平均	195	---	---	---	716

B

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
BAND	静带控制	681	@BAND	---	---	698
BCD	二进制变为 BCD 码	024	@BCD	---	---	432
BCDL	双字二进制变为 BCD 码	059	@BCDL	---	---	433
BCDS	带符号二进制变为 BCD 码	471	@BCDS	---	---	468
BCMP	不带符号块比较	068	@BCMP	---	---	268
BCMP2	扩展块比较	502	@BCMP2	---	---	271
BCNT	位计数器	067	@BCNT	---	---	513
BDSL	双字带符号二进制变为 BCD 码	473	@BDSL	---	---	470
BEND	块程序结束	801	---	---	---	983
BIN	BCD 码变为二进制	023	@BIN	---	---	429
BINL	双字 BCD 码变为双字二进制	058	@BINL	---	---	430
BINS	带符号 BCD 码变为二进制	470	@BINS	---	---	462
BISL	双字带符号 BCD 码变为二进制	472	@BISL	---	---	465
BPPS	块程序暂停	811	---	---	---	985
BPRG	块程序开始	096	---	---	---	983
BPRS	块程序重新启动	812	---	---	---	985
BREAK	中断循环	514	---	---	---	204
BSET	块设定	071	@BSET	---	---	295

C

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
CADD	日历加法	730	@CADD	---	---	916
CCL	载入条件标志	283	@CCL	---	---	967
CCS	存入条件标志	282	@CCS	---	---	965
CJP	条件跳转	510	---	---	---	195
CJPN	条件跳转	511	---	---	---	195
CLC	清进位	041	@CLC	---	---	960
CLI	清中断	691	@CLI	---	---	755
CLR$	清串	666	@CLR$	---	---	1035
CMND	发布命令	490	@CMND	---	---	890
CMP	比较	020	---	---	!CMP	252
CMPL	双字比较	060	---	---	---	254
CNR	定时器 / 计数器复位	545	@CNR	---	---	238
CNRX	定时器 / 计数器复位	548	@CNRX	---	---	238
CNT	计数器	---	---	---	---	231
CNTX	计数器	546	---	---	---	231
CNTR	可逆计数器	012	---	---	---	234

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
CNTRX	可逆计数器	548	---	---	---	234
CNTW	计数器等待	814	---	---	---	1001
CNTWX	计数器等待	818	---	---	---	1001
COLL	数据收集	081	@COLL	---	---	303
COLM	行到列	064	@COLM	---	---	459
COM	求反	029	---	---	---	488
COML	双字求反	614	@COML	---	---	490
COS	余弦	461	@COS	---	---	540
COSD	双字余弦	852	@COSD	---	---	596
CPS	带符号二进制比较	114	---	---	!CPS	258
CPSL	双字带符号二进制比较	115	---	---	---	260
CSUB	日历减	731	@CSUB	---	---	920
CTBL	比较表载入	882	@CTBL	---	---	777

D

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
DATE	时钟调整	735	@DATE	---	---	928
DBL	16 位二进制变为双字浮点	843	@DBL	---	---	580
DBLL	32 位二进制变为双字浮点	844	@DBLL	---	---	581
DEG	弧度变度	459	@DEG	---	---	536
DEGD	双字弧度变度	850	@RADD	---	---	591
DEL$	删除串	658	@DEL$	---	---	1031
DI	禁止中断	693	@DI	---	---	760
DIFD	下降沿微分	014	---	---	!DIFD	173
DIFU	上升沿微分	013	---	---	!DIFU	173
DIM	定维记录表	631	@DIM	---	---	635
DIST	单字分配	080	@DIST	---	---	301
DLNK	CPU 总线单元 I/O 刷新	226	@DLNK	---	---	837
DMPX	数据编码	077	@DMPX	---	---	445
DOWN	条件 OFF	522	---	---	---	162

E

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
EI	允许中断	694	---	---	---	762
ELSE	ELSE	803	---	---	---	988
EMBC	选择 EM 区	281	@EMBC	---	---	961
END	结束	001	---	---	---	186
EXIT NOT（操作数）	条件块退出非	806	---	---	---	991

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
EXIT（输入条件）	条件块退出	806	---	---	---	991
EXIT（操作数）	条件块退出	806	---	---	---	991
EXP	指数	467	@EXP	---	---	552
EXPD	双字指数	858	@EXPD	---	---	608

F

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
FAL	故障报警	006	@FAL	---	---	934
FALS	严重故障报警	007	---	---	---	942
FCS	帧校验和	180	@FCS	---	---	656
FDIV	浮点除法	079	@FDIV	---	---	509
FIFO	先进先出	633	@FIFO	---	---	629
FIND$	寻找串	660	@FIND$	---	---	1024
FIX	浮点到 16 位	450	@FIX	---	---	520
FIXD	双字浮点到 16 位二进制	841	@FIXD	---	---	577
FIXL	浮点到 32 位	451	@FIXL	---	---	522
FIXLD	双字浮点到 32 位二进制	842	@FIXLD	---	---	578
FLT	16 位到浮点	452	@FLT	---	---	523
FLTL	32 位到浮点	453	@FLTL	---	---	525
FOR	FOR-NEXT 循环	512	---	---	---	201
FPD	故障点检测	269	---	---	---	950
FREAD	读数据文件	700	@FREAD	---	---	899
FRMCV	从 CV 转变地址	284	@FRMCV	---	---	968
FSTR	浮点到 ASCII	448	@FSTR	---	---	561
FWRIT	写数据文件	701	@FWRIT	---	---	906
FVAL	ASCII 到浮点	449	@FVAL	---	---	566

G

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
GETR	获得记录号	636	@GETR	---	---	640
GRET	返回全部子程序	752	---	---	---	743
GSBN	进入全部子程序	751	---	---	---	740
GSBS	调用全部子程序	750	@GSBS	---	---	732

H

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
HEX	ASCII 到 16 进制	162	@HEX	---	---	453
HMS	秒到小时	066	@HMS	---	---	925

I

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
IEND	条件结束	804	---	---	---	988
IF NOT（操作数）	条件非	802	---	---	---	988
IF （输入条件）	条件分支	802	---	---	---	988
IF （操作数）	条件分支	802	---	---	---	988
IL	联锁	002	---	---	---	187
ILC	联锁清除	003	---	---	---	187
INI	模式控制	880	@INI	---	---	769
INS$	插入串	657	@INS$	---	---	1037
IORD	智能 I/O 读	222	@IORD	---	---	831
IORF	I/O 刷新	097	@IORF	---	---	825
IORS	外设服务允许	288	---	---	---	978
IOSP	外设服务禁止	287	@IOSP	---	---	976
IOWR	智能 I/O 写	223	@IOWR	---	---	834

J

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
JME	跳转结束	005	---	---	---	191
JME0	多路跳转结束	516	---	---	---	199
JMP	跳转	004	---	---	---	191
JMP0	多路跳转	515	---	---	---	199

K

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
KEEP	保持	011	---	---	!KEEP	168

L

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
LD	载入	---	@LD	%LD	!LD	142
LD <	载入小于	310	---	---	---	246
LD <$	载入串小于	672	---	---	---	1040
LD <D	载入双字浮点小于	337	---	---	---	614
LD <F	载入浮点小于	331	---	---	---	557
LD <>	载入不等于	305	---	---	---	246
LD <>$	载入串不等于	671	---	---	---	1040
LD <>D	载入双字浮点不等于	336	---	---	---	614
LD <>F	载入浮点不等于	330	---	---	---	557
LD <>L	载入双字不等于	306	---	---	---	246

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
LD <>S	载入带符号不等于	307	---	---	---	246
LD <>SL	载入双字带符号不等于	308	---	---	---	246
LD <L	载入双字小于	311	---	---	---	246
LD <S	载入带符号小于	312	---	---	---	246
LD <SL	载入双字带符号小于	313	---	---	---	246
LD =	载入等于	300	---	---	---	246
LD =$	载入串等于	670	---	---	---	1040
LD =D	载入双字浮点等于	335	---	---	---	614
LD =F	载入浮点等于	329	---	---	---	557
LD =L	载入双字等于	301	---	---	---	246
LD =S	载入带符号等于	302	---	---	---	246
LD =SL	载入双字带符号等于	303	---	---	---	246
LD >	载入大于	320	---	---	---	246
LD >$	载入串大于	674	---	---	---	1040
LD >D	载入双字浮点大于	339	---	---	---	614
LD >F	载入浮点大于	333	---	---	---	557
LD >L	载入双字大于	321	---	---	---	246
LD >S	载入带符号大于	322	---	---	---	246
LD >SL	载入双字带符号大于	323	---	---	---	246
LD NOT	载入非	---	---	---	!LD NOT	144
LD TST	载入位测试	350	---	---	---	163
LD TSTN	载入位测试	351	---	---	---	163
LD <=	载入小于等于	315	---	---	---	246
LD <=$	载入串小于等于	673	---	---	---	1040
LD <=D	载入双字浮点小于等于	338	---	---	---	614
LD <=F	载入浮点小于等于	332	---	---	---	557
LD <=L	载入双字小于等于	316	---	---	---	246
LD <=S	载入带符号小于等于	317	---	---	---	246
LD <=SL	载入双字带符号小于等于	318	---	---	---	246
LD >=	载入大于等于	325	---	---	---	246
LD >=$	载入串大于等于	675	---	---	---	1040

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
LD >=D	载入双字浮点大于等于	340	---	---	---	614
LD >=F	载入浮点大于等于	334	---	---	---	557
LD >=L	载入双字大于等于	326	---	---	---	246
LD >=S	载入带符号大于等于	327	---	---	---	246
LD >=SL	载入双字带符号大于等于	328	---	---	---	246
LEFT$	取左串	652	@LEFT$	---	---	1018
LEN$	串长	650	@LEN$	---	---	1026
LEND NOT（操作数）	循环结束非	810	---	---	---	1007
LEND（输入条件）	循环结束	810	---	---	---	1007
LEND（操作数）	循环结束	810	---	---	---	1007
LIFO	后进先出	634	@LIFO	---	---	632
LINE	列到行	063	@LINE	---	---	457
LMT	限位控制	680	@LMT	---	---	696
LOG	对数	468	@LOG	---	---	554
LOGD	双字对数	859	@LOGD	---	---	610
LOOP	循环	809	---	---	---	1007

M

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
MAX	寻找最大值	182	@MAX	---	---	646
MCMP	多字比较	019	@MCMP	---	---	263
MCRO	宏	099	@MCRO	---	---	725
MID$	取中间串	654	@MID$	---	---	1022
MIN	寻找最小值	183	@MIN	---	---	650
MLPX	数据译码	076	@MLPX	---	---	440
MOV	传送	021	@MOV	---	!MOV	279
MOV$	串传送	664	@MOV$	---	---	1013
MOVB	位传送	082	@MOVB	---	---	286
MOVD	数字传送	083	@MOVD	---	---	288
MOVL	双字传送	498	@MOVL	---	---	282
MOVR	传送至寄存器	560	@MOVR	---	---	305
MSG	显示信息	046	@MSG	---	---	913
MSKR	读中断屏蔽	692	@MSKR	---	---	750
MSKS	设置中断屏蔽	690	@MSKS	---	---	744
MTIM	多路输出定时器	543	---	---	---	226
MTIMX	多路输出定时器	554	---	---	---	226
MVN	取反传送	022	@MVN	---	---	281

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
MVNL	双字取反传送	499	@MVNL	---	---	284
MOVRW	传送定时器 / 计数器 PV 至存储器	561	---	---	---	307

N

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
NASL	左移 N 位	580	@NASL	---	---	345
NASR	右移 N 位	581	@NASR	---	---	351
NEG	二进制求补	160	@NEG	---	---	435
NEGL	双字二进制求补	161	@NEGL	---	---	437
NEXT	FOR-NEXT 循环	513	---	---	---	201
NOP	空操作	000	---	---	---	187
NOT	非	520	---	---	---	161
NSFL	N 位数据左移	578	@NSFL	---	---	341
NSFR	N 位数据右移	579	@NSFR	---	---	343
NSLL	双字左移 N 位	582	@NSLL	---	---	348
NSRL	双字右移 N 位	583	@NSRL	---	---	353

O

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
OR	或	---	@OR	%OR	!OR	150
OR <	或小于	310	---	---	---	246
OR <$	或串小于	672	---	---	---	1040
OR <>	或不等于	305	---	---	---	246
OR <>$	或串不等于	671	---	---	---	1040
OR <>D	或双字浮点小于	336	---	---	---	614
OR <>F	或浮点不等于	330	---	---	---	557
OR <>L	或双字不等于	306	---	---	---	246
OR <>S	或带符号不等于	307	---	---	---	246
OR <>SL	或双字带符号不等于	308	---	---	---	246
OR <D	或双字浮点小于	337	---	---	---	614
OR <F	或浮点小于	331	---	---	---	557
OR <L	或双字小于	311	---	---	---	246
OR <S	或带符号小于	312	---	---	---	246
OR <SL	或双字带符号小于	313	---	---	---	246
OR =	或等于	300	---	---	---	246
OR =$	或串等于	670	---	---	---	1040

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
OR =D	或双字浮点等于	335	---	---	---	614
OR =F	或浮点等于	329	---	---	---	557
OR =L	或双字等于	301	---	---	---	246
OR =S	或带符号等于	302	---	---	---	246
OR =SL	或双字带符号等于	303	---	---	---	246
OR >	或大于	320	---	---	---	246
OR >$	或串大于	674	---	---	---	1040
OR >D	或双字浮点大于	339	---	---	---	614
OR >F	或浮点大于	333	---	---	---	557
OR >L	或双字大于	321	---	---	---	246
OR >S	或带符号大于	322	---	---	---	246
OR >SL	或双字带符号大于	323	---	---	---	246
OR LD	或载入	---	---	---	---	155
OR NOT	或非	---	---	---	!OR NOT	151
OR TST	或位测试	350	---	---	---	163
OR TSTN	或位测试	351	---	---	---	163
OR <=	或小于等于	315	---	---	---	246
OR <=$	或串小于等于	673	---	---	---	1040
OR <=D	或双字浮点小于等于	338	---	---	---	614
OR <=F	或浮点小于等于	332	---	---	---	557
OR <=L	或双字小于等于	316	---	---	---	246
OR <=S	或带符号小于等于	317	---	---	---	246
OR <=SL	或双字带符号小于等于	318	---	---	---	246
OR >=	或大于等于	325	---	---	---	246
OR >=$	或串大于等于	675	---	---	---	1040
OR >=D	或双字大于等于	340	---	---	---	614
OR >=F	或浮点大于等于	334	---	---	---	557
OR >=L	或双字大于等于	326	---	---	---	246
OR >=S	或带符号大于等于	327	---	---	---	246
OR >=SL	或双字带符号大于等于	328	---	---	---	246
ORG	寻找原点	889	@ORG	---	---	802
ORW	逻辑或	035	@ORW	---	---	477
ORWL	双字逻辑或	611	@ORWL	---	---	479

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
OUT	输出	---	---	---	!OUT	166
OUTB	单位输出	534	@OUTB	---	!OUTB	184
OUT NOT	输出非	---	---	---	!OUT NOT	167

P

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
PID	PID 控制	190	---	---	---	675
PIDAT	PID 自动控制	191	---	---	---	686
PMCR	协议宏	260	@PMCR	---	---	844
PRV	高速计数器 PV 读	881	@PRV	---	---	773
PULS	设置脉冲	886	@PULS	---	---	786
PLS2	脉冲输出	887	@PLS2	---	---	789
PUSH	推入栈	632	@PUSH	---	---	626
PWM	可变占空比脉冲	891	@PWM	---	---	805
PWR	指数幂	840	@PWR	---	---	556
PWRD	双字指数幂	860	@PWRD	---	---	612

R

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
RAD	度到弧度	458	@RAD	---	---	554
RADD	双字度到弧度	849	@RADD	---	---	591
RECV	网络接收	098	@RECV	---	---	885
RET	子程序返回	093	---	---	---	732
RGHT$	取右串	653	@RGHT$	---	---	1020
RLNC	无进位循环左移	574	@RLNC	---	---	331
RLNL	无进位双字循环左移	576	@RLNL	---	---	333
ROL	循环左移	027	@ROL	---	---	324
ROLL	双字循环左移	572	@ROLL	---	---	326
ROOT	BCD 码平方根	072	@ROOT	---	---	493
ROR	循环右移	028	@ROR	---	---	327
RORL	双字循环右移	573	@RORL	---	---	329
ROTB	二进制平方根	620	@ROTB	---	---	491
RPLC$	用串取代	661	@RPLC$	---	---	1028
RRNC	无进位循环右移	575	@RRNC	---	---	334
RRNL	无进位双字循环右移	577	@RRNL	---	---	336
RSET	复位	---	@RSET	%RSET	!RSET	175
RSTA	多路复位	531	@RSTA	---	---	177
RSTB	单位复位	533	@RSTB	---	!RSTB	180
RXD	接收	235	@RXD	---	---	858

S

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
SBN	子程序入口	092	---	---	---	729
SBS	子程序调用	091	@SBS	---	---	720
SCL	标度	194	@SCL	---	---	704
SCL2	标度 2	486	@SCL2	---	---	708
SCL3	标度 3	487	@SCL3	---	---	712
SDEC	七段译码	078	@SDEC	---	---	844
SDEL	堆栈数据删除	642	@SDEL	---	---	671
SEC	小时→秒	065	@SEC	---	---	923
SEND	网络发送	090	@SEND	---	---	879
SET	置位	---	@SET	%SET	!SET	175
SETA	多位置位	530	@SETA	---	---	177
SETB	单位置位	532	@SETB	---	!SETB	180
SETR	设定记录位置	635	@SETR	---	---	638
SFT	移位寄存器	010	---	---	---	309
SFTR	可逆移位寄存器	084	@SFTR	---	---	310
SIGN	16 到 32 位带符号二进制位	600	@SIGN	---	---	439
SIN	正弦	460	@SIN	---	---	538
SIND	双字正弦	851	@SIND	---	---	594
SINS	插入堆栈数据	641	@SINS	---	---	668
SLD	一个数字左移	074	@SLD	---	---	338
SNUM	读堆栈大小	638	@SNUM	---	---	659
SNXT	步开始	009	---	---	---	808
SPED	速度输出	885	@SPED	---	---	781
SQRT	平方根	466	@SQRT	---	---	550
SQRTD	双字平方根	857	@SQRTD	---	---	606
SRCH	数据搜索	181	@SRCH	---	---	642
SRD	一个数字右移	075	@SRD	---	---	340
SREAD	读堆栈数	639	@SREAD	---	---	662
SSET	设置堆栈	630	@SSET	---	---	623
STC	设置进位	040	@STC	---	---	959
STEP	步定义	008	---	---	---	808
STUP	修改串行口设置	237	@STUP	---	---	863
SUM	求和	184	@SUM	---	---	653
SWAP	交换字节	637	@SWAP	---	---	644
SWRIT	写堆栈数据	640	@SWRIT	---	---	665

T

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
TAN	正切	462	@TAN	---	---	542
TAND	双字正切	853	@TAND	---	---	598
TCMP	表格比较	085	@TCMP	---	---	265
TIM	定时器	---	---	---	---	207
TIMH	高速定时器	015	---	---	---	211
TIMHX	高速定时器	551	---	---	---	211

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
TIML	长定时器	542	---	---	---	222
TIMLX	长定时器	553	---	---	---	222
TIMW	定时等待	813	---	---	---	998
TIMWX	定时等待	816	---	---	---	998
TIMX	定时器	505	---	---	---	207
TKOF	任务 OFF	821	@TKOF	---	---	1049
TKON	任务 ON	820	@TKON	---	---	1045
TMHH	1 毫秒定时器	540	---	---	---	216
TMHHX	1 毫秒定时器	552	---	---	---	216
TMHW	高速定时器等待	815	---	---	---	1004
TMHWX	高速定时器等待	817	---	---	---	1004
TOCV	变换地址到 CV	285	@TOCV	---	---	972
TRSM	跟踪内存采样	045	---	---	---	930
TTIM	累加定时器	087	---	---	---	219
TTIMX	累加定时器	555	---	---	---	219
TXD	传送	236	@TXD	---	---	853

U

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
UP	条件 ON	521	---	---	---	162

W

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
WAIT NOT（操作数）	一个周期与等待非	805	---	---	---	994
WAIT（输入条件）	一个周期与等待	805	---	---	---	994
WAIT（操作数）	一个周期与等待	805	---	---	---	994
WDT	延长最大循环时间	094	@WDT	---	---	963
WSFT	字移位	016	@WSFT	---	---	316

X

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
XCGL	双字数据交换	562	@XCGL	---	---	299
XCHG	数据交换	073	@XCHG	---	---	297
XCHG$	交换串	665	@XCHG$	---	---	1033
XFER	块传送	070	@XFER	---	---	293
XFRB	多位传送	062	@XFRB	---	---	290
XNRL	双字异或非	613	@XNRL	---	---	486
XNRW	异或非	037	@XNRW	---	---	485

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
XORL	双字异或	612	@XORL	---	---	483
XORW	异或	036	@XORW	---	---	481

Z

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
ZCP	区域范围比较	088	---	---	---	274
ZCPL	双字区比较	116	---	---	---	277
ZONE	静域控制	682	@ZONE	---	---	701

符号

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
+	无进位带符号二进制加法	400	@+	---	---	373
+$	链接串	656	@+$	---	---	1015
++	二进制递增	590	@++	---	---	356
++B	BCD 码递增	594	@++B	---	---	364
++BL	双字 BCD 码递增	595	@++BL	---	---	366
++L	双字二进制递增	591	@++L	---	---	358
+B	无进位 BCD 码加法	404	@+B	---	---	381
+BC	有进位 BCD 码加法	406	@+BC	---	---	384
+BCL	有进位双字 BCD 码加法	407	@+BCL	---	---	386
+BL	无进位双字 BCD 码加法	405	@+BL	---	---	382
+C	有进位带符号二进制加法	402	@+C	---	---	377
+CL	有进位双字带符号二进制加法	403	@+CL	---	---	379
+D	双字浮点加法	845	@+D	---	---	583
+F	浮点加法	454	@+F	---	---	527
+L	无进位带符号双字二进制加法	401	@+L	---	---	375
–	无进位带符号二进制减法	410	@–	---	---	387
– –	二进制递减	592	@– –	---	---	360
– –B	BCD 码递减	596	@– –B	---	---	368
– –BL	双字 BCD 码递减	597	@– –BL	---	---	370
– –L	双字二进制递减	593	@– –L	---	---	362
–B	无进位 BCD 码减法	414	@–B	---	---	398
–BC	有进位 BCD 码减法	416	@–BC	---	---	403

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
–BCL	有进位双字 BCD 码减法	417	@–BCL	---	---	404
–BL	无进位双字 BCD 码减法	415	@–BL	---	---	399
–C	有进位带符号二进制减法	412	@–C	---	---	393
–CL	有进位带符号双字二进制减法	413	@–CL	---	---	395
–D	双字浮点减法	846	@–D	---	---	585
–F	浮点减法	455	@–F	---	---	529
*	带符号二进制乘法	420	@*	---	---	406
*B	BCD 码乘法	424	@*B	---	---	413
*BL	双字 BCD 码乘法	425	@*BL	---	---	415
*D	双字浮点乘法	847	@*D	---	---	587
*F	浮点乘法	456	@*F	---	---	531
*L	带符号双字二进制乘法	421	@*L	---	---	408
*U	不带符号二进制乘法	422	@*U	---	---	410
*UL	不带符号双字二进制乘法	423	@*UL	---	---	412
–L	无进位带符号双字二进制减法	411	@–L	---	---	389
/	带符号二进制除法	430	@/	---	---	417
/B	BCD 码除法	434	@/B	---	---	425
/BL	双字 BCD 码除法	435	@/BL	---	---	427
/D	双字浮点除法	848	@/D	---	---	589
/F	浮点除法	457	@/F	---	---	533
/L	双字带符号二进制除法	431	@/L	---	---	419
/U	不带符号二进制除法	432	@/U	---	---	421
/UL	不带符号双字二进制除法	433	@/UL	---	---	423

2-4 功能代码指令列表

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
---	LD	装载	@LD	%LD	!LD	142
---	LD NOT	装载非	---	---	!LD NOT	144
---	AND	与	@AND	%AND	!AND	146
---	AND NOT	与非	---	---	!AND NOT	148
---	OR	或	@OR	%OR	!OR	150
---	OR NOT	或非	---	---	!OR NOT	151
---	AND LD	逻辑块与	---	---	---	153
---	OR LD	逻辑块或	---	---	---	155
---	OUT	输出	---	---	!OUT	166
---	OUT NOT	输出非	---	---	!OUT NOT	167
---	SET	置位	@SET	%SET	!SET	175
---	RSET	复位	@RSET	%RSET	!RSET	175
---	TIM	定时器	---	---	---	207
---	TIMX	定时器	---	---	---	207
---	CNT	计数器	---	---	---	231
000	NOP	空操作	---	---	---	187
001	END	结束	---	---	---	186
002	IL	联锁	---	---	---	187
003	ILC	联锁解除	---	---	---	187
004	JMP	跳转	---	---	---	191
005	JME	跳转结束	---	---	---	191
006	FAL	故障报警	@FAL	---	---	934
007	FALS	严重故障	---	---	---	942
008	STEP	步定义	---	---	---	808
009	SNXT	步启动	---	---	---	808
010	SFT	移位寄存器	---	---	---	309
011	KEEP	保持	---	---	!KEEP	168
012	CNTR	可逆计数器	---	---	---	234
013	DIFU	上升沿微分	---	---	!DIFU	173
014	DIFD	下降沿微分	---	---	!DIFD	173
015	TIMH	高速计数器	---	---	---	211
016	WSFT	字移位	@WSFT	---	---	316
017	ASFT	异步移位寄存器	@ASFT	---	---	313
019	MCMP	多字比较	@MCMP	---	---	263
020	CMP	比较	---	---	!CMP	252
021	MOV	传送	@MOV	---	!MOV	279
022	MVN	取反传送	@MVN	---	---	281
023	BIN	BCD 码到二进制	@BIN	---	---	429
024	BCD	二进制到 BCD 码	@BCD	---	---	432
025	ASL	算术左移	@ASL	---	---	317
026	ASR	算术右移	@ASR	---	---	321
027	ROL	循环左移	@ROL	---	---	324
028	ROR	循环右移	@ROR	---	---	327
029	COM	求反	@COM	---	---	488
034	ANDW	逻辑与	@ANDW	---	---	474
035	ORW	逻辑或	@ORW	---	---	477

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
036	XORW	异或	@XORW	---	---	481
037	XNRW	异或非	@XNRW	---	---	485
040	STC	置进位	@STC	---	---	959
041	CLC	清进位	@CLC	---	---	960
045	TRSM	跟踪内存采样	---	---	---	930
046	MSG	显示信息	@MSG	---	---	913
058	BINL	双字 BCD 码到双字二进制	@BINL	---	---	430
059	BCDL	双字二进制到 BCD 码	@BCDL	---	---	433
060	CMPL	双字不带符号比较	---	---	---	254
062	XFRB	多位传送	@XFRB	---	---	290
063	LINE	列到行	@LINE	---	---	457
064	COLM	行到列	@COLM	---	---	459
065	SEC	小时到秒	@SEC	---	---	923
066	HMS	秒到小时	@HMS	---	---	925
067	BCNT	位计数器	@BCNT	---	---	513
068	BCMP	不带符号的块比较	@BCMP	---	---	268
069	APR	数学处理	@APR	---	---	497
070	XFER	块传送	@XFER	---	---	292
071	BSET	块设定	@BSET	---	---	295
072	ROOT	BCD 平方根	@ROOT	---	---	493
073	XCHG	数据交换	@XCHG	---	---	297
074	SLD	一个数字左移	@SLD	---	---	338
075	SRD	一个数字右移	@SRD	---	---	339
076	MLPX	数据译码	@MLPX	---	---	440
077	DMPX	数据编码	@DMPX	---	---	445
078	SDEC	七段译码	@SDEC	---	---	844
079	FDIV	浮点数除法	@FDIV	---	---	509
080	DIST	单字分配	@DIST	---	---	300
081	COLL	数据收集	@COLL	---	---	302
082	MOVB	位传送	@MOVB	---	---	285
083	MOVD	数字传送	@MOVD	---	---	287
084	SFTR	可逆移位寄存器	@SFTR	---	---	310
085	TCMP	表格比较	@TCMP	---	---	265
086	ASC	ASCII 码转换	@ASC	---	---	449
087	TTIM	累加定时器	---	---	---	219
088	ZCP	区域范围比较	---	---	---	274
090	SEND	网络发送	@SEND	---	---	879
091	SBS	子程序调用	@SBS	---	---	720
092	SBN	子程序入口	---	---	---	729
093	RET	子程序返回	---	---	---	732

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
094	WDT	延长最大扫描时间	@WDT	---	---	963
096	BPRG	块程序开始	---	---	---	983
097	IORF	I/O 刷新	@IORF	---	---	825
098	RECV	网络接收	@RECV	---	---	885
099	MCRO	宏	@MCRO	---	---	725
114	CPS	带符号二进制比较	---	---	!CPS	257
115	CPSL	带符号双字二进制比较	---	---	---	260
116	ZCPL	双字区范围比较	---	---	---	277
160	NEG	二进制求补	@NEG	---	---	435
161	NEGL	双字二进制求补	@NEGL	---	---	437
162	HEX	ASCII 到十六进制	@HEX	---	---	453
180	FCS	帧校验和	@FCS	---	---	656
181	SRCH	数据搜索	@SRCH	---	---	642
182	MAX	寻找最大值	@MAX	---	---	646
183	MIN	寻找最小值	@MIN	---	---	650
184	SUM	求和	@SUM	---	---	653
190	PID	PID 控制	---	---	---	675
191	PIDAT	PID 自动控制	---	---	---	686
194	SCL	标度	@SCL	---	---	704
195	AVG	求平均值	---	---	---	716
222	IORD	智能 I/O 读	@IORD	---	---	831
223	IOWR	智能 I/O 写	@IOWR	---	---	834
226	DLNK	CPU 总线单元 I/O 刷新	@DLNK	---	---	837
235	RXD	接收	@RXD	---	---	858
236	TXD	传送	@TXD	---	---	853
237	STUP	修改串行口设置	@STUP	---	---	863
260	PMCR	协议宏	@PMCR	---	---	844
269	FPD	故障点检测	---	---	---	950
281	EMBC	选择 EM 区	@EMBC	---	---	961
282	CCS	存入条件标志	@CCS	---	---	965
283	CCL	载入条件标志	@CCL	---	---	967
284	FRMCV	从 CV 转变地址	@FRMCV	---	---	968
285	TOCV	转变地址到 CV	@TOCV	---	---	972
287	IOSP	外设服务禁止	@IOSP	---	---	976
288	IORS	外设服务允许	---	---	---	978
300	AND =	与等于	---	---	---	246
300	LD =	装载等于	---	---	---	246
300	OR =	或等于	---	---	---	246
301	AND =L	与双字等于	---	---	---	246

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
301	LD =L	载入双字等于	---	---	---	246
301	OR =L	或双字等于	---	---	---	246
302	AND =S	与带符号等于	---	---	---	246
302	LD =S	载入带符号等于	---	---	---	246
302	OR =S	或带符号等于	---	---	---	246
303	AND =SL	与双字带符号等于	---	---	---	246
303	LD =SL	载入双字带符号等于	---	---	---	246
303	OR =SL	或双字带符号等于	---	---	---	246
305	AND <>	与不等于	---	---	---	246
305	LD <>	载入不等于	---	---	---	246
305	OR <>	或不等于	---	---	---	246
306	AND <>L	与双字不等于	---	---	---	246
306	LD <>L	载入双字不等于	---	---	---	246
306	OR <>L	或双字不等于	---	---	---	246
307	AND <>S	与带符号不等于	---	---	---	246
307	LD <>S	载入带符号不等于	---	---	---	246
307	OR <>S	或带符号不等于	---	---	---	246
308	AND <>SL	与带符号双字不等于	---	---	---	246
308	LD <>SL	载入带符号双字不等于	---	---	---	246
308	OR <>SL	或带符号双字不等于	---	---	---	246
310	AND <	与小于	---	---	---	246
310	LD <	载入小于	---	---	---	246
310	OR <	或小于	---	---	---	246
311	AND <L	与双字小于	---	---	---	246
311	LD <L	载入双字小于	---	---	---	246
311	OR <L	或双字小于	---	---	---	246
312	AND <S	与带符号小于	---	---	---	246
312	LD <S	载入带符号小于	---	---	---	246
312	OR <S	或带符号小于	---	---	---	246
313	AND <SL	与带符号双字小于	---	---	---	246
313	LD <SL	载入带符号双字小于	---	---	---	246
313	OR <SL	或带符号双字小于	---	---	---	246
315	AND <=	与小于等于	---	---	---	246
315	LD <=	载入小于等于	---	---	---	246
315	OR <=	或小于等于	---	---	---	246

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
316	AND <=L	与小于等于	---	---	---	246
316	LD <=L	载入小于等于	---	---	---	246
316	OR <=L	或小于等于	---	---	---	246
317	AND <=S	与带符号小于等于	---	---	---	246
317	LD <=S	载入带符号小于等于	---	---	---	246
317	OR <=S	或带符号小于等于	---	---	---	246
318	AND <=SL	与带符号双字小于等于	---	---	---	246
318	LD <=SL	载入带符号双字小于等于	---	---	---	246
318	OR <=SL	或带符号双字小于等于	---	---	---	246
320	AND >	与大于	---	---	---	246
320	LD >	载入大于	---	---	---	246
320	OR >	或大于	---	---	---	246
321	AND >L	与双字大于	---	---	---	246
321	LD >L	载入双字大于	---	---	---	246
321	OR >L	或双字大于	---	---	---	246
322	AND >S	与带符号大于	---	---	---	246
322	LD >S	载入带符号大于	---	---	---	246
322	OR >S	或带符号大于	---	---	---	246
323	AND >SL	与带符号双字大于	---	---	---	246
323	LD >SL	载入双字带符号大于	---	---	---	246
323	OR >SL	或双字带符号大于	---	---	---	246
325	AND >=	与大于等于	---	---	---	246
325	LD >=	载入大于等于	---	---	---	246
325	OR >=	或大于等于	---	---	---	246
326	AND >=L	与双字大于等于	---	---	---	246
326	LD >=L	载入双字大于等于	---	---	---	246
326	OR >=L	或双字大于等于	---	---	---	246
327	AND >=S	与带符号大于等于	---	---	---	246

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
327	LD >=S	载入带符号大于等于	---	---	---	246
327	OR >=S	或带符号大于等于	---	---	---	246
328	AND >=SL	与双字带符号大于等于	---	---	---	246
328	LD >=SL	载入双字带符号大于等于	---	---	---	246
328	OR >=SL	或双字带符号大于等于	---	---	---	246
329	AND =F	与浮点等于	---	---	---	557
329	LD =F	载入浮点等于	---	---	---	557
329	OR =F	或浮点等于	---	---	---	557
330	AND <>F	与浮点不等于	---	---	---	557
330	LD <>F	载入浮点不等于	---	---	---	557
330	OR <>F	或浮点不等于	---	---	---	557
331	AND <F	与浮点小于	---	---	---	557
331	LD <F	载入浮点小于	---	---	---	557
331	OR <F	或浮点小于	---	---	---	557
332	AND <=F	与浮点小于等于	---	---	---	557
332	LD <=F	载入浮点小于等于	---	---	---	557
332	OR <=F	或浮点小于等于	---	---	---	557
333	AND >F	与浮点大于	---	---	---	557
333	LD >F	载入浮点大于	---	---	---	557
333	OR >F	或浮点大于	---	---	---	557
334	AND >=F	与浮点大于等于	---	---	---	557
334	LD >=F	载入浮点大于等于	---	---	---	557
334	OR >=F	或浮点大于等于	---	---	---	557
335	AND =D	与双字浮点等于	---	---	---	614
335	LD =D	载入双字浮点等于	---	---	---	614
335	OR =D	或双字浮点等于	---	---	---	614
336	AND <>D	与双字浮点不等于	---	---	---	614
336	LD <>D	载入双字浮点不等于	---	---	---	614

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
336	OR <>D	或双字浮点不等于	---	---	---	614
337	AND <D	与双字浮点小于	---	---	---	614
337	LD <D	载入双字浮点小于	---	---	---	614
337	OR <D	或双字浮点小于	---	---	---	614
338	AND <=D	与双字浮点小于等于	---	---	---	614
338	LD <=D	载入双字浮点小于等于	---	---	---	614
338	OR <=D	或双字浮点小于等于	---	---	---	614
339	AND >D	与双字浮点大于	---	---	---	614
339	LD >D	载入双字浮点大于	---	---	---	614
339	OR >D	或双字浮点大于	---	---	---	614
340	AND >=D	与双字浮点大于等于	---	---	---	614
340	LD >=D	载入双字浮点大于等于	---	---	---	614
340	OR >=D	或双字浮点大于等于	---	---	---	614
350	AND TST	与位测试	---	---	---	163
350	LD TST	载入位测试	---	---	---	163
350	OR TST	或位测试	---	---	---	163
351	AND TSTN	与位测试非	---	---	---	163
351	LD TSTN	载入位测试非	---	---	---	163
351	OR TSTN	或位测试非	---	---	---	163
400	+	无进位带符号二进制加法	@+	---	---	373
401	+L	无进位双字带符号二进制加法	@+L	---	---	375
402	+C	有进位带符号二进制加法	@+C	---	---	377
403	+CL	有进位双字带符号二进制加法	@+CL	---	---	379
404	+B	无进位 BCD 加法	@+B	---	---	384
405	+BL	无进位双字 BCD 加法	@+BL	---	---	382
406	+BC	有进位 BCD 加法	@+BC	---	---	384
407	+BCL	有进位双字 BCD 加法	@+BCL	---	---	386
410	–	无进位带符号二进制减法	@–	---	---	387
411	–L	无进位双字带符号二进制减法	@–L	---	---	389

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
412	–C	有进位带符号二进制减法	@–C	---	---	393
413	–CL	有进位双字带符号二进制减法	@–CL	---	---	395
414	–B	无进位 BCD 减法	@–B	---	---	398
415	–BL	无进位双字 BCD 减法	@–BL	---	---	399
416	–BC	有进位 BCD 减法	@–BC	---	---	403
417	–BCL	有进位双字 BCD 减法	@–BCL	---	---	404
420	*	带符号二进制乘法	@*	---	---	406
421	*L	双字带符号二进制乘法	@*L	---	---	408
422	*U	不带符号二进制乘法	@*U	---	---	410
423	*UL	双字不带符号二进制乘法	@*UL	---	---	412
424	*B	BCD 乘法	@*B	---	---	413
425	*BL	双字 BCD 乘法	@*BL	---	---	415
430	/	带符号二进制除法	@/	---	---	417
431	/L	带符号双字二进制除法	@/L	---	---	419
432	/U	不带符号二进制除法	@/U	---	---	421
433	/UL	不带符号双字二进制除法	@/UL	---	---	423
434	/B	BCD 除法	@/B	---	---	425
435	/BL	双字 BCD 除法	@/BL	---	---	427
448	FSTR	浮点变为 ASCII	@FSTR	---	---	561
449	FVAL	ASCII 变为浮点	@FVAL	---	---	566
450	FIX	浮点到 16 位	@FIX	---	---	520
451	FIXL	浮点到 32 位	@FIXL	---	---	522
452	FLT	16 位到浮点	@FLT	---	---	523
453	FLTL	32 位到浮点	@FLTL	---	---	525
454	+F	浮点加法	@+F	---	---	527
455	–F	浮点减法	@–F	---	---	529
456	*F	浮点乘法	@*F	---	---	531
457	/F	浮点除法	@/F	---	---	533
458	RAD	度到弧度	@RAD	---	---	554
459	DEG	弧度到度	@DEG	---	---	536
460	SIN	正弦	@SIN	---	---	538
461	COS	余弦	@COS	---	---	540
462	TAN	正切	@TAN	---	---	542
463	ASIN	反正弦	@ASIN	---	---	544

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
464	ACOS	反余弦	@ACOS	---	---	546
465	ATAN	反余切	@ATAN	---	---	548
466	SQRT	平方根	@SQRT	---	---	550
467	EXP	指数	@EXP	---	---	552
468	LOG	对数	@LOG	---	---	554
470	BINS	带符号 BCD 到二进制	@BINS	---	---	462
471	BCDS	带符号二进制到 BCD	@BCDS	---	---	468
472	BISL	带符号双字 BCD 到二进制	@BISL	---	---	465
473	BDSL	带符号双字二进制到 BCD	@BDSL	---	---	470
486	SCL2	标度 2	@SCL2	---	---	708
487	SCL3	标度 3	@SCL3	---	---	712
490	CMND	发布命令	@CMND	---	---	890
498	MOVL	双字传送	@MOVL	---	---	282
499	MVNL	双字取反传送	@MVNL	---	---	284
502	BCMP2	扩展块比较	@BCMP2	---	---	270
510	CJP	条件跳转	---	---	---	195
511	CJPN	条件跳转	---	---	---	195
512	FOR	FOR-NEXT 循环	---	---	---	201
513	NEXT	FOR-NEXT 循环	---	---	---	201
514	BREAK	中断循环	---	---	---	204
515	JMP0	多路跳转	---	---	---	199
516	JME0	多路跳转结束	---	---	---	199
520	NOT	非	---	---	---	161
521	UP	条件 ON	---	---	---	162
522	DOWN	条件 OFF	---	---	---	162
530	SETA	多位置位	@SETA	---	---	177
531	RSTA	多位复位	@RSTA	---	---	177
532	SETB	单位置位	@SETB	---	!SETB	180
533	RSTB	单位复位	@RSTB	---	!RSTB	180
534	OUTB	单位输出	@OUTB	---	!OUTB	184
540	TMHH	1 毫秒定时器	---	---	---	216
542	TIML	长定时器	---	---	---	222
543	MTIM	多路输出定时器	---	---	---	226
545	CNR	定时器 / 计数器复位	@CNR	---	---	238
546	CNTX	计数器	---	---	---	231
547	CNRX	定时器 / 计数器复位	---	---	---	238
548	CNTRX	可逆计数器	---	---	---	234
550	TIMX	定时器	---	---	---	207
551	TIMHX	高速定时器	---	---	---	211
552	TMHHX	1 毫秒定时器	---	---	---	216
553	TIMLX	长定时器	---	---	---	222
554	MTIMX	多路输出定时器	---	---	---	226
555	TTIMX	累加定时器	---	---	---	219
560	MOVR	传送到寄存器	@MOVR	---	---	304

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
561	MOVRW	定时器 / 计数器 PV 传送到寄存器	@MOVRW	---	---	306
562	XCGL	双字数据交换	@XCGL	---	---	298
570	ASLL	双字左移	@ASLL	---	---	319
571	ASRL	双字右移	@ASRL	---	---	322
572	ROLL	双字循环左移	@ROLL	---	---	326
573	RORL	双字循环右移	@RORL	---	---	329
574	RLNC	无进位循环左移	@RLNC	---	---	331
575	RRNC	无进位循环右移	@RRNC	---	---	334
576	RLNL	无进位双字循环左移	@RLNL	---	---	332
577	RRNL	无进位双字循环右移	@RRNL	---	---	336
578	NSFL	N 位数据左移	@NSFL	---	---	341
579	NSFR	N 位数据右移	@NSFR	---	---	343
580	NASL	左移 N 位	@NASL	---	---	345
581	NASR	右移 N 位	@NASR	---	---	350
582	NSLL	双字左移 N 位	@NSLL	---	---	348
583	NSRL	双字右移 N 位	@NSRL	---	---	353
590	++	二进制递增	@++	---	---	356
591	++L	双字二进制递增	@++L	---	---	358
592	– –	二进制递减	@– –	---	---	360
593	– –L	双字二进制递减	@– –L	---	---	362
594	++B	BCD 递增	@++B	---	---	364
595	++BL	双字 BCD 递增	@++BL	---	---	366
596	– –B	BCD 递减	@– –B	---	---	368
597	– –BL	双字 BCD 递减	@– –BL	---	---	370
600	SIGN	带符号二进制 16 位到 32 位	@SIGN	---	---	439
610	ANDL	双字逻辑与	@ANDL	---	---	476
611	ORWL	双字逻辑或	@ORWL	---	---	479
612	XORL	双字异或	@XORL	---	---	483
613	XNRL	双字异或非	@XNRL	---	---	486
614	COML	双字求反	@COML	---	---	490
620	ROTB	二进制平方根	@ROTB	---	---	491
630	SSET	设置堆栈	@SSET	---	---	623
631	DIM	定维记录	@DIM	---	---	635
632	PUSH	推入堆栈	@PUSH	---	---	626
633	FIFO	先进先出	@FIFO	---	---	629

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
634	LIFO	后进先出	@LIFO	---	---	632
635	SETR	设置记录位置	@SETR	---	---	638
636	GETR	获得计录号	@GETR	---	---	640
637	SWAP	交换字节	@SWAP	---	---	644
638	SNUM	读堆栈大小	@SNUM	---	---	659
639	SREAD	读堆栈数据	@SREAD	---	---	662
640	SWRIT	写堆栈数据	@SWRIT	---	---	665
641	SINS	插入堆栈数据	@SINS	---	---	668
642	SDEL	删除堆栈数据	@SDEL	---	---	671
650	LEN$	串长	@LEN$	---	---	1026
652	LEFT$	取左串	@LEFT$	---	---	1018
653	RGHT$	取右串	@RGHT$	---	---	1020
654	MID$	取中间串	@MID$	---	---	1022
656	+$	链接串	@+$	---	---	1015
657	INS$	插入串	@INS$	---	---	1037
658	DEL$	删除串	@DEL$	---	---	1031
660	FIND$	寻找串	@FIND$	---	---	1024
661	RPLC$	取代串	@RPLC$	---	---	1028
664	MOV$	串传送	@MOV$	---	---	1013
665	XCHG$	交换串	@XCHG$	---	---	1033
666	CLR$	清除串	@CLR$	---	---	1035
670	AND =$	与串等于	---	---	---	1040
670	LD =$	载入串等于	---	---	---	1040
670	OR =$	或串等于	---	---	---	1040
671	AND <>$	与串不等于	---	---	---	1040
671	LD <>$	载入串不等于	---	---	---	1040
671	OR <>$	或串不等于	---	---	---	1040
672	AND <$	与串小于	---	---	---	1040
672	LD <$	载入串小于	---	---	---	1040
672	OR <$	或串小于	---	---	---	1040
673	AND <=$	与串小于等于	---	---	---	1040
673	LD <=$	载入串小于等于	---	---	---	1040
673	OR <=$	或串小于等于	---	---	---	1040
674	AND >$	与串大于	---	---	---	1040
674	LD >$	载入串大于	---	---	---	1040
674	OR >$	或串大于	---	---	---	1040
675	AND >=$	与串大于等于	---	---	---	1040
675	LD >=$	载入串大于等于	---	---	---	1040

助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
675	OR >=$	或串大于等于	---	---	---	1040
680	LMT	限位控制	@LMT	---	---	696
681	BAND	静带控制	@BAND	---	---	698
682	ZONE	静域控制	@ZONE	---	---	701
690	MSKS	设置中断屏蔽	@MSKS	---	---	744
691	CLI	清中断	@CLI	---	---	755
692	MSKR	读中断屏蔽	@MSKR	---	---	750
693	DI	禁止中断	@DI	---	---	760
694	EI	允许中断	---	---	---	762
700	FREAD	读数据文件	@FREAD	---	---	899
701	FWRIT	写数据文件	@FWRIT	---	---	906
730	CADD	日历加法	@CADD	---	---	916
731	CSUB	日历减法	@CSUB	---	---	920
735	DATE	时钟调整	@DATE	---	---	928
750	GSBS	调用全局子程序	@GSBS	---	---	732
751	GSBN	进入全局子程序	---	---	---	740
752	GRET	返回全局子程序	---	---	---	743
801	BEND	块程序结束	---	---	---	983
802	IF	条件分支块	---	---	---	988
802	IF	条件分支块	---	---	---	988
802	IF NOT	条件分支块非	---	---	---	988
803	ELSE	ELSE	---	---	---	988
804	IEND	结果结束	---	---	---	988
805	WAIT	一个循环与等待	---	---	---	994
805	WAIT	一个循环与等待	---	---	---	994
805	WAIT NOT	一个循环与等待非	---	---	---	994
806	EXIT	条件块退出	---	---	---	991
806	EXIT	条件块退出	---	---	---	991
806	EXIT NOT	条件块退出非	---	---	---	991
809	LOOP	循环	---	---	---	1007
810	LEND	循环结束	---	---	---	1007
810	LEND	循环结束	---	---	---	1007
810	LEND NOT	循环结束非	---	---	---	1007
811	BPPS	块程序暂停	---	---	---	985
812	BPRS	块程序重新启动	---	---	---	985
813	TIMW	定时器等待	---	---	---	998

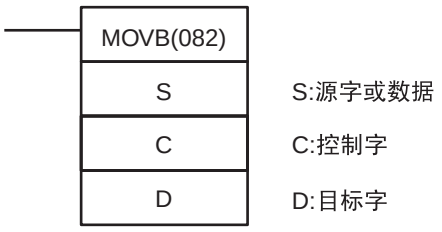
助记符	指令	函数代码	上升沿微分	下降沿微分	立即刷新功能	页
814	CNTW	计数器等待	---	---	---	1001
815	TMHW	高速定时器等待	---	---	---	1004
816	TIMWX	定时器等待	---	---	---	998
817	TMHWX	高速定时器等待	---	---	---	1004
818	CNTWX	计数器等待	---	---	---	1001
820	TKON	任务 ON	@TKON	---	---	1045
821	TKOF	任务 OFF	@TKOF	---	---	1049
840	PWR	指数幂	@PWR	---	---	556
841	FIXD	双字浮点到 16 位二进制	@FIXD	---	---	577
842	FIXLD	双字浮点到 32 位二进制	@FIXLD	---	---	578
843	DBL	16 位二进制到双字浮点	@DBL	---	---	580
844	DBLL	32 位二进制到双字浮点	@DBLL	---	---	581
845	+D	双字浮点加法	@+D	---	---	583
846	–D	双字浮点减法	@–D	---	---	585
847	*D	双字浮点乘法	@*D	---	---	587
848	/D	双字浮点除法	@/D	---	---	589
849	RADD	双字度到弧度	@RADD	---	---	591
850	DEGD	双字弧度到度	@RADD	---	---	593
851	SIND	双字正弦	@SIND	---	---	594
852	COSD	双字余弦	@COSD	---	---	596
853	TAND	双字正切	@TAND	---	---	598
854	ASIND	双字反正弦	@ASIND	---	---	600
855	ACOSD	双字反余弦	@ACOSD	---	---	602
856	ATAND	双字反正切	@ATAND	---	---	604
857	SQRTD	双字平方根	@SQRTD	---	---	606
858	EXPD	双字指数	@EXPD	---	---	608
859	LOGD	双字对数	@LOGD	---	---	610
860	PWRD	双字指数幂	@PWRD	---	---	612
880	INI	模式控制	@INI	---	---	769
881	PRV	高速计数器 PV 读	@PRV	---	---	773
882	CTBL	载入比较表	@CTBL	---	---	777
885	SPED	速度输出	@SPED	---	---	781
886	PULS	设置脉冲	@PULS	---	---	786
887	PLS2	脉冲输出	@PLS2	---	---	789
888	ACC	加速控制	@ACC	---	---	795
889	ORG	寻找原点	@ORG	---	---	802
891	PWN	可变占空比脉冲	@PWN	---	---	805

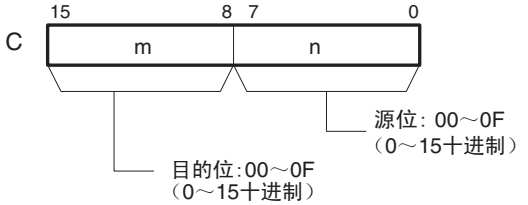
第 3 章
指令

本章介绍了 CS/CJ 系列 PLC 编程所用的每条指令。指令按功能分类，如*第 2 章指令一览的分类*。

3-1 指令符号编排说明

按功能分组描述指令。*参考 2-3 按助记符字母顺序排列指令表。*
每个功能的描述排列成如下表所示的形式。

<table>
<tr><th colspan="2">项目</th><th>内容</th></tr>
<tr><td colspan="2">名称和助记符</td><td>每节的标题包括指令的名称及助记符，在括弧中为函数代码。例：位传送：
MOVB(082)</td></tr>
<tr><td colspan="2">用途</td><td>节标题后是指令基本用途的描述。</td></tr>
<tr><td colspan="2">梯形图符号和操作数名称</td><td>以下所给的位传送指令的例子显示CX-programmer中用于表达指令的梯形图符号。每个操作数的名称和梯形图符号一起表示出来。
 
</td></tr>
<tr><td rowspan="2">变化</td><td>变化</td><td rowspan="2">在特殊条件下能用于控制指令执行的变化，用助记符表示。任何指令不支持的变化都用 “不支持”表示。
• ON 条件每个循环执行：只要指令接收到 ON 执行条件，它就执行。
• 上升沿微分执行一次：仅当执行条件从 OFF 到 ON 变化时，在下一个循环内执行指令。
• 下降沿微分执行一次：仅当执行条件从 ON 到 OFF 变化时，在下一个循环内执行指令。
• 一直执行：指令不需要一个执行条件，每个循环都执行。
• 产生 ON 条件…，指令在每个循环执行，并为下一个指令产生一个执行条件。
<table><tr><td rowspan="3">变化</td><td>ON 条件每个循环执行</td><td>MOVB(082)</td></tr><tr><td>上升沿微分执行一次</td><td>@MOVB(082)</td></tr><tr><td>下降沿微分执行一次</td><td>不支持</td></tr></table></td></tr>
<tr><td>变化</td><td>变化</td></tr>
<tr><td></td><td>立即刷新功能</td><td>对有些指令可指定立即刷新，使指令执行时刷新 I/O。如果支持立即刷新，那么使用助记符形式给出功能。如果一个指令不支持立即刷新，那么就给出 “不支持”。
<table><tr><td>立即刷新功能</td><td>不支持</td></tr></table></td></tr>
<tr><td colspan="2">适用程序区</td><td>指定能使用指令的程序区。“OK”表示指令能在该区域使用
<table><tr><th>块程序区</th><th>步程序区</th><th>子程序</th><th>中断任务</th></tr><tr><td>OK</td><td>OK</td><td>OK</td><td>OK</td></tr></table></td></tr>
</table>

<table>
<tr><th>项目</th><th>内容</th></tr>
<tr><td>操作数</td><td>如有必要，给出所用指定操作数的字和位的意义，如控制字。
 
</td></tr>
<tr><td>操作数规定</td><td>每个操作数可使用的内存地址以下述表格形式列出。左边用于列标题的字母和梯形图符号所使用的一样。当一个区不能指定位操作数时，用“·····.”表示。
<table>
<tr><th>区域</th><th>S</th><th>C</th><th>D</th></tr>
<tr><td>CIO 区</td><td colspan="3">CIO 0000 ～ CIO 6143</td></tr>
<tr><td>工作区</td><td colspan="3">W000 ～ W511</td></tr>
<tr><td>保持位区</td><td colspan="3">H000 ～ H511</td></tr>
<tr><td>辅助位区</td><td colspan="2">A000 ～ A959</td><td>A448 ～ A959</td></tr>
<tr><td>定时器区</td><td colspan="3">T0000 ～ T4095</td></tr>
<tr><td>计数器区</td><td colspan="3">C0000 ～ C4095</td></tr>
<tr><td>DM 区</td><td colspan="3">D00000 ～ D32767</td></tr>
<tr><td>无区号 EM 区</td><td colspan="3">E00000 ～ E32767</td></tr>
</table></td></tr>
<tr><td>描述</td><td>描述了指令功能和指令中所用的操作数。</td></tr>
<tr><td>标志</td><td>标志表指示了指令执行后条件标志的状态。没列出的任何标志不受指令的影响。
“OFF”表示：不管执行指令的结果任何，指令执行后标志位立即变为 OFF。
<table>
<tr><th>名称</th><th>标记</th><th>操作</th></tr>
<tr><td>错误标志</td><td>ER</td><td>控制数据在范围内时为 ON。
其它情况为 OFF。</td></tr>
<tr><td>等于标志</td><td>=</td><td>OFF</td></tr>
<tr><td>负标志</td><td>N</td><td>OFF</td></tr>
</table></td></tr>
<tr><td>注意</td><td>提供使用指令时所需要的注意事项。一定要阅读并遵循这些注意事项。</td></tr>
<tr><td>例</td><td>提供以指定操作数使用指令的一个例子，以进一步解释指令的功能。</td></tr>
</table>

常数

操作数以常数输入如下表所示。

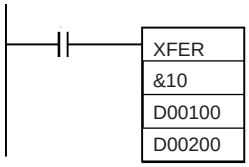
操作数描述及操作数规定

- 指定位串的操作数（通常以十六进制形式输入）：指定位串的操作数仅以十六进制形式给定，例如，对 MOV(021) 指令， S 操作数仅用“#0000 ～ #FFFF”指定。然而 CX-Programmer 中，位串也可以用前缀 & 表示的十进制形式输入。
- 指定数值的操作数（通常以十进制形式输入，包括跳转号）：指定数值的操作数以十进制数和十六进制形式给定：例如，给 XFER(070) 指令的 N 操作数是：“# 0000 ～# FFFF”和 “&0 ～ &65535”。

- 表示控制号的操作数（除跳转号外）：控制号以十进制形式给定，例如：给 SBS(091) 指令的 N 操作数十 “0 ～ 1023”。

例

在此例子中，使用 CX-Programmer 记号给定一个常数，例如，以带有前缀 & 的十进制给定一个指定数值的操作数，如下例所示：



用编程设备输入常数的方法如下表所示。

操作数	CX-Programmer	编程器
指定位串的操作数（通常以十六进制形式输入）	带有前缀 & 作为十进制数输入，或带有前缀 # 作为十六进制数输入。（见注）	按 Cont/# 键输入十六进制值，缺省前缀 # 。按 CHG 键能在十六进制 （用 + /- ）和不带符号的十进制 （& 前缀）之间切换。
指定数字值的操作数（通常以十进制形式输入）		
指定控制号的操作数（除跳转号外）	带有前缀 # 作为十进制数输入。（见注）	直接以十进制形式输入。 如果自动加上前缀 &，按 CHG 键能在不带符号的十进制 （前缀 &），十六进制 （前缀 #），和带符号的十进制 （用 + /- ）之间切换。如果没显示前缀，那么输入值必须是十进制形式。

注 当在 CX-Programmer 上输入操作数时，输入范围以及适当的前缀将显示。

条件标志

本章中用编程器标记作为条件标志。用 CX-Programmer 时，条件标志预先登录为全局符号，在符号名前加 “P _”。

标志	编程器标记	CX-Programmer 标记
错误标志	ER	P_ER
访问错误标志	AER	P_AER
进位标志	CY	P_CY
大于标志	>	P_GT
等于标志	=	P_EQ
小于标志	<	P_LT
负标志	N	P_N
上溢标志	OF	P_OF
下溢标志	UF	P_UF
大于或等于标志	>=	P_GE
不等于标志	<>	P_NE

标志	编程器标记	CX-Programmer 标记
小于或等于标志	<=	P_LE
常 ON 标志	ON	P_On
常 OFF 标志	OFF	P_Off

3-2 指令更新和新指令

本节列出了对 CS1 带 -EV1 后缀的 CPU 单元和 CS1-H/CJ1-H CPU 单元所更新的指令。

3-2-1 对 CS 系列 1 版本 CPU 单元的更新指令

对 CPU 单元 1 版本，下列指令已经更新，更详细的情况参考指定页码。
功能仅仅支持 1 版本 CPU 单元 （以 -EV1 标注）。

名称	助记忆符	功能代码	功能	更新	页码
读数据文件	FREAD	700	支持 CSV 和 TEXT。数据格式 （以前仅支持二进制数据）	通过增加数据格式 , 进位返回和指定进位 , 控制数据内容已经改变。	899
写数据文件	FWRIT	701			906
发布命令	CMND	490	CPU 单元能发送 FINS 命令到自身 （以前不支持）。	能够发送 FINS 命令到 CPU 单元，增加了 CMND(490) 功能。	890

3-2-2 对 CS1-H/CJ1-H CPU 单元的更新指令

新指令

下列指令已经添加到 CS1-H/CJ1-H CPU 单元。

顺序输出指令

单个位设置， SETB(532)
单个位复位， RSTB(533)
单个位输出， OUTB(534)

数据比较指令

区域范围比较， ZCP(088)
双字区域范围比较， ZCPL(116)

浮点计算和变换指令

浮点数据比较指令：=F, <>F, <F, <=F, >F, 和 >=F (329 ～ 334)

浮点到 ASCII， FSTR(448)
ASCII 到浮点， VAL(449)

双精度浮点计算和变换指令

双精度浮点数据比较指令：=D, <>D, <D, <=D, >D, 和 >=D (335 ～ 340)

双字浮点到 16 位二进制：FIXD(841)
双字浮点到 32 位二进制：FIXLD(8420)
16 位二进制到双字浮点：DBL(843)
32 位二进制到双字浮点：DBLL(844)
双字浮点加法：+D(845)
双字浮点减法：–D(846)

双字浮点乘法：*D(847)
双字浮点除法：/D(848)
双字度到弧度：RADD(849)
双字弧度到度：DEGD(850)
双字正弦：SIND(851)
双字余弦：COSD(852)
双字正切：TAND(853)
双字反正弦：ASIND(854)
双字反余弦：ACOSD(855)
双字反正切：ATAND(856)
双字平方根：SQRTD(857)
双字指数：EXPD(858)
双字对数：LOGD(859)
双字指数幂：PWRD(860)

表格处理指令

读堆栈大小：SNUM(638)
读堆栈数据：SREAD(639)
写堆栈数据：SWRIT(640)
插入堆栈数据：SINS(641)
删除堆栈数据：SDEL(642)

数据控制指令

PID 自动控制：PIDAT(191)

子程序指令

调用全局子程序：GSBS(750)
进入全局子程序：GSBN(751)
全局子程序返回：GRET(752)

I/O 单元指令

CPU 总线单元 I/O 刷新：DLNK(226)

其它指令

存储条件标志：CCS(282)
载入条件标志：CCL(283)
从 CV 转变地址：FRMCV(284)
转变地址到 CV：TOCV(285)
外设服务禁止：IOSP(287)
外设服务允许：IORS(288)

新指令

CS1-H 和 CJ1-H CPU 单元已经更新下列指令。

特殊数学指令

数学处理：APR(069)

故障诊断指令

故障报警：FAL(006)
严重故障报警：FALS(007)

3-3 顺序输入指令

3-3-1 加载：LD

用途

表明一个逻辑开始，并且根据指定的操作位 ON/OFF 状态创建一个 ON/OFF 执行条件。

梯形图符号

变化

变化	每个循环操作位为 ON 时逻辑开始并产生 ON	LD
	在上升沿时逻辑开始产生 ON 一次	@LD
	在下降沿时逻辑开始产生 ON 一次	%LD
立即刷新功能 （见注）		!LD
组合变化	刷新输入位，逻辑开始并在上升沿时产生 ON 一次（见注）	!@LD
	刷新输入位，逻辑开始并在下降沿时产生 ON 一次（见注）	!%LD

注 CS1D CPU 单元不支持立即刷新功能。

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	LD 位操作数
CIO 区	CIO 000000 ～ CIO 614315
工作区	W00000 ～ W51115
保持位区	H00000 ～ H51115
辅助位区	A00000 ～ A95915
定时器区	T0000 ～ T4095
计数器区	C0000 ～ C4095
任务标志区	TK0000 ～ TK0031
条件标志	ER, CY, N, OF, UF, >, =, <, >=, <>, <=, A1, A0
时钟脉冲	0.0 2s, 0.1 s, 0.2 s, 1 s, 1 min
TR 区	TR0 ～ TR15
DM 区	---
无区号 EM 区	---
有区号 EM 区	---
二进制间接 DM/EM 地址	---
BCD 间接 DM/EM 地址	---
常数	---
数据寄存器	---

区域	LD 位操作数
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) , –(– –)IR0 ～ ,–(– –)IR15

描述

LD 用于从母线开始的第一个常开位或者一个逻辑块的第一个常开位。如果没有立即刷新功能，那么读 I/O 内存的指定位。如果有立即刷新功能，那么读并使用基本输入单元的输入端的状态。

在下述情况中， LD 用作表示一个逻辑开始的指令。

- 直接联到母线。
- 用 AND LD 或 OR LD 连接逻辑块，即在逻辑块起始处。

AND LOAD 和 OR LOAD 指令用于用 LD 或 LD NOT 开始的逻辑块的串接或并接。

当相关的输出不能直接联到母线时，执行条件至少需要一个 LOAD 或 LOAD NOT 指令。如果没有 LOAD 或 LOAD NOT 指令，在外设检查程序时将出现程序错误。

当用 AND LOAD 或 OR LOAD 指令联接逻辑块时， AND LOAD/OR LOAD 指令的总数必须等于 LOAD/LOAD NOT 指令的总数减 1。如果不相等，将会出现编程错误。详细资料参阅 *3-3-7AND LD* 和 *3-3-8OR LD 指令*。

标志

此指令不影响任何标志。

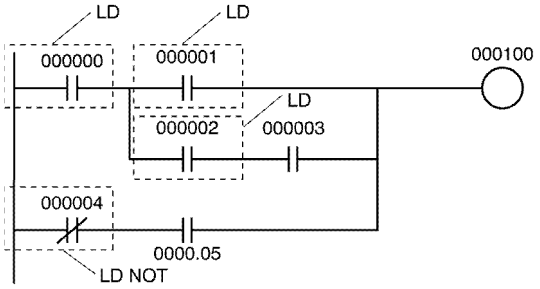
注意

LD 指令有上升沿微分 (@) 或下降沿微分 (%)。如果指定了上升沿微分 , 那么执行条件在操作位状态从 OFF 到 ON 时仅变 ON 一个循环。如果指定了下降沿微分，那么执行条件在操作位状态从 ON 到 OFF 时仅变为 ON 一个循环。

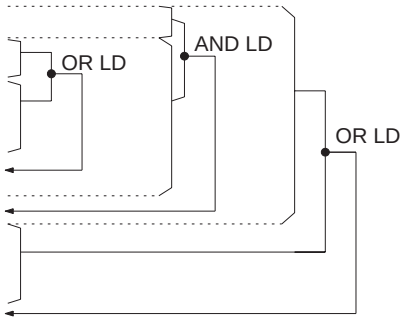
LD 指令还有立即刷新 (!) 功能。对于基本输入单元 （但不包括从机架上的基本输入单元或 C200H 组 2 的多点输入单元），一个立即刷新功能指令就在指令执行前刷新输入位的状态。

对 LD 指令，还能够把立即刷新和上、下沿微分组合使用 （! @ 或 !%）。如果指定了其中一个，那么就在执行指令前，刷新基本输入单元的输入，并且在状态从 OFF 变 ON 或从 ON 变 OFF 后，执行条件仅变为 ON 一个循环。

例



指令	操作数
LD	000000
LD	000001
LD	000002
AND	000003
OR LD	---
AND LD	---
LD NOT	000004
AND	000005
OR LD	---
OUT	000100



3-3-2 加载非：LD NOT

用途

表明一个逻辑开始，并且根据把一个指定操作位的 ON/OFF 状态取反来建立一个 ON/OFF 的执行条件。

梯形图符号

变化

变化	每个循环操作位为 OFF 时，逻辑开始并产生 ON	LD NOT
	在上升沿时逻辑开始产生 ON 一次 （见注 1）	@LD NOT
	在下降沿时逻辑开始产生 ON 一次 （见注 1）	%LD NOT
立即刷新功能 （见注 2）		!LD NOT
组合变化	刷新输入位，开始逻辑，并在上升沿产生 ON 一次 （见注 1）	!@LD NOT
	刷新输入位，开始逻辑，并在下降沿产生 ON 一次 （见注 1）	!%LD NOT

注

1. 下面变量仅对 CS1-H,CJ1-H 或 CJ1M CPU 单元适用： @LD NOT， %LD NOT，!@LD NOT，和 !%LD NOT。
2. CS1D CPU 单元不支持立即刷新功能。

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数据规定

区域	LD NOT 位操作数
CIO 区	CIO 000000 ～ CIO 614315
工作区	W00000 ～ W51115
保持位区	H00000 ～ H51115
辅助位区	A00000 ～ A95915
定时器区	T0000 ～ T4095
计数器区	C0000 ～ C4095
任务标志区	TK0000 ～ TK0031
条件标志	ER, CY, N, OF, UF, >, =, <, >=, <>, <=, ON, OFF, AER
时钟脉冲	0.0 2s, 0.1 s, 0.2 s, 1 s, 1 min
TR 区	---
DM 区	---
无区号 EM 区	---
有区号 EM 区	---
二进制间接 DM/EM 地址	---
BCD 间接 DM/EM 地址	---
常数	---
数据寄存器	---
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

LD NOT 用于从母线开始的第一个常闭位或者一个逻辑块的第一个常闭位。如果没有立即刷新功能，那么读 I/O 内存的指定位并取反。如果有立即刷新功能，那么读基本输入单元的输入端的状态并取反使用。

在下述情况中， LD NOT 用作表示一个逻辑开始的指令。

- 直接联到母线。
- 用 AND LD 或 OR LD 连接的逻辑块 （即在逻辑块起始处）。

AND LD 和 OR LOAD 指令用于用 LD 或 LD NOT 开始的逻辑块的串接或并接。

当相关的输出不能直接联到母线时，执行条件至少需要一个 LOAD 或 LOAD NOT 指令。如果没有 LOAD 或 LOAD NOT 指令，在外设检查程序时将出现程序错误。

当用 AND LOAD 或 OR LOAD 指令联接逻辑块时， AND LOAD/OR LOAD 指令的总数必须等于 LOAD/LOAD NOT 指令的总数减 1。如果不相等，将会出现编程错误。

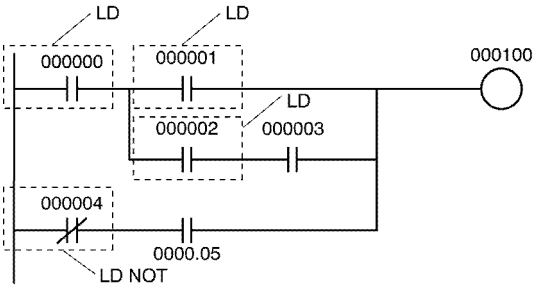
标志

此指令不影响任何标志。

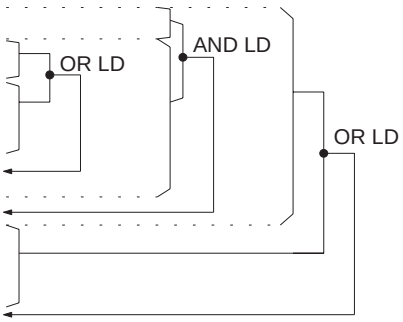
注意

LD NOT 还有立即刷新 (!) 功能。对于基本输入单元（但不包括从机架上的基本输入单元或 C200H 组 2 的多点输入单元），一个立即刷新指令就在指令执行前刷新输入位的状态。

例



指令	操作数
LD	000000
LD	000001
LD	000002
AND	000003
OR LD	---
AND LD	---
LD NOT	000004
AND	000005
OR LD	---
OUT	000100



3-3-3 与：AND

用途

把指定的操作位状态和当前的执行条件逻辑与。

梯形图符号

变化

变化	每个循环与结果为 ON 时产生 ON	AND
	在上升沿时产生 ON 一次	@AND
	在下降沿时产生 ON 一次	%AND
立即刷新功能 （见注）		!AND
组合变化	刷新输入位，并在上升沿产生 ON 一次 （见注）	!@AND
	刷新输入位，并在下降沿产生 ON 一次 （见注）	!%AND

注 CS1D CPU 单元不支持立即刷新功能。

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数据规定

区域	AND 位操作数
CIO 区	CIO 000000 ～ CIO 614315
工作区	W00000 ～ W51115
保持位区	H00000 ～ H51115
辅助位区	A00000 ～ A95915
定时器区	T0000 ～ T4095
计数器区	C0000 ～ C4095
任务标志区	TK0000 ～ TK0031
条件标志	ER, CY, N, OF, UF, >, =, <, >=, <>, <=, ON, OFF, AER
时钟脉冲	0.02 s, 0.1 s, 0.2 s, 1 s, 1 min
TR 区	---
DM 区	---
无区号 EM 区	---
有区号 EM 区	---
二进制间接 DM/EM 地址	---
BCD 间接 DM/EM 地址	---
常数	---
数据寄存器	---
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

AND 用于常开位串联连接。 AND 不能直接连到母线，并且不能用作一个逻辑块的开始。如果没有立即刷新功能，那么读 I/O 内存的指定位。如果有立即刷新功能，那么读基本输入单元的输入端的状态。

标志

此指令不影响任何标志。

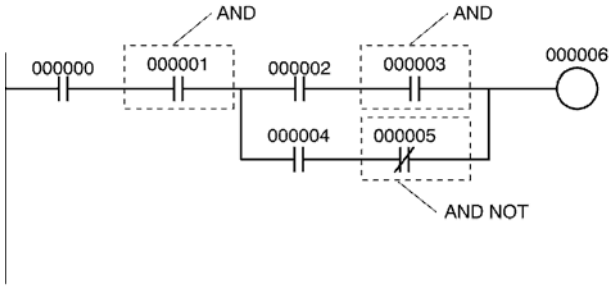
注意

AND 指令有上升沿微分 (@) 或下降沿微分 (%) 。如果指定了上升沿微分 , 那么执行条件在操作位状态从 OFF 到 ON 时仅变 ON 一个循环。如果指定了下降沿微分，那么执行条件在操作位状态从 ON 到 OFF 时仅改变 ON 一个循环。

AND 指令还有立即刷新 (!) 功能。对于基本输入单元 （但不包括从机架上的基本输入单元或 C200H 组 2 的多点输入单元），一个立即刷新功能指令就在指令执行前刷新输入位的状态。

对 AND 还能够把立即刷新和上、下沿微分组合使用 （! @ 或 !%）。如果指定了其中一个，那么就在执行指令前，刷新基本输入单元的输入，并且在状态从 OFF 变 ON 或从 ON 变 OFF 后，执行条件仅变 ON 一个循环。

例



指令	操作数
LD	000000
AND	000001
LD	000002
AND	000003
LD	000004
AND NOT	000005
OR LD	---
AND LD	---
OUT	000006

3-3-4 与非：AND NOT

用途

把指定的操作位状态取反并且和当前的执行条件逻辑与。

梯形图符号

变化

变化	每个循环 “与非”结果为 ON 时并产生 ON	AND NOT
	在上升沿时产生 ON 一次 （见注 1）	@AND NOT
	在下降沿时产生 ON 一次 （见注 1）	%AND NOT
立即刷新功能 （见注 2）		!AND NOT
组合变化	刷新输入位，并在上升沿产生 ON 一次 （见注 1）	!@AND NOT
	刷新输入位，并在下降沿产生 ON 一次 （见注 1）	!%AND NOT

注 1. 下面变量仅对 CS1-H， CJ1-H 或 CJ1M CPU 单元适用： @AND NOT， %AND NOT， !@AND NOT， !%AND NOT。

2. CS1D CPU 单元不支持立即刷新功能。

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数据规定

区域	AND NOT 位操作数
CIO 区	CIO 000000 ～ CIO 614315
工作区	W00000 ～ W51115
保持位区	H00000 ～ H51115
辅助位区	A00000 ～ A95915
定时器区	T0000 ～ T4095
计数器区	C0000 ～ C4095

区域	AND NOT 位操作数
任务标志区	TK0000 ～ TK0031
条件标志	ER, CY, N, OF, UF, >, =, <, >=, <>, <=, ON, OFF, AER
时钟脉冲	0.02 s, 0.1 s, 0.2 s, 1 s, 1 min
TR 区	---
DM 区	---
无区号 EM 区	---
有区号 EM 区	---
二进制间接 DM/EM 地址	---
BCD 间接 DM/EM 地址	---
常数	---
数据寄存器	---
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

AND NOT 用于常闭位串接。ANDNOT 不能直接连到母线，并且不能用作一个逻辑块的开始。如果没有立即刷新功能，那么读 I/O 内存的指定位并取反。如果有立即刷新功能，那么读基本输入单元的输入端的状态。

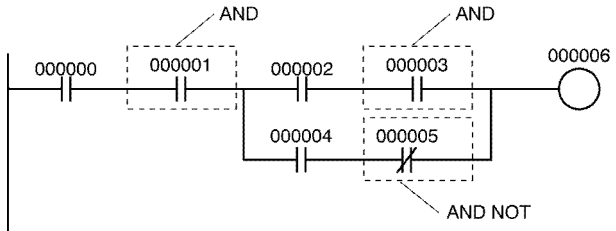
标志

此指令不影响任何标志。

注意

AND NOT 能指定立即刷新 (!) 功能。对于基本输入单元（但不包括从机架上的基本输入单元或 C200H 组 2 的多点输入单元），一个立即刷新指令就在指令执行前刷新输入位的状态。

例



指令	操作数
LD	000000
AND	000001
LD	000002
AND	000003
LD	000004
AND NOT	000005
OR LD	---
AND LD	---
OUT	000006

3-3-5 或：OR

用途

把指定的操作位状态 ON/OFF 状态和当前的执行条件逻辑或。

梯形图符号

变化

变化	每个循环 “或”结果为 ON 时产生 ON	OR
	在上升沿时产生 ON 一次	@OR
	在下降沿时产生 ON 一次	%OR
立即刷新功能 （见注）		!OR
组合变化	刷新输入位，并在上升沿产生 ON 一次 （见注）	!@OR
	刷新输入位，并在下降沿产生 ON 一次 （见注）	!%OR

注 CS1D CPU 单元不支持立即刷新功能。

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数据规定

区域	OR 位操作数
CIO 区	CIO 000000 ～ CIO 614315
工作区	W00000 ～ W51115
保持位区	H00000 ～ H51115
辅助位区	A00000 ～ A95915
定时器区	T0000 ～ T4095
计数器区	C0000 ～ C4095
任务标志区	TK0000 ～ TK0031
条件标志	ER, CY, N, OF, UF, >, =, <, >=, <>, <=, ON, OFF, AER
时钟脉冲	0.02 s, 0.1 s, 0.2 s, 1 s, 1 min
DM 区	---
无区号 EM 区	---
有区号 EM 区	---
二进制间接 DM/EM 地址	---
BCD 间接 DM/EM 地址	---
常数	---
数据寄存器	---
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15

描述

OR 用于常开位并联连接。一个常开位和一个用 LOAD 或 LOAD NOT 指令（连到母线或逻辑块开始处）开始的逻辑块形成一个逻辑或。如果没有立即刷新功能，那么读 I/O 内存的指定位。如果有立即刷新功能，那么读基本输入单元的输入端的状态。

标志

此指令不影响任何标志。

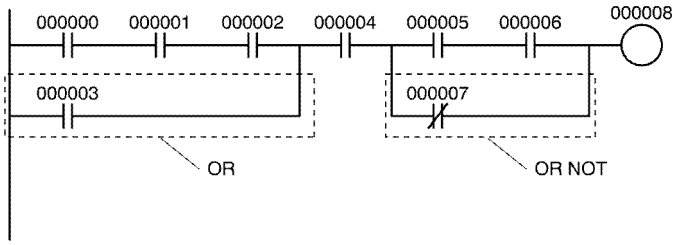
注意

OR 指令有上升沿微分 (@) 或下降沿微分 (%) 。如果指定了上升沿微分，那么执行条件在操作位状态从 OFF 到 ON 时仅变 ON 一个循环。如果指定了下降沿微分，那么执行条件在操作位状态从 ON 到 OFF 时仅变 ON 一个循环。

OR 指令还有立即刷新 (!) 功能。对于基本输入单元（但不包括从机架上的基本输入单元或 C200H 组 2 的多点输入单元），一个立即刷新功能指令就在指令执行前刷新输入位的状态。

对 OR 指令，还能够把立即刷新和上、下沿微分组合使用 （! @ 或 !%）。如果指定了其中一个，那么就在执行指令前，刷新基本输入单元的输入，并且在状态从 OFF 变 ON 或从 ON 变 OFF 后，执行条件仅变 ON 一个循环。

例



指令	操作数
LD	000000
AND	000001
AND	000002
OR	000003
AND	000004
LD	000005
AND	000006
OR NOT	000007
AND LD	---
OUT	000008

3-3-6 或非：OR NOT

用途

把指定的位的状态取反并且和当前的执行条件逻辑或。

梯形图符号

变化

变化	每个循环 “或非”结果为 ON 时产生 ON	OR NOT
	在上升沿时产生 ON 一次 （见注 1）	@OR NOT
	在下降沿时产生 ON 一次 （见注 1）	%OR NOT
立即刷新功能 （见注 2）		!OR NOT
组合变化	刷新输入位，并在上升沿产生 ON 一次 （见注 1）	!@OR NOT
	刷新输入位，并在下降沿产生 ON 一次 （见注 1）	!%OR NOT

注 1. 下面变量仅对 CS1-H，CJ1-H 或 CJ1M CPU 单元适用： @OR NOT，%OR NOT， !@OR NOT， !%OR NOT。

2. CS1D CPU 单元不支持立即刷新功能。

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数据规定

区域	OR NOT 位操作数
CIO 区	CIO 000000 ～ CIO 614315
工作区	W00000 ～ W51115
保持位区	H00000 ～ H51115
辅助位区	A00000 ～ A95915
定时器区	T0000 ～ T4095
计数器区	C0000 ～ C4095
任务标志区	TK0000 ～ TK0031
条件标志	ER, CY, N, OF, UF, >, =, <, >=, <>, <=, A1, A0
时钟脉冲	0.02 s, 0.1 s, 0.2 s, 1 s, 1 min
TR 区	---
DM 区	---
有区号 EM 区	---
二进制间接 DM/EM 地址	---
BCD 间接 DM/EM 地址	---
常数	---
数据寄存器	---
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

OR 用于常闭位并联连接。一个常闭位和一个用 LOAD 或 LOAD NOT 指令（连到母线或逻辑块开始处）开始的逻辑块形成一个逻辑或。如果没有立即刷新功能，那么读 I/O 内存的指定位并取反。如果有立即刷新功能，那么读基本输入单元的输入端的状态。

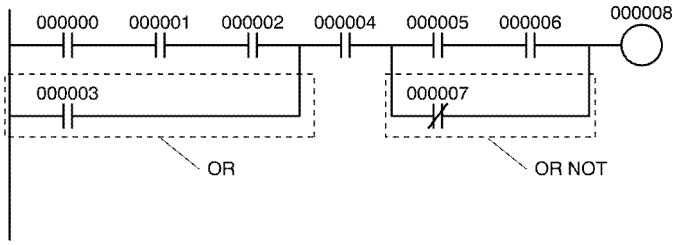
标志

此指令不影响任何标志。

注意

OR NOT 能指定立即刷新 (!) 功能。对于基本输入单元 （单不包括从机架上的基本输入单元或 C200H 组 2 的多点输入单元），一个立即刷新指令就在指令执行前刷新输入位的状态。

例



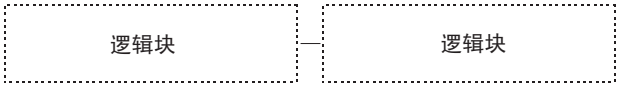
指令	操作数
LD	000000
AND	000001
AND	000002
OR	000003
AND	000004
LD	000005
AND	000006
OR NOT	000007
AND LD	---
OUT	000008

3-3-7 逻辑块与：AND LD

用途

逻辑块之间作一逻辑与。

梯形图符号



变化

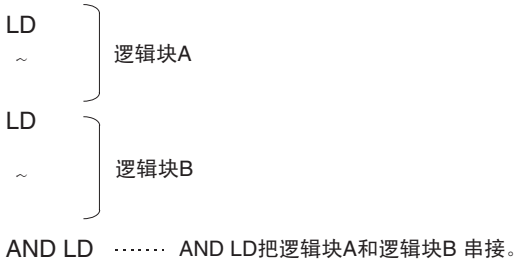
变化	每个循环 “与”结果为 ON 时产生 ON	AND LD
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

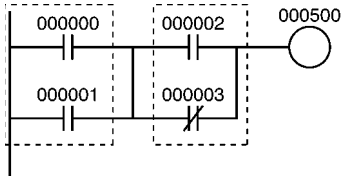
描述

AND LD 把指令前的二个逻辑块串接。



一个逻辑块包含从 LOAD 或 LOAD NOT 指令开始到同一梯级中下一个 LOAD 或 LOAD NOT 指令前的所有指令。

在下图中，用虚线表示两个逻辑块。这个例子显示：当左边逻辑块的任何一个条件位 ON 时（即当 CIO000000 或 CIO000001 为 ON 时）并且当右边逻辑块任何一个执行条件为 ON 时，（即当 CIO 000002 为 ON 或者 CIO 000003 为 OFF 时）将产生一个 ON 执行条件。



标志

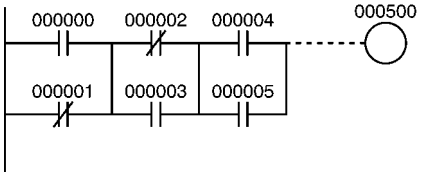
此指令不影响任何标志。

注意

使用此指令（AND LD），可以串联三个或更多个的逻辑块。首先连接两个逻辑块，然后按顺序连接下一个逻辑块，在三个或更多的逻辑块后面还能继续使用此指令，并且串联它们。

当用 AND LOAD 或 OR LOAD指令连接一个逻辑块时，AND LOAD/OR LOAD 指令的数目等于 LOAD/LOAD NOT 指令的总数减 1。如果他们不匹配，将会出现程序错误。

例



编程例 (1)

指令	操作数
LD	000000
OR NOT	000001
LD NOT	000002
OR	000003
AND LD	---
LD	000004
OR	000005
AND LD	---
. .	. .
OUT	000500

编程例 (2)

指令	操作数
LD	000000
OR NOT	000001
LD NOT	000002
OR	000003
LD	000004
OR	000005
. .	. .

指令	操作数
AND LD	---
AND LD	---
. .	. .
OUT	000500

AND LD 指令可以反复使用。用 (2) 的方法编程时， AND LOAD 指令的数目比其前面的 LOAD 和 LOAD NOT 指令的数目少 1。

用 (2) 的方法编程时，确保 AND LOAD 前面的 LOAD 及 LOAD NOT 指令的总数不超过 8 个。若要用 9 个或 9 个以上时，请用方法 (1)。若用方法 (2) 编程时其总数超过 9 个，那么当用外设检查程序时，将会出现程序错误。

编程

地址	指令	操作数
000000	LD	000000
000001	OR	000001
000002	LD	000002
000003	OR NOT	000003
000004	AND LD	---
000005	OUT	000500

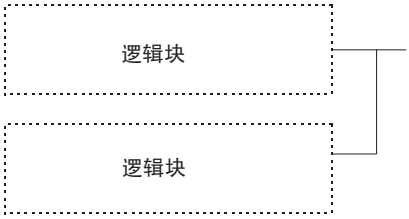
第二个 LD：在与前面程序块串联连接的下一个程序块的起始位处使用。

3-3-8 逻辑块或：OR LD

用途

逻辑块之间作一逻辑或。

梯形图符号



变化

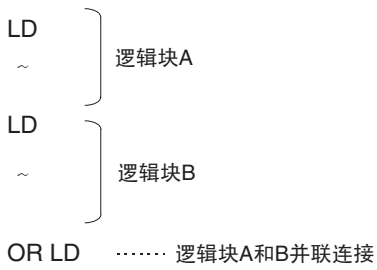
变化	每个循环 “或”结果为 ON 时产生 ON	OR LD
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

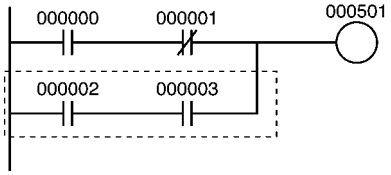
描述

OR LD 把指令前的二个逻辑块并联。



一个逻辑块包含从 LOAD 或 LOAD NOT 指令开始到同一梯级中下一个 LOAD 或 LOAD NOT 指令前的所有指令。

下图在上、下逻辑块之间需要一个 OR LOAD 指令，当 CIO000000 为 ON 时并且 CIO000001 为 OFF 时，或当 CIO 000002 和 CIO 000003 都为 ON 时，将产生一个 ON 执行条件。除了把当前执行条件和最后未用执行条件相或之外，OR LOAD 指令的操作和助记符与 AND LOAD 的相同。



标志

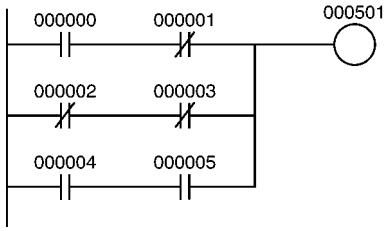
此指令不影响任何标志。

注意

使用此指令，可以并接三个或更多个的逻辑块。首先连接两个逻辑块，然后按顺序连接下一个逻辑块，在三个或更多的逻辑块后面还能继续使用此指令，并且并联它们。

当用 AND LOAD 或 OR LOAD指令连接一个逻辑块时，AND LOAD/OR LOAD 指令的数目等于 LOAD/LOAD NOT 指令的总数减 1。如果他们不匹配，将会出现程序错误。

例



编程例 (1)

指令	操作数
LD	000000
AND NOT	000001
LD NOT	000002
AND NOT	000003
OR LD	---
LD	000004
AND	000005
OR LD	---
. .	. .
OUT	000501

编程例 (2)

指令	操作数
LD	000000
AND NOT	000001
LD NOT	000002
AND NOT	000003
LD	000004
AND	000005
. .	. .
OR LD	---
OR LD	---
. .	. .
OUT	000501

OR LD 指令可以反复使用。用 (2) 的方法编程时， OR LOAD 指令的数目比其前面的 LOAD 和 LOAD NOT 指令的数目少 1。

用 (2) 的方法编程时，确保 OR LOAD 前面的 LOAD 及 LOAD NOT 指令的总数不超过 8 个。若要用 9 个或 9 个以上时，请用方法 (1)。若用方法 (2) 编程时其总数超过 9 个，那么当用外设检查程序时，将会出现程序错误。

编程

地址	指令	操作数
000100	LD	000000
000101	AND NOT	000001
000102	LD	000002
000103	AND	000003
000104	OR LD	---
000105	OUT	000501

第二个 LD：在与前面程序块并联连接的下一个程序块的起始位处使用。

3-3-9 微分和立即刷新指令

LOAD， AND 和 OR 指令除了它的标准形式外，还有微分和立即刷新变化，并且还有两种组合有效。

LOAD NOT， AND NOT， OR NOT， OUT 和 OUT NOT 指令除了它们的标准形式外，还具有立即刷新的变化。

对于常规和微分指令，立即刷新指令和立即刷新微分指令，其指令处理 I/O 数据的时序不同。

常规和微分指令执行时，使用的是前次 I/O 刷新处理的数据输入，并且其结果随着下一次的 I/O 处理而输出。这里 “I/O 刷新”意思在 CPU 的内存和 I/O 单元之间的数据交换。

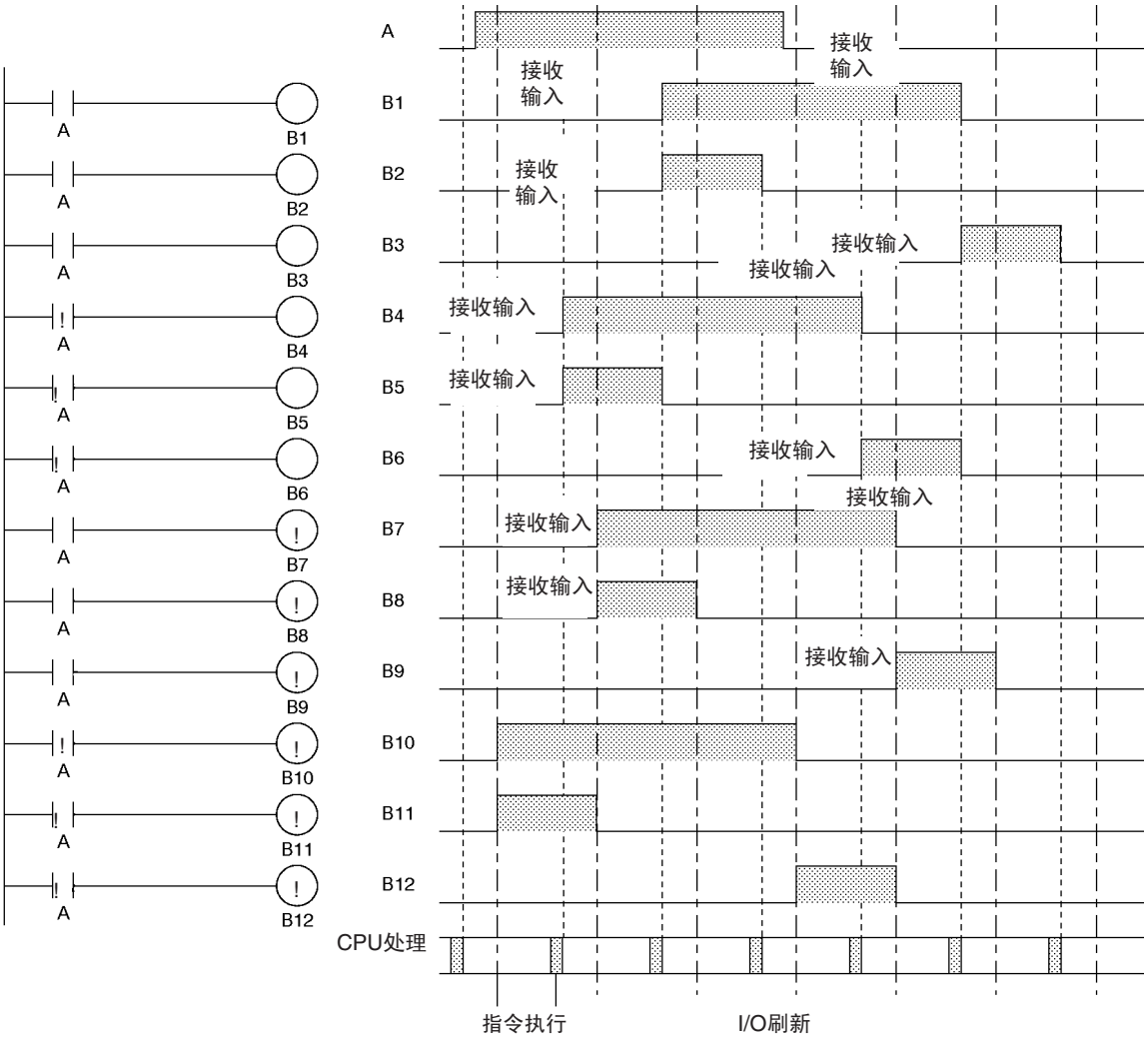
除了以上 I/O 刷新外，一个立即刷新指令是对指令所访问字的 I/O 单元进行数据交换，一个立即刷新指令同时刷新包括指定位的八个位 （最左或最右八位）。

立即刷新指令不能用于从机架上的单元。

指令变化	助记符	功能	I/O 刷新
常规	LD,AND,OR,LD NOT ,AND NOT,OR NOT	指定位的 ON/OFF 状态由 CPU 循环刷新，并且反映到下一次的指令执行。	循环刷新
	OUT,OUT NOT	指令执行后，指定位的 ON/OFF 状态在下一次循环刷新时输出。	
上升沿微分	@LD,@AND,@OR	当指定位从 OFF 变为 ON 时，指令执行一次，并且 ON 状态维持一个循环	
下降沿微分	%LD,%AND,%OR	当指定位从 ON 变为 OFF 时，指令执行一次，并且 ON 状态维持一个循环	
立即刷新	!LD,!AND,!OR,!LD NOT,!AND NOT, !OR NOT	当指定位的输入数据由 CPU 读出，并执行指令。	指令执行前
	!OUT,!OUT NOT	指令执行后，输出指定位的数据。	指令执行后
上升沿微分 / 立即刷新	!@LD,!@AND,!@OR	由 CPU 刷新指定位输入数据，并当位从 OFF 变为 ON 时，指令执行一次，且 ON 状态维持一个循环。	指令执行前
下降沿微分 / 立即刷新	!%LD,!%AND,!%OR	由 CPU 刷新指定位输入数据，并当位从 ON 变为 OFF 时，指令执行一次，且 ON 状态维持一个循环。	

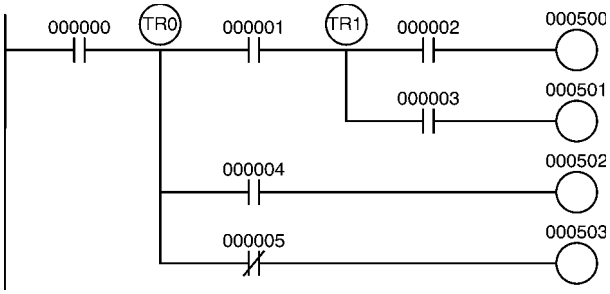
3-3-10 I/O 指令操作时序

下图显示一个从 LD 和 OUT 组成的程序中，指令操作的不同时序。



3-3-11 TR 位

当用助记符编程时， TR 位用于暂时保存程序中执行条件的 ON/OFF 状态。直接用梯形图形式编程时，则不用 TR，因为该处理由外设自动地执行。下例显示使用 TR 位地简单用法。

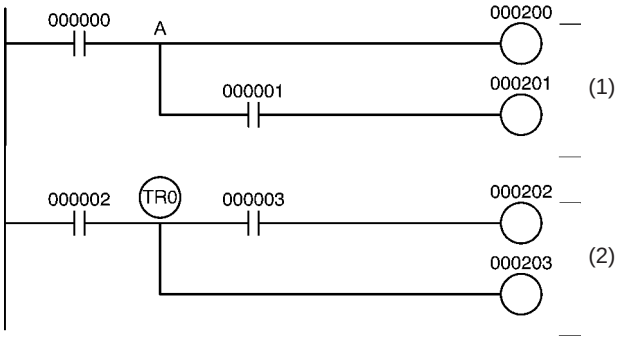


地址	指令	操作数
000000	LD	000000
000001	OUT	TR0
000002	AND	000001
000003	OUT	TR1
000004	AND	000002
000005	OUT	000500
000006	LD	TR1
000007	AND	000003
000008	OUT	000501
000009	LD	TR0
000010	AND	000004
000011	OUT	000502
000012	LD	TR0
000013	AND NOT	000005
000014	OUT	000503

TR0 ～ TR15 的使用

TR0 ～ TR15 仅与 LOAD 和 OUTPUT 指令一起使用。对这些位地址的使用次序没有限制。

有时重写程序后可简化程序，这样就不需要 TR 位。在下图给出了一种需要 TR 位和不需要 TR 位的情况。



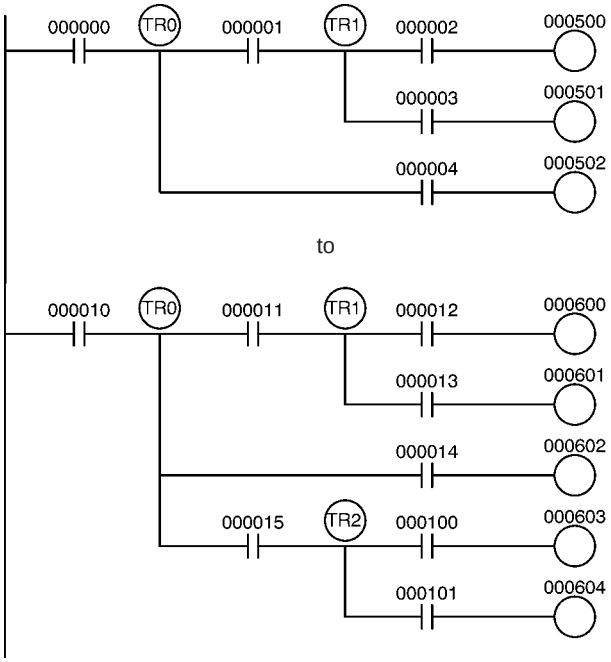
在指令块 (1) 中，A 点的 ON/OFF 状态与输出 CIO 00200 的状态相同，因此可编人 AND 000001 和 OUT 000201，而无需用一个 TR 位编程。在指令块 (2) 中，分支点的状态和输出 CIO 000202 的状态相同，所以必须使用一个 TR 位。对于这种情况，用指令块 (1) 代替指令块 (2) 可以减少程序步数。

TR0 ～ TR15 注意事项

在有多个输出分支的程序中，TR 位仅用于保持 (OUT TR0 ～ TR15) 和恢复 (LD TR0 ～ TR15) 分支点的有许多点的 ON/OFF 状态。它们不同于一般的位，不能和 AND 或 OR 指令及包括 NOT 的指令一起用。

TR0 ～ TR15 输出重复

在下图所示的多输出分支中，在同一程序块内 TR 位地址不能重复使用，然而，它可在不同的程序块中使用。



3-3-12 NOT: NOT(520)

用途

执行条件取反。

梯形图符号

变化

变化	每个循环把执行条件取反	NOT(520)
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

描述

NOT(520) 放在一个执行条件和另一个指令之间把转执行条件取反。

标志

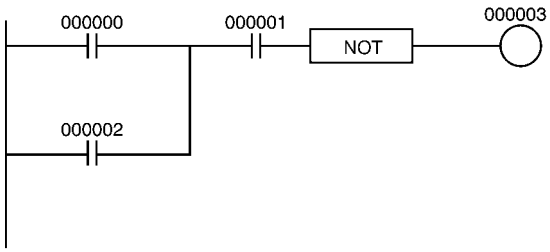
NOT(520) 不影响任何标志。

注意

NOT(520) 是一个中间指令，即它不能当作右侧指令使用。NOT(520) 后面需编一个右侧指令。

举例

在下例中， NOT(520) 把执行条件取反。



下表显示这个程序段的运行。

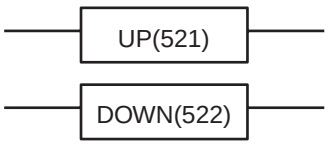
输入位状态			输出位状态
CIO000000	CIO000001	CIO000002	CIO000003
1	1	1	0
1	1	0	0
1	0	1	1
0	1	1	0
1	0	0	1
0	1	0	1
0	0	1	1
0	0	0	1

3-3-13 条件 ON/OFF：UP(521) 和 DOWN(522)

用途

当 UP(521) 所接收的执行条件从 OFF 变为 ON 时，使下一个指令的执行条件变 ON 一个循环。当 DOWN(522) 所接收的执行条件从 ON 变为 OFF 时，使下一个指令的执行条件变 ON 一个循环。

梯形图符号



变化

变化	上升沿微分产生 ON 一次	UP(521)
立即刷新功能		不支持

变化	下降沿微分产生 ON 一次	UP(522)
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

描述

UP(521) 放在一个执行条件和另一个指令之间把执行条件变成一个上升沿微分条件。当执行条件从 OFF 变为 ON 时， UP(521) 使连接的指令执行一次。

DOWN(522) 放在一个执行条件和另一个指令之间把执行条件变成一个下降沿微分条件。当执行条件从 ON 变为 OFF 时， DOWN(522) 使连接的指令执行一次。

DIFU(013) 和 DIFD(014) 指令也有同样的用途，但是他们需要工作位。UP(521) 和 DOWN(522) 由于减少了工作位数和所用的程序地址而简化了编程。

标志

UP(521) 和 DOWN(522) 不影响任何标志。

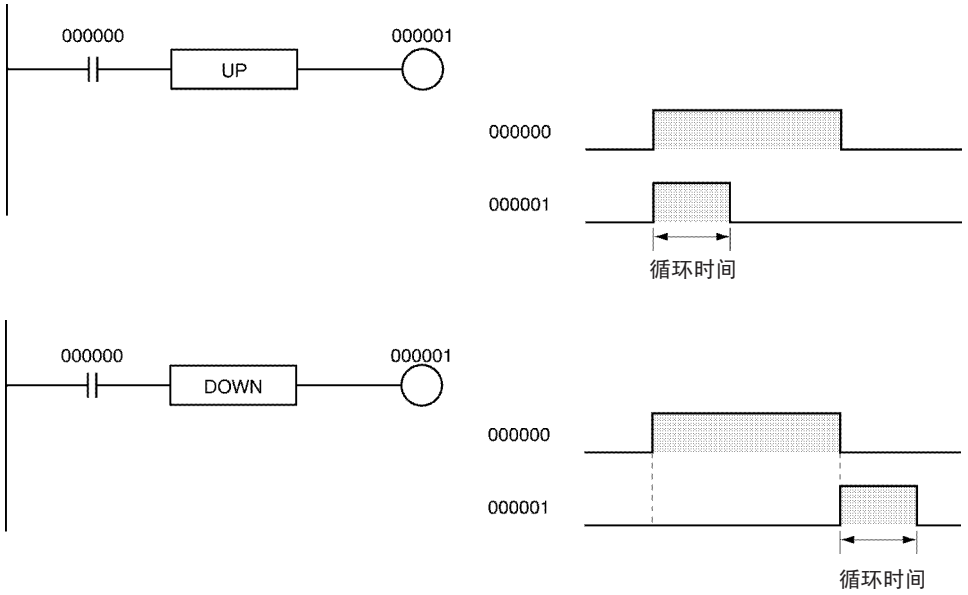
注意

UP(521) 和 DOWN(522) 是一个中间指令，即它们不能当作右侧指令使用。UP(521) 和 DOWN(522) 后面需编一个右侧指令。

当它编在联锁程序段，跳转程序段或一个子程序中时，UP(521) 和 DOWN(522) 的运行不仅取决于指令的执行条件，还取决于程序段中的执行条件。详细情况*参考 3-5-3 节联锁和联锁清除：IL(002) 和 ILG(003)， 3-5-4 节跳转和跳转结束 :JMP(004) 和 JME(005)*，以及 *3-20 中断控制指令*章节。

举例

在下例中，当CIO 000000从OFF变为ON时，CIO 000001只变ON一个循环。



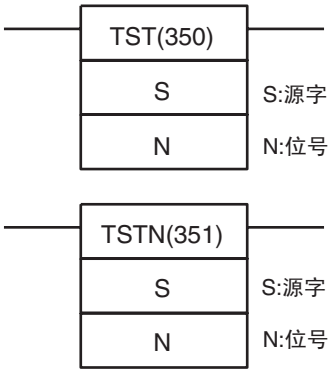
3-3-14 位测试：TST(350) 和 TSTN(351)

用途

LD TST(350)，AND TST(350) 和 OR TST(350) 在程序中使用类似于 LD，AND 和 OR 指令，当指定字中的指定位为 ON 时执行条件为 ON，反之为 OFF。

LD TSTN(351)， AND TSTN(351) 和 OR TSTN(351) 在程序中使用类似于 LD NOT，AND NOT 和 OR NOT 指令：当指定字中的指定位为 ON 时执行条件为 OFF，反之为 ON。

梯形图符号



变化

变化	每个循环执行	TST(350)
立即刷新功能		不支持

变化	每个循环执行	TSTN(351)
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

N: 位号

位号必须在 0000 ～ 000F （十六进制数）或 &0000 ～ &0015 （十进制数）之间。当指定一个字的地址时，仅这个字的最右位（0～F十六进制数）是有效的。

操作数规定

区	S	N
CIO 区	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W511	
保持位区	H000 ～ H511	
辅助位区	A000 ～ A959	
定时器区	T0000 ～ T4095	
计数器区	C0000 ～ C4095	
DM 区	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	#0000 ～ #000F（二进制）或 &0 ～ &15
数据寄存器	DR0 ～ DR15	
索引寄存器	---	
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 , IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

描述

LD TST(350)， AND TST(350) 和 OR TST(350) 在程序中的使用类似于 LD，AND 和 OR，当指定字中的指定位为 ON 时，执行条件为 ON，反之为 OFF。不象 LD， AND 和 OR，在 TST(350) 中 DM 和 EM 区中的位可用作操作数。

LD TSTN(351)，AND TSTN(351) 和 OR TSTN(351) 在程序中的使用类似于 LD NOT， AND NOT 和 OR NOT，当指定字中的指定位为 ON 时，执行条件为 OFF，反之为 ON。不象 LD NOT， AND NOT 和 OR NOT，在 TSTN(351) 中 DM 和 EM 区中的位可用作操作数。

标志

名称	标记	操作
错误标志	ER	OFF 或不变 （见注）

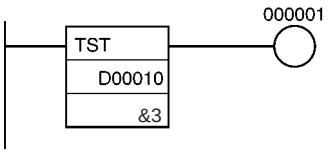
名称	标记	操作
等于标志	=	OFF 或不变 （见注）
负标志	N	OFF 或不变 （见注）

注 在 CS1 和 CJ1 CPU 单元中，这些变为 OFF。
在 CS1-H 和 CJ1-H CPU 单元中，这些标志保持不变。

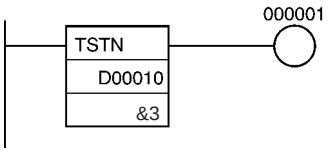
注意 TST(350)和TSTN(351)是中间指令，即它们不能作为右侧指令使用。TST(350)或 TSTN(351) 后面必须编一个右侧指令。

例 LD TST(350) 和 LD TSTN(351)

在下例中，当 D00010 的位 3 变为 ON 时， CIO 000001 变为 ON。

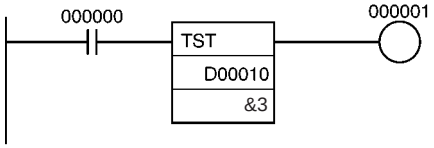


在下例中，当 D00010 的位 3 变为 OFF 时， CIO 000001 变为 ON。

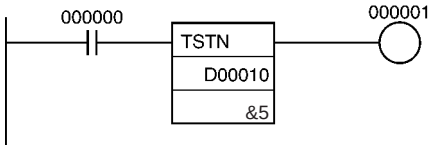


AND TST(350) 和 AND TSTN(351)

在下例中，当CIO 000000和D00010的位3都为ON时，CIO 000001变为ON。

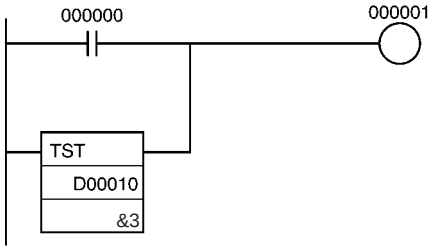


在下例中，当 CIO 000000 为 ON 和 D00010 的位 5 位 OFF 时， CIO 000001 变为 ON。

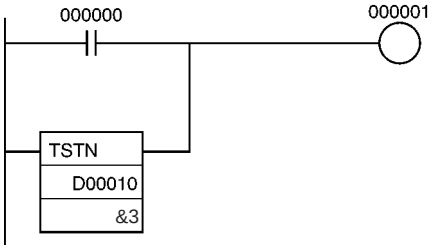


OR TST(350) 和 OR TSTN(351)

在下例中，当 CIO 000000 或 D00010 的位 3 为 ON 时，CIO 000001 变为 ON。



在下例中，当 CIO 000000 为 ON 或 D00010 的位 3 位 OFF 时， CIO 000001 变为 ON。



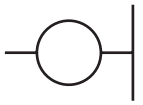
3-4 顺序输出指令

3-4-1 输出：OUT

用途

把逻辑处理的结果 （执行条件）输出到指定的位。

梯形图符号



变化

变化	ON 条件时每次循环执行	OUT
	上升沿微分产生 ON 一次	不支持
	下降沿微分产生 ON 一次	不支持
立即刷新功能 （见注）		!OUT

注 CS1D CPU 单元不支持立即刷新功能。

适用程序区

块程序区	步程序区	子程序	中断任务
不允许	OK	OK	OK

操作数规定

区域	输出位操作数
CIO 区	CIO 000000 ～ CIO 614315
工作区	W00000 ～ W51115
保持位区	H00000 ～ H51115
辅助位区	A44800 ～ A95915
定时器区	---
计数器区	---
TR 区	TR0 ～ TR15
DM 区	---
无区号 EM 区	---
有区号 EM 区	---
二进制间接 DM/EM 地址	---
BCD 间接 DM/EM 地址	---
常数	---
数据寄存器	---

区域	输出位操作数
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ ,IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

如果没有立即刷新功能，那么把执行条件的状态写到 I/O 内存的指定位。如果有立即刷新功能，那么把执行条件的状态写到基本输出单元的输出端及 I/O 内存的输出位。

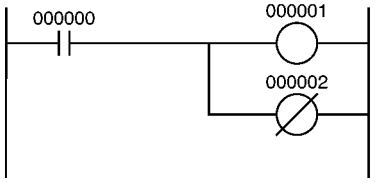
标志

此指令不影响任何标志。

注意

OUT 和 OUT NOT 可指定立即刷新 (!)。一个立即刷新指令在对基本输出单元（不包括基架上的基本输出单元或 C200H 组 2 的多点输出单元）输出端的状态执行刷新的同时，还把执行条件的状态写到 I/O 内存指定输出位。

举例



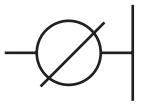
指令	操作数
LD	000000
OUT	000001
OUT NOT	000002

3-4-2 输出非：OUT NOT

用途

把逻辑处理的结果 （执行条件）取反，并输出到指定的位。

梯形图符号



变化

变化	ON 条件时每次循环执行	OUT NOT
	上升沿微分产生 ON 一次	不支持
	下降沿微分产生 ON 一次	不支持
立即刷新功能 （见注）		!OUT NOT

注 CS1D CPU 单元不支持立即刷新功能。

适用程序区

块程序区	步程序区	子程序	中断任务
不允许	OK	OK	OK

操作数规定

区域	输出位操作数
CIO 区	CIO 000000 ～ CIO 614315
工作区	W00000 ～ W51115
保持位区	H00000 ～ H51115
辅助位区	A44800 ～ A95915
定时器区	---
计数器区	---
TR 区	TR0 ～ TR15
DM 区	---
无区号 EM 区	---
有区号 EM 区	---
二进制间接 DM/EM 地址	---
BCD 间接 DM/EM 地址	---
常数	---
数据寄存器	---
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ ,IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

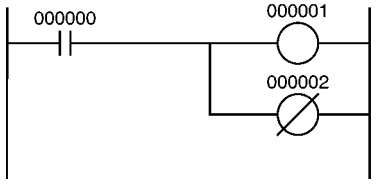
描述

如果没有立即刷新功能，那么把执行条件的状态取反，并写到 I/O 内存的指定位。如果有立即刷新功能，那么把执行条件的状态取反，并写到基本输出单元的输出端及 I/O 内存的输出位。

标志

此指令不影响任何标志。

举例



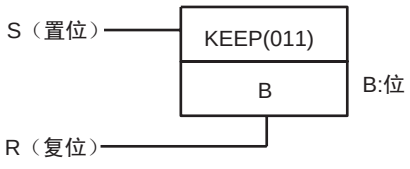
指令	操作数
LD	000000
OUT	000001
OUT NOT	000002

3-4-3 保持：KEEP(011)

用途

操作就象一个锁存继电器。

梯形图符号



变化

变化	ON 条件时每次循环执行	KEEP(011)
	上升沿微分产生 ON 一次	不支持
	下降沿微分产生 ON 一次	不支持
立即刷新功能 （见注）		!KEEP(011)

注 CS1D CPU 单元不支持立即刷新功能。

适用程序区

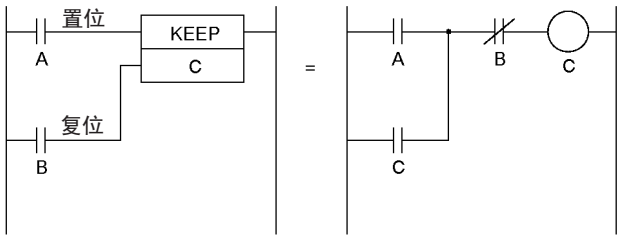
块程序区	步程序区	子程序	中断任务
不允许	OK	OK	OK

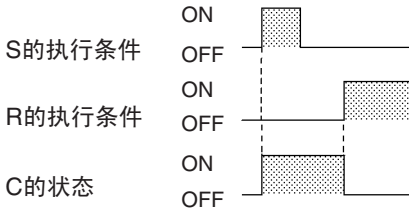
操作数规定

区域	B
CIO 区	CIO 000000 ～ CIO 614315
工作区	W00000 ～ W51115
保持位区	H00000 ～ H51115
辅助位区	A44800 ～ A95915
定时器区	---
计数器区	---
DM 区	---
无区号 EM 区	---
有区号 EM 区	---
二进制间接 DM/EM 地址	---
BCD 间接 DM/EM 地址	---
常数	---
数据寄存器	---
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –) IR0 ～ , –(– –) IR15

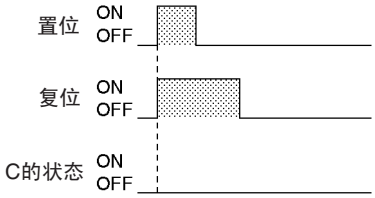
描述

当 S 为 ON，指定位变 ON，此后无论 S 端保持 ON 状态还是变为 OFF 状态，指定位一直保持 ON 状态到复位信号有效。当 R 为 ON，指定位变为 OFF 状态。执行条件 KEEP(011) 位状态之间的关系见下图。

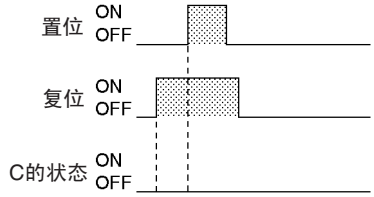




如果 S 和 R 同时为 ON，那么复位输入优先。

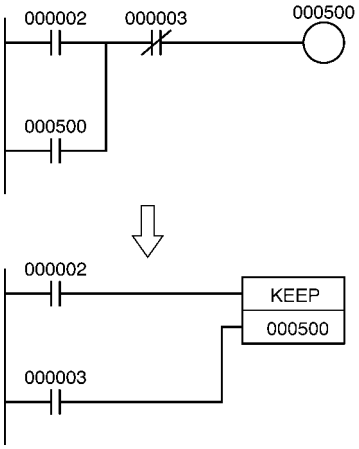


当 R 为 ON 时，不接受置位输入 (S)。

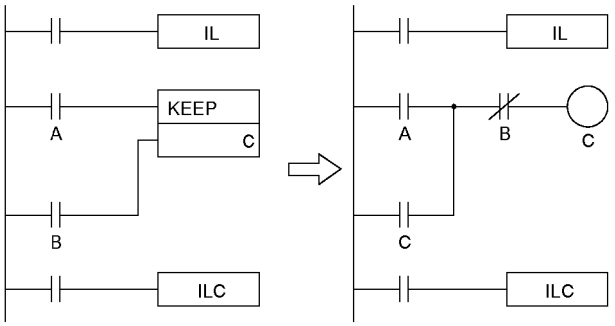


KEEP(011) 有一条立即刷新变化 (!KEEP(011)) 指令。当一个外部输出位已指定给 !KEEP(011) 指令中的 B 时，那么在 !KEEP(011) 指令执行时，B 的任何变化将被刷新并且立即反映到输出位。（如果是组 2 高密度 I/O 单元、高密度特殊 I/O 单元或安装在 SYSMAC BUS 远程 I/O 机架上单元中的位，则改变不会立即反映）。

KEEP(011) 操作就象自保持位，但是用 KEEP(011) 编程的自保持位少了一条指令。

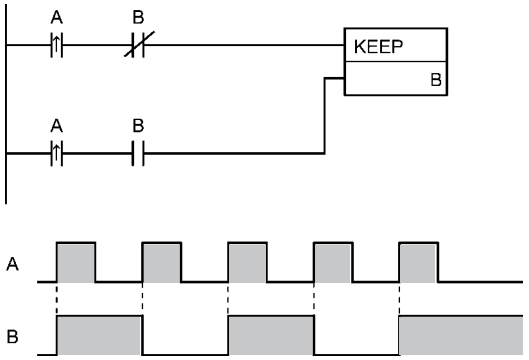


即使在联锁程序中，用 KEEP(011) 编程的自保持位将保持原状态，与不用 KEEP(011) 保持位的编程不同。

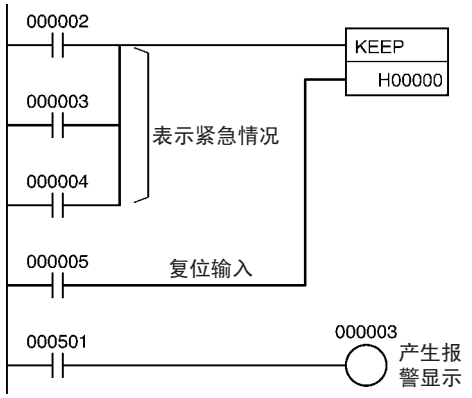


输出位C将保留它联锁时原先的状态。 输出位C在联锁时将变为OFF。

KEEP(011) 可用来产生一如下所示触发器。



如果 B 使用了保持区的位，那么即使在电源中断期间，位状态仍保持不变。因此， KEEP(011) 指令可用于位信号在电源中断后再启动 PLC 仍保持不变的编程。如下图是一个用于紧急状态下关断系统后产生报警信息的例子。



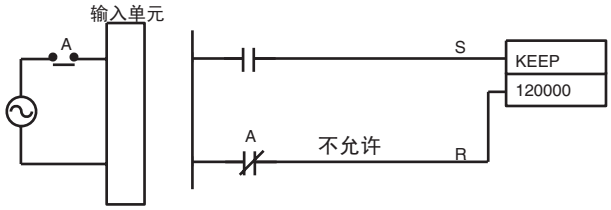
通过使 IOM 保持位变 ON，并且设置 PLC 设置中 IOM 保持位为保持， I/O 区位的状态在断电能够保持。在这种情况下。重新启动 PLC 后，KEEP(011) 中的 I/O 区位将维持状态，就象保持位一样。改变 PLC 设置后，必须重新启动 PLC，否则新的设定将不被采用。

标志

KEEP(011) 不影响任何标志。

注意

当输入装置用 AC 供电时，对于 KEEP(011) 的复位 (R)，不可用一个常闭条件作为输入位。在断开可编程程序控制器 DC 电源（相对于 AC 供电的输入装置）的延时会引起 KEEP(011) 的操作数位被复位，该情况为如下所示。

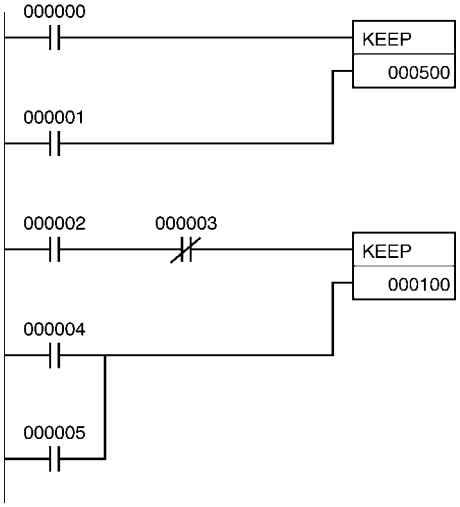


KEEP(011) 操作数的输入顺序在梯形图和助记符中不相同。
梯形图顺序：置位输入→ KEEP(011) →复位输入
助记符顺序：置位输入→复位输入→ KEEP(011)

例

在下面例子中当 CIO 000000 变 ON 时，CIO 00500 变 ON。在 CIO 000001 变 ON 之前， CIO 00500 保持 ON。

在下面例子中当 CIO 000002 变 ON 并且 CIO 000003 为 OFF 时， CIO 00100 变 ON。 CIO 00100 保持 ON 直至 CIO 000004 或 CIO 000005 变 ON。



编程

地址	指令	操作数
000100	LD	000000
000101	LD	000001
000102	KEEP (011)	000500
000103	LD	000002
000104	AND NOT	000003
000105	LD	000004
000106	OR	000005
000107	KEEP (011)	000100

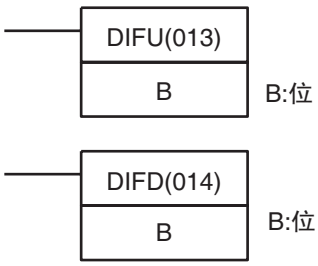
注 梯形图和助记符形式 KEEP(011) 输入顺序不同。在梯形图形式中，输入顺序为：置位输入→ KEEP(011) →复位输入。助记符顺序为：置位输入→复位输入→ KEEP(011)。

3-4-4 上升沿 / 下降沿微分：DIFU(013) 和 DIFD(014)

用途

当执行条件从OFF变为ON时（上升沿），DIFU(013)使指定位变ON一个循环。
当执行条件从ON变为OFF时（下降沿），DIFD(014)使指定位变ON一个循环。

梯形图符号



变化

变化	ON 条件时每次循环执行	不支持
	上升沿微分时执行一次	DIFU(013)
	下降沿微分时执行一次	不支持
立即刷新功能 （见注）		!DIFU(013)

注 CS1D CPU 单元不支持立即刷新功能。

变化	ON 条件时每次循环执行	不支持
	上升沿微分时执行一次	DIFD(014)
	下降沿微分时执行一次	不支持
立即刷新功能 （见注）		!DIFD(014)

注 CS1D CPU 单元不支持立即刷新功能。

适用程序区

块程序区	步程序区	子程序	中断任务
不允许	OK	OK	OK

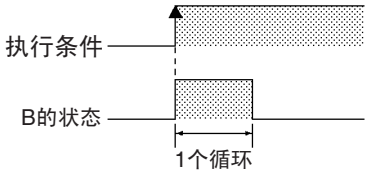
操作数规定

区域	B
CIO 区	CIO 000000 ～ CIO 614315
工作区	W00000 ～ W51115
保持位区	H00000 ～ H51115
辅助位区	A44800 ～ A95915
定时器区	---
计数器区	---
DM 区	---
无区号 EM 区	---
有区号 EM 区	---
二进制间接 DM/EM 地址	---
BCD 间接 DM/EM 地址	---
常数	---
数据寄存器	---

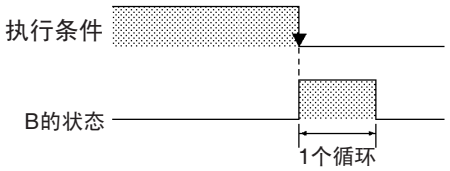
区域	B
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –) IR0 ～ ,15–(– –) IR

描述

当执行条件从 OFF 变为 ON 时， DIFU(013) 使 B 变 ON。
当 DIFU(013) 到下一个循环时， B 变为 OFF。



当执行条件从 ON 为 OFF， DIFD(013) 使 B 变 ON。
当 DIFD(014) 到下一个循环时， B 变为 OFF。



DIFU(013) 和 DIFD(014) 有立即刷新变化（!DIFU(013) 和 !DIFD(014)）。当以其中一种指令形式指定了外部输出位为 B 时，在指令执行时 B 的任何变化将被刷新并且立即反映到输出位。（如果是组 2 高密度 I/O 单元、高密度特殊 I/O 单元或安装在 SYSMAC BUS 远程 I/O 机架上单元中的位，改变不会立即反映）。

当执行条件从 OFF → ON 或 ON → OFF 时， UP(521) 和 DOWN(522) 可用于执行一条指令仅一个循环。详细情况请参考 *3-3-13 条件 ON/OFF：UP(521) 和 DOWN(522)*。

标志

DIFU(013) 和 DIFD(014) 不影响任何标志。

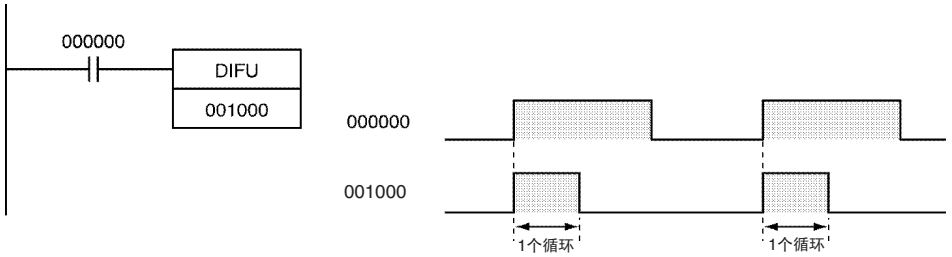
注意

当编在一个联锁程序段，一个跳转程序段或一个子程序中时， DIFU(013) 或 DIFD(014) 的运行不仅取决于本身指令的执行条件，还取决于程序段的执行条件。详细情况参阅 *3-5-3 联锁和联锁清除：IL(002) 和 ILC(003)， 3-5-4 跳转和跳转结束：JMP(004) 和 JMP(005)，以及 3-20 中断控制指令*。

举例

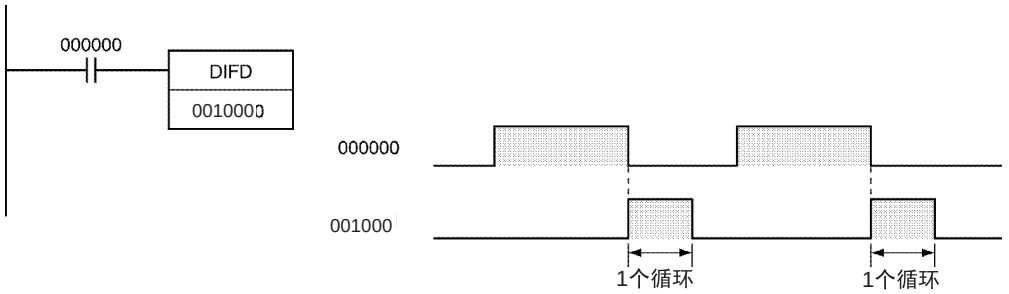
DIFU(013) 的操作

在以下例子中，CIO 000000 从 OFF 变为 ON 时，CIO 001000 变 ON 一个循环。



DIFD(014) 的操作

在以下例子中，CIO 000000 从 ON 变为 OFF 时，CIO 001000 变 ON 一个循环。

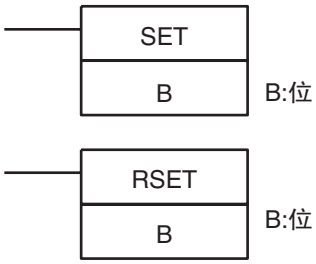


3-4-5 置位和复位：SET 和 RSET

用途

当执行条件为 ON 时， SET 置操作数位为 ON。
当执行条件为 ON 时， RSET 使操作数位为 OFF。

梯形图符号



变化

变化	ON 条件时每次循环执行	SET
	上升沿微分时执行一次	@SET
	下降沿微分时执行一次	%SET
立即刷新功能 （见注）		!SET
组合变化	上升沿微分时立即刷新一次 （见注）	!@SET
	下降沿微分时立即刷新一次 （见注）	!%SET

注 CS1D CPU 单元不支持立即刷新功能。

变化	ON 条件时每次循环执行	RSET
	上升沿微分时执行一次	@RSET
	下降沿微分时执行一次	%RSET
立即刷新功能 （见注）		!RSET
组合变化	上升沿微分时立即刷新一次 （见注）	!@RSET
	下降沿微分时立即刷新一次 （见注）	!%RSET

注 CS1D CPU 单元不支持立即刷新功能。

适用程序区

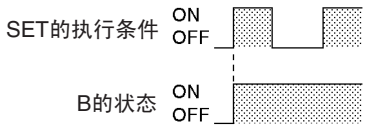
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数规定

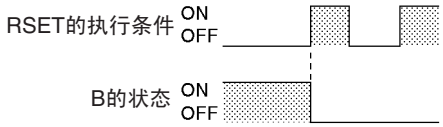
区	B
CIO 区	CIO 000000 ～ CIO 614315
工作区	W00000 ～ W51115
保持位区	H00000 ～ H51115
辅助位区	A44800 ～ A95915
定时器区	---
计数器区	---
DM 区	---
无区号 EM 区	---
有区号 EM 区	---
二进制间接 DM/EM 地址	---
BCD 间接 DM/EM 地址	---
常数	---
数据寄存器	---
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –) IR0 ～ ,–(– –) IR15

描述

当执行条件为 ON 时，SET 置操作数位为 ON，并且当执行条件为 OFF 时，不影响操作数的状态。使用 RSET 可使曾用 SET 变 ON 的位变为 OFF。



当执行条件为 ON 时，RSET 置操作数位为 OFF，并且当执行条件为 OFF 时，不影响操作数的状态。使用 SET 可使曾用 RSET 变 OFF 的位变为 ON。



SET 和 RSET 有立即刷新变化（! SET 和 ! RSET）。当以其中一种指令形式指定了外部输出位为 B 时，在指令执行时 B 的任何变化将被刷新并且立即反映到输出位。（如果是组 2 高密度 I/O 单元、高密度特殊 I/O 单元或安装在 SYSMAC BUS 远程 I/O 机架上单元中的位，状态改变不会立即反映）。

KEEP(011) 指令的置位和复位必须编在一起，但是 SET 和 RSET 指令在编程时完全独立。此外，相同的位可多次用作 SET 或 RSET 指令的操作数。

标志

SET 和 RSET 不影响任何标志。

注意

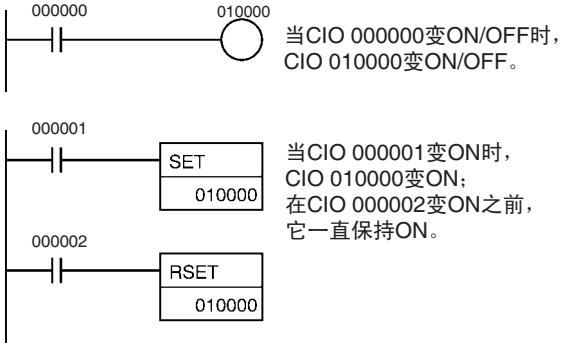
SET 和 RSET 不能用来置位或复位定时器和计数器。

当 SET 或 RSET 编在 IL(002) 和 ILC(003) 或 JMP(004) 和 JMP(005) 之间，如果程序联锁或跳转，那么指定位的状态不变。

举例

OUT/OUT NOT 和 SET 和 RSET 之间的区别

SET 的操作不同于 OUT 的操作，因为当执行条件为 OFF 时，OUT 指令使操作数为变 OFF。同样的，不同于 OUT NOT，因为当执行条件为 OFF 时， OUT 指令使操作数为变 ON。

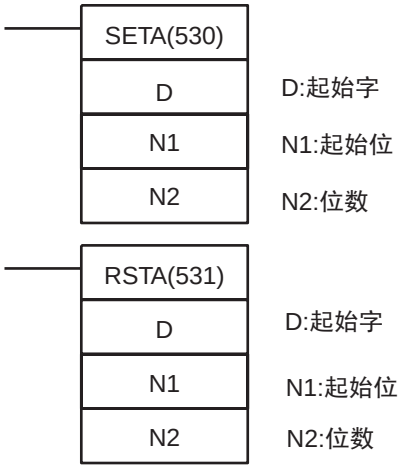


3-4-6 多位置位 / 复位：SETA(530)/RSTA(531)

用途

SETA(530) 使指定的连续位为 ON。
RSTA(531) 使指定的连续位为 OFF。

梯形图符号



变化

变化	ON 条件时每次循环执行	SETA(530)
	上升沿微分时执行一次	@SETA(530)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

变化	ON 条件时每次循环执行	RSTA(531)
	上升沿微分时执行一次	@RSTA(531)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

D: 起始字

指定第一个其中的一些位将变 ON 或 OFF 的字。

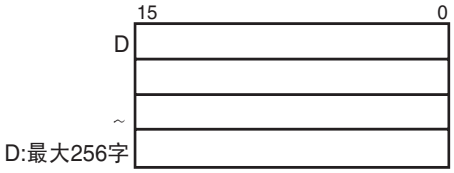
N1: 起始位

指定第一个将变 ON 或 OFF 的位。 N1 必须是 #0000 ～ #000F(&0 ～ &15)。

N2: 位数

指定将变 ON 或 OFF 的位数。 N2 必须是 #0000 ～ #FFFF(&0 ～ &65535)。

注 变 ON 或 OFF 的位必须在同一个数据区。（字的范围是 D ～ D+N2 ÷ 16）。



操作数规定

区	D	N1	N2
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A448 ～ A959	A000 ～ A959	

区	D	N1	N2
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---	#0000 ～ #000F（二进制）或 &0 ～ &15	#0000 ～ #FFFF（二进制）或 &0 ～ &65535
数据寄存器	---	DR0 ～ DR15	
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –) IR0 ～ , –(– –) IR15		

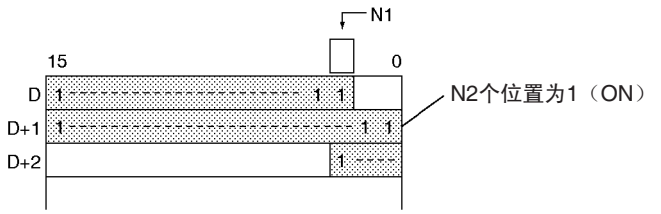
描述

下面对 SETA(530) 和 RSTA(531) 的操作分别进行描述。

SETA(530) 的操作

SETA(530) 使从 D 的位 N1 开始继续到左位 （较高位）的 N2 个位变 ON。所有其它位保持不变。（如果置 N2 为 0，那么没有任何变化）。

由 SETA(530) 变 ON 的位可以用任何其它指令（不一定非要 RSTA(531) 指令）变为 OFF。

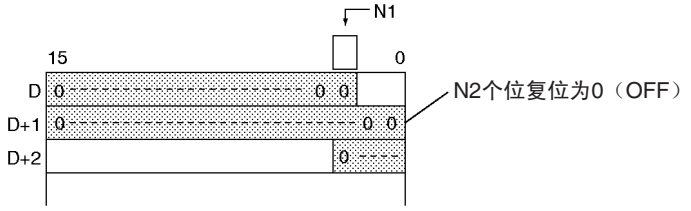


SETA(530) 可用于使一般情况下仅能由字访问的数据区（如 DM 和 EM 区）的位变为 ON。

RSTA(531) 的操作

RSTA(531) 使从 D 的位 N1 开始继续到左位（较高位）的 N2 个位变 OFF。所有其它位保持不变。（如果置 N2 为 0 ，那么没有任何变化）。

由 SETA(531) 变 OFF 的位可以用任何其它指令 （不一定非要 SETA(530) 指令）变为 ON。



RSTA(533) 可用于使一般情况下仅能字访问的数据区（如 DM 和 EM 区）的位变 OFF。

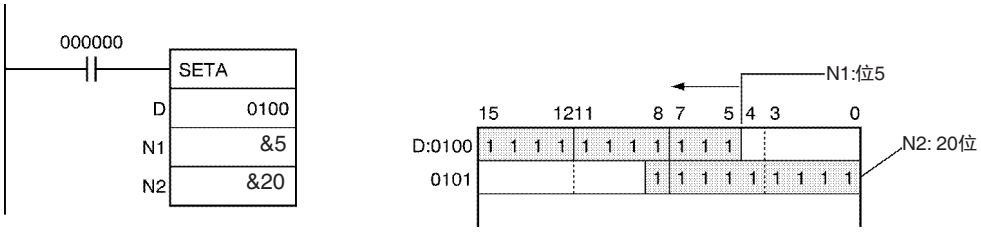
标志

名称	标识	操作
错误标志	ER	如果 N1 不在 0000 ～ 000F 指定范围内时 ON。其它所有情况为 OFF。

例

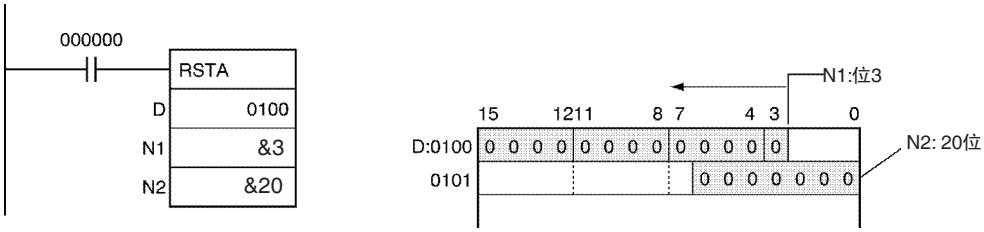
SETA(530) 举例

在下面例子中，CIO 000000 变 ON 时，从 CIO 0100 位 5 开始的 20 位（0014 十六进制数）都变 ON。



RSTA(531) 举例

在下面例子中，CIO 000000 变 ON 时，从 CIO 0100 位 3 开始的 20 位（0014 十六进制数）都变 OFF。



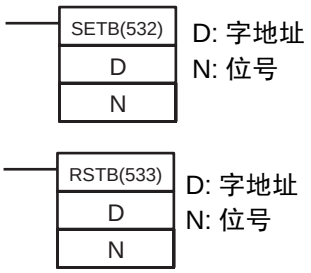
3-4-7 单位置位 / 复位：SETB(532)/RSTB(533)

用途

SETB 置指定位为 ON。
RSTB 置指定位为 OFF。

仅 CS1-H、 CJ1-H、 CJ1M 和 CS1D CPU 单元支持这些指令。

梯形图符号



变化

变化	ON 条件时每次循环执行	SETB(532)
	上升沿微分时执行一次	@SETB(532)
	下降沿微分时执行一次	不支持
立即刷新功能 （见注）		!SETB(532)
组合变化	上升沿微分时执行一次，位立即刷新 （见注）	!@SETB(532)
	下降沿微分时执行一次，位立即刷新	不支持

注 CS1D CPU 单元不支持立即刷新功能。

变化	ON 条件时每次循环执行	RSTB(533)
	上升沿微分时执行一次	@RSTB(533)
	下降沿微分时执行一次	不支持
立即刷新功能 （见注）		!RSTB(533)
组合变化	上升沿微分时执行一次，位立即刷新 （见注）	!@RSTB(533)
	下降沿微分时执行一次，位立即刷新	不支持

注 CS1D CPU 单元不支持立即刷新功能。

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

D: 字地址

指定将变 ON 或 OFF 的位所在字的地址。

N: 起始位

指定将变 ON 或 OFF 的位。 N 必须是 #0000 ～ #000F(&0 ～ &15)。

操作数规定

区	D	N
CIO 区	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W511	
保持位区	H000 ～ H511	
辅助位区	A448 ～ A959	A000 ～ A959
定时器区	T0000 ～ T4095	
计数器区	C0000 ～ C4095	
DM 区	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	

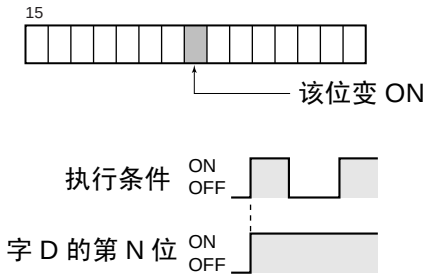
区	D	N
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	#0000 to #000F （二进制）或 &0 to &15
数据寄存器	DR0 ～ DR15	
索引寄存器	---	
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –) IR0 ～ , –(– –) IR15	

描述

下面对 SETB(532) 和 RSTB(533) 的操作分别进行描述。

SETB(532) 的操作

当执行条件为 ON 时，SETB(532) 使字 D 的第 N 位变 ON。当执行条件为 OFF 时，该位状态保持不变。与 SET 不同，SETB(532) 能使 DM 区或 EM 区的位变为 ON。

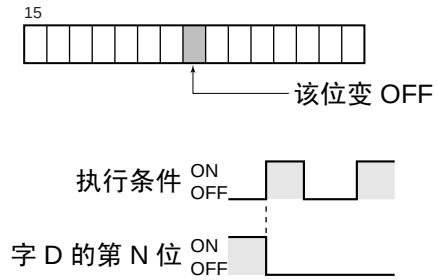


由 SETB(532) 变 ON 的位可以用任何其它指令 （不一定要 RSTB(533) 指令）变为 OFF。

仅 CS1-H、 CJ1-H、 CJ1M 和 CS1D CPU 单元支持 SETB(532) 。

RSTB(533) 的操作

当执行条件为 ON 时，RSTB(533) 使字 D 的第 N 位变 ON。当执行条件为 OFF 时，该位状态保持不变。（SETB(532) 能将该位变位 ON）与 RST 不同，RSTB(533) 能使 DM 区或 EM 区的位变为 OFF。



由 RSTB(533) 变 OFF 的位可以用任何其它指令（不一定要 SETB(532) 指令）变为 ON。

仅 CS1-H、 CJ1-H、 CJ1M 和 CS1D CPU 单元支持 RSTB(533)。

标志

名称	标记	操作
错误标志	ER	如果 N 不在 0000 ～ 000F(&0~&15) 指定范围内时 ON。 其它所有情况为 OFF。

注意

SETB(532) 和 RSTB(533) 不能用来置位或复位定时器和计数器。

当 SETB(532) 或 RSTB(533) 编在 IL(002) 和 ILC(003) 或 JMP(004) 和 JMP(005) 之间，如果程序联锁或跳转，那么指定位的状态不变，即联锁条件或跳转条件为 OFF。

SETB(532) 和 RSTB(533) 有立即刷新变化（!SETB(532) 和 !RSTB(533)）。当一个外部输出位已被这些指令中的一个指定，那么在执行条件执行时，指定位的任何变化将被刷新并立即反映到输出位。（如果指定位是组 2 高密度 I/O 单元、高密度特殊 I/O 单元或安装在 SYSMAC BUS 远程 I/O 机架上单元中的位，则改变不会立即反映）。

SET/RSET 和 SETB(532)/RSTB(533) 的区别

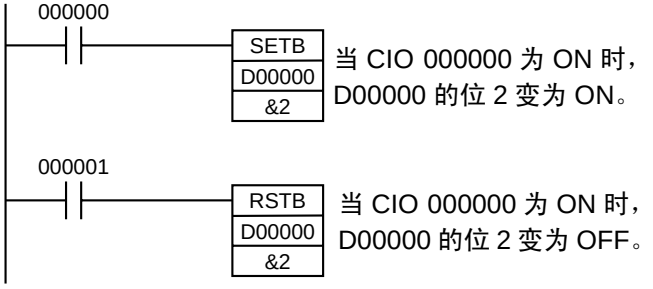
SET 和 RSET 指令和 SETB(532) 和 RSTB(533) 在操作方式上有一定的区别。

1. 当指定位在 CIO、 W、 H 或 A 区时，这些指令操作方式相同。
2. 与SET和RSET不同，SETB(532)和RSTB(533)能控制在DM和EM区的位。

OUTB(534) 和 SETB(532) 和 RSTB(533) 的区别

OUTB(534) 指令和 SETB(532) 和 RSTB(533) 在操作方式上有一定的区别。

1. SETB(532) 和 RSTB(533) 仅当执行条件为 ON 时，才改变指定位的状态。当执行条件为 OFF 时，这些指令对指定位状态没有影响。
2. OUTB(534) 当执行条件为 ON 时变指定位为 ON。当执行条件为 OFF 时，变指定位为 OFF。
3. 对 KEEP(011) 的置位和复位输入必须与这个指令一起被编入程序，但 SETB(532) 和RSTB(533)完全可以独立编程。而且，同一位可作为操作数在任意次SETB(532) 和 RSTB(533) 指令中使用。



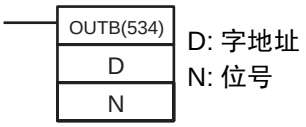
3-4-8 位输出：OUTB(534)

用途

OUTB(534)将指令的执行条件的状态输出给指定位。与OUT不同，OUTB(534)能控制 DM 区或 EM 区的位。

仅 CS1-H、 CJ1-H、 CJ1M 和 CS1D CPU 单元支持这条指令。

梯形图符号



变化

变化	ON 条件时每次循环执行	OUTB(534)
	上升沿微分时执行一次	@OUTB(534)
	下降沿微分时执行一次	不支持
立即刷新功能 （见注）		!OUTB(534)

注 CS1D CPU 单元不支持立即刷新功能。

适用程序区

块程序区	步程序区	子程序	中断任务
不允许	OK	OK	OK

操作数

D: 字地址

指定包含被控位的字。

N: 起始位

指定被控位。 N 必须是在 #0000 ～ #000F(&0 ～ &15) 内。

操作数规定

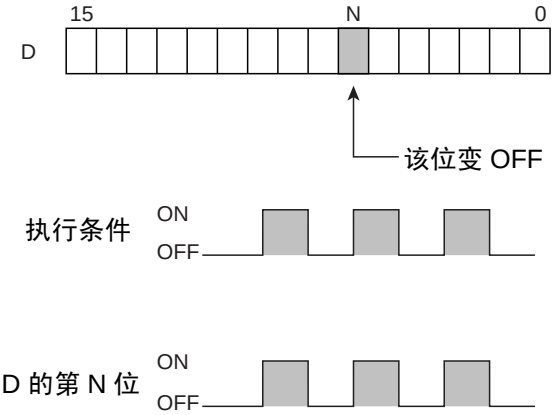
区	D	N
CIO 区	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W511	
保持位区	H000 ～ H511	
辅助位区	A448 ～ A959	A000 ～ A959
定时器区	T0000 ～ T4095	
计数器区	C0000 ～ C4095	
DM 区	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	#0000 ～ #000F （二进制）或 &0 ～ &15
数据寄存器	DR0 ～ DR15	

区	D	N
索引寄存器	---	
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –) IR0 ～ , –(– –) IR15	

描述

当执行条件为 ON 时， OUTB(534) 使字 D 的第 N 位变 ON。

当执行条件为 OFF 时， OUTB(534) 使字 D 的第 N 位变为 OFF。



如果没有使用立即刷新功能，那么把执行条件的状态 （能流）写到 I/O 内存的指定位。如果使用立即刷新功能，那么把执行条件的状态 （能流）写到基本输出单元的输出端及 I/O 内存的输出位。

仅 CS1-H、 CJ1-H、 CJ1M 和 CS1D CPU 单元支持 OUTB(534)。

标志

OUTB(534) 不影响任何标志。

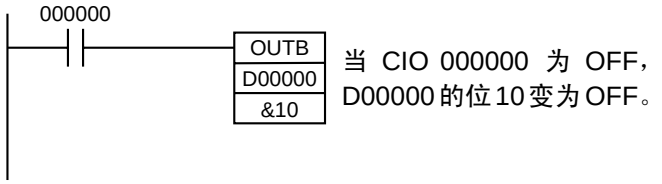
注意

可以指定立即刷新 (!OUTB(534))。仅当对一个分配在基本输出单元 （不包括 C200H 组 2 多点输出单元或在从架上基本输出单元的位）的位执行指令时，一个立即刷新指令刷新输出端的状态，同时，将执行条件的状态（能流）写到 I/O 内存指定输出位。

当 OUTB(534) 编在 IL(002) 和 ILC （003 之间，如果程序联锁，那么指定位的状态变为 OFF。（这与在联锁程序段中 OUT 指令是相同的）。

当一个字被指定为位号 (N)，N 中仅位 00 ～ 03 可用。例如，如果 N 包含 FFFA Hex， OUTB(534) 将控制 D 字的位 10 。

例



3-5 顺序控制指令

3-5-1 结束：END(001)

用途

表示一个程序结束。

梯形图符号

变化

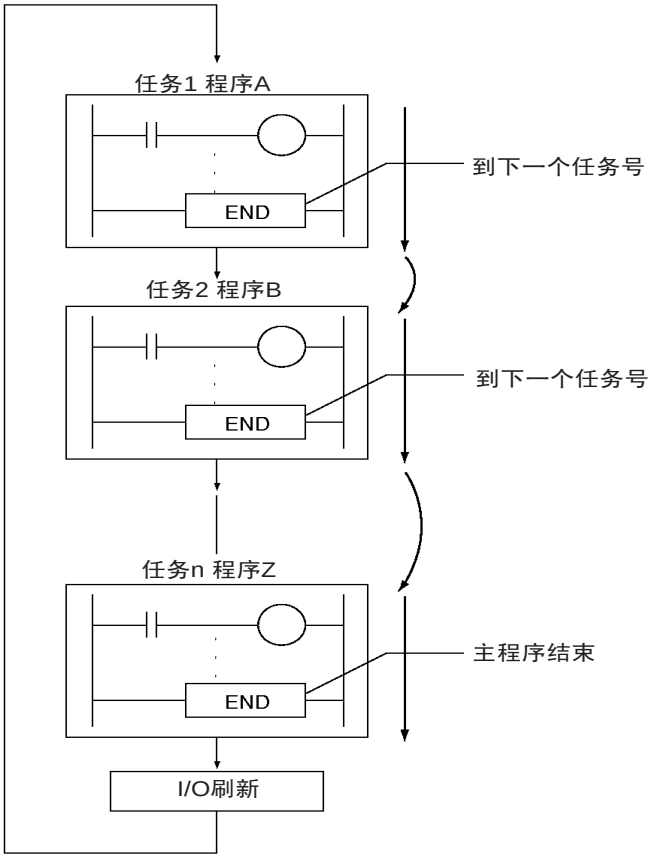
变化	ON 条件时每次循环执行	END(001)
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
不允许	不允许	不允许	OK

描述

END(001) 完成该循环的程序执行。写在 END(001) 后面的任何指令都不执行。继续执行下一个任务号的程序，当在程序中执行了最高任务号的程序，END(001) 表示整个主程序的结束。



注意

在每个程序结束处放置 END(001)，如果程序中没有 END(001) 指令，那么会产生编程错误。

3-5-2 空操作：NOP(000)

用途

这条指令没有功能。（对于 NOP(000) 不执行任何处理）

梯形图符号

NOP(000) 没有梯形图符号。

变化

变化	ON 条件时每次循环执行	NOP(000)
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

描述 虽然 NOP(000) 不执行什么处理，但这条指令可用来在将来要插入指令的地方代替程序行，这样，以后指令插入后，程序地址将不会有变化。

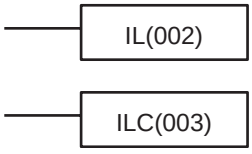
标志 NOP(000) 不影响任何标志。

注意 NOP(000) 仅能用在助记符中，而在梯形图中则不能使用。

3-5-3 互锁和互锁清除：IL(002) 和 ILC(003)

用途 当 IL(002) 的执行条件 OFF 时，互锁 IL(002) 和 ILC(003) 之间的所有输出，IL(002) 和 ILC(003) 通常是成对使用。

梯形图符号



变化

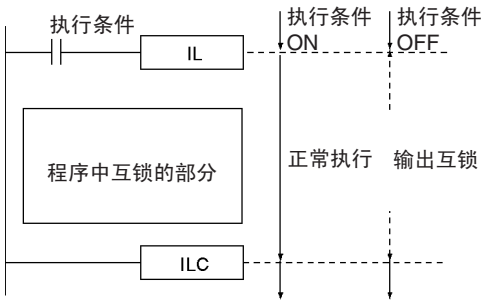
变化	OFF 时互锁 /ON 时不互锁	IL(002)
立即刷新功能		不支持

变化	ON 条件时每次循环执行	ILC(003)
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
不允许	不允许	OK	OK

描述 当 IL(002) 的执行条件 OFF 时， IL(002) 和 ILC(003) 之间所有指令输出互锁，当 IL(002) 的执行条件 ON 时， IL(002) 和 ILC(003) 之间的指令正常执行。



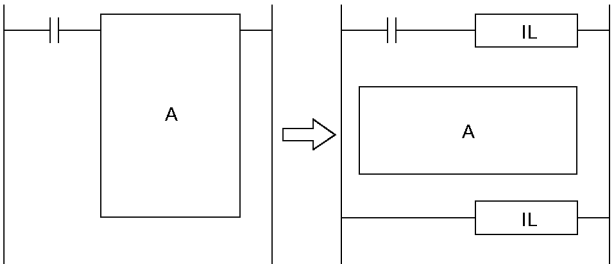
在 IL(002) 和 ILC(003) 间互锁部分的各种处理如下表所示。

指令		处理
在 OUT、 OUT NOT 和 OUTB(534) 中指定的位		OFF
TIM， TIMX(550)， TIMH(015)， TIMHX(551)， TMHH(540)， TMHHX(552)， TIML(542) 和 TIMXL(553)	完成标志	OFF （复位）
	PV	时间设置值 （复位）
在所有其它指令中指定的位或字 *		保持以前状态

注 在所有另外的指令包括 TTIM(087),TTIMX(555),MTIM(543),MTIMX(554),SET，RSET,CNT,CNTX(546),CNTR(012),CNTRX(548),SFT 和 KEEP(011) 的位和字保持以前状态。

如果希望某些位在互锁的程序部分保持 ON，就在 IL （002）前用 SET 把这些位置成 ON。

用 IL(002) 和 ILC(003) 来切换一个程序部分，效率会更高。当控制的几个过程有着相同的执行条件，在 IL(002) 和 ILC(003) 间执行这些过程可以花更少的程序步。



IL(002)/ILC(003) 和 JMP(004)/JME(005) 之间的差别如下表所示。

项目	IL(002)/ILC(003) 中的处理	JMP(004)/JME(005) 中的处理
执行指令	除 OUT、 OUT NOT、 OUTB （534）和定时器外，不执行其它指令。	不执行任何指令
指令中的输出状态	除 OUT、 OUT NOT、 OUTB （534）和定时器外，其它所有输出保持以前状态。	所有输出保持以前状态

项目	IL(002)/ILC(003) 中的处理	JMP(004)/JME(005) 中的处理
在 OUT， OUT NOT， OUTB(534) 中的位	OFF	所有输出保持以前状态
定时器指令中的状态 （除了 TTIM(087)， TTIMX(555)， MTIM(543) 和 MTIMX(554)）	复位	正在执行的定时器 （仅 TIM， TIMX(550)， TIMH(015)， TIMHX(551)， TMHH(540)， TMHHX(552)）继续计时，因为即使定时器指令没有执行，当前值仍然会被刷新。

标志

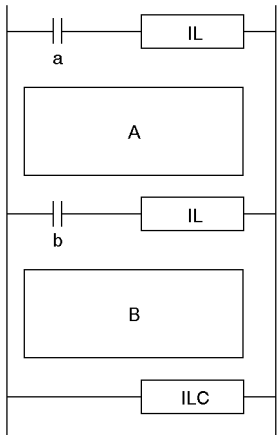
名称	标记	操作
错误标志	ER	OFF
等于标志	=	OFF 或不变 （见注）
负标志	N	OFF 或不变 （见注）

注 在 CS1、 CJ1CPU 单元中，等于和负标志变为 OFF。
在 CS1-H、 CJ1-H、 CJ1M 和 CS1D CPU 单元中，等于和负标志保持不变。

注意 当程序的一部分互锁后，循环时间不会缩短，因为互锁指令在内部执行。

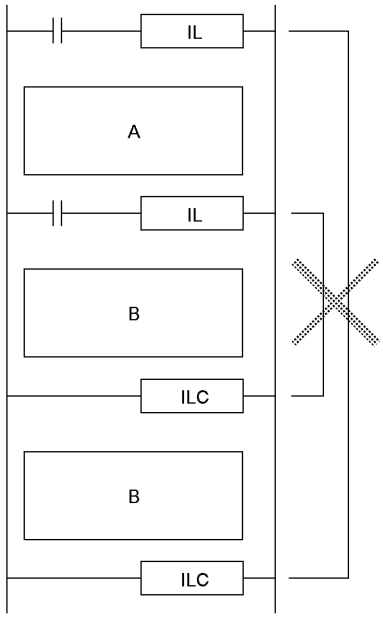
当 DIFU(013)、 DIFD(014) 和微分指令编在 IL(002) 和 ILC(003) 之间时，指令是不能单独依靠执行条件的状态。如果 DIFU(013) 或 DIFD(014) 在一个互锁部分中，并且 IL(002) 的执行条件 OFF，那么 DIFU(013)、 DIFU(014) 和一条微分指令的执行条件的改变不被记录。

一般地， IL(002) 和 ILC(003) 成对使用，尽管后面图例所示可以有多于一个 IL(002) 和单个 ILC(003) 一起使用。如果 IL(002) 和 ILC(003) 不成对使用，当执行程序检查时会产生错误信息，但程序仍能正确执行。



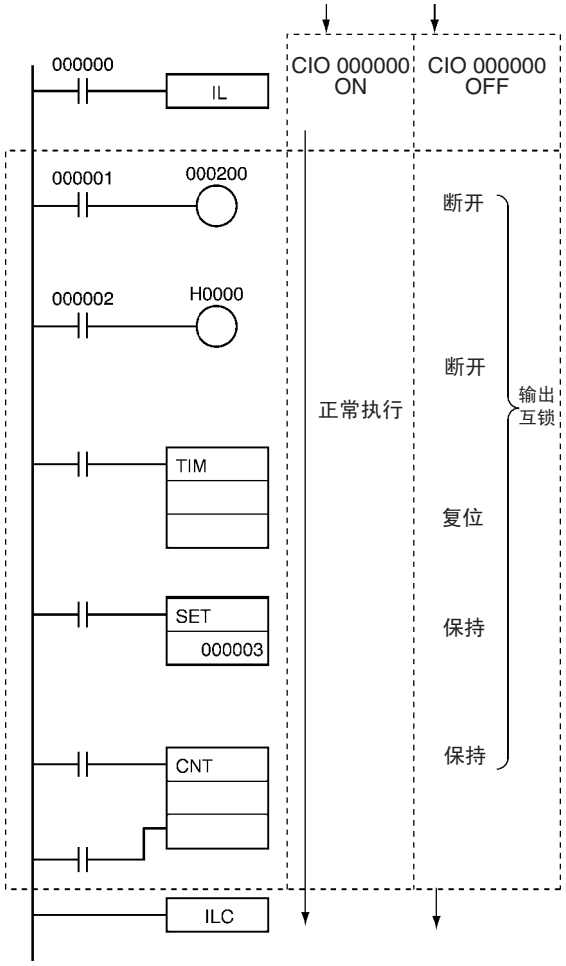
执行条件		程序部分	
a	b	A	B
OFF	ON	互锁	互锁
OFF	OFF	互锁	互锁
ON	OFF	不互锁	互锁
ON	ON	不互锁	不互锁

IL(002) 和 ILC(003) 不能嵌套，如下图所示。



例

在下面的例子中，当 CIO 000000 是 OFF 时，在 IL(002) 和 ILC(003) 之间的输出都被互锁，当下面的例子中的 CIO 000000 是 ON 时，在 IL(002) 和 ILC(003) 之间的指令正常执行。

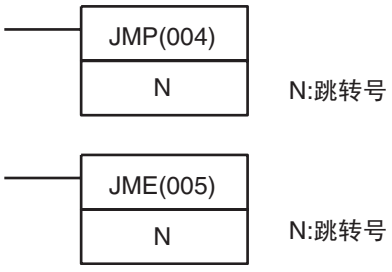


3-5-4 跳转和跳转结束：JMP(004) 和 JME(005)

用途

当 JMP(004) 的执行条件 OFF 时，程序直接跳到程序中有相同跳转号的首个 JME(005) 执行。 JMP(004) 和 JME(005) 成对使用。

梯形图符号



变化

变化	OFF 时跳转 /ON 时不跳转	JMP(004)
立即刷新功能		不支持

变化	ON 条件时每次循环执行	JME(005)
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	不允许	OK	OK

操作数

N: 跳转号

跳转号必须时 0000 ～ 03FF （0 ～ 1023，十进制）

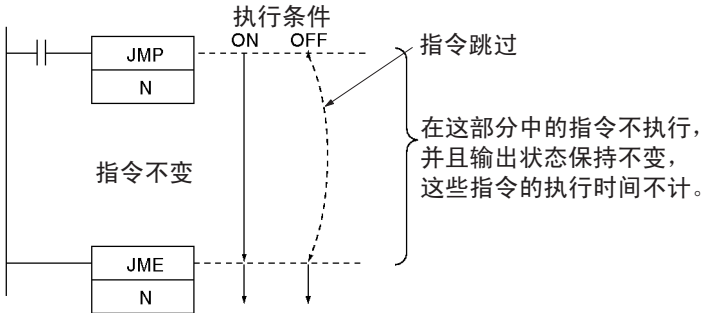
操作数规定

区	N	
	JMP(004)	JME(005)
CIO 区	CIO 0000 ～ CIO 6143	---
工作区	W000 ～ W511	---
保持位区	H000 ～ H511	---
辅助位区	A000 ～ A959	---
定时器区	T0000 ～ T4095	---
计数器区	C0000 ～ C4095	---
DM 区	D00000 ～ D32767	---
无区号 EM 区	E00000 ～ E32767	---
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	---
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	---
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	---
常数	#0000 ～ #03FF（二进制） 或 &0 ～ &1023	#0000 ～ #03FF（二进制） 或 &0 ～ &1023
数据寄存器	DR0 ～ DR15	---
索引寄存器	---	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15	---

描述

当 JMP(004) 的执行条件 ON 时不跳转，程序如写入的那样连续执行。

当 JMP(004) 的执行条件 OFF 时，程序直接跳转到首个在程序中有相同跳转号的 JME(005) 执行， JMP(004) 和 JME(005) 间的指令不执行，这样 ,JMP(004) 和 JME(005) 间的输出状态保持。在块程序中，不管执行条件的状态，JMP(004) 和 JME(005) 间的指令被跳过。



因为当 JMP(004) 的执行条件 OFF 时，所有 JMP(004) 和 JME(005) 间的指令被跳过，循环扫描时间减少了跳过指令的执行时间。相比较，在 JMPO(515) 和 JMEO(516) 间执行 NOP(000) 指令处理，因此，循环时间并不减少得象跳转指令的那么多。

以下表格比较各种跳转指令。

项目	JMP(004) JME(005)	CJP(510) JME(005)	CJPN(511) JME(005)	JMP0(515) JME0(516)
跳转执行条件	OFF	ON	OFF	OFF
允许项目	总共 1,024			没有限制
跳转时的指令处理	不支持			NOP(000) 处理
跳转的指令执行时间	无			就象 NOP(000) 指令
跳转时输出状态 （位和字）	位和字保持以前状态			
跳转时正在运行的定时器状态	运行的定时器继续计时			
在块程序中的过程	始终跳转	ON 时跳转	OFF 时跳转	不允许

标志 (JMP)

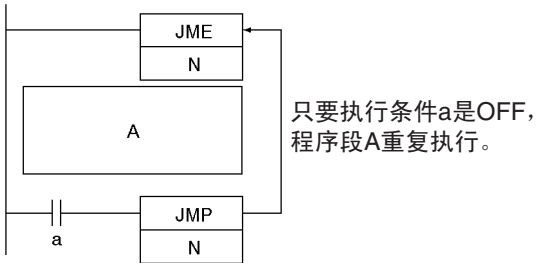
名称	标记	操作
错误标志	ER	如果 N 不在指定的范围 0000 ～ 03FF 内为 ON。 如果程序中有 JMP(004) 而没有相同跳转号的 JME(005) 时 ON。 如果在任务中的 JMP(004) 没有在任务中有相同跳转号的 JME(005) 时 ON。 在其它所有情况下 OFF。

注意

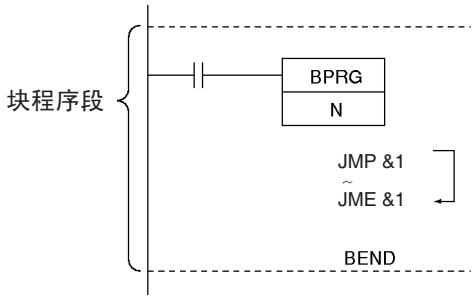
在跳转指令中的全部输出 （位和字）保持以前状态，运行的定时器 （TIM、TIMH(015) 和 TMHH(540)）继续计时，因为即使定时器指令不执行，当前值仍会改变。

当有两个或以上有着相同跳转号的 JME(005) 指令，仅低地址的指令有效，高地址的 JME(005) 指令被忽略。

当在程序中 JME(005) 在 JMP(004) 前面，只要 JMP(004) 的执行条件 OFF，JME(005) 和 JMP(004) 间的指令就会重复执行，如果执行条件不变 ON，或在最大的循环时间里不执行 END(001)，会产生循环时间过长的错误。



在块程序中，不管 JMP(004) 的执行条件的状态如何， JMP(004) 和 JME(005) 之间的指令始终被跳过。



因为任务之间不允许跳转，JMP(004) 和 JME(005) 对必须在同一任务中。如果在同一任务中，相对于 JMP(004) 指令的 JME(005) 指令没有编进程序，那么会产生错误。

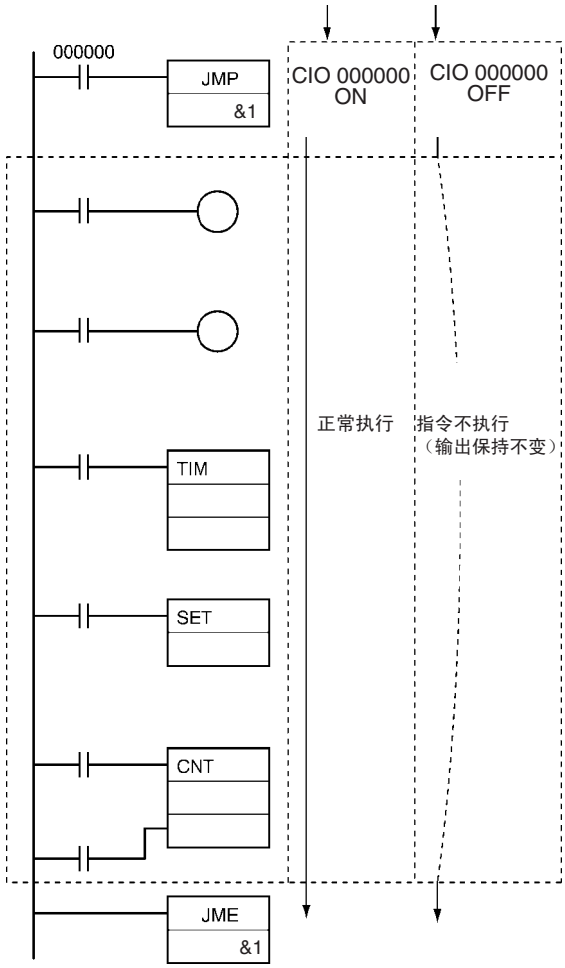
编在 JMP(004) 和 JME(005) 间的 DIFU(013)、DIFD(014) 和微分指令的操作不能单独依靠执行条件的状态，在 JMP(004) 的执行条件变 ON，跳转部分中的 DIFU(013)、DIFD(014) 或微分指令立即执行，DIFU(013)、DIFD(014) 和微分指令的执行条件会和跳转有效前存在的执行条件比较 （也就是， JMP(004) 的执行条件变 OFF 前）。

例

基本操作

在下面例子中当 CIO 000000 为 OFF 时， JMP(004) 和 JME(005) 间的指令不执行，并且输出保持原来的状态。

在下面例子中当 CIO 000000 为 ON， JMP （004）和 JME （005）间的指令正常执行。



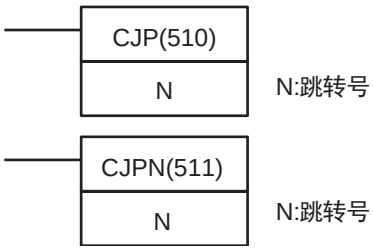
3-5-5 有条件跳转：CJP(510)/CJPN(511)

用途

CJP(510) 的操作基本上和 JMP(004) 相反，当 CJP(510) 的执行条件 ON 时，程序直接跳到程序中有相同跳转号的首个 JME(005) 执行。 CJP(510) 和 JME(005) 成对使用。

CJPN(511) 的处理几乎和 JMP(004) 一样，当 CJPN(511) 的执行条件 OFF 时，程序直接跳到程序中有相同跳转号的首个 JME(005) 执行。 CJPN(511) 和 JME(005) 成对使用。

梯形图符号



变化

变化	ON 时跳转 /OFF 时不跳转	CJP(510)
立即刷新功能		不支持

变化	OFF 时跳转 /ON 时不跳转	CJPN(511)
立即刷新功能		不支持

变化	ON 条件时每次循环执行	JME(005)
立即刷新功能		不支持

适用编程区

块程序区	步程序区	子程序	中断任务
OK	不允许	OK	OK

操作数

N: 跳转号

跳转号必须是在 0000 ～ 03FF （0 ～ 1023，十进制）内。

操作数规定

区	N		
	CJP(510)	CJPN(511)	JME(005)
CIO 区	CIO 0000 ～ CIO 6143		---
工作区	W000 ～ W511		---
保持位区	H000 ～ H511		---
辅助位区	A000 ～ A959		---
定时器区	T0000 ～ T4095		---
计数器区	C0000 ～ C4095		---
DM 区	D00000 ～ D32767		---
无区号 EM 区	E00000 ～ E32767		---
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		---
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		---
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		---
常数	#0000 ～ #03FF（二进制）或 &0 ～ &1023		#0000 ～ #03FF（二进制）或 &0 ～ &1023
数据寄存器	DR0 to DR15		---
索引寄存器	---		---
使用索引寄存器间接寻址	,IR0 to ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15		---

描述

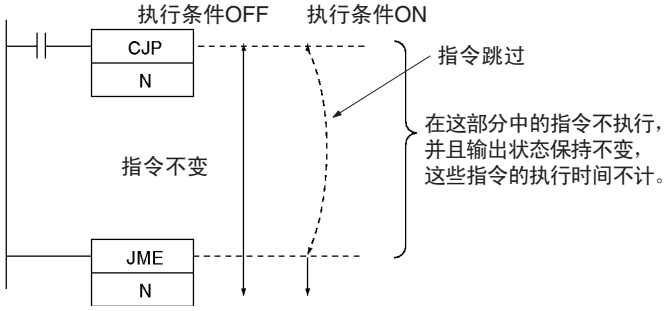
CJP(510) 和 CJPN(511) 的操作区别仅在于执行条件。 CJP(510) 在执行条件 ON 时，程序直接跳转到首个 JME(005) 执行，而 CJPN(511) 在执行条件 OFF 时，程序直接跳转到首个 JME(005) 执行。

由于跳转的指令不执行，循环时间会减少全部跳转的指令执行时间。

CJP(510) 的处理

当 CJP(510) 的执行条件 OFF 时，不跳转并且程序如写入的那样连续执行。

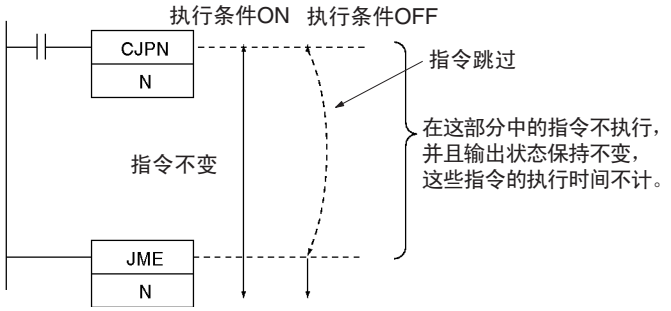
当 CJP(510) 的执行条件 ON 时，程序直接跳转到有着相同跳转号的 JME(005) 执行。



CJPN(511) 的处理

当 CJPN(511) 的执行条件 ON 时，不跳转并且程序如写入的那样连续执行。

当CJPN(511)的执行条件OFF时，程序直接跳转到有着相同跳转号的JME(005)执行。



标志

下表显示了 CJP(510) 和 CJPN(511) 对标志的影响。

名称	标记	操作
错误标志	ER	如果 N 不在指定的范围 0000 ～ 03FF 内为 ON。 如果程序中有 CJP(510) 而没有相同跳转号的 JME(005) 时 ON。 如果在任务中的 CJP(510) 没有在任务中有相同跳转号的 JME(005) 时 ON。 在其它所有情况下 OFF。

描述

在跳过指令中所有输出（位和字）保持原来状态。运行中的定时器（TIM, TIMX(550) ,TIMH(015),TIMHX(551),TMHH(540) 和 TMHHX(552)）继续计时。因为即使定时器指令不执行， PV 值仍会改变。

当有两个或以上有相同的跳转号的 JME(005) 指令，仅低地址的指令有效，高地址的 JME(005) 被忽略。

当在程序中 JME(005) 超前 CJP(510) 或 CJPN(511) 指令，只要执行条件保持 OFF (CJP(510)) 或 ON (CJPN(511)) 之间的指令就会重复执行，如果不改变执行条件。在最大循环时间内不执行 END(001)，跳转不结束就会产生循环扫描时间太长错误。

在块程序中 CJP(510) 或 CJPN(511) 指令会正常处理。

当 CJP(510) 的执行条件 ON 或 CJPN(511) 的执行条件 OFF 时，程序会直接跳到 JME 指令执行，而不执行 CJP(510)/CJPN(511) 和 JME 之间的指令，这些指令不需要执行时间，这样循环时间会减少。

当 JMPO 的执行条件为 OFF 时，JMPO 和 JMEO 间执行 NOP 操作，需要执行时间，这样，循环时间不减少。

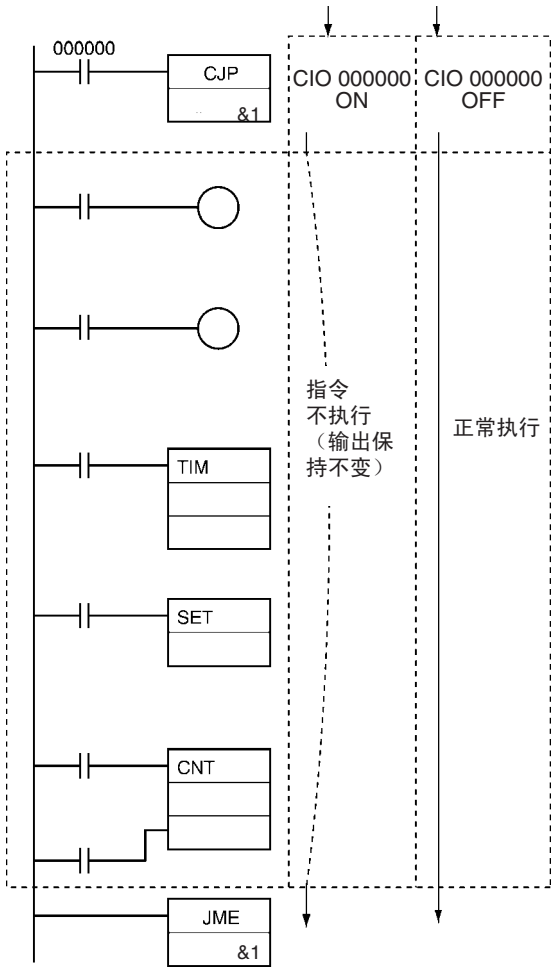
当在一个任务中编程有 CJP(510) 或 CJPN(511) 指令时，必须有一个相同跳转号的 JME(005)，因为任务间是不允许跳转的。如果在同一任务中相应的 JME(005) 没有编入将会产生一个错误。

编在跳转程序部分中的 DIFU(013)、 DIFD(014) 和微分指令的处理不能单独根据执行条件的状态，在 CJP(510) 的执行条件变 OFF（CJPN(511) 为 ON）后，跳转程序中的 DIFU(013)、 DIFD(014) 或微分指令立即执行时， DIFU(013)、DIFD(014) 或微分指令的执行条件会和跳转有效前存在的执行条件相比较。

例

在下面例子中，当 CIO 000000 变 ON 时， CJP(510) 和 JME(005) 间的指令不执行，并且输出保持原来状态。

在下面例子中，当 CIO 000000 变 OFF 时，CJP(510) 和 JME(005) 间的指令正常执行。



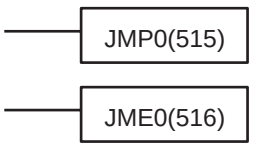
注　对于 CJPN(511)， CIO 000000 的 ON/OFF 状态要相反。

3-5-6　多重跳转和跳转结束：JMP0(515) 和 JME0(516)

用途

当JMP0(515)的执行条件OFF时，在程序中从JMP0(515)到下一个JME0(516)间的所有指令作为 NOP(000) 处理，JMP0(515) 和 JME0(516) 成对使用，在程序中可用的次数没有限制。

梯形图符号



变化

变化	OFF 时跳转 /ON 时不跳转	JMP0(515)
立即刷新功能		不支持

变化	ON 条件时每次循环执行	JME0(516)
立即刷新功能		不支持

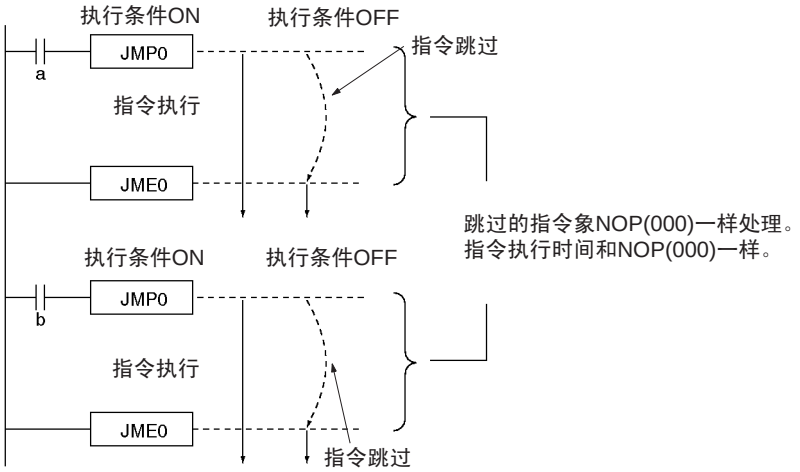
适用程序区

块程序区	步程序区	子程序	中断任务
不允许	不允许	OK	OK

描述

JMP0(515) 在执行条件 ON 时，不跳转并且程序如写入的那样连续执行。

JMP0(515) 在执行条件 OFF 时，程序中将所有从 JMP0(515) 和 JME0(516) 间的指令作 NOP(000) 处理。不象 JMP(004)、 CJP(510) 和 CJPN(511) 指令，JMP0(515) 不需要跳转号，所以这些指令可放在程序的任何位置。



不象JMP(004)，CJP(510)和CJPN(511)，在程序中直接跳转到首个JME(005)。从 JMP0(515) 和 JME0(516) 间的指令作 NOP(000) 执行，跳转的指令执行时间会减少，但不消除。跳过的指令本身不执行，并且输出 （位和字）保持原来的状态。

注意

在程序中可使用多对 JMP0(515) 和 JME0(516) 指令，但对与对之间不能嵌套。

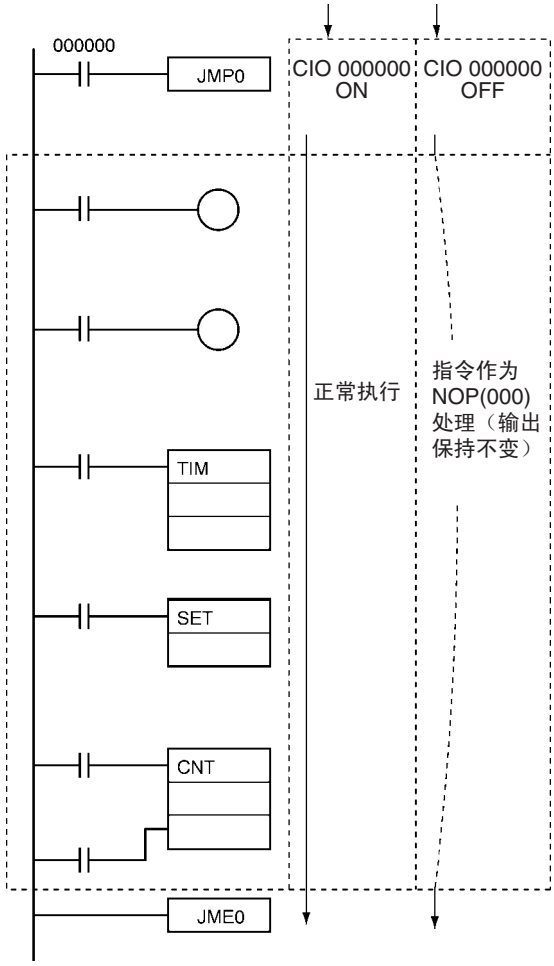
JMP0(515) 和 JME0(516) 不能用在块程序中。

JMP0(515) 和 JME0(516) 对必须用在同一任务中，因为任务之间的跳转是不允许的。

对于 DIFU(013)，DIFD(014) 和微分指令，编在 JMP0(515) 和 JME0(516) 中的的处理不能单独根据执行条件的状态，在 JMP0(515) 的执行条件变 ON 后，跳转程序中的 DIFU(013)， DIFD(014) 或微分指令立即执行时， DIFU(013)，DIFD(014)或微分指令的执行条件会和跳转有效前存在的执行条件相比较。（也就是在 JMP0(515) 的执行条件变 OFF 前）。

例

在下面例子中，当 CIO 000000 变 OFF 时，JMP0(515) 和 JME(005) 间的指令作为 NOP(000) 指令处理，并且输出保持原来状态。
在下面例子中，当 CIO 000000 变 ON 时， JMP0(515) 和 JME(005) 间的指令正常执行。

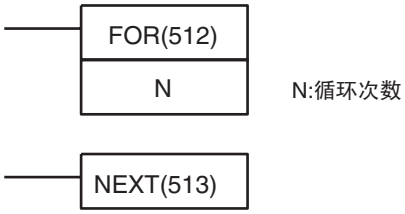


3-5-7 FOR-NEXT 循环：FOR(512)/NEXT(513)

用途

FOR(512) 和 NEXT(513) 间的指令重复指定的次数，FOR(512) 和 NEXT(513) 成对使用。

梯形图符号



变化

变化	ON 条件时每次循环执行	FOR(512)
	ON 条件时每次循环执行	NEXT(513)
立即刷新功能		不支持

适用编程区

块程序区	步程序区	子程序	中断任务
不允许	OK	OK	OK

操作数

N: 循环次数

循环次数必须是 0000 ～ FFFF （0 ～ 65535，十进制）

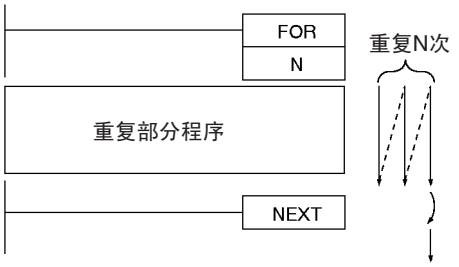
操作数规定

区	N
CIO 区	CIO 0000 ～ CIO 6143
工作区	W000 ～ W511
保持位区	H000 ～ H511
辅助位区	A000 ～ A959
定时器区	T0000 ～ T4095
计数器区	C0000 ～ C4095
DM 区	D00000 ～ D32767
无区号 EM 区	E00000 ～ E32767
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	#0000 ～ #FFFF （二进制）或 &0 ～ &65,535
数据寄存器	DR0 ～ DR15
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –) IR0 ～ , –(– –) IR15

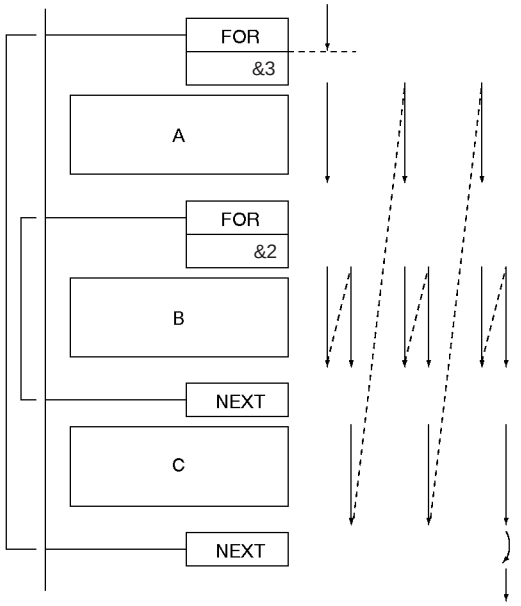
描述

FOR(512) 和 NEXT(513) 中的指令执行 N 次，然后，NEXT(513) 后程序继续执行指令， BREAK(514) 指令可用来退出循环。

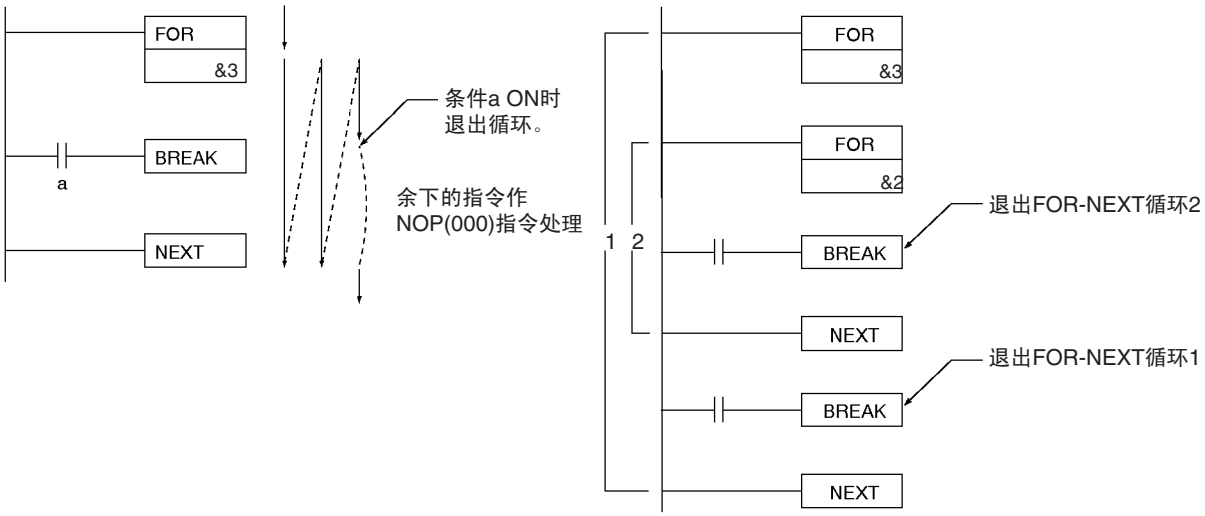
如果 N 设置为 0，FOR(512) 和 NEXT(513) 间的指令作为 NOP(000) 指令处理。循环可用最小的编程量来处理数据表格。



FOR-NEXT 循环可嵌套多至 15 级，在下面例子中，程序段 A，B 和 C 如下执行: A → B → B → C, A → B → B → C 和 A → B → B → C



使用 BREAK(514) 从一个 FOR-NEXT 循环中退出。从嵌套循环总退出，需要多个（嵌套级数） BREAK(514) 指令。 BREAK(514) 后循环中余下的指令作 NOP(000) 指令处理。



循环可变方法

有两种方法重复一个程序段直到需要的执行条件输入。

1,2,3... 1. FOR-NEXT 循环和 BREAK

用最大重复数 N 开始 FOR-NEXT 循环，在循环中用需要的执行条件编入 BREAK(514)，如果输入执行条件，循环会在 N 次重复前结束。

2. JME(005) -JMP(004) 循环

在 JMP(004) 前用 JME(005) 编一个循环，只要 JMP(004) 的执行条件 OFF，JME(005) 和 JMP(004) 间的指令会重复执行，（如果执行条件不变 ON 或在最大循环时间内不执行 END(001)，会产生循环时间太长错误）。

标志

名称	标记	操作
错误标志	ER	超过 15 个嵌套循环时 ON。 在其它所有情况下 OFF。
等于标志	=	OFF
负标志	N	OFF

注意

把 FOR(512) 和 NEXT(513) 编在同一个任务中，如果这些指令不在同一任务中，不执行重复。

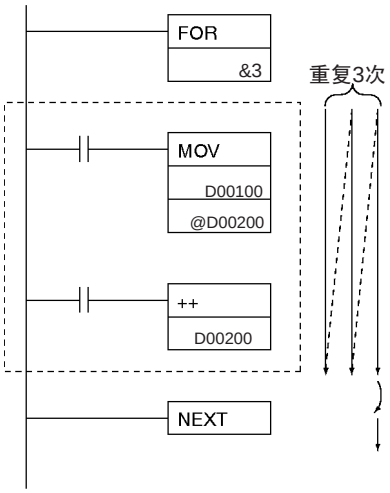
象 JMP(004) 的跳转指令可以在 FOR-NEXT 循环中执行，但不能跳出 FOR-NEXT 循环。

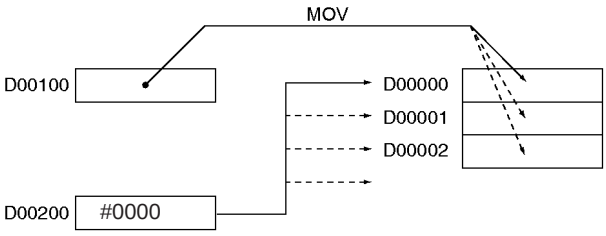
下列指令不能用在 FOR-NEXT 循环中：

- 块编程指令
- 多重跳转和跳转结束：JMP(515) 和 JME(516)
- 步定义和步开始：STEP(008)/SNXT(009)

例

在下面例子中，循环程序段 D00100 的内容传给 D00200 中所示地址，然后 D00200 中的内容 +1。





3-5-8 退出循环：BREAK(514)

用途

编在 FOR-NEXT 循环中，所给的执行条件满足时，退出循环操作。循环中余下的指令作 NOP(000) 指令处理。

梯形图符号

变化

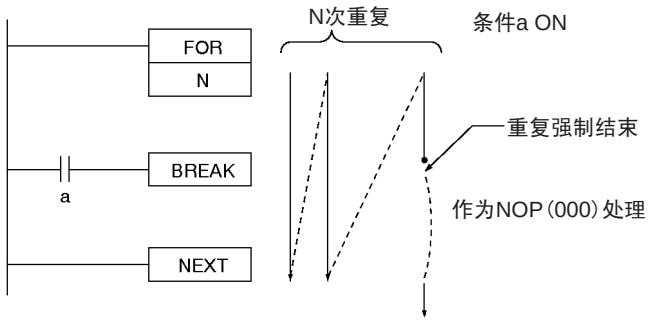
变化	ON 条件时每次循环执行	BREAK(514)
	上升沿微分时执行一次	不支持
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
不允许	OK	OK	OK

描述

当 BREAK(514) 执行后，编程在 FOR(512) 和 NEXT(513) 中的 BREAK 退出 FOR-NEXT 循环，当 BREAK(514) 指令执行后， NEXT(513) 前的剩余指令作为 NOP(000) 指令处理。



标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	OFF
负标志	N	OFF

注意

一条 BREAK(514) 指令仅退出一个循环，因此从嵌套循环中退出需要多个（嵌套级数） BREAK(514) 指令。

BREAK(514) 仅能用在 FOR-NEXT 循环中。

3-6 定时器和计数器指令

这一节用来描述定义和使用定时器和计数器的指令。

指令	助记符	函数代码	页数
定时器	TIM/TIMX	---/551	207
高速定时器	TIMH/TIMHX	015/551	211
1 毫秒定时器	TMHH/TIMHHX	540/552	216
累积定时器	TTIM/TTIMX	087/555	219
长定时器	TIML/TIMLX	542/553	222
多输出定时器	MTIM/MTIMX	543/554	226
计数器	CNT/CNTX	---/546	224
可逆计数器	CNTR/CNTRX	012/548	234
复位定时器 / 计数器	CNR/CNRX	545/547	238

定时器和计数器 PV 的刷新方法

■ 综述

由 CS1 和 CJ1 CPU 单元支持的定时器和计数器指令都用 BCD 码数据，并且所有的设置值都用 BCD 码数输入。对于在 CS 和 CJ 系列中的其它 CPU 单元（即，CS1-H、CJ1-H、CJ1M 和 CS1D CPU 单元，见注 1 和 2），刷新方法可用 BCD 码或二进制设置。

使用二进制数据代替 BCD 码，允许的最大设置值 (SV) 范围从 0 ～ 9999 增加为 0 ～ 65535。这使用其它指令计算的二进制数可直接作为定时器 / 计数器的 SV。甚至当间接设置 SV （即，使用存贮字的内容）。时，刷新也是有效的。（即，根据设定的刷新方法将地址字中的内容作为 BCD 码或二进制数）。

刷新方法的详细情况参考 *CS/CJ 系列编程手册*。

注 1. 对于 2002 年 5 月 31 日以前生产的 CS1-H 和 CJ1-H CPU 单元，二进制指令被显示在编程器以存储 BCD 码操作的等值指令。（例如， TIMX0&16 将显示为 TIM0&16）。然而，指令将使用二进制模式操作。

2. 仅 3.0 及以后的 CX-Programmer 可选择刷新方法。2.1 及以前的版本不能，在手持编程器上也不能选择刷新方法。

3. 用二进制刷新模式的用户程序不能被 CX-Programmer 及以前的版本读出，必须改变为 BCD 模式才能读出。

■ 适用指令

分类	指令	存储器	
		BCD	二进制
定时器 / 计数器指令	定时器	TIM	TIMX(550)
	高速定时器	TIMH(015)	TIMHX(551)
	1 毫秒定时器	TMHH(540)	TMHHX(552)
	累积定时器	TTIM(087)	TTIMX(555)
	长定时器	TIML(542)	TIMLX(553)
	多输出定时器	MTIM(543)	MTIMX(554)
	计数器	CNT	CNTX(546)
	可逆计数器	CNTR(012)	CNTRX(548)
	复位定时器 / 计数器	CNR(545)	CNRX(547)
块编程指令	定时器等待	TIMW(813)	TIMWX(816)
	高速定时器等待	TMHW(815)	TMHWX(817)
	计数器等待	CNTW(814)	CNTWX(818)

定时器的基本规定

下表显示了定时器的基本规定。

项目	TIM/TIMX(550)	TIMH(015)/ TIMHX(551)	TMHH(540)/ TMHHX(552)	TTIM(087)/ TTIMX(555)	TIML(542)/ TIMLX(553)	MTIM(543)/ MTIMX(554)
定时方式	递减	递减	递减	递增	递减	递增
定时单位	0.1 s	0.01 s	0.001 s	0.1 s	0.1 s	0.1 s
最大 SV	TIM: 999.9 s TIMX: 6,553.5 s	TIMH: 99.99 s TIMHX: 655.35 s	TMHH: 9.999 s TMHHX: 65.535 s	TTIM: 999.9 s TTIMX: 6,553.5 s	TIML: 115 days TIMLX: 49,710 days	MTIM: 999.9 s MTIMX: 6,553.5 s
输出 / 指令	1	1	1	1	1	8
定时器号	使用	使用	使用	使用	不使用	不使用
完成标志刷新	执行时	执行时	每 1 毫秒中断	执行时	执行时	执行时
定时器 PV 刷新	见注 1	见注 2	每 1 毫秒	执行时	执行时	执行时

项目		TIM/TIMX(550)	TIMH(015)/ TIMHX(551)	TMHH(540)/ TMHHX(552)	TTIM(087)/ TTIMX(555)	TIML(542)/ TIMLX(553)	MTIM(543)/ MTIMX(554)
复位后的值	完成标志	OFF	OFF	OFF	OFF	OFF	OFF
	PV	SV	SV	SV	0	SV	0

注 1. TIM PV 的刷新，在每次循环程序执行的结束或如果循环时间超出 80ms 每隔 80ms 中断时执行。

2. TIMH(015)/TIMHX(551)PV的刷新，在每次循环程序执行的结束或每隔10ms中断时执行。

定时器操作

下表显示操作和程序条件对定时器的影响。

项目		TIM/ TIMX(550)	TIMH(015)/ TIMHX(551)	TMHH(540)/ TMHHX(552)	TTIM(087)/ TTIMX(555)	TIML(542)/ TIMLX(553)	MTIM(543)/ MTIMX(554)
操作模式改变		PV = 0 完成标志 = OFF				---	---
电源中断 / 复位		PV = 0 完成标志 = OFF				---	---
CNR(545)/CNRX(547) 的执行		二进制：PV = FFFF，完成标志 = OFF BCD：PV = FFFF 或 9999 ，完成标志 = OFF				不可用	不可用
在跳转程序段中的处理 (JMP(004)-JME(005))		操作中的定时器继续计时			定时器状态保持		
在互锁程序段中的处理 (IL(002)-ILC(003))		PV = SV 完成标志 = OFF			定时器状态保持	PV = SV 完成标志 =OFF	定时器状态保持
强制复位	完成标志	ON				---	---
	PV	置为 0				---	---
强制复位	完成标志	OFF				---	---
	PV	置位到 SV			置为 0	---	---

3-6-1 定时器：TIM/TIMX(550)

用途

TIM 或 TIMX(550) 是单位为 0.1s 的递减定时器，设置值的设置范围 TIM 为 0 ～ 999.9 s； TIMX(550) 为 0 ～ 6， 553.5s。定时器的精度为 0 ～ 0.01s。

注 CS1D CPU 单元的定时器精度是 10 毫秒 + 循环时间。

梯形图符号

PV 刷新方法	符号	操作数
BCD	TIM N　N:定时器号 S　S:设置值	N:0000 ～ 4095 （十进制） S:#0000 ～ #9999 (BCD)
二进制	TIMX(550) N　N:定时器号 S　S:设置值	N:00000 ～ 4095 （十进制） S:&0 ～ &65535（十进制） #0000 ～ #FFFF （十六进制）

变化

变化	ON 条件时每次循环执行	TIM/TIMX(550)
	上升沿微分时执行一次	不支持
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
不允许	OK	OK	不允许

操作数

N: 定时器号

定时器号必须在 0000 ～ 4095 间 （十进制）

S: 设置值

设置值必须在 #0000 ～ 9999(BCD) 之间。
（如果设置值为 #0000，当 TIM/TIMX(550) 执行后，完成标志变为 ON）。

操作数规定

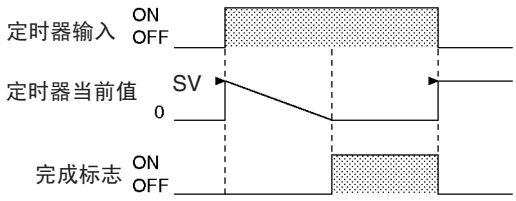
区	N	S
CIO 区	---	CIO 0000 ～ CIO 6143
工作区	---	W000 ～ W511
保持位区	---	H000 ～ H511
辅助位区	---	A000 ～ A959
定时器区	0000 ～ 4095 （十进制）	T0000 ～ T4095
计数器区	---	C0000 ～ C4095
DM 区	---	D00000 ～ D32767
无区号 EM 区	---	E00000 ～ E32767
有区号 EM 区	---	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	---	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 地址	---	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_032767 (n = 0 ～ C)
常数	---	BCD: #0000 ～ 9999 (BCD) “&”不能用 二进制： &0 ～ &65535 （十进制） #0000 ～ #FFFF （十六进制）
数据寄存器	---	DR0 ～ DR15
索引寄存器	---	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15	

描述

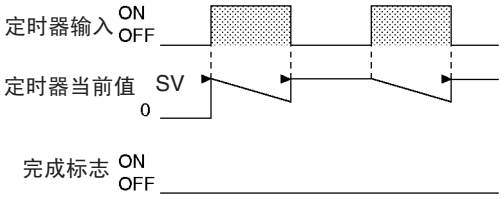
当定时器输入 OFF 时，指定定时器 N 复位，即，定时器的当前值复位到设置值，并且完成标志变为 OFF。

当定时器输入从 OFF 到 ON，TIM/TIMX(550) 开始从当前值递减，只要定时器输入保持 ON，当前值会连续递减，且当前值达到 0000 时，定时器完成标志会变 ON。

在定时器计时到后，定时器的 PV 和完成标志状态会保持，要重新启动定时器，定时器输入必须变 OFF，然后再一次变 ON，或者定时器的 PV 值必须改为一个非零值 （例，使用 MOV(021) 指令）。



下面时间图表示了当定时器计时完成前，定时器输入变 OFF，定时器 PV 值和完成标志的状态。



标志

名称	标记	操作
错误标志	ER	如果 N 通过索引寄存器间接寻址，但是索引寄存器中的地址不是定时器的 PV 的地址时为 ON。 如果是 BCD 模式下，而 S 不包含 BCD 数据时 ON。 其它情况时 OFF。
等于标志	=	OFF 或不变 （见注）
负标志	N	OFF 或不变 （见注）

注 在 CS1 和 CJ1 CPU 单元中，这些标志变为 OFF。
在 CS1-H、 CJ1-H、 CJ1M 和 CS1D CPU 单元中，这些标志保持不变。

注意

TIM,TIMX(550),TIMH(015),TIMHX(551),TMHH(540),TMHHX(552),TTIM(087),TTIMX(555),TIMW(813),TIMWX(816),TMHW(815) 和 TMHWX(817) 指令共享定时器号。当程序检查时，如果两个定时器使用相同的定时器号，但不同时使用，会产生一条重复错误，但定时器会正常操作。如果同时使用，使用相同定时器号的定时器不会正常操作。

当 CPU 单元的循环时间超过 80ms 时，定时器号为 2048 ～ 4095 的定时器将不能正常操作。当循环时间超过80ms时，请用定时器号0000～2047的定时器。

即使定时器在待机时，用定时器号 0000 ～ 2047 编程的定时器的当前值也会改变，而用定时器号 2048 ～ 4095 编程的定时器的当前值在待机时会保持。

在下面情况下，定时器会被复位或暂停 （当定时器复位后，它的 PV 值复位到 SV 值，并且完成标志为 OFF）。

条件	PV	完成标志
操作模式从运行或监视模式转换到编程模式或者反过来操作时 [1]	0000	OFF
电源断开和复位 [2]	0000	OFF
执行 CNR(545)/CNRX(547) 定时器 / 计数器的复位命令 [3]	BCD：9999 二进制：FFFF	OFF
在互锁程序中 (IL(002) -ILC(003)) 处理	复位到 SV	OFF
在跳转程序中 (JMP(004)-JME(005)) 处理	PV 保持递减	保持以前状态

注 1. 如果 IOM 保持位 (A50012) 已经变 ON，当操作模式改变后，定时器的完成标志和 PV 值的状态会保持。

2. 如果 IOM 保持位 (A50012) 已经变 ON，并且 IOM 保持位自己的状态在 PLC 设置中保护，即使电源断开，定时器的完成标志和 PV 值的状态仍会保持。

3. 当 TIM/TIMX(550) 执行后， PV 值被设置为 SV 值。

当 TIM/TIMX(550) 在 IL(002) 和 ILC(003) 之间的程序段中，并且程序段是互锁的， PV 值会被复位为 SV 值，并且完成标志变 OFF。

当定时器号为 0000 ～ 2047 的 TIM/TIMX(550) 的定时器在跳转程序段中 (JMP(004)，CJMP(510)，CJPN(511)，JME(005)) 正在运行，程序段跳过时，定时器的 PV 会继续计时。（见注）跳过的 TIM/TIMX(550) 指令将不会执行，但是在每次循环程序执行结束处 PV 仍会被刷新。

注 在 CS1D CPU 单元，上述情况 PV 将不会刷新。

当一个 TIM/TIMX(550) 定时器被强制置位，完成标志会变 ON，PV 会被设置为 0000。当一个 TIM/TIMX(550) 定时器被强制复位，完成标志会变 OFF， PV 被 F 复位为 SV。

等于标志和负标志的操作与 CPU 的型号有关。详细参考上述*标志*。

定时器完成标志仅在 TIM/TIMX(550) 执行时刷新，因此，定时器计时到完成标志变 ON 可能有一段到循环周期的延时。

如果用在线编辑把一个定时器转换成相同定时器号的另一类型的定时器，（例如，TIM/TIMX(550)↔TIMH(015)/TIMHX(551)或TIM/TIMX(550)↔ TMHH(540)/TMHHX(552)），一定要复位完成标志，除非完成标志被复位，否则定时器不能正常运作。

一个 TIM/TIMX(550) 指令的 PV 和完成标志依据所用的定时器号不同有下列的刷新方法。

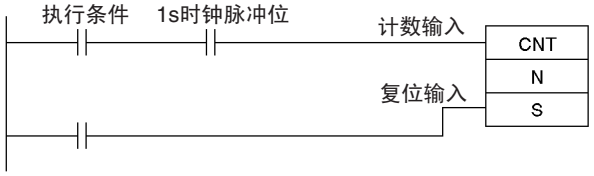
定时器号为 0000 ～ 2047 的定时器

TIM/TIMX(550) 执行	PV 在每次 TIM/TIMX(550) 执行时刷新。 如果 PV 是 0000，完成标志变 ON。 如果 PV 不是 0000，完成标志变 OFF。
执行所有任务后	PV 还在每次循环程序执行结束处更新。
80ms 中断刷新	如果循环时间超过 80ms，定时器 PV 每隔 80ms 通过中断更新。

定时器号为 2048 ～ 4095 的定时器

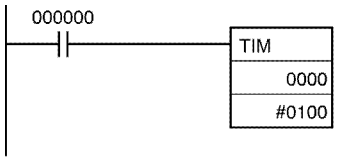
TIM 执行	PV 在每次 TIM 执行时刷新。 如果 PV 是 0000，完成标志变 ON。 如果 PV 不是 0000，完成标志变 OFF。

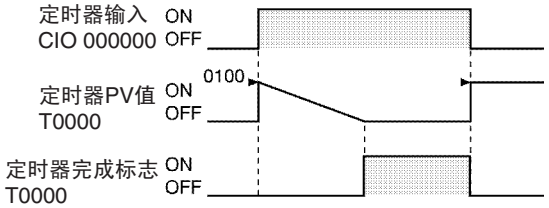
电源断开后，定时器会复位 （PV=SV，完成标志 OFF），除非 IOM 保持位 (A50012)ON，且该位在PLC设置中被保护。也可以用一个时钟脉冲位和一个计数器编一个定时器，它在电源断开的事件中保持 PV 值，如下图所示。



例

在下面例子中，定时器输入 CIO 000000 从 OFF 变到 ON 时，定时器 PV 会开始从 SV 递减计数，在 PV 到达 0000 后，定时器完成标志 T0000 会变 ON。CIO 000000 变 OFF 时，定时器 PV 会复位为 SV，完成标志会变 OFF。





3-6-2 高速定时器：TIMH(015)/TIMHX(551)

用途

TIMH(015)/TIMHX(551) 是单位为 10ms 的递减定时器， TIMH(015) 设置值的设置范围为 0 ～ 99. 99s。TIMHX(551) 设置值的设置范围为 0 ～ 655. 35s。定时器的精度为 0 ～ 0.01s。

注 CS1D CPU 单元的定时器精度是 10 毫秒 + 循环时间。

梯形图符号

PV 刷新方法	符号	操作数
BCD	TIMH(015) N N:定时器号 S S:设置值	N:0000 ～ 4095 （十进制） S:#0000 ～ #9999 (BCD)
二进制	TIMHX(551) N N:定时器号 S S:设置值	N:00000 ～ 4095 （十进制） S:&0 ～ &65535（十进制） #0000 ～ #FFFF （十六进制）

变化

变化		
	ON 条件时每次循环执行	TIMH(015)/ TIMHX(551)
	上升沿微分时执行一次	不支持
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
不允许	OK	OK	不允许

操作数

N: 定时器号

定时器号必须是 0000 ～ 4095 （十进制）

S: 设置值

设置值必须在 #0000 ～ 9999(BCD) 之间。

操作数规定

区	N	S
CIO 区	---	CIO 0000 ～ CIO 6143
工作区	---	W000 ～ W511
保持位区	---	H000 ～ H511
辅助位区	---	A000 ～ A959
定时器区	0000 ～ 4095 （十进制）	T0000 ～ T4095
计数器区	---	C0000 ～ C4095
DM 区	---	D00000 ～ D32767
无区号 EM 区	---	E00000 ～ E32767
有区号 EM 区	---	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	---	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000～@ En_32767 (n = 0 ～ C)

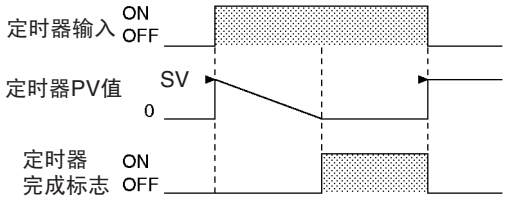
区	N	S
BCD 间接 DM/EM 地址	---	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---	BCD: #0000 ～ 9999 (BCD) “&”不能用 二进制： &0 ～ &65535 （十进制） #0000 ～ #FFFF （十六进制）
数据寄存器	---	DR0 ～ DR15
索引寄存器	---	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15	

描述

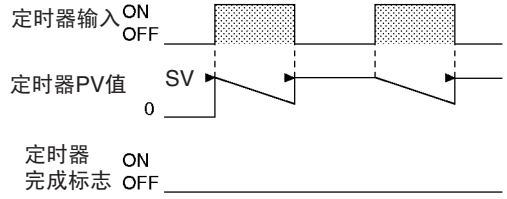
当定时器输入 OFF 时，指定定时器 N 复位，即，定时器的当前值复位到设置值，并且完成标志变 OFF。

当定时器输入从 OFF 到 ON， TIMH(015)/TIMHX(551) 开始从当前值递减，只要定时器输入保持 ON，当前值会连续递减时，且当前值到达 0000 时，定时器完成标志会变 ON。

在定时器计时到后，定时器的 PV 和完成标志状态会保持，要重新启动定时器，定时器输入必须变为 OFF，然后再一次变 ON，或者定时器的 PV 值必须改为一个非零值 （例，使用 MOV(021) 指令）。



下面时间图表示了当定时器计时完成前，定时器输入变 OFF，定时器 PV 值和完成标志的状态。



标志

名称	标记	操作
错误标志	ER	如果 N 通过索引寄存器间接寻址，但是索引寄存器中的地址不是定时器的 PV 的地址时为 ON。 如果是 BCD 模式而 S 不包含 BCD 数据时 ON。 其它情况时 OFF。
等于标志	=	OFF 或不变 （见注）
负标志	N	OFF 或不变 （见注）

注 在 CS1 和 CJ1 CPU 单元中，这些标志变为 OFF。
在 CS1-H， CJ1-H， CJ1M 和 CS1D CPU 单元中，这些标志保持不变。

注意

TIM，TIMX(550)，TIMH(015)，TIMHX(551)，TMHH(540)，TMHHX(552)，TTIM(087)， TTIMX(555)， TIMW(813)， TIMWX(816)， TMHW(815)，和 TMHWX(817) 指令共享定时器号。当程序检查时，如果两个定时器使用相同的定时器号，但并不同时使用，会产生一条重复错误，但定时器仍会正常操作。如果同时使用，使用相同定时器号的定时器不会正常操作。

当 CPU 单元的循环时间超过 80ms 时，定时器号为 2048 ～ 4095 的定时器将不能正常操作。当循环时间超过80ms时，请用定时器号0000～2047的定时器。

定时器号 0000 ～ 0255 的 TIMH(015)/TIMHX(551) 定时器每 10ms 刷新一次。当在用户程序中 PV 值需要刷新时，用这些编号定时器。

即使定时器在待机时，用定时器号 0000 ～ 2047 编程的定时器的当前值也会改变，而用定时器号 2048 ～ 4095 编程的定时器的当前值在待机时会保持。

等于标志和负标志的操作与 CPU 的型号有关。详细参考上述*标志*。

指令 TIMH(015)/TIMHX(551) 执行时，定时器的完成标志会被更新。（这个操作与 CV 系列和 CVM1 PLC 系列不同）。

在下面情况下，定时器会被复位或暂停 （当定时器复位后，它的 PV 值复位到 SV 值，并且完成标志为 OFF）。

条件	PV	完成标志
操作模式从运行或监视模式转换到编程模式或者反过来操作 [1]	0000	OFF
电源断开和复位 [2]	0000	OFF
执行 CNR(545)/CNRX(547) 定时器 / 计数器的复位指令 [3]	BCD:9999 二进制：FFFF	OFF
在互锁程序中 (IL(002)–ILC(003)) 处理	复位到 SV	OFF
在跳转程序中 (JMP(004)–JME(005)) 处理	PV 保持递减	保持以前状态

注 1. 如果 IOM 保持位 (A50012) 已经变 ON，当操作模式被改变，定时器的完成标志和 PV 值的状态保持不变。

2. 如果 IOM 保持位 (A50012) 已经变 ON，并且 IOM 保持位自己的状态在 PLC 设置中保护，即使电源断开，定时器的完成标志和 PV 值的状态仍会保持。
3. 当 TIMH(015)/TIMHX(551) 执行后， PV 值被设置为 SV 值。

当定时器号为 0000 ～ 2047 的 TIMH(015)/TIMHX(551) 的定时器在跳转程序段中 (JMP(004)， CJMP(510)， CJPN(511)， JME(005)) 正在运行，程序段跳过时，定时器的 PV 会继续计时。（见注）（跳过的 TIMH(015)/TIMHX(551) 指令将不会执行，但是 PV 在每 10ms 和每次循环程序执行结束处会被刷新）。

注 在 CS1D CPU 单元，上述情况 PV 将不会刷新。

当 TIMH(015)/TIMHX(551) 在 IL(002) 和 ILC(003) 之间的程序段中，并且程序段被互锁， PV 值会被复位为 SV 值，并且完成标志变 OFF。

当一个 TIMH(015)/TIMHX(551) 定时器被强制置位，完成标志会变 ON，PV 会被设置为 0000。当一个 TIMH(015)/TIMHX(551) 定时器被强制复位，完成标志会变 OFF， PV 被复位为 SV。

等于标志和负标志的操作与 CPU 的型号有关。详细参考上述*标志*。

定时器完成标志仅在 TIMH(015)/TIMHX(551) 执行时刷新，因此，定时器计时到完成标志变 ON 可能有一段到循环周期的延时。

如果用在线编辑把一个定时器转换成相同定时器号的另一类型的定时器，（例如， TIMH(015)/TIMHX(551)↔TIM/TIMX(550) 或 TIMH(015)/TIMHX(551) ↔ TMHH(540)/TMHHX(552)), 一定要复位完成标志，除非完成标志被复位，否则定时器不能正常运作。

一个 TIMH(015) /TIMHX(551) 指令的 PV 和完成标志依据所用的定时器号不同有下列的刷新方法。

定时器号为 0000 ～ 0255 的定时器

TIMH(015)/ TIMHX(551) 执行	如果 PV 是 0000，完成标志变 ON。 如果 PV 不是 0000，完成标志变 OFF。
10ms 中断刷新	定时器 PV 每隔 10ms 中断更新。

定时器号为 0256 ～ 2047 的定时器

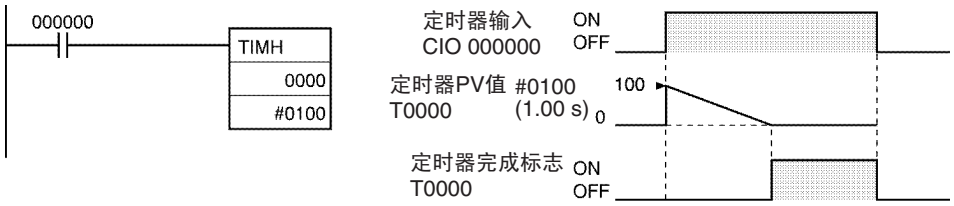
TIMH(015)/ TIMHX(551) 执行	PV 在每次 TIMH(015)/TIMHX(551) 执行时刷新。 如果 PV 是 0000，完成标志变 ON。 如果 PV 不是 0000，完成标志变 OFF。
执行所有任务后	PV 还在每次循环程序执行结束处更新。
80ms 中断刷新	如果循环时间超过 80ms，定时器 PV 每隔 80ms 通过中断更新。

定时器号为 2048 ～ 4095 的定时器

TIMH(015)/ TIMHX(551) 执行	PV 在每次 TIMH(015) 执行时刷新。 如果 PV 是 0000，完成标志变 ON。 如果 PV 不是 0000，完成标志变 OFF。

例

在下面例子中，定时器输入 CIO 000000 从 OFF 变到 ON 时，定时器 PV 会开始从 SV(#0064=100=1.00s) 递减计数，在 PV 到达 0000 后，定时器完成标志 T0000 会变 ON。
CIO 000000 变 OFF 时，定时器 PV 会复位为 SV，完成标志会变 OFF。



3-6-3 1MS 定时器：TMHH(540)/TMHHX(552)

用途

TMHH(540)/TMHHX(552) 是单位为 1ms 的递减定时器，TMHH(540) 设置值的设置范围为 0 ～ 9.999 s。 TMHHX(552) 设置值的设置范围为 0 ～ 65. 535s。定时器的精度为 -0.001 ～ 0 s。

注 CS1D CPU 单元的定时器精度是 10 毫秒 + 循环时间

梯形图符号

PV 刷新方法	符号	操作数
BCD	TMHH(540) N　N:定时器号 S　S:设置值	N:0000 ～ 15 （十进制） S:#0000 ～ #9999 (BCD)
二进制	TMHHX(552) N　N:定时器号 S　S:设置值	N:00000 ～ 15 （十进制） S:&0 ～ &65535 （十进制） #0000 ～ #FFFF （十六进制）

变化

变化		TMHH(540)/ TMHHX(552)
	ON 条件时每次循环执行	
	上升沿微分时执行一次	不支持
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

使用程序区

块程序区	步程序区	子程序	中断任务
不允许	OK	OK	不允许

操作数

N: 定时器号

定时器号必须是 0000 ～ 0015 之间 （十进制）

S: 设置值

设置值必须在 #0000 ～ 9999(BCD) 之间。

操作数规定

区	N	S
CIO 区	---	CIO 0000 ～ CIO 6143
工作区	---	W000 ～ W511
保持位区	---	H000 ～ H511
辅助位区	---	A000 ～ A959
定时器区	0000 ～ 0015 （十进制）	T0000 ～ T4095
计数器区	---	C0000 ～ C4095
DM 区	---	D00000 ～ D32767
无区号 EM 区	---	E00000 ～ E32767
有区号 EM 区	---	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	---	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 地址	---	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---	BCD： #0000 ～ 9999 (BCD) “&”不能用 二进制： &0 ～ &65535 （十进制） #0000 ～ #FFFF （十六进制）
数据寄存器	---	DR0 ～ DR15
索引寄存器	---	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15	

描述

当定时器输入 OFF 时，指定定时器 N 复位，即定时器的当前值复位到设置值，并且完成标志变 OFF。

当定时器输入从 OFF 到 ON， TMHH(540)/TMHHX(552) 开始从当前值递减，只要定时器输入保持 ON，当前值会连续递减时，且当前值到达 0000 时，定时器完成标志会变 ON。

在定时器计时到后，定时器的 PV 和完成标志状态会保持，要重新启动定时器，定时器输入必须变为 OFF，然后再一次变 ON，或者定时器的 PV 值必须改为一个非零值 （例，使用 MOV(021) 指令）。

标志

名称	标记	操作
错误标志	ER	如果 N 通过索引寄存器间接寻址，但是索引寄存器中的地址不是定时器的 PV 的地址时为 ON。 如果是 BCD 模式而 S 不包含 BCD 数据时 ON。 其它情况时 OFF。
等于标志	=	OFF 或不变 （见注）
负标志	N	OFF 或不变 （见注）

注 在 CS1 和 CJ1 CPU 单元中，这些标志变为 OFF。
在 CS1-H， CJ1-H， CJ1M 和 CS1D CPU 单元中，这些标志保持不变。

注意 TIM， TIMX(550)， TIMH(015)， TIMHX(551)， TMHH(540)， TMHHX(552) ， TTIM(087) ， TTIMX(555) ， TIMW(813) ， TIMWX(816) ,TMHW(815)，和 TMHWX(817) 指令共享定时器号。当程序检查时，如果两个定时器使用相同的定时器号，但并不同时使用，会产生一条重复错误，但定时器会正常操作。如果同时使用，使用相同定时器号的定时器不会正常操作。

定时器完成标志仅在 TMHH(540)/TMHHX(552) 执行时刷新，因此，定时器计时到完成标志变 ON 可能有一段到循环周期的延时。

即使定时器在待机时，用定时器号 0000 ～ 2047 编程的定时器的当前值仍会改变，而用定时器号 2048 ～ 4095 编程的定时器的当前值在待机时会保持。

在下面情况下，定时器会被复位或暂停 （当定时器复位后，它的 PV 值复位到 SV 值，并且完成标志为 OFF）。

条件	PV	完成标志
操作模式从运行或监视模式转换到编程模式或者反过来操作 [1]	0000	OFF
电源断开和复位 [2]	0000	OFF
执行 CNR(545)/CNRX(547) 定时器 / 计数器的复位指令 [3]	BCD：9999 二进制： FFFF	OFF
在互锁程序中 (IL(002)–ILC(003)) 处理	复位到 SV	OFF
在跳转程序中 (JMP(004)–JME(005)) 处理	PV 保持递减	保持以前状态

注
1. 如果 IOM 保持位 (A50012) 已经变 ON，当操作模式改变后，定时器的完成标志和 PV 值的状态会保持。
2. 如果 IOM 保持位 (A50012) 已经变 ON，并且 IOM 保持位自己的状态在 PLC 设置中保护，即使电源断开，定时器的完成标志和 PV 值的状态仍会保持。
3. 当 TMHH(540)/TMHHX(552) 执行后， PV 值被设置为 SV 值。

当定时器号为 0000 ～ 2047 的 TMHH(540)/TMHHX(552) 定时器在跳转程序段中 (JMP(004)， CJMP(510)， CJPN(511)， JME(005)) 正在运行，程序段跳过时，定时器的 PV 会继续计时。（见注）（跳过的 TMHH(540)/TMHHX(552) 指令将不会执行，但是 PV 每隔 1ms 会被刷新）。

注 在 CS1D CPU 单元，上述情况 PV 将不会刷新。

当 TMHH(540)/TMHHX(552) 在 IL(002) 和 ILC(003) 之间的程序段中，并且程序段被互锁， PV 值会被复位为 SV 值，并且完成标志变 OFF。

当一个 TMHH(540)/TMHHX(552) 定时器被强制置位，完成标志会变 ON， PV 会被设置为 0000。当一个 TMHH(540)/TMHHX(552) 定时器被强制复位，完成标志会变 OFF， PV 被复位为 SV。

等于标志和负标志的操作与 CPU 的型号有关。详细参考上述*标志*。

如果用在线编辑把一个定时器转换成相同定时器号的另一类型的定时器，（例如，TMHH(540)/TMHHX(552) ↔ TIM/TIMX(550) 或 TMHH(540)/TMHHX(552) ↔ TIMH(015)/TIMHX(551)），一定要复位完成标志，除非完成标志被复位，否则定时器不能正常运作。

一个 TMHH(540)/TMHHX(552) 指令的 PV 和完成标志的刷新如下表所示。

TMHH(540)/ TMHHX(552) 执行	如果 PV 是 0000，完成标志变 ON。 如果 PV 不是 0000，完成标志变 OFF。
1ms 中断刷新	定时器 PV 每隔 1ms 中断刷新。

3-6-4 累积定时器：TTIM(087)/TTIMX(555)

用途

TTIM(087) 或 TTIMX(550) 是单位为 0.1s 的递减定时器， TTIM(087) 设置值的设置范围为 0 ～ 999.9 s。TIMHX(551) 设置值的设置范围为 0 ～ 6,553. 5s。定时器的精度为 -0.01 ～ 0 s。

注 CS1D CPU 单元的定时器精度是 10 毫秒 + 循环时间。

梯形图符号

PV 刷新方法	符号	操作数
BCD	定时器输入 — TTIM(087) N — N:定时器号 S — S:设置值 复位输入	N:0000 ～ 15（十进制） S:#0000 ～ #9999 (BCD)
二进制	定时器输入 — TTIMX(555) N — N:定时器号 S — S:设置值 复位输入	N:00000 ～ 15（十进制） S:&0 ～ &65535（十进制） #0000 ～ #FFFF（十六进制）

变化

变化	ON 条件时每次循环执行	TTIM(087)/ TTIMX(555)
	上升沿微分时执行一次	不支持
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
不允许	OK	OK	不允许

操作数

N: 定时器号

定时器号必须是在 0000 ～ 4095（十进制）之间。

S: 设置值

设置值必须在 #0000 ～ 9999(BCD) 之间。

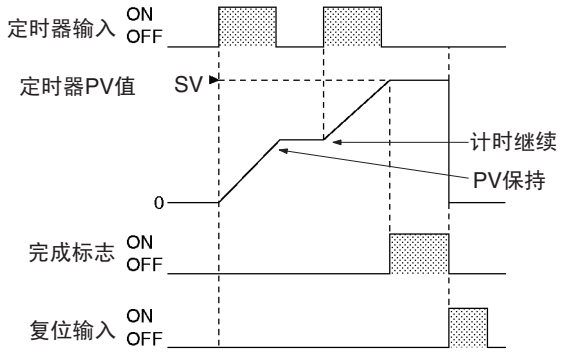
操作数规定

区	N	S
CIO 区	---	CIO 0000 ～ CIO 6143
工作区	---	W000 ～ W511
保持位区	---	H000 ～ H511
辅助位区	---	A000 ～ A959
定时器区	0000 ～ 4095 （十进制）	T0000 ～ T4095
计数器区	---	C0000 ～ C4095
DM 区	---	D00000 ～ D32767
无区号 EM 区	---	E00000 ～ E32767
有区号 EM 区	---	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	---	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 地址	---	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---	BCD: #0000 ～ 9999 (BCD) “&”不能用 二进制： &0 ～ &65535 （十进制） #0000 ～ #FFFF （十六进制）
数据寄存器	---	DR0 ～ DR15
索引寄存器	---	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15	

描述

当定时器输入 ON，TTIM(087)/TTIMX(555) 开始从当前值递增。当定时器输入 OFF，定时器当前值会停止递增，但保持原值。当定时器输入又为 ON，定时器继续计时。当前值 PV 到达 SV 时，定时器完成标志会变 ON。

定时器计时到后，定时器的 PV 和完成标志状态会保持。有三种方法重新启动定时器：定时器的 PV 值必须改为一个非零值 （例，使用 MOV(021) 指令）。复位输入变为 ON，或执行 CNR(545)/CNRX(547)。



标志

名称	标记	操作
错误标志	ER	如果 N 通过索引寄存器间接寻址，但是索引寄存器中的地址不是定时器的 PV 地址时为 ON。 如果是 BCD 模式而 S 不包含 BCD 数据时 ON。 其它情况时 OFF。

注意

TIM， TIMX(550)， TIMH(015)， TIMHX(551)， TMHH(540)， TMHHX(552)， TTIM(087)， TTIMX(555)， TIMW(813)， TIMWX(816)， TMHW(815)，和 TMHWX(817) 指令共享定时器号。如果两个定时器使用相同的定时器号，但并不同时使用，当程序检查时，会产生一条重复错误，但定时器会正常操作。如果同时使用，使用相同定时器号的定时器不会正常操作。

在下面情况下，定时器会被复位或暂停 （当定时器复位后，它的 PV 值复位到 SV 值，并且完成标志为 OFF）。

条件	PV	完成标志
操作模式从运行或监视模式转换到编程模式或者反过来 [1]	0000	OFF
电源断开和复位 [2]	0000	OFF
执行 CNR(545)/CNRX(547) 定时器 / 计数器的复位指令 [3]	BCD:9999 二进制： FFFF	OFF
在互锁程序中 (IL(002)–ILC(003)) 处理	保持以前状态	保持以前状态
在跳转程序中 (JMP(004)–JME(005)) 处理	保持以前状态	保持以前状态

注 1. 如果 IOM 保持位 (A50012) 已经变 ON，当操作模式改变后，定时器的完成标志和 PV 值的状态会保持。

2. 如果 IOM 保持位 (A50012) 已经变 ON，并且 IOM 保持位自己的状态在 PLC 设置中保护，即使电源断开，定时器的完成标志和 PV 值的状态仍会保持。

3. 当 TTIM(087)/TTIMX(555) 执行后， PV 值被设置为 SV 值。

当 TTIM(087)/TTIMX(555) 在 IL(002) 和 ILC(003) 之间的程序段中，并且程序段是互锁的，PV值保持以前状态（它不会被复位）。当TTIM(087)/TTIMX(555）编在 IL(002) 和 ILC(003) 之间的程序段中时，必须考虑这一情况。

当 TTIM(087)/TTIMX(555) 在跳转程序段 JMP(004) 和 JME(005) 之间操作，该程序段被跳过，定时器的 PV 保持以前状态。当 TTIM(087)/TTIMX(555) 编在跳转程序段 JMP(004) 和 JME(005) 之间时必须考虑这一情况。

当一个 TTIM(087)/TTIMX(555) 定时器被强制置位，完成标志会变 ON， PV 会被设置为 0000。当一个 TTIM(087)/TTIMX(555) 定时器被强制复位，完成标志会变 OFF，PV 被复位为 0000 。强制置位和强制复位操作优先于定时器状态输入和复位输入。

定时器 PV 仅在 TTIM(087)/TTIMX(555) 执行时刷新，因此，当循环时间超过 100ms，定时器不会正常工作，因为定时器递增是以 100ms 为单位的。

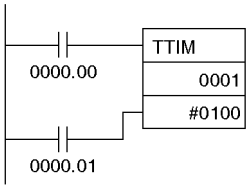
定时器完成标志仅在 TTIM(087)/TTIMX(555) 执行时刷新，因此，定时器计时到完成标志变 ON 可能有一段到循环周期的延时。

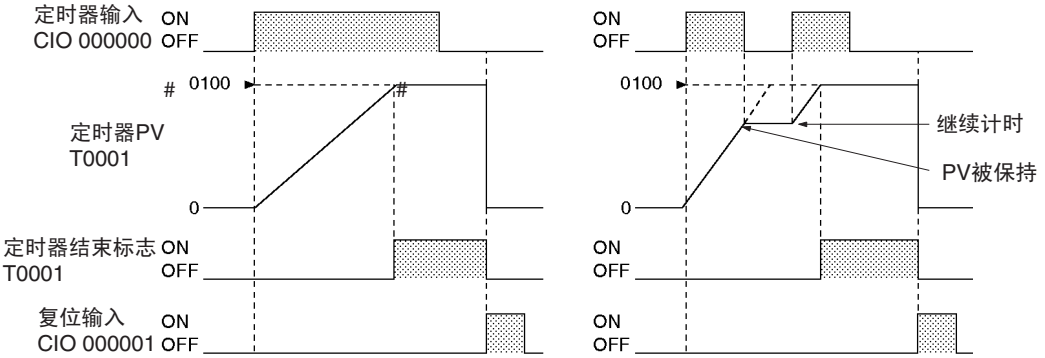
典型的定时器，如 TIM/TIMX(550)，是递减计数器，直到计时结束，PV 显示剩下的时间。TTIM(087)/TTIMX(555) 的 PV 显示用去了多少时间，所以，在许多计算和显示输出中能不作变换就使用。

例

在下面例子中，定时器输入 CIO 000000 为 ON 时，定时器 PV 会开始从 0 计数。当 PV 到达 SV 时，定时器完成标志 T0000 会变 ON。
如果复位输入为 ON，定时器 PV 复位到 0000，完成标志 (T0000) 变为 OFF。（通常为复位定时器，复位输入先为 ON，然后，定时器输入为 ON，启动计时）

如果在到达 SV 之前，定时器输入变为 OFF，定时器将停止计时，但 PV 仍保持原值。定时器输入又变为 ON，定时器将从原来的 PV 值开始计时。





3-6-5 长定时器：TIML(542)/TIMLX(553)

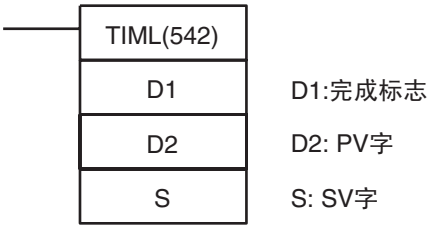
用途

TTIM(087) 或 TTIMX(550) 是单位为 1s 的递减定时器，TTIM(087) 能计时 115 天。 TIMHX(551) 能计时 49,710 天。定时器的精度为 0 ～ 0.01s。

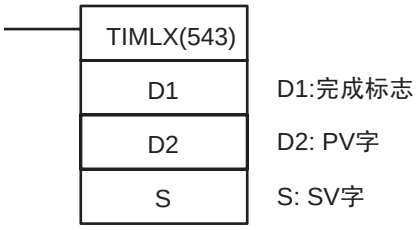
注 CS1D CPU 单元的定时器精度是 10ms+ 循环时间。

梯形图符号

BCD



二进制



变化

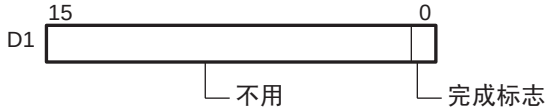
变化	ON 条件时每次循环执行	TIML(542)/ TIMLX(553)
	上升沿微分时执行一次	不支持
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
不允许	OK	OK	不允许

操作数

D1: 完成标志
D1 的位 0 作为 TIML(542)/TIMLX(553) 的完成标志。



D2: PV 字
D2+1 和 D2 包括了 8 位二进制或 BCD PV。（D2+1 和 D2 必须在同一数据区）。PV 的范围对于 TIML(542) 是 #00000000 ～ #99999999（十进制），对于 TIMLX(553) 是 #00000000 ～ #44294967294（十进制）或 #00000000 ～ #FFFFFFFF（十六进制）。

S: SV 字
S+1 和 S 包括了 8 位二进制或 BCD SV。（S+1 和 S 必须在同一数据区）。设置值对于 TIML(542) 必须在 #00000000 ～ #99999999（十进制）之间，对于 TIMLX(553) 必须在 #00000000 ～ #44294967294（十进制）或 #00000000 ～ #FFFFFFFF（十六进制）之间。

操作数规定

区域	D1	D2	S
CIO 区	CIO 0000 ～ CIO 6143	CIO 0000 ～ CIO 6142	
工作区	W000 ～ W511	W000 ～ W510	
保持位区	H000 ～ H511	H000 ～ H510	
辅助位区	A448 ～ A959	A448 ～ A958	A000 ～ A958
定时器区	---	---	T0000 ～ T4094
计数器区	---	---	C0000 ～ C4094
DM 区	D00000～D32767	D00000 ～ D32766	
无区号 EM 区	E00000～E32767	E00000 ～ E32766	

区域	D1	D2	S
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	En_00000 ～ En_32766 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---		BCD: #00000000 ～ 99999999 (BCD) 不能用 “&” 二进制: &00000000 ～ &4294967294（十进制）或 #00000000 ～ #FFFFFFFF（十六进制）
数据寄存器	---		
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15		

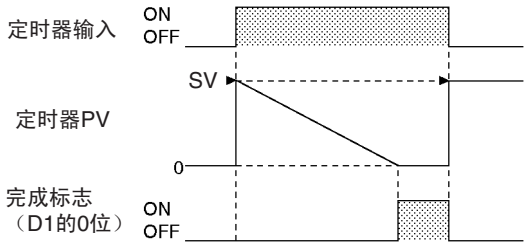
描述

TIML(542)/TIMLX(553) 是单位为 0.1s 的减时 ON- 延时定时器，它用 8 位 SV 和 8 位 PV。

当定时器输入 OFF 时，定时器复位，即，定时器的当前值复位到设置值，并且完成标志变 OFF。

当定时器输入由 OFF 变为 ON，TIML(542)/TIMLX(553) 开始减 D2+1 和 D2 中的 PV。只要定时器输入保持 ON，PV 会继续往下计时，在 PV 到达 00000000 时，完成标志会变 ON。

在定时器计时到后，定时器的 PV 和完成标志状态会保持。要重新启动定时器时，定时器输入必须变为 OFF，然后再次 ON，或者定时器的 PV 值必须改为一个非零值 （例，使用 MOV(021) 指令）。



标志

名称	标记	操作
错误标志	ER	D2+1 和 D2 中包含的 PV 值不是 BCD 时 ON。 S+1 和 S 中包含的 SV 值不是 BCD 时 ON。 其它情况时 OFF。

注意

与大多数定时器不同，TIML(542)/TIMLX(553) 不使用定时器号。（定时器区 PV 刷新对 TIML(542)/TIMLX(553) 不执行）。

由于 TIML(542)/TIMLX(553) 完成标志在数据区，因此可以象其它位一样强制置位和强制复位，但 PV 不会改变。

定时器 PV 仅在 TIML(542)/TIMLX(553) 执行时刷新，因此，当循环时间超过 100ms，定时器不会正常工作，因为定时器递增是以 100ms 为单位的。

定时器完成标志仅在 TIML(542)/TIMLX(553) 执行时刷新，因此，定时器计时到完成标志变 ON 可能有一段到循环周期的延时。

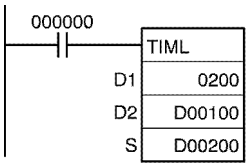
当 TIML(542)/TIMLX(553) 在 IL(002) 和 ILC(003) 之间的程序段中，并且程序段是互锁的， PV 被复位 SV，完成标志变 OFF。

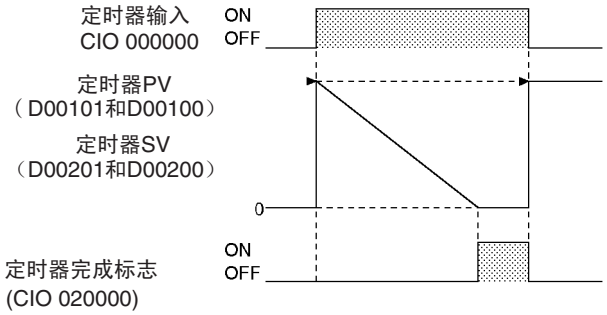
当 TIML(542)/TIMLX(553) 在跳转程序段 JMP(004) 和 JME(005) 之间操作，该程序段被跳过，定时器的 PV 保持以前状态。当 TIML(542)/TIMLX(553) 编在跳转程序段 JMP(004) 和 JME(005) 之间时必须考虑这一情况。

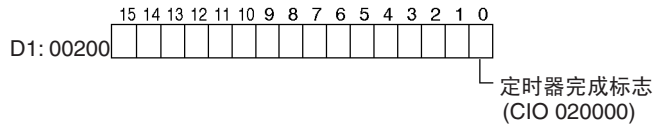
确定完成标志和 PV 所指定的字（D1， D2 和 D2+1）没有被用在其它指令中，如果这些字受其它指令影响，定时器可能不会正确计时。

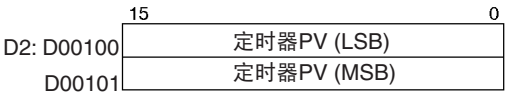
例

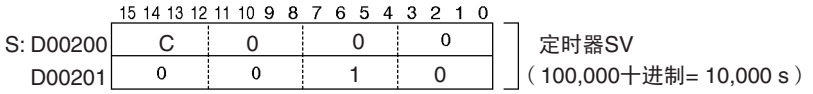
在下面例子中，定时器输入 CIO 000000 为 ON 时，定时器 PV （D00101 和 D00100）会被置成 SV （D00201 和 D00200），并且 PV 会开始往下计时，在 PV 到达 00000000 时，定时器完成标志 (CIO 020000) 会变 ON。
CIO 000000 为 OFF 时，定时器 PV 复位到 SV，完成标志 (CIO 020000) 变为 OFF。











3-6-6 多输出定时器：MTIM(543)/MTIMX(554)

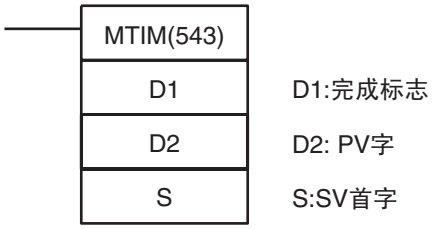
用途

MTIM(543)/MTIMX(554) 是一个具有 8 个独立的 SV 和完成标志的 0.1s 的递增定时器， MTIM(543) 的设定值为 0 ～ 999.9s， MTIMX(554) 的设定值为 0 ～ 6,553.5s。定时器的精度为 0 ～ 0.01s。

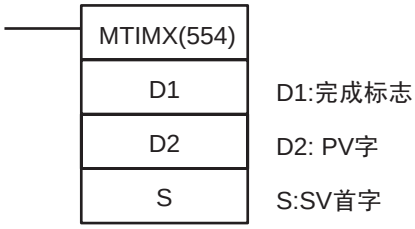
注 CS1D CPU 单元的定时器精度是 10ms+ 循环时间。

梯形图符号

BCD



二进制



变化

变化	ON 条件时每次循环执行	MTIM(543)/ MTIMX(554)
	上升沿微分时执行一次	不支持
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

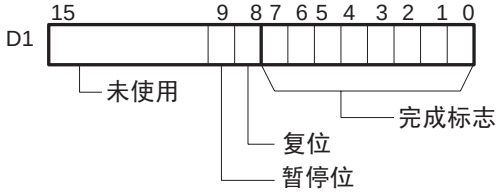
适用程序区

块程序区	步程序区	子程序	中断任务
不允许	OK	OK	不允许

操作数

D1: 完成标志

D1 包括了 8 位完成标志以及暂停和复位位。



D2: PV 字

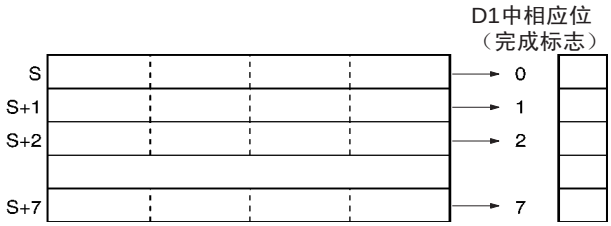
D2 包括了 4 个数字二进制或 BCD PV。

数据	范围
BCD	#0000 ～ #9999
二进制	&0 ～ &65535 （十进制） #0000 ～ #FFFF （十六进制）

S:SV 设置值首字

S ～ S+7 包括了 8 个独立的 SV。每个 SV 如下表所示：

数据	范围
BCD	#0000 ～ #9999
二进制	&0 ～ &65535 （十进制） #0000 ～ #FFFF （十六进制）



数据	范围
BCD	用于 8 个定时器中每个 SV 的一个字： #0000 ～ #9999
二进制	用于 8 个定时器中每个 SV 的一个字： &0 ～ &65535 （十进制） #0000 ～ #FFFF （十六进制）

注 S ～ S+7 必须在同一数据区内。

操作数规定

区域	D1	D2	S
CIO 区	CIO 0000 ～ CIO 6143		CIO 0000 ～ CIO 6136
工作区	W000 ～ W511		W000 ～ W504
保持位区	H000 ～ H511		H000 ～ H504
辅助位区	A448 ～ A959		A000 ～ A952
定时器区	T0000 ～ T4095		T0000 ～ T4088
计数器区	C0000 ～ C4095		C0000 ～ C4088
DM 区	D00000 ～ D32767		D00000 ～ D32760
无区号 EM 区	E00000 ～ E32767		E00000 ～ E32760
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		En_00000 ～ En_32760 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---		
数据寄存器	---	DR0 ～ DR15	---

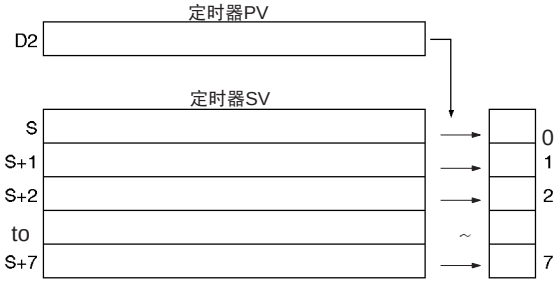
区域	D1	D2	S
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15		

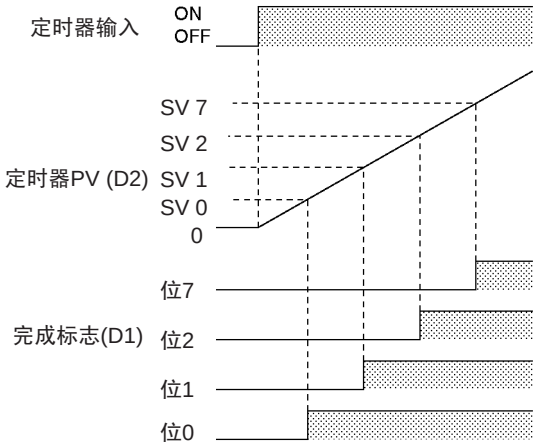
描述

MTIM(543)/MTIMX(554) 执行条件 ON，复位和定时器都 OFF 时，MTIM(543)/MTIMX(554) 在 D2 中增加 PV 值。如果暂停位 ON，定时器会停止增加 PV，但 PV 会保持原值，当暂停位再次变 OFF， MTIM(543)/MTIMX(554) 恢复计时。

每次 MTIM(543)/MTIMX(554) 执行后， PV （D2 中的内容）会和 S ～ S+7 中的 8 个 SV 相比较。如果其中一些小于或等于 PV，相应完成标志 （D1 位 00 ～ 07）会变 ON。

当 PV 到达 9999， PV 将被复位为 0000 并且所有完成标志会变 OFF。如果当定时器正在运行或暂停时复位位变 ON， PV 会复位到 0000，并且所有完成标志会变 OFF。





下表显示了 MTIM(543)/MTIMX(554) 的复位位和暂停位组合的 4 种可能操作。

复位位 （位 08）	暂停位 （位 09）	操作
OFF	OFF	当 SV ≤ PV 时， PV 会被更新，相应的完成标志会变 ON。
	ON	PV 不会被更新， MTIM(543)/MTIMX(554) 作为 NOP(000) 对待。

复位位（位 08）	暂停位（位 09）	操作
ON	OFF	PV 复位到 0000，并且完成标志会变 OFF，PV 不会被更新。
	ON	

仅当 MTIM(543)/MTIMX(554) 的执行条件 ON 时，复位和暂停才有效。

标志

名称	标记	操作
错误标志	ER	D2 中包含的 PV 值不是 BCD 时 ON。 其它情况时 OFF。

注意

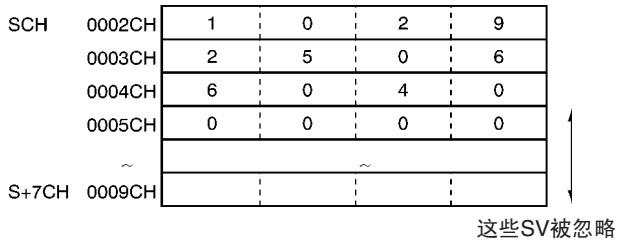
与大多数定时器不同， MTIM(543)/MTIMX(554) 不使用定时器号。（定时器区 PV 刷新对 MTIM(543)/MTIMX(554) 不执行）。

当 PV 到达 9999， PV 会复位到 0000，并且所有完成标志会变 OFF。

如果在 BCD 模式下 ,S ～ S+7 中的 SV 不包含 BCD 数据，SV 会被忽略，不会发生错误，并且错误标志不会变 ON。

由于 MTIM(543)/MTIMX(554) 完成标志在数据区，因此可以象其它位一样强制置位和强制复位，但 PV 不会改变。

当使用 8 个或更少的 SV，设置最后 SV 后面的字为 0000， MTIM(543)/MTIMX(554) 会忽略值为 0000 的 SV 及余下所有的 SV。



定时器 PV 仅在 MTIM(543)/MTIMX(554) 执行时刷新，因此，当循环时间超过 100ms，定时器不会正常工作，因为定时器递增是以 100ms 为单位的。要保证精确计时及防止长循环时间出现问题，在程序的几个地方输入相同的 MTIM(543)/MTIMX(554) 指令。

定时器完成标志仅在 MTIM(543)/MTIMX(554) 执行时刷新，因此，定时器计时到完成标志变 ON 可能有一段到循环周期的延时。

当 MTIM(543)/MTIMX(554) 在 IL(002) 和 ILC(003) 之间的程序段中，并且程序段是互锁的，定时器的 PV 保持以前状态（它不会被复位）。当 MTIM(543)/MTIMX(554) 编在连锁程序段 IL(002) 和 ILC(003) 之间时必须考虑这一情况。

当 MTIM(543)/MTIMX(554) 在跳转程序段 JMP(004) 和 JME(005) 之间操作，该程序段被跳过，定时器的 PV 保持以前状态。当 MTIM(543)/MTIMX(554) 编在跳转程序段 JMP(004) 和 JME(005) 之间时必须考虑这一情况。

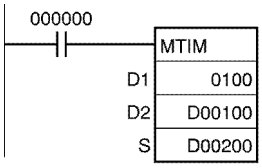
确定完成标志和 PV 所指定的字 （D1 和 D2）没被用在其它指令中，如果这些字受其它指令影响，定时器可能不会正确计时。

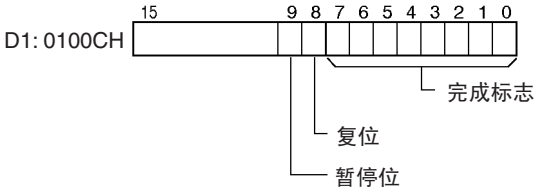
如果 D1 指定为 CIO 区域中的字，SET 和 RSET 指令可用来控制暂停和复位位。

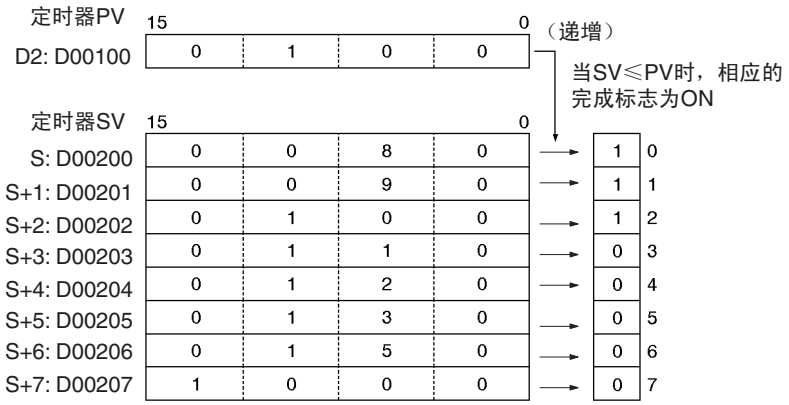
例

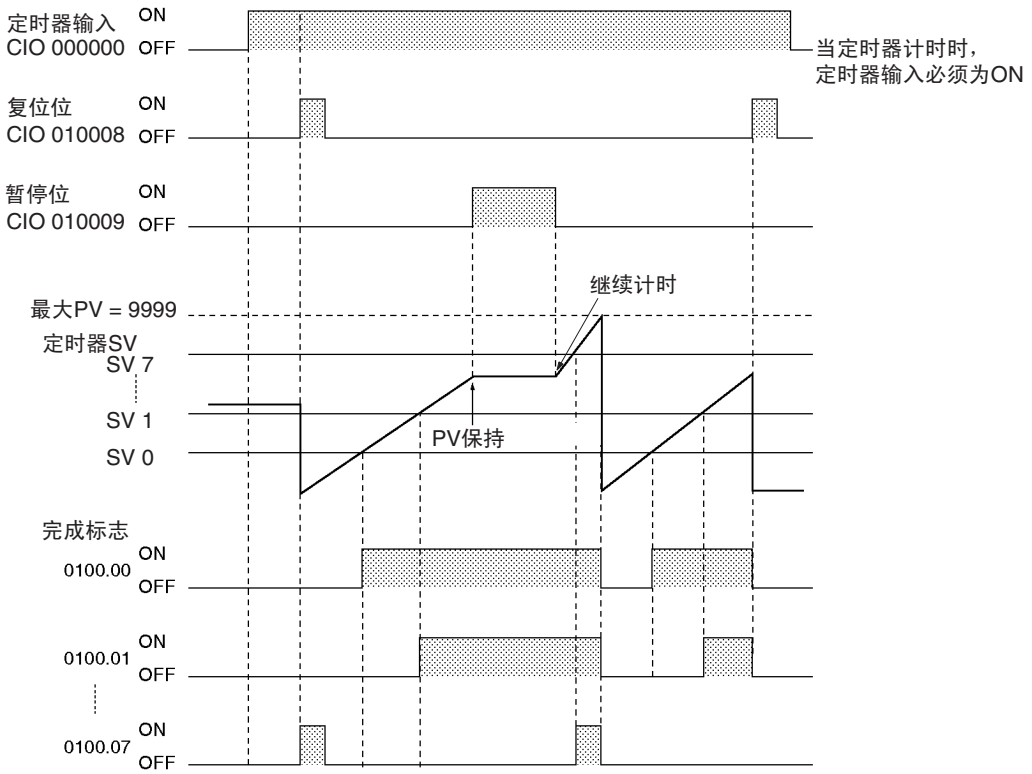
在下面例子中，CIO 000000 为 ON，暂停位 (CIO 010009) 为 OFF，当复位位 (CIO 010008) 从 ON 变为 OFF，定时器 PV 从 0000 往上计时。

D00200 ～ D002007 中的 8 个 SV 会和 PV 比较，当 SV ≤ PV 时，相应的完成标志 (CIO 010000 ～ CIO 10007) 变为 ON。









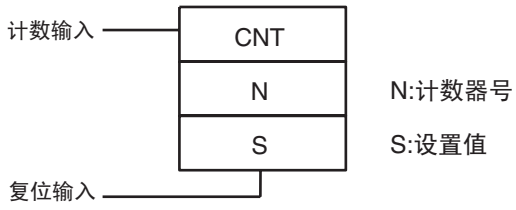
3-6-7 计数器：CNT/CNTX(546)

用途

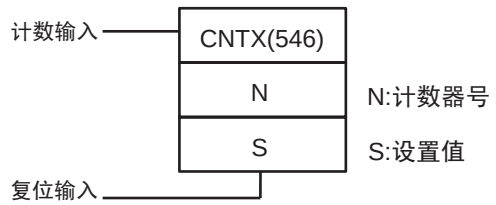
CNT/CNTX(546) 是递减计数器。设置值范围对 CNT 为 0 ～ 9,999 对 CNTX(546) 为 0 ～ 65,535。

梯形图符号

BCD



二进制



变化

变化	ON 条件时每次循环执行	CNT/ CNTX(546)
	上升沿微分时执行一次	不支持
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
不允许	OK	OK	OK

操作数

N: 计数器号
定时器号必须是在 0000 ～ 4095 间 （十进制）

S: 设置值

数据	范围
BCD	#0000 ～ #9999
二进制	&0 ～ &65535 （十进制） #0000 ～ #FFFF （十六进制）

操作数规定

区域	N	S
CIO 区	---	CIO 0000 ～ CIO 6143
工作区	---	W000 ～ W511
保持位区	---	H000 ～ H511
辅助位区	---	A000 ～ A959
定时器区	---	T0000 ～ T4095
计数器区	0000 ～ 4095 （十进制）	C0000 ～ C4095
DM 区	---	D00000 ～ D32767
无区号 EM 区	---	E00000 ～ E32767
有区号 EM 区	---	En_00000 ～ En_32767 (n = 0 ～ C)

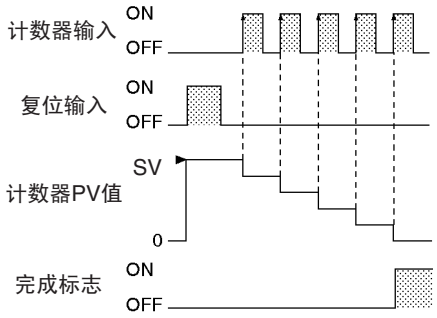
区域	N	S
二进制间接 DM/EM 地址	---	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 地址	---	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---	BCD: #0000 ～ 9999 (BCD) “&”不能用 二进制： &0 ～ &65535 （十进制） #0000 ～ #FFFF（十六进制）
数据寄存器	---	DR0 ～ DR15
索引寄存器	---	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15,IR0 ～ IR15	

描述

每次计数器输入从 OFF 到 ON，计数器 PV 减 1。当 PV 到 0，计数器完成标志变为 ON。

一旦完成标志变为 ON，使复位输入为 ON 或用 CNR(545)/CNRX(547) 复位计数器。否则，计数器不能启动。

当复位输入为 ON，计数器被复位，计数输入被忽略。（当计数器被复位，它的 PV 值被复位，完成标志变为 OFF）。



标志

名称	标记	操作
错误标志	ER	如果 N 通过索引寄存器间接寻址，但是索引寄存器中的地址不是计数器的 PV 的地址时为 ON。 如果 BCD 模式下，而 S 不包含 BCD 数据时 ON。 其它情况时 OFF。
等于标志	=	OFF 或不变 （见注）
负标志	N	OFF 或不变 （见注）

注 在 CS1 和 CJ1 CPU 单元中，这些标志变为 OFF。
在 CS1-H， CJ1-H， CJ1M 和 CS1D CPU 单元中，这些标志保持不变。

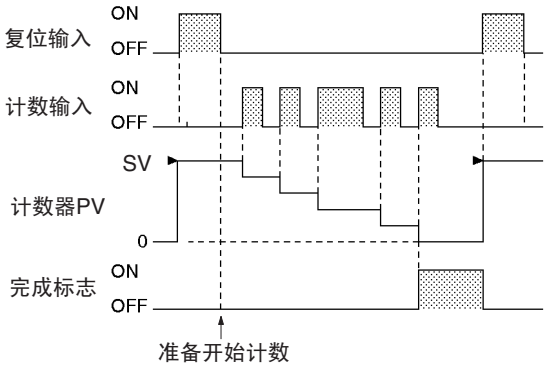
注意

CNT，CNTX(546)，CNTR(012)，CNTRX(548)，CNTW(814) 和 CNTWX(818) 指令共享计数器号。如果两个计数器使用相同的计数器号，但并不同时使用，当程序检查时，会产生一个重复错误，但计数器会正常操作。如果同时使用，使用相同计数器号的计数器不会正常操作。

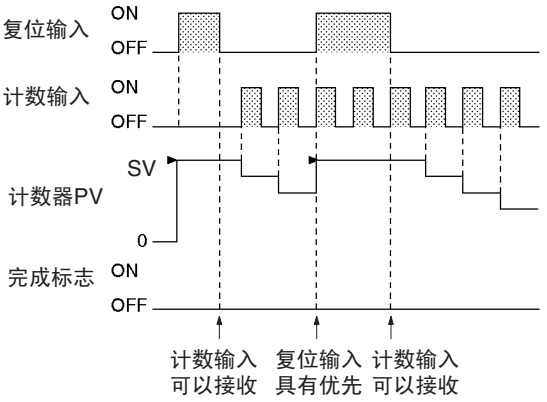
当计数器输入从 OFF 到 ON，计数器 PV 刷新，每次 CNT/ CNTX(546) 执行后，完成标志被刷新，如果 PV 是 0，完成标志变为 ON。如果 PV 不是 0，完成标志变为 OFF。

当一个 CNT/CNTX(546) 计数器被强制置位，完成标志会变 ON， PV 复位为 0000。当一个 CNT/CNTX(546) 计数器被强制复位，完成标志会变 OFF， PV 置位为 SV。

在用计数输入开始计数前，一定要使计数输入从 OFF → ON → OFF 来复位计数器，如下图所示。如果复位输入为 ON，计数输入不会被接收。

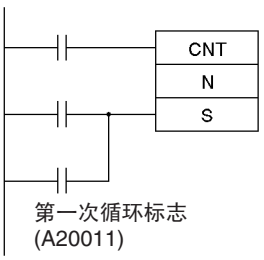


复位输入有优先权，如果复位输入和计数输入同时 ON，计数器会复位。（PV 会复位到 SV，完成标志也会变 OFF）。



等于标志和负标志的操作与 CPU 的型号有关。详细参考上述*标志*。

即使电源中断，计数器 PV 仍然保持，如果希望从 SV 开始计数，而不是从保持的 PV 恢复计数，增加第一次循环标志 (A20011) 作为计数器的复位输入。



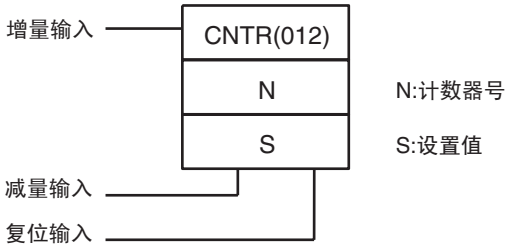
3-6-8 可逆计数器 :CNTR(012)/CNTRX(548)

用途

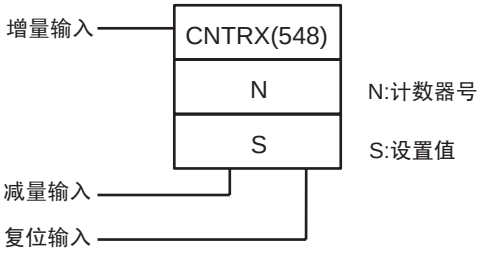
CNTR(012)/CNTRX(548) 操作一个可逆计数器 .

梯形图符号

BCD



二进制



变化

变化	ON 条件时每次循环执行	CNTR(012)/ CNTRX(548)
	上升沿微分时执行一次	不支持
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
不允许	OK	OK	OK

操作数

N: 计数器号

定时器号必须是在 0000 ～ 4095 间 （十进制）

S: 设置值

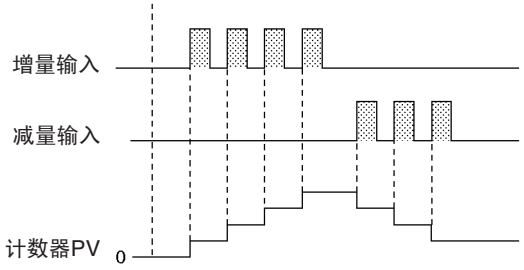
数据	范围
BCD	#0000 ～ #9999
二进制	&0 ～ &65535 （十进制） #0000 ～ #FFFF （十六进制）

操作数规定

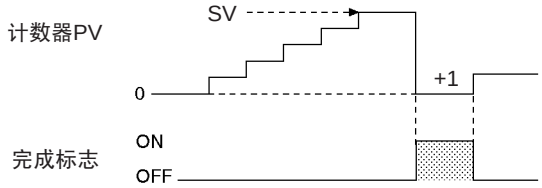
区域	N	S
CIO 区	---	CIO 0000 ～ CIO 6143
工作区	---	W000 ～ W511
保持位区	---	H000 ～ H511
辅助位区	---	A000 ～ A959
定时器区	---	T0000 ～ T4095
计数器区	0000 ～ 4095（十进制）	C0000 ～ C4095
DM 区	---	D00000 ～ D32767
无区号 EM 区	---	E00000 ～ E32767
有区号 EM 区	---	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	---	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 地址	---	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---	BCD: #0000 ～ 9999 (BCD) “&”不能用 二进制： &0 ～ &65535 （十进制） #0000 ～ #FFFF （十六进制）
数据寄存器	---	DR0 ～ DR15
索引寄存器	---	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15	

描述

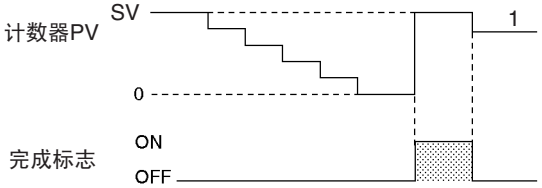
每次增量输入从 OFF 到 ON，计数器 PV 增 1，而每次减量输入从 OFF 到 ON，计数器 PV 减 1。 PV 在 0 ～ SV 之间变动。



在增量时， PV 从 SV 增加返回到 0 时，完成标志变 ON，一旦完成标志变为 ON， PV 从 0 增加到 1 时，完成标志又变回 OFF。



在减量时，PV 从 0 减小到 SV 时，完成标志变 ON，PV 从 SV 减小到 SV-1 时，完成标志又变回 OFF。



标志

名称	标记	操作
错误标志	ER	如果 N 通过索引寄存器间接寻址，但是索引寄存器中的地址不是计数器的 PV 的地址时为 ON。 如果是 BCD 模式而 S 不包含 BCD 数据时 ON。 其它情况时 OFF。

注意

CNT，CNTX(546)，CNTR(012)，CNTRX(548)，CNTW(814) 和 CNTWX(818) 指令共享计数器号。如果两个计数器使用相同的计数器号，但并不同时使用，当程序检查时，会产生一个重复错误，但计数器会正常操作。如果同时使用，使用相同计数器号的计数器不会正常操作。

如果增量输入减量输入同时从 OFF 到 ON，PV 不会改变。当复位输入 ON 后，PV 会复位到 0，两个计数输入都会被忽略。

仅当 PV 从 SV 增加到 0 或从 0 减小到 SV 时，完成标志变 ON；其它所有情况均为 OFF。

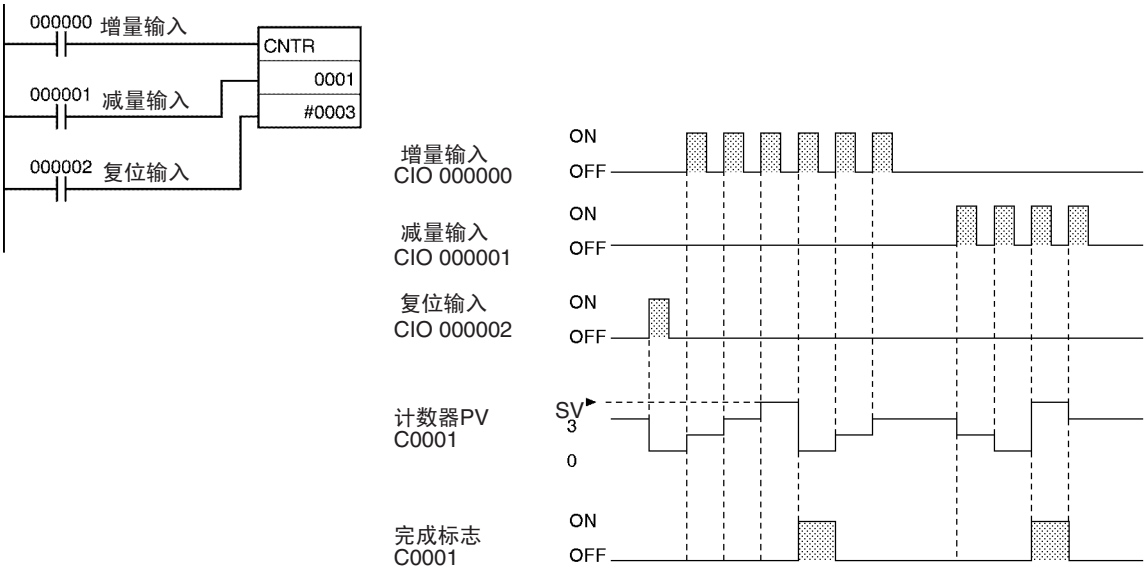
当用助记符输入 CNTR(012)/CNTRX(548) 指令，首先输入增量 (II)，然后是减量输入 (DI)，复位输入 (R)，最后是 CNT/CNTX(546) 指令。当用梯形图输入时，首先输入增量输入，然后是 CNT/CNTX(546) 指令，减量输入 (DI)，最后是复位输入 (R)。

例

CNTR(012)/CNTRX(548) 的基本操作

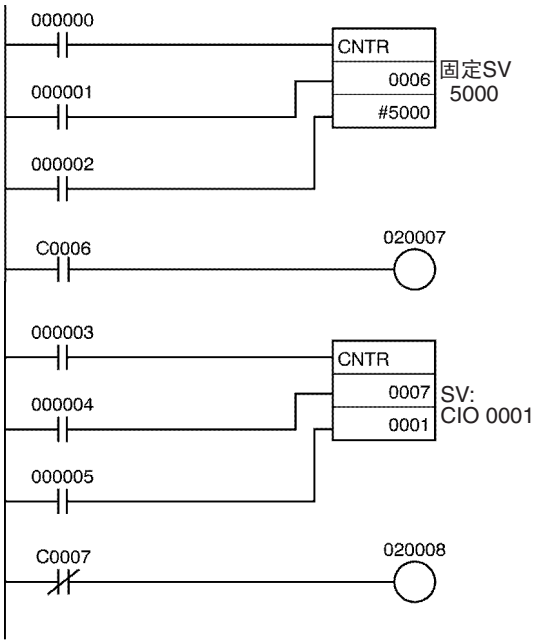
改变复位输入 (CIO 000002)ON 然后 OFF，计数器 PV 复位到 0。每次增量输入 (CIO 000000) 从 OFF 变到 ON， PV 增加 1，当 PV 从 SV(3) 加，会自动复位到 0，并且完成标志变 ON。

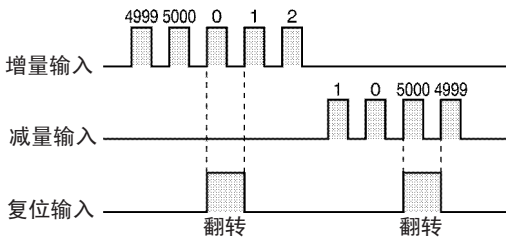
同样地，每次减量输入 (CIO 000001) 从 OFF 变到 ON， PV 减少 1，当 PV 从 0 开始减少， PV 会自动置为 SV(3)，并且完成标志变 ON。



在字中指定 SV

在下面例子中， CNTR(012)0007 的 SV 通过 CIO 0001 的内容确定。当 CIO 0001 的内容由外部开关控制，设置值可由开关手动改变。





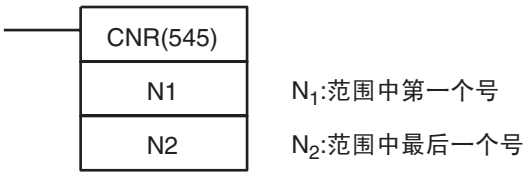
3-6-9 复位定时器 / 计数器：CNR(545)/CNRX(547)

用途

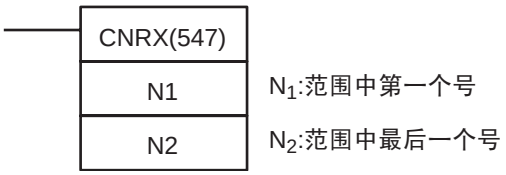
对指定定时器或计数器范围内的定时器或计数器复位。

梯形图符号

BCD



二进制



变化

变化	ON 条件时每次循环执行	CNR(545)/ CNRX(547)
	上升沿微分时执行一次	@CNR(545)/ CNRX(547)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

N_1: 范围中第一个号
N_1 必须是 T0000 ～ T4095 间的定时器号或 C0000 ～ C4095 间的计数器号。

N_2: 范围中最后一个号
N_2 必须是 T0000 ～ T4095 间的定时器号或 C0000 ～ C4095 间的计数器号。

注 N_1 和 N_2 必须在同一数据区，即， N_1 和 N_2 必须是定时器号或计数器号。

操作数规定

区域	N_1	N_2
CIO 区	---	---
工作区	---	---
保持位区	---	---
辅助位区	---	---
定时器区	C0000 ～ C4095	C0000 ～ C4095
计数器区	T0000 ～ T4095	T0000 ～ T4095
DM 区	---	---
无区号 EM 区	---	---
有区号 EM 区	---	---
二进制间接 DM/EM 地址	---	---
BCD 间接 DM/EM 地址	---	---

区域	N_1	N_2
常数	---	---
数据寄存器	---	---
索引寄存器	---	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15	

描述

CNR(545)/CNRX(547) 复位从 N_1 ～ N_2 所有定时器或计数器的复位标志，同时，PV 会全部设置到最大值（对 BCD 是 9999，对二进制是 FFFF）。（PV 会在下一次定时器或计数器指令执行时设置成 SV）。

由 CNR(545)/CNRX(547) 复位的定时器

下面的定时器如果定时器号在指定的范围内会被复位：TIM,TIMX(550),TIMH(015),TIMHX(551),TMHH(540),TMHHX(552),TTIM(087),TTIMX(555),TIMW(813),TIMWX(816),TMHW(815) 和 TMHWX(817)。当一个定时器被复位，它的完成标志变为 OFF，它的 PV 被设置成最大值 9999。

注 TIML(542),TIMLX(553),MTIM(543) 和 MTIMX(554) 定时器不会被 CNR(545)/CNRX(547) 复位，因为这些定时器不用定时器号。

由 CNR(545)/CNRX(547) 复位的计数器

下面的计数器如果计数器号在指定的范围内会被复位：CNT,CNTX(546),CNTR(012),CNTRX(548),CNTW(814) 和 CNTWX(818)。当一个计数器被复位，它的完成标志变为 OFF，它的 PV 被设置成最大值 9999。

标志

名称	标记	操作
错误标志	ER	如果 N_1 通过索引寄存器间接寻址，但是索引寄存器中的地址不是计数器的 PV 的地址时为 ON。 如果 N_2 通过索引寄存器间接寻址，但是索引寄存器中的地址不是计数器的 PV 的地址时为 ON。 如果 N_1 和 N_2 不在同一数据区时为 ON。 其它情况时 OFF。

注意

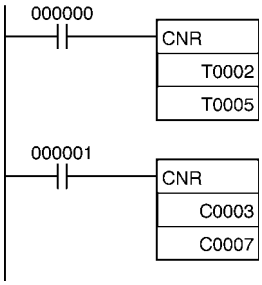
CNR(545)/CNRX(547) 不复位定时器 / 计数器指令本身，它复位分配给这些指令的 PV 和完成标志。在许多情况下， CNR(545)/CNRX(547) 的效果和直接复位指令不同。例如，TIM/TIMX(550) 指令被直接复位，它们的 PV 被设置为 SV，但是当定时器由 CNR(545)/CNRX(547) 复位时， PV 设置为最大值 9999。

当 N1 和 N2 被指定为 N1>N2，仅定时器 / 计数器的完成标志被复位。

例

在下面的例子中，当 CIO 000000 为 ON 时，定时器 T0002 ～ T0005 完成标志为 OFF，并且这些定时器 PV 设置为最大值 9999。

当 CIO 000001 为 ON 后，计数器 C0003 ～ C0007 完成标志为 OFF，并且这些计数器 PV 设置为最大值 9999。



3-6-10 定时器和计数器应用举例

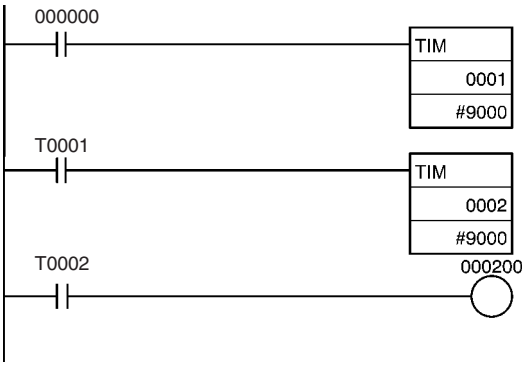
下面例子显示了定时器和计数器指令的各种应用，包括长时间定时器，二级计数器， ON/OFF 延时，单稳脉冲位和闪烁位。

例 1:
长时间定时器

下面程序举例中，显示了用标准 TIM 和 CNT 指令由三种方法产生长时间定时器。

二条 TIM 指令

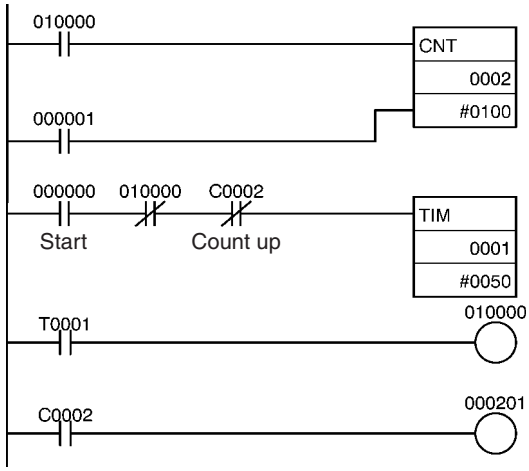
在这个例子中，二条 TIM 指令组合成一个 30 分钟的定时器。



地址	指令	操作数
000000	LD	000000
000001	TIM	0001
		#9000
000002	LD	T0001
000003	TIM	0002
		#9000
000004	LD	T0002
000005	OUT	000200

TIM 和 CNT 指令

在这个例子中，使用一条 TIM 指令和一条 CNT 指令组合成一个 500 秒定时器。TIM0001 每 5 秒产生一个脉冲， CNT0002 计数这些脉冲，这种组合的设置值为定时器间隔×计数器 SV。这样，定时器 SV 会是 5 秒× 100=500 秒。用这种组合，长时间定时器的PV实际上是计数器的PV，它在电源中断时也能保持。

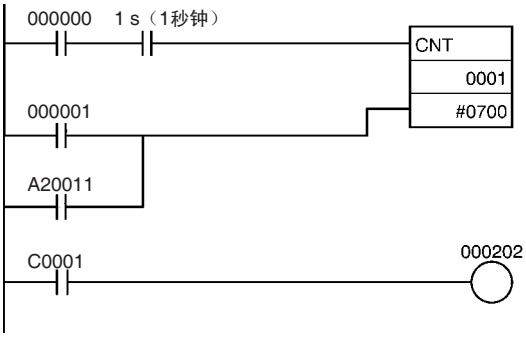


地址	指令	操作数
000000	LD	010000
000001	LD	000001
000002	CNT	0002
		#0100
000003	LD	000000
000004	AND NOT	010000
000005	AND NOT	C0002
000006	TIM	0001
		#0050
000007	LD	T0001
000008	OUT	010000
000009	LD	C0002
000010	OUT	000201

时钟脉冲与 CNT 指令

在此例中，CNT 指令对 1 秒时钟脉冲进行计数，从而产生一个 700 秒定时器。

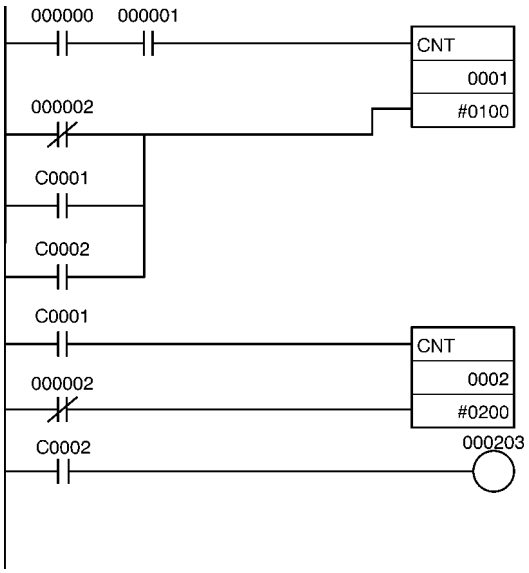
如果第一个循环标志 (A20011) 与计数器的复位输入 (CIO 000001) 进行 OR 运算，那么计数器的 PV（当前值）在程序执行而非重新对原先的 PV 进行计数时复位为 SV(0700) （设定值）。



地址	指令	操作数
000000	LD	000000
000001	AND	1 s
000002	LD	000001
000003	OR	A20011
000004	CNT	0001
		#0700
000005	LD	C0001
000006	OUT	000202

例 2: 二级计数器

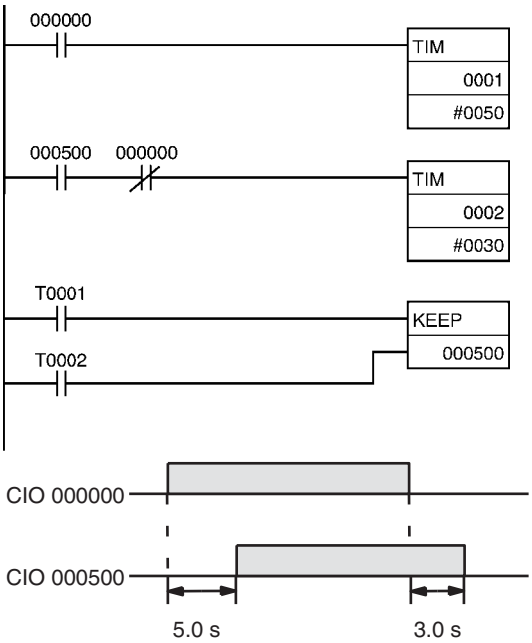
当 SV 需要高于 9999 时，可以像下面例子中所示的那样，把两个计数器进行组合。这种情况下，结合两个 CNT 指令组成一个 SV 为 20,000 的 BCD 计数器。



地址	指令	操作数
000000	LD	000000
000001	AND	000001
000002	LD NOT	000002
000003	OR	C0001
000004	OR	C0002
000005	CNT	0001
		#0100
000006	LD	C0001
000007	LD NO	000002
000008	CNT	0002
		#0200
000009	LD	C0002
000010	OUT	000203

例 3：ON/OFF 延时

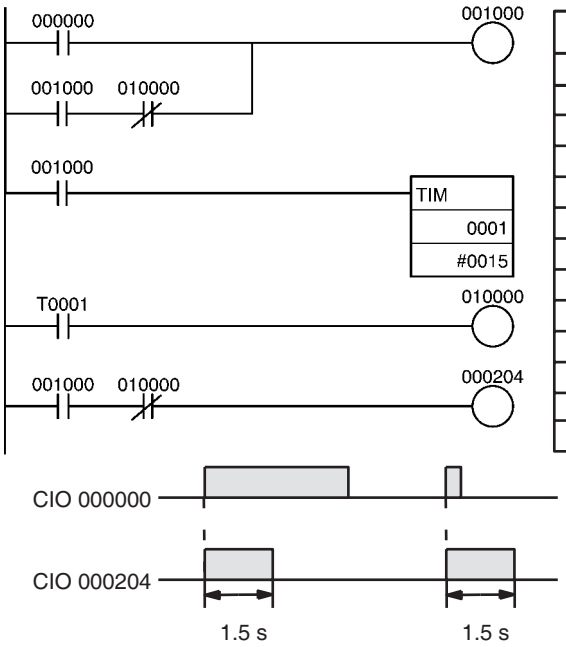
在此例中，两个 TIM 定时器结合 KEEP(011) 组成一个 ON 延时和 OFF 延时功能，在 CIO 000000 变 ON 5.0 秒后，CIO 000500 变为 ON，而在 CIO 000000 变 OFF 3.0 秒后， CIO 000500 变为 OFF。



地址	指令	操作数
000000	LD	000000
000001	TIM	0001
		#0050
000002	LD	000500
000003	AND NOT	000000
000004	TIM	0002
		#0030
000005	LD	T0001
000006	LD	T0002
000007	KEEP(011)	000500

例 4：单稳脉冲位

一个 TIM 定时器能够结合 OUT 或 OUT NOT 去控制特定位 ON 或 OFF 状态的持续时间。在这个例子中，CIO 000000 变 ON 后 CIO 000204 将为 ON，并保持 1.5 秒 （T0001 的 SV）。



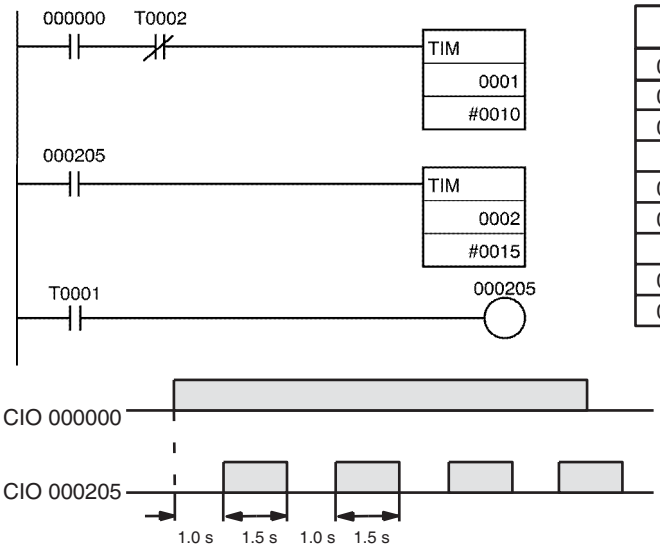
地址	指令	操作数
000000	LD	000000
000001	LD	001000
000002	AND NOT	010000
000003	OR	000000
000004	OUT	001000
000005	LD	001000
000006	TIM	0001
		#0015
000007	LD	T0001
000008	OUT	010000
000009	LD	001000
000010	AND NOT	010000
000011	OUT	000204

例 4：闪烁位

下面程序举例演示了产生闪烁位的两种方法。第二个例子就像模仿一个时钟脉冲。

两条 TIM 指令

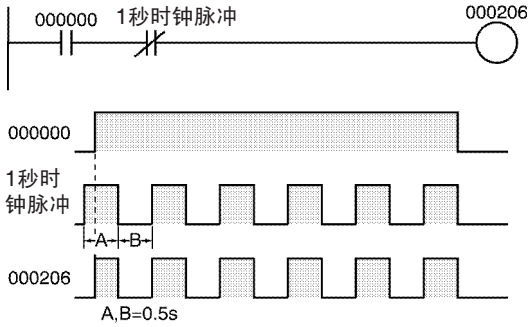
结合两个 TIM 定时器可在执行条件为 ON 时，以规则的间隔使某一位为 ON 和 OFF。在此例中，只要 CIO 000000 为 ON，CIO 000205 会 OFF 1.0 秒 , 然后 ON 1.5 秒。



地址	指令	操作数
000000	LD	000000
000001	AND	T0002
000002	TIM	0001
		#0010
000003	LD	000205
000004	TIM	0002
		#0015
000005	LD	T0001
000006	OUT	000205

时钟脉冲

所要求的执行条件可与一个时钟脉冲结合，以模仿时钟脉冲 （0.1s， 0.2s 或 1.0s）。



地址	指令	操作数
000000	LD	000000
000001	AND	1s
000002	OUT	000206

3-6-11 定时器 / 计数器号间接寻址

定时器和计数器可使用索引寄存器间接寻址。当使用索引寄存器来间接寻址时，使用 MOVRW （561）指令 （传送定时器 / 计数器 PV 到寄存器中去）去把所需的定时器 / 计数器 PV 的 PLC 存储地址保存到指定的索引寄存器中。

下列定时器和计数器可使用索引寄存器间接寻址：TIM， TIMX(550)， TIMH(015)， TIMHX(551)， TTIM(087)， TTIMX(555)， TMHH(540)， TMHHX(552)， TIMW(813)， TIMWX(816)， TMHW(815)， TMHWX(817)， CNT， CNTX(546)， CNTR(012)， CNTRX(548)， CNTW(814)，和 CNTWX(818)。（这些定时器和计数器使用定时器号和计数器号）。

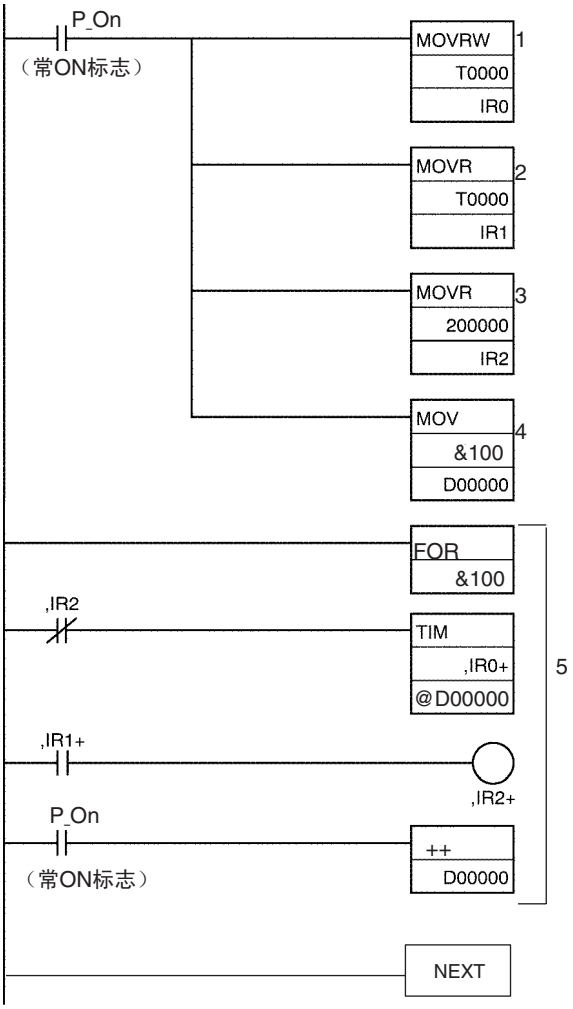
如果指定的索引寄存器中的 PLC 存储地址不是定时器或计数器 PV 的地址时，定时器或计数器指令不会被执行。

使用索引寄存器间接寻址定时器和计数器能减少程序量并增加灵活性。例如，可以编制一个公共的子程序。

例

下面例子演示了一个程序段使用间接寻址定义并启动 100 个定时器，其 SV 包含在 D00100 ～ D00199 中。IR0 包含了定时器 PV 的 PLC 存储地址，IR1 包含了定时器完成标志的 PLC 存储地址。

DM 地址	内容	功能
D00100	0010	T0000 的 SV
D00101	0100	T0001 的 SV
D00102	0050	T0002 的 SV
.	.	.
.	.	.
.	.	.
D00199	0999	T0099 的 SV



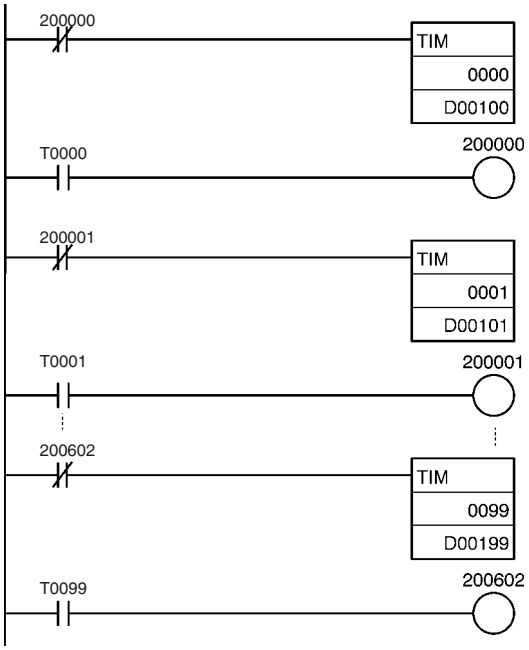
1,2,3... 1. MOVRW(561) 传送定时器 T0000 的 PV 的 PLC 存储地址到 IR0 中，然后 IR0 就可使用在定时器号的地方。

2. MOVR(560) 传送定时器 T0000 完成标志的 PLC 存储地址到 IR1 中。
3. MOVR(560) 传送 CIO 200000 的 PLC 存储地址到 IR2 中。
4. MOV(021) 传送 &0100 到 D00000 中作为定时器 SV 的间接寻址。
5. 在这个循环执行 100 次的过程中，每次 IR0,IR1, 和 D00000 的内容都增加 1，并启动定时器 T0000 ～ T0099。

在上面的程序循环中，有 4 个输入参数在这个公共子程序中使用，以启动所有 100 个定时器。

IR0 定时器 PV 的 PLC 存储地址
IR1 定时器完成标志的 PLC 存储地址
IR2 定时器执行条件的 PLC 存储地址
D00000 包含定时器 SV 的字的 DM 地址

上面的子程序相当于下面的 400 条指令。



地址	指令	操作数
000000	LD NOT	200000
000001	TIM	0000
		D00100
000002	LD	T0000
000003	OUT	200000
000004	LD NOT	200001
000005	TIM	0001
		D00101
000006	LD	T0001
000007	OUT	200001
000008	LD NOT	200002
000009	TIM	0002
		D00102
000010	LD	T0002
000011	OUT	200002
≈	≈	≈
000396	LD NOT	200602
000397	TIM	0099
		D00199
000398	LD	T0000
000399	OUT	200602

3-7 比较指令

这一节描述用于比较各种长度数据以及用各种方法进行比较的指令。

指令	助记符	功能代码	页数
输入比较指令	LD, AND, OR =, <>, <, <=, >, >=, L, S	300 ～ 328	246
比较	CMP	020	252
双字比较	CMPL	060	255
带符号二进制比较	CPS	114	257
双字带符号二进制比较	CPSL	115	260
多个比较	MCMP	019	263
表比较	TCMP	085	266
块比较	BCMP	068	268
扩展块比较	BCMP2	502	270

3-7-1 输入比较指令 (300 ～ 328)

用途

输入比较指令用于比较两个值（常数和 / 或指定字的内容），并在比较条件为真时产生一个 ON 执行条件，输入比较指令可用来比较单字或双字带符号或无符号数据。

注 请参阅 *3-15-21 单精度浮点数比较指令*和 *3-16-21 双精度浮点数比较指令*。

梯形图符号

变化

变化	每次循环比较为真时 ON	输入比较指令
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

单字数据指令的操作数规定

区域	S_1	S_2
CIO 区	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W511	
保持位区	H000 ～ H511	
辅助位区	A000 ～ A959	
定时器区	T0000 ～ T4095	
计数器区	C0000 ～ C4095	
DM 区	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	

区域	S_1	S_2
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_ 32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#0000 ～ #FFFF （二进制）	
数据寄存器	DR0 ～ DR15	
索引寄存器	---	
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

双字数据指令的操作数规定

区域	S_1	S_2
CIO 区	CIO 0000 ～ CIO 6142	
工作区	W000 ～ W510	
保持位区	H000 ～ H510	
辅助位区	A000 ～ A958	
定时器区	T0000 ～ T4094	
计数器区	C0000 ～ C4094	
DM 区	D00000 ～ D32766	
无区号 EM 区	E00000 ～ E32766	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#00000000 ～ #FFFFFFFF （二进制）	
数据寄存器	---	

区域	S_1	S_2
索引寄存器	IR0 ～ IR15 （仅适用于无符号数据）	
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

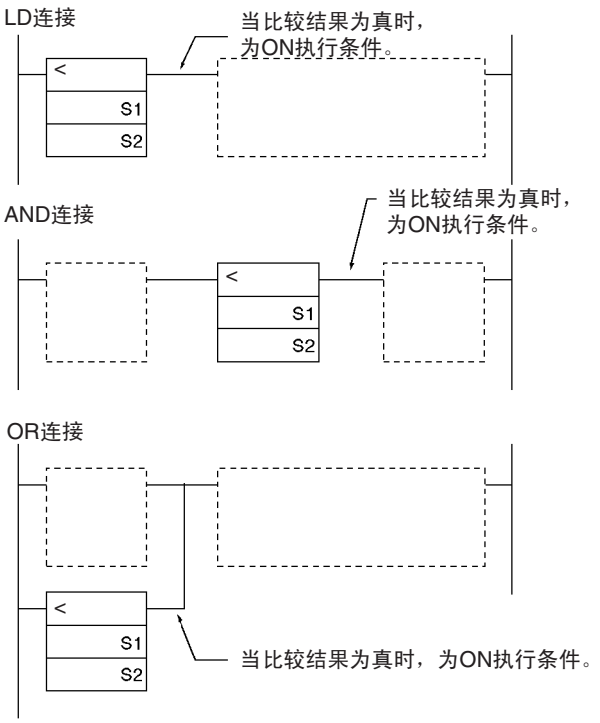
描述

输入比较指令把 S_1 和 S_2 当作带符号或不带符号值进行比较，并在比较条件为真时产生一个 ON 执行条件。与 CMP(020) 和 CMPL(060) 指令不同，输入比较指令将直接反映为执行条件的结果，因此无需通过算术标志访问比较结果，这样程序将更加简捷。

指令输入

输入比较指令就像 LD， AND，和 OR 指令控制随后的指令执行一样对待。

输入类型	操作
LD	这条指令可直接连到左边的母线。
AND	这条指令不能直接连到左边的母线。
OR	这条指令可直接连到左边的母线。



选项

输入比较指令能比较带符号或不带符号数据，并且能比较单字或双字数值。如果未指定选项，则作为单字无符号数据比较。指令有三种输入方式和两个选项，共有 72 个不同输入比较指令。

运算符	选项 （数据格式）	选项 （数据长度）
= （等于）	None: 不带符号数据	None: 单字数据
< > （不等于）	S: 带符号数据	L: 双字数据
< （小于）		
<= （小于等于）		
> （大于）		
>= （大于等于）		

无符号输入比较指令 （即指令无 S 选项）能处理无符号二进制或 BCD 数据，带符号输入比较指令 （即指令有 S 选项）可处理带符号二进制数据。

输入比较指令总结

下表列举了 72 种输入比较指令的函数代码，助记符，名称和功能。（对于单字比较：C1＝S_1，C2＝S_2；对于双字比较：C1＝S_1＋1，S_1，C2＝S_2＋1，S_2）

代码	助记符	名称	功能
300	LD=	装载等于	C1 = C2 时为真
	AND=	与等于	
	OR=	或等于	
301	LD=L	装载双字等于	
	AND=L	与双字等于	
	OR=L	或双字等于	
302	LD=S	装载带符号等于	
	AND=S	与带符号等于	
	OR=S	或带符号等于	
303	LD=SL	装载双字带符号等于	
	AND=SL	与双字带符号等于	
	OR=SL	或双字带符号等于	
305	LD<>	装载不等于	C1 ≠ C2 时为真
	AND<>	与不等于	
	OR<>	或不等于	
306	LD<>L	装载双字不等于	
	AND<>L	与双字不等于	
	OR<>L	或双字不等于	
307	LD<>S	装载带符号不等于	
	AND<>S	与带符号不等于	
	OR<>S	或带符号不等于	
308	LD<>SL	装载双字带符号不等于	
	AND<>SL	与双字带符号不等于	
	OR<>SL	或双字带符号不等于	
310	LD<	装载小于	C1 < C2 时为真
	AND<	与小于	
	OR<	或小于	
311	LD<L	装载双字小于	
	AND<L	与双字小于	
	OR<L	或双字小于	
312	LD<S	装载带符号小于	
	AND<S	与带符号小于	
	OR<S	或带符号小于	
313	LD<SL	装载双字带符号小于	
	AND<SL	与双字带符号小于	
	OR<SL	或双字带符号小于	

代码	助记符	名称	功能
315	LD<=	装载小于等于	C1 ≤ C2 时为真
	AND<=	与小于等于	
	OR<=	或小于等于	
316	LD<=L	装载双字小于等于	
	AND<=L	与双字小于等于	
	OR<=L	或双字小于等于	
317	LD<=S	装载带符号小于等于	
	AND<=S	与带符号小于等于	
	OR<=S	或带符号小于等于	
318	LD<=SL	装载双字带符号小于等于	C1 ≤ C2 时为真
	AND<=SL	与双字带符号小于等于	
	OR<=SL	或双字带符号小于等于	
320	LD>	装载大于	C1 > C2 时为真
	AND>	与大于	
	OR>	或大于	
321	LD>L	装载双字大于	
	AND>L	与双字大于	
	OR>L	或双字大于	
322	LD>S	装载带符号大于	
	AND>S	与带符号大于	
	OR>S	或带符号大于	
323	LD>SL	装载双字带符号大于	
	AND>SL	与双字带符号大于	
	OR>SL	或双字带符号大于	
325	LD>=	装载大于等于	C1 ≥ C2 时为真
	AND>=	与大于等于	
	OR>=	或大于等于	
326	LD>=L	装载双字大于等于	
	AND>=L	与双字大于等于	
	OR>=L	或双字大于等于	
327	LD>=S	装载带符号大于等于	
	AND>=S	与带符号大于等于	
	OR>=S	或带符号大于等于	
328	LD>=SL	装载双字带符号大于等于	
	AND>=SL	与双字带符号大于等于	
	OR>=SL	或双字带符号大于等于	

标志

名称	标记	操作
错误标志	ER	OFF 或不变 （见注）
大于标志	>	单字数据 S_1>S_2 时 ON。 双字数据 S_1+1， S_1>S_2+1， S_2 时 ON。 其它情况下 OFF。
大于等于标志	> =	单字数据 S_1>=S_2 时 ON。 双字数据 S_1+1， S_1>=S_2+1， S_2 时 ON。 其它情况下 OFF。
等于标志	=	单字数据 S_1=S_2 时 ON。 双字数据 S_1+1， S_1=S_2+1， S_2 时 ON。 其它情况下 OFF。

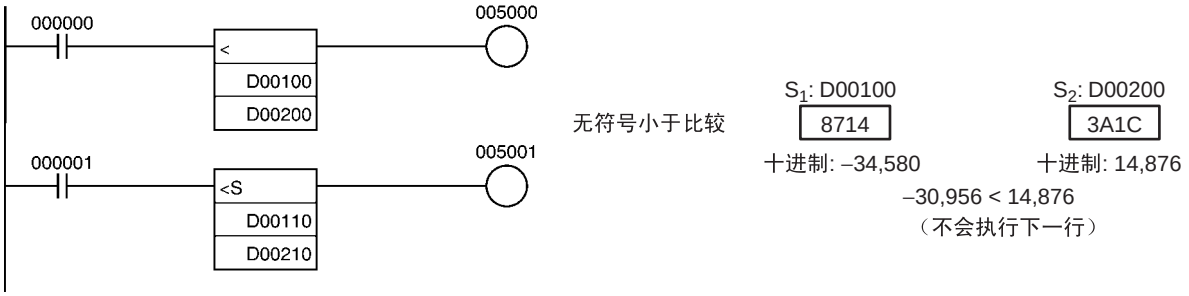
名称	标记	操作
不等于标志	=	单字数据 $S_1 \neq S_2$ 时 ON。 双字数据 S_1+1， $S_1 \neq S_2+1$， S_2 时 ON。 其它情况下 OFF。
小于标志	<	单字数据 $S_1<S_2$ 时 ON。 双字数据 S_1+1， $S_1<S_2+1$， S_2 时 ON。 其它情况下 OFF。
小于等于标志	< =	单字数据 $S_1<=S_2$ 时 ON。 双字数据 S_1+1， $S_1<=S_2+1$， S_2 时 ON。 其它情况下 OFF。
负数标志	N	OFF 或不变 （见注）

注 在 CS1 和 CJ1 CPU 单元中，这些标志均为 OFF。
在 CS1-H， CJ1-H， CJ1M，以及 CS1D CPU 单元中，这些标志保持不变。

注意 输入比较指令不能作为右侧指令使用，也就是说在它们与母线之间必须要有其它指令。

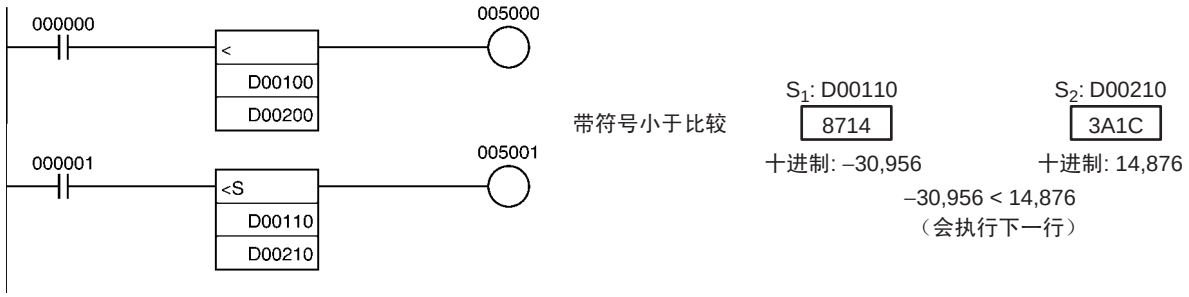
例 **与小于：AND ＜ (310)**

在下面例子中，当 CIO 000000 为 ON 时，D00100 和 D00200 中的内容作为无符号二进制数据比较。如果 D00100 中的内容小于 D00200 中的内容， CIO 005000 变 ON，执行进行到下一行；如果 D00100 中的内容不小于 D00200 中的内容，此行余下的指令被忽略，执行移到下一行指令。



与带符号小于：AND ＜ S(312)

在下面例子中，当 CIO 000001 为 ON 时，D00110 和 D00210 中的内容作为带符号二进制数据比较。如果 D00110 中的内容小于 D00210 中的内容， CIO 005001 变 ON，执行进行到下一行；如果 D00110 中的内容不小于 D00210 中的内容，此行余下的指令被忽略，执行移到下一指令行。



3-7-2 比较：CMP(020)

用途

比较两个无符号二进制值 （常数和 / 或指定字的内容），并输出结果到辅助区的算术标志中。

梯形图符号

变化

变化	ON 条件时每次循环执行	CMP(020)
	上升沿微分时执行一次	不支持
	下降沿微分时执行一次	不支持
立即刷新功能 （见注）		!CMP(020)

注 CS1D CPU 单元不支持立即刷新功能。

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

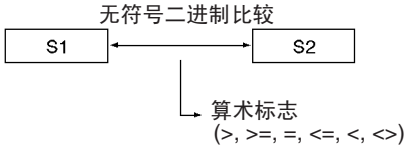
操作数规定

区域	S_1	S_2
CIO 区	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W511	
保持位区	H000 ～ H511	
辅助位区	A000 ～ A959	
定时器区	T0000 ～ T4095	
计数器区	C0000 ～ C4095	
DM 区	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#0000 ～ #FFFF （二进制）	
数据寄存器	DR0 ～ DR15	

区域	S_1	S_2
索引寄存器	---	
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

描述

CMP(020) 比较 S_1 和 S_2 中的无符号二进制数据，并输出结果到辅助区中的算术标志中 （大于，大于等于，等于，小于等于，小于和不等于标志）。



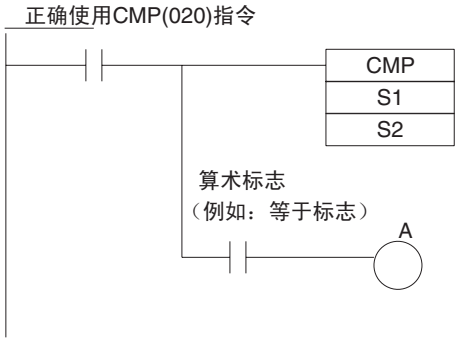
条件标志状态

下表显示了在执行 CMP(020) 指令后算术标志的状态。（状态 “…”表示标志可以为 ON 或 OFF）。

CMP(020) 结果	标志状态					
	>	> =	=	< =	<	< >
$S_1 > S_2$	ON	ON	OFF	OFF	OFF	ON
$S_1 = S_2$	OFF	ON	ON	ON	OFF	OFF
$S_1 < S_2$	OFF	OFF	OFF	ON	ON	ON

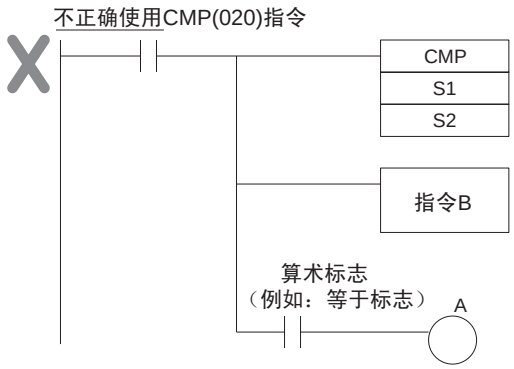
在程序中使用 CMP(020) 结果

当执行 CMP(020) 指令，其结果反映在算术标志中。用与控制 CMP(020) 指令相同的输入条件分支来控制所希望的输出或右侧指令，如下图所示。在这种情况下，当 $S_1 = S_2$，等于标志和输出 A 将变 ON。



在程序中使用 CMP(020) 结果

不要在 CMP(020) 指令和算术标志控制指令之间编写其它指令，因为其它指令可能会改变算术标志的状态。在这种情况下，指令 B 的结果可能会改变指令 CMP(020) 的结果。



立即刷新变化 (!CMP(020)) 指令可用于 S_1 和 / 或 S_2 中指定分配给外部输入的字。在执行 !CMP(020) 指令后，将执行在 S_1 和 / 或 S_2 中指定的外部输入字的输入刷新，并且比较刷新过的值。（不可对分配给组 2 高密度 I/O 单元或安装在从属机架上的单元的输入执行立即刷新操作）

标志

名称	标记	操作
错误标志	ER	OFF 或不变 （见注）
大于标志	>	$S_1>S_2$ 时 ON 其它情况下 OFF
大于等于标志	> =	$S_1>=S_2$ 时 ON 其它情况下 OFF
等于标志	=	$S_1=S_2$ 时 ON 其它情况下 OFF
不等于标志	=	$S_1<>S_2$ 时 ON 其它情况下 OFF
小于标志	<	$S_1<S_2$ 时 ON 其它情况下 OFF
小于等于标志	< =	$S_1<=S_2$ 时 ON 其它情况下 OFF
负数标志	N	OFF 或不变 （见注）

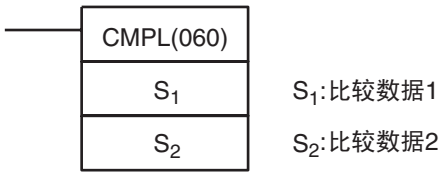
注 在 CS1 和 CJ1 CPU 单元中，这些标志均为 OFF。
在 CS1- H， CJ1- H， CJ1M，以及 CS1D CPU 单元中，这些标志保持不变。

注意 不要在 CMP(020) 和访问 CMP(020) 结果的输入条件之间编写其它指令，因为其它指令可能会改变算术标志的状态。

3-7-3 双字比较：CMPL(060)

用途 比较两个双字无符号二进制值 （常数和 / 或指定字的内容），并输出结果到辅助区的算术标志中。

梯形图符号



变化

变化	ON 条件时每次循环执行	CMPL(060)
	上升沿微分时执行一次	不支持
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

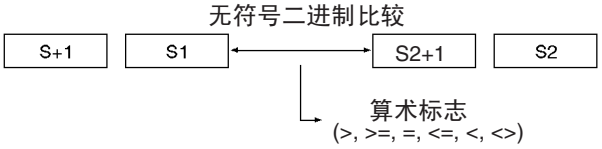
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	S_1	S_2
CIO 区	CIO 0000 ～ CIO 6142	
工作区	W000 ～ W510	
保持位区	H000 ～ H510	
辅助位区	A000 ～ A958	
定时器区	T0000 ～ T4094	
计数器区	C0000 ～ C4094	
DM 区	D00000 ～ D32766	
无区号 EM 区	E00000 ～ E32766	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#00000000 ～ #FFFFFFFF （二进制）	
数据寄存器	---	
索引寄存器	IR0 ～ IR15	
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

描述

CMPL(060) 比较 S_1 + 1，S_1 和 S_2 + 1，S_2 中的双字无符号二进制数据，并输出结果到辅助区中的算术标志中（大于，大于等于，等于，小于等于，小于和不等于标志）。



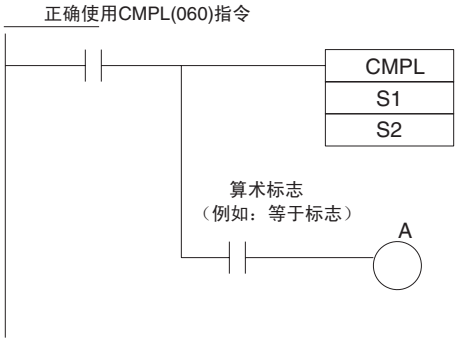
算术标志状态

下表显示了执行 CMPL(060) 指令后算术标志的状态 （状态 “…”表示标志可以为 ON 或 OFF）。

CMPL(060) 结果	标志					
	>	> =	=	< =	<	< >
S_1 +1, S_1 > S_2+1, S_2	ON	ON	OFF	OFF	OFF	ON
S_1+1, S_1 = S_2+1, S_2	OFF	ON	ON	ON	OFF	OFF
S_1+1, S_1 < S_2+1, S_2	OFF	OFF	OFF	ON	ON	ON

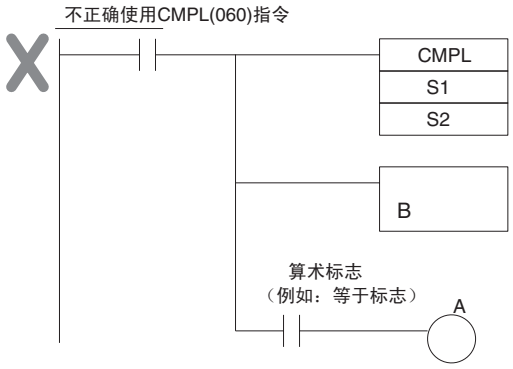
在程序中使用 CMPL(060) 结果

当执行 CMPL(060) 指令，其结果反映在算术标志中。用与控制 CMPL(060) 指令相同的输入条件分支来控制所希望的输出或右侧指令，如下图所示。在这种情况下，当 S_1+1,S_1 ＝ S_2+1,S_2，等于标志和输出 A 将变 ON。



在程序中使用 CMPL(060) 结果

不要在CMPL(060) 指令和算术标志控制指令之间编写其它指令，因为其它指令可能会改变算术标志的状态。在这种情况下，指令 B 的结果可能会改变 CMPL(060) 的结果。



标志

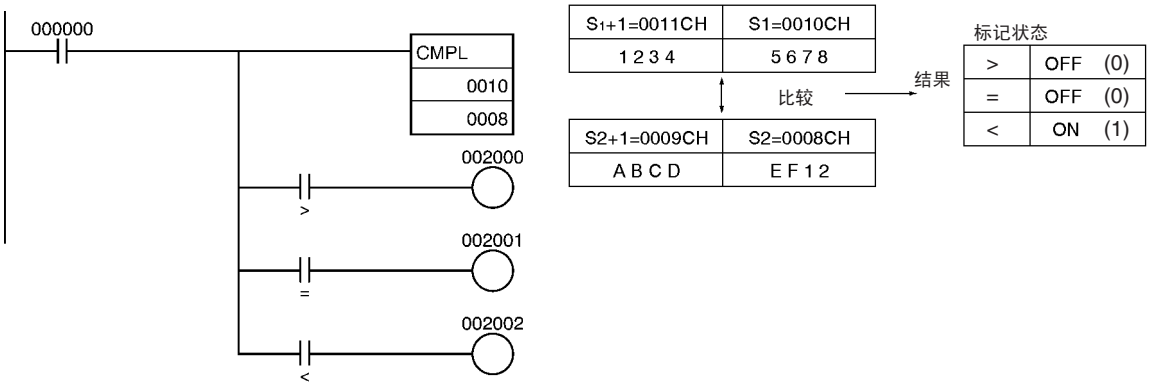
名称	标记	操作
错误标志	ER	OFF 或不变 （见注）
大于标志	>	S_1 ＋ 1， S_1>S_2 ＋ 1， S_2 时 ON 其它情况下 OFF。
大于等于标志	> =	S_1 ＋ 1， S_1>=S_2 ＋ 1， S_2 时 ON 其它情况下 OFF。
等于标志	=	S_1 ＋ 1， S_1=S_2 ＋ 1， S_2 时 ON 其它情况下 OFF。
不等于标志	<>	S_1 ＋ 1， S_1<>S_2 ＋ 1， S_2 时 ON 其它情况下 OFF。

名称	标记	操作
小于标志	<	S_1 + 1， S_1<S_2 + 1， S_2 时 ON 其它情况下 OFF。
小于等于标志	< =	S_1 + 1， S_1<=S_2 + 1， S_2 时 ON 其它情况下 OFF。
负数标志	N	OFF 或不变 （见注）

注 在 CS1 和 CJ1 CPU 单元中，这些标志均为 OFF。
在 CS1-H， CJ1-H， CJ1M，以及 CS1D CPU 单元中，这些标志保持不变。

注意 不要在 CMPL(060) 和访问 CMPL(060) 结果的输入条件之间编写其它指令，因为其它指令可能会改变算术标志的状态。

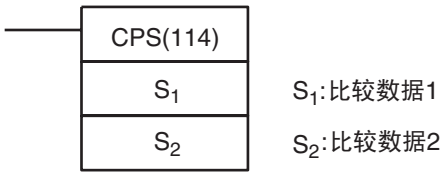
例 在下面例子中，当 CIO 000000 为 ON 时， CIO 0011 和 CIO 0010 中的 8 位无符号二进制数与 CIO 0009 和 CIO 0008 中的 8 位无符号二进制数进行比较，并且把结果输到算术标志中。记录在大于，等于，小于标志中的结果立即存入 CIO 000200 （大于）， CIO 000201 （等于）和 CIO 000202 （小于）中。



3-7-4 带符号二进制比较：CPS(114)

用途 比较两个带符号二进制值 （常数和 / 或指定字的内容），并输出结果到辅助区的算术标志中。

梯形图符号



变化

变化		
	ON 条件时每次循环执行	CPS(114)
	上升沿微分时执行一次	不支持
	下降沿微分时执行一次	不支持
立即刷新功能		!CPS(114)

注 CS1D CPU 单元不支持立即刷新功能。

适用程序区

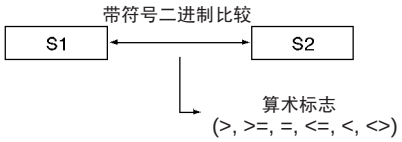
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	S_1	S_2
CIO 区	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W511	
保持位区	H000 ～ H511	
辅助位区	A000 ～ A959	
定时器区	T0000 ～ T4095	
计数器区	C0000 ～ C4095	
DM 区	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#0000 ～ #FFFF （二进制）	
数据寄存器	DR0 ～ DR15	
索引寄存器	---	
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

描述

CPS(114) 比较 S_1 和 S_2 中的带符号二进制数据，并输出结果到辅助区中的算术标志中 （大于，大于等于，等于，小于等于，小于和不等于标志）。



注 CPS(114) 把 S_1 和 S_2 中的数据看成为范围从 8000 ～ 7FFF（十进制为－ 32768 ～ 32767）的带符号二进制数。

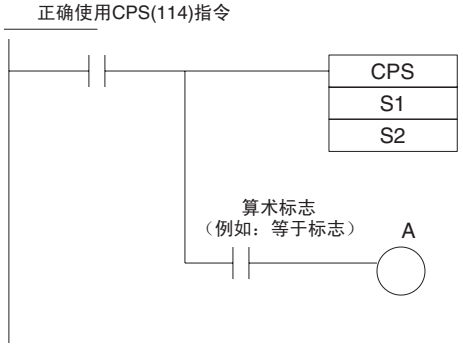
算术标志状态

下表显示了执行 CPS(114) 指令后算术标志的状态 （状态 “…” 表示标志可以为 ON 或 OFF）。

CPS(114) 结果	标志状态					
	>	> =	=	< =	<	< >
$S_1 > S_2$	ON	ON	OFF	OFF	OFF	ON
$S_1 = S_2$	OFF	ON	ON	ON	OFF	OFF
$S_1 < S_2$	OFF	OFF	OFF	ON	ON	ON

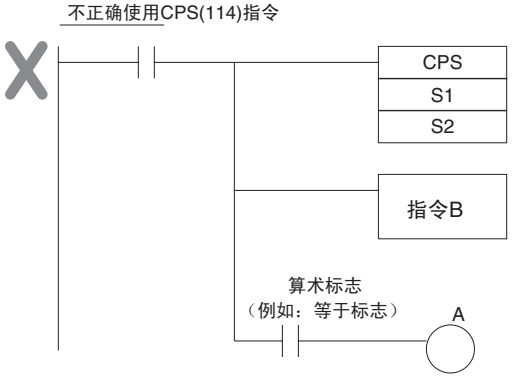
在程序中使用 CPS(114) 结果

当执行 CPS(114) 指令，其结果反映在算术标志中。用与控制 CPS(114) 指令相同的输入条件分支来控制所希望的输出或右侧指令，如下图所示。在这种情况下，当 $S_1 = S_2$，等于标志和输出 A 将变 ON。



在程序中使用 CPS(114) 结果

不要在 CPS(114) 指令和算术标志控制指令之间编写其它指令，因为其它指令可能会改变算术标志的状态。在这种情况下，指令 B 的结果可能会改变 CPS(114) 的结果。



立即刷新变化 (!CPS(114)) 指令可用于 S_1 和 / 或 S_2 中指定分配给外部输入的字。在执行 !CPS(114) 指令后，将执行在 S_1 和 / 或 S_2 中指定的外部输入字的输入刷新，并且比较刷新过的值。（不可对分配给组 2 高密度 I/O 单元或安装在从属机架上的单元的输入执行立即刷新操作）。

标志

名称	标记	操作
错误标志	ER	OFF 或不变 （见注）
大于标志	>	$S_1 > S_2$ 时 ON 其它情况下 OFF。
大于等于标志	> =	$S_1 \geq S_2$ 时 ON 其它情况下 OFF。
等于标志	=	$S_1 = S_2$ 时 ON 其它情况下 OFF。
不等于标志	<>	$S_1 \neq S_2$ 时 ON 其它情况下 OFF。
小于标志	<	$S_1 < S_2$ 时 ON 其它情况下 OFF。
小于等于标志	< =	$S_1 \leq S_2$ 时 ON 其它情况下 OFF。
负数标志	N	OFF 或不变 （见注）

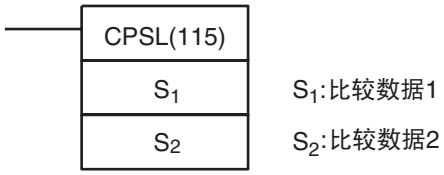
注 在 CS1 和 CJ1 CPU 单元中，这些标志均为 OFF。
在 CS1- H， CJ1- H， CJ1M，以及 CS1D CPU 单元中，这些标志保持不变。

注意 不要在 CPS(114) 和访问 CPS(114) 结果的输入条件之间编写其它指令，因为其它指令可能会改变算术标志的状态。

3-7-5 双字带符号二进制比较：CPSL(115)

用途 比较两个双字带符号二进制值 （常数和 / 或指定字的内容），并输出结果到辅助区的算术标志中。

梯形图符号



变化

变化	ON 条件时每次循环执行	CPSL(115)
	上升沿微分时执行一次	不支持
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

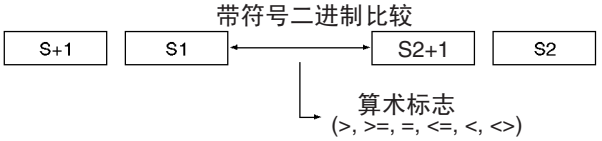
操作数规定

区域	S_1	S_2
CIO 区	CIO 0000 ～ CIO 6142	
工作区	W000 ～ W510	
保持位区	H000 ～ H510	
辅助位区	A000 ～ A958	
定时器区	T0000 ～ T4094	
计数器区	C0000 ～ C4094	
DM 区	D00000 ～ D32766	
无区号 EM 区	E00000 ～ E32766	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#00000000 ～ #FFFFFFFF （二进制）	
数据寄存器	---	

区域	S_1	S_2
索引寄存器	---	
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

描述

CPSL(115) 比较 S_1+1， S_1 和 S_2+1， S_2 中的双字带符号二进制数据，并输出结果到辅助区中的算术标志中 （大于，大于等于，等于，小于等于，小于和不等于标志）。



注 CPSL(115) 把 S_1 和 S_2 中的数据看成为范围从 80000000 ～ 7FFFFFFF（十进制为－ 2,147,483,648 ～ 2,147,483,647）的双字带符号二进制数。

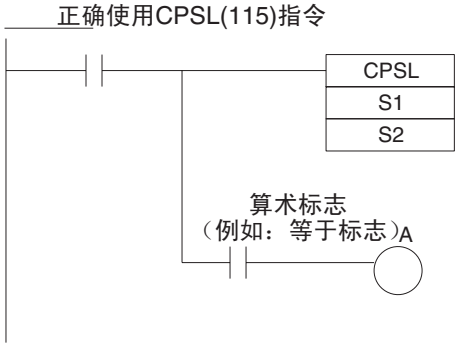
算术标志状态

下表显示了执行 CPSL(115) 指令后算术标志的状态 （状态 “…”表示标志可以为 ON 或 OFF）。

CPSL(115) 结果	标志状态					
	>	> =	=	< =	<	< >
S_1 +1, S_1 > S_2+1, S_2	ON	ON	OFF	OFF	OFF	ON
S_1+1, S_1 = S_2+1, S_2	OFF	ON	ON	ON	OFF	OFF
S_1+1, S_1 < S_2+1, S_2	OFF	OFF	OFF	ON	ON	ON

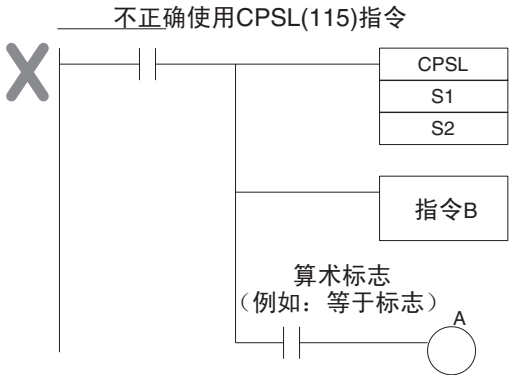
在程序中使用 CPSL(115) 结果

当执行 CPSL(115) 指令，其结果反映在算术标志中。用与控制 CPSL(115) 指令相同的输入条件分支来控制所希望的输出或右侧指令，如下图所示。在这里，当 S_1+1， S_1 ＝ S_2+1， S_2 等于标志和输出 A 将变 ON。



在程序中使用 CPSL(115) 结果

不要在 CPSL(115) 指令和算术标志控制指令之间编写其它指令，因为其它指令可能会改变算术标志的状态。在这种情况下，指令 B 的结果可能会改变 CPSL(115) 的结果。



标志

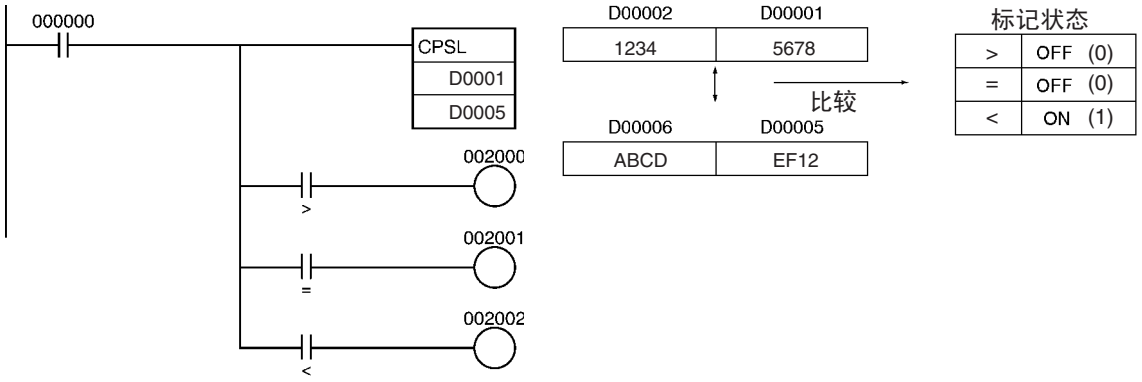
名称	标记	操作
错误标志	ER	OFF 或不变 （见注）
大于标志	>	S_1 +1， $S_1 > S_2$+1， S_2 时 ON 其它情况下 OFF。
大于等于标志	> =	S_1 +1， $S_1 \geq S_2$+1， S_2 时 ON 其它情况下 OFF。
等于标志	=	S_1 +1， $S_1 = S_2$+1， S_2 时 ON 其它情况下 OFF。
不等于标志	=	S_1 +1， $S_1 \neq S_2$+1， S_2 时 ON 其它情况下 OFF。
小于标志	<	S_1 +1， $S_1 < S_2$+1， S_2 时 ON 其它情况下 OFF。
小于等于标志	< =	S_1 +1， $S_1 \leq S_2$+1， S_2 时 ON 其它情况下 OFF。
负数标志	N	OFF 或不变 （见注）

注 在 CS1 和 CJ1 CPU 单元中，这些标志均为 OFF。
在 CS1-H， CJ1-H， CJ1M，以及 CS1D CPU 单元中，这些标志保持不变。

注意 不要在 CPSL(115) 和访问 CPSL(115) 结果的输入条件之间编写其它指令，因为其它指令可能会改变算术标志的状态。

例 在下面例子中，当 CIO 000000 为 ON 时， D00002 和 D00001 中的 8 位带符号二进制数与 D00006 和 D00005 中的 8 位带符号二进制数进行比较，并且把结果输到算术标志中。

- 如果 D00002 和 D00001 中的内容大于 D00006 和 D00005 中的内容，则大于标志变 ON，使得 CIO 002000 变 ON。
- 如果 D00002 和 D00001 中的内容等于 D00006 和 D00005 中的内容，则等于标志变 ON，使得 CIO 002001 变 ON。
- 如果 D00002 和 D00001 中的内容小于 D00006 和 D00005 中的内容，则小于标志变 ON，使得 CIO 002002 变 ON。

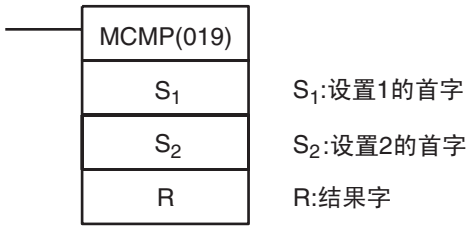


3-7-6 多个比较：MCMP （019）

用途

16 个连续字与另外 16 个连续字相比较，并使结果字中相应于两字内容不相等的位变 ON。

梯形图符号



变化

变化	ON 条件时每次循环执行	MCMP(019)
	上升沿微分时执行一次	@MCMP(019)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

S_1: 设置 1 的首字

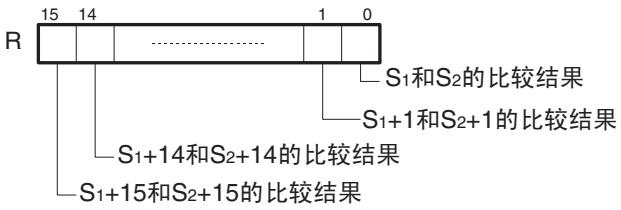
指定第一个 16 连续字的首字， S_1 和 S_1 ＋ 15 必须在同一数据区中。

S_2: 设置 2 的首字

指定第二个 16 连续字的首字， S_2 和 S_2 ＋ 15 必须在同一数据区中。

R: 结果字

R 的每一位包含了在 16 字设置中两个字比较的结果，R 中的位 n(n ＝ 00 ～ 15) 包含了字 S_1 ＋ n 和 S_2 ＋ n 的比较结果。



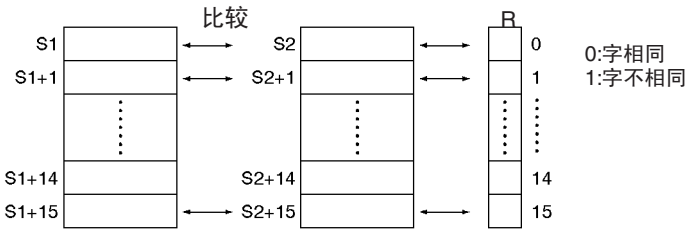
操作数规定

区域	S_1	S_2	R
CIO 区	CIO 0000 ～ CIO 6128		CIO 0000 ～ CIO 6143
工作区	W000 ～ W496		W000 ～ W511
保持位区	H000 ～ H496		H000 ～ H511
辅助位区	A000 ～ A944		A448 ～ A959
定时器区	T0000 ～ T4080		T0000 ～ T4095
计数器区	C0000 ～ C4080		C0000 ～ C4095
DM 区	D00000 ～ D32752		D00000～D32767
无区号 EM 区	E00000 ～ E32752		E00000～E32767
有区号 EM 区	En_00000 ～ 32752 (n = 0 ～ C)		En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---		
数据寄存器	---		DR0 ～ DR15
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

描述

MCMP(019)把S_1～S_1＋15的16字的内容与S_2～S_2＋15的16字的内容进行比较，当内容不相等时，字 R 中相应的位变 ON。

S_1的内容与S_2的内容相比较，S_1＋1的内容与S_2＋1的内容相比较，…，S_1＋15 的内容与 S_2 ＋ 15 的内容相比较，如果 S_1 ＋ n 的内容等于 S_2 ＋ n 的内容，R 中位 n 变 OFF，如果内容不相等，R 中位 n 变 ON，如果 16 对字的内容都相等，在指令执行后，等于标志变 ON。

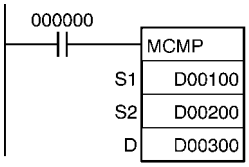


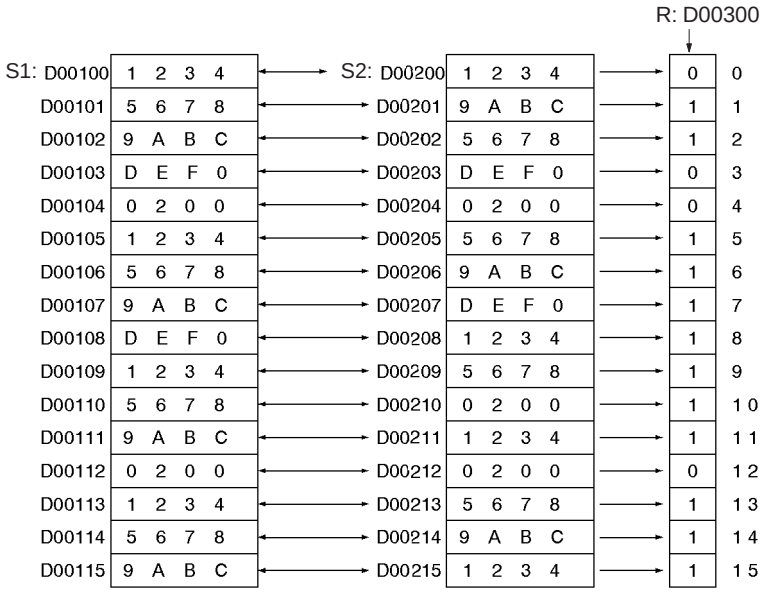
标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	结果字为 0000 （设置的两个 16 字包含了相同的数据）时为 ON 其它情况下 OFF。

例

在下面例子中，当 CIO 000000 为 ON 时，MCMP(019) 依次比较字 D00100 ～ D00115 与 D00200 ～ D00215，当字不相等时， D00300 中的相应位变 ON。



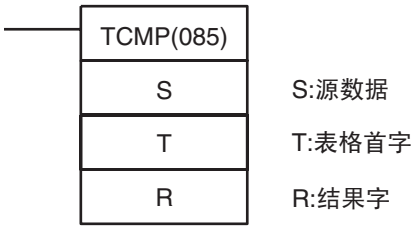


3-7-7 表比较：TCMP(085)

用途

比较源数据和 16 个连续字的内容，当两字的内容相等时，结果字中的相应位变 ON。

梯形图符号



变化

变化	ON 条件时每次循环执行	TCMP(085)
	上升沿微分时执行一次	@TCMP(085)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

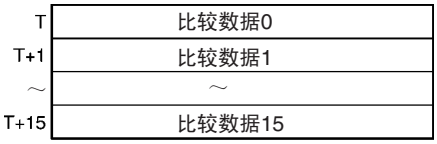
操作数

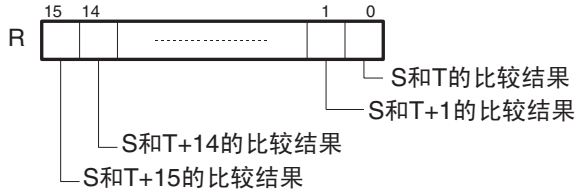
T: 表格首字

指定 16 字表格的起始字， T 和 T ＋ 15 必须在同一数据区中。

R: 结果字

R 的每一位包含了 S 和 16 字表格中每个字的比较结果，R 中的位 n(n ＝ 00 ～ 15) 包含了字 S 和 T ＋ n 的比较结果。





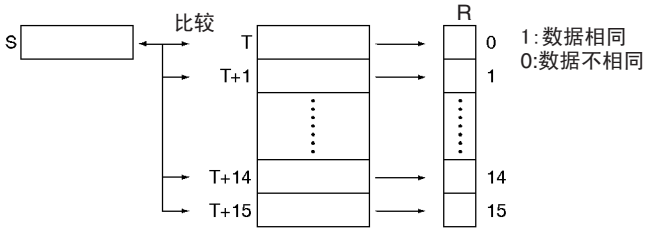
操作数规定

区域	S	T	R
CIO 区	CIO 0000 ～ CIO 6143	CIO 0000 ～ CIO 6128	CIO 0000 ～ CIO 6143
工作区	W000 ～ W511	W000 ～ W496	W000 ～ W511
保持位区	H000 ～ H511	H000 ～ H496	H000 ～ H511
辅助位区	A000 ～ A959	A000 ～ A944	A448 ～ A959
定时器区	T0000 ～ T4095	T0000 ～ T4080	T0000 ～ T4095
计数器区	C0000 ～ C4095	C0000 ～ C4080	C0000 ～ C4095
DM 区	D00000～D32767	D00000～D32752	D00000～D32767
无区号 EM 区	E00000～E32767	E00000～E32752	E00000～E32767
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	En_00000 ～ En_32752 (n = 0 ～ C)	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #FFFF （二进制）	---	
数据寄存器	DR0 ～ DR15	---	DR0 ～ DR15
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

描述

TCMP(085) 把源数据 (S) 与 16 字 (T ～ T ＋ 15) 中的每一个进行比较，如果数据相等，字 R 中的相应位变 ON。即，如果 T ＋ n 的内容与 S 相等，字 R 中的位 n 变 ON ； 如果不相等，则字 R 中的 n 位为 OFF。

S 与 T 的内容相比较，若相等，则 R 中的位 00 变 ON ； 若不等，则 R 中的位 00 变 OFF ； S 与 T ＋ 1 的内容相比较，若相等，则 R 中的位 01 变 ON ； 若不等，则 R 中的位 01 变 OFF ； …S 与 T ＋ 15 的内容相比较，若相等，则 R 中的位 15 变 ON ； 若不等，则 R 中的位 15 变 OFF。

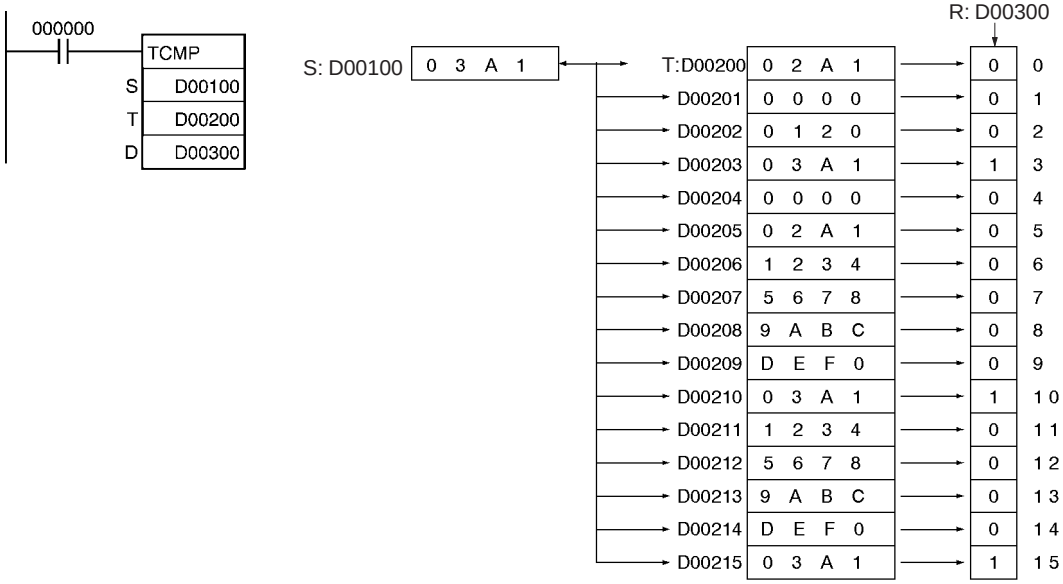


标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	结果字为 0000 （表格中 16 个字都不等于 S）时为 ON。 其它情况下 OFF。

例

在下面例子中，当 CIO 000000 为 ON 时， TCMP(085) 把字 D00100 的内容与 D00200 ～ D00215 的内容进行依次比较，当内容相等时，D00300 中的相应位变 ON ； 当内容不相等时， D00300 中的相应位为 OFF。

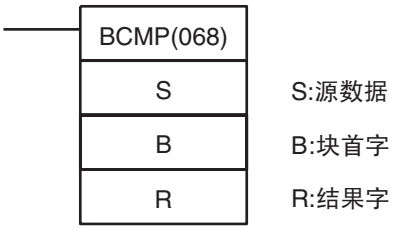


3-7-8 块比较：BCMP （068）

用途

比较源数据和 16 个范围 （由 16 个下限和 16 个上限定义），当源数据在范围内时，结果字中的相应位变 ON。

梯形图符号



变化

变化	ON 条件时每次循环执行	BCMP(068)
	上升沿微分时执行一次	@BCMP(068)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

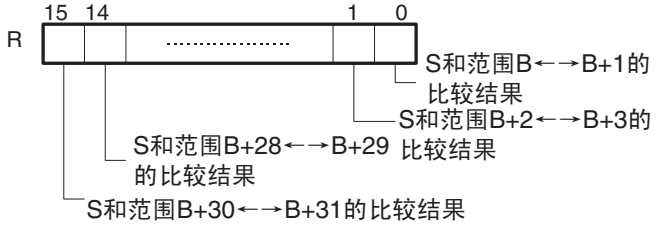
操作数

B: 块首字

指定一个32个字块的起始字（16对下/上限）。B和B＋31必须在同一数据区内。

R: 结果字

R 的每一位包含了 S 与由 16 个范围确定的 32 个字块中的某一个的比较结果，R 中的位 n(n ＝ 00 ～ 15) 包含了 S 和字块的第 n 对的比较结果。



操作数规定

区域	S	B	R
CIO 区	CIO 0000 ～ CIO 6143	CIO 0000 ～ CIO 6112	CIO 0000 ～ CIO 6143
工作区	W000 ～ W511	W0000 ～ W480	W000 ～ W511
保持位区	H000 ～ H511	H000 ～ H480	H000 ～ H511
辅助位区	A000 ～ A959	A000 ～ A928	A448 ～ A959
定时器区	T0000 ～ T4095	T0000 ～ T4064	T0000 ～ T4095
计数器区	C0000 ～ C4095	C0000 ～ C4064	C0000 ～ C4095
DM 区	D00000～D32767	D00000～D32736	D00000～D32767
无区号 EM 区	E00000～E32767	E00000～E32736	E00000～E32767
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	En_00000 ～ En_32736 (n = 0 ～ C)	En_00000 ～ En_32767 (n = 0 ～ C)

区域	S	B	R
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #FFFF （二进制）	---	
数据寄存器	DR0 ～ DR15	---	DR0 ～ DR15
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

描述

BCMP(068) 把源数据 (S) 与由 B ～ B ＋ 31 定义上下限的 16 个范围进行比较。每对的第一个字 (B ＋ 2n) 为范围 n 提供下限，第二个字 (B ＋ 2n ＋ 1) 为范围 n 提供上限 (n ＝ 0 ～ 15)。如果 S 在这些范围 （包括上，下限）的任何一个内，字 R 中的相应位变 ON，其余位变 OFF。

B	≤ S ≤	B+1	R 的位 00
B+2	≤ S ≤	B+3	R 的位 01
B+4	≤ S ≤	B+5	R 的位 02
B+6	≤ S ≤	B+7	R 的位 03
B+8	≤ S ≤	B+9	R 的位 04
B+10	≤ S ≤	B+11	R 的位 05
B+12	≤ S ≤	B+13	R 的位 06
B+14	≤ S ≤	B+15	R 的位 07
B+16	≤ S ≤	B+17	R 的位 08
B+18	≤ S ≤	B+19	R 的位 09
B+20	≤ S ≤	B+21	R 的位 10
B+22	≤ S ≤	B+23	R 的位 11
B+24	≤ S ≤	B+25	R 的位 12
B+26	≤ S ≤	B+27	R 的位 13
B+28	≤ S ≤	B+29	R 的位 14
B+30	≤ S ≤	B+31	R 的位 15

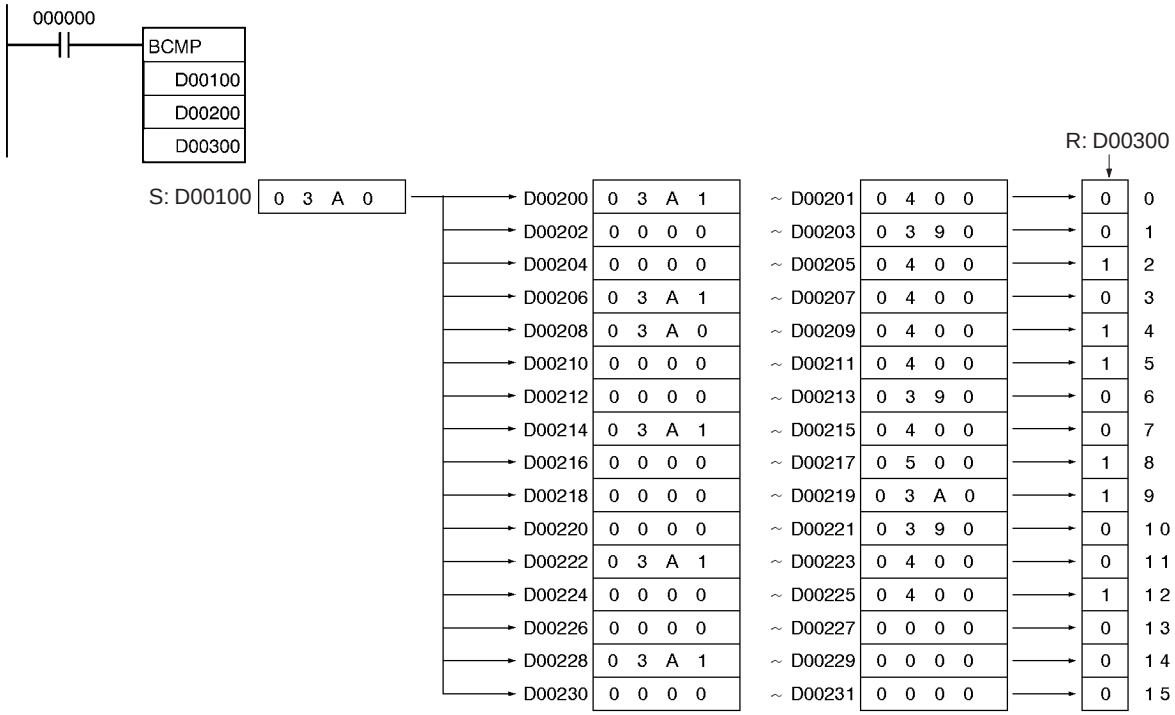
例如：如果 S 位于第一个范围内 (B<=S<=B+1)，则字 R 中的位 00 变 ON ；如果 S 位于第二个范围内 (B ＋ 2<=S<=B+3)，则字 R 中的位 01 变 ON ；…；如果 S 位于第十六个范围内 (B ＋ 30<=S<=B+31)，则字 R 中的位 15 变 ON。而字 R 中的其余位变 OFF。

标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	结果字为 0000 （S 不在 16 个范围中的任何一个内）时为 ON。 其它情况下 OFF。

注意 如果下限大于上限，不会发生错误，但 0 （不在范围内）将被输出给字 R 的相应位。

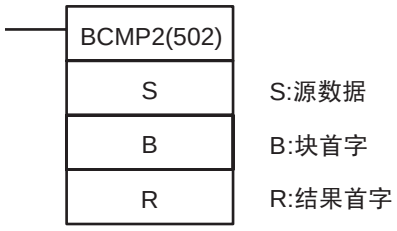
例 在下面例子中，当 CIO 000000 ON 时，BCMP(068) 指令把 D00100 的内容与在 D00200 ～ D00231 中定义的 16 个范围进行比较。当 S 位于范围内时，D00300 的相应位变 ON ； 当 S 不在范围内时，相应位变 OFF。



3-7-9 扩展块比较：BCMP2(512) （仅限于 CJ1M）

用途 比较源数据与最多可达 256 个范围（由 256 个下限和 256 个上限定义），当源数据在范围内时，结果字中的相应位变 ON。仅有 CJ1M CPU 单元支持 BCMP2(502) 指令。

梯形图符号



变化

变化	ON 条件时每次循环执行	BCMP2(502)
	上升沿微分时执行一次	@BCMP2(502)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

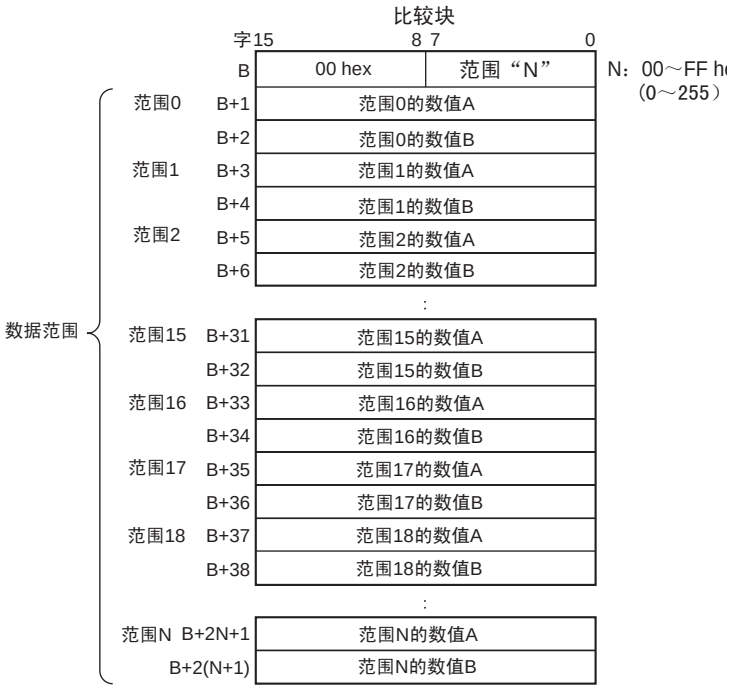
适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数

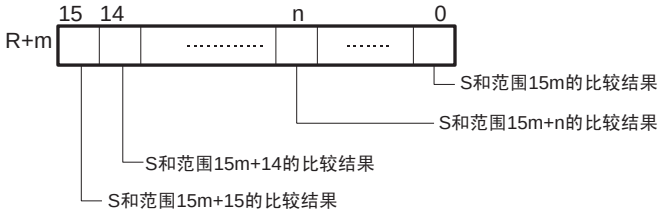
B: 块首字

指定最多可包含 513 个字（包括多达 256 对下 / 上限）的比较块的起始字。所有的字必须在同一数据区内。



R: 结果字

字 R 中的每一位包含了 S 与由比较块定义的某个范围的比较结果。结果字的最大个数为 16，即， m = 0 ～ 15。



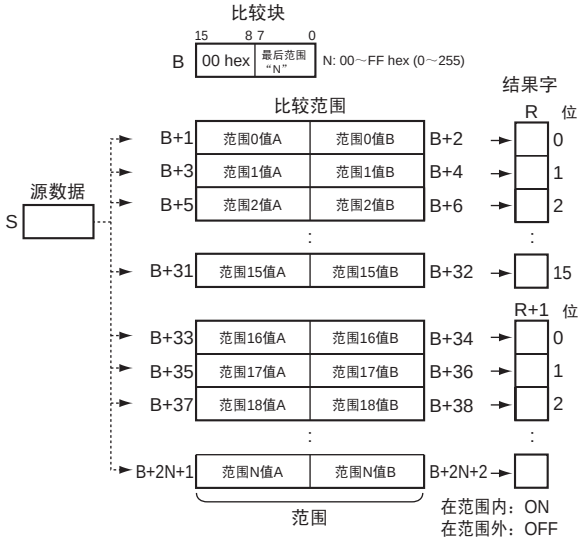
操作数规定

区域	S	B	R
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	---		
有区号 EM 区	---		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767		
常数	#0000 ～ #FFFF（二进制）	---	
数据寄存器	DR0 ～ DR15	---	
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

描述

BCMP2(502) 把源数据 (S) 与由比较块中定义上下限的范围进行比较。如果 S 在这些范围中的任何一个内 （包括上下限），则结果字 （R ～ R ＋ 15（最大））中的相应位变 ON，字 R 中的其余位变 OFF。

范围标号由在 B 的低字节中设置的 N 值决定。N 可在 0 ～ 255 之间。B 的高字节必须为 00H。

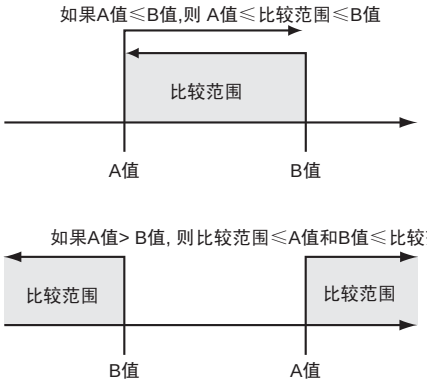


范围个数

比较块中的范围个数在块的第一个字中设置。最多可设置 256 个范围。

设定范围

如下所示，每个范围的 A 值和 B 值将通过比较哪个值大来决定如何执行比较操作。



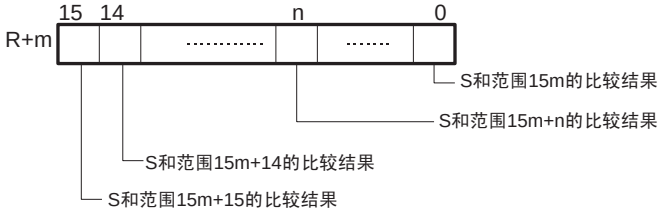
例

当 B+1 ≤ B+2 时
如果 B+1 ≤ S ≤ B+2， 则字 R 中的位 0 变 ON，
如果 B+3 ≤ S ≤ B+4， 则字 R 中的位 1 变 ON，
如果 S < B+5 and B+6 < S， 则字 R 中的位 2 变 OFF， 以及
如果 S < B+7 and B+8 < S， 则字 R 中的位 3 变 OFF。

当 B+1 > B+2 时
如果 S ≤ B+2 and B+1 ≤ S， 则字 R 中的位 0 变 ON，
如果 S ≤ B+4 and B+3 ≤ S， 则字 R 中的位 1 变 ON，
如果 B+6 < S < B+5， 则字 R 中的位 2 变 OFF， 以及
如果 B+8 < S < B+7， 则字 R 中的位 3 变 OFF。

结果存储地址

结果输出到字 R 的相应位中。如果超过 16 个比较范围，将使用紧随字 R 后的连续字。结果字的最大个数为 16，即 m ＝ 0 ～ 15。



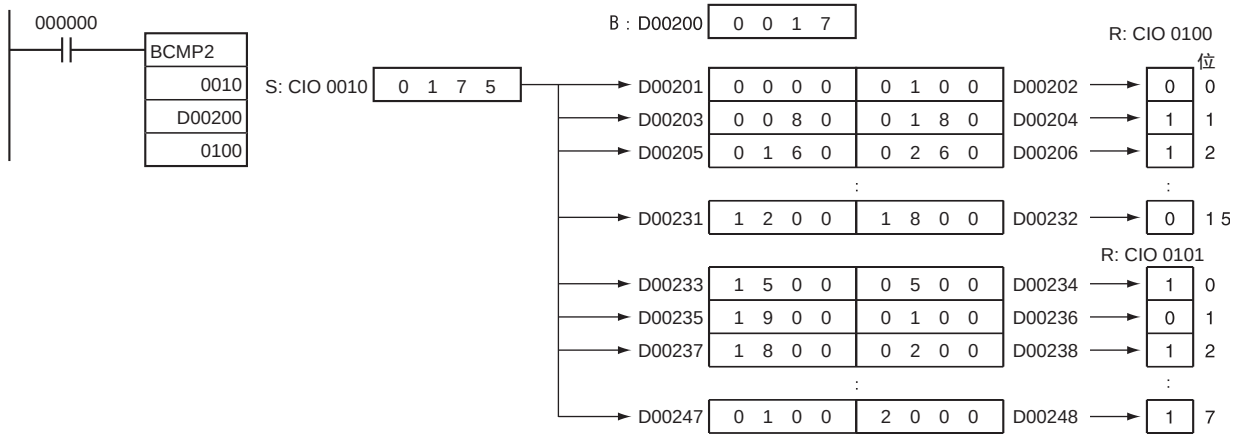
标志

名称	标记	操作
错误标志	ER	OFF

例

在下例中，当 CIO 000000 为 ON 时， BCMP2(502) 把 CIO 0010 的内容与在 D00200 ～ D00247（N ＝ 17H ＝ 23D，即 24 个范围）定义的 24 个范围进行比较，并当 S 位于范围内时，把 CIO 0100 和 CIO 0101 中的相应位变 ON，而当 S 不在范围内时，把相应位变 OFF。例如，如果 CIO 0010 中的源数据位于由 D00201 和 D00202 定义的范围内，那么 CIO 0100 中的位 00 变 ON ；而当它不在范围内时， CIO 0100 中的位 00 变 OFF。同样，把 CIO 0010 中源数据与由 D00203 和 D00204，D00207 和 D00208，以及比较块中的其它字定义的

范围进行比较，并根据比较结果对 CIO 0100 中的位 1， CIO 1010 中的位 7，以及结果字中的其它位进行操作。



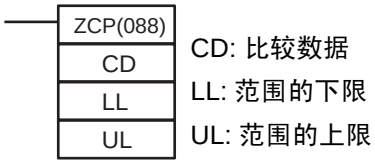
3-7-10 区域比较：ZCP(088)

用途

将一个 16 位无符号二进制值 (CD) 和由下限 LL 与上限 UL 定义的范围比较。其结果输出到算术标志中。

仅有 CS1- H， CJ1- H， CJ1M，以及 CS1D CPU 单元支持这条指令。

梯形图符号



变化

变化	ON 条件时每次循环执行	ZCP(088)
	上升沿微分时执行一次	不支持
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	CD	LL	UL
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		

区域	CD	LL	UL
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #FFFF （二进制）		
数据寄存器	DR0 ～ DR15		
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

描述

ZCP(088) 把 CD 中的 16 位无符号二进制数据与由 LL 和 UL 定义的范围进行比较，并把结果输出到辅助区内的大于，等于，小于标志中。（小于等于，大于等于，以及不等于标志保持不变）。

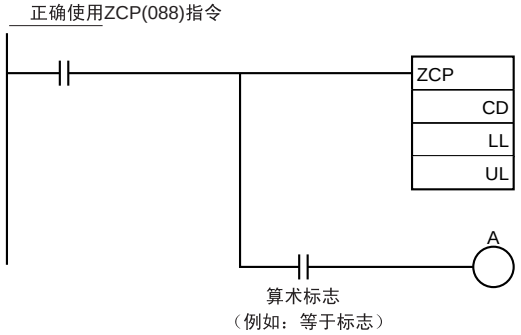
算术标志状态

下表列举了执行 ZCP(088) 指令后算术标志的状态。

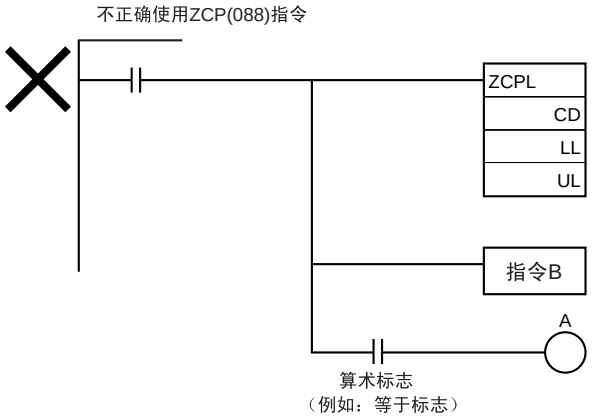
ZCP(088) 结果	标志状态		
	>	=	<
CD > UL	ON	OFF	OFF
CD = UL	OFF	ON	
LL < CD < UL			
CD = LL			
CD < LL		OFF	ON

在程序中使用 ZCP(088) 结果

执行 ZCP(088) 指令后，其结果将反映在算术标志中。用和控制 ZCP(088) 一样的输入条件分支去控制所需的输出或右侧指令，如下图所示。这种情况下，当 LL ≤ CD ≤ UL 时，等于标志和输出 A 将变 ON。



不要在 ZCP(088) 和算术标志控制的指令之间编写其它指令，因为其它指令可能会改变算术标志的状态。这种情况下，指令 B 的结果可能会改变 ZCP(088) 的结果。



标志

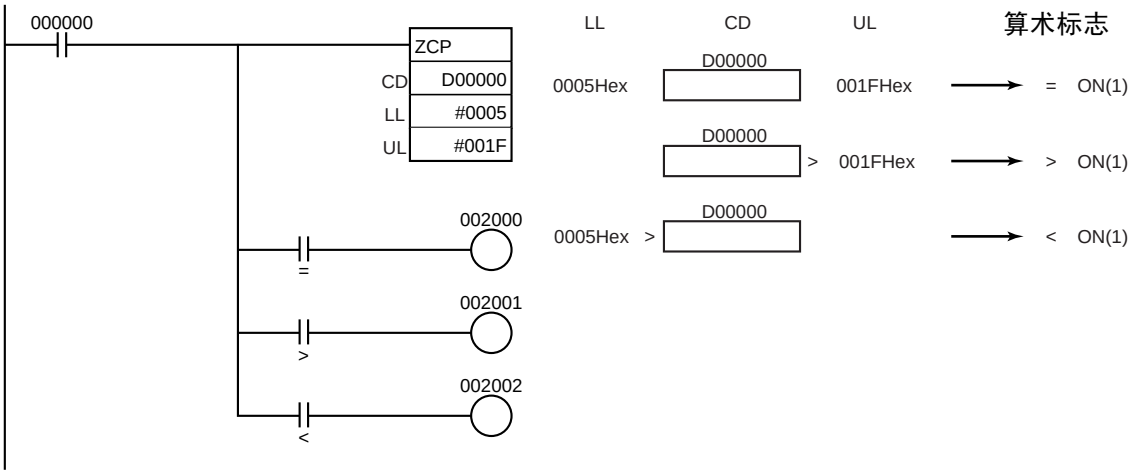
名称	标记	操作
错误标志	ER	LL > UL 时 ON。
大于标志	>	CD > UL 时 ON。 其它情况下 OFF。
大于等于标志	> =	保持不变
等于标志	=	LL ≤ CD ≤ UL 时 ON。 其它情况下 OFF。
不等于标志	<>	保持不变
小于标志	<	CD < LL 时 ON。 其它情况下 OFF。
小于等于标志	< =	保持不变
负数标志	N	保持不变

注意

不要在 ZCP(088) 和访问 ZCP(088) 结果的输入条件之间编写其它指令，因为其它指令可能会改变算术标志的状态。

例

在下面例子中，当 CIO 000000 为 ON 时，D00000 中的 16 位无符号二进制数据与范围 0005H ～ 001FH（十进制：0 ～ 31）进行比较，并把结果输出到算术标志中。
如果 0005 Hex ≤ 的内容 D00000 ≤ 001F Hex， CIO 000200 变 ON。
如果 D00000 的内容 > 001F Hex， CIO 000201 变 ON。
如果 D00000 的内容 < 0005 Hex， CIO 000202 变 ON。



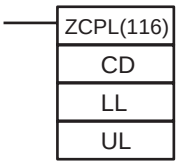
3-7-11 双字区域比较：ZCPL(116)

用途

比较一个 32 位无符号二进制值 (CD ＋ 1，CD) 和由下限 (LL ＋ 1，LL) 与上限 (UL ＋ 1， UL) 定义的范围。其结果输出到算术标志中。

仅有 CS1-H， CJ1-H， CJ1M，以及 CS1D CPU 单元支持这条指令。

梯形图符号



CD: 比较数据首字
LL: 范围的下限首字
UL: 范围的上限首字

变化

变化	ON 条件时每次循环执行	ZCP(088)
	上升沿微分时执行一次	不支持
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	CD	LL	UL
CIO 区	CIO 0000 ～ CIO 6142		
工作区	W000 ～ W510		
保持位区	H000 ～ H510		
辅助位区	A000 ～ A958		
定时器区	T0000 ～ T4094		
计数器区	C0000 ～ C4094		
DM 区	D00000 ～ D32766		
无区号 EM 区	E00000 ～ E32766		
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)		

区域	CD	LL	UL
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 0000 ～ #FFFF FFFF （二进制）		
数据寄存器	---		
索引寄存器	IR0 ～ IR15		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

描述

ZCPL(116) 把 CD ＋ 1， CD 中的 32 位无符号二进制数据与由 LL ＋ 1， LL 和 UL ＋ 1， UL 定义的范围进行比较，并把结果输出到辅助区内的大于，等于，小于标志中。（小于等于，大于等于，以及不等于标志保持不变）。

算术标志状态

下表列举了执行 ZCPL(116) 指令后算术标志的状态。

ZCPL(116) 结果	标志状态		
	>	=	<
CD+1, CD > UL+1, UL	ON	OFF	OFF
CD+1, CD = UL+1, UL	OFF	ON	
LL+1, LL < CD+1, CD < UL+1, UL			
CD+1, CD = LL+1, LL			
CD+1, CD < LL+1, LL		OFF	ON

在程序中使用 ZCPL(116) 结果

执行 ZCPL(116) 指令后，其结果将反映在算术标志中。用和控制 ZCPL(116) 一样的输入条件分支去控制所需的输出或右侧指令 , 如下图所示。

不要在 ZCPL(116) 和算术标志控制的指令之间编写其它指令，因为其它指令可能会改变算术标志的状态。

除了 ZCPL(116) 是用于比较 32 位数值而非 16 位数值之外，其操作基本上和 ZCP(088) 的操作是一样的。参阅 *3-7-10 区域比较: ZCP(088)* 中的如何在程序中使用比较结果以及程序举例部分。

标志

名称	标记	操作
错误标志	ER	LL+1，LL > UL+1，UL 时 ON。
大于标志	>	CD > UL+1， UL 时 ON。 其它情况下 OFF。

名称	标记	操作
大于等于标志	> =	保持不变
等于标志	=	LL+1， LL ≤ CD+1， CD ≤ UL+1， UL 时 ON。 其它情况下 OFF。
不等于标志	<>	保持不变
小于标志	<	CD+1， CD < LL+1， LL 时 ON。 其它情况下 OFF。
小于等于标志	< =	保持不变
负数标志	N	保持不变

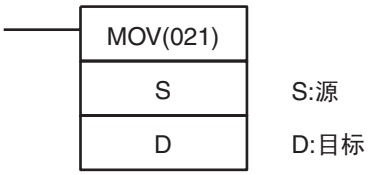
注意 不要在 ZCPL(116) 和访问 ZCPL(116) 结果的输入条件之间编写其它指令，因为其它指令可能会改变算术标志的状态。

3-8 数据传送指令

3-8-1 传送：MOV(021)

用途 传送数据的一个字到指定字中。

梯形图符号



变化

变化	ON 条件时每次循环执行	MOV(021)
	上升沿微分时执行一次	@MOV(021)
	下降沿微分时执行一次	不支持
立即刷新功能 （见注）		!MOV(021)
组合变化	上升沿微分立即刷新执行一次 （见注）	!@MOV(021)

注 CS1D CPU 单元不支持立即刷新。

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

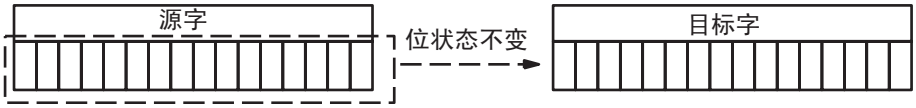
操作数规定

区域	S	D
CIO 区	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W511	
保持位区	H000 ～ H511	
辅助位区	A000 ～ A959	A448 ～ A959
定时器区	T0000 ～ T4095	
计数器区	C0000 ～ C4095	
DM 区	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	

区域	S	D
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#0000 ～ #FFFF （二进制）	---
数据寄存器	DR0 ～ DR15	
索引寄存器	---	
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –) IR0 ～ , –(– –) IR15	

描述

传送 S 到 D。如果 S 是一个常数，此数值可用来作为数据设定。



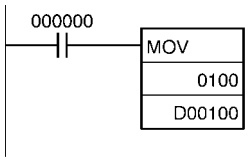
MOV(021) 具有立即刷新变化 (! MOV(021))，可为 S 指定一个外部输入位，并且为 D 指定一个外部输出位。用于 S 的输入位一般将在执行前刷新，用于 D 的输出位在执行后刷新，除非此位被分配给了组 2 高密度 I/O 单元，高密度专用 I/O 单元，或者安装在 SYSMAC BUS 远程 I/O 从属机架上的单元。

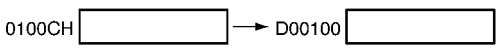
标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	正在传送的数据是 0000 时 ON。 其它情况下 OFF。
负数标志	N	正在传送的数据最左位是 1 时 ON。 其它情况下 OFF。

例

在下面例子中，当CIO 000000为ON时，CIO 0100 的内容被拷贝到D00100中。



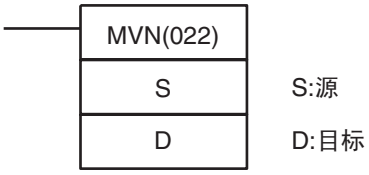


3-8-2 传送反：MVN(022)

用途

传送数据的一个字的补码到指定字中。

梯形图符号



变化

变化	ON 条件时每次循环执行	MVN(022)
	上升沿微分时执行一次	@MVN(022)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

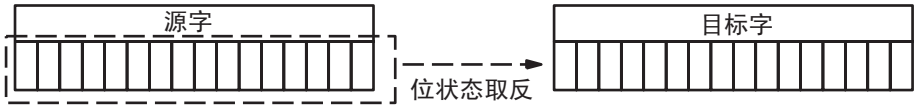
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	S	D
CIO 区	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W511	
保持位区	H000 ～ H511	
辅助位区	A000 ～ A959	A448 ～ A959
定时器区	T0000 ～ T4095	
计数器区	C0000 ～ C4095	
DM 区	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#0000 ～ #FFFF （二进制）	---
数据寄存器	DR0 ～ DR15	
索引寄存器	---	
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –) IR0 ～ , –(– –) IR15	

描述

MVN(022) 指令对 S 中的位进行取反，并把结果传送到 D 中。S 中的内容保持不变。

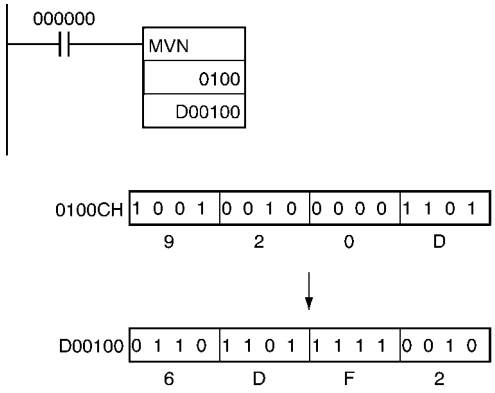


标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	指令执行后， D 的内容是 0000 时 ON。 其它情况下 OFF。
负数标志	N	指令执行后， D 最左位是 1 时 ON。 其它情况下 OFF。

例

在下面例子中，当 CIO 000000 为 ON 时，CIO 0100 中位的状态被取反，并把结果拷贝到 D00100 中。

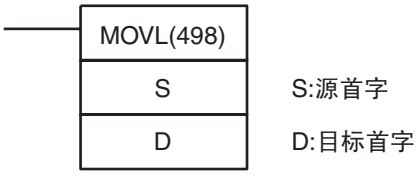


3-8-3 双字传送：MOVL(498)

用途

传送数据的两个字到指定字中。

梯形图符号



变化

变化	ON 条件时每次循环执行	MOVL(498)
	上升沿微分时执行一次	@MOVL(498)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	S	D
CIO 区	CIO 0000 ～ CIO 6142	
工作区	W000 ～ W510	
保持位区	H000 ～ H510	
辅助位区	A000 ～ A958	A448 ～ A958
定时器区	T0000 ～ T4094	
计数器区	C0000 ～ C4094	
DM 区	D00000 ～ D32766	
无区号 EM 区	E00000 ～ E32766	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#00000000 ～ #FFFFFFFF （二进制）	---
数据寄存器	---	
索引寄存器	IR0 ～ IR15	
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –) IR0 ～ , 1–(– –) IR5	

描述

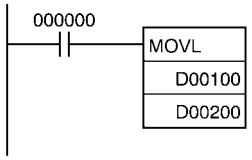
MOVL(498) 指令把 S ＋ 1 和 S 传送到 D ＋ 1 和 D 中。如果 S ＋ 1 和 S 为常数，则此数字可用与数据设置。

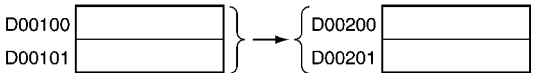
标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	指令执行后， D ＋ 1 和 D 的内容是 00000000 时 ON。 其它情况下 OFF。
负数标志	N	指令执行后， D ＋ 1 最左位是 1 时 ON。 其它情况下 OFF。

例

在下面例子中，当 CIO 000000 为 ON 时，D00101 和 D00100 的内容被拷贝到 D00201 和 D00200 中。



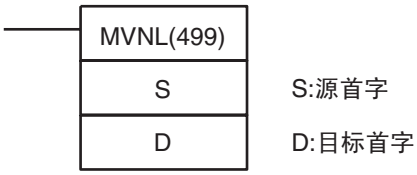


3-8-4 双字传送反：MVNL(499)

用途

传送数据的两个字的补码到指定字中。

梯形图符号



变化

变化	ON 条件时每次循环执行	MVNL(499)
	上升沿微分时执行一次	@MVNL(499)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

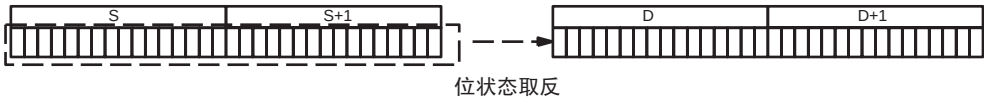
操作数规定

区域	S	D
CIO 区	CIO 0000 ～ CIO 6142	
工作区	W000 ～ W510	
保持位区	H000 ～ H510	
辅助位区	A000 ～ A958	A448 ～ A958
定时器区	T0000 ～ T4094	
计数器区	C0000 ～ C4094	
DM 区	D00000 ～ D32766	
无区号 EM 区	E00000 ～ E32766	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	

区域	S	D
常数	#00000000 ～ #FFFFFFFF（二进制）	---
数据寄存器	---	
索引寄存器	---	
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –) IR0 ～ , –(– –) IR15	

描述

MVNL(499) 指令对 S+1 和 S 中的位进行取反，并把结果传送到 D ＋ 1 和 D 中。S ＋ 1 和 S 中的内容保持不变。

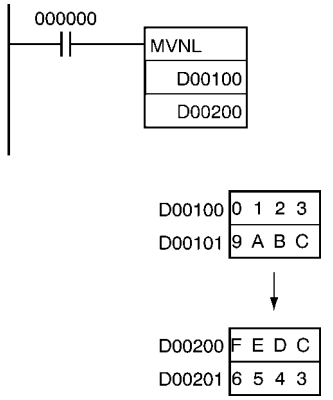


标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	指令执行后， D ＋ 1 和 D 的内容是 00000000 时 ON。 其它情况下 OFF。
负数标志	N	指令执行后， D ＋ 1 的左高位是 1 时 ON。 其它情况下 OFF。

例

在下面例子中，当 CIO 000000 为 ON 时，D00101 和 D00100 中位的状态被取反，并把结果拷贝到 D00201 和 D00200 中。（D00101 和 D00100 中的内容保持不变）。

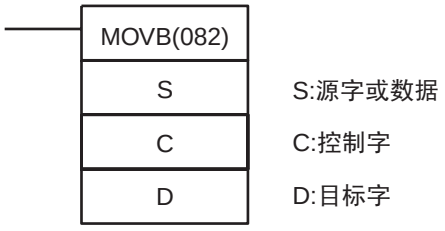


3-8-5 位传送：MOVB(082)

用途

传送指定位。

梯形图符号



变化

变化	ON 条件时每次循环执行	MOVB(082)
	上升沿微分时执行一次	@MOVB(082)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

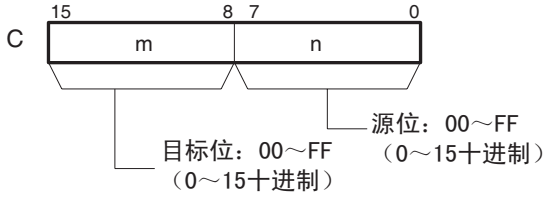
适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数

C: 控制字

C 中最右边的两个数字指定 S 中的哪一位是源位，C 中最左边的两个数字指定 D 中的哪一位是目标位。

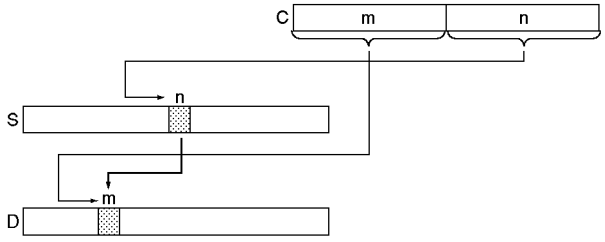


操作数规定

区域	S	C	D
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #FFFF （二进制）	仅有指定值	---
数据寄存器	DR0 ～ DR15		
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –) IR0 ～ , –(– –) IR15		

描述

MOVB(082) 指令把 S 中的指定位 (n) 拷贝给 D 中的指定位 (m)。目标字中的其余位保持不变。



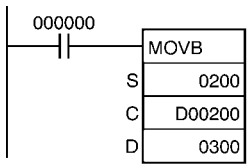
注 可指定一个字同时作为 S 和 D，并在同一个字内进行位拷贝。

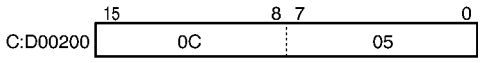
标志

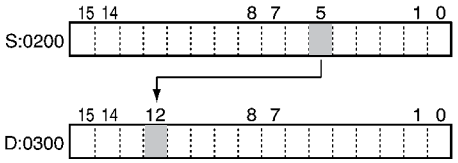
名称	标记	操作
错误标志	ER	C 中最右边的两个数字和最左边的两个数字都不在指定的范围 (00 ～ 0F) 内时 ON。 其它情况下 OFF。

例

在下面例子中，当CIO 000000为ON时，依据控制字的数值0C05，把源字(CIO 0200) 中的第五位拷贝给目标字 (CIO 0300) 中的第十二位。





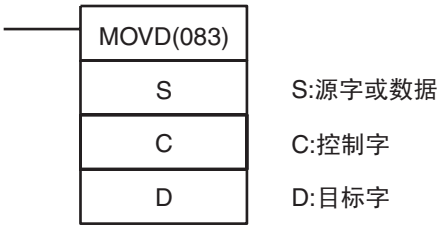


3-8-6 传送数字：MOVD(083)

用途

传送一个或多个指定的数字。（每个数字由 4 位组成）。

梯形图符号



变化

变化	ON 条件时每次循环执行	MOVD(083)
	上升沿微分时执行一次	@MOVD(083)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

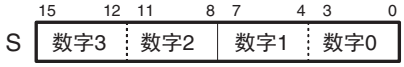
适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数

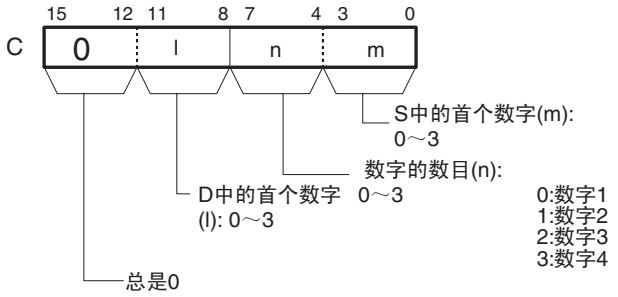
S: 源字

按从右到左方向读取源数字，如果必要，绕回最右边的数字 （数字 0）。



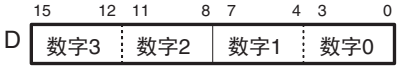
C: 控制字

如下图所示， C 中的前三个数字表示第一个源数字 (m)，传送的数字个数 (n)，以及第一个目标数字 (l)。



D: 目标数字

按从右到左方向写入目标数字，如果必要，绕回最右边的数字 （数字 0）。



操作数规定

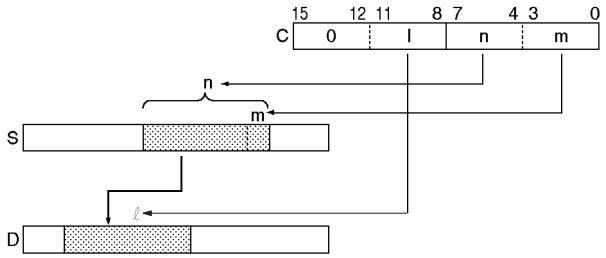
区域	S	C	D
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #FFFF （二进制）	仅有指定值	---
数据寄存器	DR0 ～ DR15		

区域	S	C	D
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –) IR0 ～ , –(– –) IR15		

描述

MOVD(083) 指令把 S 中的 n 个数字（从数字 m 开始）的内容拷贝到 D 中（从数字 1 开始）。只改变被指定的数字，其余的保持不变。

如果读取或写入的数字的个数超出 S 或 D 中最左边的数字， MOVD(083) 将绕回到这个字的最右边的数字。



注 可指定一个字同时作为 S 和 D，并在同一个字内进行位拷贝。

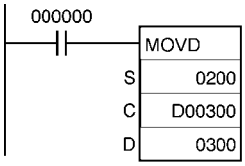
标志

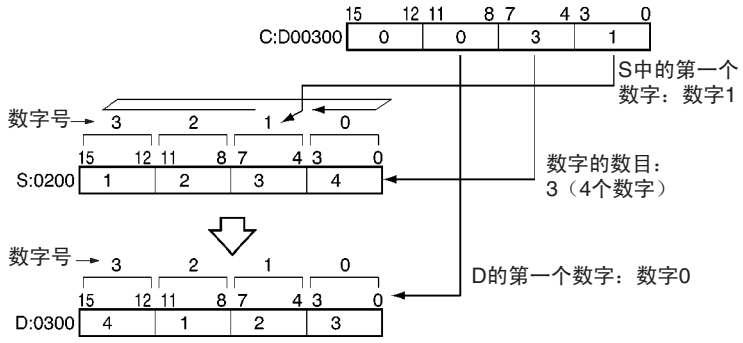
名称	标记	操作
错误标志	ER	C 中前三个数字中的任何一个不在指定的范围 (0 ～ 3) 内时 ON。 其它情况下 OFF。

例

传送四个数字

在下面例子中，当 CIO 000000 为 ON 时，把 CIO 0200 中的数据的四个数字拷贝到 CIO 0300 中。依据控制字的数值 (0031)，从 CIO 0200 的数字 1 和 CIO 0300 的数字 0 间开始进行传送。

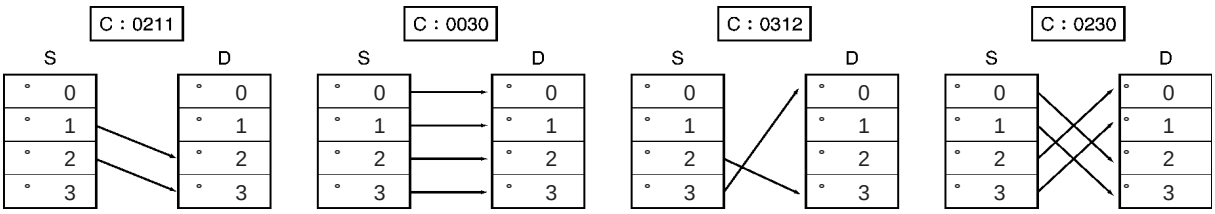




注 在读完 S 最左边的数字 （数字 3）后， MOVD(083) 绕回到最右边的数字 （数字 0）。

控制字 C 的举例

下图显示了在各种不同的 C 值下数据传送的例子。

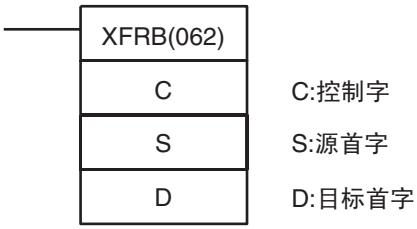


3-8-7 多位传送：XFRB(062)

用途

传送指定数目的连续位。

梯形图符号



变化

变化	ON 条件时每次循环执行	XFRB(062)
	上升沿微分时执行一次	@XFRB(062)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

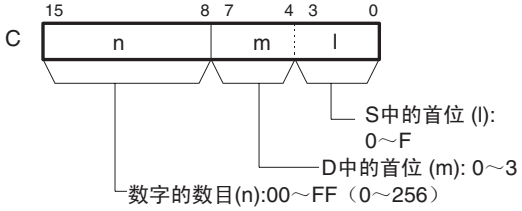
适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数

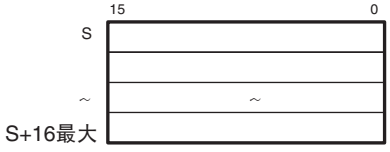
C: 控制字

如下图所示， C 中的前三个数字表示第一个源数字 (m)，传送的数字个数 (n)，以及第一个目标数字 (e)。



S: 源首字

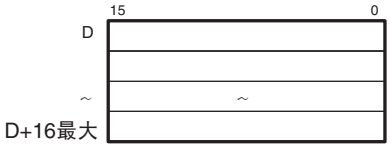
指定源起始字。按从右到左方向读取位，如果必要，从与之连续的字中继续读取。（最多可到 S ＋ 16）



注 源字必须在同一数据区内。

D: 目标首字

指定目标起始字。按从右到左方向写入位，如果必要，继续向与之连续的字中写入位。（最多可到 D ＋ 16）



注 目标字必须在同一数据区内。

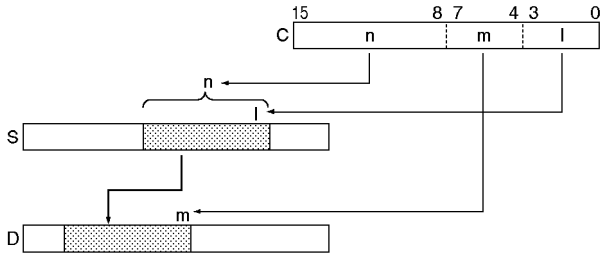
操作数规定

区域	C	S	D
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	仅有指定值	---	---
数据寄存器	DR0 ～ DR15	---	
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ 5+(++) ,–(– –) IR0 ～ , –(– –) IR15		

描述

XFRB(062) 最多可把源字中的 255 个连续位 （从 S 的位 1 开始）传送到目标字 D 中 （从 D 的位 m 开始）。目标字中未被源位覆盖的位保持不变。

如下图所示，开始位和位的个数均在控制字 C 中指定。



源字与目标字可以重叠。通过传送重叠了几个字的数据，数据可在数据区内更有效地被打包。（在处理用于控制位置的位置数据时特别有用）。

由于源字与目标字可以重叠， XFRB(062) 可和 ANDW(034) 结合起来使用，通过 n 个空格来移动 m 位。

标志

名称	标记	操作
错误标志	ER	OFF

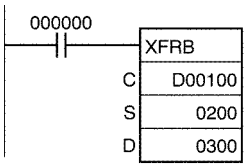
注意

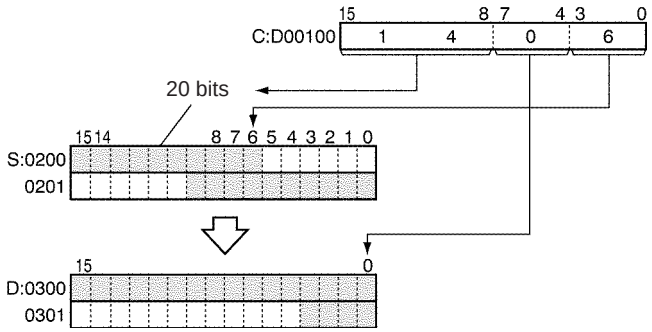
每执行 XFRB(062) 指令一次，就可以传送最多可达 255 位的数据。

必须确保源字和目标字不超出数据区的末端。

例

在下面例子中，当 CIO 000000 为 ON 时，把从 CIO 020006 开始的 20 位拷贝给从 CIO 030000 开始的 20 位。



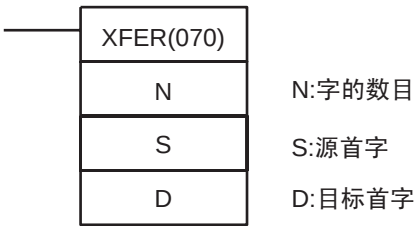


3-8-8 块传送：XFER(070)

用途

传送指定数目的连续字。

梯形图符号



变化

变化	ON 条件时每次循环执行	XFER(070)
	上升沿微分时执行一次	@XFER(070)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

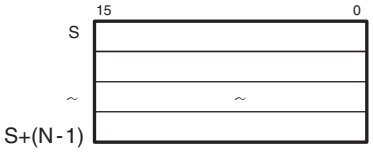
操作数

N: 字的个数

指定要传送的字的个数。N 的范围为 0000 ～ FFFF（十进制：0 ～ 65,535）。

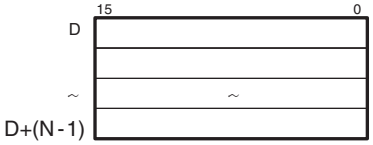
S: 源首字

指定源起始字。 S 和 S ＋ (N–1) 必须在同一数据区内。



D: 目标首字

指定目标起始字。 D 和 D ＋ (N–1) 必须在同一数据区内。



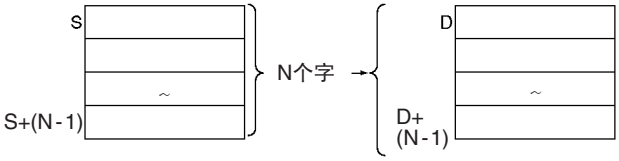
操作数规定

区域	N	S	D
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		

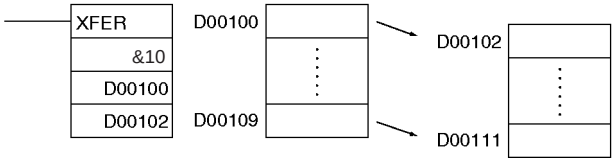
区域	N	S	D
常数	#0000 ～ #FFFF（二进制）或 &0 ～ &65535	---	---
数据寄存器	DR0 ～ DR15	---	
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –) IR0 ～ , –(– –) IR15		

描述

XFER(070) 指令把从 S 开始的 N 个字 (S ～ S ＋ (N–1)) 拷贝给从 D 开始的 N 个字 (D ～ D ＋ (N–1))。



源字和目标字可以重叠，因此 XFER(070) 指令可以执行字移动操作。



标志

名称	标记	操作
错误标志	ER	OFF

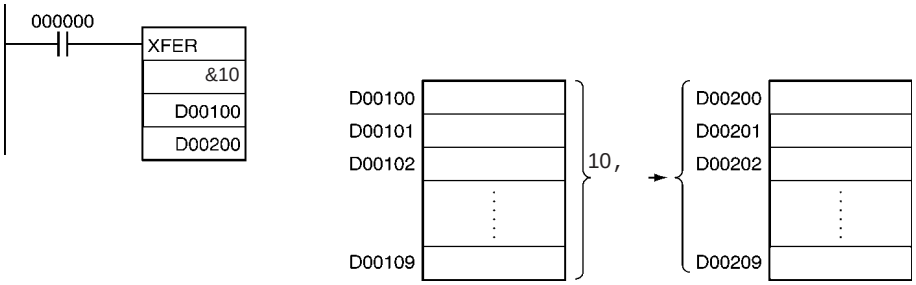
注意

必须确保源字 (S ～ S ＋ N–1) 和目标字 (D ～ D ＋ N–1) 不超出数据区的末端。

当传送大量的字时，完成 XFER(070) 指令的执行需要一定的时间。在这种情况下，如果在指令执行过程中发生电源中断，XFER(070) 指令的执行可能不会完成。

例

在下面例子中，当 CIO 000000 为 ON 时，把 D00100 ～ D00109 中的 10 个字拷贝到 D00200 ～ D00209 中。

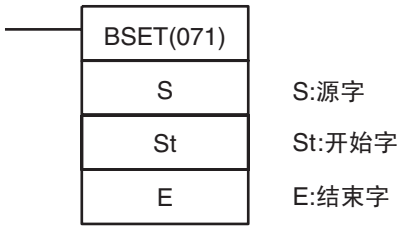


3-8-9 块设置：BSET(071)

用途

把同一个字拷贝到一个连续字的范围中。

梯形图符号



变化

变化	ON 条件时每次循环执行	BSET(071)
	上升沿微分时执行一次	@BSET(071)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

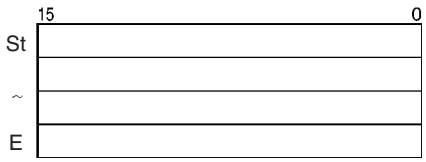
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

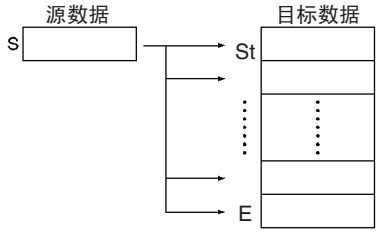
操作数

S: 源字
指定源数据或包含源数据的字。

St: 开始字
指定目标范围的开始字。

E: 结束字
指定目标范围的结束字。





注 St 和 E 必须在同一数据区内。

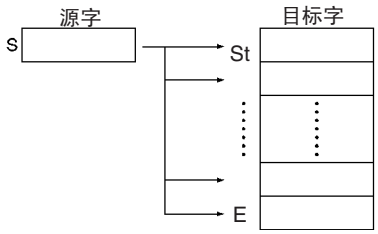
操作数规定

区域	S	St	E
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959	A448 ～ A959	
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		

区域	S	St	E
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #FFFF （二进制）	---	
数据寄存器	DR0 ～ DR15	---	
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –) IR0 ～ , 15–(– –) IR		

描述

BSET(071) 指令把同一个源字 (S) 拷贝到范围 St ～ E 中的所有目标字中。



标志

名称	标记	操作
错误标志	ER	St>E 时 ON 其它情况下 OFF。

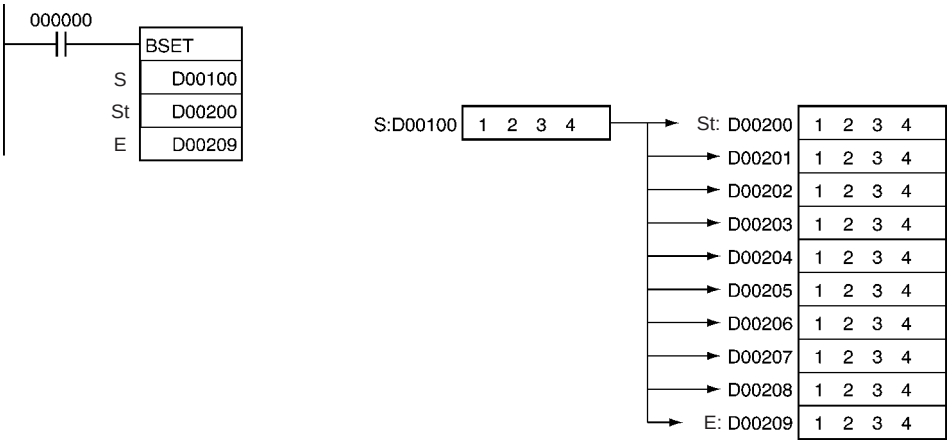
注意

必须确保起始字 (St) 和结束字 (E) 在同一数据区内，并且 St ≤ E。

当把源数据传送给大量的目标字时，完成 BSET(071) 指令的执行需要一定的时间。在这种情况下，如果在指令执行过程中发生电源中断， BSET(071) 指令的执行可能不会完成。

例

在下面例子中，当CIO 000000为ON时，把D00100中的源数据拷贝到D00200 ～ D00209 中。

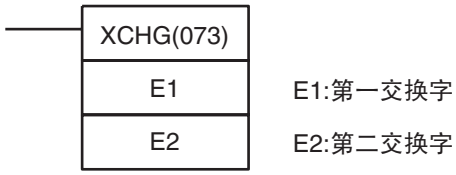


3-8-10 数据交换：XCHG(073)

用途

交换两个指定字的内容。

梯形图符号



变化

变化	ON 条件时每次循环执行	XCHG(073)
	上升沿微分时执行一次	@XCHG(073)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

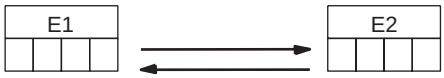
操作数规定

区域	E1	E2
CIO 区	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W511	
保持位区	H000 ～ H511	
辅助位区	A448 ～ A959	
定时器区	T0000 ～ T4095	
计数器区	C0000 ～ C4095	
DM 区	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	

区域	E1	E2
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	
数据寄存器	DR0 ～ DR15	
索引寄存器	---	
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –) IR0 ～ , –(– –) IR15	

描述

XCHG(073) 指令把 E1 和 E2 中的内容进行交换。



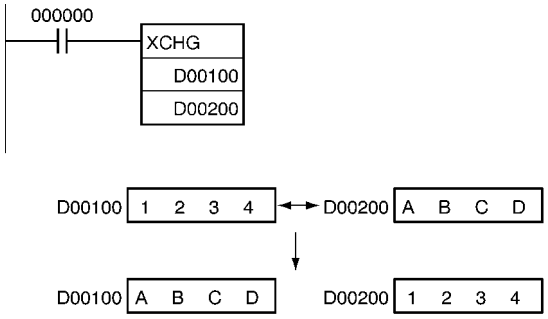
标志

名称	标记	操作
错误标志	ER	OFF 或不变 （见注）
等于标志	=	OFF 或不变 （见注）
负数标志	N	OFF 或不变 （见注）

注 在 CS1 和 CJ1 CPU 单元中，这些标志都变 OFF。
在 CS1- H， CJ1- H， CJ1M 和 CS1D CPU 单元中，这些标志保持不变。

例

在下面例子中，当 CIO 000000 为 ON 时，D00100 的内容与 D00200 的内容进行交换。

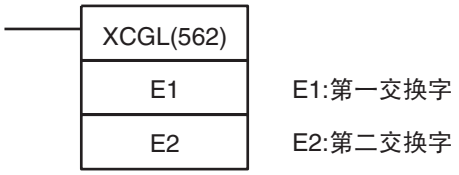


3-8-11 双数据交换：XCGL(562)

用途

把一对连续字的内容与另一对连续字的内容进行交换。

梯形图符号



变化

变化	ON 条件时每次循环执行	XCGL(562)
	上升沿微分时执行一次	@XCGL(562)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

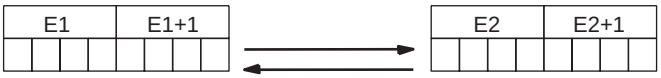
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

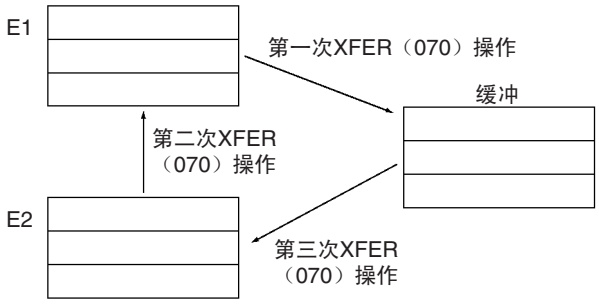
区域	E1	E2
CIO 区	CIO 0000 ～ CIO 6142	
工作区	W000 ～ W510	
保持位区	H000 ～ H510	
辅助位区	A448 ～ A958	
定时器区	T0000 ～ T4094	
计数器区	C0000 ～ C4094	
DM 区	D00000 ～ D32766	
无区号 EM 区	E00000 ～ E32766	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	---
数据寄存器	---	
索引寄存器	IR0 ～ IR15	
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –) IR0 ～ ,–(– –) IR15	

描述

XCGL(562) 指令把 E1 ＋ 1 和 E1 的内容与 E2 ＋ 1 和 E2 的内容进行交换。



若要交换 3 个或更多的字，可使用 XFER(070) 指令把字传送到第三组字（缓冲区）中，如下图所示。

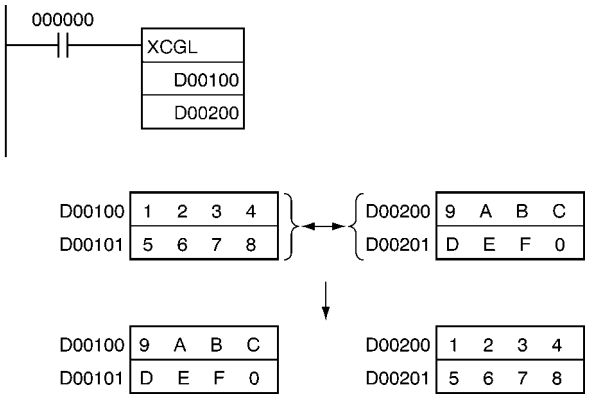


标志

名称	标记	操作
错误标志	ER	OFF 或不变 （见注）
等于标志	=	OFF 或不变 （见注）
负数标志	N	OFF 或不变 （见注）

注 在 CS1 和 CJ1 CPU 单元中，这些标志都变 OFF。
在 CS1- H， CJ1- H， CJ1M 和 CS1D CPU 单元中，这些标志保持不变。

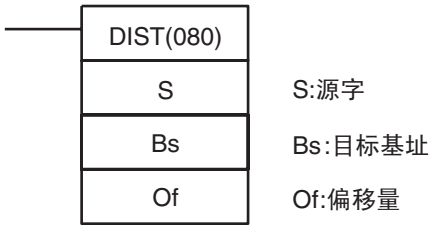
例 在下面例子中，当 CIO 000000 为 ON 时，D00100 和 D00101 的内容与 D00200 和 D00201 的内容进行交换。



3-8-12 单字分配：DIST(080)

用途 把源字传送到由基地址加偏移量计算得出的目标字中。

梯形图符号



变化

变化	ON 条件时每次循环执行	DIST(080)
	上升沿微分时执行一次	@DIST(080)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

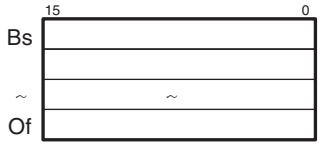
操作数

Bs: 目标基址

指定目标基址。偏移量加上这个地址得出目标字。

Of: 偏移量

这个值加上基地址得出目标字。偏移量可以是 0000 ～ FFFF（十进制：0 ～ 65535）中的任何值，但 Bs 和 Bs + Of 必须在同一数据区内。

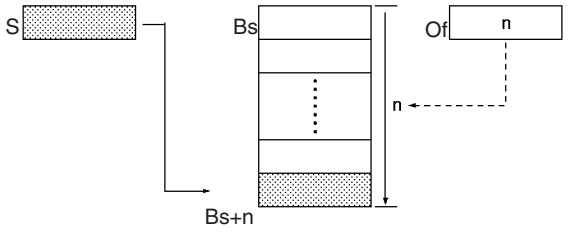


操作数规定

区域	S	Bs	Of
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959	A448 ～ A959	A000 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #FFFF（二进制）	---	#0000 ～ #FFFF（二进制）或 &0 ～ &65535
数据寄存器	DR0 ～ DR15	---	DR0 ～ DR15
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –) IR0 ～ , –(– –) IR15		

描述

DIST(080) 指令把 S 拷贝到由 Bs 加 Of 计算得出的目标字中，通过改变偏移量的值，同一个 DIST(080) 指令可用来把源数据分配到数据区的各个字中。



标志

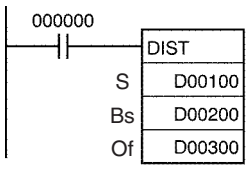
名称	标记	操作
错误标志	ER	OFF
等于标志	=	源数据是 0000 时 ON。 其它情况下 OFF。
负数标志	N	源数据最左位是 1 时 ON。 其它情况下 OFF。

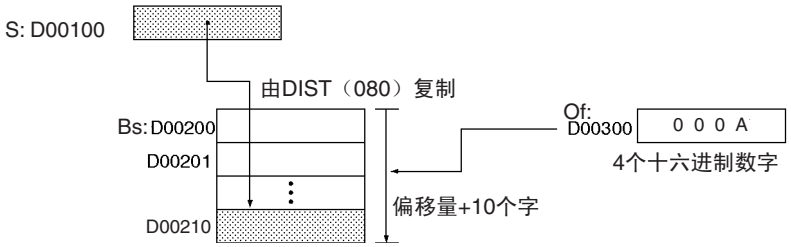
注意

必须确保偏移量不超出数据区的末端，即 Bs 和 Bs ＋ Of 必须在同一数据区内。

例

在下面例子中，当 CIO 000000 为 ON 时，如果 D00300 的内容是 10(0AH) 的话，将把 D00100 的内容拷贝到 D00210(D00200 ＋ 10) 中。通过改变 D00300 中偏移量的值，可以把 D00100 的内容拷贝到其它字中。



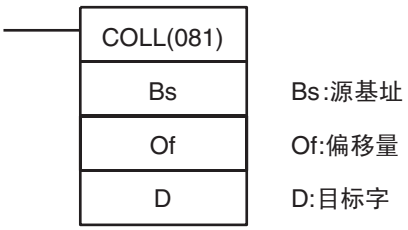


3-8-13 数据收集：COLL(081)

用途

把源字 （由基址加偏移量计算得出）传送到目标字中。

梯形图符号



变化

变化	ON 条件时每次循环执行	COLL(081)
	上升沿微分时执行一次	@COLL(081)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

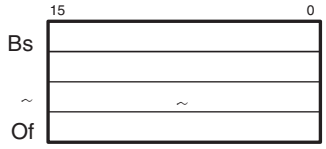
操作数

Bs: 源基址

指定源基地址。偏移量加上这个地址得出源字。

Of: 偏移量

这个值加上基地址得出源字。偏移量可以是 0000 ～ FFFF（十进制：0 ～ 65535）中的任何值，但 Bs 和 Bs + Of 必须在同一数据区内。

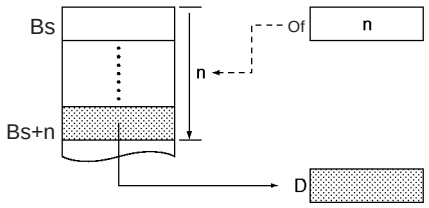


操作数规定

区域	Bs	Of	D
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---	#0000 ～ #FFFF（二进制）或 &0 ～ &65535	---
数据寄存器	---	DR0 ～ DR15	
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –) IR0 ～ , –(– –) IR15		

描述

COLL(081) 指令把源字（由 Bs 加 Of 计算得出）拷贝到目标字中，通过改变偏移量 Of 的值，同一个 COLL(081) 指令可用来从数据区内的各个源字中收集数据。



标志

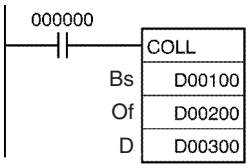
名称	标记	操作
错误标志	ER	OFF
等于标志	=	源数据是 0000 时 ON。 其它情况下 OFF。
负数标志	N	源数据最左位是 1 时 ON。 其它情况下 OFF。

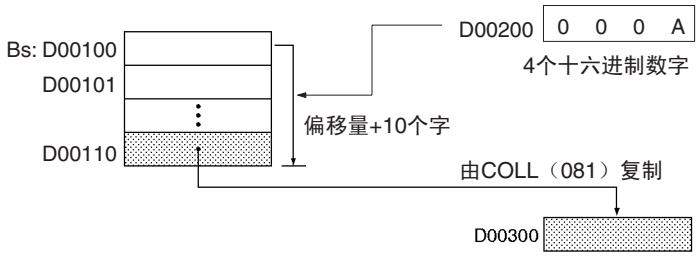
注意

必须确保偏移量不超出数据区的末端，即 Bs 和 Bs ＋ Of 必须在同一数据区内。

例

在下面例子中，当 CIO 000000 为 ON 时，如果 D00200 的内容是 10(0AH) 的话，将把 D00110(D00100 ＋ 10) 的内容拷贝到 D00300 中。通过改变 D00200 中偏移量的值，可以把其它字的内容拷贝到 D00300 中。





3-8-14 传送至寄存器：MOVR(560)

用途

在指定的索引寄存器中设置指定字，位，或定时器 / 计数器完成标志的 PLC 存储地址。（使用 MOVRW(561) 指令在索引寄存器中设置定时器 / 计数器 PV 的 PLC 存储地址）。

梯形图符号

变化

变化	ON 条件时每次循环执行	MOVR(560)
	上升沿微分时执行一次	@MOVR(560)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数

D: 目标

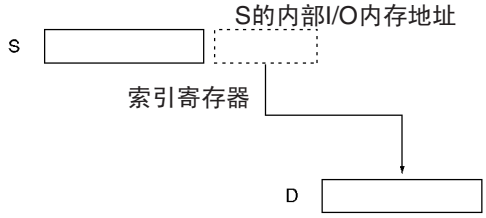
目标必须是一个索引寄存器 (IR0 ～ IR15)。

操作数规定

区域	S	D
CIO 区	CIO 0000 ～ CIO 6143 CIO 000000 ～ CIO 614315	---
工作区	W000 ～ W511 W00000 ～ W51115	---
保持位区	H000 ～ H511 H00000 ～ H51115	---
辅助位区	A000 ～ A447 A448 ～ A959 A00000 ～ A44715 A44800 ～ A95915	---
定时器区	T0000 ～ T4095 （完成标志）	---
计数器区	C0000 ～ C4095 （完成标志）	---
任务标志	TK0000 ～ TK0031	---
DM 区	D00000 ～ D32767	---
无区号 EM 区	E00000 ～ E32767	---
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	---
二进制间接 DM/EM 地址	---	
BCD 间接 DM/EM 地址	---	
常数	---	
数据寄存器	---	
索引寄存器	---	IR0 ～ IR15
使用索引寄存器间接寻址	---	

描述

MOVR(560) 指令找出 S 的 PLC 存储地址 （绝对地址），并把这个地址写入 D（一个索引寄存器）中。



如果在 S 中指定了一个定时器 / 计数器，MOVR(560) 指令会把定时器 / 计数器完成标志的 PLC 存储地址写入 D 中，使用 MOVRW(561) 指令把定时器 / 计数器 PV 的 PLC 存储地址写入 D 中。

标志

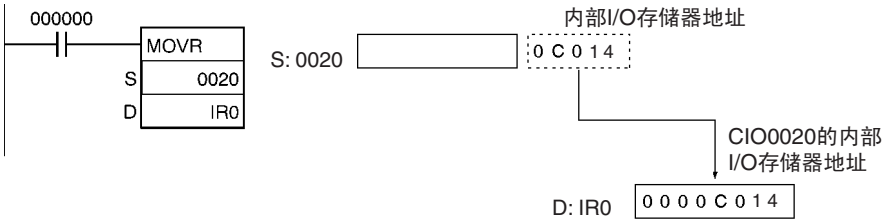
名称	标记	操作
错误标志	ER	OFF 或不变 （见注）
等于标志	=	OFF 或不变 （见注）
负数标志	N	OFF 或不变 （见注）

注 在 CS1 和 CJ1 CPU 单元中，这些标志都变为 OFF。
在 CS1- H， CJ1-H， CJ1M，以及 CS1D CPU 单元中，这些标志保持不变。

注意 MOVR(560) 指令不能用来设置定时器 / 计数器 PV 的 PLC 存储地址，而应用 MOVRW(561) 指令来设置定时器 / 计数器 PV 的 PLC 存储地址。
中断任务中的索引寄存器的内容在它被设置前，事先是不知道的。所以在中断任务中使用索引寄存器前，一定要用 MOVR(560) 指令来设置寄存器。

在中断任务中对 IR 或 DR 中的内容所作的任何修改不会影响循环任务中寄存器的内容。

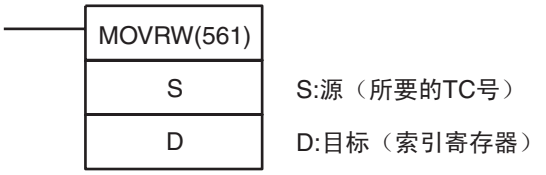
例 在下面例子中，当 CIO 000000 为 ON 时， MOVR(560) 指令把 CIO 0020 的 PLC 存储地址写入 IR0 中。



3-8-15 传送定时器 / 计数器 PV 至寄存器：MOVRW(561)

用途 在指定的索引寄存器中设置指定定时器 / 计数器 PV 值的 PLC 存储地址。（使用 MOVR(560) 指令在索引寄存器中设置字，位，或定时器 / 计数器完成标志的 PLC 存储地址）。

梯形图符号



变化

变化	ON 条件时每次循环执行	MOVR(561)
	上升沿微分时执行一次	@MOVR(561)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

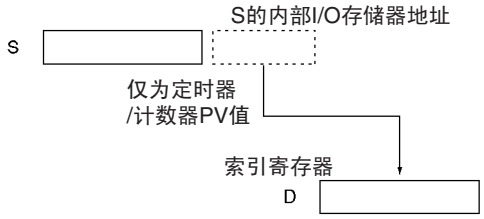
操作数 D: 目标
目标必须是一个索引寄存器 (IR0 ～ IR15)。

操作数规定

区域	S	D
CIO 区	---	
工作区	---	
保持位区	---	
辅助位区	---	
定时器区	T0000 ～ T4095 （当前值）	---
计数器区	C0000 ～ C4095 （当前值）	---
DM 区	---	
无区号 EM 区	---	
有区号 EM 区	---	
二进制间接 DM/EM 地址	---	
BCD 间接 DM/EM 地址	---	
常数	---	
数据寄存器	---	
索引寄存器	---	IR0 ～ IR15
使用索引寄存器间接寻址	---	

描述

MOVRW(561) 指令找出在 S 中指定的定时器 / 计数器 PV 值的 PLC 存储地址，并把这个地址写入 D （一个索引寄存器）中。



MOVRW(561) 指令会把定时器 / 计数器 PV 值的 PLC 存储地址设置到 D 中，使用 MOVR(560) 指令设置定时器 / 计数器完成标志的 PLC 存储地址。

标志

名称	标记	操作
错误标志	ER	OFF 或不变 （见注）
等于标志	=	OFF 或不变 （见注）
负数标志	N	OFF 或不变 （见注）

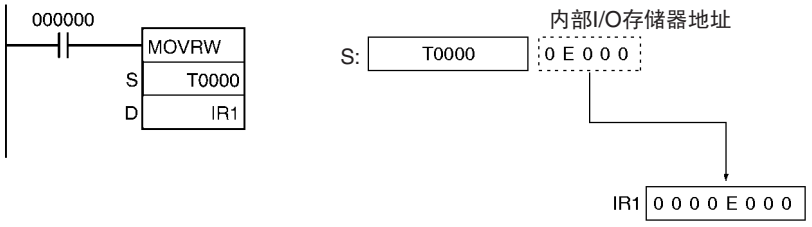
注 在 CS1 和 CJ1 CPU 单元中，这些标志都变 OFF。
在 CS1- H， CJ1- H， CJ1M，以及 CS1D CPU 单元中，这些标志保持不变。

注意

MOVRW(561) 指令不能用来设置数据区的字，位或定时器 / 计数器完成标志的 PLC 存储地址，而应使用 MOVR(560) 指令来设置这些 PLC 存储地址。

例

在下面例子中，当 CIO 000000 为 ON 时，MOVRW(561) 指令把定时器 T0000 PV 值的 PLC 存储地址写入 IR1 中。



3-9 数据移位指令

这一小节介绍用于在字内或字间，以不同方向，不同数量数据的移动指令。

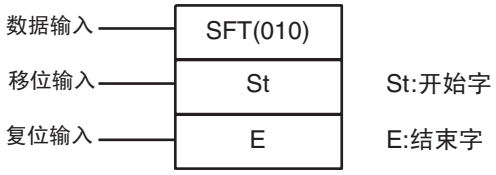
指令	助记符	功能代码	页数
移位寄存器	SFT	010	309
可逆移位寄存器	SFTR	084	310
异步移位寄存器	ASFT	017	313
字移位	WSFT	016	316
算术左移	ASL	025	318
双字左移	ASLL	570	319
算术右移	ASR	026	321
双字右移	ASRL	571	322
循环左移	ROL	027	324
双字循环左移	ROLL	572	326
无进位循环左移	RLNC	574	331
无进位双字循环左移	RLNL	576	332
循环右移	ROR	028	328
双字循环右移	RORL	573	329
无进位循环右移	RRNC	575	335
无进位双字循环右移	RRNL	577	336
单数字左移	SLD	074	338
单数字右移	SRD	075	339
N 位数据左移	NSFL	578	341
N 位数据右移	NSFR	579	343
N 位左移	NASL	580	345
双字 N 位左移	NSLL	582	348
N 位右移	NASR	581	350
双字 N 位右移	NSRL	583	353

3-9-1 移位寄存器：SFT(010)

用途

操作一个移位寄存器。

梯形图符号



变化

变化	ON 条件时每次循环执行	SFT(010)
	上升沿微分时执行一次	不支持
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
不允许	OK	OK	OK

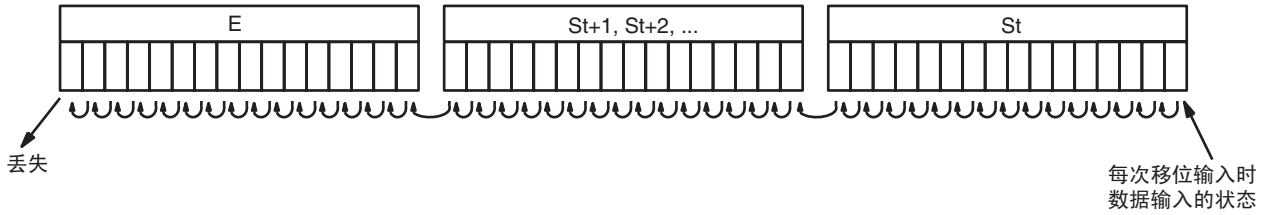
注 St 和 E 必须在同一数据区内。

操作数规定

区域	St	E
CIO 区	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W511	
保持位区	H000 ～ H511	
辅助位区	A448 ～ A959	
定时器区	---	
计数器区	---	
DM 区	---	
无区号 EM 区	---	
有区号 EM 区	---	
二进制间接 DM/EM 地址	---	
BCD 间接 DM/EM 地址	---	
常数	---	
数据寄存器	---	
索引寄存器	---	
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15	

描述

当移位输入的执行条件由 OFF 变为 ON 时， St ～ E 的所有数据左移一位（从最右边到最左边），并且把数据输入的 ON/OFF 状态放在最右边位。



标志

名称	标记	操作
错误标志	ER	St 和 E 的间接 IR 地址不在 CIO， AR， HR，或 WR 数据区内时 ON。 其它情况下 OFF。

注意

被移出移位寄存器的位数据将丢失。

当复位输入变 ON 时，移位寄存器中从最右边目标字 (St) 到最左边目标字 (E) 中的所有位将被复位 （即被置 0）。复位输入优先与其它输入。

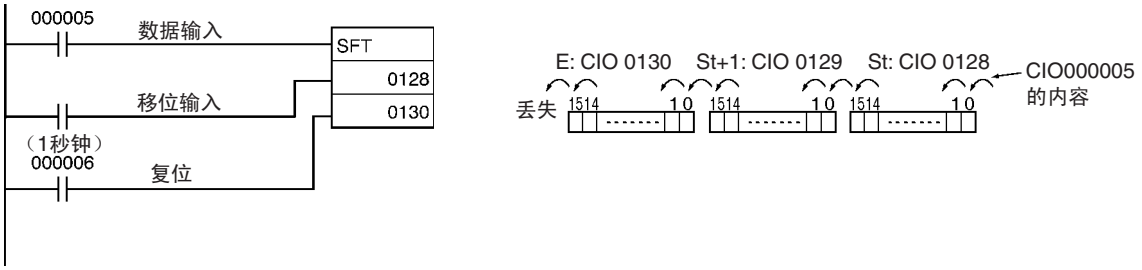
St 必须小于等于 E，但是即使 St 被设成大于 E 也不会发生错误，并且 St 中数据的某个字将被移位。

当 St 和 E 被设计为间接使用索引寄存器，而 I/O 内存的实际地址不在数据存储区内时，将发生错误，并且错误标志变 ON。

例

超出 16 位的移位寄存器

下面例子显示了一个使用字 CIO 0128 ～ CIO 0130 的 48 位移位寄存器。它使用了 1S 的时钟脉冲，这样每隔一秒钟，就把 CIO 000005 产生的执行条件移入 CIO 012800 和 CIO 013015 之间的 3 字寄存器中。

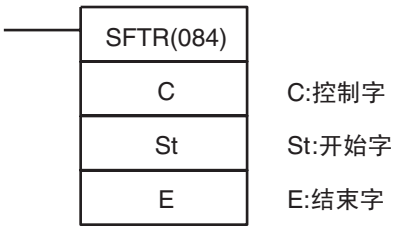


3-9-2 可逆移位寄存器：SFTR(084)

用途

产生一个可向右也可向左移动数据的移位寄存器。

梯形图符号



变化

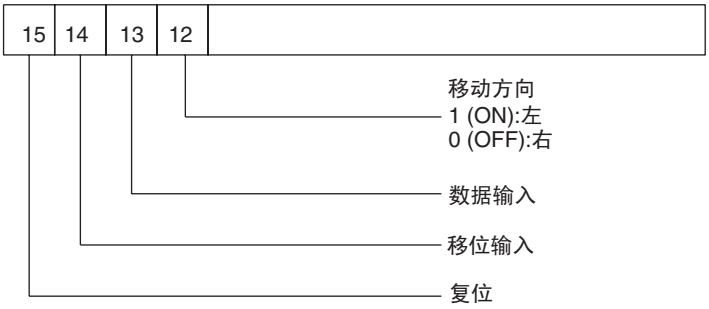
变化	ON 条件时每次循环执行	SFTR(084)
	上升沿微分时执行一次	@SFTR(084)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数

C: 控制字



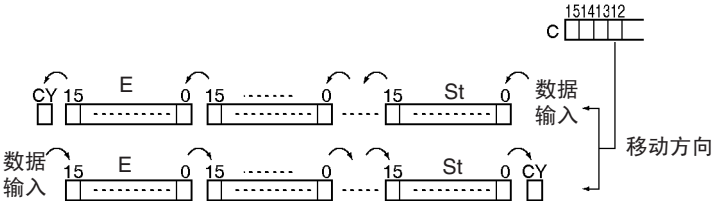
注 St 和 E 必须在同一数据区内。

操作数规定

区域	C	St	E
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959	A448 ～ A959	
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---		
数据寄存器	DR0 ～ DR15	---	
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

描述

当移位输入位 （C 中的位 14）的执行条件变为 ON 时， St ～ E 的所有数据按指定的移动方向（由 C 中的位 12 指定）移动一位，并且把数据输入的 ON/OFF 状态放在最右或最左位，而把被移出移位寄存器的位数据放在进位标志 (CY) 中。



标志

名称	标记	操作
错误标志	ER	St ＞ E 时 ON。 其它情况下 OFF。
进位标志	CY	移入 1 时 ON。 移入 0 时 OFF。 复位位被置 1 时 OFF。

注意

当复位位 （C 中的位 15）被置为 OFF 时，就可使用上述的移位操作。

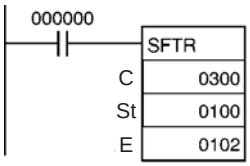
当复位位 （C 中的位 15）变 ON 时，移位寄存器中 St ～ E 之间的所有位将被复位 （即被置 0）。

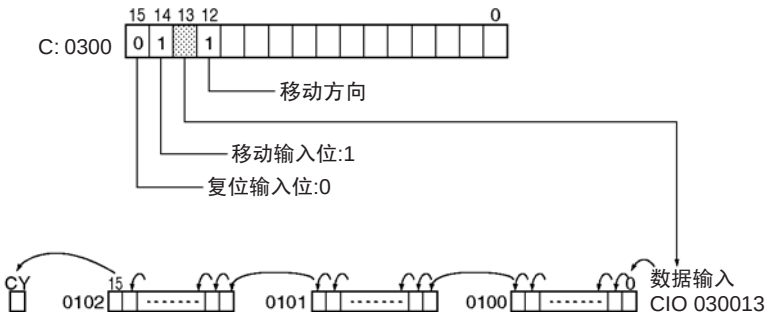
当 St 大于 E 时，产生一个错误，并且错误标志变 ON。

例

<u>移位数据</u>

当 CIO 000000 为 ON 时，如果移位输入位 CIO 030014 为 ON，并且复位位 CIO 030015 为 OFF，字 CIO 0100 ～ CIO 0102 会按 CIO 030012 指定的方向（例：1 表示向右）移动一位，并且输入位 CIO 030013 的内容移到最右位 CIO 010000 中。 CIO 010015 的内容移到进位标志 (CY) 中。





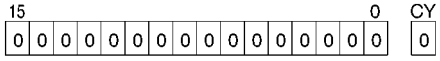
<u>复位数据</u>

当 CIO 000000 为 ON 时，如果 CIO 030014 为 ON，并且复位位 CIO 030015 为 ON，字 CIO 0100 ～ CIO 0102 和进位标志被复位为 OFF。

控制数据

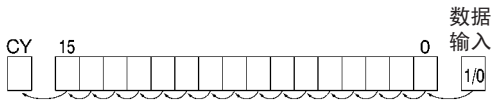
复位数据

St ～ E 中的所有位和进位标志被置 0，并且当复位位 CIO 030015 为 ON 时，不接受其它数据。



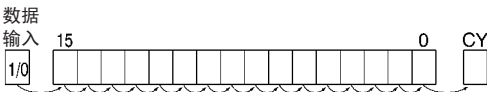
左移数据 （从最右位到最左位）

当移位输入位 （C 中的位 14）为 ON 时，输入位 （C 中的位 13）的内容被移到开始字的位 00 中，其后的每一位被左移一位，结束字中位 15 的状态被移到进位标志中。



右移数据 （从最左位到最右位）

当移位输入位 （C 中的位 14）为 ON 时，输入位 （C 中的位 13）的内容被移到结束字的位 15 中，其后的每一位被右移一位，开始字中位 00 的状态被移到进位标志中。

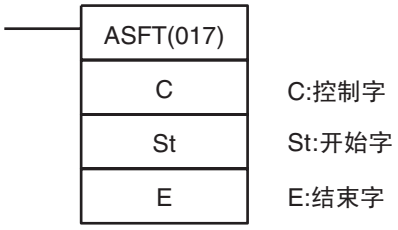


3-9-3 异步移位寄存器：ASFT(017)

用途

在指定字范围内，或向 St 方向或向 E 方向移动所有非零字数据，以替换 0000H 字数据。

梯形图符号



变化

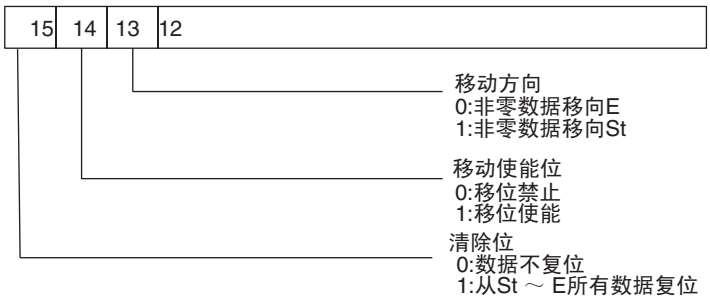
变化	ON 条件时每次循环执行	ASFT(017)
	上升沿微分时执行一次	@ASFT(017)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

C: 控制字



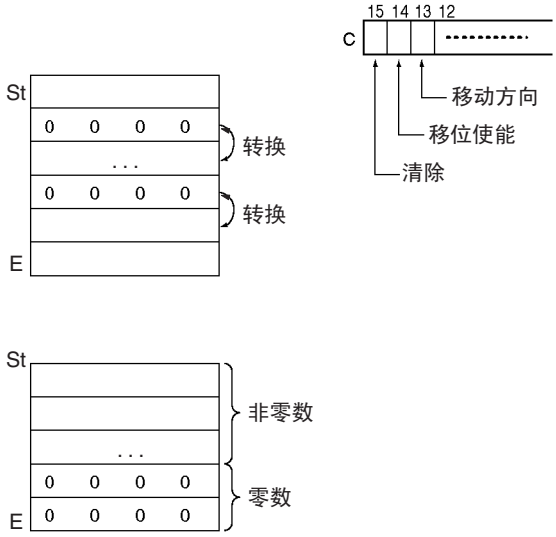
注 St 和 E 必须在同一数据区内。

操作数规定

区域	C	St	E
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959	A448 ～ A959	
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---		
数据寄存器	DR0 ～ DR15	---	
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

描述

当移位使能位（C 中的位 14）为 ON 时，只要移动方向上包含全为零的字，字 St ～ E 范围中内容不为零的所有字将按移动方向位（C 中的位 13）指定的方向移动一个字。如果 ASFT(017) 指令被重复足够的次数，所有全零的字会被非零的字替换，这将导致 St ～ E 之间的数据被分成零数据和非零数据。



标志

名称	标记	操作
错误标志	ER	St ＞ E 时 ON。 其它情况下 OFF。

注意

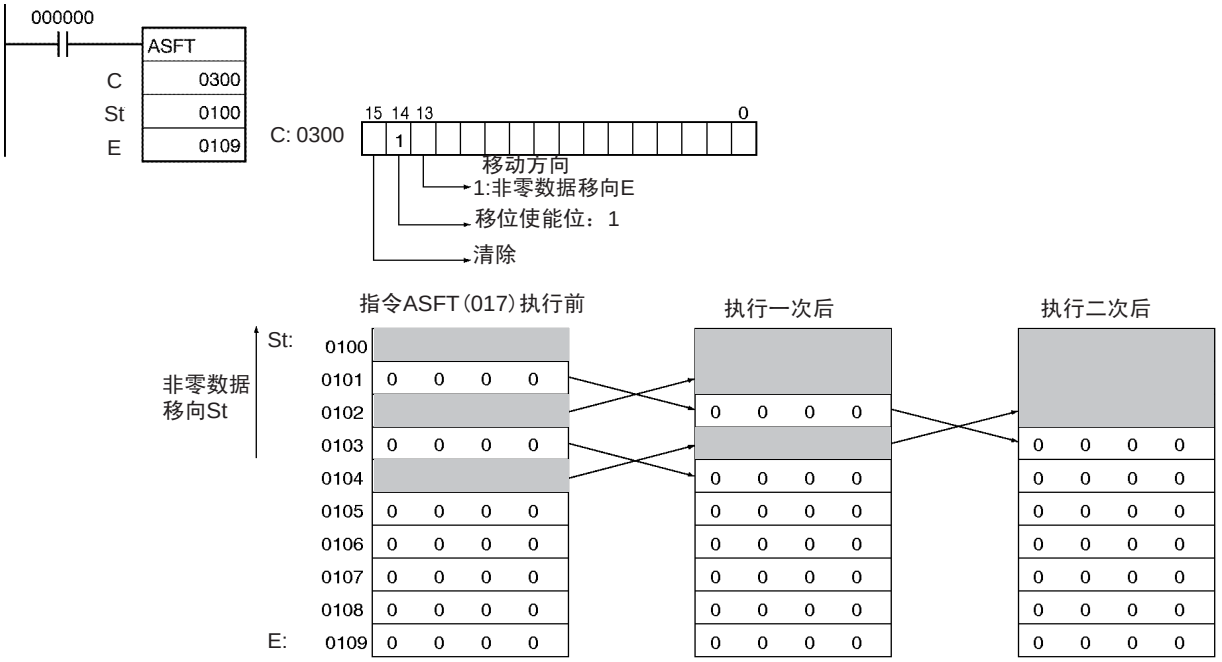
当清除标志 （C 中的位 15）变 ON 时，移位寄存器中 St ～ E 之间的所有位都被复位（即被置 0）。清除标志比移位使能位（C 中的位 14）有更高的优先级。

当 St 大于 E 时，会发生错误，并且错误标志变 ON。

例

移位数据：

当 CIO 000000 为 ON 时，如果移位使能位 CIO 030014 为 ON，并且在非零数据左边的字全部为零，CIO 0100 ～ CIO 0109 之间内容为非零数据的所有字将按移位方向位 CIO 030013 设置的方向 （例：1 表示向 St 方向）移动。

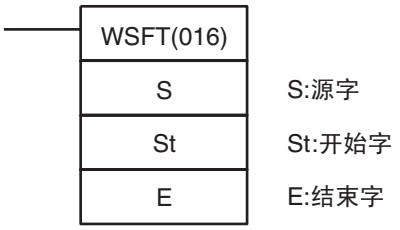


3-9-4 字移位：WSFT(016)

用途

以字为单位在 St 和 E 之间移动数据。

梯形图符号



变化

变化	ON 条件时每次循环执行	WSFT(016)
	上升沿微分时执行一次	@WSFT(016)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

注 St 和 E 必须在同一数据区内。

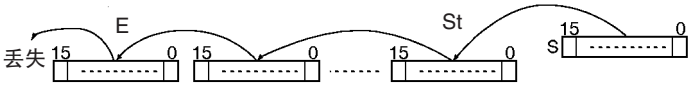
操作数规定

区域	S	St	E
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959	A448 ～ A959	
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #FFFF （二进制）	---	
数据寄存器	DR0 ～ DR15	---	

区域	S	St	E
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

描述

WSFT(016) 指令以字为单位按 St 到 E 方向移动数据，源字 S 的数据放在 St 中， E 中的内容将丢失。



标志

名称	标记	操作
错误标志	ER	St ＞ E 时 ON。 其它情况下 OFF。

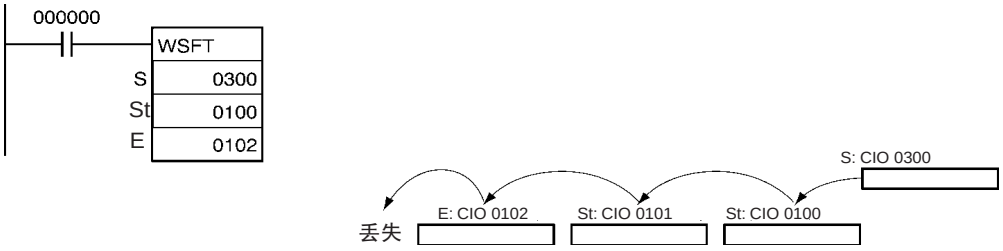
注意

当 St 大于 E 时，会发生错误，并且错误标志变 ON。

注 当移动大量数据时，指令执行时间相当长。必须确保在 WSFT(016) 指令执行时不切断电源，以防移动操作中途停止。

例

当 CIO 000000 为 ON 时，CIO 0100 ～ CIO 0102 之间的数据向 E 方向移动一个字。把 CIO 0300 的内容保存在 CIO 0100 中，丢弃 CIO 0102 的内容。

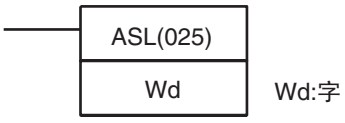


3-9-5 算术左移：ASL(025)

用途

Wd 的内容左移一位。

梯形图符号



变化

变化		
	ON 条件时每次循环执行	ASL(025)
	上升沿微分时执行一次	@ASL(025)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

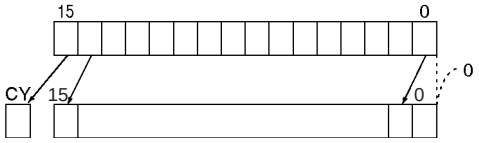
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	Wd
CIO 区	CIO 0000 ～ CIO 6143
工作区	W000 ～ W511
保持位区	H000 ～ H511
辅助位区	A448 ～ A959
定时器区	T0000 ～ T4095
计数器区	C0000 ～ C4095
DM 区	D00000 ～ D32767
无区号 EM 区	E00000 ～ E32767
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767
常数	---
数据寄存器	DR0 ～ DR15
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15

描述

ASL(025) 指令把 Wd 的内容左移一位（从最右位到最左位）。在最右位中放置“0”，并且把最左位的数据移到进位标志 (CY) 中。



标志

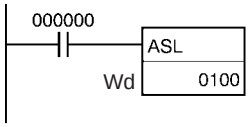
名称	标记	操作
错误标志	ER	OFF
等于标志	=	移位结果为 0 时 ON。 其它情况下 OFF。
进位标志	CY	把 1 移入进位标志 (CY) 中时 ON。 其它情况下 OFF。
负数标志	N	移位结果最左位为 1 时 ON。 其它情况下 OFF。

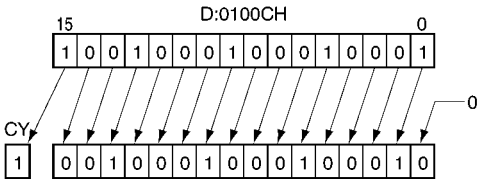
注意

当执行 ASL(025) 指令时，错误标志变 OFF。
移动 Wd 内容的结果是 0 时，等于标志变 ON。
移动 Wd 内容的最左位的结果是 1 时，负数标志变 ON。

例

当 CIO 000000 为 ON 时，CIO 0100 将左移一位。在 CIO 010000 中放置“0”，CIO 010015 的内容移到进位标志 (CY) 中。



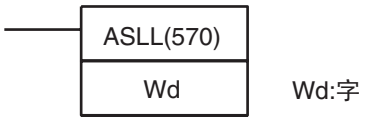


3-9-6 双字左移：ASLL(570)

用途

Wd 和 Wd ＋ 1 的内容左移一位。

梯形图符号



变化

变化	ON 条件时每次循环执行	ASLL(570)
	上升沿微分时执行一次	@ASLL(570)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

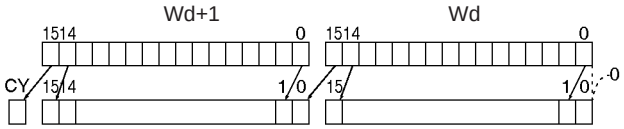
操作数规定

区域	Wd
CIO 区	CIO 0000 ～ CIO 6142
工作区	W000 ～ W510
保持位区	H000 ～ H510
辅助位区	A448 ～ A958
定时器区	T0000 ～ T4094
计数器区	C0000 ～ C4094
DM 区	D00000 ～ D32766
无区号 EM 区	E00000 ～ E32766
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)

区域	Wd
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---
数据寄存器	---
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

ASLL(570) 指令把 Wd 和 Wd ＋ 1 的内容左移一位 （从最右位到最左位）。在 Wd 最右位中放置“0”，并且把 Wd ＋ 1 的最左位内容移到进位标志 (CY) 中。



标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	移位结果为 0 时 ON。 其它情况下 OFF。
进位标志	CY	把 1 移入进位标志 (CY) 中时 ON。 其它情况下 OFF。
负数标志	N	移位结果最左位为 1 时 ON。 其它情况下 OFF。

注意

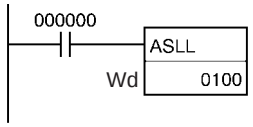
当执行 ASLL(570) 指令时，错误标志变 OFF。

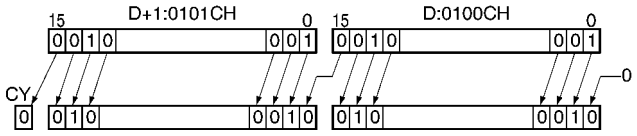
移动 Wd 和 Wd ＋ 1 内容的结果是 0 时，等于标志变 ON。

移动 Wd ＋ 1 内容的最左位的结果是 1 时，负数标志变 ON。

例

当 CIO 000000 为 ON 时，字 CIO 0100 和字 CIO 0101 将左移一位。在 CIO 010000 中放放入 “0”， CIO 010015 的内容移到进位标志 (CY) 中。



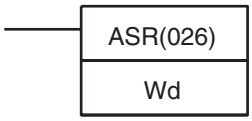


3-9-7 算术右移：ASR(026)

用途

Wd 的内容向右移 1 位。

梯形图符号



Wd:字

变化

变化	ON 条件时每次循环执行	ASR(026)
	上升沿微分时执行一次	@ASR(026)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

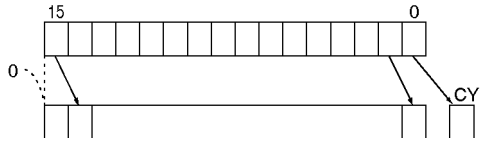
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	Wd
CIO 区	CIO 0000 ～ CIO 6143
工作区	W000 ～ W511
保持位区	H000 ～ H511
辅助位区	A448 ～ A959
定时器区	T0000 ～ T4095
计数器区	C0000 ～ C4095
DM 区	D00000 ～ D32767
无区号 DM 区	E00000 ～ E32767
有区号 DM 区	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---
数据寄存器	DR0 ～ DR15
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

ASR(026) 把的内容向右移动 1 位，（从最左位向最右位）。“0”被放置到最左位，并且最右位的内容会被移入进入位标志 (CY)。



标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	移动结果为 0 时 ON。 其它情况下 OFF。
进位标志	CY	1 被移入进位标志 (CY) 时 ON。 其它情况下 OFF。
负数标志	N	OFF

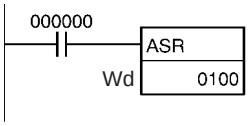
注意

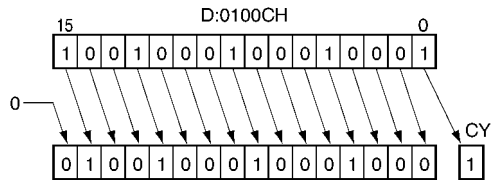
ASR(026) 执行后，错误标志和负标志会变为 OFF。

移位结果 Wd 的内容为 0 时，等于标志会变为 ON。

例

当 CIO 000000 为 ON 后，字 CIO 0100 会向右移 1 位，“0”被放置到 CIO 010015 中，并且 CIO 010000 的内容会移入进位标志 (CY)。



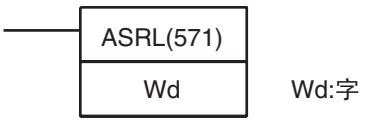


3-9-8 双字右移：ASRL(571)

用途

Wd 和 Wd+1 的内容向右移动 1 位。

梯形图符号



变化

变化	ON 条件时每次循环执行	ASRL(571)
	上升沿微分时执行一次	@ASRL(571)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

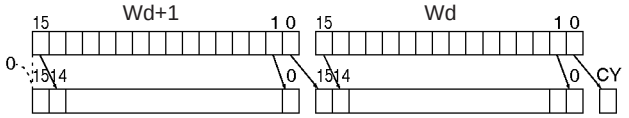
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	Wd
CIO 区	CIO 0000 ～ CIO 6142
工作区	W000 ～ W510
保持位区	H000 ～ H510
辅助位区	A448 ～ A958
定时器区	T0000 ～ T4094
计数器区	C0000 ～ C4094
DM 区	D00000 ～ D32766
无区号 DM 区	E00000 ～ E32766
有区号 DM 区	En_00000 ～ En_32766 (n = 0 ～ C)
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---
数据寄存器	---
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

ASR(571) 把 Wd 和 Wd+1 的内容向右移动 1 位 ,（从最左位向最右位）。“0”被放置到 Wd+1 的最左位，并且 Wd 的最右位内容会移入进位 (CY)。



标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	移位结果为 0 时 ON。 其它情况下 OFF。
进位标志	CY	1 被移入进位标志 (CY) 时 ON。 其它情况下 OFF。
负数标志	N	OFF

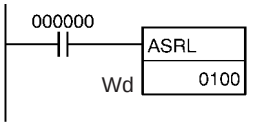
注意

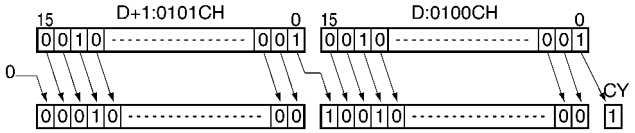
ASR(571) 执行后，错误标志和负标志会变为 OFF。

如移位结果 Wd 和 Wd+1 的内容为 0 时，等于标志会变为 ON。

例

当 CIO 000000 为 ON 后，字 CIO 0100 和 CIO 0101 会向右移 1 位， "0" 被放置到 CIO 010115 中，并且 CIO 010000 的内容会移入进位标志 (CY)。



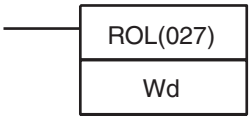


3-9-9 循环右移：ROL(027)

用途

所有 Wd 的位包括进位标志 (CY) 向左移 1 位。

梯形图符号



Wd:字

变化

变化	ON 条件时每次循环执行	ROL(027)
	上升沿微分时执行一次	@ROL(027)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

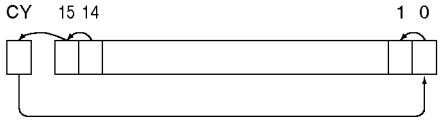
操作数规定

区域	Wd
CIO 区	CIO 0000 ～ CIO 6143
工作区	W000 ～ W511
保持位区	H000 ～ H511
辅助位区	A448 ～ A959
定时器区	T0000 ～ T4095
计数器区	C0000 ～ C4095
DM 区	D00000 ～ D32767
无区号 DM 区	E00000 ～ E32767
有区号 DM 区	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---

区域	Wd
数据寄存器	DR0 ～ DR15
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

ROL(027) 把 Wd 的所有位包括进位标志 (CY) 向左移动 1 位（从最右位向最左位）。



标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	移动结果为 0 时 ON。 其它情况下 OFF。
进位标志	CY	1 被移入进位标志 (CY) 时 ON。 其它情况下 OFF。
负数标志	N	移位结果最左位为 1 时 ON。 其它情况下 OFF。

注意

ROL(027) 执行后，错误标志会变为 OFF。

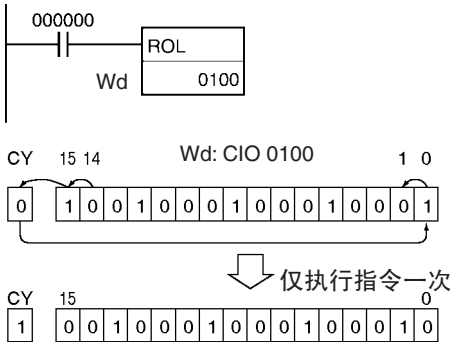
移位结果 Wd 的内容为 0 时，等于标志会变为 ON。

如果移动 Wd 内容的最左位的结果是 1 时，负数标志变 ON。

注 通过使用设置进位 (STC(040)) 或清除进位 (CLC(041)) 可以在执行这条指令前，设置进位标志内容为 1 或 0。

例

当 CIO 000000 为 ON 后，字 CIO 0100 和进位标志 (CY) 会向左移 1 位，CIO 010015 的内容会移入进位标志 (CY)，进位标志内容会移入 CIO 010000。

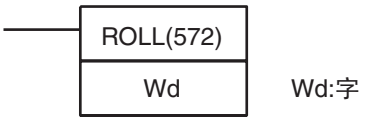


3-9-10 双字循环左移：ROLL(572)

用途

所有 Wd 和 Wd+1 的位，包括进位标志 (CY)，向左移 1 位。

梯形图符号



变化

变化	ON 条件时每次循环执行	ROLL(572)
	上升沿微分时执行一次	@ROLL(572)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

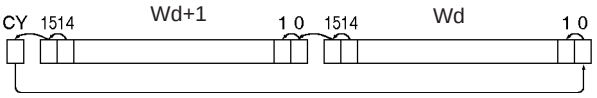
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	Wd
CIO 区	CIO 0000 ～ CIO 6142
工作区	W000 ～ W510
保持位区	H000 ～ H510
辅助位区	A448 ～ A958
定时器区	T0000 ～ T4094
计数器区	C0000 ～ C4094
DM 区	D00000 ～ D32766
无区号 DM 区	E00000 ～ E32766
有区号 DM 区	En_00000 ～ En_32766 (n = 0 ～ C)
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---
数据寄存器	---
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

ROLL(572) 向左移动 Wd 和 Wd+1 的所有位。包括进位标志 (CY)，（从最右位向最左位）。



标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	移动结果为 0 时 ON。 其它情况下 OFF。
进位标志	CY	1 被移入进位标志 (CY) 时 ON。 其它情况下 OFF。
负数标志	N	移位结果最左位为 1 时 ON。 其它情况下 OFF。

注意

ROLL(572) 执行后，错误标志会变为 OFF。

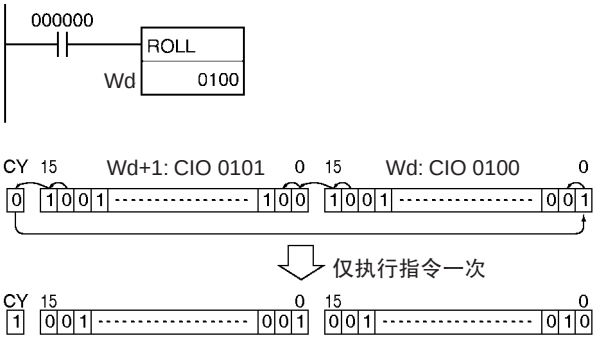
移位结果 Wd 和 Wd+1 的内容为 0 时，等于标志会变为 ON。

移位结果 Wd+1 的最左位内容为 1 时，负标志会变为 ON。

注 通过使用设置进位 (STC(040)) 或清除进位 (CLC(041)) 可以在执行这条指令前，设置进位标志内容为 1 或 0。

例

当 CIO 000000 为 ON 后，字 CIO 0100，CIO 0101 和进位标志 (CY) 会向左移 1 位，CIO 010015 的内容会移入进位标志 (CY)，进位标志内容会移入 CIO 010000。

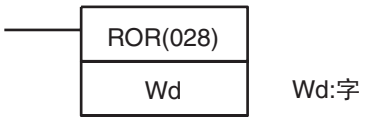


3-9-11 循环右移：ROR(028)

用途

所有 Wd 的位，包括进位标志 (CY)，向右移动 1 位。

梯形图符号



变化

变化	ON 条件时每次循环执行	ROR(028)
	上升沿微分时执行一次	@ROR(028)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

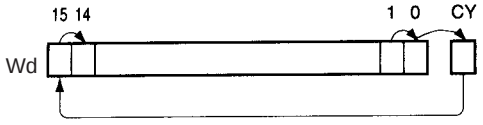
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	Wd
CIO 区	CIO 0000 ～ CIO 6143
工作区	W000 ～ W511
保持位区	H000 ～ H511
辅助位区	A448 ～ A959
定时器区	T0000 ～ T4095
计数器区	C0000 ～ C4095
DM 区	D00000 ～ D32767
无区号 DM 区	E00000 ～ E32767
有区号 DM 区	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---
数据寄存器	DR0 ～ DR15
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

ROLL(028) 向右移动 Wd 的所有位，包括进位标志 (CY)，（从最右位向最左位）。



标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	移动结果为 0 时 ON。 其它情况下 OFF。
进位标志	CY	1 被移入进位标志 (CY) 时 ON。 其它情况下 OFF。
负数标志	N	移位结果最左位为 1 时 ON。 其它情况下 OFF。

注意

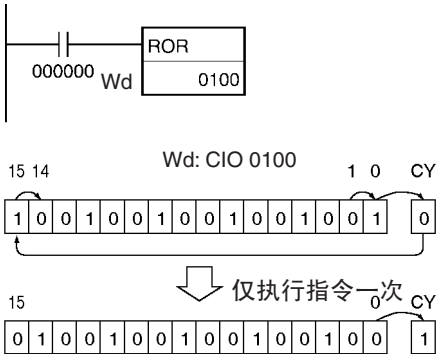
ROR(028) 执行后，错误标志会变为 OFF。

移位结果 Wd 和 Wd+1 的内容为 0 时，等于标志会变为 ON。

移位结果 Wd+1 的最左位内容为 1 时，负标志会变为 ON。

注 通过使用设置进位(STC(040))或清除进位(CLC(041))可以在执行这条指令前，设置进位标志内容为 1 或 0。

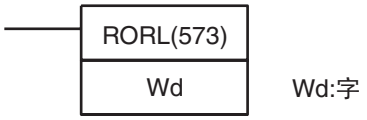
例 当 CIO 000000 为 ON 后，字 CIO 0100 和进位标志 (CY) 会向右移动 1 位，CIO 010000 的内容会移入进位标志 (CY)，进位标志内容会移入 CIO 010015。



3-9-12 双循环右移：RORL(573)

用途 所有 Wd 和 Wd+1 的位，包括进位标志 (CY)，向右移动 1 位。

梯形图符号



变化

变化	ON 条件时每次循环执行	RORL(573)
	上升沿微分时执行一次	@RORL(573)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

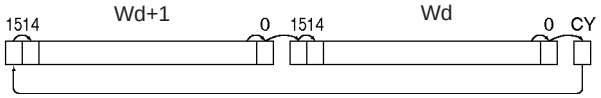
操作数规定

区域	Wd
CIO 区	CIO 0000 ～ CIO 6142
工作区	W000 ～ W510
保持位区	H000 ～ H510
辅助位区	A448 ～ A958
定时器区	T0000 ～ T4094
计数器区	C0000 ～ C4094
DM 区	D00000 ～ D32766
无区号 DM 区	E00000 ～ E32766
有区号 DM 区	En_00000 ～ En_32766 (n = 0 ～ C)
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)

区域	Wd
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---
数据寄存器	---
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

RORL(573) 移动 Wd 和 Wd+1 的所有位，包括进位标志 (CY)，向右移一位（从最左位向最右位）。



标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	移动结果为 0 时 ON。 其它情况下 OFF。
进位标志	CY	1 被移入进位标志 (CY) 时 ON。 其它情况下 OFF。
负数标志	N	移位结果最左位为 1 时 ON。 其它情况下 OFF。

注意

RORL(573) 执行后，错误标志会变为 OFF。

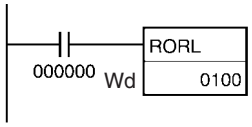
移位结果 Wd 和 Wd+1 的内容为 0 时，等于标志会变为 ON。

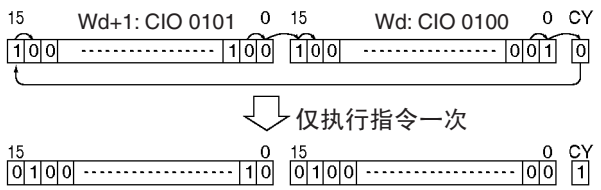
移位结果 Wd+1 的最左位内容为 1 时，负标志会变为 ON。

注 通过使用设置进位 (STC(040)) 或清除进位 (CLC(041)) 可以在执行这条指令前，设置进位标志内容为 1 或 0。

例

当 CIO 000000 为 ON 后，字 CIO 0100，CIO 0101 和进位标志 (CY) 会向右移动 1 位，CIO 010000 的内容会移入进位标志 (CY)，进位标志内容会移入 CIO 010115。



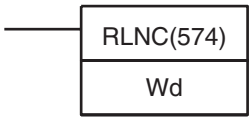


3-9-13 无进位循环左移：RLNC(574)

用途

所有 Wd 的位，不包括进位标志 (CY)，向左移动 1 位。

梯形图符号



变化

变化	ON 条件时每次循环执行	RLNC(574)
	上升沿微分时执行一次	@RLNC(574)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

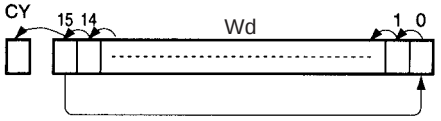
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	Wd
CIO 区	CIO 0000 ～ CIO 6143
工作区	W000 ～ W511
保持位区	H000 ～ H511
辅助位区	A448 ～ A959
定时器区	T0000 ～ T4095
计数器区	C0000 ～ C4095
DM 区	D00000 ～ D32767
无区号 DM 区	E00000 ～ E32767
有区号 DM 区	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---
数据寄存器	DR0 ～ DR15
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

RLNC(574) 把 Wd 的所有位向左移动，（从最右位向最左位）。
Wd 最左位的内容移入最右位和进位标志 (CY) 。



标志

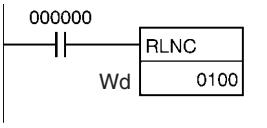
名称	标记	操作
错误标志	ER	OFF
等于标志	=	移动结果为 0 时 ON。 其它情况下 OFF。
进位标志	CY	1 被移入进位标志 (CY) 时 ON。 其它情况下 OFF。
负数标志	N	移位结果最左位为 1 时 ON。 其它情况下 OFF。

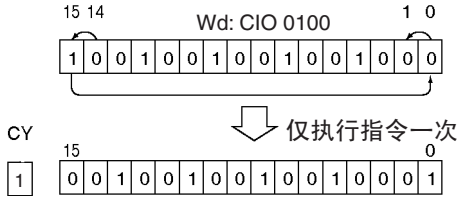
注意

RLNC(574) 执行后，错误标志会变为 OFF。
移位结果 Wd 的内容为 0 时，等于标志会变为 ON。
移位结果 Wd 的最左位内容为 1 时，负标志会变为 ON。

例

当 CIO 000000 为 ON 后，字 CIO 0100 会向左移动 1 位，（不包括进位标志 (CY)）。 CIO 010015 的内容会移入 CIO 010000。



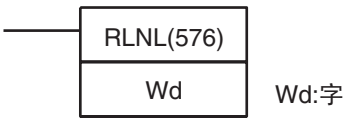


3-9-14 无进位双循环左移：RLNL(576)

用途

所有 Wd 和 Wd+1 的位，不包括进位标志 (CY)，向左移动 1 位。

梯形图符号



变化

变化	ON 条件时每次循环执行	RLNL(576)
	上升沿微分时执行一次	@RLNL(576)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

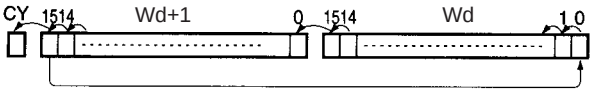
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	Wd
CIO 区	CIO 0000 ～ CIO 6142
工作区	W000 ～ W510
保持位区	H000 ～ H510
辅助位区	A448 ～ A958
定时器区	T0000 ～ T4094
计数器区	C0000 ～ C4094
DM 区	D00000 ～ D32766
无区号 DM 区	E00000 ～ E32766
有区号 DM 区	En_00000 ～ En_32766 (n = 0 ～ C)
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---
数据寄存器	---
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

RLNL(576) 向左移动 Wd 和 Wd+1 的所有位（从最右位向最左位）。Wd+1 最左位内容移入 Wd 的最右位和进位标志 (CY)。



标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	移动结果为 0 时 ON。 其它情况下 OFF。
进位标志	CY	1 被移入进位标志 (CY) 时 ON。 其它情况下 OFF。
负数标志	N	移位结果最左位为 1 时 ON。 其它情况下 OFF。

注意

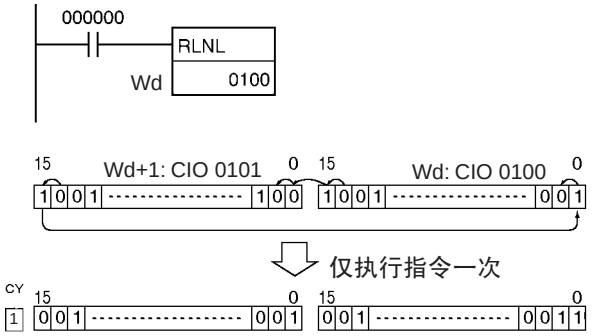
RLNL(576) 执行后，错误标志会变为 OFF。

移位结果 Wd 和 Wd+1 的内容为 0 时，等于标志会变为 ON。

移位结果 Wd+1 的最左位内容为 1 时，负标志会变为 ON。

例

当 CIO 000000 为 ON 后，字 CIO 0100， CIO 0101 会向左移动 1 位，（不包括进位标志 (CY)）。 CIO 010115 的内容会移入 CIO 010000。

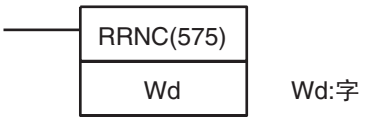


3-9-15 无进位循环右移：RRNC(575)

用途

所有 Wd 的位，不包括进位标志 (CY)，向右移动 1 位， Wd 最右位内容移入最左位和进位标志 (CY)。

梯形图符号



变化

变化	ON 条件时每次循环执行	RRNC(575)
	上升沿微分时执行一次	@RRNC(575)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

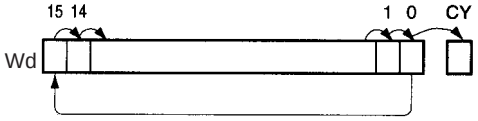
操作数规定

区域	Wd
CIO 区	CIO 0000 ～ CIO 6143
工作区	W000 ～ W511
保持位区	H000 ～ H511
辅助位区	A448 ～ A959
定时器区	T0000 ～ T4095
计数器区	C0000 ～ C4095
DM 区	D00000 ～ D32767
无区号 DM 区	E00000 ～ E32767
有区号 DM 区	En_00000 ～ En_32767 (n = 0 ～ C)

区域	Wd
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---
数据寄存器	DR0 ～ DR15
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

RRNC(575) 把 Wd 的所有位向右移动，不包括进位标志 (CY)，（从最右位向最左位）。



标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	移动结果为 0 时 ON。 其它情况下 OFF。
进位标志	CY	1 被移入进位标志 (CY) 时 ON。 其它情况下 OFF。
负数标志	N	移位结果最左位为 1 时 ON。 其它情况下 OFF。

注意

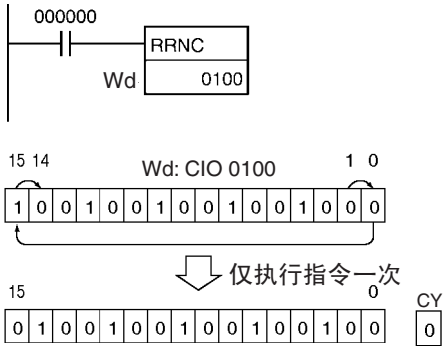
RRNC(575) 执行后，错误标志会变为 OFF。

移位结果 Wd 的内容为 0 时，等于标志会变为 ON。

移位结果 Wd 的最左位内容为 1 时，负标志会变为 ON。

例

当 CIO 000000 为 ON 后，字 CIO 0100 会向右移动 1 位，（不包括进位标志 (CY)）。 CIO 010000 的内容会移入 CIO 010015。

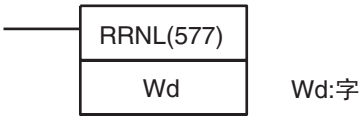


3-9-16 无进位双循环右移：RLNL(577)

用途

所有 Wd 和 Wd+1 的位，不包括进位标志 (CY)，向右移动 1 位。Wd 最右位内容移入 Wd+1 的最左位和进位标志 (CY)。

梯形图符号



变化

变化	ON 条件时每次循环执行	RRNL(577)
	上升沿微分时执行一次	@RRNL(577)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

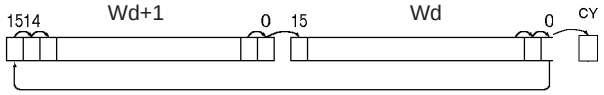
操作数规定

区域	Wd
CIO 区	CIO 0000 ～ CIO 6142
工作区	W000 ～ W510
保持位区	H000 ～ H510
辅助位区	A448 ～ A958
定时器区	T0000 ～ T4094
计数器区	C0000 ～ C4094
DM 区	D00000 ～ D32766
无区号 DM 区	E00000 ～ E32766
有区号 DM 区	En_00000 ～ En_32766 (n = 0 ～ C)
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)

区域	Wd
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---
数据寄存器	---
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

RRNL(577) 向右移动 Wd 和 Wd+1 的所有位（从最左位向最右位）。不包括进位标志 (CY)。



标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	移动结果为 0 时 ON。 其它情况下 OFF。
进位标志	CY	1 被移入进位标志 (CY) 时 ON。 其它情况下 OFF。
负数标志	N	移位结果最左位为 1 时 ON。 其它情况下 OFF。

注意

RRNL(577) 执行后，错误标志会变为 OFF。

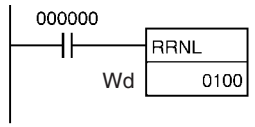
移位结果 Wd 和 Wd+1 的内容为 0 时，等于标志会变为 ON。

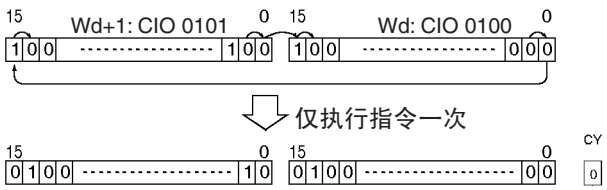
移位结果 Wd+1 的最左位内容为 1 时，负标志会变为 ON。

注 通过使用设置进位 (STC(040)) 或清除进位 (CLC(041)) 可以在执行这条指令前，设置进位标志内容为 1 或 0。

例

当 CIO 000000 为 ON 后，字 CIO 0100， CIO 0101 会向右移动 1 位，（不包括进位标志 (CY)）， CIO 010000 的内容会移入 CIO 010115。



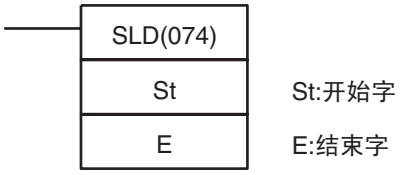


3-9-17 一个数字左移：SLD(074)

用途

向左移动一个数字 （4 位）的数据。

梯形图符号



变化

变化	ON 条件时每次循环执行	SLD(074)
	上升沿微分时执行一次	@SLD(074)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

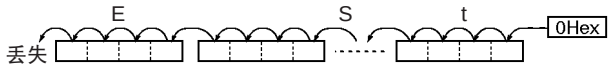
注 St 和 E 必须在同一数据区内。

操作数规定

区域	St	E
CIO 区	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W511	
保持位区	H000 ～ H511	
辅助位区	A448 ～ A959	
定时器区	T0000 ～ T4095	
计数器区	C0000 ～ C4095	
DM 区	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	
数据寄存器	---	
索引寄存器	---	
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

描述

SLD(074) 把 S 和 E 之间的数据向左移一个数字 （4 位）为单位移动。“0”被放置到最右边的数字（S 的位 3 ～位 0），最左位数字（E 的位 15 ～位 12）内容丢失。



标志

名称	标记	操作
错误标志	ER	St 大于 E 时 ON。 其它情况 OFF。

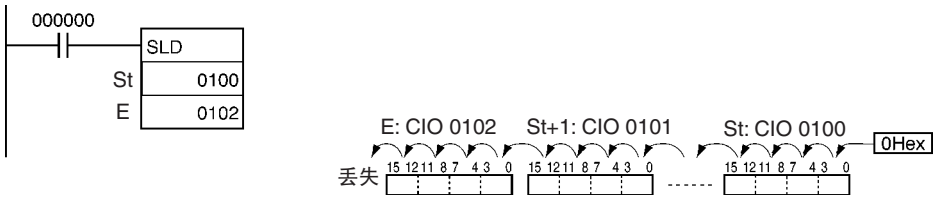
注意

当 St 大于 E 时，会产生错误，错误标志会变为 ON。

注 当要移动大量数据时，指令执行时间相当长，在 SLD(074) 在执行时，一定不要切断电源，防止移动操停在半途中。

例

当 CIO 000000 为 ON 后，字 CIO 0100 ～ CIO 0102 会向左移动一个数字 （4 位）。零被放到字CIO 0100的位0～3。并且 CIO 0102的位12～15的内容丢失。

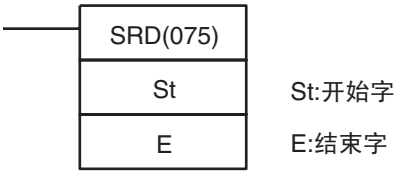


3-9-18 一个数字右移：SRD(075)

用途

向右移动一个数字 （4 位）的数据。

梯形图符号



变化

变化	ON 条件时每次循环执行	SRD(075)
	上升沿微分时执行一次	@SRD(075)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

注 St 和 E 必须在同一数据区内。

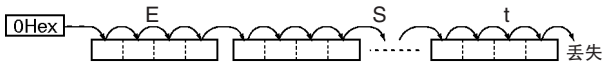
操作数规定

区域	St	E
CIO 区	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W511	
保持位区	H000 ～ H511	
辅助位区	A448 ～ A959	

区域	St	E
定时器区	T0000 ～ T4095	
计数器区	C0000 ～ C4095	
DM 区	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	
数据寄存器	---	
索引寄存器	---	
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15,IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15	

描述

SRD(075) 把 St 和 E 之间的数据向右移动一个数字（4 位），“0”被放到最左的数字（E 的位 15 ～ 12），并且最右的数字（St 的位 3 ～ 0）内容丢失。



标志

名称	标记	操作
错误标志	ER	St 大于 E 时 ON。 其它情况下 OFF。

注意

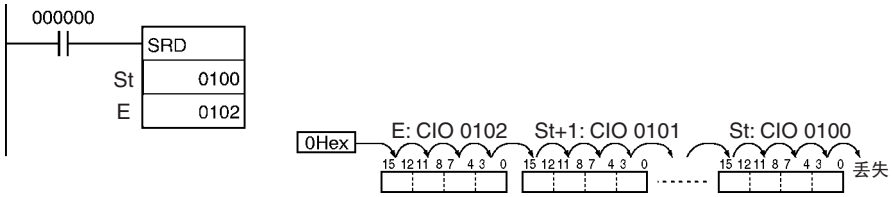
St 大于时 E，会产生错误，并且错误标志会变 ON。

当 SRD(075) 执行后，等于标志和负标志会变 OFF。

注 当移动大量数据时，指令执行时间相当长，在 SRD(075) 正在执行时，一定要注意不要切断电源，避免移动操作停在半途中。

例

当 CIO 00000 为 ON 后，字 CIO 0100 ～ CIO 0102 会向右移动一个数字（4 位），零被放到 CIO 0102 的位 12 ～ 15，字 CIO 0100 的位 0 ～ 3 的内容丢失。

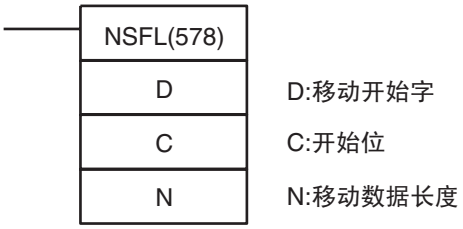


3-9-19 N 位数据左移：NSFL(578)

用途

向左移动指定数目的位。

梯形图符号



变化

变化	ON 条件时每次循环执行	NSFL(578)
	上升沿微分时执行一次	@NSFL(578)
	下升沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数

C: 0000 ～ 000F Hex (0 ～ 15)
N: 0000 ～ FFFF Hex (0 ～ 65535)

注 所有移位寄存器中的字必须在同一数据区。

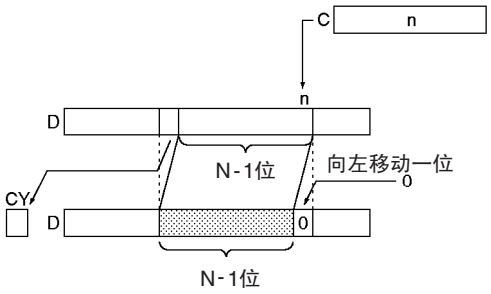
操作数规定

区域	D	C	N
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A448 ～ A959	A000 ～ A959	
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		

区域	D	C	N
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---	#0000 ～ #000F（二进制）或 &0 ～ &15	#0000 ～ #FFFF（二进制）或 &0 ～ &65535
数据寄存器	---	DR0 ～ DR15	
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

描述

NSFL (578) 把从 D 中指定最右边位 (C) 开始左移动且由移动数据长度 (N) 指定的位数。（从最右位向最左位）。“0”被放入开始位，并且移动区中最左位的内容移入进位标志 (CY)。



标志

名称	标记	操作
错误标志	ER	C 数据不在 0000 ～ 000F 十六进制之间时 ON。 其它情况 OFF。
进位标志	CY	1 被移入进位标志 (CY) 时 ON。 其它情况 OFF。

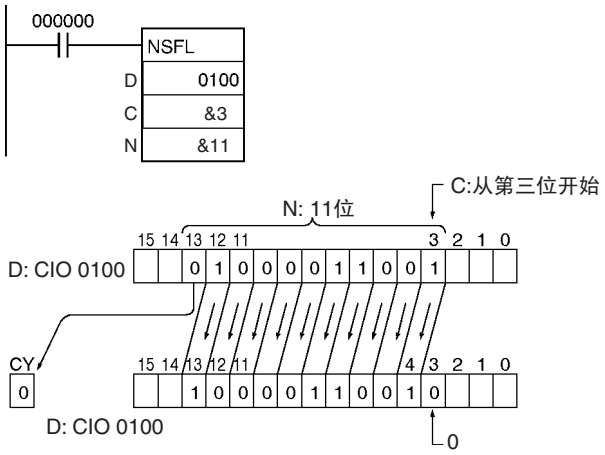
注意

当移入数据长度 (N) 是 0 时，开始位的内容会复制到进位标志 (CY)，并且开始位内容不变。

仅移动区域中被移入最右边位会改变。（也就是最左边字的数据）。

例

当 CIO 000000 ON 后，从位 3 开始到移动数据长度 (B Hex) 的所有位会向左（从最右位向最左位）移动一位。“0”被放入 CIO 0100 的位移位区域的最左位内容 （CIO 0100 的位 13）复制到进位标志 (CY)。

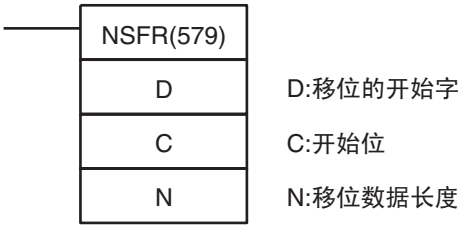


3-9-20 N 位数据右移：NSFR(579)

用途

向右移动指定数目的位。

梯形图符号



变化

变化	ON 条件时每次循环执行	NSFR(579)
	上升沿微分时执行一次	@NSFR(579)
	下升沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数

C: 0000 ～ 000F Hex (0 ～ 15)
N: 0000 ～ FFFF Hex (0 ～ 65535)

注 移位寄存器中的所有字必须在同一数据区。

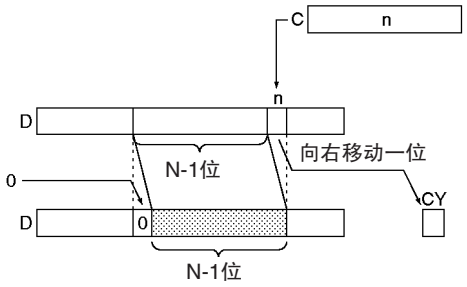
操作数规定

区域	D	C	N
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A448 ～ A959	A000 ～ A959	
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		

区域	D	C	N
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---	#0000 ～ #000F（二进制）或 &0 ～ &15	#0000 ～ #FFFF（二进制）或 &0 ～ &65535
数据寄存器	---	DR0 ～ DR15	
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

描述

NSFR(579) 把从 D 中指定最右边位 (C) 开始，右移动且由移动数据长度 (N) 指定的位数。（向最右字和最左位）。“0”会被放入开始位，并且移动区中最右位的内容会移入进位标志 (CY)。



标志

名称	标记	操作
错误标志	ER	C 数据不在 0000~000F 十六进制之间时 ON。 其它情况 OFF。
进位标志	CY	1 被移入进位标志 (CY) 时 ON。 其它情况 OFF。

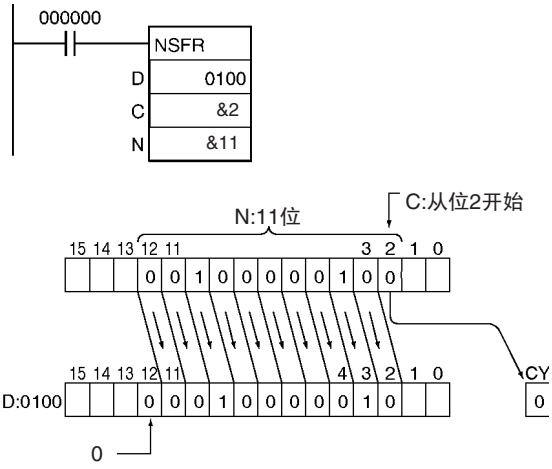
注意

当移入数据长度为 0 时，开始位的内容会复制给进位标志 (CY)，并且本身内容不变。

仅移动数据区域中被移入最右边的字会改变。（也就是最左边字的数据）。

例

当 CIO 000000 ON 后，从开始位 2 到移动数据长度 11 位 (B Hex) 结束的所有位，会向右移动一位。（从最左位向最右位）。“0”被移入 CIO 0100 的位 12，移位区的最右位的内容 （CIO 0100 的位 2）复制给进位标志 (CY)。

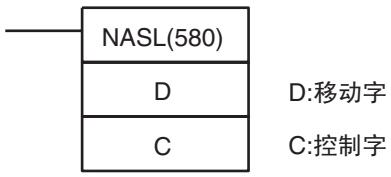


3-9-21 N 位数据左移：NASL(580)

用途

把指定的字数据的 16 位 向左移动指定数目的位

梯形图符号



变化

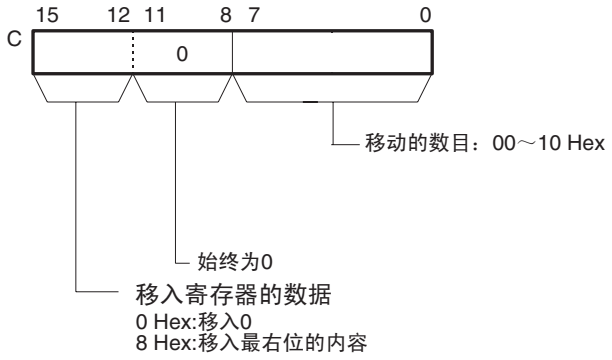
变化	ON 条件时每次循环执行	NASL(580)
	上升沿微分时执行一次	@NASL(580)
	下升沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数

C: 控制字



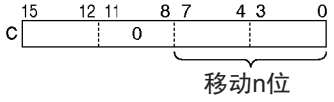
操作数规定

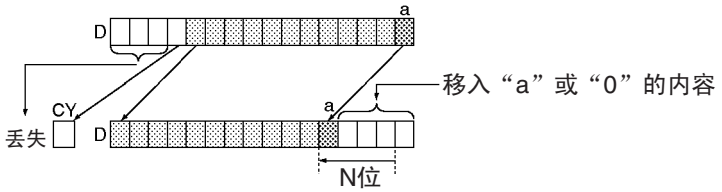
区域	D	C
CIO 区	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W511	
保持位区	H000 ～ H511	

区域	D	C
辅助位区	A448 ～ A959	A000 ～ A959
定时器区	T0000 ～ T4095	
计数器区	C0000 ～ C4095	
DM 区	D00000 ～ D32767	
无区号 DM 区	E00000 ～ E32767	
有区号 DM 区	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	仅为指定的值
数据寄存器	DR0 ～ DR15	
索引寄存器	---	
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15	

描述

NASL(580) 把从 D（移动字）向左（从最右位向最左位）移动指定数目的二进制位 （在 C 中指定），零或最右位的值被放入移动字从最右位开始的指定数目的位中。





标志

名称	标记	操作
错误标志	ER	控制字 C （移动的位数）不在范围内时 ON。 其它情况下 OFF。
等于标志	=	移动结果为 0 时 ON。 其它情况下 OFF。
进位标志	CY	1 被移入进位标志 (CY) 时 ON。 其它情况下 OFF。
负数标志	N	移位结果最左位为 1 时 ON。 其它情况下 OFF。

注意

对于移出指定字的那些位，最后位的内容移入进位标志 (CY)，其它所有数据丢失。

当要移动的位数 （在 C 中指定）是 “0” 时，数据不会移动，但是指定字中的数据，相应的标志会变 ON 和 OFF。

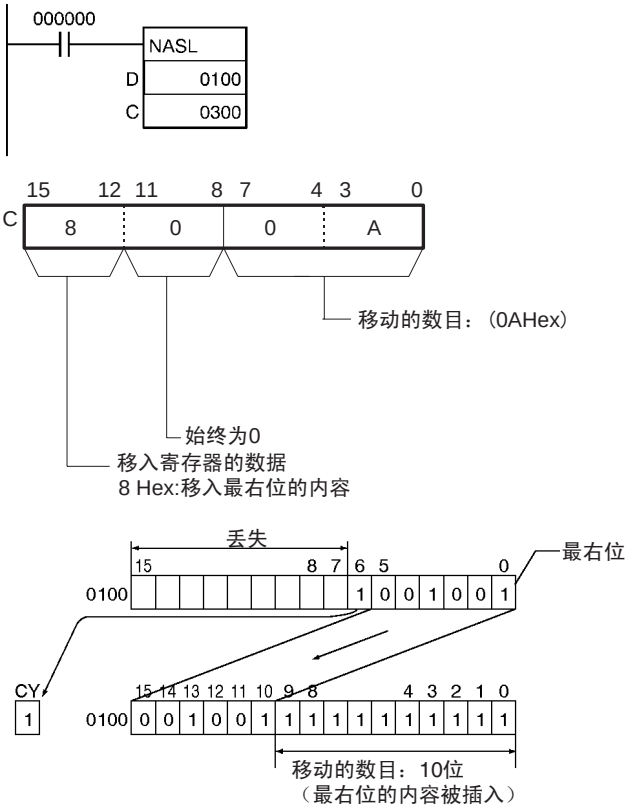
当控制字 C 的内超出范围，会产生错误，并且错误标志会变 ON。

移位结果 D 的内容是 000Hex，等于标志会变 ON。

移位结果 D 的最左位内容是 1 时，负标志会变 ON。

例

当 CIO 000000 为 ON 后，CIO 0100 的内容向左移动 10 位（从最右位向最左位），要移动的位数在字 CIO 0300（控制数据）的位 0 ～ 7 中指定，CIO 0100 的位 0 内容复制到数据被移动的位中，移出范围的最右位的内容移入进位标志 (CY)，所有其它数据丢失。

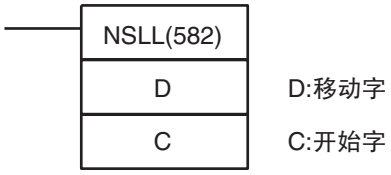


3-9-22 双字 N 位数据左移：NSLL(582)

用途

把指定的字数据的 32 位 向左移动指定数目的位

梯形图符号



变化

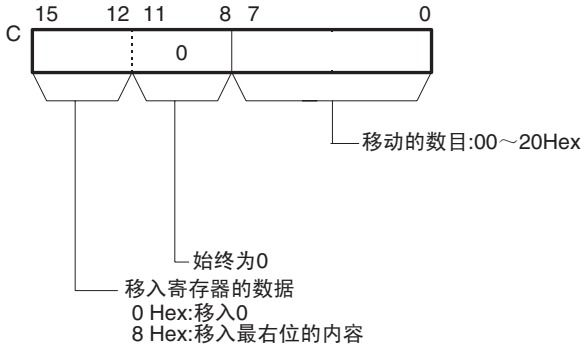
变化	ON 条件时每次循环执行	NSLL(582)
	上升沿微分时执行一次	@NSLL(582)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数

C: 控制字



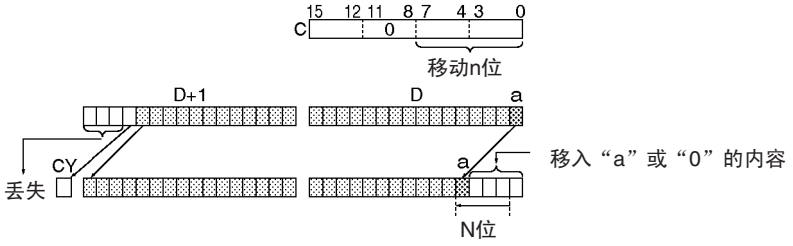
操作数规定

区域	D	C
CIO 区	CIO 0000 ～ CIO 6142	CIO 0000 ～ CIO 6143
工作区	W000 ～ W510	W000 ～ W511
保持位区	H000 ～ H510	H000 ～ H511
辅助位区	A448 ～ A958	A000 ～ A959
定时器区	T0000 ～ T4094	T0000 ～ T4095
计数器区	C0000 ～ C4094	C0000 ～ C4095
DM 区	D00000 ～ D32766	D00000 ～ D32767
无区号 EM 区	E00000 ～ E32766	E00000 ～ E32767
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	

区域	D	C
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	仅为指定的值
数据寄存器	---	DR0 ～ DR15
索引寄存器	---	
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15	

描述

NSLL(582) 把从 D 和 D+1 （移动字）向左 （从最右位向最左位）移动指定数目的二进制位 （在 C 中指定），零或最右位的值被放入移动字从最右位开始的指定数目的位中。



标志

名称	标记	操作
错误标志	ER	控制字 C （移动的位数）不在范围内时 ON。 其它情况下 OFF。
等于标志	=	移动结果为 0 时 ON。 其它情况下 OFF。
进位标志	CY	1 被移入进位标志 (CY) 时 ON。 其它情况下 OFF。
负数标志	N	移位结果最左位为 1 时 ON。 其它情况下 OFF。

注意

对于移出指定字的那些位，最后一位的内容移入进位标志 (CY)，其它所有数据丢失。

当要移动的位数 （在 C 中指定）是 “0”时，数据不会移动，但是指定字中的数据有关的标志会变 ON 和 OFF。

当控制字 C 的内超出范围，会产生错误，并且错误标志会变 ON。

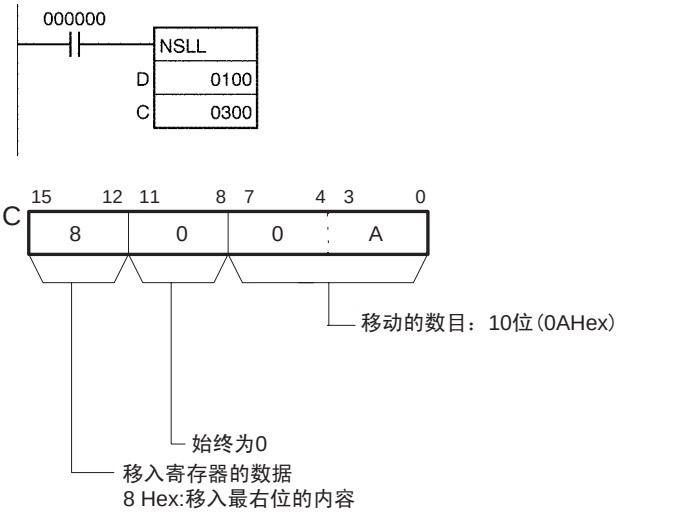
移位结果 D 的内容是 0000，就等于标志会变 ON。

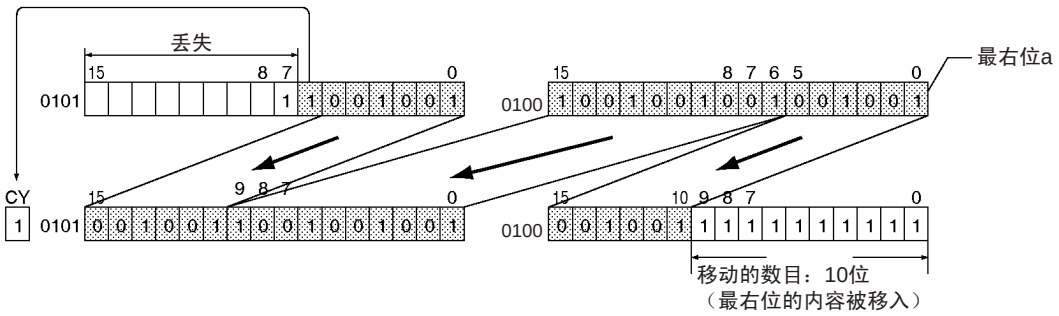
移位结果 D， D+1 的最左位是 1 时，负标志会变 ON。

例

当 CIO 000000 变 ON 后， CIO 0100 和 CIO 0101 会向左移动 10 位（从最右位向最左位），移动的位数在字 CIO 0300 （控制数据）的位 0 ～ 7 中指定，

CIO 0100 的位 0 内容复制到数据被移动的位中，移出范围的最右位的内容移入进位标志 (CY)，所有其它数据丢失。



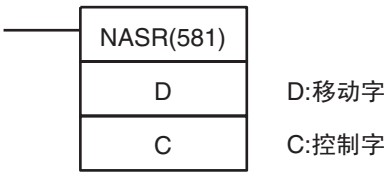


3-9-23 N 位右移：NASR(581)

用途

把指定的 16 位字数据向右移动指定数目的位

梯形图符号



变化

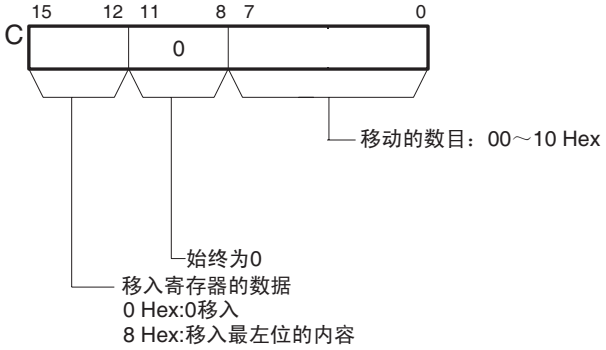
变化	ON 条件时每次循环执行	NASR(581)
	上升沿微分时执行一次	@NASR(581)
	下升沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数

C: 控制字

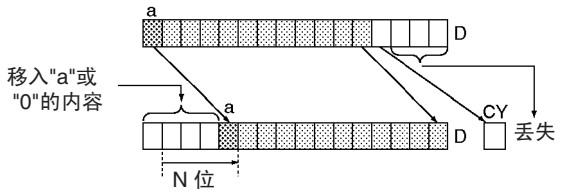


操作数规定

区域	D	C
CIO 区	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W511	
保持位区	H000 ～ H511	
辅助位区	A448 ～ A959	A000 ～ A447 A448 ～ A959
定时器区	T0000 ～ T4095	
记数器区	C0000 ～ C4095	
DM 区	D00000 ～ D32767	
无区号 DM 区	E00000 ～ E32767	
有区号 DM 区	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	仅为指定的数
数据寄存器	DR0 ～ DR15	
索引寄存器	---	
使用索引寄存器 间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

描述

NASR(581) 把 D （移动字）向右 （从最右位向最左位）移动指定数目的二进制位 （在 C 中指定），零或最左位的值被放入移动字从最左位开始的指定数目的位中。



标志

名称	标志	处理
错误标志	ER	控制字 C （移动的位数）不在范围内时 ON。 其它情况下 OFF。
等于标志	=	移动结果为 0 时 ON。 其它情况下 OFF。
进位标志	CY	1 被移入进位标志 (CY) 时 ON。 其它情况下 OFF。
负标志	N	移位结果最左位为 1 时 ON。 其它情况下 OFF。

注意

对于移出指定字的位，最后一位的内容移入进位标志 (CY)，其它所有数据丢失。

当要移动的位数 （在 C 中指定）是 “0” 时，数据不会移动，但是指定字中的数据，有关的标志会变 ON 和 OFF。

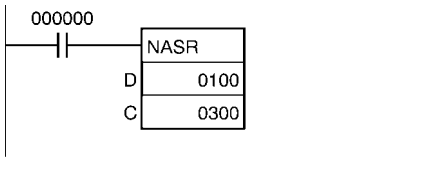
当控制字 C 的内超出范围，会产生错误，并且错误标志会变 ON。

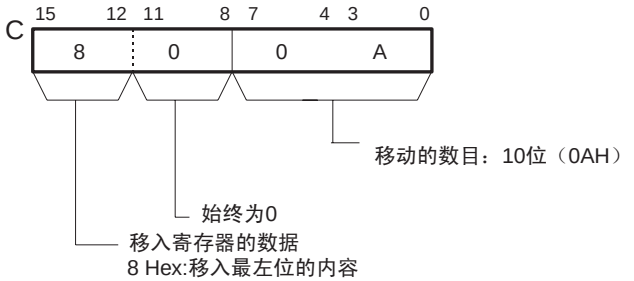
移位结果 D 的内容是 0000Hex，就等于标志会变 ON。

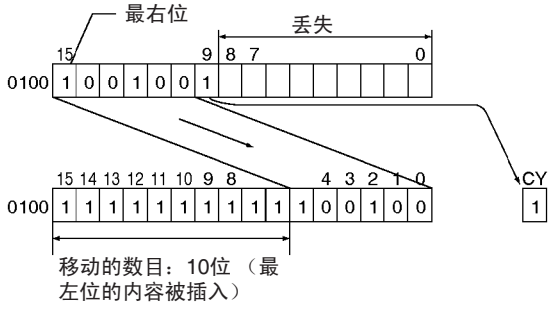
移位结果 D 的最左位内容是 1 时，负标志会变 ON。

例

当 CIO 000000 变 ON 后，CIO 0100 会向右移动 10 位（从最左位向最右位），移动的位数在字 CIO 0300 的位 0~7 中指定，CIO 0100 的位 15 内容复制到数据被移动的位中，移出范围的最右位的内容移入进位标志 (CY)，所有其它数据丢失。





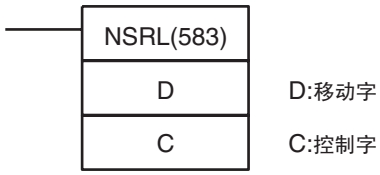


3-9-24 双字 N 位右移：NSRL(583)

用途

把指定的 32 位字数据向右移动指定数目的位 .

梯形图符号



变化

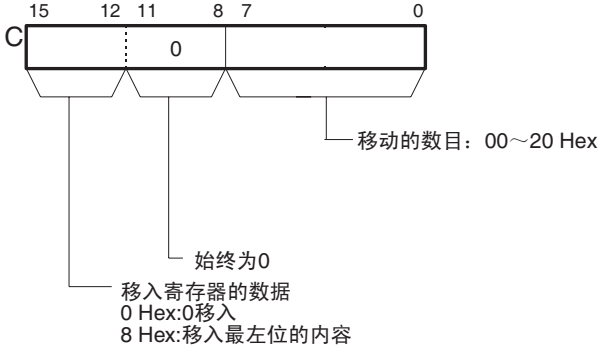
变化	ON 条件时每次循环执行	NSRL(583)
	上升沿微分时执行一次	@NSRL(583)
	下升沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数

C: 控制字



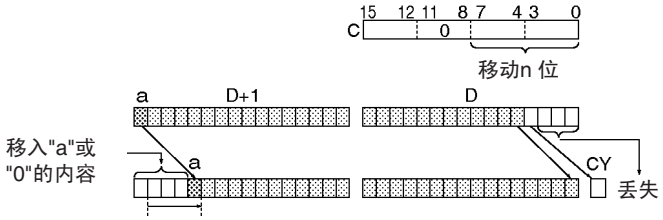
操作数规定

区域	D	C
CIO 区	CIO 0000 ～ CIO 6142	CIO 0000 ～ CIO 6143
工作区	W000 ～ W510	W000 ～ W511
保持位区	H000 ～ H510	H000 ～ H511
辅助位区	A448 ～ A958	A000 ～ A959
定时器区	T0000 ～ T4094	T0000 ～ T4095
计数器区	C0000 ～ C4094	C0000 ～ C4095

区域	D	C
DM 区	D00000 ～ D32766	D00000 ～ D32767
无区号 DM 区	E00000 ～ E32766	E00000 ～ E32767
有区号 DM 区	En_00000 ～ En_32766 (n = 0 ～ C)	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	仅为指定的数
数据寄存器	---	DR0 ～ DR15
索引寄存器	---	
使用索引寄存器间接寻址	,IR0 ～ ,IR15 -2048 ～ +2047 ,IR0 ～ -2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

描述

NRSL(583) 把 D 和 D+1 （移动字）向右 （从最左位向最右位）移动指定数目的二进制位 （在 C 中指定），零或最左位的值被放入移动字从最左位开始的指定数目的位中。



标志

名称	标识	处理
错误标志	ER	控制字 C （移动的位数）不在范围内时 ON。 其它情况下 OFF。
等于标志	=	移动结果为 0 时 ON。 其它情况下 OFF。
进位标志	CY	1 被移入进位标志 (CY) 时 ON。 其它情况下 OFF。
负标志	N	移位结果最左位为 1 时 ON。 其它情况下 OFF。

注意

对于移出指定字的位，最后一位的内容移入进位标志 (CY)，其它所有数据丢失。

当要移动的位数 （在 C 中指定）是 “0” 时，数据不会移动，但是指定字中的数据，有关的标志会变 ON 和 OFF。

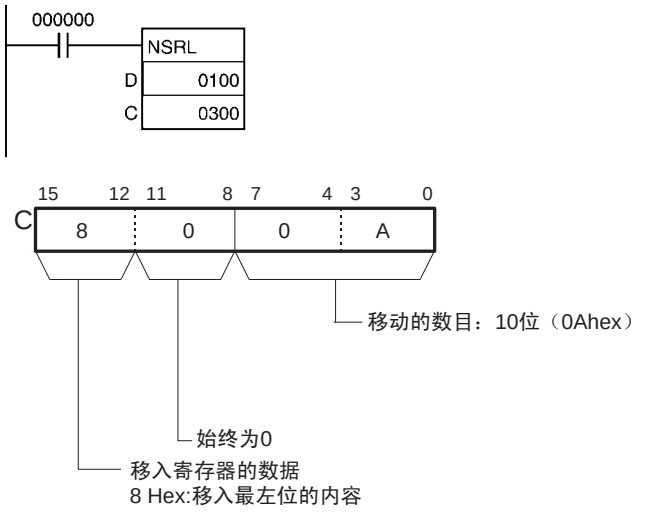
当控制字 C 的内超出范围，会产生错误，并且错误标志会变 ON。

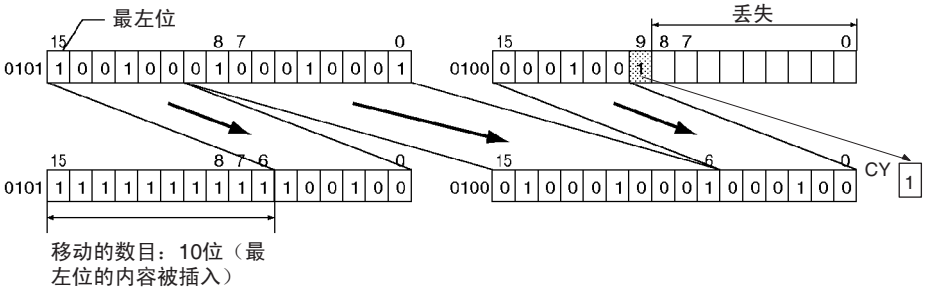
移位结果 D， D+1 的内容是 00000000Hex，就等于标志会变 ON。

移位结果 D+1 的最左位内容是 1 时，负标志会变 ON。

例

当 CIO 000000 变 ON 后， CIO 0100 和 CIO 0101 会向右 （从最左位向最右位）移动 10 位，移动的位数在字 CIO 0300 （控制数据）的位 0~7 中指定，CIO 0101 的位 15 内容复制到数据被移动的位中，移出范围的最左位的内容移入进位标志 (CY)，所有其它数据丢失。





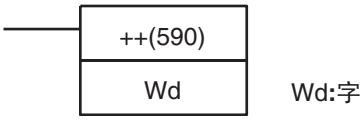
3-10 递增 / 递减指令

3-10-1 二进制递增：++(590)

用途

指定字的 4 位数十六进制内容加 1。

梯形图符号



变化

变化	ON 条件时每次循环执行	++(590)
	上升沿微分时执行一次	@++(590)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

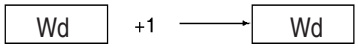
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	Wd
CIO 区	CIO 0000 ～ CIO 6143
工作区	W000 ～ W511
保持位区	H000 ～ H511
辅助位区	A448 ～ A959
定时器区	T0000 ～ T4095
计数器区	C0000 ～ C4095
DM 区	D00000 ～ D32767
无区号 DM 区	E00000 ～ E32767
有区号 DM 区	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---
数据寄存器	DR0 ～ DR15
索引寄存器	---
使用索引寄存器 间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

++(590) 指令使 Wd 二进制内容加 1，只要 ++(590) 的执行条件 ON，每次循环指定字会增加 1，当使用这条指令的上升沿微分变化 (@++(590)) 时，仅在执行条件从 OFF 变到 ON 时指定字递增。



如果结果是 0000，等于标志会变 ON，当一个数从 F 变到 0 时，进位标志会变 ON，当结果中 Wd 的位 15 是 ON 时，负标志会变 ON。

当 Wd 的内容从 FFFF 变到 0000 时，等于标志和进位标志都会变 ON。

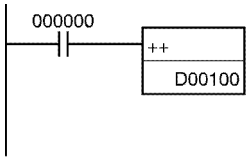
标志

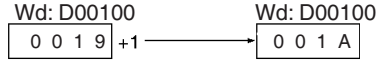
名称	标识	处理
错误标志	ER	OFF
等于标志	=	执行后 Wd 的内容是 0000 时 ON。 其它情况下 OFF。
进位标志	CY	在执行期间 Wd 中的一个数从 F 变到 0 时 ON。 其它情况下 OFF。
负标志	N	执行后 Wd 的位 15 是 ON 时 ON。 其它情况下 OFF。

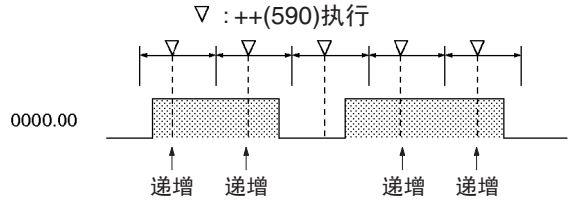
例

++(590) 操作

在下面例子中，只要 CIO 000000 ON。每次循环 D00100 的内容会增加 1。

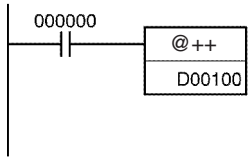


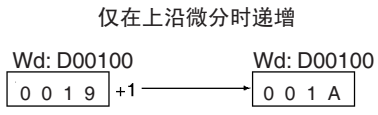


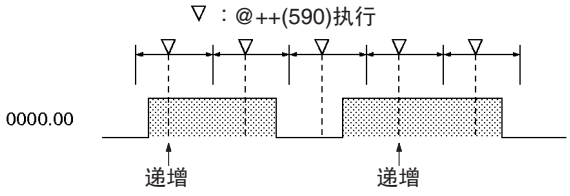


@++(590) 操作

在下面例子中使用上升沿微分变化，因此仅当 CIO000000 从 OFF 变到 ON 时，D00100 的内容增加 1。





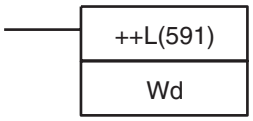


3-10-2 双字二进制递增：++L(591)

用途

指定字 8 位数十六进制内容增加 1。

梯形图符号



变化

变化	ON 条件时每次循环执行	++L(591)
	上升沿微分时执行一次	@++L(591)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

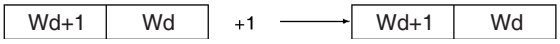
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	Wd
CIO 区	CIO 0000 ～ CIO 6142
工作区	W000 ～ W510
保持位区	H000 ～ H510
辅助位区	A448 ～ A958
定时器区	T0000 ～ T4094
计数器区	C0000 ～ C4094
DM 区	D00000 ～ D32766
无区号 DM 区	E00000 ～ E32766
有区号 DM 区	En_00000 ～ En_32766 (n = 0 ～ C)
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---
数据寄存器	---
索引寄存器	IR0 ～ IR15
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

++L(591) 指令使 Wd+1 和 Wd 的 8 位数十六进制数内容加 1，只要 ++L(591) 的执行 ON，每次循环指定字的内容会增加 1，当使用这条指令的上升沿微分变化 (@++L(590)) 时，仅在执行条件从 OFF 变到 ON 时，指定字的内容递增。



如果结果是 00000000，等于标志会变 ON，当一个从数 F 变到 0 时，进位标志会变 ON，当结果中 Wd+1 的位 15 是 ON 时，负标志会变 ON。

当Wd的内容从FFFFFFFF变到00000000时，等于标志和进位标志都会变ON。

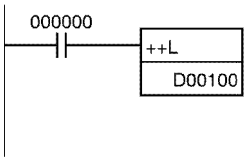
标志

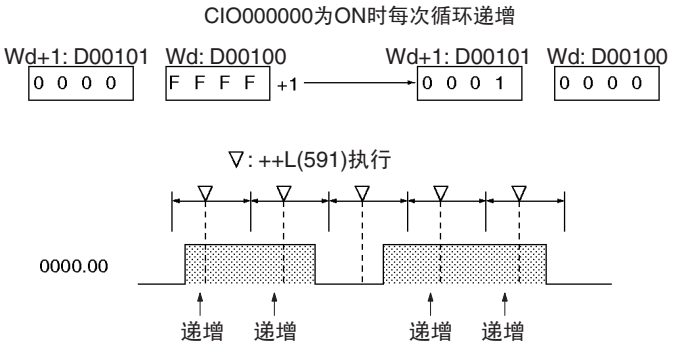
名称	标识	处理
错误标志	ER	OFF
等于标志	=	执行后 Wd 的内容是 00000000 时 ON。 其它情况下 OFF。
进位标志	CY	在执行期间 Wd+1 中的一个数从 F 变到 0 时 ON。 其它情况下 OFF。
负标志	N	执行后 Wd 的位 15 是 ON 时 ON。 其它情况下 OFF。

例

++L(591) 操作

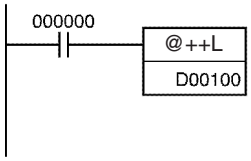
在下面例子中，只要 CIO 000000 ON。每次循环 D00101 和 D00100 的 8 位数的十六进制内容会增加 1。

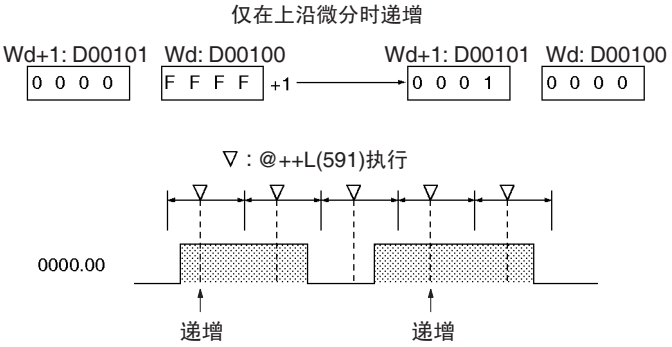




@++L(591) 操作

在下面例子中使用上升沿微分变化，因此仅当 CIO000000 从 OFF 变到 ON 时，D00101 和 D00100 的内容增加 1。



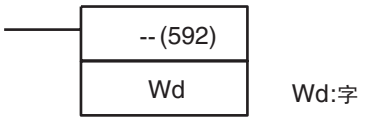


3-10-3 二进制递减：--(592)

用途

指定字的 4 位数十六进制内容减 1。

梯形图符号



变化

变化	ON 条件时每次循环执行	– – (592)
	上升沿微分时执行一次	@– – (592)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

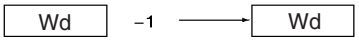
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	Wd
CIO 区	CIO 0000 ～ CIO 6143
工作区	W000 ～ W511
保持位区	H000 ～ H511
辅助位区	A448 ～ A959
定时器区	T0000 ～ T4095
计数器区	C0000 ～ C4095
DM 区	D00000 ～ D32767
无区号 DM 区	E00000 ～ E32767
有区号 DM 区	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767
常数	---
数据寄存器	DR0 ～ DR15
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

--(592) 指令使 Wd 二进制内容减 1，只要 ++(592) 的执行条件 ON，每次循环指定字会减 1，当使用这条指令的上升沿微分变化 (@++(592)) 时，仅在执行条件从 OFF 变到 ON 时指定字递减。



如果结果是 0000，等于标志会变 ON，当一个数从 0 变到 F 时，进位标志会变 ON，当结果中 Wd 的位 15 是 ON 时，负标志会变 ON。
当 Wd 的内容从 0000 变到 FFFF 时，进位标志和负标志都会变 ON。

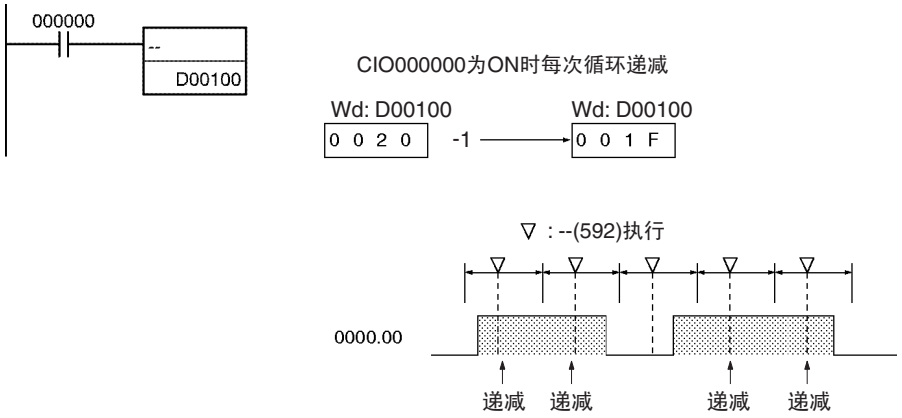
标志

名称	标识	处理
错误标志	ER	OFF
等于标志	=	执行后 Wd 的内容是 0000 时 ON。 其它情况下 OFF。
进位标志	CY	在执行期间 Wd 中的一个数从 0 变到 F 时 ON。 其它情况下 OFF。
负标志	N	执行后 Wd 的位 15 是 ON 时 ON。 其它情况下 OFF。

例

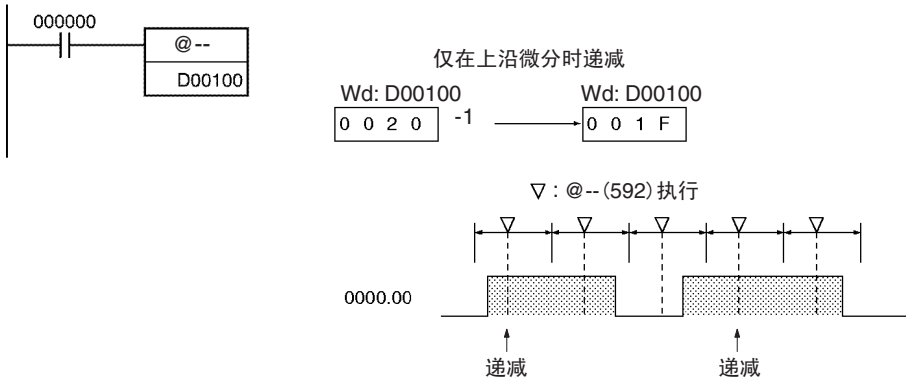
--(592) 操作

在下面例子中，只要 CIO 000000 ON。每次循环 D00100 的内容会减 1。



@--(592) 操作

在下面例子中使用上升沿微分变化，因此仅当 CIO000000 从 OFF 变到 ON 时，D00100 的内容减 1。

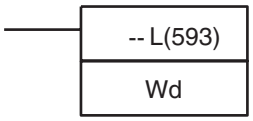


3-10-4 双字二进制递减：--L(593)

用途

指定字的 8 位数十六进制内容减 1。

梯形图符号



Wd: 起始字

变化

变化	ON 条件时每次循环执行	– –L(593)
	上升沿微分时执行一次	@– –L(593)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

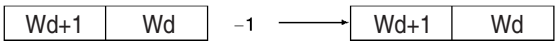
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	Wd
CIO 区	CIO 0000 ～ CIO 6142
工作区	W000 ～ W510
保持位区	H000 ～ H510
辅助位区	A448 ～ A958
定时器区	T0000 ～ T4094
计数器区	C0000 ～ C4094
DM 区	D00000 ～ D32766
无区号 DM 区	E00000 ～ E32766
有区号 DM 区	En_00000 ～ En_32766 (n = 0 ～ C)
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---
数据寄存器	---
索引寄存器	IR0 ～ IR15
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

--L(593) 指令从 Wd+1 和 Wd 的 8 位数十六进制内容中减去 1，只要 --L(593) 的执行条件 ON，每次循环指定字会减 1，当使用这条指令的上升沿微分变化 (@ --L(593)) 时，指定字的内容仅在执行条件从 OFF 变到 ON 时递减。



如果结果是 00000000，等于标志会变 ON，当一个数 0 变到 F 时，进位标志会变 ON，当结果中 Wd+1 的位 15 是 ON 时，负标志会变 ON。

当内容从 00000000 变到 FFFFFFFF 时，进位标志和等于标志都会变 ON。

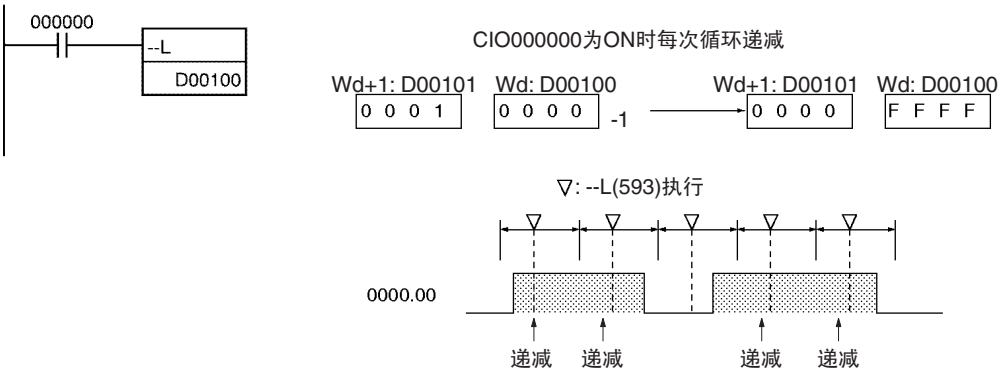
标志

名称	标识	处理
错误标志	ER	OFF
等于标志	=	执行后结果是 00000000 时 ON。 其它情况下 OFF。
进位标志	CY	在执行期间 Wd+1 或 Wd 中的一个数从 0 变到 F 时 ON。 其它情况下 OFF。
负标志	N	执行后 Wd+1 的位 15 是 ON 时 ON。 其它情况下 OFF。

例

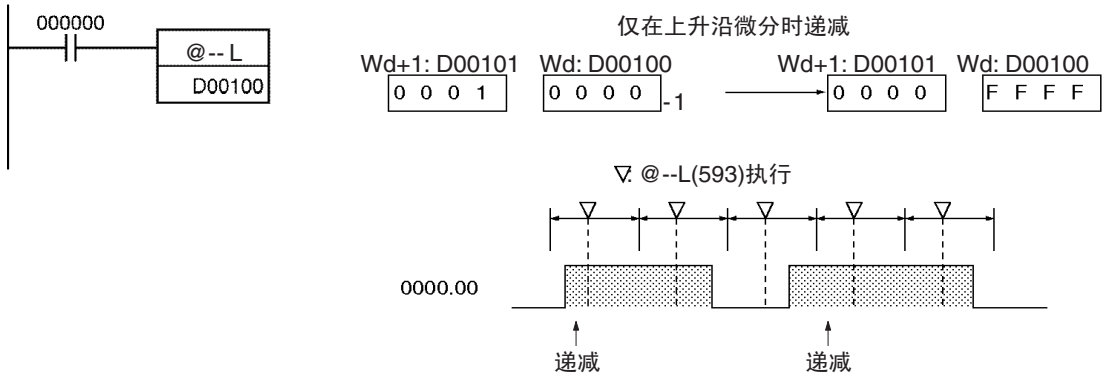
--L(593) 操作

在下面例子中，只要 CIO 000000 ON。每次循环 D00101 和 D00100 的 8 位数的十六进制内容会减去 1。



@--L(593) 操作

在下面例子中使用上升沿微分变化，因此D00101和D00100的内容 仅在CIO000000从 OFF 变到 ON 时减去 1。

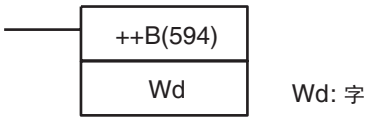


3-10-5 BCD 递增：++B(594)

用途

指定字的 4 位数 BCD 制内容加 1。

梯形图符号



变化

变化	ON 条件时每次循环执行	++B(594)
	上升沿微分时执行一次	@++B(594)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

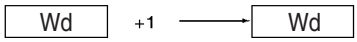
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	Wd
CIO 区	CIO 0000 ～ CIO 6143
工作区	W000 ～ W511
保持位区	H000 ～ H511
辅助位区	A448 ～ A959
定时器区	T0000 ～ T4095
计数器区	C0000 ～ C4095
DM 区	D00000 ～ D32767
无区号 DM 区	E00000 ～ E32767
有区号 DM 区	En_00000 ～ En_32767 (n= 0 ～ C)
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---
数据寄存器	DR0 ～ DR15
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

++B(594) 指令使 Wd 的 BCD 内容加 1，只要 ++B(594) 的执行条件 ON，每次循环指定字会增加 1。当使用这条指令的上升沿微分变化 (@++B(594)) 时指定字仅当执行条件从 OFF 变到 ON 时递增。



如果结果是 0000，等于标志会变 ON，当一个数 9 变到 0 时，进位标志会变 ON。

当 Wd 的内容从 9999 变到 0000 时，等于标志和进位标志都会变 ON。

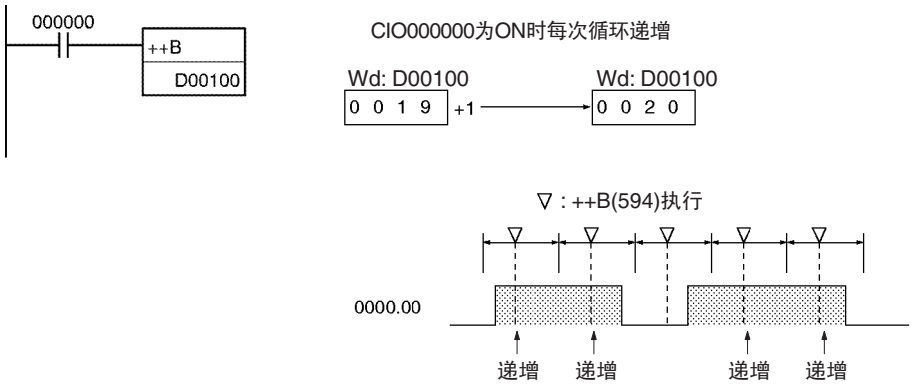
标志

名称	标识	处理
错误标志	ER	Wd 的内容不是 BCD 时 ON。 其它情况下 OFF。
等于标志	=	执行后 Wd 的内容是 00000000 时 ON。 其它情况下 OFF。
进位标志	CY	在执行期间 Wd 从 9 变到 0 时 ON。 其它情况下 OFF。

注意 Wd 的内容必须是 BCD，如果不是 BCD，会产生错误，并且错误标志变 ON。

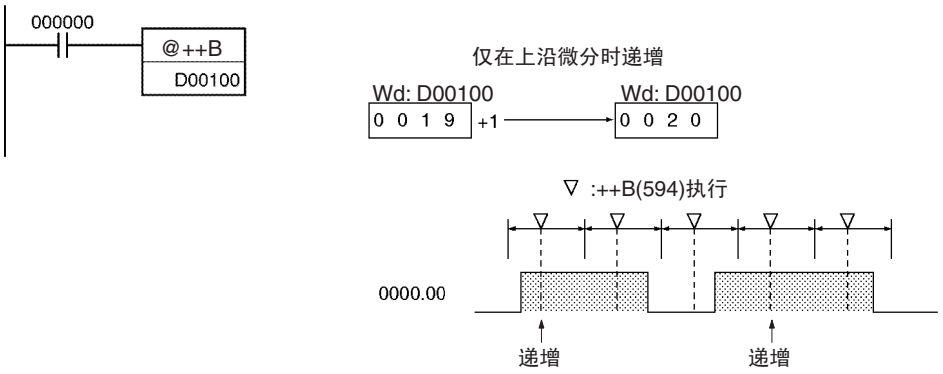
例 ++B(594) 操作

在下面例子中，只要CIO 000000 ON。每次循环D00100的BCD内容会增加1。



@++B(594) 操作

在下面例子中使用上升沿微分变化，因此仅当 D00100 的内容仅在 CIO000000 从 OFF 变到 ON 时增加 1。

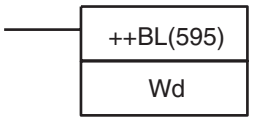


3-10-6 双字 BCD 递增：++BL(595)

用途

指定字 8 位数 BCD 制内容增加 1。

梯形图符号



变化

变化	ON 条件时每次循环执行	++BL(595)
	上升沿微分时执行一次	@++BL(595)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

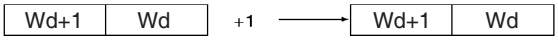
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	Wd
CIO 区	CIO 0000 ～ CIO 6142
工作区	W000 ～ W510
保持位区	H000 ～ H510
辅助位区	A448 ～ A958
定时器区	T0000 ～ T4094
计数器区	C0000 ～ C4094
DM 区	D00000 ～ D32766
无区号 DM 区	E00000 ～ E32766
有区号 DM 区	En_00000 ～ En_32766 (n = 0 ～ C)
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---
数据寄存器	---
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

++BL(595) 指令使 Wd+1 和 Wd 的 8 位数 BCD 内容加 1，只要 ++BL(595) 的执行条件 ON，每次循环指定字的内容会增加 1，当使用这条指令的上升沿微 分变化 (@++BL(595)) 时，指定字的内容仅在执行条件从 OFF 变到 ON 时递增



如果结果是 00000000，等于标志会变 ON，当一个数 9 变到 0 时，进位标志会变 ON。

当内容从 99999999 变到 00000000 时，等于标志和进位标志都会变 ON。

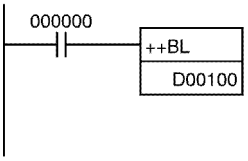
标志

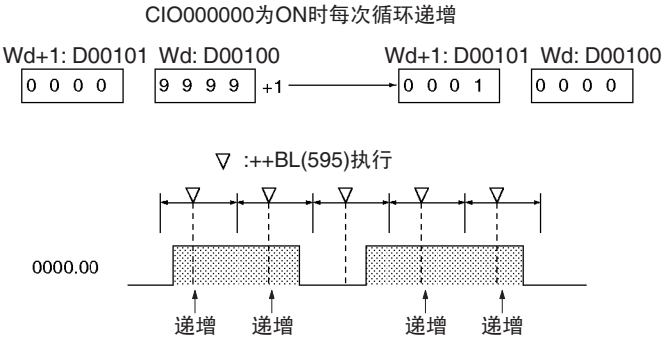
名称	标识	处理
错误标志	ER	Wd+1 的内容不是 BCD 时 ON。 其它情况下 OFF。
等于标志	=	执行后 Wd 的内容是 00000000 时 ON。 其它情况下 OFF。
进位标志	CY	在执行期间 Wd+1 或 Wd 中的一个数从 9 变到 0 时 ON。 其它情况下 OFF。

注意 Wd+1 和 Wd 的内容必须是 BCD，如果不是 BCD，会产生错误，并且错误标志变 ON。

例 ++BL(595) 操作

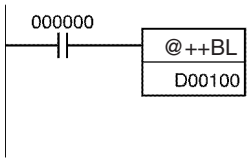
在下面例子中，只要 CIO 000000 ON。每次循环 D00101 和 D00100 的 8 位数的 BCD 内容会增加 1。

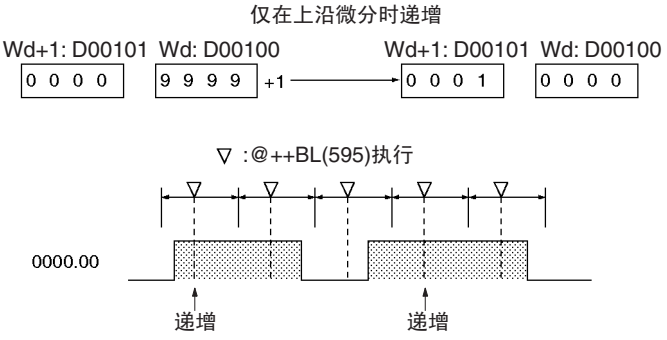




@++BL(595) 操作

在下面例子中使用上升沿微分变化，因此 D00101 和 D00100 的 BCD 内容仅当 CIO000000 从 OFF 变到 ON 时增加 1。



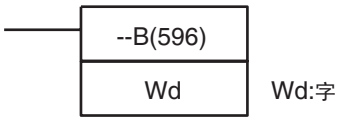


3-10-7 BCD 递减：--B(596)

用途

指定字的 4 位数 BCD 内容减去 1。

梯形图符号



变化

变化	ON 条件时每次循环执行	– –B(596)
	上升沿微分时执行一次	@– –B(596)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

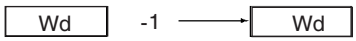
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	Wd
CIO 区	CIO 0000 ～ CIO 6143
工作区	W000 ～ W511
保持位区	H000 ～ H511
辅助位区	A448 ～ A959
定时器区	T0000 ～ T4095
计数器区	C0000 ～ C4095
DM 区	D00000 ～ D32767
无区号 DM 区	E00000 ～ E32767
有区号 DM 区	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---
数据寄存器	DR0 ～ DR15
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

--B(596) 指令从 Wd 的 BCD 内容中减 1，只要 --B(596) 的执行条件 ON，每次循环指定字会减 1，当使用这条指令的上升沿微分变化 (@--B(596)) 时，指定的字仅在执行条件从 OFF 变到 ON 时递减。



如果结果是0000，等于标志会变ON，当一个数0变到9时，进位标志会变ON。

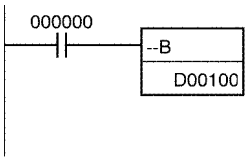
标志

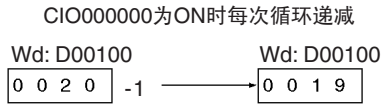
名称	标识	处理
错误标志	ER	Wd 的内容不是 BCD 时 ON。 其它情况下 OFF。
等于标志	=	执行后 Wd 的内容是 0000 时 ON。 其它情况下 OFF。
进位标志	CY	在执行期间 Wd 中的一个数从 0 变到 9 时 ON。 其它情况下 OFF。

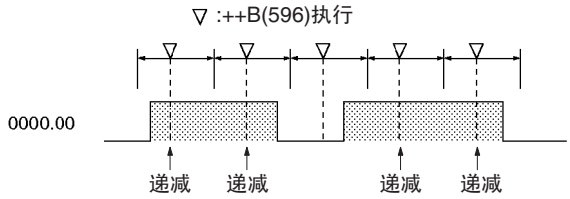
注意 Wd 的内容必须是 BCD，如果不是 BCD，会产生错误，并且错误标志变 ON。

例 --B(596) 操作

在下面例子中，只要 CIO 000000 ON。每次循环 D00100 的 BCD 内容会减 1。

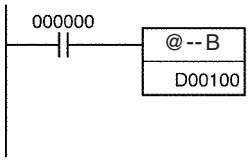


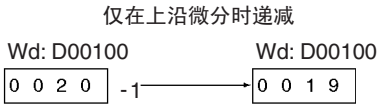


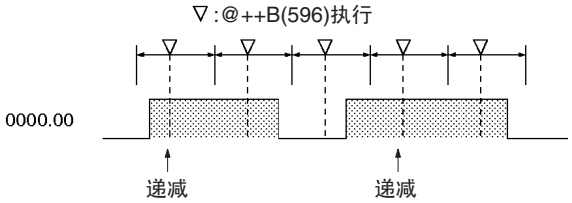


@--B(596) 操作

在下面例子中使用上升沿微分变化，因此当CIO000000从OFF变到ON时D00100的 BCD 码内容降减 1。





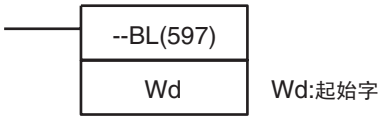


3-10-8 双字 BCD 递减：--BL(597)

用途

指定字的 8 位数 BCD 内容减去 1。

梯形图符号



变化

变化	ON 条件时每次循环执行	– –BL(597)
	上升沿微分时执行一次	@– –BL(597)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

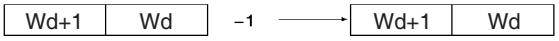
操作数规定

区域	Wd
CIO 区	CIO 0000 ～ CIO 6142
工作区	W000 ～ W510
保持位区	H000 ～ H510
辅助位区	A448 ～ A958
定时器区	T0000 ～ T4094
计数器区	C0000 ～ C4094
DM 区	D00000 ～ D32766
无区号 DM 区	E00000 ～ E32766
有区号 DM 区	En_00000 ～ En_32766 (n = 0 ～ C)
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---
数据寄存器	---
索引寄存器	---
使用索引寄存器 间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

--BL(597) 指令使 Wd+1 和 Wd 的 8 位 BCD 码内容减 1，只要 --BL(597) 的执行条件 ON，每次循环指定字的内容会减 1，当使用这条指令的上升分变化 (@-

-BL(597)) 时，指定字的内容仅在执行条件从 OFF 变到 ON 时递减。



如果结果是 00000000，等于标志会变 ON，当一个数 9 变到 0 时，进位标志会变 ON。

标志

名称	标志	处理
错误标志	ER	如果 Wd+1 和 Wd 的内容不是 BCD 码时 ON。 其它情况下 OFF。
等于标志	=	执行结果 0000 0000 时为 ON。 其它情况下 OFF。
进位标志	CY	如果执行时 Wd+1 或 Wd 的某个数字从 0 变为 9 时 ON。 其它情况下 OFF。

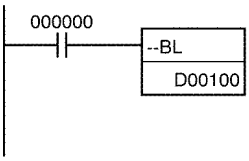
注意

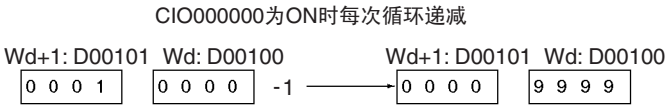
Wd+1和Wd的内容必须是BCD码。如果不是BCD码，将会出现错误，并且错误标志置 ON。

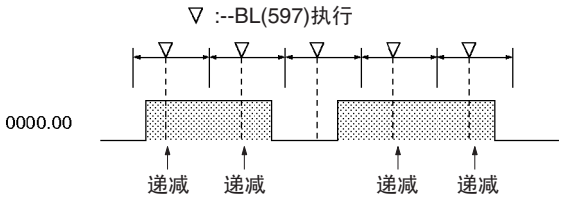
例

++BL(597) 操作

在下面例子中，只要 CIO 000000 ON。每次循环 D00101 和 D00100 的 8 位数的 BCD 内容会减 1。

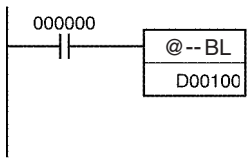


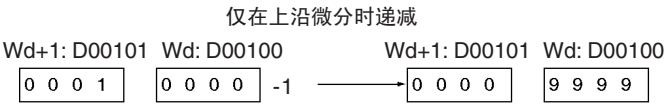


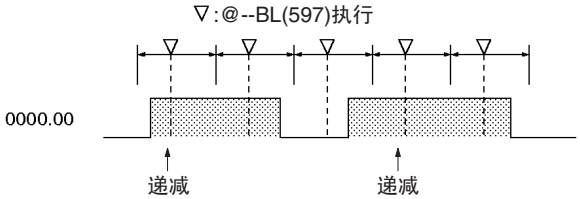


@++BL(597) 操作

在下面例子中使用上升沿微分变化，因此仅当 CIO000000 从 OFF 变到 ON 时，D00101 和 D00100 的内容减 1。







3-11 四则运算指令

本节描述了用 BCD 码或二进制数执行算术操作的四则运算指令。

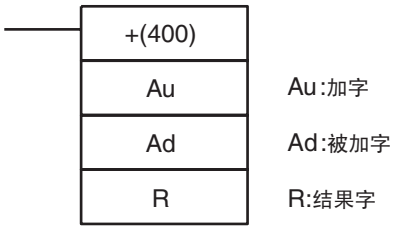
指令	助记符	函数代码	页数
不带进位的有符号二进制加	+	400	373
不带进位的有符号双字二进制加	+L	401	375
带进位的有符号二进制加	+C	402	377
带进位的有符号双字二进制加	+CL	403	379
不带进位的 BCD 加	+B	404	381
不带进位的双字 BCD 加	+BL	405	382
带进位的 BCD 加	+BC	406	384
带进位的双字 BCD 加	+BCL	407	386
不带进位的有符号二进制减	–	410	387
不带进位的有符号双字二进制减	–L	411	389
带进位的有符号二进制减	–C	412	393
带进位的有符号双字二进制减	–CL	413	395
不带进位的 BCD 减	–B	414	398
不带进位的双字 BCD 减	–BL	415	399
带进位的 BCD 减	–BC	416	403
带进位的双字 BCD 减	–BCL	417	404
有符号二进制乘	*	420	406
有符号双字二进制乘	*L	421	408
无符号二进制乘	*U	422	410
无符号双字二进制乘	*UL	423	412
BCD 乘	*B	424	413
双字 BCD 乘	*BL	425	415
有符号二进制除	/	430	417
有符号双字二进制除	/L	431	419
无符号二进制乘	/U	432	421
无符号双字二进制除	/UL	433	423
BCD 除	/B	434	425
双字 BCD 除	/BL	435	427

3-11-1 不带进位的有符号二进制加：+(400)

用途

4 个数字 （单字）十六进制数据和 / 或常数相加。

梯形图符号



变化

变化	ON 条件时每次循环执行	+(400)
	上升沿微分时执行一次	@+(400)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

应用程序区

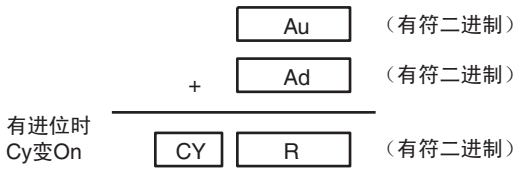
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	Au	Ad	R
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 DM 区	E00000 ～ E32767		
有区号 DM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #FFFF （二进制）		---
数据寄存器	DR0 ～ DR15		
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

描述

+(400) 把 Au 和 Ad 中的二进制值相加，并且把结果送给 R。



标志

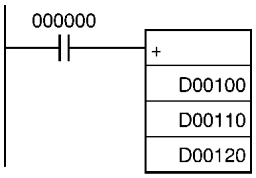
名称	标志	处理
错误标志	ER	OFF
等于标志	=	结果为 0 时置 ON。 其它情况下 OFF。
进位标志	CY	当结果有进位时置 ON。 其它情况下 OFF。
上溢出标志	OF	当两个正数相加结果在 8000~FFFF 之间时置 ON。 其它情况下 OFF。
下溢出标志	UF	当两个负数相加结果在 0000~7FFF 之间时置 ON 其它情况下 OFF。
负标志	N	最左边位为 1 时置 ON。 其它情况下 OFF。

注意

执行 +(400) 时，错误标志将置 OFF。
如果加的结果， R 的内容为 0000，等于标志将置 ON。
如果加的结果有进位，进位标志将置 ON。
如果两个正数相加的结果为负（在 8000~FFFF 范围内），上溢出标志将置 ON。
如果两个负数相加的结果为正（在 0000~7FFF 范围内），下溢出标志将置 ON。
如果加的结果， R 的最左边的位的内容为 1，负标志位将置 ON。

例

在下面例子中，当 CIO000000 置 ON 时， D00100 和 D00110 将作为 4 位有符号二进制数相加，并且结果送到 D00120。

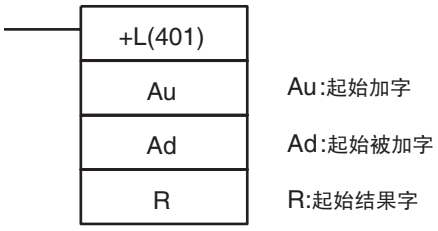


3-11-2 不带进位的有符号双字二进制加：+L(401)

用途

8 个数字 （双字）十六进制数据和 / 或常数相加。

梯形图符号



变化

变化	ON 条件时每次循环执行	+L(401)
	上升沿微分时执行一次	@+L(401)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

应用程序区

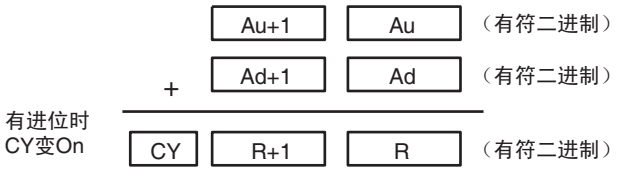
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	Au	Ad	R
CIO 区	CIO 0000 ～ CIO 6142		
工作区	W000 ～ W510		
抱迟位区	H000 ～ H510		
辅助位区	A000 ～ A958		A448 ～ A958
定时器区	T0000 ～ T4094		
计数器区	C0000 ～ C4094		
DM 区	D00000 ～ D32766		
无区号 DM 区	E00000 ～ E32766		
有区号 DM 区	En_00000 ～ En_32766 (n = 0 ～ C)		
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#00000000 ～ #FFFFFFFF （二进制）		---
数据寄存器	---		
索引寄存器	IR0 ～ IR15		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

描述

+L(401) 把 Au 和 Au+1， Ad 和 Ad+1 中的二进制值相加，并且把结果送给 R。



标志

名称	标志	处理
错误标志	ER	OFF
等于标志	=	结果为 0 时置 ON。 其它情况下 OFF
进位标志	CY	当结果导致进位时置 ON。 其它情况下 OFF。
上溢出标志	OF	当两个正数相加结果在 80000000~FFFFFFFF 之间时置 ON。 其它情况下 OFF
下溢出标志	UF	当两个负数相加结果在 00000000~7FFFFFFF 之间时置 ON。 其它情况下 OFF。
负标志	N	最左边位为 1 时置 ON。 其它情况下 OFF。

注意

执行 +L(401) 时，错误标志将置 OFF。

如果加的结果， R， R+1 的内容为 00000000，等于标志将置 ON。

如果加的结果有进位，进位标志将置 ON。

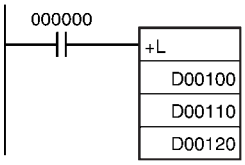
如果两个正数相加的结果为负（在 80000000~FFFFFFFF 范围内），上溢出标志将置 ON。

如果两个负数相加的结果为正（在 00000000~7FFFFFFF 范围内），下溢出标志将置 ON。

如果加的结果， R+1 的最左边的位的内容为 1，负标志位将置 ON。

例

当 CIO000000 置 ON 时，D00101，D00100 和 D00111，D00110 将作为 8 个数字有符号十六进制数相加，并且结果送到 D0121 和 D00120。

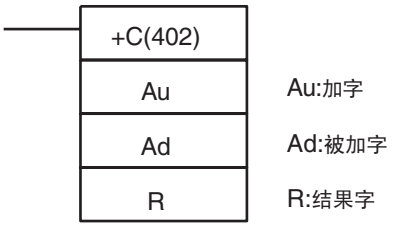


3-11-3 带进位的有符号二进制加：+C(402)

用途

4 个数字 （单字）十六进制数据和 / 或常数相加。

梯形图符号



变化

变化	ON 条件时每次循环执行	+C(402)
	上升沿微分时执行一次	@+C(402)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

应用程序区

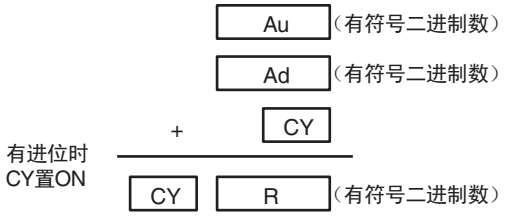
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	Au	Ad	R
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 DM 区	E00000 ～ E32767		
有区号 DM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #FFFF （二进制）		---
数据寄存器	DR0 ～ DR15		
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15		

描述

+C(402) 把 Au, Ad 和 CY 中的二进制数相加，并且把结果送给 R。



标志

名称	标志	处理
错误标志	ER	OFF
等于标志	=	相加结果为 0 时置 ON。 其它情况下 OFF。
进位标志	CY	当相加导致有进位时置 ON。 其它情况下 OFF。
上溢出标志	OF	当两个正数和 CY 相加的结果在 8000~FFFF 之间时置 ON。 其它情况下 OFF。
下溢出标志	UF	当两个负数和 CY 相加的结果在 0000~7FFF 之间时置 ON 其它情况下 OFF。
负标志	N	当结果的最左边位为 1 时置 ON。 其它情况下 OFF。

注意

执行 +C(402) 时，错误标志将置 OFF。

如果相加后， R 的内容为 0000，等于标志将置 ON。

如果相加导致进位，进位标志将置 ON。

如果两个正数和 CY 相加的结果为负（在 8000~FFFF 范围内），上溢出标志将置 ON。

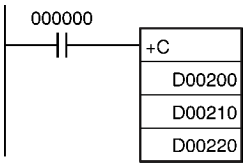
如果两个负数和 CY 相加的结果为正（在 0000~7FFF 范围内），下溢出标志将置 ON。

如果由于相加， R 的最左边的位的内容为 1，负标志位将置 ON。

注 执行清除进位 (CLC(041)) 指令来清除进位标志 (CY)。

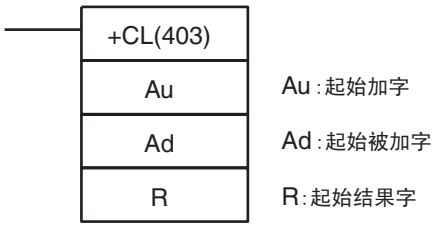
例

当 CIO 000000 置 ON 时， D00200， D00210 和 CY 将作为 4 个数字有符号十六进制数相加，并且结果送到 D00220。



3-11-4 带进位的有符号双字二进制加：+CL(403) 数相加。

梯形图符号



变化

变化	ON 条件时每次循环执行	+CL(403)
	上升沿微分时执行一次	@+CL(403)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

应用程序区

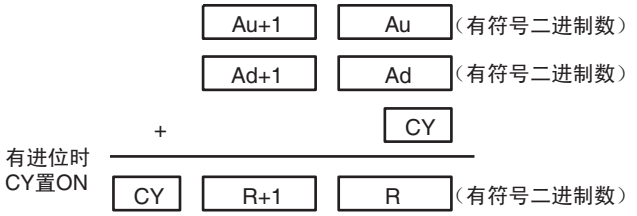
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	Au	Ad	R
CIO 区	CIO 0000 ～ CIO 6142		
工作区	W000 ～ W510		
保持位区	H000 ～ H510		
辅助位区	A000 ～ A958		A448 ～ A958
定时器区	T0000 ～ T4094		
计数器区	C0000 ～ C4094		
DM 区	D00000 ～ D32766		
无区号 DM 区	E00000 ～ E32766		
有区号 DM 区	En_00000 ～ En_32766 (n = 0 ～ C)		
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#00000000 ～ #FFFFFFFF （二进制）		---
数据寄存器	---		
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

描述

+CL(403) 把 Au 和 Au+1, Ad 和 Ad +1 及 CY 中的二进制数相加，并且把结果送给 R。



标志

名称	标志	处理
错误标志	ER	OFF
等于标志	=	结果为 0 时置 ON。 其它情况下 OFF。
进位标志	CY	结果有进位时置 ON。 其它情况下 OFF。
上溢出标志	OF	当两个正数和 CY 相加的结果在 80000000~FFFFFFFF 之间时置 ON。 其它情况下 OFF。
下溢出标志	UF	当两个负数和 CY 相加的结果在 00000000~7FFFFFFF 之间时置 ON。 其它情况下 OFF。
负标志	N	当结果的最左边位为 1 时置 ON。 其它情况下 OFF。

注意

执行 +CL(403) 时，错误标志将置 OFF。

如果相加后， R， R+1 的内容为 00000000，等于标志将置 ON。

如果相加结果有进位，进位标志将置 ON。

如果两个正数和 CY 相加的结果为负（在 80000000~FFFFFFFF 范围内），上溢出标志将置 ON。

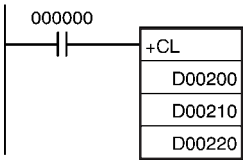
如果两个负数和 CY 相加的结果为正（在 00000000~7FFFFFFF 范围内），下溢出标志将置 ON。

如果由于相加， R+1 的最左边的位的内容为 1，负标志位将置 ON。

注 执行清除进位 (CLC(041)) 指令来清除进位标志 (CY)。

例

当 CIO 000000 置 ON 时，D00201，D00200 和 D00211，D00210 及 CY 将作为 8 个数字有符号十六进制数相加，并且结果送到 D00221 和 D00220。

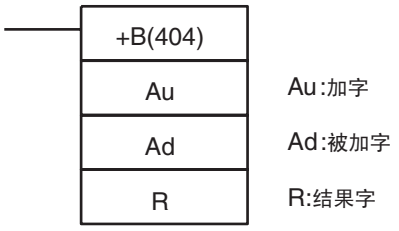


3-11-5 不带进位的 BCD 加：+B(404)

用途

4 个数字 （单字）有符号十六进制数和 / 或常数相除。

梯形图符号



变化

变化	ON 条件时每次循环执行	+B(404)
	上升沿微分时执行一次	@+B(404)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

应用程序区

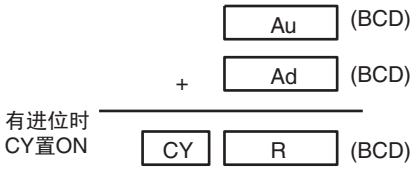
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	Au	Ad	R
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 DM 区	E00000 ～ E32767		
有区号 DM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	0000 ～ 9999 (BCD)		---
数据寄存器	DR0 ～ DR15		
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

描述

+B(404) 把 Au 和 Ad 中的 BCD 数相加，并且把结果送给 R。



标志

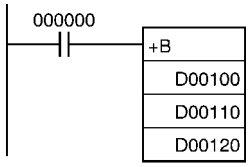
名称	标志	处理
错误标志	ER	当 Au 不是 BCD 码时置 ON。 当 Ad 不是 BCD 码时置 ON。 其它情况下 OFF。
等于标志	=	结果为 0 时置 ON。 其它情况下 OFF。
进位标志	CY	相加导致进位时置 ON。 其它情况下 OFF。

注意

如果 Au 或 Ad 不是 BCD 码，将出现错误并且错误标志将置 ON。
如果相加后， R 的内容为 0000，则等于标志将置 ON。
如果相加导致进位，进位标志将置 ON。

例

在下面例子中，当 CIO 000000 置 ON 时，D00100 和 D00110 将作为 4 个 BCD 数字相加，并且结果送到 D00120。

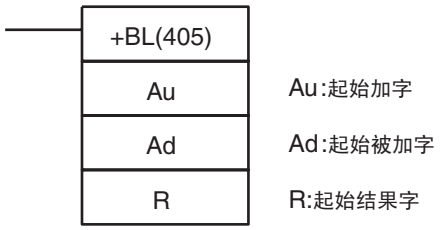


3-11-6 不带进位的双字 BCD 加：+BL(405)

用途

8 个数字 （单字） BCD 数据和 / 或常数相加。

梯形图符号



变化

变化	ON 条件时每次循环执行	+BL(405)
	上升沿微分时执行一次	@+BL(405)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

应用程序区

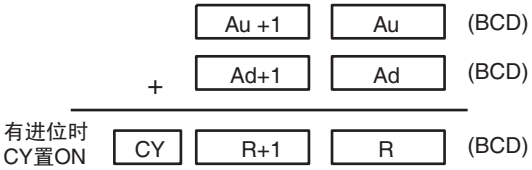
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	Au	Ad	R
CIO 区	CIO 0000 ～ CIO 6142		
工作区	W000 ～ W510		
保持位区	H000 ～ H510		
辅助位区	A000 ～ A958		A448 ～ A958
定时器区	T0000 ～ T4094		
计数器区	C0000 ～ C4094		
DM 区	D00000 ～ D32766		
无区号 DM 区	E00000 ～ E32766		
有区号 DM 区	En_00000 ～ En_32766 (n = 0 ～ C)		
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#00000000 ～ #99999999 (BCD)		---
数据寄存器	---		
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

描述

+BL(405)把Au和Au+1, Ad和Ad +1中的BCD数相加，并且把结果送给R，R+1。



标志

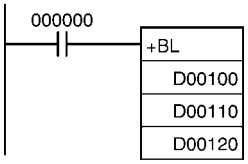
名称	标志	处理
错误标志	ER	当 Au， Au+1 不是 BCD 码时置 ON。 当 Ad， Ad+1 不是 BCD 码时置 ON。 其它情况下 OFF。
等于标志	=	结果为 0 时置 ON。 其它情况下 OFF。
进位标志	CY	结果有进位时置 ON。 其它情况下 OFF。

注意

如果 Au ,Au+1 或 Ad,Ad+1 不是 BCD 码，将出现错误并且错误标志将置 ON。
如果相加后， R,R+1 的内容为 00000000，则等于标志将置 ON。
如果相加导致进位，进位标志将置 ON。

例

在下面例子中，当 CIO 000000 置 ON 时， D00101 和 D00100， D00111 和 D00110 将作为 8 个 BCD 数字相加，并且结果送到 D00121 和 D00120。

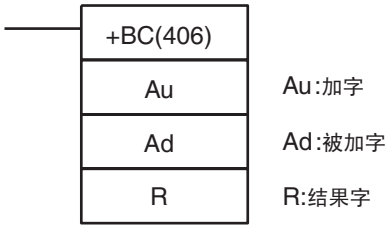


3-11-7 带进位的 BCD 加：+BC(406)

用途

4 个数字 （单字） BCD 码数据和 / 或常数及进位标志相加。

梯形图符号



变化

变化	ON 条件时每次循环执行	+BC(406)
	上升沿微分时执行一次	@+BC(406)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

应用程序区

块程序区	步程程序区	子程序区	中断任务
OK	OK	OK	OK

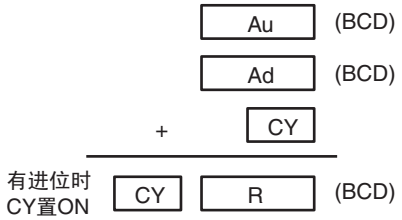
操作数规定

区域	Au	Ad	R
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 DM 区	E00000 ～ E32767		
有区号 DM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		

区域	Au	Ad	R
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ 9999 (BCD)		---
数据寄存器	DR0 ～ DR15		
索引寄存器	---		
使用索引寄存器 间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

注意

+BC(406) 把 Ad， Au 和 CY 中的 BCD 数相加，并且把结果送给 R。



标志

名称	标志	处理
错误标志	ER	Au 不是 BCD 码时置 ON。 Ad 不是 BCD 码时置 ON。 其它情况下 OFF。
等于标志	=	结果为 0 时置 ON。 其它情况下 OFF。
进位标志	CY	相加导致进位时置 ON。 其它情况下 OFF。

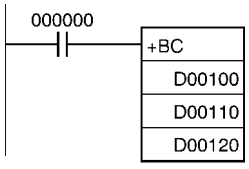
注意

如果 Au 或 Ad 不是 BCD 码，将出现错误并且错误标志将置 ON。
如果相加后， R 的内容为 0000，则等于标志将置 ON。
如果相加导致进位，进位标志将置 ON。

注 执行清除进位 (CLC(41)) 指令来清除进位标志 (CY)。

例

在下面例子中，当 CIO 000000 置 ON 时， D00100，D00110 和 CY 将作为 4 个 BCD 数字相加，并且结果送到 D00120。

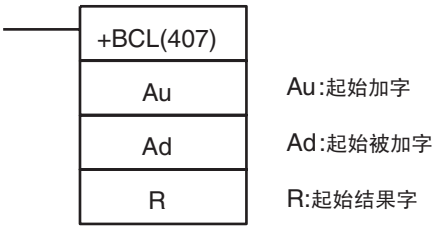


3-11-8 带进位的双字 BCD 加：+BCL(407)

用途

8 个数字 （双字） BCD 数据和 / 或常数及进位标志 (CY) 相加。

梯形图符号



变化

变化	ON 条件时每次循环执行	+BCL(407)
	上升沿微分时执行一次	@+BCL(407)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

应用程序区

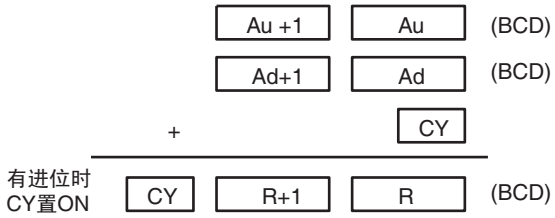
块程序区	步程程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	Au	Ad	R
CIO 区	CIO 0000 ～ CIO 6142		
工作区	W000 ～ W510		
保持位区	H000 ～ H510		
辅助位区	A000 ～ A958		A448 ～ A958
定时器区	T0000 ～ T4094		
计数器区	C0000 ～ C4094		
DM 区	D00000 ～ D32766		
无区号 DM 区	E00000 ～ E32766		
有区号 DM 区	En_00000 ～ En_32766 (n = 0 ～ C)		
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#00000000 ～ #99999999 (BCD)		---
数据寄存器	---		
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

描述

+BCL(407) 把 Au 和 Au+1, Ad 和 Ad +1 中的 BCD 数和 CY 相加，并且把结果送给 R， R+1。



标志

名称	标志	处理
错误标志	ER	当 Au， Au+1 不是 BCD 码时置 ON。 当 Ad， Ad+1 不是 BCD 码时置 ON。 其它情况下 OFF。
等于标志	=	结果为 0 时置 ON。 其它情况下 OFF。
进位标志	CY	结果导致进位时置 ON。 其它情况下 OFF。

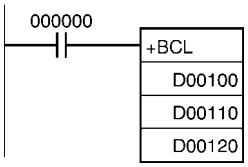
注意

如果 Au，Au+1 或 Ad，Ad+1 不是 BCD 码，将出现错误并且错误标志将置 ON。
如果相加后， R， R+1 的内容为 00000000，则等于标志将置 ON。
如果相加导致进位，进位标志将置 ON。

注 执行清除进位 (CLC(041)) 指令来清除进位标志 (CY)。

例

在下面例子中，当CIO 000000置ON时，D00101，D00100，D00111 ，D00110 及 CY 将作为 8 个 BCD 数字相加，并且结果送到 D00121 和 D00120。

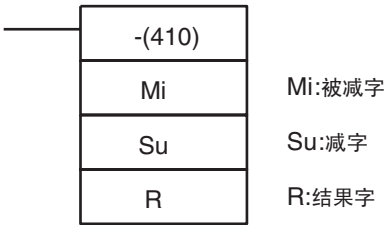


3-11-9 不带进位的有符号二进制减：-(410)

用途

4 个数字 （单字）十六进制数据和 / 或常数相加。

梯形图符号



变化

变化	ON 条件时每次循环执行	–(410)
	上升沿微分时执行一次	@–(410)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

应用程序区

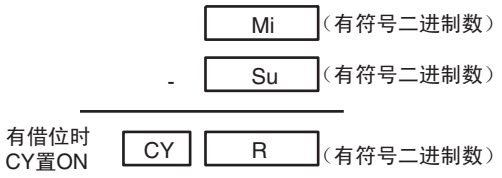
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	Mi	Su	R
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D0000 ～ D4095		
无区号 DM 区	E00000 ～ E32767		
有区号 DM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #FFFF （二进制）		---
数据寄存器	DR0 ～ DR15		
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

描述

-(400) 从 Mi 中减去 Su 中的二进制数，并且把结果送给 R。结果为负时，将 2 的补码送给 R。（处理 2 的补码的例子参阅 3-11-10 不带进位的有符号双字二进制减：-L(411)）。



标志

名称	标志	处理
错误标志	ER	OFF
等于标志	=	结果为 0 时置 ON。 其它情况下 OFF。

名称	标志	处理
进位标志	CY	相减导致借位时置 ON。 其它情况下 OFF。
上溢出标志	OF	当从一个正数减去一个负数的 结果在 8000~FFFF 之间时置 ON。 其它情况下 OFF。
下溢出标志	UF	当从一个负数减去一个正数的 结果在 0000~7FFF 之间时置 ON 其它情况下 OFF。
负标志	N	当结果的最左边位为 1 时置 ON。 其它情况下 OFF。

注意

执行 -(410) 时，错误标志将置 OFF。

如果由于相减， R 的内容为 0000，等于标志将置 ON。

如果相减导致借位，则进位标志将置 ON。

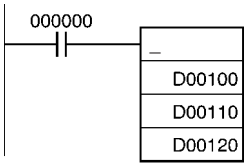
当从一个正数减去一个负数的结果为负数 （在 8000~FFFF 之间），则上溢出标志将时置 ON。

如果从一个负数减去一个正数的结果为正数（在 0000~7FFF 范围内），下溢出标志将置 ON。

如果由于相减， R 的最左边的位的内容为 1，负标志位将置 ON。

例

在下面例子中，当 CIO000000 置 ON 时，D00110 将作为 4 数字有符号二进制从 D00100 中减去，并且结果送到 D00120。



3-11-10 不带进位的有符号双字二进制减：-L(411)

用途

8 个数字 （双字）十六进制数据和 / 或常数相减

梯形图符号

变化

变化	ON 条件时每次循环执行	–L(411)
	上升沿微分时执行一次	@–L(411)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

应用程序区

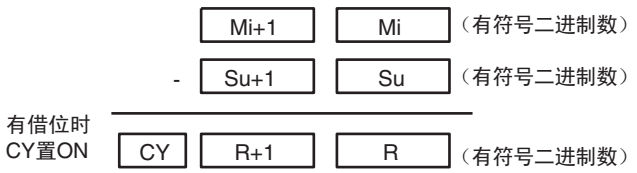
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	Mi	Su	R
CIO 区	CIO 0000 ～ CIO 6142		
工作区	W000 ～ W510		
保持位区	H000 ～ H510		
辅助位区	A000 ～ A958		A448 ～ A958
定时器区	T0000 ～ T4094		
计数器区	C0000 ～ C4094		
DM 区	D00000 ～ D32766		
无区号 DM 区	E00000 ～ E32766		
有区号 DM 区	En_00000 ～ En_32766 (n = 0 ～ C)		
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#00000000 ～ #FFFFFFFF （二进制）		---
数据寄存器	---		
索引寄存器	IR0 ～ IR15		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

注意

-L(411)把Mi和Mi+1中减去Su和Su+1中的二进制数并且把结果送给R，R+1。当结果为负数时，结果作为 2 的补码送给 R 和 R+1。



标志

名称	标志	处理
错误标志	ER	OFF
等于标志	=	结果为 0 时置 ON。 其它情况下 OFF。
进位标志	CY	当结果导致借位时置 ON。 其它情况下 OFF。
上溢出标志	OF	当从一个正数减去一个负数的结果在 80000000~FFFFFFFF 之间时置 ON。 其它情况下 OFF。

名称	标志	处理
下溢出标志	UF	当从一个负数中减去一个正数的结果在 00000000~7FFFFFFF 之间时置 ON。 其它情况下 OFF。
负标志	N	当结果的最左边位为 1 时置 ON。 其它情况下 OFF。

注意

执行 -L(411) 时，错误标志将置 OFF。

如果由于相减， R， R+1 的内容为 0000000，等于标志将置 ON。

如果相减导致借位，则进位标志将置 ON。

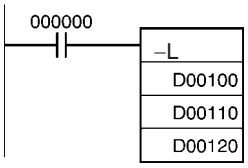
如果从一个正数中减去一个负数的结果为负 （在 80000000~FFFFFFFF 范围内），上溢出标志将置 ON。

如果当从一个负数中减去一个正数的结果为正（在 00000000~7FFFFFFF 范围内），下溢出标志将置 ON。

如果由于相减， R+1 的最左边的位的内容为 1，负标志位将置 ON。

例

在下面的例子中，当 CIO000000 置 ON 时，D00111 和 D00110 将作为 8 个数字有符号二进制数从 D00101 和 D00100 中减去，并且结果送到 D0121 和 D00120。



例

如果相减的结果是负数（Mi<Su 或 Mi+1,Mi<Su+1,Su），则结果作为 2 的补码输出，并且进位标志 (CY) 将置 ON，表明相减的结果为负数。为把 2 的补码转换成实际的数字，需要一个从零减去结果，使用进位标志 (CY) 作为执行条件的指令。

注 2 的补码

2 的补码是用 1 减去每一个二进制数并且结果加 1 得到的数。例如，1101 的 2 的补码计算如下：1111（F 十六进制）-1101（十六进制）+1（十六进制）=0011（3 十六进制）。 3039（十六进制）的 2 的补码如下计算：FFFF（十六进制）-3039（十六进制） +0001 （十六进制） =CFC7（十六进制）。所以，如果是 4 位数字十六进制数， 2 的补码可如下计算：FFFF （十六进制） -a （十六进制） +0001（十六进制）=b（十六进制）。为了从 2 的补码 b（十六进制）得到实际的数字：a（十六进制）=10000（十六进制）-b（十六进制）。例如，要从 2 的补码 CFC7 （十六进制）得到实际的数字：10000 （十六进制） -CFC7=3039。

例1

	有符号数	无符号数
FFFF Hex →	-1	65535
-) 0001 Hex →	-) +1	-) -1
FFFE Hex →	-2见注1	65534 见注2
负标志ON进位标志OFF		

例2

	有符号数	无符号数
FFFD Hex →	-3	65533
-) FFFF Hex →	-) -1	-) 65535
FFFE Hex →	-2见注3	65534见注4
负标志ON进位标志OFF		

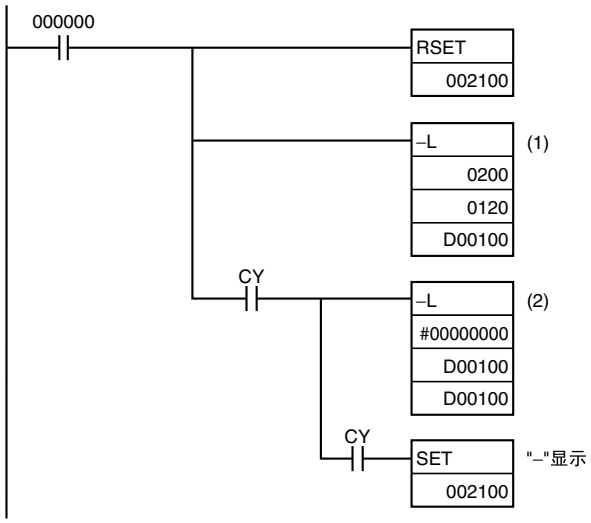
注

1. 因为负标志置ON，结果（FFFE十六进制）是负数（2的补码）因此是-2。
2. 因为进位标志置OFF，结果（FFFE十六进制）是无符号正数65534。
3. 因负标志置ON，结果（FFFE，十六进制）是负数（2的补码），因此是-2。
4. 因为进位标志置ON，结果（FFFE，十六进制）是负数（2的补码），当转换实际数时变为-2。

程序举例

20F55A10-B8A360E3=-97AE06D3。

在本例中，CIO 0201 和 CIO 0200 中的 8 位数字二进制数减去 CIO 0121 和 CIO 0120 中的数，结果以 8 位数字二进制的形式送到 D00101 和 D00100，如果结果为负， (2) 中的指令将执行，实际的结果将送到 D00101 和 D00100。



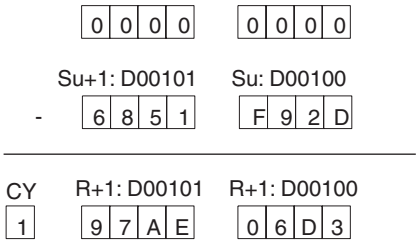
在1的减法

	Mi+1: CIO 0201	Mi: CIO 0200
	2 0 F 5	5 A 1 0
	Su+1: CIO 0121	Su: CIO 0120
-	B 8 A 3	6 0 E 3

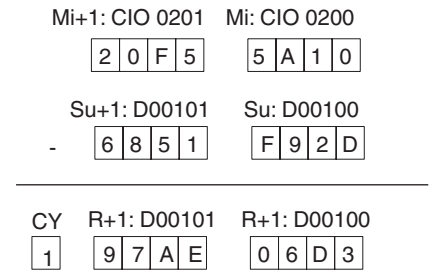
CY	R+1: D00101	R+1: D00100
1	6 8 5 1	F 9 2 D

进位标志 (CY) 为 ON，因此结果被 00000000 减后得到实际的数字。

在2的减法



最终相减结果



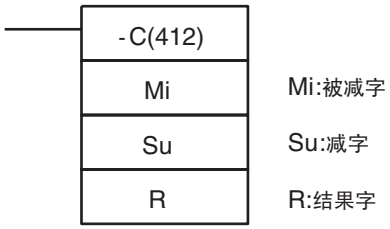
进位标志 (CY) 置 ON，因此实际数字为 -97AE06D3，因为 D00101 和 D00100 是负数， CY 用来置 CIO 002100 为 ON 以表示负值。

3-11-11 带进位的有符号二进制减：-C(412)

用途

4 个数字 （单字）十六进制数据和 / 或常数及进位标志 (CY) 相减。

梯形图符号



变化

变化	ON 条件时每次循环执行	–C(412)
	上升沿微分时执行一次	@–C(412)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

应用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

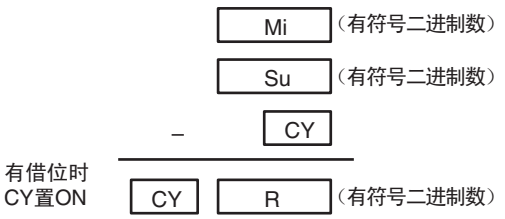
操作数规定

区域	Mi	Su	R
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		A448 ～ A959

区域	Mi	Su	R
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 DM 区	E00000 ～ E32767		
有区号 DM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #FFFF （二进制）		---
数据寄存器	DR0 ～ DR15		
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

描述

-C(412) 从 Mi 中减去 Su 中的二进制数和 CY，并且把结果送给 R。结果为负时，将 2 的补码送给 R。



标志

名称	标志	处理
错误标志	ER	OFF
等于标志	=	当相减结果为 0 时置 ON。 其它情况下 OFF。
进位标志	CY	当相减导致借位时置 ON。 其它情况下 OFF。
上溢出标志	OF	当从一个正数减去一个负数和 CY 的结果在 8000~FFFF 之间时置 ON。 其它情况下 OFF。
下溢出标志	UF	当从一个负数减去一个正数和 CY 的结果在 0000~7FFF 之间时置 ON 其它情况下 OFF。
负标志	N	当结果的最左边位为 1 时置 ON。 其它情况下 OFF。

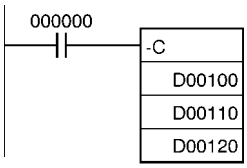
注意

执行 -C(412) 时，错误标志将置 OFF。
如果由于相减， R 的内容为 0000，等于标志将置 ON。
如果相减导致借位，则进位标志将置 ON。
当从一个正数减去一个负数和 CY 的结果为负数 （在 8000~FFFF 之间），则上溢出标志将时置 ON。
如果从一个负数减去一个正数和 CY 的结果为正数（在 0000~7FFF 范围内），下溢出标志将置 ON。
如果由于相减， R 的最左边的位的内容为 1，负标志位将置 ON。

注 执行清除进位 (CLC(041)) 指令来清除进位标志 (CY)。

例

在下面例子中，当 CIO000000 置 ON 时，以 4 数字有符号二进制形式从 D00100 中减去 D00110 和 CY，并且将结果送到 D00120。

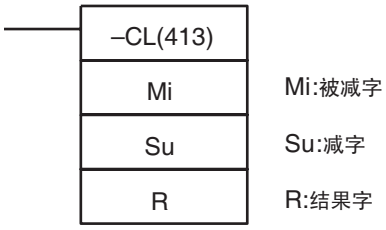


3-11-12 带进位的有符号双字二进制减：-CL(413)

用途

8 个数字 （双字）十六进制数据和 / 或常数相减及进位标志 (CY)。

梯形图符号



变化

变化	ON 条件时每次循环执行	–CL(413)
	上升沿微分时执行一次	@–CL(413)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

应用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

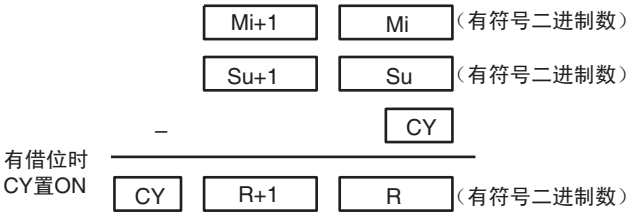
操作数规定

区域	Mi	Su	R
CIO 区	CIO 0000 ～ CIO 6142		
工作区	W000 ～ W510		
保持位区	H000 ～ H510		
辅助位区	A000 ～ A958		A448 ～ A958
定时器区	T0000 ～ T4094		
计数器区	C0000 ～ C4094		

区域	Mi	Su	R
DM 区	D00000 ～ D32766		
无区号 DM 区	E00000 ～ E32766		
有区号 DM 区	En_00000 ～ En_32766 (n = 0 ～ C)		
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#00000000 ～ #FFFFFFFF （二进制）		---
数据寄存器	---		
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

描述

-CL(413) 把 Mi 和 Mi+1 中减去 Su 和 Su+1 中的二进制数及 CY，并且把结果送给 R， R+1。当结果为负数时，结果作为 2 的补码送给 R 和 R+1。



标志

名称	标志	处理
错误标志	ER	OFF
等于标志	=	结果为 0 时置 ON。 其它情况下 OFF。
进位标志	CY	当结果为 0 时置 ON。 其它情况下 OFF。
上溢出标志	OF	当从一个正数中减去一个负数和 CY 的 结果在 80000000~FFFFFFFF 之间时置 ON。 其它情况下 OFF。
下溢出标志	UF	当从一个负数中减去一个正数和 CY 的 结果在 00000000~7FFFFFFF 之间时置 ON。 其它情况下 OFF。
负标志	N	当结果的最左边位为 1 时置 ON。 其它情况下 OFF。

注意

执行 -CL(413) 时，错误标志将置 OFF。

如果由于相减， R， R+1 的内容为 0000000，等于标志将置 ON。

如果相减导致借位，则进位标志将置 ON。

如果从一个正数中减去一个负数和 CY 的结果为负（在 80000000~FFFFFFFF 范围内），上溢出标志将置 ON。

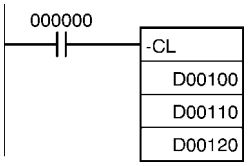
如果当从一个负数中减去一个正数和CY的结果为正（在00000000~7FFFFFFF 范围内），下溢出标志将置 ON。

如果由于相减， R+1 的最左边的位为 1，负标志位将置 ON。

注 执行清除进位 (CLC(041)) 指令来清除进位标志 (CY)。

例

在下面的例子中，当 CIO000000 置 ON 时，以 8 位数字有符号二进制数形式从 D00101 和 D00100 中减去 D00101，D00100 和 CY，并且结果送到 D0121 和 D00120。



如果相减的结果是负数（Mi<Su 或 Mi+1,Mi<Su+1,Su），则结果作为 2 的补码输出，并且进位标志 (CY) 将置 ON。为把 2 的补码转换成实际的数字，需要一个从零减去结果，将进位标志 (CY) 作为执行条件的指令。进位标志置 ON，表明相减结果为负数。

注 2 的补码

2 的补码是用 1 减去每一个二进制数并且结果加 1 得到。

例 : 二进制数 1101 的 2 的补码计算如下：

1111(F hex)-1101(D hex)+1(1 hex)=0011(3 hex)。

例 :4 位数字十六进制数 3039 的 2 的补码可如下：

FFFFhex-3039 hex+0001hex=CFC7 hex。

因此 4 位数字十六进制数 “a” 的 2 的补码可如下：

FFFFhex-a hex+0001 hex=b hex。

从 2 的补码 “b” 得到真实的数字：

a hex =10000 hex-b hex

例 : 从 2 的补码 CFC7 得到真实的数字：

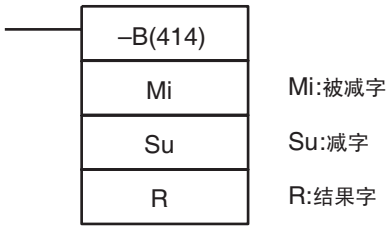
10000 hex-CFC7 hex =3039 hex

3-11-13 不带进位的 BCD 减：-B(414)

用途

4 位 （单字） BCD 和 / 或常数相减。

梯形图符号



变化

变化	ON 条件时每次循环执行	–B(414)
	上升沿微分时执行一次	@–B(414)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

应用程序区

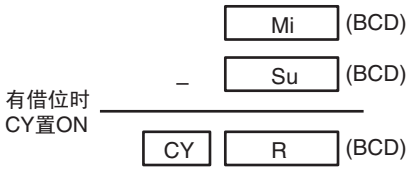
块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	Mi	Su	R
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 DM 区	E00000 ～ E32767		
有区号 DM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	0000 ～ 9999 (BCD)		---
数据寄存器	DR0 ～ DR15		
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

描述

-B(414) 从 Mi 中减去 Su 中的 BCD，并且把结果送给 R。如果相减的结果为负时，则结果以 10 的补码输出。



标志

名称	标志	处理
错误标志	ER	当 Mi 不是 BCD 码时置 ON。 当 Su 不是 BCD 码时置 ON。 其它情况下 OFF。
等于标志	=	当结果为 0 时置 ON。 其它情况下 OFF。
进位标志	CY	当相减导致借位时置 ON。 其它情况下 OFF。

注意

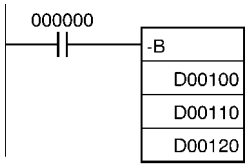
如果 Mi 和 / 或 Su 不是 BCD 码，将出现一个错误并且错误标志将置 ON。

如果相减后， R 的内容为 0000，则等于标志将置 ON。

如果相减导致进位，进位标志将置 ON。

例

在下面例子中，当 CIO 000000 置 ON 时，以 4 位 BCD 码形式从 D00100 减去 D00110，并且结果送到 D00120。

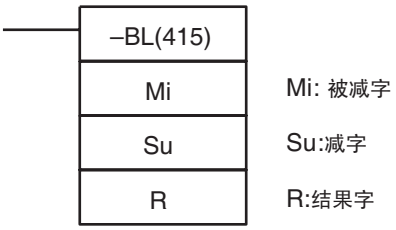


3-11-14 不带进位的双字 BCD 减：-BL(415)

用途

8 位 （双字） BCD 数和 / 或常数相减。

梯形图符号



变化

变化	ON 条件时每次循环执行	BL(415)
	上升沿微分时执行一次	@BL(415)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

应用程序区

块程序区	步程序区	子程序区	中断任务
OK	OK	OK	OK

操作数规定

区域	Mi	Su	R
CIO 区	CIO 0000 ～ CIO 6142		
工作区	W000 ～ W510		
保持位区	H000 ～ H510		
辅助位区	A000 ～ A958		A448 ～ A958
定时器区	T0000 ～ T4094		
计数器区	C0000 ～ C4094		
DM 区	D00000 ～ D32766		
无区号 DM 区	E00000 ～ E32766		
有区号 DM 区	En_00000 ～ En_32766 (n = 0 ～ C)		
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#00000000 ～ #99999999 (BCD)		---
数据寄存器	---		
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

描述

-BL(415) 把 Mi 和 Mi+1 中减去 Su 和 Su+1 中的 BCD 码值，并且把结果送给 R,R+1。当结果为负数时，结果作为 10 的补码送给 R 和 R+1。

标志

名称	标志	处理
错误标志	ER	当 Au， Au+1 不是 BCD 码时置 ON。 当 Ad， Ad+1 不是 BCD 码时置 ON。 其它情况下 OFF。

名称	标志	处理
等于标志	=	结果为 0 时置 ON。 其它情况下 OFF。
进位标志	CY	结果有进位时置 ON。 其它情况下 OFF。

注意

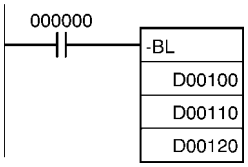
如果 Mi 和 Mi+1 与 / 或 Su 和 Su+1 不是 BCD 码，并且将产生错误并且错误标志将置 ON。

如果相减， R,R+1 的内容为 00000000，则等于标志将置 ON。

如果相减导致进位，则进位标志将置 ON。

例

在下面例子中，当 CIO 000000 置 ON 时，以 8 位 BCD 码形式从 D00101 和 D00100 中减去 D00111 和 D00110，并且结果送到 D00121 和 D00120。



如果相减的结果是负数 （Mi<Su 或 Mi+1,Mi<Su+1,Su），则结果作为 10 的补码输出，并且进位标志 (CY) 将置 ON。为把 10 的补码转换成真实的数字，需要一个从零减去结果，将进位标志 (CY) 作为执行条件的指令。进位标志置ON，表明相减结果为负数。

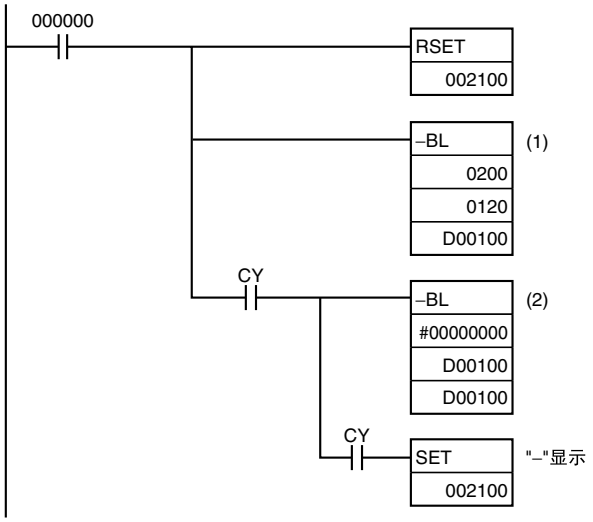
注 **10 的补码**

10 的补码是用 9 减去每一位并把结果加 1 得到的结果。例如，7556 的 10 的补码计算如下：9999-7556+1=2444，对于一个四位数， A 的 10 的补码是 9999-A+1=B。为从 10 的补码 B 得到真实的数字：A=10000-B。例如，为从 10 的补码 2444 得到真实的数字：10000-2444=7556。

程序举例

9,583,960-17,072,641= -7,488,681。

在本例中，从 CIO 0201 和 CIO 0200 中减去 CIO 0121 和 CIO0120 中的 8 位 BCD 码，结果以 8 位 BCD 码的形式送给 D00101 和 D00100。结果为负数，因此（2）中的指令将被执行，并且真实的值将被送到 D00101 和 D00100。



在1处的减法

Mi+1: CIO 0201 Mi: CIO 0200

0	9	5	8

3	9	6	0

Su+1: CIO 0121 Su: CIO 0120

– | 1 | 7 | 0 | 7 |

2	6	4	1

09583960 + (100000000 – 17072641)

CY R+1: D00101 R+1: D00100

1

9	2	5	1

1	3	1	9

进位标志 (CY) 为 ON，因此结果从 00000000 中减。

在2处的减法

0	0	0	0

0	0	0	0

Su+1: D00101 Su: D00100

– | 9 | 2 | 5 | 1 |

1	3	1	9

00000000 + (100000000 – 92511319)

CY R+1: D00101 R+1: D00100

1

0	7	4	8

8	6	8	1

最终相减结果

Mi+1: CIO 0201 Mi: CIO 0200

2	0	F	5

5	A	1	0

Su+1: D00101 Su: D00100

– | 6 | 8 | 5 | 1 |

F	9	2	D

CY R+1: D00101 R+1: D00100

1

0	7	4	8

8	6	8	1

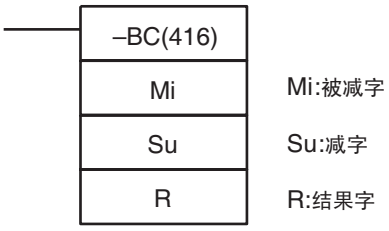
进位标志 (CY) 置 ON，因此实际数字为 -7，488，681，因为 D00101 和 D00100 是负数， CY 用来置 CIO 002100 位为 ON 以表示负值。

3-11-15 带进位的 BCD 减：-BC(416)

用途

4 位 （单字） BCD 数与 / 或常数及进位标志 (CY) 相减。

梯形图符号



变化

变化	ON 条件时，每个周期执行	–BC(416)
	上升沿微分时执行一次	@–BC(416)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

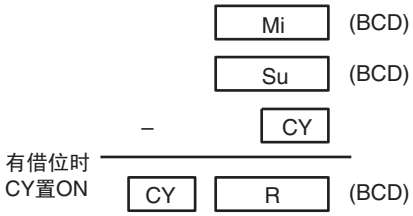
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数说明

区域	Mi	Su	R
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ D32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #9999 (BCD)		---
数据寄存器	DR0 ～ DR15		
索引寄存器	---		
适用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

-BC(416) 从 Mi 中减去 Su 中的 BCD 值和 CY。并把结果为负数，则作为 2 的补码送给 R。



标志

名称	标识	操作
错误标志	ER	当 Mi 不是 BCD 码时置 ON。 当 Su 不是 BCD 码时置 ON。 所有其它情况时置 OFF。
等于标志	=	结果为 0 时置 ON。 所有其它情况时置 OFF。
进位标志	CY	当相减导致借位时置 ON。 所有其它情况时置 OFF。

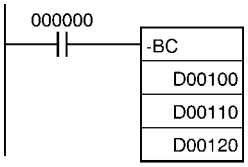
注意

如果 Mi 与 / 或 Su 不是 BCD 码，将产生一个错误，并且错误标识将置 ON。
如果由于相减， R 的内容为 0000，则等于标志将置 ON。
如果相减导致错位，则进位标志将置 ON。

注 执行清除进位 (CLC(041)) 指令来清除进位标志 (CY)。

例

在下例中，当 CIO 000000 为 ON 时，以 4 位 BCD 码形式从 D00100 中减去 D00110 和 CY, 并把结果送给 D00120。

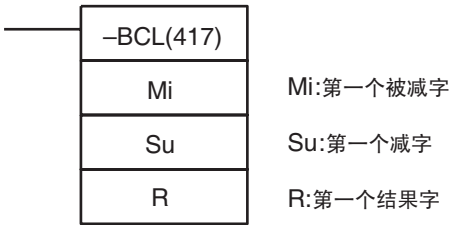


3-11-16 带进位的双字 BCD 减：-BCL(417)

用途

8 个数字 （双字） BCD 数据和 / 或常数及进位标志 (CY) 相减。

梯形图符号



变化

变化	ON 条件时，每个周期执行	–BCL(417)
	上升沿微分时执行一次	@–BCL(417)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断程序
OK	OK	OK	OK

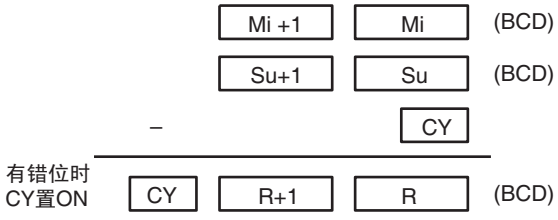
操作数定义

区域	Mi	Su	R
CIO 区	CIO 0000 ～ CIO 6142		
工作区	W000 ～ W510		
保持位区	H000 ～ H510		
辅助位区	A000 ～ A958		A448 ～ A958
定时器区	T0000 ～ T4094		
计数器区	C0000 ～ C4094		
DM 区	D00000 ～ D32766		
无区号 EM 区	E00000 ～ E32766		
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#00000000 ～ #99999999 (BCD)		---
数据寄存器	---		
索引寄存器	---		
适用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

-BCD(417) 从 Mi 和 Mi+1 中减去 Su,Su+1 的 BCD 值和 CY, 并把结果输出给 R,R+1。

如果结果为负数，则作为 10 的补码输出到 R,R+1。



标志

名称	标识	操作
错误标志	ER	当 Mi 与 / 或 Mi+1 不是 BCD 码时置 ON。 当 Su 与 / 或 Su+1 不是 BCD 码时置 ON。 所有其它情况时置 OFF。
等于标志	=	结果为 0 时置 ON。 所有其它情况时置 OFF。
进位标志	CY	当相减导致借位时置 ON。 所有其它情况时置 OFF。

注意

如果 Mi,Mi+1 与 / 或 Su,Su+1 不是 BCD 码，将产生错误并且错误标志将置 ON。

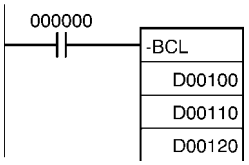
如果由于相减， R,R+1 的内容为 00000000，则等于标志将置 ON。

如果由于相减导致错位，则进位标志将置 ON。

注 执行清除进位 (CLC(041)) 指令来清除进位标志 (CY)。

例

在下例中，当 CIO000000 为 ON 时，以 8 位 BCD 值形式从 D00101 和 D00100 减去 D00111， D00110 和 CY，并把结果输出到 D00121 和 D00120。



如果相减的结果是一个负数 （Mi<Su 或 Mi+1,Mi<Su+1,Su），结果作为 10 的补码输出，进位标志 (CY) 将置 ON. 为把 10 的补码转换成真实的数字，需要一个从 0 减去该结果的程序，进位标志 (CY) 作为输入条件。进位标志置 ON 表示相减结果为负数。

注 10 的补码

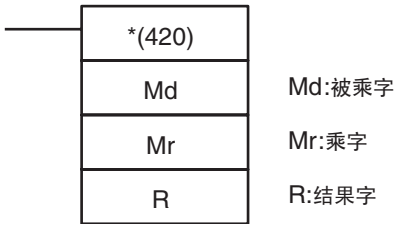
10 的补码是用 9 减去每一位并把结果加 1 得到的值。例如， 7556 的 10 的补码计算如下: 9999-7556+1=2444。对于一个四位数，A 的 10 的补码时 9999-A+1=B, 为从 10 的补码得到真实的数，A=10000-B. 例如，为从 10 的补码得到真实的数，10000-2444=7556。

3-11-17 有符号二进制乘：*(420)

用途

4 位有符号十六进制数和 / 或常数的乘法。

梯形图符号



变化

变化	ON 条件时，每个周期执行	*(420)
	上升沿微分时执行一次	@*(420)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数说明

区域	Md	Mr	R
CIO 区	CIO 0000 ～ CIO 6143		CIO 0000 ～ CIO 6142
工作区	W000 ～ W511		W000 ～ W510
保持位区	H000 ～ H511		H000 ～ H510
辅助位区	A000 ～ A959		A448 ～ A958
定时器区	T0000 ～ T4095		T0000 ～ T4094
计数器区	C0000 ～ C4095		C0000 ～ C4094
DM 区	D00000 ～ D32767		D00000 ～ D32766
无区号 EM 区	E00000 ～ E32767		E00000 ～ E32766
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		En_00000 ～ En_32766 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #FFFF （二进制）		---
数据寄存器	DR0 ～ DR15		---
索引寄存器	---		
适用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

*(420) 把 Md 和 Mr 中有符号二进制数相乘，并把结果输出给 R,R+1。

标志

名称	标识	操作
错误标志	ER	OFF
等于标志	=	结果为 0 时置 ON。 所有其它情况时置 OFF。
负标志	N	当结果的最左边位为 1 时置 ON。 所有其它情况时置 OFF。

注意

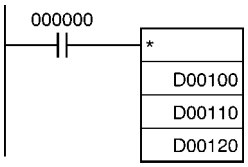
执行 *(420) 时，错误标志将置 OFF。

如果由于相乘， R 的内容为 0000，则等于标志将置 ON。

如果由于相乘， R+1 和 R 的最左边位的内容为 1，则负标志位将置 ON。

例

在下例中，当 CIO 000000 为 ON 时， D00100 和 D00110 以 4 位有符号十六进制数形式相乘，并把结果送给 D00120。



3-11-18 有符号双字二进制乘：*L(421)

用途

8 个数字有符号十六进制数和 / 或常数的相乘。

梯形图符号

变化

变化	ON 条件时，每个周期执行	*L(421)
	上升沿微分时执行一次	@*L(421)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

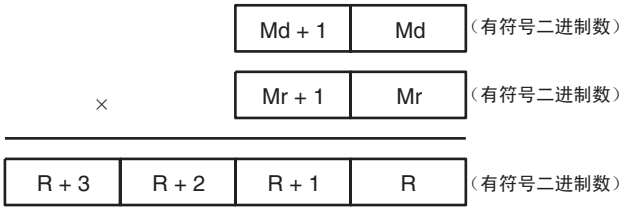
操作数定义

区域	Md	Mr	R
CIO 区	CIO 0000 ～ CIO 6142		CIO 0000 ～ CIO 6140
工作区	W000 ～ W510		W000 ～ W508
保持位区	H000 ～ H510		H000 ～ H508
辅助位区	A000 ～ A958		A448 ～ A956
定时器区	T0000 ～ T4094		T0000 ～ T4092

区域	Md	Mr	R
计数器区	C0000 ～ C4094		C0000 ～ C4092
DM 区	D00000 ～ D32766		D00000 ～ D32764
无区号 EM 区	E00000 ～ E32766		E00000 ～ E32764
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)		En_00000 ～ En_32764 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#00000000 ～ #FFFFFFFF （二进制）		---
数据寄存器	---		
索引寄存器	---		
适用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

*L(421) 把 Md 和 Md+1 与 Mr 和 Mr+1 中的符号二进制数相乘，并把结果输出到 R,R+1,R+2 和 R+3。



标志

名称	标识	操作
错误标志	ER	OFF
等于标志	=	结果为 0 时置 ON。 所有其它情况时置 OFF。
负标志	N	当结果的最左边位为 1 时置 ON。 所有其它情况时置 OFF。

注意

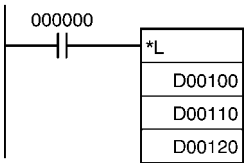
执行 *L(421) 时，错误标志将置 OFF。

如果由于相乘， R,R+1,R+2,R+3 的内容均为 0000，则等于标志将置 ON。

如果由于相乘， R+3 的最左边位的内容为 1，则负标志位将置 ON。

例

在下例中，当 CIO 000000 为 ON 时，D00101 和 D00100 及 D00111 和 D00110 将以8位有符号十六进制数形式相乘，并把结果送给D00123，D00i22，D00121 和 D00120。

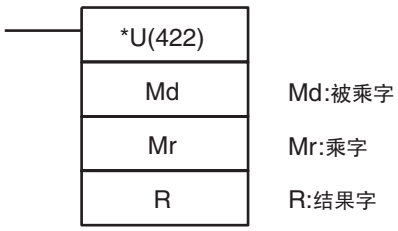


3-11-19 无符号二进制乘：*U(422)

用途

4 个数字无符号十六进制数和 / 或常数的相乘。

梯形图符号



变化

变化	ON 条件时，每个周期执行	*U(422)
	上升沿微分时执行一次	@*U(422)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数说明

区域	Md	Mr	R
CIO 区	CIO 0000 ～ CIO 6143		CIO 0000 ～ CIO 6142
工作区	W000 ～ W511		W000 ～ W510
保持位区	H000 ～ H511		H000 ～ H510
辅助位区	A000 ～ A959		A448 ～ A958
定时器区	T0000 ～ T4095		T0000 ～ T4094
计数器区	C0000 ～ C4095		C0000 ～ C4094
DM 区	D00000 ～ D32767		D00000 ～ D32766
无区号 EM 区	E00000 ～ E32767		E00000 ～ E32766
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		En_00000 t ～ En_32766 (n = 0 ～ C)

区域	Md	Mr	R
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_ 32767 (n = 0 ～ C)		
常数	#0000 ～ #FFFF （二进制）		---
数据寄存武器	DR0 ～ DR15		---
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

*U(420) 把 Md 和 Mr 中的二进制数相乘并把结果输出到 R,R+1。

标志

名称	标识	操作
错误标志	ER	OFF
等于标志	=	结果为 0 时置 ON。 所有其它情况时置 OFF。
负标志	N	当结果得最左边位为 1 时置 ON。 所有其它情况时置 OFF。

注意

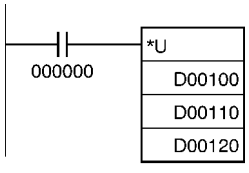
执行 *U(422) 时，错误标志将置 OFF.

如果由于相乘， R,R+1 的内容为 0000，则等于标志将置 ON。

如果由于相乘， R+1 的最左边内容为 1，则负标志将置 ON。

例

在下例当中，当 CIO 000000 为 ON 时， D00100 和 D00110 将作为 4 个 数字无符号二进制相乘，并把 结果输出到 D00121 和 D00120.

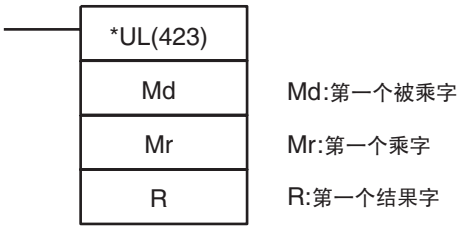


3-11-20 无符号双字二进制乘：*UL(423)

用途

把 8 个数字无符号十六进制数和 / 或常数相乘。

梯形图符号



变化

变化	ON 条件时，每个周期执行	*UL(423)
	上升沿微分时执行一次	@*UL(423)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断程序
OK	OK	OK	OK

操作数定义

区域	Md	Mr	R
CIO 区	CIO 0000 ～ CIO 6142		CIO 0000 ～ CIO 6140
工作区	W000 ～ W510		W000 ～ W508
保持位区	H000 ～ H510		H000 ～ H508
辅助位区	A000 ～ A958		A448 ～ A956
定时器区	T0000 ～ T4094		T0000 ～ T4092
计数器区	C0000 ～ C4094		C0000 ～ C4092
DM 区	D00000 ～ D32766		D00000 ～ D32764
无区号 EM 区	E00000 ～ E32766		E00000 ～ E32764
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)		En_00000 ～ En_32764 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#00000000 ～ #FFFFFFFF （二进制）		---
数据寄存器	---		

区域	Md	Mr	R
索引寄存器	---		
使用索引寄存器 的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

*UL(423) 把 Md 和 Md+1 与 Mr 和 Mr+1 种的无符号二进制数相乘并把结果送给 R,R+1,R+2 和 R+3。

标志

名称	标识	操作
错误标志	ER	OFF
等于标志	=	结果为 0 时置 ON。 所有其它情况时置 OFF。
负标志	N	结果的最左边为 1 时置 ON。 所有其它情况时置 OFF。

注意

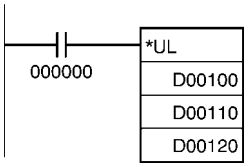
执行 *UL(423) 时，错误标志将置 OFF。

如果由于相乘， R,R+1,R+2,R+3 的内容全部为 0000，则等于标志将置 ON。

如果由于相乘， R+3 的最左边位的内容为 1，则负标志将置 ON。

例

在下例中，当 CIO 000000 为 ON 时， D00101,D00100 及 D00111 和 D00110 将以 8 位无符号二进制形式相乘，并把结果输出给 D00123,D00122,D00121 和 D00120。



3-11-21 BCD 乘法：*B(424)

用途

4 个数字 （单字） BCD 数据和 / 或常数相乘。

梯形图符号

变化

变化	ON 条件时，每个周期执行	*B(424)
	上升沿微分时执行一次	@*B(424)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断程序
OK	OK	OK	OK

操作数定义

区域	Md	Mr	R
CIO 区	CIO 0000 ～ CIO 6143		CIO 0000 ～ CIO 6142
工作区	W000 ～ W511		W000 ～ W510
保持位区	H000 ～ H511		H000 ～ H510
辅助位区	A000 ～ A959		A448 ～ A958
定时器区	T0000 ～ T4095		T0000 ～ T4094
计数器区	C0000 ～ C4095		C0000 ～ C4094
DM 区	D00000 ～ D32767		D00000 ～ D32766
无区号 EM 区	E00000 ～ E32767		E00000 ～ E32766
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		En_00000 ～ En_32766 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #9999 (BCD)		---
数据寄存器	DR0 ～ DR15		---
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

*B(424) 把 Md 和 Mr 中的 BCD 内容相乘，并把结果输出给 R。

标志

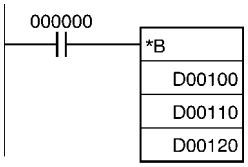
名称	标识	操作
错误标志	ER	Md 不是 BCD 码时置 ON。 Mr 不是 BCD 码时置 ON。 所有其它情况时置 OFF。
等于标志	=	结果为 0 时置 ON。 所有其它情况时置 OFF。

注意

如果 Md 和 / 或 Mr 不是 BCD 码，将产生错误，并且错误标志将置 ON。
如果由于相乘， R,R+1 的内容为 0000，则等于标志将置 ON。

例

在下例中，当 CIO 000000 为 ON 时，D00100 和 D00110 将作为 4 位 BCD 码相乘，并把结果送给 D00121 和 D00120。



3-11-22 双字 BCD 乘：*BL(425)

用途

8 个数字 （双字） BCD 数据和 / 或常数相乘。

梯形图符号

变化

变化	ON 条件时，每个周期执行	*BL(425)
	上升沿微分时执行一次	@*BL(425)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断程序
OK	OK	OK	OK

操作数定义

区域	Md	Mr	R
CIO 区	CIO 0000 ～ CIO 6142		CIO 0000 ～ CIO 6140
工作区	W000 ～ W510		W000 ～ W508
保持位区	H000 ～ H510		H000 ～ H508
辅助位区	A000 ～ A958		A448 ～ A956
定时器区	T0000 ～ T4094		T0000 ～ T4092
计数器区	C0000 ～ C4094		C0000 ～ C4092

区域	Md	Mr	R
DM 区	D00000 ～ D32766		D00000 ～ D32764
无区号 EM 区	E00000 ～ E32766		E00000 ～ E32764
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)		En_00000 ～ En_32764 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#00000000 ～ #99999999 (BCD)		---
数据寄存器	---		
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

*BL(425) 把 Md 和 Md+1 与 Mr 和 Mr+1 中的 BCD 数相乘，并把结果送给 R,R+1,R+2 和 R+3。

标志

名称	标识	操作
错误标志	ER	当 Md 和 / 或 Md+1 不是 BCD 码时置 ON。 当 Mr 和 / 或 Mr+1 不是 BCD 码时置 ON。 所有其它情况时置 OFF。
等于标志	=	结果为 0 时置 ON。 所有其它情况时置 OFF。

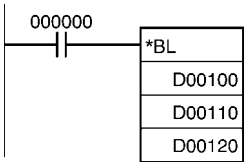
注意

如果 Md， Md+1 和 / 或 Mr， Mr+1 不是 BCD 码，将产生错误，并且错误标志将置 ON。

如果由于相乘， R,R+1,R+2,R+3 的内容为 00000000，则等于标志将置 ON。

例

在下例中，当 CIO 000000 为 ON 时，D00101 和 D00100 及 D00111 和 D00110 将以 8 位无符号 BCD 码形式相乘，并把结果送给 D00123， D00122,D00121 和 D00120。

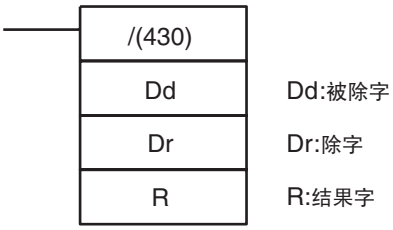


3-11-23 有符号二进制除：/(430)

用途

4 个数字 （单字）有符号十六进制数和 / 或常数相除。

梯形图符号



变化

变化	ON 条件时，每个周期执行	/(430)
	上升沿微分时执行一次	@/(430)
	下降沿微分时执行一次	不支持
下降沿微分时执行一次		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数说明

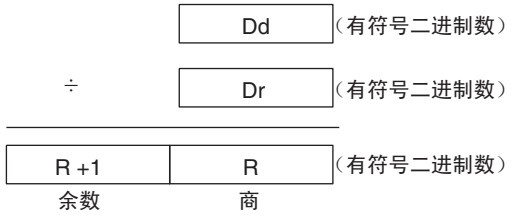
区域	Dd	Dr	R
CIO 区	CIO 0000 ～ CIO 6143		CIO 0000 ～ CIO 6142
工作区	W000 ～ W511		W000 ～ W510
保持位区	H000 ～ H511		H000 ～ H510
辅助位区	A000 ～ A959		A448 t ～ A958
定时器区	T0000 ～ T4095		T0000 ～ T4094
计数器区	C0000 ～ C4095		C0000 ～ C4094
DM 区	D00000 ～ D32767		D00000 ～ D32766
无区号 EM 区	E00000 ～ E32767		E00000 ～ E32766
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		En_00000 ～ En_32766 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		

区域	Dd	Dr	R
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #FFFF （二进制）	#0001 ～ #FFFF （二进制）	---
数据寄存器	DR0 ～ DR15		---
索引寄存器	---		
使用索引寄存器 的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

/(430) 把 Dd 中的有符号二进制数 （16 位）除以 Dr 中的数，并把结果输出到 R,R+1。

商放在 R 中，余数放在 R+1 中。



标志

名称	标识	操作
错误标志	ER	结果为 0 时置 ON。 所有其它情况时置 OFF。
等于标志	=	由于相除， R 为 0 时置 ON。 所有其它情况时置 OFF。
负标志	N	R 的最左边位为 1 时置 ON。 所有其它情况时置 OFF。

注意

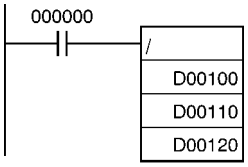
当 Dr 的内容为 0，将产生错误标志并且错误标志将置 ON。

如果由于相除， R 的内容为 0000，则等于标志将置 ON。

如果由于相除， R 的最左边位的内容为 1，则负标志位将置 ON。

例

在下例中，当 CIO 000000 为 ON 时， D00100 将作为 4 个数字有符号二进制数被 D00110 相除，商被放在 D00120, 余数放在 D00121。

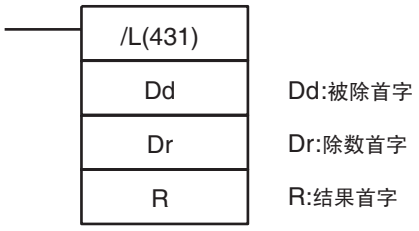


3-11-24 有符号双字二进制除：/L(431)

用途

8 个数字 （双字）有符号十六进制数和 / 或常数相除。

梯形图符号



变化

变化	ON 条件时，每个周期执行	/L(431)
	上升沿微分时执行一次	@/L(431)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

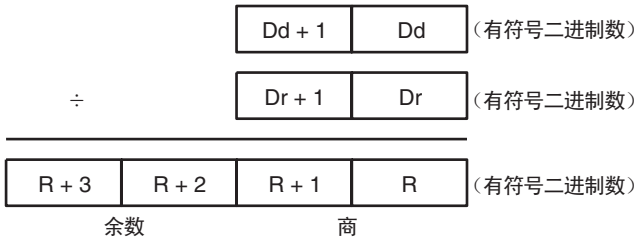
操作数定义

区域	Dd	Dr	R
CIO 区	CIO 0000 ～ CIO 6142		CIO 0000 ～ CIO 6140
工作区	W000 ～ W510		W000 ～ W508
保持位区	H000 ～ H510		H000 ～ H508
辅助位区	A000 ～ A958		A448 ～ A956
定时器区	T0000 ～ T4094		T0000 ～ T4092
计数器区	C0000 ～ C4094		C0000 ～ C4092
DM 区	D00000 ～ D32766		D00000 ～ D32764
无区号 EM 区	E00000 ～ E32766		E00000 ～ E32764
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)		En_00000 ～ En_32764 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#00000000 ～ #FFFFFFFF （二进制）	#00000001 ～ #FFFFFFFF （二进制）	---
数据寄存器	---		

区域	Dd	Dr	R
索引寄存器	---		
使用索引寄存器 的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

注意

/L(431) 把 Dd 和 Dd+1 中的有符号二进制数除以 Dr 和 Dr+1 中的数，并把结果输出到 R,R+1， R+2 和 R+3，商输出给 R 和 R+1，余数输出给 R+2 和 R+3。



标志

名称	标识	操作
错误标志	ER	结果为 0 时置 ON。 所有其它情况时置 OFF。
等于标志	=	由于相除， R+1,R 为 0 时置 ON。 所有其它情况时置 OFF。
负标志	N	R+1,R 的最左边位为 1 时置 ON。 所有其它情况时置 OFF。

注意

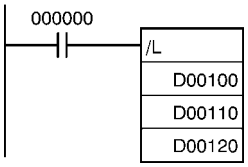
结果的余数 R+3,R+2 为 0 时，错误标志将置 ON。

如果由于相除， R+1,R 的内容为 00000000，则等于标志将置 ON。

如果由于相除， R+1,R 的最左边位的内容为 1，则负标志位将置 ON。

例

在下例中，当 CIO 000000 为 ON 时， D00101 和 D00100 将作为 8 个数字有符号十六进制数除以 D00111 和 D00110，商被输出在 D00121 和 D00120, 余数被输出到 D00123 和 D00122。

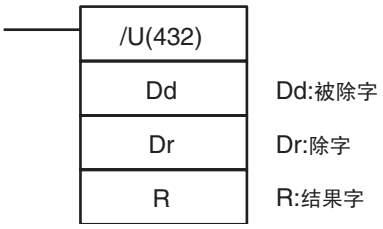


3-11-25 无符号二进制除 :/U(432)

用途

4 个数字 （单字）无符号十六进制数和 / 或常数相除。

梯形图符号



变化

变化	ON 条件时，每个周期执行	/U(432)
	上升沿微分时执行一次	@/U(432)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断程序
OK	OK	OK	OK

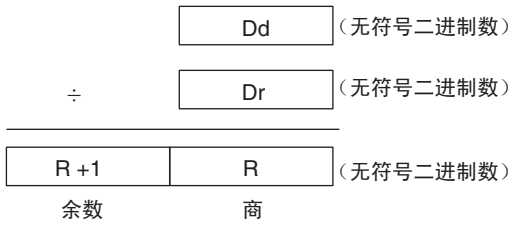
操作数定义

区域	Dd	Dr	R
CIO 区	CIO 0000 ～ CIO 6143		CIO 0000 ～ CIO 6142
工作区	W000 ～ W511		W000 ～ W510
保持位区	H000 ～ H511		H000 ～ H510
辅助位区	A000 ～ A959		A448 ～ A958
定时器区	T0000 ～ T4095		T0000 ～ T4094
计数器区	C0000 ～ C4095		C0000 ～ C4094
DM 区	D00000 ～ D32767		D00000 ～ D32766
无区号 EM 区	E00000 ～ E32767		E00000 ～ E32766
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		En_00000 ～ En_32766 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #FFFF （二进制）	#0001 ～ #FFFF （二进制）	---
数据寄存器	DR0 ～ 15		---

区域	Dd	Dr	R
索引寄存器	---		
使用索引寄存器 的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

/U(432) 把 Dd 中无符号二进制数除以 Dr 中的数，并把商送到 R, 余数送到 R+1。



标志

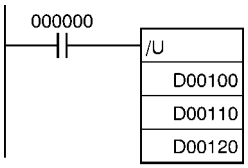
名称	标识	操作
错误标志	ER	结果为 0 时置 ON。 所有其它情况时置 OFF。
等于标志	=	由于相除， R 为 0 时置 ON。 所有其它情况时置 OFF。
负标志	N	R 的最左边位为 1 时置 ON。 所有其它情况时置 OFF。

注意

如果由于相除， R+1 的内容为 0，则错误标志将置 ON。
如果由于相处， R 的内容为 0000，则等于标志将置 ON。
如果由于相处， R 的最左边内容为 1，则负标志将置 ON。

例

在此例当中，当 CIO 000000 为 ON 时，D00100 将作为 4 个数字无符号二进制值除以 D00110，商被送到 D00120，余数被送到 D00121。



3-11-26 无符号双字二进制除：/UL(433)

用途

8 个数字 （双字）无符号十六进制和 / 或常数相除。

梯形图符号

变化

变化	ON 条件时，每个周期执行	/UL(433)
	上升沿微分时执行一次	@/UL(433)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断程序
OK	OK	OK	OK

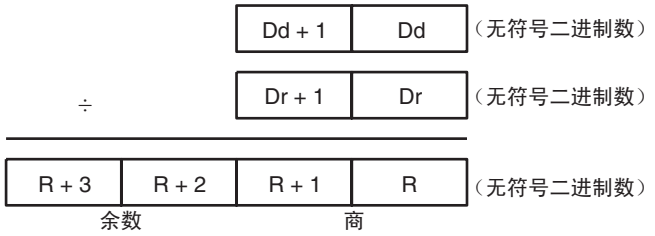
操作数定义

区域	Dd	Dr	R
CIO 区	CIO 0000 ～ CIO 6142		CIO 0000 ～ CIO 6140
工作区	W000 ～ W510		W000 ～ W508
保持位区	H000 ～ H510		H000 ～ H508
辅助位区	A000 ～ A958		A448 ～ A956
定时器区	T0000 ～ T4094		T0000 ～ T4092
计数器区	C0000 ～ C4094		C0000 ～ C4092
DM 区	D00000 ～ D32766		D00000 ～ D32764
无区号 EM 区	E00000 ～ E32766		E00000 ～ E32764
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)		En_00000 ～ En_32764 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#00000000 ～ #FFFFFFFF （二进制）	#00000001 ～ #FFFFFFFF （二进制）	---
数据寄存器	---		

区域	Dd	Dr	R
索引寄存器	---		
使用索引寄存器 的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

/UL(433) 把 Dd 和 Dd+1 中的无符号二进制数除以 Dr 和 Dr+1 中的数，并把商送到 R,R+1, 余数送到 R+2 和 R+3。



标志

名称	标识	操作
错误标志	ER	结果为 0 时置 ON。 所有其它情况时置 OFF。
等于标志	=	由于相除， R+1,R 为 0 时置 ON。 所有其它情况时置 OFF。
负标志	N	R+1,R 的最左边位为 1 时置 ON。 所有其它情况时置 OFF。

注意

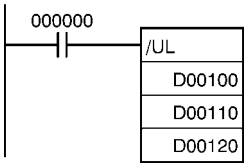
Dr,Dr+1 的内容为 0 时错误标志将置 ON。

如果由于相除， R,R+1 的内容都为 0000，则等于标志将置 ON。

如果由于相除， R+1 的最左边位的内容为 1，则负标志将置 ON 。

例

在下例中，当 CIO 000000 为 ON 时，D00100 和 D00101 将作为 8 个数字无符号十六进制除以 D00111 和 D00110, 并把商送给 D00121 和 D00120，余数送给 D00123 和 D00122。

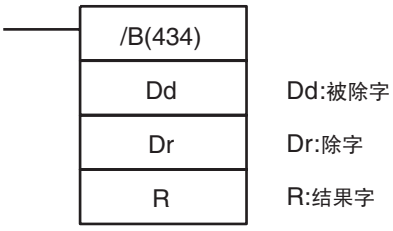


3-11-27 BCD 除：/B(434)

用途

4 个数字 （单字） BCD 码和 / 或常数相除。

梯形图符号



变化

变化	ON 条件时，每个周期执行	/B(434)
	上升沿微分时执行一次	@/B(434)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

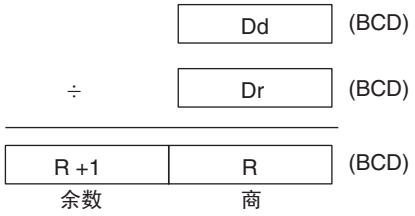
操作数定义

区域	Dd	Dr	R
CIO 区	CIO 0000 ～ CIO 6143		CIO 0000 ～ CIO 6142
工作区	W000 ～ W511		W000 ～ W510
保持位区	H000 ～ H511		H000 ～ H510
辅助位区	A000 ～ A959		A448 ～ A958
定时器区	T0000 ～ T4095		T0000 ～ T4094
计数器区	C0000 ～ C4095		C0000 ～ C4094
DM 区	D00000 ～ D32767		D00000 ～ D32766
无区号 EM 区	E00000 ～ E32767		E00000 ～ E32766
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		En_00000 ～ En_32766 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #9999 (BCD)	#0001 ～ #9999 (BCD)	---
数据寄存器	DR0 ～ DR15		---

区域	Dd	Dr	R
索引寄存器	---		
使用索引寄存器 的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

/B(434) 将 Dd 的 BCD 内容除以 Dr 的内容，并输出商到 R，余数到 R+1。



标志

名称	标识	操作
错误标志	ER	Dd 不是 BCD 码时置 ON。 Dr 不是 BCD 码时置 ON。 余数为 0 时置 ON。 所有其它情况时置 OFF。
等于标志	=	R 为 0 时置 ON。 所有其它情况时置 OFF。

注意

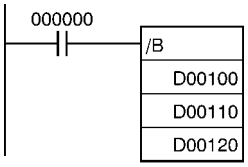
如果 Dd 或 Dr 不是 BCD 码，或者如果余数 （R+1）为 0，将产生错误，并且错误标志将置 ON。

如果由于相除， R 的内容为 0000，则等于标志将置 ON。

如果由于相除， R 的最左边位的内容为 1，则负标志位将置 ON。

例

在下例中，当 CIO 000000 为 ON 时， D00100 将作为 4 个数字 BCD 码除以 D00110，商被输出在 D00120, 余数被输出到 D00121。

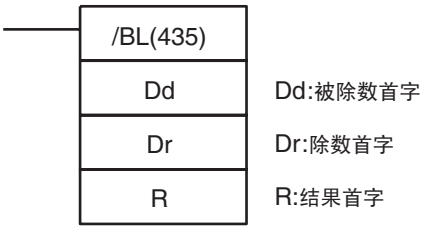


3-11-28 双字 BCD 除：/BL(435)

用途

8 个数字 （双字） BCD 码和 / 或常数相除。

梯形图符号



变化

变化	ON 条件时，每个周期执行	/BL(435)
	上升沿微分时执行一次	@/BL(435)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

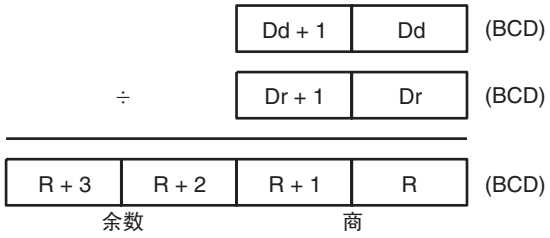
操作数定义

区域	Dd	Dr	R
CIO 区	CIO 0000 ～ CIO 6142		CIO 0000 ～ CIO 6140
工作区	W000 ～ W510		W000 ～ W508
保持位区	H000 ～ H510		H000 ～ H508
辅助位区	A000 ～ A958		A448 ～ A956
定时器区	T0000 ～ T4094		T0000 ～ T4092
计数器区	C0000 ～ C4094		C0000 ～ C4092
DM 区	D00000 ～ D32766		D00000 ～ D32764
无区号 EM 区	E00000 ～ E32766		E00000 ～ E32764
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)		En_00000 ～ En_32764 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#00000000 ～ #99999999 (BCD)	#00000001 ～ #99999999 (BCD)	---
数据寄存器	---		

区域	Dd	Dr	R
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

/BL(435) 把 Dd 和 Dd+1 中的 BCD 码除以 Dr 和 Dr+1 中的数，并把商输出给 R 和 R+1，余数输出给 R+2 和 R+3。



标志

名称	标识	操作
错误标志	ER	Dd,Dd+1 不是 BCD 码时置 ON。 Dr,Dr+1 不是 BCD 码时置 ON。 所有其它情况时置 OFF。
等于标志	=	结果为 0 时置 ON。 所有其它情况时置 OFF。

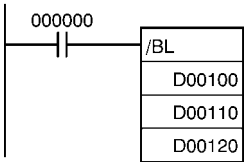
注意

如果 Dd，Dd+1 和 / 或 Dr，Dr+1 不是 BCD 码，或者 Dr,Dr+1 为 0，将产生错误，并且错误标志将置 ON。

如果由于相除， R,R+1 的内容为 00000000，则等于标志将置 ON。

例

在下例中，当 CIO 000000 为 ON 时，D00101 和 D00100 将作为 8 个数字 BCD 码除以 D00111 和 D00110，商被输出到 D00121 和 D00120, 余数被输出到 D00123 和 D00122。



3-12 转换指令

本节描述了用于数据转换的指令

指令	助记符	功能号	页数
BCD 码到二进制	BIN	023	429
双字 BCD 码到双字二进制	BINL	058	430
二进制到 BCD 码	BCD	024	432
双字二进制到双字 BCD 码	BCDL	059	433
2 的补码	NEG	160	435
双字 2 的补码	NEGL	161	437

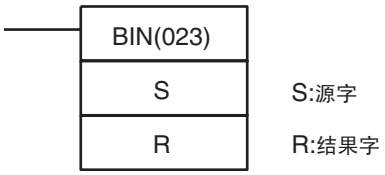
指令	助记符	功能号	页数
16 位到 32 位有符号二进制数	SIGN	600	439
数据解码	MLPX	076	440
数据编码	DMPX	077	445
ASCII 码转换	ASC	086	449
ASCII 码到十六进制数	HEX	162	453
列到行	LINE	063	457
行到列	COLM	064	459
有符号 BCD 码到二进制数	BINS	470	462
有符号双字 BCD 码到二进制数	BISL	472	465
有符号二进制数到 BCD 码	BCDS	471	468
有符号双字二进制数到 BCD 码	BDSL	473	470

3-12-1 BCD 到二进制数：BIN(023)

用途

把 BCD 码转换成二进制数。

梯形图符号



变化

变化	ON 条件时，每个周期执行	BIN(023)
	上升沿微分时执行一次	@BIN(023)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断程序
OK	OK	OK	OK

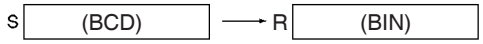
操作数定义

区域	S	R
CIO 区	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W511	
保持位区	H000 ～ H511	
辅助位区	A000 ～ A959	A448 ～ A959
定时器区	T0000 ～ T4095	
计数器区	C0000 ～ C4095	
DM 区	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	

区域	S	R
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	
数据寄存器	DR0 ～ DR15	
索引寄存器	---	
使用索引寄存器 的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

说明

BIN(023) 把 S 中的 BCD 码转换成二进制数，并把结果字写进 R。

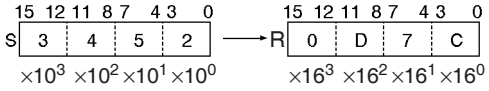


标志

名称	标识	操作
错误标志	ER	S 的内容不是 BCD 码时置 ON。 所有其它情况时置 OFF。
等于标志	=	结果为 0000 时置 ON。 所有其它情况时置 OFF
负标志	N	OFF

例

下图显示了 BCD 码到二进制数转换的例子。

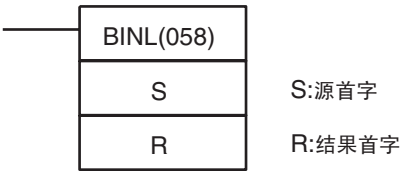


3-12-2 双字 BCD 码到双字二进制：BINL(058)

用途

8 个数字的 BCD 数到 8 个数字的十六进制 （32 位二进制）数的转换。

梯形图符号



变化

变化	ON 条件时，每个周期执行	BINL(058)
	上升沿微分时执行一次	@BINL(058)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断程序
OK	OK	OK	OK

操作数定义

区域	S	R
CIO 区	CIO 0000 t ～ CIO 6142	
工作区	W000 ～ W510	
保持位区	H000 ～ H510	
辅助位区	A000 ～ A958	A448 ～ A958
定时器区	T0000 ～ T4094	
计数器区	C0000 ～ C4094	
DM 区	D00000 ～ D32766	
无区号 EM 区	E00000 ～ E32766	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
C 常数	---	
数据寄存器	---	
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

说明

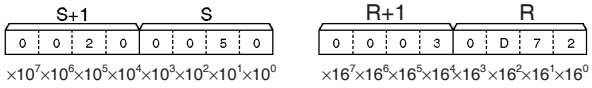
BINL(058) 把 S 和 S+1 中的 8 个数字的 BCD 数转换成 8 个数字十六进制（32 位二进制）数，并把结果写到 R 和 R+1。

标志

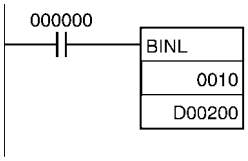
名称	标识	操作
错误标志	ER	如果 S+1,S 内容不是 BCD 码时置 ON。 所有其它情况时置 OFF。
等于标志	=	结果为 0 时置 ON。 所有其它情况时置 OFF。
负标志	N	OFF

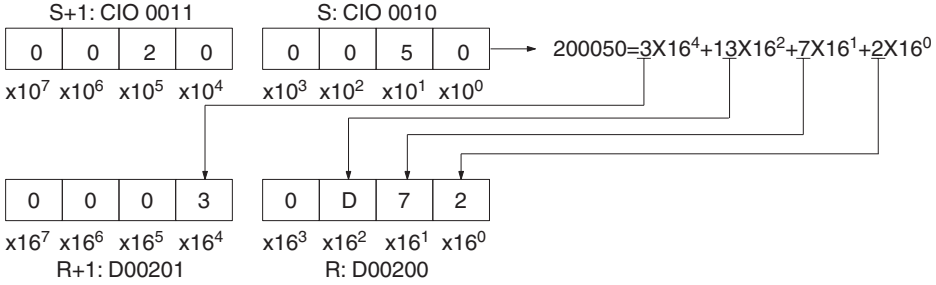
例

下图显示了 8 个数字的 BCD 码转换到二进制数的例子。



在下例中当 CIO 000000 为 ON 时， CIO 0010 和 CIO 0011 中的 8 个数字的 BCD 数转换成十六进制数，并保存在 D00200 和 D00201 中。



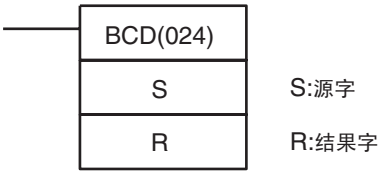


3-12-3 二进制数到 BCD 码：BCD(024)

用途

把二进制字转换成 BCD 字。

梯形图符号



变化

变化	ON 条件时，每个周期执行	BCD(024)
	上升沿微分时执行一次	@BCD(024)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断程序
OK	OK	OK	OK

操作数

S: 源字

S 必须在十六进制 0000 ～ 270F （十进制 0000 ～ 9999）之间。

操作数定义

区域	S	R
CIO 区	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W511	
保持位区	H000 ～ H511	
辅助位区	A000 ～ A959	A448 ～ A959
定时器区	T0000 ～ T4095	
计数器区	C0000 ～ C4095	
DM 区	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	

区域	S	R
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	
数据寄存器	DR0 ～ DR15	
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

说明

BCD(024) 把 S 中的二进制数转换成 BCD 数，并把结果写进 R。

标志

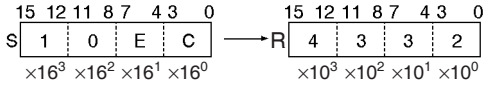
名称	标识	操作
错误标志	ER	如果 S 的内容超过 270F （9999 十进制）则置 ON。 其它情况时置 OFF。
等于标志	=	结果为 0 时置 ON。 所有其它情况时置 OFF。

注意

S 的内容不能超过 270F （9999 十进制）。

例

下图显示了 BCD 码到二进制数转换的例子。

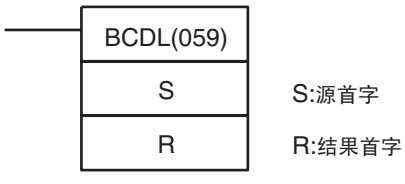


3-12-4 双字二进制数到双字 BCD 码：BCDL(059)

用途

8 个数字的十六进制 （32 位二进制）数到 8 个数字的 BCD 码的转换。

梯形图符号



变化

变化	ON 条件时，每个周期执行	BCDL(059)
	上升沿微分时执行一次	@BCDL(059)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

S: 源首字

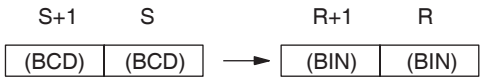
S+1和S的内容必须在十六进制数00000000～05F5E0FF（十进制数00000000～99999999）之间。

操作数定义

区域	S	R
CIO 区	CIO 0000 ～ CIO 6142	
工作区	W000 ～ W510	
保持位区	H000 ～ H510	
辅助位区	A000 ～ A958	A448 ～ A958
定时器区	T0000 ～ T4094	
计数器区	C0000 ～ C4094	
DM 区	D00000 ～ D32766	
无区号 EM 区	E00000 ～ E32766	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	
数据寄存器	---	
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

说明

BCDL(059) 把 S 和 S+1 中的 8 个数字的十六进制 （32 进制）数转换为 8 个数字的 BCD 码，并把结果写到 R 和 R+1。



标志

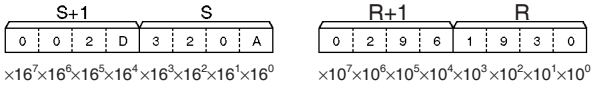
名称	标识	操作
错误标志	ER	如果 S 和 S+1 的内容超过 05F5E0FF （十进制 99999999），则置 ON。 其它情况时置 OFF。
等于标志	=	结果为 0 时置 ON。 所有其它情况时置 OFF。

注意

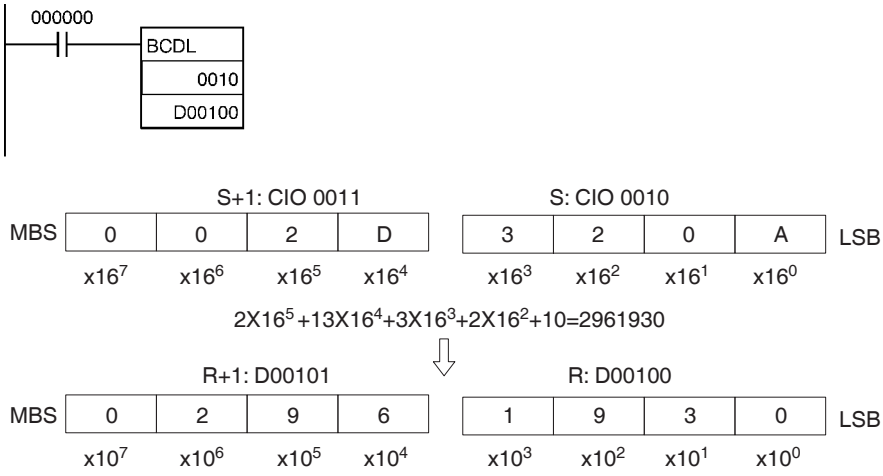
S 和 S+1 的内容不能超过 05F5E0FF （十进制 99999999）。

例

下图显示了一个 8 个数字的 BCD 码到二进制数的转换的例子。



在下例中当 CIO 000000 为 ON 时， CIO 0011 和 CIO 0010 中的十六进制数转换成 BCD 码，并存储在 D00100 和 D00101 中。

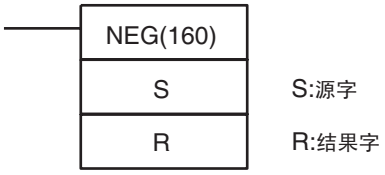


3-12-5 2 的补码：NEG(160)

用途

计算十六进制数的一个字的 2 的补码。

梯形图符号



变化

变化	ON 条件时，每个周期执行	NEG(160)
	上升沿微分时执行一次	@NEG(160)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

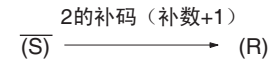
操作数定义

区域	S	R
CIO 区	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W511	
保持位区	H000 ～ H511	
辅助位区	A000 ～ A959	A448 ～ A959
定时器区	T0000 ～ T4095	
计数器区	C0000 ～ C4095	
DM 区	D00000 ～ D32767	

区域	S	R
无区号 EM 区	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#0000 ～ #FFFF （二进制）	---
数据寄存器	DR0 ～ DR15	
索引寄存器	---	
适用索引寄存器 的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

说明

NEG(160)计算S的2的补码，并把结果写到R，2的补码的计算为S的位的状态反转后加 1。



注 本操作 （位状态取反并加 1）等于从 0000 减去 S 的内容。

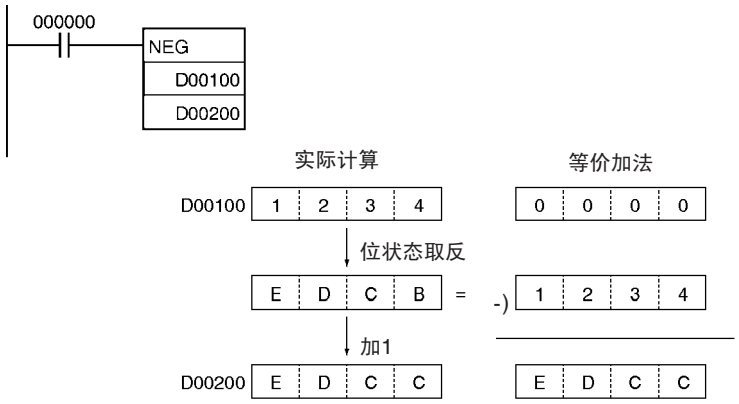
标志

名称	标识	操作
错误标志	ER	OFF
等于标志	=	结果为 0 时置 ON。 其它情况时置 OFF。
负标志	N	如果结果的第十五位为 ON 时置 ON。 其它情况时置 OFF。

注 十六进制 8000 的结果为十六进制 8000。

例

在下例中当 CIO 000000 为 ON 时， NEG(160) 计算 D 00100 的 2 的补码，并把结果写到 D00200 中。

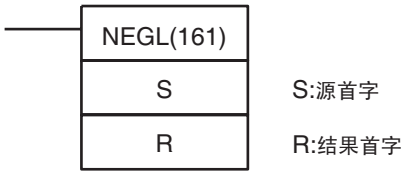


3-12-6 双字 2 的补码：NEGL(161)

用途

计算 2 个十六进制字的 2 的补码。

梯形图符号



变化

变化	ON 条件时，每个周期执行	NEGL(161)
	上升沿微分时执行一次	@NEGL(161)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数说明

区域	S	R
CIO 区	CIO 0000 ～ CIO 6142	
工作区	W000 ～ W510	
保持位区	H000 ～ H510	
辅助位区	A000 ～ A958	A448 ～ A958
定时器区	T0000 ～ T4094	
计数器区	C0000 ～ C4094	
DM 区	D00000 ～ D32766	
无区号 EM 区	E00000 ～ E32766	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	

区域	S	R
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#00000000 ～ #FFFFFFFF （二进制）	---
数据寄存器	---	
索引寄存器	---	
使用索引寄存器 的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

注 R 和 R+1 必须在相同的数据区中

说明 NEGL(161) 计算 S+1 和 S 中的 2 的补码，并把结果写到 R+1 和 R 中。2 的补码的计算时把 S+1 和 S 中的位状态取反并加 1。

2的补码（补数+1）

$\overline{(S+1, S)} \longrightarrow (R+1, R)$

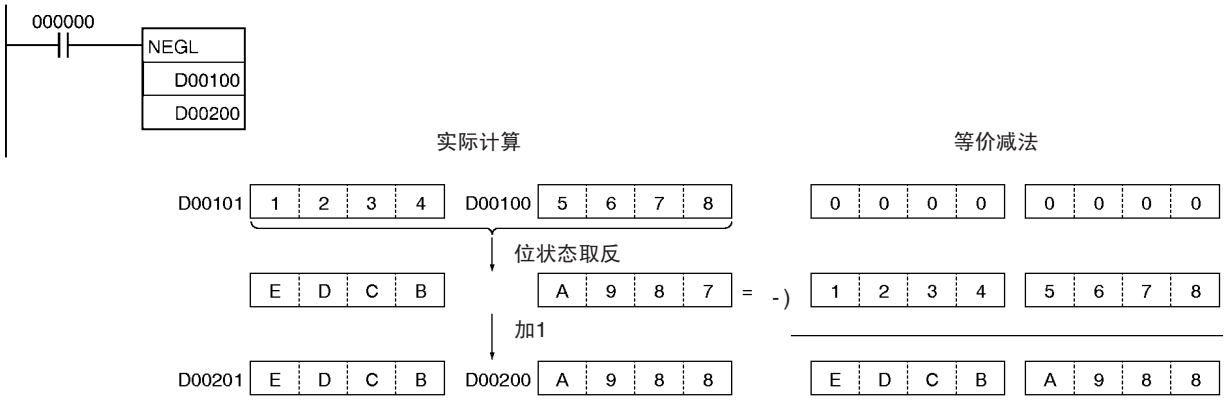
注 本操作 （位状态取反并加 1）等价于从 00000000 减去 S+1 和 S 中的内容。

标志

名称	标识	操作
错误标志	ER	OFF
等于标志	=	结果为 00000000 时置 ON。 所有其它情况时置 OFF。
负标志	N	R+1 的第 15 位为 ON 则置 ON。 其余情况下置 OFF。

注 十六进制 8000 的结果为十六进制 8000。

例 在下例中当 CIO 000000 为 ON 时，NEGL(161) 计算 D00101 和 D00100 中的内容的 2 的补码，并把结果写到 D00201 和 D00200.

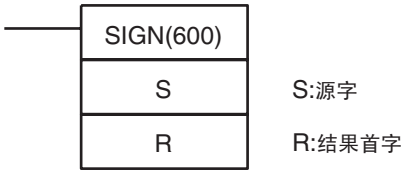


3-12-7 16 位到 32 位有符号二进制数：SIGN(600)

用途

把 16 位有符号二进制数扩展为 32 位等数值数。

梯形图符号



变化

变化	ON 条件时，每个周期执行	SIGN(600)
	上升沿微分时执行一次	@SIGN(600)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断程序
OK	OK	OK	OK

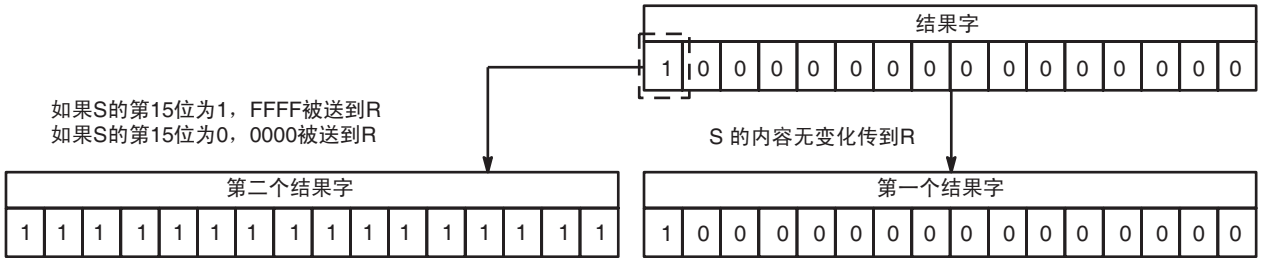
操作数定义

区域	S	R
CIO 区	CIO 0000 ～ CIO 6143	CIO 0000 ～ CIO 6142
工作区	W000 ～ W511	W000 ～ W510
保持位区	H000 ～ H511	H000 ～ H510
A 辅助位区	A000 ～ A959	A448 ～ A958
定时器区	T0000 ～ T4095	T0000 ～ T4094
计数器区	C0000 ～ C4095	C0000 ～ C4094
DM 呵	D00000 ～ D32767	D00000 ～ D32766
无区号 EM 区	E00000 ～ E32767	E00000 ～ E32766
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	En_00000 ～ En_32766 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#0000 ～ #FFFF （二进制）	---
数据寄存器	DR0 ～ DR15	---
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

注 R 和 R+1 必须在同一数据区。

说明

SIGN(600) 把 S 中的 16 位有符号二进制数转换成等价的 32 位有符号二进制数，并把结果写进 R+1 和 R。转换是把 S 的内容复制到 R，如果 S 的第 15 位为 1，则把 FFFF 写到 R+1, 或者如果 S 的第 15 位为 0，则把 0000 写到 R+1。

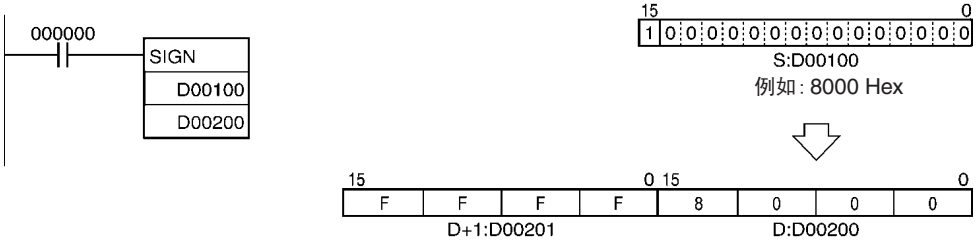


标志

名称	标识	操作
错误标志	ER	OFF
等于标志	=	结果为 00000000 时置 ON。 所有其它情况时置 OFF。
负标志	N	如果 R+1 的第 15 位为 ON 则置 ON。 所有其它情况时置 OFF。

例

下例中，当 CIO 000000 为 ON，SIGN(600) 把 D00100 的 16 位有符号二进制数（#8000=-32768 十进制）内容转换成等价的 32 位数（#FFFF80000=-32768 十进制），并把结果写到 D00201 和 D00200。

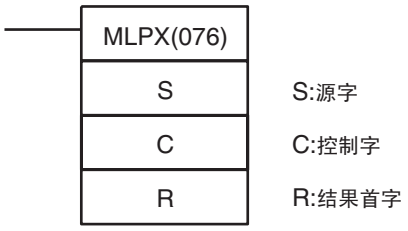


3-12-8 数据译码：MLPX(076)

用途

读源字中指定数字 （或字节）的数值，把结果字 （或 16 字范围）中的相应位置 ON，结果字 （或 16 字范围）中的其余位置 OFF。

梯形图符号



变化

变化	ON 条件时，每个周期执行	MLPX(076)
	上升沿微分时执行一次	@MLPX(076)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断程序
OK	OK	OK	OK

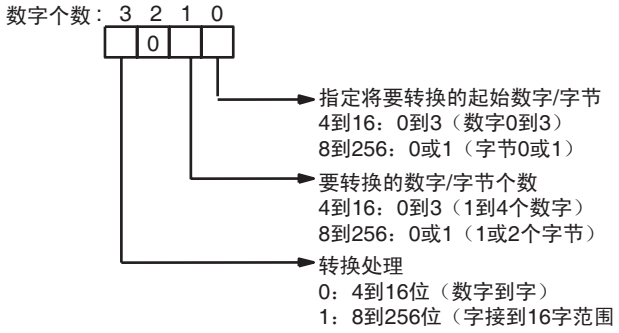
操作数

S: 源字

源字中的数据表明将置 ON 的位的位置。

C：控制字

控制字指定 MLPX(076) 是执行 4 ～ 16 位转换还是 8 ～ 256 位转换，转换的数字或字节的个数，以及开始数字或字节。



R: 结果首字

根绝转换处理的类型和被转换的数字 / 字节的个数，可以是 1 ～ 32 字的一种。结果字必须在同一数据区。

操作数定义

区域	S	C	R
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---	仅指定值	---
数据寄存器	DR0 ～ DR15		---

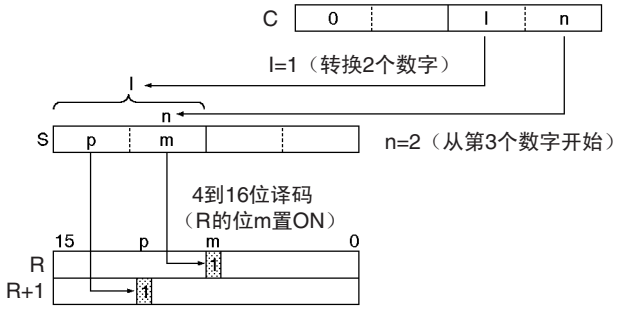
区域	S	C	R
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

MLPX(076) 可执行 4 ～ 16 位或 8 ～ 256 位转换。把 C 的最左边位数设为 0，以指定 4 ～ 16 位转换。设为 1，以指定 8 ～ 256 位转换。

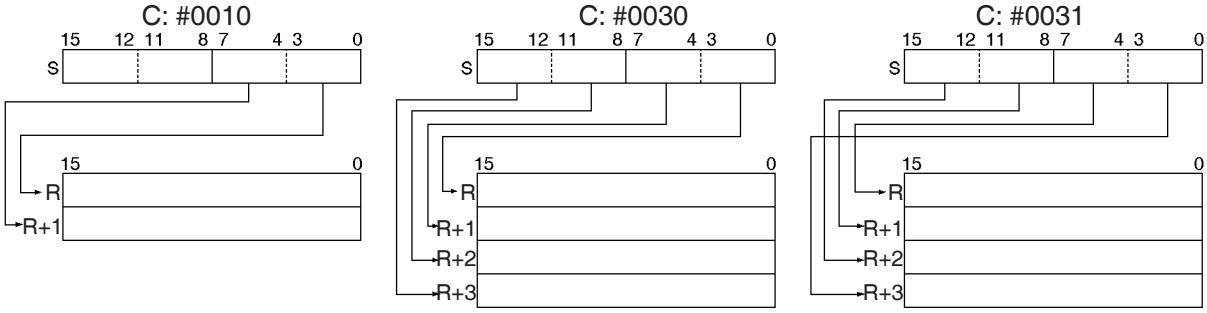
4 ～ 16 位转换

当 C 的最左边的数字为 0，则 MLPX(076) 取 S 中指定的数字值 （0 ～ F），并把结果字中相应位置 ON。结果字中的其余位将被置 OFF。最多 4 个数字可被转换。



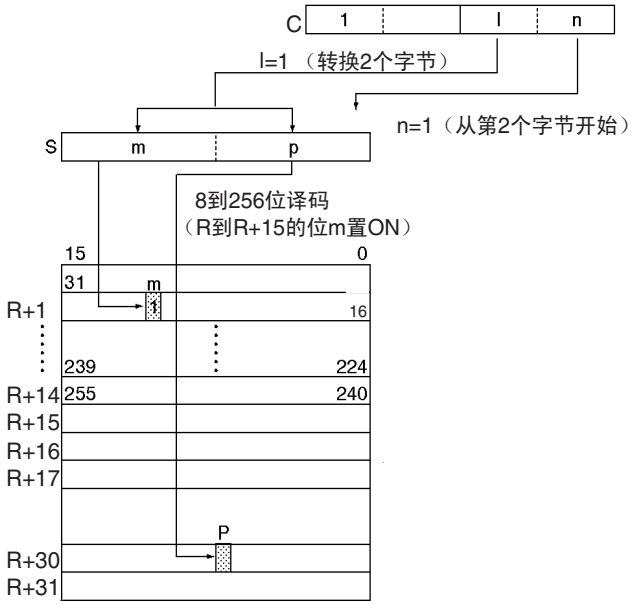
当两个或两个以上数字被转换，MLPX(076) 将从右至左读 S 中的数字，如果需要，将从最左边数字回读到最右边数字。

下图显示了 C 的一些举例数值和它们产生的 4 ～ 16 位转换。



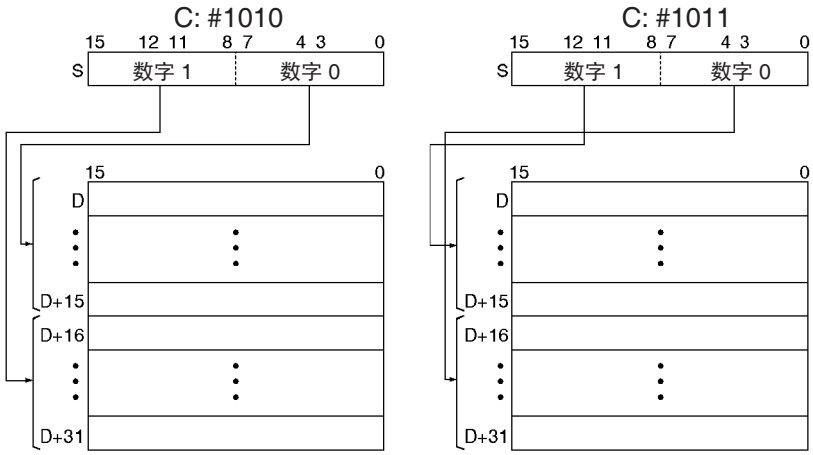
8 ～ 256 位转换

当 C 的最左边为 1， MLPX(076) 取 S 中指定字节的值 （00 ～ FF），并把 16 个结果字范围内的相应位置 ON，结果字中的其余位将被置 OFF。最多两个字节可被转换。



当两个字节转换时， MLPX(076) 从右至左读 S 中的字节，如果最左边位字节（字节 1）被指定为起始字节，则它将回读到最右边的字节。

下图给出了 C 的一些举例值以及它们产生的 8 ～ 256 位转换。



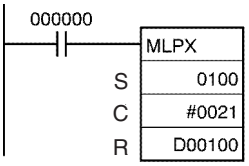
标志

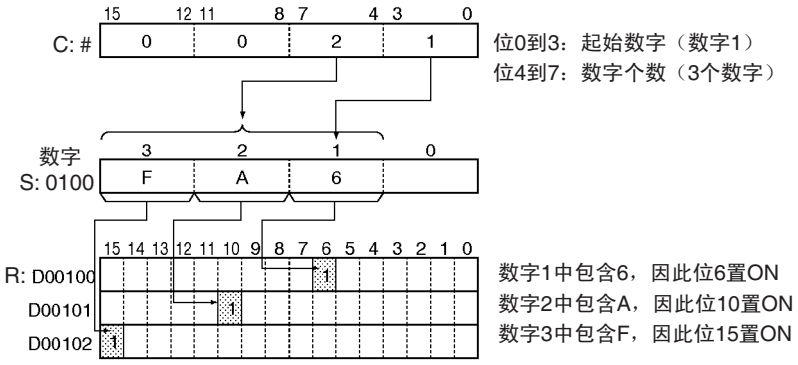
名称	标识	操作
错误标志	ER	如果 C 不在指定范围内则置 ON。 其余情况下置 OFF。

例

4 ～ 16 位转换

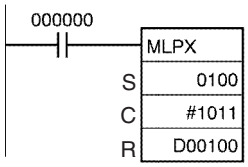
在下例中档 CIO 000000 为 ON 时，MLPX(076) 将转换 S 中从数字 1（第 2 个数字）开始的 3 个数字，如 C(#0021) 所指示 。 D00100， D00101 和 D00102 中的相应位将被置 ON。

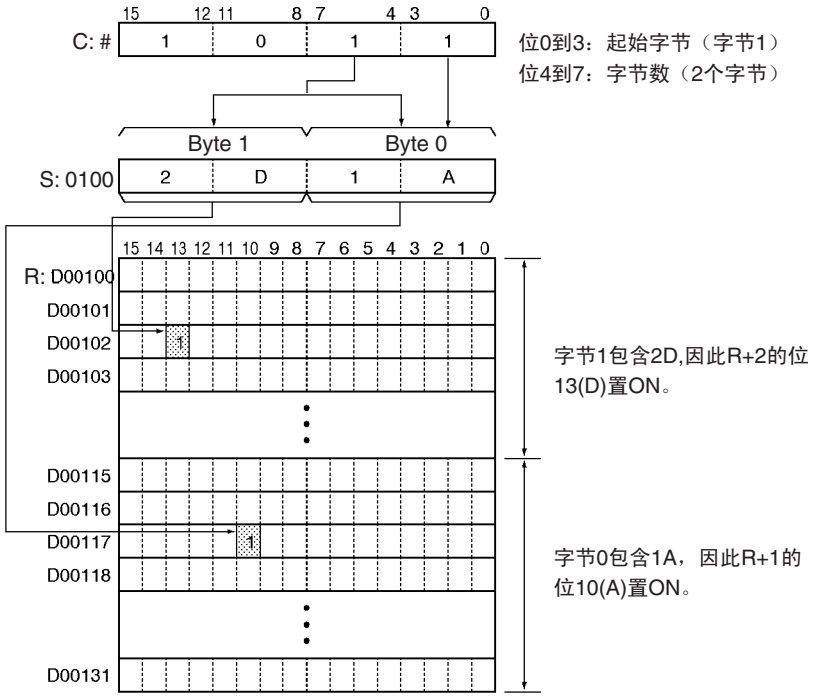




8 ～ 256 位转换

在下例中当 CIO 000000 为 ON 时，MLPX(076) 将转换 S 中从字节 1（最左边的字节）开始的 2 个字节，如 C(#1011) 所指示。 D00100 ～ D00115 以及 D00116 ～ D00131 中的相应位将被置 ON。



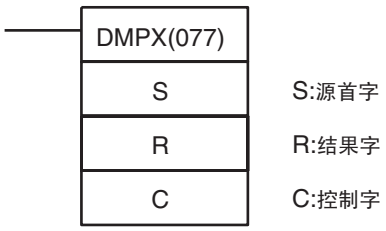


3-12-9 数据编码：DMPX(077)

用途

在源字 （或 16 字范围）内寻找第一个或最后一个 ON 的位置，并把该值写入结果字中的指定数字 （或字节）。

梯形图符号



变化

变化	ON 条件时，每个周期执行	DMPX(077)
	上升沿微分时执行一次	@DMPX(077)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

S：源首字

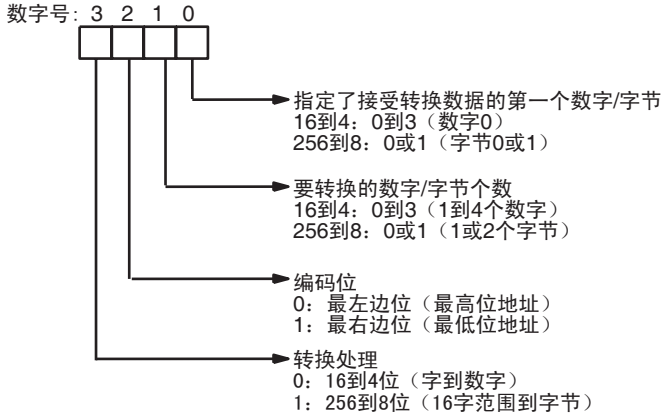
根据转换处理的类型和被转换的数字 / 字节个数可以是 1 ～ 32 个源字的一种。源字必须在同一数据区。

R：结果字

源字中为 ON 的位的位置被写入 R 中从指定的第一个数字 / 字节开始的数字 / 字节中。

C：控制字

控制字指定 DMPX(077) 将进行 16 ～ 4 位的转换还是 256 ～ 8 位的转换，最左边的位还是最右边的 ON 位被编码，要转换的数字或字节个数，以及将写入结果的起始数字或字节。



操作数定义

区域	S	R	C
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		

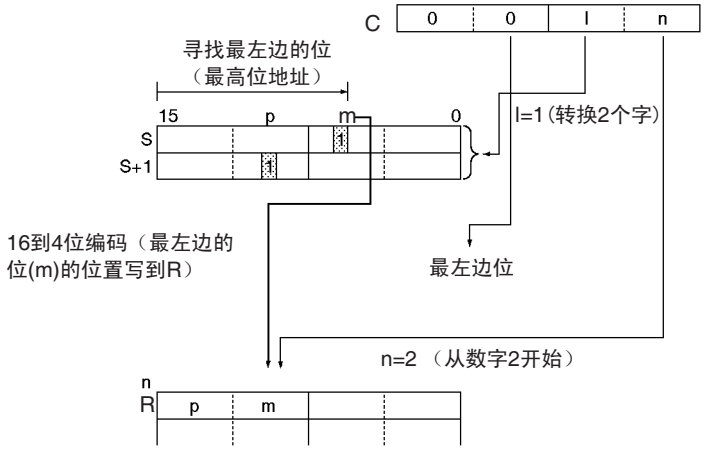
区域	S	R	C
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959	A448 ～ A959	A000 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---	---	仅为指定值
数据寄存器	---	DR0 ～ DR15	
索引寄存器	---		
适用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

DMPX(077) 可进行 16 ～ 4 位或 256 ～ 8 位转换。把 C 的最左边的数字设为 0 指定 16 ～ 4 位转换。设为 1，指定 256 ～ 8 位转换。

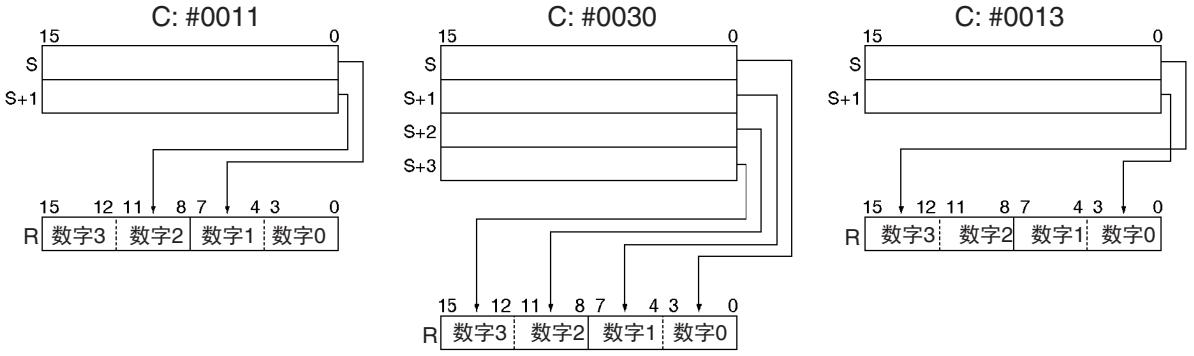
16 ～ 4 位转换

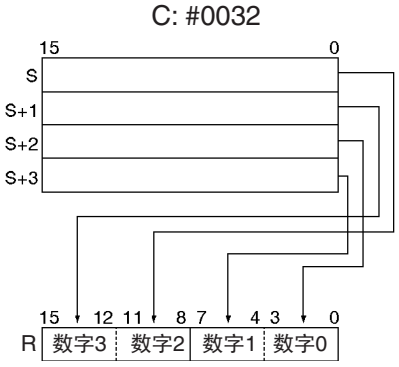
当C的第4个（最左边）数字为0时，DMPX(077) 在最多4个源字里寻找最左边或最右边的为 ON 位的位置，并把这些位置写进 R，从指定数字开始。（把 C 的第三个数字设为0，寻找最左边的为ON位。设为1，寻找最右边的为ON位）。



当转换2个或2个以上数字时，DMPX(077) 将把值从右至左写进R，如果需要，将在最左边的数字之后绕回到最右边的数字。

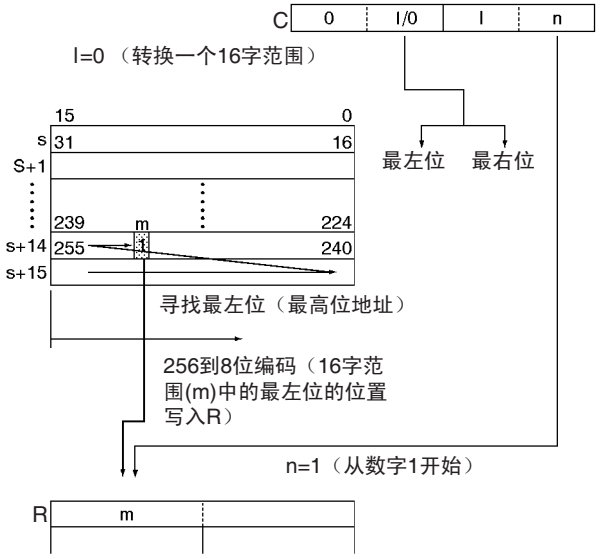
下图给出了 C 的举例值，以及他们产生的 16 ～ 4 位的转换。



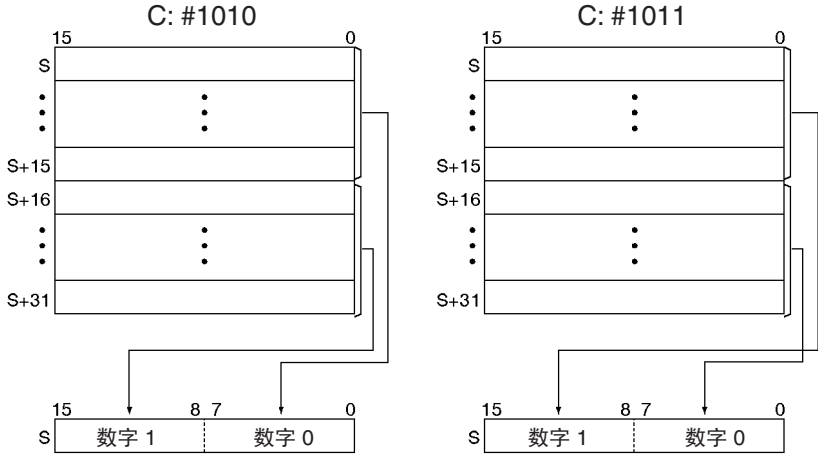


256 ～ 8 位转换

当C的第4个（最左边）数字为1时，DMPX(077)在1个或2个16字范围的源字内寻找最左边（最高位地址）或最右边（最低位地址）的置 ON 位。从指定字节开始，把 这些位的位置写进 R （把 C 的第三个数字设为 0，寻找最左边的为 ON 位。或设为 1，寻找最右边的置 ON 位）。



当转换 2 个字节时， DMPX(077) 将把数值从右至左写进 R 的字节中，如果最左字节 （字节 1）被指定为起始字节，则将绕回到最右字节。
下图显示了 C 的举例值，以及他们产生的 256 ～ 8 位转换。



标志

名称	标记	操作
错误标志	ER	如果源字中的内容为 0000Hex （即没有位编译）则置 ON。 如果 C 不在指定的范围内则置 ON。 其余情况下置 OFF。

注意

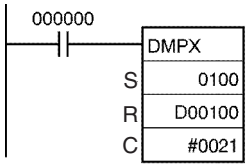
如果转换的数据包含 0000hex，而其他数据需要被编译，则通过多次使用 DMPX(077) 指令来分别转换数据。

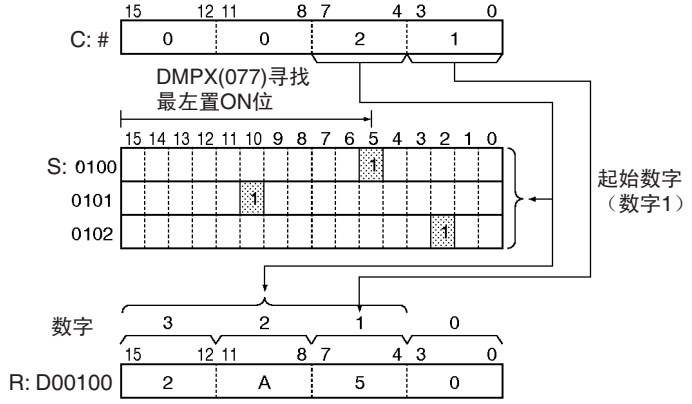
```
DMPX(077)   D0000  D0100  #0300
```

```
DMPX(077)   D0000  D0100  #0000
DMPX(077)   D0001  D0100  #0001
DMPX(077)   D0002  D0100  #0002
DMPX(077)   D0003  D0100  #0003
```

例

在下例中，当 CIO 000000 为 ON 时，DMPX(077) 将在 CIO 0100,CIO 0101 和 CIO 0102 中寻找最左的置 ON 位，并把这些位置写进从数字 1 （第二个数字）开始的 R 中的 3 个数字，如 C(#0021) 所指示。



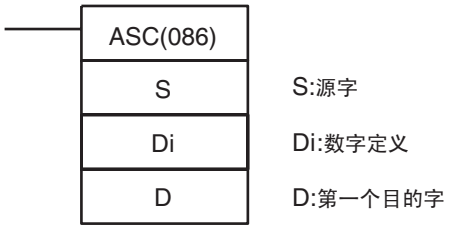


3-12-10 ASCII 码转换：ASC(086)

用途

把源字中的 4 位十六进制数字转换为对应的 8 位 ASCII 码。

梯形图符号



变化

变化	ON 条件时，每个周期执行	ASC(086)
	上升沿微分时执行一次	@ASC(086)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

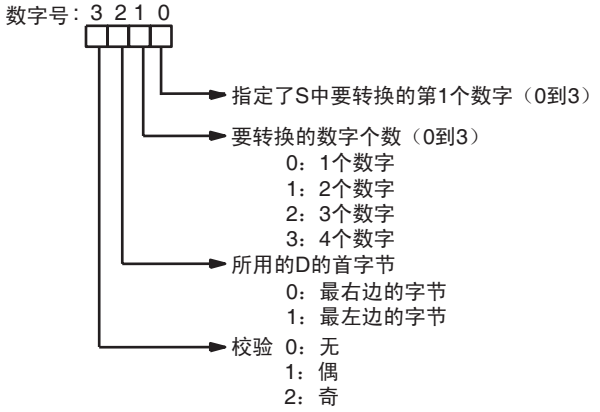
操作数

S：源字

可转换源字中的最多 4 个数字。数字从 0 ～ 3，从右到左。

Di：数字定义

数字定义指定了用于转换的各种参数，如下图所示。



D：第一个目的字

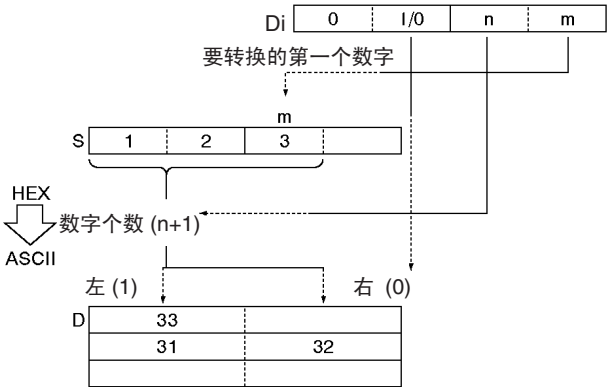
被转换的 ASCII 码数据被写到 D 中从指定字节开始的目的字。如果转换 4 个数字，并且最左边的字节选为 D 中的首字节，则需要 3 个目的字（D ～ D+3）。目的字必须在同一个数据区。

操作数定义

区域	S	Di	D
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---	仅指定值	---
数据寄存器	DR0 ～ DR15		---
索引寄存器	---		
适用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

ASC(086) 以 4 个十六进制数字，处理 S 中的内容，把 S 中指定的数字转换成对应的 8 位 ASCII 码，并把数据写到以 D 中指定的字节开始的目的字中。



注 参阅 CS1 系列编程器操作手册 （W341）中的附录 A，扩展 ASCII 字符表。

校验

可以在数据传输中指定ASCII数据校验用于错误控制，每个ASCII字符的最左边位将对偶，奇或无校验自动调整。

当指定为无校验 (0) 时，最左边位总是 0 ，当指定为偶校验 (1) 时，最左边位将被调整以使置 ON 位的总数为偶数。当规定为奇校验 (2) 时，每个 ASCII 字符的最左边的位将被调整，以使置 ON 位的总数为奇数。校验位的状态不影响 ASCII 代码的意义。

偶校验举例：

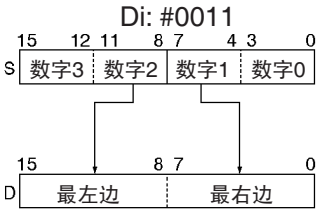
当调整为偶校验时， ASCII 码“31”(00110001) 将为“B1”（10110001：校验位置 ON，生成偶数的置 ON 位）； ASCII 码“36”(00110110) 将为“36”（00110110：校验位仍为 OFF，因为置 ON 位的个数已经为偶数）。

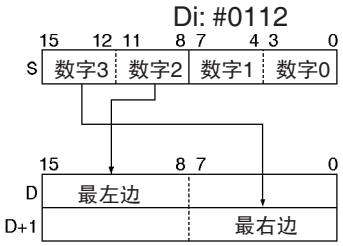
奇校验举例：

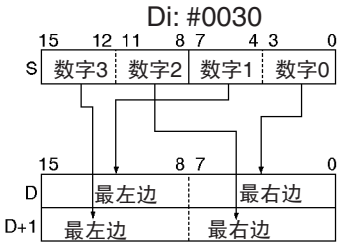
当调整为奇校验时， ASCII“36”(00110110) 将为“B6”（10110110：校验位置 ON，生成奇数的置 ON 位）； ASCII“46”(01000110) 将为“46”（01000110：校验位仍为 OFF，因为置 ON 位的个数已经为奇数）。

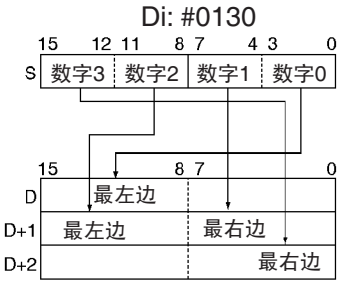
Di 举例

当转换 2 个或 2 个以上数字时， ASC(086) 将从右至左读 S 中的字节，如果需要将绕回到最右边的字节。下图显示了 Di 的一些举例值以及它们的转换。







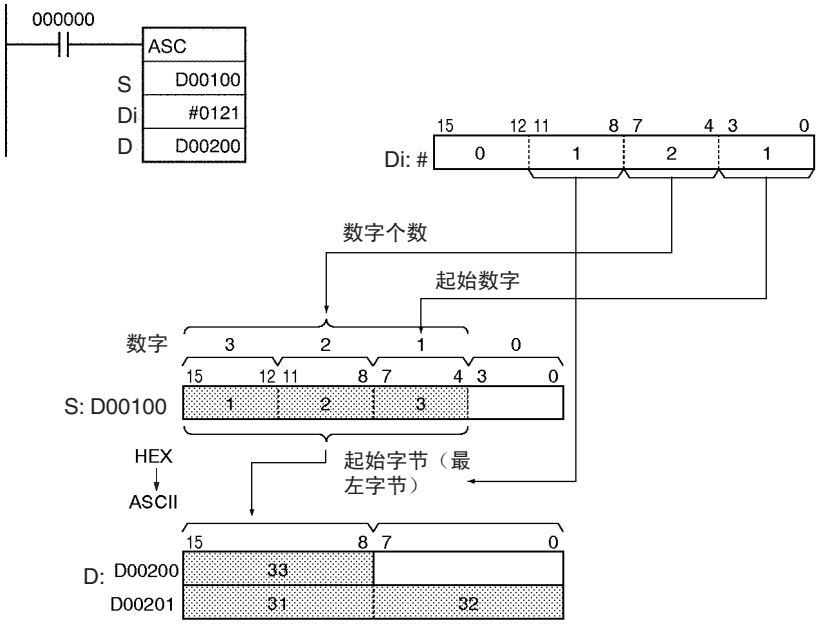


标志

名称	标志	操作
错误标志	ER	如果 Di 不在指定的范围内则置 ON。 其余情况下置 OFF。

例

在下例中，当 CIO 000000 为 ON 时，ACS(086) 把 D0 0100（从数字 1 开始）中的 3 个十六进制数字转换成对应的 ACSII 码，并把此数据写到 D00200 和 D00201, 从 D00200 中的最左字节开始。在此情况下，数字定义 #0121 指定无校验，起始字节 （写入时） = 最左字节，读数字个数 =3，起始数字（读时） = 数字 1。

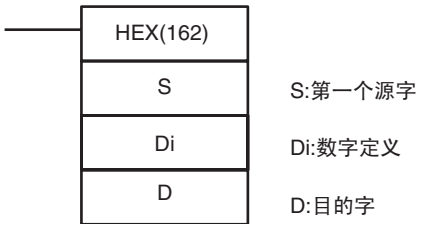


3-12-11 ASCII 码到十六进制：HEX(162)

用途

把源字中最多 4 个字节的 ASCII 数据，转换成对应的十六进制数，并写到指定的目的字中。

梯形图符号



变化

变化	ON 条件时，每个周期执行	HEX(162)
	上升沿微分时执行一次	@HEX(162)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

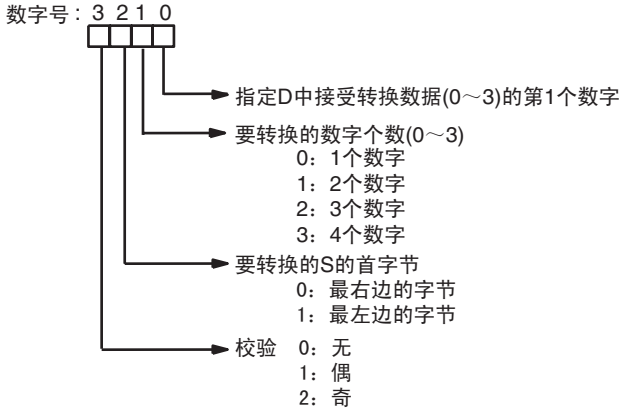
操作数

S：第一个源字

源字的内容以 ASCII 码数据处理，可使用最多 3 个源字。（如果转换 4 个字节，最左字节作为 S 中的首字节需要 3 个源字）。源字必须在同一数据区。

Di：数字定义

数字定义指定了用于转换的各种参数，如下图所示。



D：目的字

转换好的十六进制数字从指定的第一个数字开始，从右至左写入 D。目的字中没有被转换的数字覆盖的所有数字，将被无变化的保留。

操作数定义

区域	S	Di	D
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---	仅指定值	---
数据寄存器	---	DR0 ～ DR15	---
索引寄存器	---		
适用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

HEX(162) 以代表十六进制数字 （0 ～ 9 和 A ～ F）的 ASCII 码数据处理源字的内容，把指定个数的字节转换成十六进制，并把十六进制数据写到从指定数字开始的目的字中。

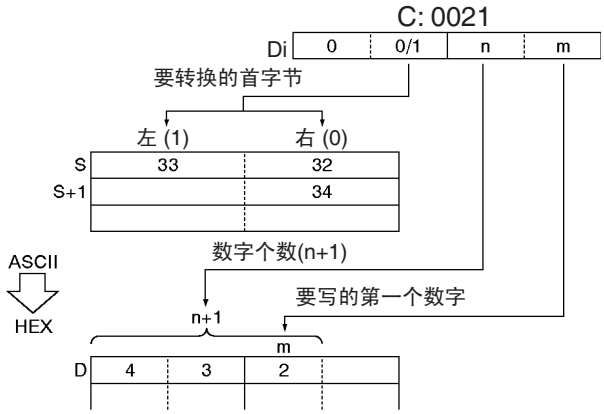
如果源字包含不是十六进制数字的等价 ASCII 码数据，将出现错误。下表显示十六进制数字和它们的等价 ASCII 码 （不包括校验位）。

标志

十六进制数字 （4 位）	等价 ASCII 码 （2 个十六进制数字）
0 ～ 9	30 ～ 39
A ～ F	41 ～ 46

注 参阅 *CS1 系列编程器操作手册 (W341) 的附录 A*, 扩展 ASCII 字符表。

下图显示了 Di=00212 时， HEX(162) 的基本操作。



校验

在数据传输时用于错误控制，可以指定 ASCII 数据的校验。

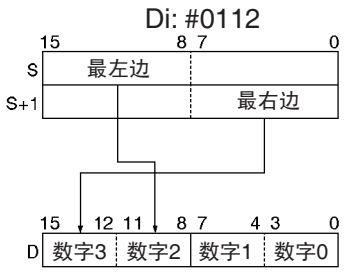
每个字节的最左位是校验位。如果没有校验，校验位总是 0，如果是偶校验，校验位状态将导致偶数个置 ON 位，如果是奇校验，校验位状态将导致奇数个置 ON 位。

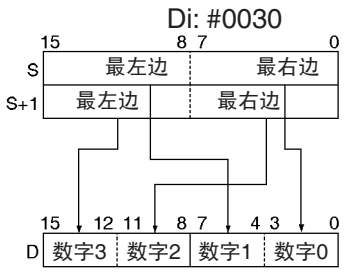
下表显示了 HEX(162) 对于每个校验设置的操作。

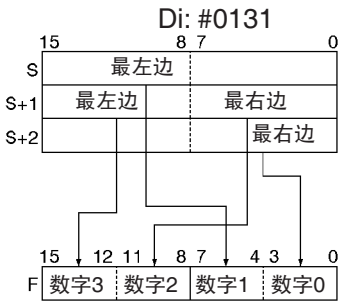
校验设置 （Di 的最左数字）	HEX(162) 的操作
无校验 （0）	仅当每个字节中的校验位为 0 时执行 HEX(162)。如果校验位为非 0，将出现错误。
偶校验 （1）	仅当每个字节中有偶数个置 ON 位时执行 HEX(162)。如果一个字节有奇数个置 ON 位，将出现错误。
奇校验 （2）	仅当每个字节中有奇数个置 ON 位时执行 HEX(162)。如果一个字节有偶数个置 ON 位，将出现错误。

Di 举例

当转换 2 个或更多字节时，HEX(162) 将把转换的数字从右至左写到目的字中，如果需要，将绕回到最右的数字。下图显示了 Di 和它们产生的转换的举例值。







标志

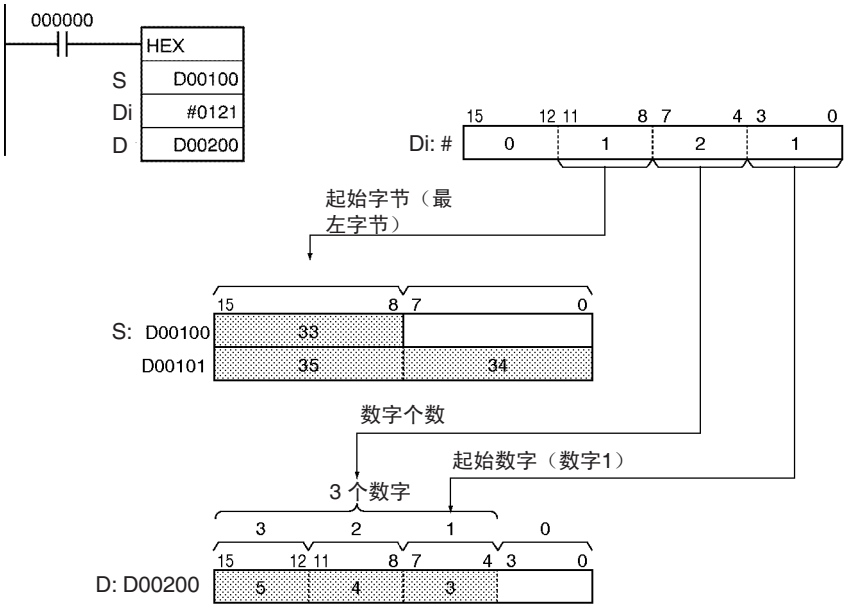
名称	标记	操作
错误标志	ER	如果在 ASCII 数据中有校验错误，则置 ON。 如果源字中的 ASCII 数据不等价于十六进制数则置 ON。 如果 Di 的内容不在指定范围内则置 ON。 其余情况下置 OFF。

注意

如果在 ASCII 数据中有校验错误，源字中的 ASCII 数据不等价于十六进制数，或者 Di 的内容不在指定范围内，将出现错误，并且错误标志将置 ON。

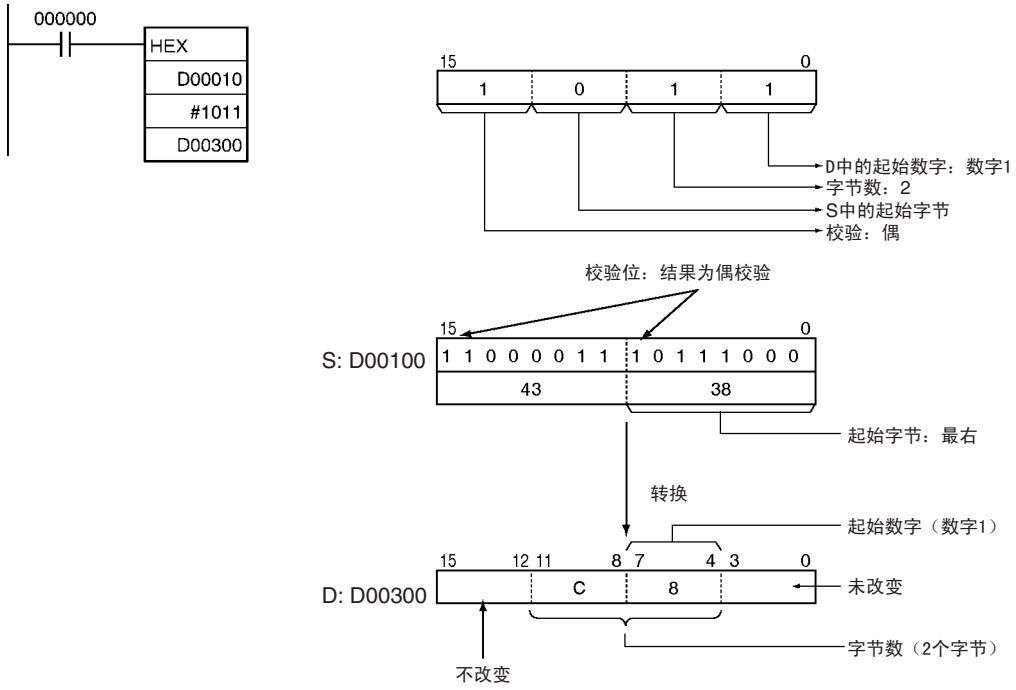
例

在下例中，当 CIO 000000 为 ON 时， HEX(162) 根据数字规定的设置转换 D00100 和 D00101 中的 ASCII 数据 （Di=#0121 指定无校验，起始字节（读时） = 最左字节，读入的字节个数 =3，起始数字 （写时） = 数字 1） HEX(162) 把从 D00100 的最左字节开始的 3 个字节 ASCII 数据 （3 个字符）转换成等价的十六进制数，并把此数据写入 D00200, 从数字 1 开始。



在下例中，当 CIO 000000 为 ON 时， HEX(162) 转换 D00010 中从最右字节开始的 ASCII 数据，并且从 D00300 中数字 1 开始写入等价的十六进制数。

数字定义设置 #1011，指定偶校验，起始字节 （读时） = 最右字节，读的字节个数 =2，并且起始数字 （写时） = 数字 1）。

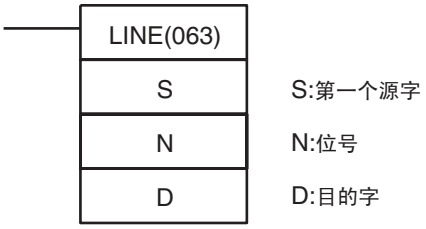


3-12-12 列到行：LINE(063)

用途

把 16 字范围的一列位 （16 个连续字中相同位号）转换成 16 位目的字。

梯形图符号



变化

变化	ON 条件时，每个周期执行	LINE(063)
	上升沿微分时执行一次	@LINE(063)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

S：源首字

指定第一个源字。 S 和 S+15 必须在同一个数据区。

N：位号

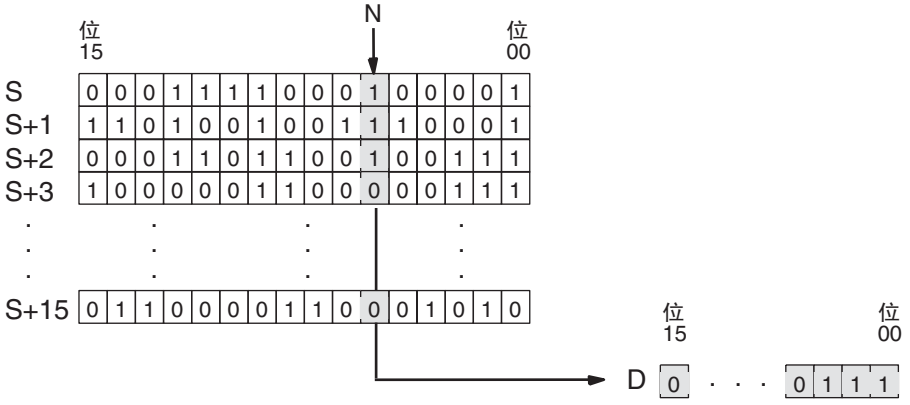
指定从源字中被复制的位号 （0000 ～ 000F 或 &0 ～ &15）。

操作数定义

区域	S	N	D
CIO 区	CIO 0000 ～ CIO 6128	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W496	W000 ～ W511	
保持位区	H000 ～ H496	H000 ～ H511	
辅助位区	A000 ～ A944	A000 ～ A959	A448 ～ A959
定时器区	T0000 ～ T4080	T0000 ～ T4095	
计数器区	C0000 ～ C4080	C0000 ～ C4095	
DM 区	D00000 ～ D32752	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32752	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32752 (n = 0 ～ C)	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---	#0000 ～ 000F（二进制）或 &0 ～ &15	---
数据寄存器	---	DR0 ～ DR15	
索引寄存器	---		
适用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

LINE(063) 把 16 位中位号为 N 的位从 16 字范围 S ～ S+15 复制到目的字 D。S+m 的第 N 位复制到 D 的第 m 位。例如， S 的第 N 位复制到 D 的位 00，S+15 的第 N 位复制到 D 的第 15 位。

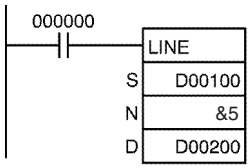


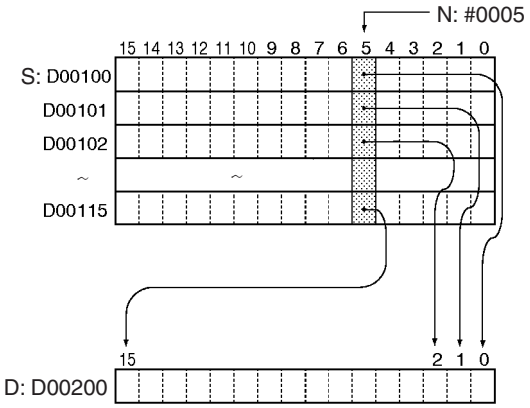
标志

名称	标记	操作
错误标志	ER	如果 N 不在指定的 0000~000F 范围内则置 ON。 其余情况下置 OFF。
等于标志	=	如果执行后 D 为 0000 则置 ON。 其余情况下置 OFF。

例

在下例中，当 CIO 000000 为 ON， LINE(063) 把从 D00100 ～ D00115 的第 5 位复制到 D00200 的 16 个位中。



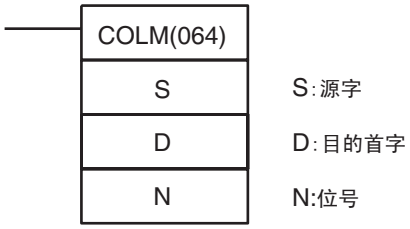


3-12-13 行到列：COLM(064)

用途

把源字的 16 位转换成目的字的 16 字范围内的一列位 （16 个连续字内同一位号）。

梯形图符号



变化

变化	ON 条件时，每个周期执行	COLM(064)
	上升沿微分时执行一次	@COLM(064)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

D：目的首字

指定第一个目的字。 D 和 D+15 必须在同一数据区。

N：位号

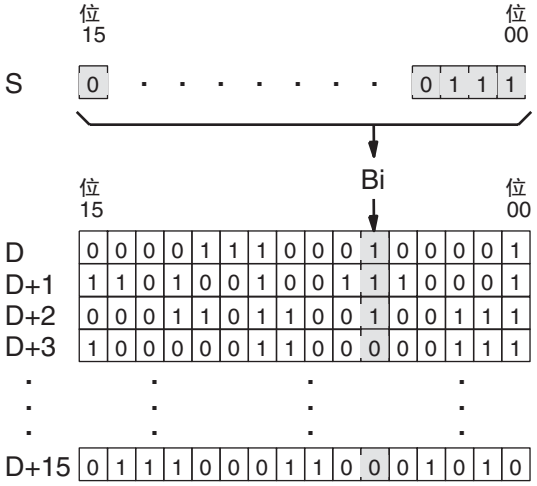
指定被源字覆盖的位号 （0000 ～ 000F 或 &0 ～ &15）。

操作数定义

区域	S	D	N
CIO 区	CIO 0000 ～ CIO 6143	CIO 0000 ～ CIO 6128	CIO 0000 ～ CIO 6143
工作区	W000 ～ W511	W000 ～ W496	W000 ～ W511
保持位区	H000 ～ H511	H000 ～ H496	H000 ～ H511
辅助位区	A000 ～ A959	A448 ～ A944	A000 ～ A959
定时器区	T0000 ～ T4095	T0000 ～ T4080	T0000 ～ T4095
计数器区	C0000 ～ C4095	C0000 ～ C4080	C0000 ～ C4095
DM 区	D00000 ～ D32767	D00000 ～ D32752	D00000 ～ D32767
无区号 EM 区	E00000 ～ E32767	E00000 ～ E32752	E00000 ～ E32767
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	En_00000 ～ En_32752 (n = 0 ～ C)	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #FFFF（二进制）	---	#0000 ～ #000F（二进制）或 &0 ～ &15
数据寄存器	DR0 ～ DR15	---	DR0 ～ DR15
索引寄存器	---		
适用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

COLM(064) 把 S 的 16 位复制到 16 字范围 D ～ D+15 的第 N 位，共 16 位，即 S 的第 00 位复制到 D 的第 N 位， S 的第 15 位复制到 D+15 的第 N 位。

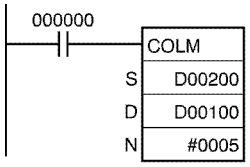


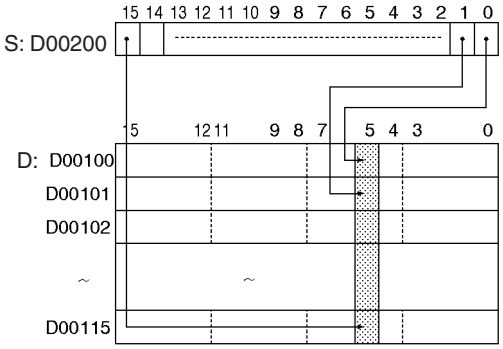
标志

名称	标记	操作
错误标志	ER	如果 N 不在指定的 0000~000F 范围内则置 ON。 其余情况下置 OFF。
等于标志	=	执行后所有 16 个字 D 到 D+15 的第 N 位都为 0 则置 ON。 其余情况下置 OFF。

例

在下例中当 CIO 000000 为 ON 时 COLM(064) 把 D00200 中的 16 位 （位 00 ～ 15）复制到从 D00100 到 D00115 的第 5 位。



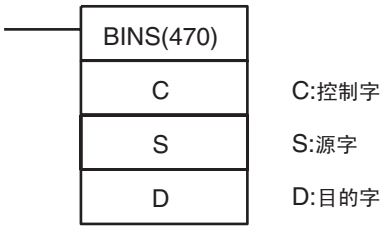


3-12-14 有符号 BCD 到二进制：BINS(470)

用途

有符号 BCD 数据的一个字转换成有符号二进制数据的一个字。

梯形图符号



变化

变化	ON 条件时，每个周期执行	BINS(470)
	上升沿微分时执行一次	@BINS(470)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

C：控制字

指定有符号 BCD 码格式。 C 必须在 0000 ～ 0003 之间。

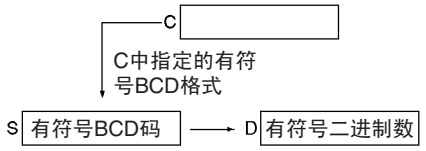
操作数定义

区域	C	S	D
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #0003 （二进制）	---	
数据寄存器	DR0 ～ DR15		

区域	C	S	D
索引寄存器	---		
适用索引寄存器 的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

BINS(470) 把有符号 BCD 数据转换成有符号二进制数据。首先，字 S 中的有符号 BCD 数据格式和范围对照控制字 (C) 中的设置而检查。如果源数据正确， S 中的有符号 BCD 数据转换成有符号二进制数并输出给 D。如果源数据不正确，错误标志将置 ON，并且指令不执行。



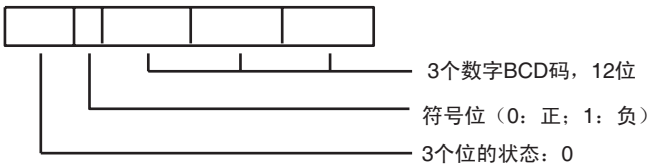
当转换的数据为负，它将作为 2 的补码输出，并且负标志将置 ON。NEG(160) 可用来求出一个有符号二进制负数的绝对值。详细情况参阅 *3-11-5 2 的补码*: NEG(160)。

源数据中的 -0 值将作为 0 来处理，不会引起错误。而且，当 C=0000 时， S 的第 13 到第 15 位的状态不被检查。

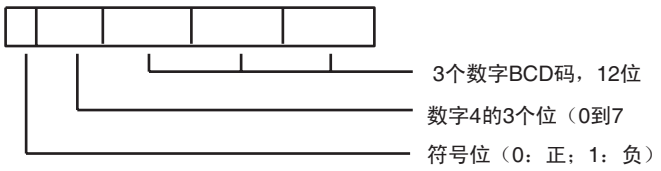
注 一些特殊 I/O 单元输出有符号 BCD 数据。如果首先用 BINS(470) 把数据转换成有符号二进制数，使用此数据的计算一般来说将更简单。

控制字指定有符号 BCD 格式，如下所示。

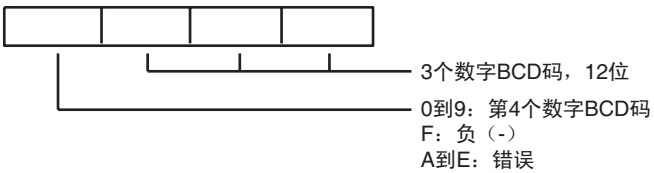
C=0000 （输入数据范围：-999 ～ 999BCD）



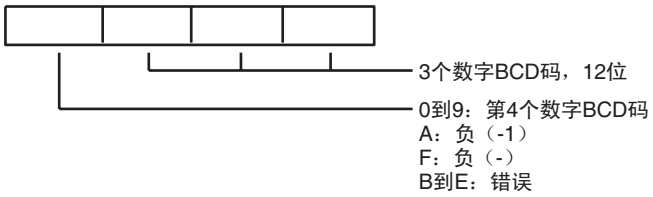
C=0001 （输入数据范围：-7999 ～ 7999BCD）



C=0002 （输入数据范围：-999 ～ 9999BCD）



C=0003 （输入数据范围：-1999 ～ 9999BCD）



下表显示了每个有符号 BCD 格式的可能的 BCD 值，以及相应的有符号二进制值。

设置	有符号 BCD 值	有符号二进制值
C=0000	–999 ～ –1 和 0 ～ 999	FC19 ～ FFFF 和 0000 ～ 03E7
C=0001	–7999 ～ –1 和 0 ～ 7999	E0C1 ～ FFFF 和 0000 ～ 1F3F
C=0002	–999 ～ –1 和 0 ～ 9999	FC19 ～ FFFF 和 0000 ～ 270F
C=0003	–1999 ～ –1 和 0 ～ 9999	F831 ～ FFFF 和 0000 ～ 270F

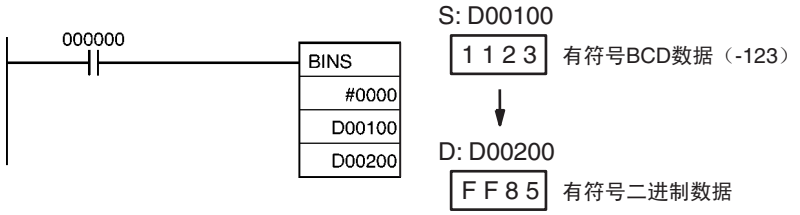
标志

名称	标记	操作
错误标志	ER	如果 C 不在指定的 0000~0003 范围内则置 ON。 如果 C=0002, 并且 S 的最左数字为 A ～ E，则置 ON。 如果 C=0003, 并且 S 的最左数字为 B ～ E，则置 ON。 如果 S 的内容不是 BCD 码，则置 ON。 其余情况下置 OFF。
等于标志	=	如果执行后 D 为 0000，则置 ON。 其余情况下置 OFF。
负标志	N	如果执行后 D 的第 15 位为 ON，则置 ON。 其余情况下置 OFF。

例

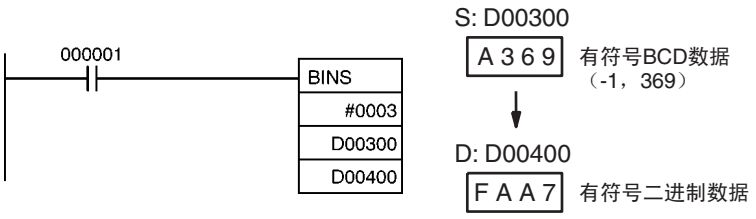
BCD 格式 0(C=#0000)

在下例中，当 CIO 000000 为 ON 时，D00100 中的有符号 BCD 数据格式和范围对照控制字 (0000) 中指定的格式检查。源数据正确，因此 D00100 中的有符号 BCD 数据转换为有符号二进制数，并且输出到 D00200。



BCD 格式 0(C=#0003)

在下例中，当 CIO 000001 为 ON 时，D00100 中的有符号 BCD 数据格式和范围对照控制字 (0003) 中指定的格式检查。源数据正确，因此 D00300 中的有符号 BCD 数据转换为有符号二进制数，并且输出到 D00400。

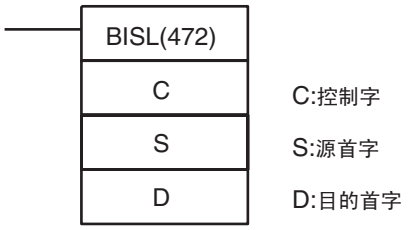


3-12-15 有符号双字 BCD 到二进制：BISL(472)

用途

把有符号双字 BCD 数据转换成有符号双字二进制数据。

梯形图符号



变化

变化	ON 条件时，每个周期执行	BISL(472)
	上升沿微分时执行一次	@BISL(472)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

C：控制字

规定了有符号 BCD 格式。 C 必须在 0000 ～ 0003 之间。

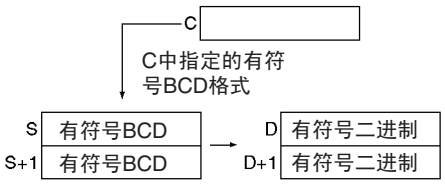
操作数定义

区域	C	S	D
CIO 区	CIO 0000 ～ CIO 6143	CIO 0000 ～ CIO 6142	
工作区	W000 ～ W511	W000 ～ W510	
保持位区	H000 ～ H511	H000 ～ H510	
辅助位区	A000 ～ A959	A000 ～ A958	A448 ～ A958
定时器区	T0000 ～ T4095	T0000 ～ T4094	
计数器区	C0000 ～ C4095	C0000 ～ C4094	
DM 区	D00000 ～ D32767	D00000 ～ D32766	
无区号 EM 区	E00000 ～ E32767	E00000 ～ E32766	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	En_00000 ～ En_32766 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #0003 （二进制）	---	
数据寄存器	DR0 ～ DR15	---	

区域	C	S	D
索引寄存器	---		
适用索引寄存器 的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

BISL(472) 把 S+1 和 S 中的有符号双字 BCD 数据转换成有符号双字二进制数据，并把结果写到 D+1 和 D。首先字 S+1 和 S 中的有符号 BCD 数据格式和范围依照控制字 (C) 中的设置而检查。如果源程序正确，有符号 BCD 数据 S+1 和 S 转换成有符号二进制数，并且输出到 D+1 和 D。如果源数据不正确，错误标志将置 ON，并且指令不执行。



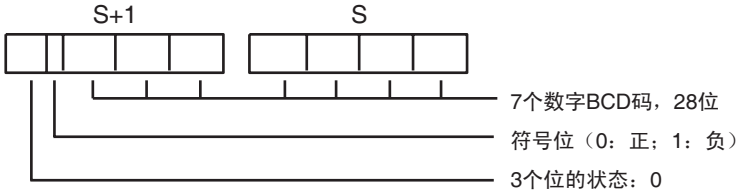
当转换的数据为负，将作为 2 的补码输出，并且负标志将置 ON。 NEGL(161) 可用来求出一个有符号双字二进制负数的绝对值。详细情况参阅 *3-12-6* **双字 *2* 的补码：*NEGL(161)***

源数据中的 -0 值将作为 0 来处理，不会引起错误。而且，当 C=0000 时， S+1 的第 13 到第 15 位的状态不被检查。

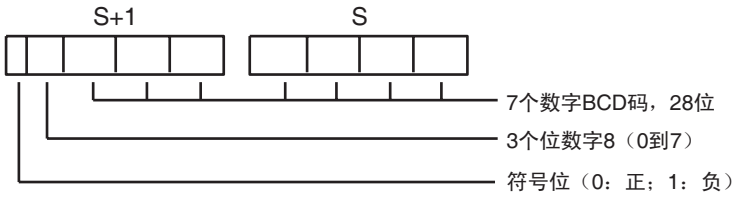
注 一些特殊 I/O 单元输出有符号 BCD 数据。如果首先用 BISL(472) 把数据转换成有符号二进制数，使用此数据的计算一般来说将更简单。

控制字指定有符号 BCD 格式，如下所示。

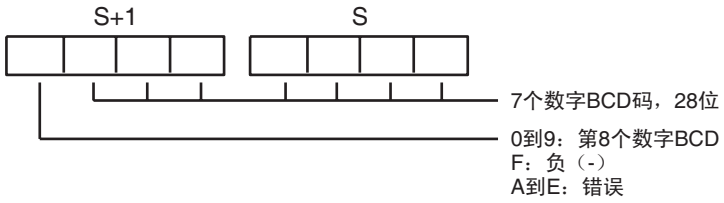
C=0000 （输入数据范围：-9999999 ～ 9999999BCD）



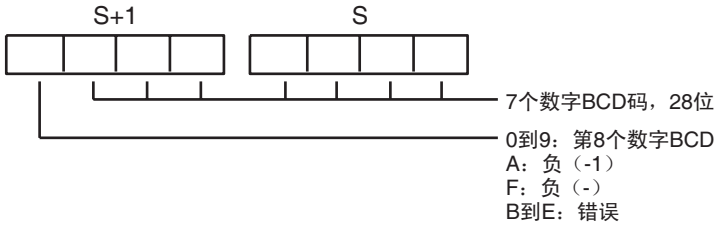
C=0001 （输入数据范围：-79999999 ～ 79999999BCD）



C=0002 （输入数据范围：-9999999 ～ 99999999BCD）



C=0003 （输入数据范围：-19999999 ～ 99999999BCD）



下表显示每个有符号 BCD 格式的可能的 BCD 值以及相应的有符号二进制值。.

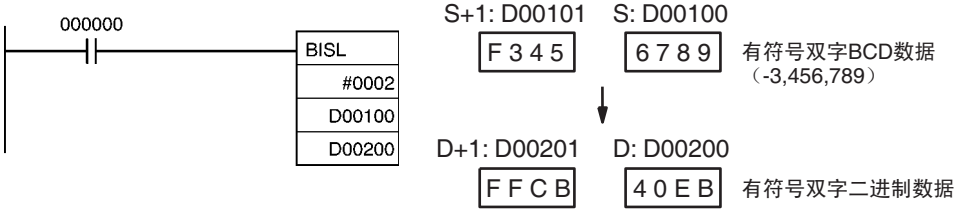
设置	有符号 BCD 值	有符号二进制
C=0000	–999 9999 ～ –1	FF67 6981 ～ FFFF FFFF
	0 ～ 999 9999	0000 0000 ～ 0098 967F
C=0001	–7999 9999 ～ –1	FB3B 4C01 ～ FFFF FFFF
	0 ～ 7999 9999	0000 0000 ～ 04C4 B3FF
C=0002	–999 9999 ～ –1	FF67 6981 ～ FFFF FFFF
	0 ～ 9999 9999	0000 0000 ～ 05F5 E0FF
C=0003	–1999 9999 ～ –1	FECE D301 ～ FFFF FFFF
	0 ～ 9999 9999	0000 0000 ～ 05F5 E0FF

标志

名称	标记	操作
错误标志	ER	如果 C 不在指定的 0000 ～ 0003 范围内则置 ON。 如果 C=0002, 并且 S+1 的最左数字为 A ～ E, 则置 ON。 如果 C=0003, 并且 S+1 的最左数字为 B ～ E, 则置 ON。 如果 S+1 或 S 的内容不是 BCD 码，则置 ON。 其余情况下置 OFF。
等于标志	=	如果执行后 D+1 的内容包括 00000000，则置 ON。 所有其它情况时置 OFF。
负标志	N	如果执行后 D+1 的第 15 位为 ON 则置 ON。 所有其它情况时置 OFF。

例

在下例中当 CIO 000000 为 ON 时，在 D00101 和 D00100 的有符号双字 BCD 数据格式和范围依照控制字 (0002) 中指定的格式而检查。源数据正确，因此 D00101 和 D00100 的有符号双字 BCD 数据转换成有符号双字二进制，并且输出到 D00201 和 D00200。

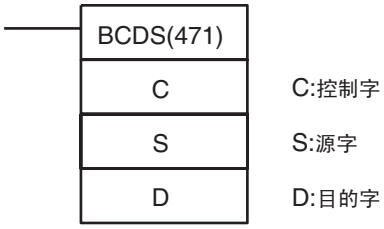


3-12-16 有符号二进制到 BCD：BCDS(471)

用途

有符号二进制数据的一个字转换成有符号 BCD 数据的一个字。

梯形图符号



变化

变化	ON 条件时，每个周期执行	BCDS(471)
	上升沿微分时执行一次	@BCDS(471)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

C：控制字

指定有符号 BCD 格式。 C 必须在 0000 ～ 0003 之间。

S：源字

包括要转换的有符号二进制数据。 S 的内容必须在 C 指定的 BCD 格式的有效范围内。

设置	允许的 S 值
C=0000	FC19 ～ FFFF 或 0000 ～ 03E7
C=0001	E0C1 ～ FFFF 或 0000 ～ 1F3F
C=0002	FC19 ～ FFFF 或 0000 ～ 270F
C=0003	F831 ～ FFFF 或 0000 ～ 270F

D：目的字

包含转换后的有符号 BCD 数据。看下面解释 BCD 格式的说明。

操作数定义

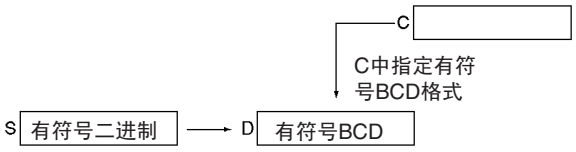
区域	C	S	D
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		

区域	C	S	D
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #0003 （二进制）	---	
数据寄存器	DR0 ～ DR15		
索引寄存器	---		
适用索引寄存器 的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ 1–2048 ～ +2047 ,IR5 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

BCDS(471) 把有符号二进制数据转换成有符号 BCD 数据。首先检查字 S 中的有符号二进制数据来确认此数据在控制字 (C) 指定的有符号 BCD 格式的有效范围内。如果源程序正确，S 中的有符号二进制数据转换为有符号 BCD 码，并且输出到 D。

如果源数据不正确，错误标志将置 ON，并且指令将不被执行。

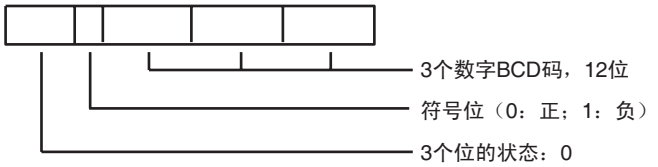


注

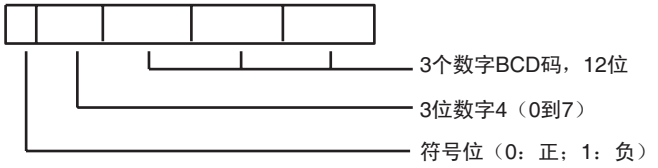
1. 源数据中的 -0 值将作为 0 处理，并且不会引起错误。
2. 一些特殊 I/O 单元需要有符号 BCD 数据输入。 BCDS(471) 可用于把转换有符号二进制数据输出到这些单元。

控制字指定有符号 BCD 格式，如下所示。

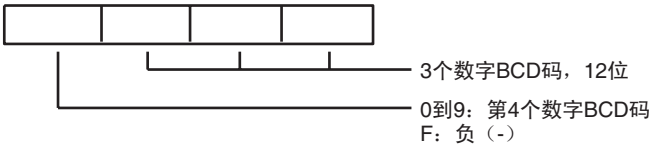
C=0000 （输出数据范围：-999 ～ 999BCD）



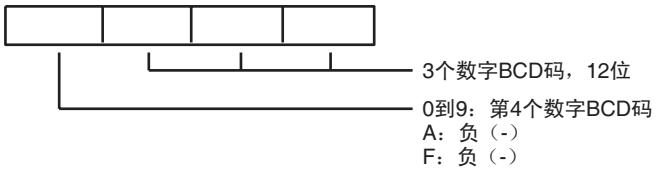
C=0001 （输出数据范围：-7999 ～ 7999BCD）



C=0002 （输出数据范围：-999 ～ 9999BCD）



C=0003 （输入数据范围：-1999 ～ 9999BCD）



下表显示了对每一个有符号 BCD 格式的可能的有符号二进制值。如果源数据不在 有符号 BCD 格式指定允许的范围内，将出现错误。

设置	有符号二进制值	有符号 BCD 值
C=0000	FC19 ～ FFFF 和 0000 ～ 03E7	–999 ～ –1 和 0 ～ 999
C=0001	E0C1 ～ FFFF 和 0000 ～ 1F3F	–7999 ～ –1 和 0 ～ 7999
C=0002	FC19 ～ FFFF 和 0000 ～ 270F	–999 ～ –1 和 0 ～ 9999
C=0003	F831 ～ FFFF 和 0000 ～ 270F	–1999 ～ –1 和 0 ～ 9999

标志

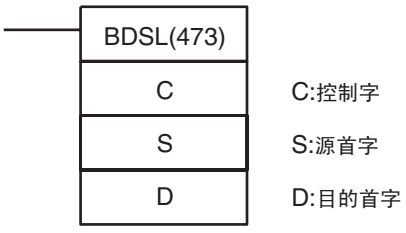
名称	标记	操作
错误标志	ER	如果 C 不在指定的 0000 ～ 0003 范围内则置 ON。 如果 C=0000 ，并且源数据不在允许范围 （FC19 ～ FFFF 和 0000 ～ 03E7）内则置 ON。 如果 C=0001，并且源数据不在允许范围 （E0C1 ～ FFFF 和 0000 ～ 1F3F）内则置 ON。 如果 C=0002，并且源数据不在允许范围 （FC19 ～ FFFF 和 0000 ～ 270F）内则置 ON。 如果 C=0003，并且源数据不在允许范围 （F831 ～ FFFF 和 0000 ～ 270F）内则置 ON。 其余情况下置 OFF。
等于标志	=	如果执行后 D 为 0000，则置 ON。 所有其它情况时置 OFF。
负标志	N	如果 C=0000 或 C=0001, 并且执行后结果的符号位为 ON，则置 ON。 如果 C=0002，并且结果的最左数字为 F 则置 ON。 如果 C=0003，并且结果的最左数字为 A 或 F，则置 ON。 所有其它情况时置 OFF。

3-12-17 有符号双字二进制到 BCD：BDSL(473)

用途

把有符号双字二进制数据转换成有符号双字 BCD 数据。

梯形图符号



变化

变化	ON 条件时，每个周期执行	BDSL(473)
	上升沿微分时执行一次	@BDSL(473)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

操作数

C：控制字

指定有符号 BCD 格式， C 必须在 0000 ～ 0003 之间。

S：源首字

源字 S+1 和 S 包含要转换的有符号双字二进制数据。它们的内容必须在 C 指定的 BCD 格式的有效范围之内。

设置	S+1 和 S 的允许值
C=0000	FF67 6981 ～ FFFF FFFF 或 0000 0000 ～ 0098 967F
C=0001	FB3B 4C01 ～ FFFF FFFF 或 0000 0000 ～ 04C4 B3FF
C=0002	FF67 6981 ～ FFFF FFFF 或 0000 0000 ～ 05F5 E0FF
C=0003	FECE D301 ～ FFFF FFFF 或 0000 0000 ～ 05F5 E0FF

D：目的首字

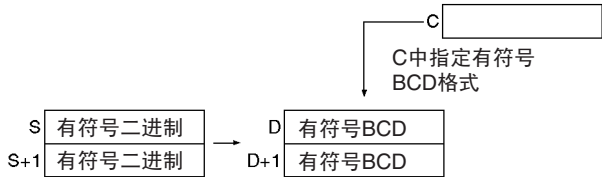
目的字 D+1 和 D 包含转换好的有符号双字 BCD 数据。看下面 BCD 格式解释的说明。

操作数定义

区域	C	S	D
CIO 区	CIO 0000 ～ CIO 6143	CIO 0000 ～ CIO 6142	
工作区	W000 ～ W511	W000 ～ W510	
保持位区	H000 ～ H511	H000 ～ H510	
辅助位区	A000 ～ A959	A000 ～ A958	A448 ～ A958
定时器区	T0000 ～ T4095	T0000 ～ T4094	
计数器区	C0000 ～ C4095	C0000 ～ C4094	
DM 区	D00000 ～ D32767	D00000 ～ D32766	
无区号 EM 区	E00000 ～ E32767	E00000 ～ E32766	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	En_00000 ～ En_32766 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #0003 （二进制）	---	
数据寄存器	DR0 ～ DR15	---	
索引寄存器	---		
适用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

BDSL(473) 把有符号双字二进制数据转换成有符号双字 BCD 数据。首先检查 S+1 和 S 中的有符号双字二进制数据，以确认数据在控制字 （C）指定的有符号 BCD 格式的有效范围内。如果源程序正确， S+1 和 S 中的有符号双字二进制数据转换成有符号双字 BCD 数据，并且输出到 D+1 和 D。如果源数据不正确，错误标志将置 ON，并且指令不会被执行。

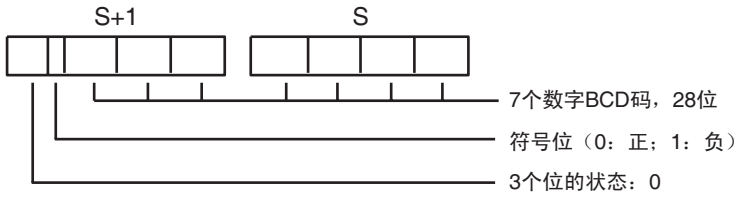


注 1. 源数据中的 -0 值将作为 0 处理，并且不会引起错误。

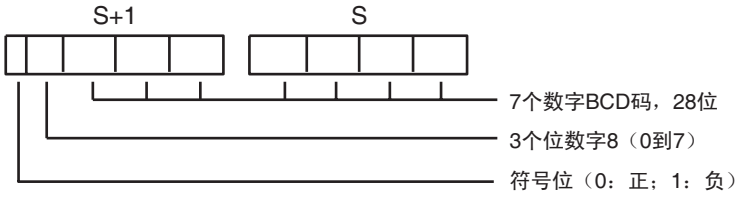
2. 一些特殊 I/O 单元需要有符号 BCD 数据输入。BDSL(473) 可用于转换有符号二进制数据输出到这些单元。

控制字指定将用作结果的有符号 BCD 格式，如下所示。

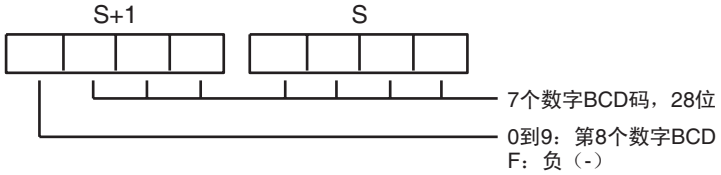
C=0000 （输出数据范围：-9999999 ～ 9999999BCD）



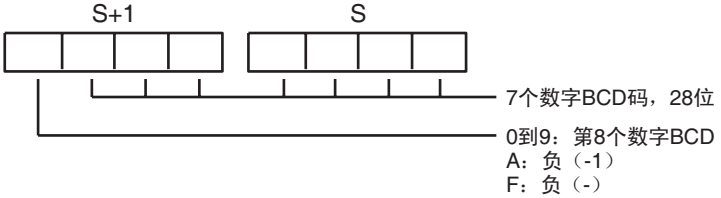
C=0001 （输出数据范围：-79999999 ～ 79999999BCD）



C=0002 （输出数据范围：-9999999 ～ 99999999BCD）



C=0003 （输出数据范围：-19999999 ～ 99999999BCD）



下表显示了每一个有符号 BCD 格式的可能的有符号双字二进制值。如果源数据不在指定的有符号 BCD 格式所允许的范围内，将出现错误。

设置	有符号二进制值	有符号 BCD 值
C=0000	FF67 6981 ～ FFFF FFFF	–999 9999 ～ –1
	0000 0000 ～ 0098 967F	0 ～ 999 9999

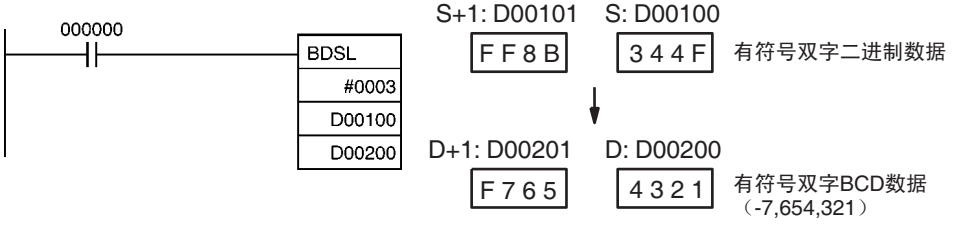
设置	有符号二进制值	有符号 BCD 值
C=0001	FB3B 4C01 ～ FFFF FFFF	–7999 9999 ～ –1
	0000 0000 ～ 04C4 B3FF	0 ～ 7999 9999
C=0002	FF67 6981 ～ FFFF FFFF	–999 9999 ～ –1
	0000 0000 ～ 05F5 E0FF	0 ～ 9999 9999
C=0003	FECE D301 ～ FFFF FFFF	–1999 9999 ～ –1
	0000 0000 ～ 05F5 E0FF	0 ～ 9999 9999

标志

名称	标记	操作
错误标志	ER	如果 C 不在指定的 0000 到 0003 范围内则置 ON。 如果 C=0000, 并且源数据不在范围 （FF67 6981 ～ FFFF FFFF 或 00000000 ～ 0098967F）内则置 ON。 如果 C=0001, 并且源数据不在范围 （FB3B4C01 ～ FFFF FFFF 或 00000000 ～ 04C4B3FF）内则置 ON。 如果 C=0002, 并且源数据不在范围 （FF67 6981 ～ FFFF FFFF 或 0000 0000 ～ 05F5 E0FF）内则置 ON。 如果 C=0003, 并且源数据不在范围 （FECE D301 ～ FFFF FFFF 或 0000 0000 ～ 05F5 E0FF）内则置 ON。 其余情况下置 OFF。
等于标志	=	如果执行后 D 为 0000，则置 ON。 所有其它情况时置 OFF。
负标志	N	如果 C=0000 或 C=0001, 并且执行后结果的符号位为 ON，则置 ON。 如果 C=0002，并且结果的最左数字为 F 则置 ON。 如果 C=0003，并且结果的最左数字为 A 或 F，则置 ON。 所有其它情况时置 OFF。

例

在下例中当 CIO 000000 为 ON，D00101 和 D00100 中的有符号双字二进制数据依照控制字 (0003) 中指定的格式检查，源数据正确，因此 D00101 和 D00100 中的有符号双字二进制数据被转换成有符号双字 BCD 数，并且输出到 D00201 和 D00200。



3-13 逻辑指令

本节描述了执行字数据逻辑操作的指令。

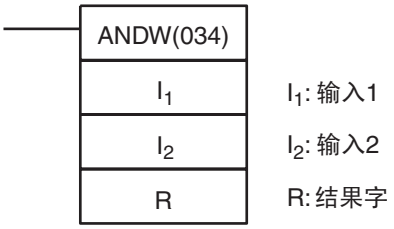
指令	助记符	函数代码	页数
逻辑与	ANDW	034	474
双字逻辑与	ANDL	610	476
逻辑或	ORW	035	477
双字逻辑或	ORWL	611	479
异或	XORW	036	481
双字异或	XORL	612	483
异或非	XNRW	037	485
双字异或非	XNRL	613	486
补码	COM	029	488
双字补码	COML	614	490

3-13-1 逻辑与：ANDW(034)

用途

将一个字的数据和 / 或常数相应位进行逻辑与。

梯形图符号



变化

变化	ON 条件时，每个周期执行	ANDW(034)
	上升沿微分时执行一次	@ANDW(034)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断程序
OK	OK	OK	OK

操作数定义

区域	I_1	I_2	R
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		

区域	I_1	I_2	R
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #FFFF （二进制）		---
数据寄存区	DR0 ～ DR15		
索引寄存区	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

ANDW(034) 将 I_1 和 I_2 指定的数据进行逻辑与，并且把结果输出到 R。

- 逻辑与依次取 I_1 和 I_2 中的相应位。
- 当 I_1 和 I_2 中相应位的内容都是 0 或其中一个是 0，0 将输出到 R 中的相应位。

$I_1, I_2 \rightarrow R$

I_1	I_2	R
1	1	1
1	0	0
0	1	0
0	0	0

标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	结果为 0 时置 ON 所有其它情况时置 OFF
负标志	N	R 的最左边为 1 时置 ON 所有其它情况时置 OFF

注意

执行 ANDW(034) 时，错误标志将置 OFF。

如果由于相与， R 的内容为十六进制 0000，则等于标志将置 ON。

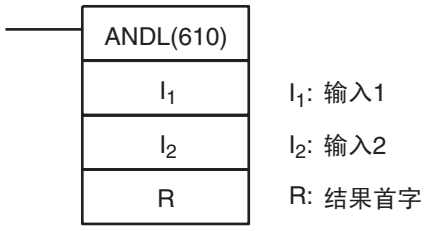
如果由于相与， R 的最左边为 1，则负标志将置 ON。

3-13-2 双字逻辑与：ANDL(610)

用途

将双字的数据和 / 或常数中的相应位进行逻辑与。

梯形图符号



变化

变化	ON 条件时，每个周期执行	ANDL(610)
	上升沿微分时执行一次	@ANDL(610)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断程序
OK	OK	OK	OK

操作数定义

区域	I_1	I_2	R
CIO 区	CIO 0000 ～ CIO 6142		
工作区	W000 ～ W510		
保持位区	H000 ～ H510		
辅助位区	A000 ～ A958		A448 ～ A958
定时器区	T0000 ～ T4094		
计数器区	C0000 ～ C4094		
DM 区	D00000 ～ D32766		
无区号 EM 区	E00000 ～ E32766		
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#00000000 ～ #FFFFFFFF （二进制）		---
数据寄存区	---		
索引寄存区	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

ANDL(610) 将 I_1,I_1+1 和 I_2,I_2+1 中指定的数据进行逻辑与，并且把结果送到 R,R+1。

(I_1, I_1+1), (I_2, I_2+1) → (R, R+1)

I_1, I_1+1	I_2, I_2+1	R, R+1
1	1	1
1	0	0
0	1	0
0	0	0

标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	结果为 0 时置 ON 所有其它情况时置 OFF
负标志	N	R+1 的最左位为 1 时置 ON 所有其它情况时置 OFF

注意

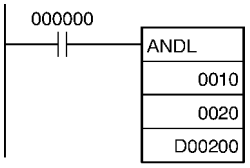
执行 ANDL(610) 时，错误标志将置 OFF。

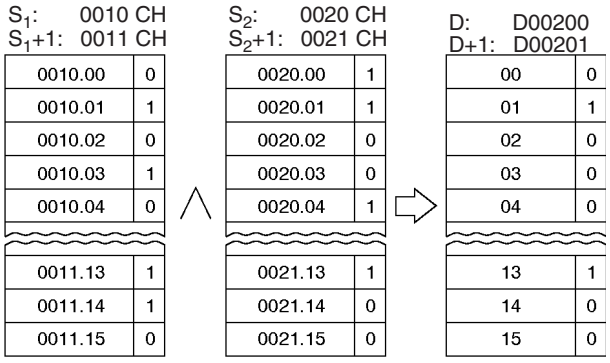
如果由于相与， R,R+1 的内容位十六进制 00000000，则等于标志将置 ON。

如果由于相与， R+1 的最左位为 1，则负标志将置 ON。

例

当执行条件 CIO 000000 为 ON，逻辑与取 CIO 0011,CIO 0010 和 CIO 0021，CIO 0020 中的相应位，并且结果将输出到 D00201 和 D00200 中的相应位。





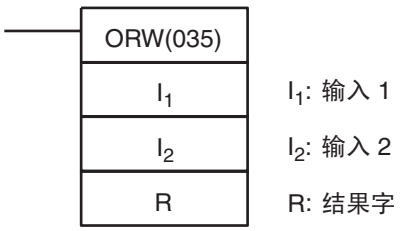
注：纵向箭头表示逻辑与

3-13-3 逻辑或：ORW(035)

用途

将一个字的数据和 / 或常数的相应位进行逻辑或。

梯形图符号



变化

变化	ON 条件时，每个周期执行	ORW(035)
	上升沿微分时执行一次	@ORW(035)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断程序
OK	OK	OK	OK

操作数定义

区域	I_1	I_2	R
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #FFFF （二进制）		---
数据寄存区	DR0 ～ DR15		
索引寄存区	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

ORW(035) 将 I_1 和 I_2 中指定数据进行逻辑或，并且把结果送到 R。

- 逻辑或依次取 I_1 和 I_2 中的相应位。
- 当 I_1 和 I_2 中相应位的任一个是 1 或都是 1， 1 将输出到 R 中的相应位。

$I_1+I_2 \rightarrow R$

I_1	I_2	R
1	1	1
1	0	1

I_1	I_2	R
0	1	1
0	0	0

标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	结果为 0 时置 ON。 所有其它情况时置 OFF。
负标志	N	R 的最左位为 1 时置 ON。 所有其它情况时置 OFF。

注意

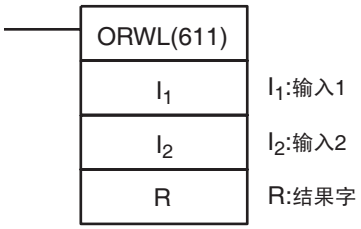
执行 ORW(035) 时，错误标志将置 OFF。
如果由于相或， R 的内容为十六进制 0000，则等于标志将置 ON。
如果由于相或， R 的最左位为 1，则负标志将置 ON。

3-13-4 双字逻辑或：ORWL(611)

用途

将双字的数据和 / 或常数的相应位进行逻辑或。

梯形图符号



变化

变化	ON 条件时，每个周期执行	ORWL(611)
	上升沿微分时执行一次	@ORWL(611)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断程序
OK	OK	OK	OK

操作数定义

区域	I_1	I_2	R
CIO 区	CIO 0000 ～ CIO 6142		
工作区	W000 ～ W510		
保持位区	H000 ～ H510		
辅助位区	A000 ～ A958		A448 ～ A958
定时器区	T0000 ～ T4094		
计数器区	C0000 ～ C4094		
DM 区	D00000 ～ D32766		
无区号 EM 区	E00000 ～ E32766		

区域	I_1	I_2	R
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#00000000 ～ #FFFFFFFF （二进制）		---
数据寄存器	---		
索引寄存器	---		
使用索引寄存器 的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

ORWL(611) 将 I_1 和 I_2 中指定数据作为双字数据进行逻辑或，并把结果输出到 R,R+1.

- 当I_1,I_1+1,I_2和I_2+1的相应位的任意一个为1，1将输出到R+1中的相应位。当二个都为 0， 0 将输出到 R+1 中的相应位。

(I_1,I_1+1) + (I_2,I_2+1) → (R, R+1)

I_1, I_1+1	I_2,I_2+1	R, R+1
1	1	1
1	0	1
0	1	1
0	0	0

标志

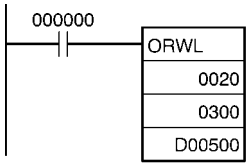
名称	标记	操作
错误标志	ER	OFF
等于标志	=	结果为 0 时置 ON。 所有其它情况时置 OFF。
负标志	N	R+1 的最左位为 1 时置 ON。 所有其它情况时置 OFF。

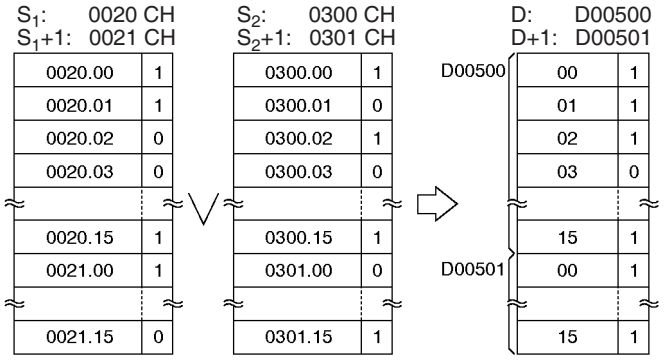
注意

执行 ORWL(611) 时，错误标志将置 OFF。
如果由于相或， R,R+1 的内容为十六进制 00000000，则等于标志将置 ON。
如果由于相或， R+1 的最左位为 1，则负标志将置 ON。

例

当执行条件CIO 000000为ON，逻辑或取CIO 0021,CIO 0020和CIO 0301,CIO0300中的相应位，并且结果将输出到 D00501 和 D00500 中的相应位。





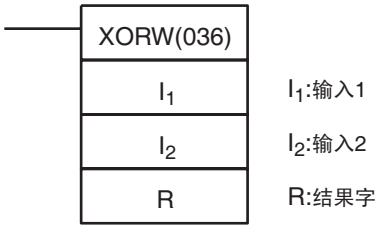
注：纵向箭头表示逻辑或

3-13-5 异或：XORW(036)

用途

将一个字的数据和 / 或常数相应位进行逻辑异或。

梯形图符号



变化

变化	ON 条件时，每个周期执行	XORW(036)
	上升沿微分时执行一次	@XORW(036)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断程序
OK	OK	OK	OK

操作数定义

区域	I_1	I_2	R
CIO Area	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		

区域	I_1	I_2	R
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #FFFF （二进制）		---
数据寄存器	DR0 ～ DR15		
索引寄存器	---		
使用索引寄存器 的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

XORW(036) 将 I_1 和 I_2 中指定的数据进行逻辑异或，并且把结果输出到 R。

- 逻辑异或依次取 I_1 和 I_2 中的相应位。
- 当 I_1 和 I_2 的相应位的内容不同时，1 将输出到 R 的相应位，当相同时，0 将输出到 R 中的相应位。

$I_1,I_2+I_1,I_2 \rightarrow R$

I_1	I_2	R
1	1	0
1	0	1
0	1	1
0	0	0

标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	结果为 0 时置 ON。 所有其它情况时置 OFF。
负标志	N	R 的最左位为 1 时置 ON。 所有其它情况时置 OFF。

注意

执行 XORW(036) 时，错误标志将置 OFF。

如果由于相异或， R 的内容为十六进制 0000，则等于标志将置 ON。

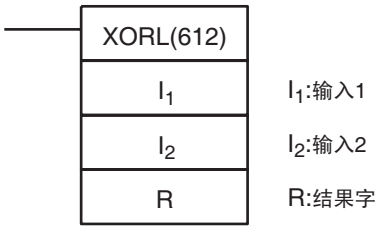
如果由于相异或， R 的最左位为 1，则负标志将置 ON。

3-13-6 双字异或：XORL(612)

用途

将双字的数据和 / 或常数相应位进行逻辑异或。

梯形图符号



变化

变化	ON 条件时，每个周期执行	XORL(612)
	上升沿微分时执行一次	@XORL(612)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断程序
OK	OK	OK	OK

操作数定义

区域	I_1	I_2	R
CIO 区	CIO 0000 ～ CIO 6142		
工作区	W000 ～ W510		
保持位区	H000 ～ H510		
辅助位区	A000 ～ A958		A448 ～ A958
定时器区	T0000 ～ T4094		
计数器区	C0000 ～ C4094		
DM 区	D00000 ～ D32766		
无区号 EM 区	E00000 ～ E32766		
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#00000000 ～ #FFFFFFFF （二进制）		---
数据寄存器	---		
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

XORL(612) 取 I_1 和 I_2 中指定的数据的逻辑异或。以双字数据形式表示，并把结果送到 R， R+1。

- 当I_1,I_1+1，I_2和I_2+1中的相应位的内容不相同，1将输出到R,R+1的相应位，当相应位相同时， 0 将输出到 R,R+1 中的相应位。

$(I_1,I_1+1), (\overline{I_2,I_2+1}) + (\overline{I_1,I_1+1}), (I_2,I_2+1) \rightarrow (R, R+1)$

I_1,I_1+1	I_2,I_2+1	R, R+1
1	1	0
1	0	1
0	1	1
0	0	0

标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	结果为 0 时置 ON。 所有其它情况时置 OFF。
负标志	N	R+1 的最左位为 1 时置 ON。 所有其它情况时置 OFF。

注意

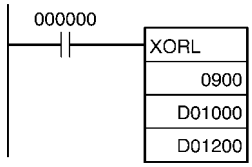
执行 XORL(612) 时，错误标志将置 OFF。

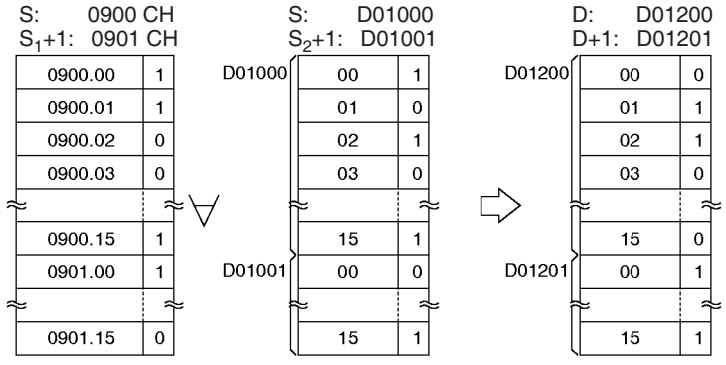
如果由于异或， R,R+1 的内容为十六进制 00000000，则等于标志将置 ON。

如果由于异或， R+1 的最左位为 1，则负标志将置 ON。

例

当执行条件 CIO 000000 为 ON，逻辑异或取 CIO 0901,CIO 0900 和 D01001,D01000 中的相应位，并把结果送到 D01201 和 D01200 中的相应位。





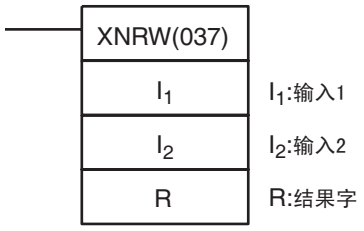
注：符号表示逻辑异或

3-13-7 异或非：XNRW(037)

用途

将一个字的数据和 / 或常数的相应位进行逻辑异或非。

梯形图符号



变化

变化	ON 条件时，每个周期执行	XNRW(037)
	上升沿微分时执行一次	@XNRW(037)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断程序
OK	OK	OK	OK

操作数定义

区域	I_1	I_2	R
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #FFFF （二进制）		---
数据寄存器	DR0 ～ DR15		
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

XNRW(037) 将 I_1 和 I_2 中指定的数据进行逻辑异或非，并把结果送到 R。

- 逻辑异或非依次取 I_1 和 I_2 中的相应位。
- 当 I_1 和 I_2 的相应位的内容不同时，0 将输出到 R 的相应位，当它们同时，1 将输出到 R 的相应位。

$I_1, I_2 + I_1, \overline{I_2} \rightarrow R$

I_1	I_2	R
1	1	1
1	0	0
0	1	0
0	0	1

标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	结果为 0 时置 ON。 所有其它情况时置 OFF。
负标志	N	R 的最左位为 1 时置 ON。 所有其它情况时置 OFF。

注意

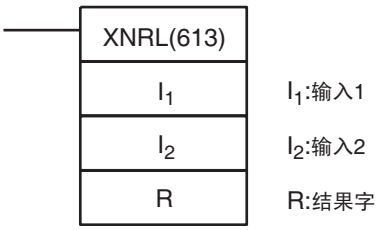
执行 XNRW(037) 时，错误标志将置 OFF。
如果由于异或非， R 的内容为十六进制 0000，则等于标志将置 ON。
如果由于异或非， R 的最左位为 1，则负标志将置 ON。

3-13-8 双字异或非：XNRL(613)

用途

将双字的数据和 / 或常数的相应位进行逻辑异或非。

梯形图符号



变化

变化	ON 条件时，每个周期执行	XNRL(613)
	上升沿微分时执行一次	@XNRL(613)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	块程序区	子程序	中断程序
OK	OK	OK	OK

操作数定义

区域	I_1	I_2	R
CIO 区	CIO 0000 ～ CIO 6142		
工作区	W000 ～ W 510		
保持位区	H000 ～ H510		
辅助位区	A000 ～ A958		A448 ～ A958
定时器区	T0000 ～ T4094		
计数器区	C0000 ～ C4094		
DM 区	D00000 ～ D32766		
无区号 EM 区	E00000 ～ E32766		
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#00000000 ～ #FFFFFFFF （二进制）		---
数据寄存器	---		
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

XNRL(613) 将 I_1 和 I_2 中指定的数据进行逻辑异或非，并把结果送到 R,R+1。

- 当 I_1，I_1+1 和 I_2，I_2+1 中相应位的内容不同时，0 将输出到 R,R+1 中的相应位。当它们相应位相同时， 1 将输出到 R,R+1 中的相应位。

$(I_1,I_1+1), (I_2,I_2+1) + (\overline{I_1,I_1+1}), (\overline{I_2,I_2+1}) \rightarrow (R, R+1)$

I_1,I_1+1	I_2,I_2+1	R, R+1
1	1	1
1	0	0
0	1	0
0	0	1

标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	结果为 0 时置 ON。 所有其它情况时置 OFF。
负标志	N	R+1 的最左位为 1 时置 ON。 所有其它情况时置 OFF。

注意

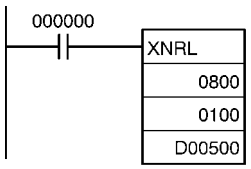
执行 XNRL(613) 时，错误标志将置 OFF。

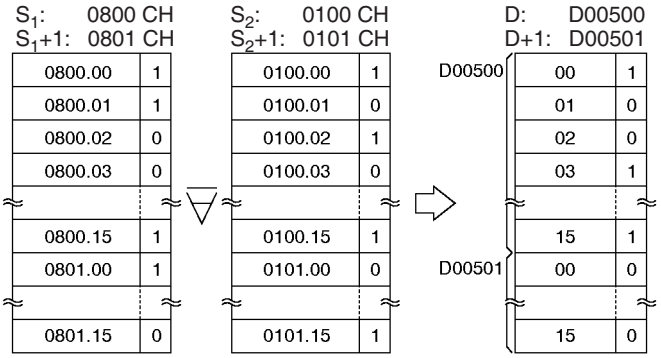
如果由于异或非，R,R+1 的内容为十六进制 00000000，则等于标志将置 ON。

如果由于异或非， R+1 的最左位为 1，则负标志将置 ON。

例

当执行条件 CIO 000000 为 ON，逻辑异或取 CIO 0801,CIO 0800 和 CIO 0101,CIO 0100 中的相应位，并把结果送到 D00501 和 D00500 中的相应位。





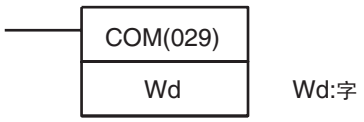
注：符号表示逻辑异或非

3-13-9 补码：COM(029)

用途

把 Wd 中所有置 ON 的位置为 OFF，所有置 OFF 的位置为 ON。

梯形图符号



变化

变化	ON 条件时，每个周期执行	COM(029)
	上升沿微分时执行一次	@COM(029)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断程序
OK	OK	OK	OK

操作数定义

区域	Wd
CIO 区	CIO 0000 ～ CIO 6143
工作区	W000 ～ W511
保持位区	H000 ～ H511
辅助位区	A448 ～ A959
定时器区	T0000 ～ T4095
计数器区	C0000 ～ C4095
DM 区	D00000 ～ D32767

区域	Wd
无区号 EM 区	E00000 ～ E32767
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---
数据寄存器	DR0 ～ DR15
索引寄存区	---
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

说明

COM(029) 把 Wd 中的每一个指定位的状态取反
$\overline{Wd} \rightarrow Wd$: 1 → 0 和 0 → 1

注 使用COM指令时，注意当执行条件为ON，每个位的状态在每个循环都将改变。

标志

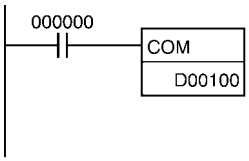
名称	标记	操作
错误标志	ER	OFF
等于标志	=	结果为 0 时置 ON。 所有其它情况时置 OFF。
负标志	N	R 的最左位为 1 时置 ON。 所有其它情况时置 OFF。

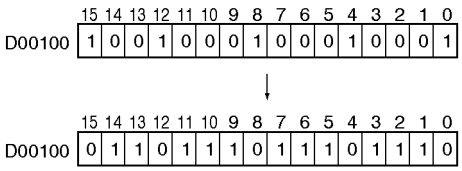
注意

执行 COM(029) 时，错误标志将置 OFF。
如果 COM 指令的结果， R 的内容为十六进制 0000，则等于标志将置 ON。
如果 COM 指令的结果， R 的最左位为 1，则负标志将置 ON。

例

在下例中，当 CIO 000000 为 ON，则 D00100 的每个位的状态将取反。



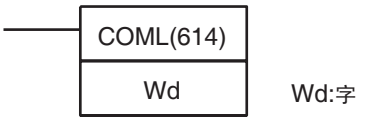


3-13-10 双字补码：COML(614)

用途

把 Wd 和 Wd+1 中所有置 ON 的位置为 OFF，所有置 OFF 的位置为 ON。

梯形图符号



变化

变化	ON 条件时，每个周期执行	COML(614)
	上升沿微分时执行一次	@COML(614)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断程序
OK	OK	OK	OK

操作数定义

区域	Wd
CIO 区	CIO 0000 ～ CIO 6142
工作区	W000 ～ W510
保持位区	H000 ～ H510
辅助位区	A448 ～ A958
定时器区	T0000 ～ T4094
计数器区	C0000 ～ C4094
DM 区	D00000 ～ D32766
无区号 EM 区	E00000 ～ E32766
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---
数据寄存器	---
索引寄存器	---
使用索引寄存器 的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

说明

COML(614) 把 Wd 和 Wd+1 中的每一个指定位的状态取反。
(Wd+1, Wd)→(Wd+1, Wd)

注 使用COM指令时，注意当执行条件为ON，每个位的状态在每个循环都将改变。

标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	结果为 0 时置 ON。 所有其它情况时置 OFF。
负标志	N	R+1 的最左位为 1 时置 ON。 所有其它情况时置 OFF。

注意

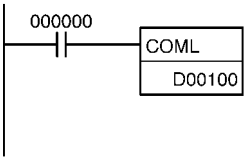
执行 COML(614) 时，错误标志将置 OFF。

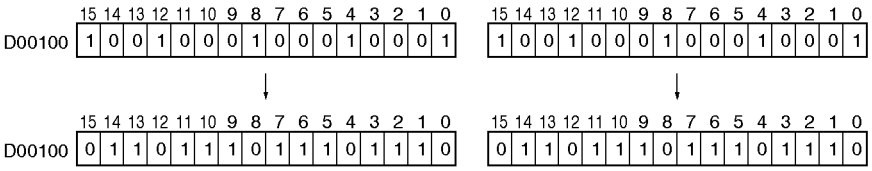
如果 COML 指令的结果，R 和 R+1 的内容为十六进制 00000000，则等于标志将置 ON。

如果 COML 指令的结果， R+1 的最左位为 1，则负标志将置 ON。

例

在下例中，当CIO 000000为ON，则D00100和D00101的每个位的状态将取反。





3-14 特殊算术指令

本节描述了用于特殊算术计算的指令。

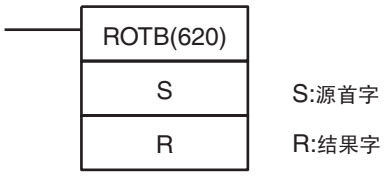
指令	助记符	函数代码	页数
二进制平方根	ROTB	620	491
BCD 平方根	ROOT	072	493
算术处理	APR	069	497
浮点除	FDIV	079	508
位计数器	BCNT	067	513

3-14-1 二进制平方根：ROTB(620)

用途

计算指定字的 32 位有符号二进制内容 （正值）的平方根，并且把结果的整数部分输出到指定的结果字。

梯形图符号



变化

变化	ON 条件时，每个周期执行	ROTB(620)
	上升沿微分时执行一次	@ROTB(620)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

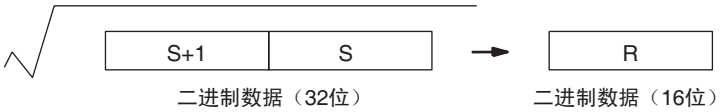
块程序区	步程序区	子程序	中断程序
OK	OK	OK	OK

操作数定义

区域	S	R
CIO 区	CIO 0000 ～ CIO 6142	CIO 0000 ～ CIO 6143
工作区	W000 ～ W510	W000 ～ W511
保持位区	H000 ～ H510	H000 ～ H511
辅助位区	A000 ～ A958	A448 ～ A959
定时器区	T0000 ～ T4094	T0000 ～ T4095
计数器区	C0000 ～ C4094	C0000 ～ C4095
DM 区	D00000 ～ D32766	D00000 ～ D32767
无区号 EM 区	E00000 ～ E32766	E00000 ～ E32767
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#00000000 ～ #FFFFFFFF (二进制)	---
数据寄存器	DR0 ～ DR15	
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

说明

ROTB(620) 计算 S+1 和 S 中的 32 位二进制数的平方根，并把结果的整数部分输出到 R。非整数余数被舍去。



为字S+1和S指定的数据范围是00000000到3FFFFFFFF。如果指定从4000000到 7FFFFFF 的一个数，它将用 3FFFFFF 作平方根计算。如果源字内容大于 7FFFFFF, 即 S+1 的第 15 位为 1 ，则将出现错误。

标志

名称	标记	操作
错误标志	ER	S+1 的第 15 位为 1 置 ON。 其余情况下置 OFF。
等于标志	=	结果为 0000 时置 ON。 所有其它情况时置 OFF。
上溢出标志	OF	如果 R+1 和 R 的内容为 40000000 到 7FFFFFFF 则置 ON。 所有其它情况时置 OFF。
下溢出标志	UF	OFF
负标志	N	OFF

注意

S+1 和 S 的内容必须小于 80000000。
该指令的操作数 （S+1,S 和 R）都作为二进制处理。
如果输入数据为 BCD 码，使用 ROOT(072) 指令。

例

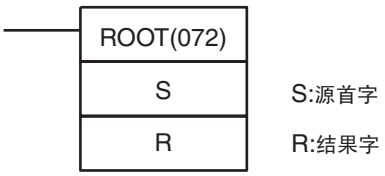
在下例中档 CIO 000000 为 ON 时，ROTB(620) 计算 CIO 0002 和 CIO 0001 中数据的平方根，并把结果的整数部分写到 D00100。

3-14-2 BCD 平方根：ROOT(072)

用途

计算 8 个数字 BCD 的平方根，并把结果的整数部分输出到指定的结果字。

梯形图符号



变化

变化	ON 条件时，每个周期执行	ROOT(072)
	上升沿微分时执行一次	@ROOT(072)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断程序
OK	OK	OK	OK

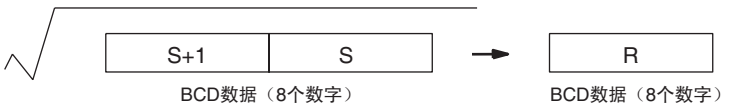
操作数定义

区域	S	R
CIO 区	CIO 0000 ～ CIO 6142	CIO 0000 ～ CIO 6143
工作区	W000 ～ W510	W000 ～ W511
保持位区	H000 ～ H510	H000 ～ H511
辅助位区	A000 ～ A958	A448 ～ A959

区域	S	R
定时器区	T0000 ～ T4094	T0000 ～ T4095
计数器区	C0000 ～ C4094	C0000 ～ C4095
DM 区	D00000 ～ D32766	D00000 ～ D32767
无区号 EM 区	E00000 ～ E32766	E00000 ～ E32767
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#00000000 ～ #99999999 (BCD)	---
数据寄存器	---	DR0 ～ DR15
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

说明

ROOT(072) 计算 S+1 和 S 中的 8 个数字 BCD 数的平方根，并把结果的整数部分输出到 R。非整数余数被舍去。



标志

名称	标记	操作
错误标志	ER	如果 S+1 和 S 中的数据不是 BCD 码则置 ON。 其余情况下置 OFF。
等于标志	=	结果为 0000 时置 ON。 所有其它情况时置 OFF。

注意

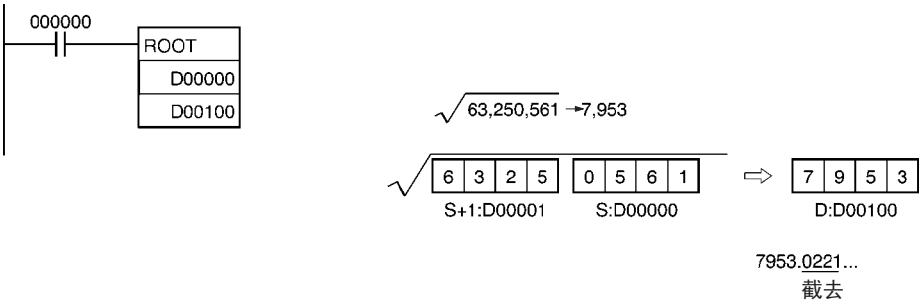
该指令的操作数（S+1,S 和 R）都作为 BCD 码处理。如果输入数据是二进制，使用 ROTB(620) 指令。

例

8 个数字数的平方根

在下例中当 CIO 000000 为 ON 时， ROOT(072) 计算 D000001 和 D00000 中数据的平方根，并把结果的整数部分写到 D00100。

注 8 个数字小数点后的数字将截去。



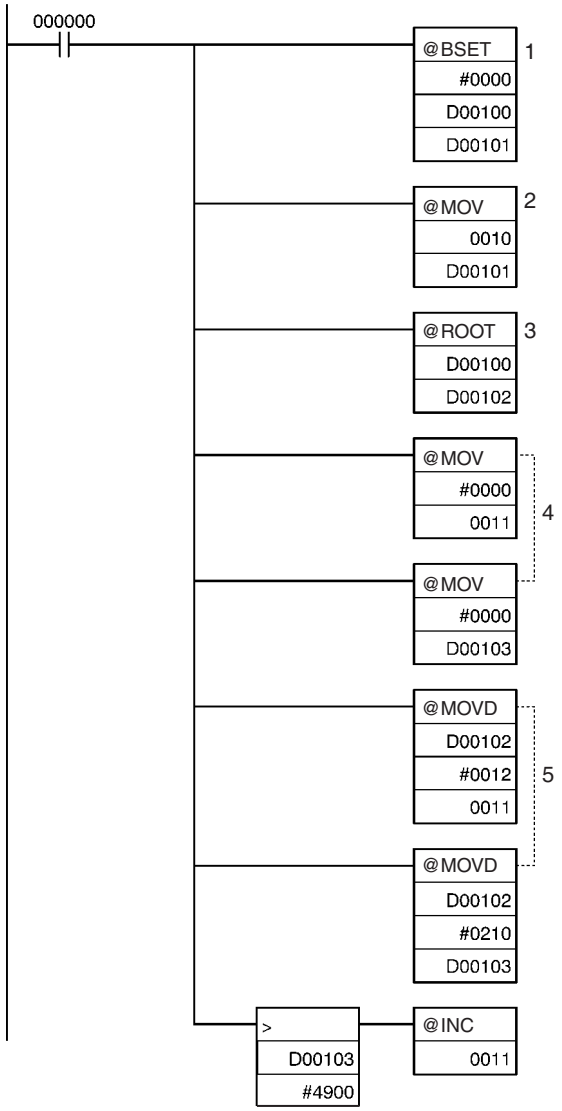
4 个数字数的平方根

下例表示了怎样取得 4 个数字数的平方根，并把结果四舍五入。此程序例计算 CIO 0010 中的 4 个数字数的平方根，把结果四舍五入并写到 CIO 0011。（首先，4个数字数乘以10000 。并把结果除以100，增加了100倍计算的精确度）。

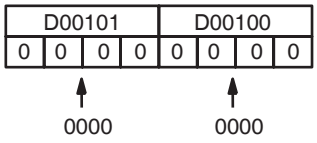
注 小数点后的数字被四舍五入为 4 个数字数。

$\sqrt{6017}$=77.56....→78

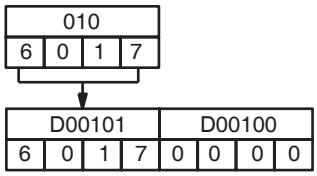
小数点后的值将被四舍五入。



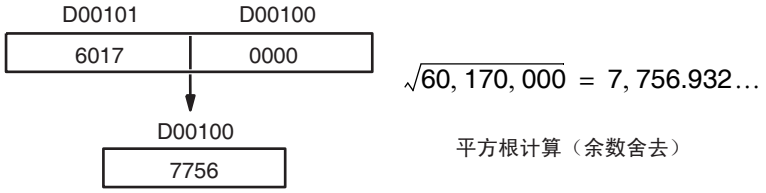
1,2,3... 1. 源字 （D00101 和 D00100）被清为 00000000。



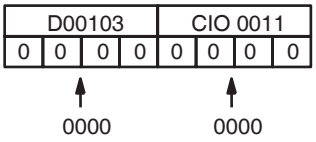
2. 4 个数字被传送到 D00101。



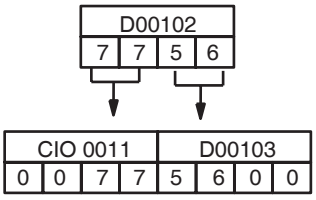
3. ROOT(072) 计算 D00101 和 D00100 的平方根并把结果写到 D00102。



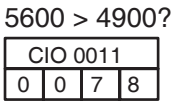
4. D00103 和结果字 (CIO 0011) 被清为 00000000。



5. 平方根计算的结果除以 100，整数部分写到 CIO 0011, 余数写到 D00103。



6. 如果D00103的内容大于4900，CIO 0011增加1，在此情况下，结果为78 。

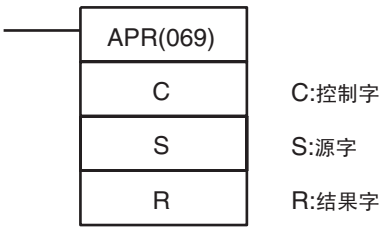


3-14-3 算术处理：APR(069)

用途

计算源数据的正弦，余弦或线性归纳。
线性归纳功能允许 X 和 Y 之间的任何关系近似于线段。

梯形图符号



变化

变化	ON 条件时，每个周期执行	APR(069)
	上升沿微分时执行一次	@APR(069)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断程序
OK	OK	OK	OK

操作数

正弦函数 (C=0000 Hex)

操作数	数值	数据范围
C	0000 Hex	---
S	0000 ～ 0900 (BCD)	0° ～ 90°
D	0000 ～ 9999 (BCD)	0.0000 ～ 0.9999
	9999 (BCD)	1.0000

余弦函数 (C=0001 Hex)

操作数	数值	数据范围
C	0001 Hex	---
S	0000 ～ 0900 (BCD)	0° ～ 90°
D	0000 ～ 9999 (BCD)	0.0000 ～ 0.9999
	9999 (BCD)	1.0000

线性归纳函数 (C= 数据域地址)

操作数	数值	数据范围
C	数据域地址	---
S	16 位无符号 BCD 码	0000 ～ 9999
	16 位无符号二进制数	0 ～ 65,535
	16 位有符号二进制数 [1]	–32,768 ～ 32,767
	2 位有符号二进制数 [1]	–2,147,483,648 ～ 2,147,483,647
	浮点数 [1]	$-\infty$, $-3.402823 \times 10^{38} \sim -1.175494 \times 10^{-38}$, $1.175494 \times 10^{-38} \sim 3.402823 \times 10^{38}$, $+\infty$
D	16 位无符号 BCD 码	0000 ～ 9999
	16 位无符号二进制数	0 ～ 65,535
	16 位有符号二进制数 [1]	–32,768 ～ 32,767
	2 位有符号二进制数 [1]	–2,147,483,648 ～ 2,147,483,647
	浮点数 [1]	$-\infty$, $-3.402823 \times 10^{38} \sim -1.175494 \times 10^{-38}$, $1.175494 \times 10^{-38} \sim 3.402823 \times 10^{38}$, $+\infty$

注 1. 有符号二进制数和浮点数均只由 CS1-H,CJ1-H 和 CJ1M CPU 单元支持。

2. 如果 C 是一个字地址， APR(069) 根据从 C 开始的一个表中预先输入的坐标 （形成线段），对 S 中的 X 值归纳 Y 值。详细请参见下面说明部分。

操作数定义

区域	C	S	R
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		

区域	C	S	R
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	仅为指定值		---
数据寄存器	---	DR0 ～ DR15	
索引寄存器	---		
使用索引寄存器 的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

APR （069）的运行依靠控制字 C。如果 C 是 0000 或 0001， APR(069) 计算 S 的正弦或余弦值， S 以十分之一度为单位。

如果 C 是一个字地址， APR(069) 根据从 C 开始的一个表中预先输入的坐标（形成线段），对 S 中的 X 值归纳 Y 值。

正弦函数 (C=0000)

当 C 为 0000，APR（069）计算 SIN（S）并把结果写到 R。S 的范围为 0000 到 0900BCD （0.0° ～ 90.0° ） R 的范围为 0000 到 9999BCD （0.0000 到 0.9999），结果中小数点后第四位以后的数被舍去。

余弦函数 (C=0001)

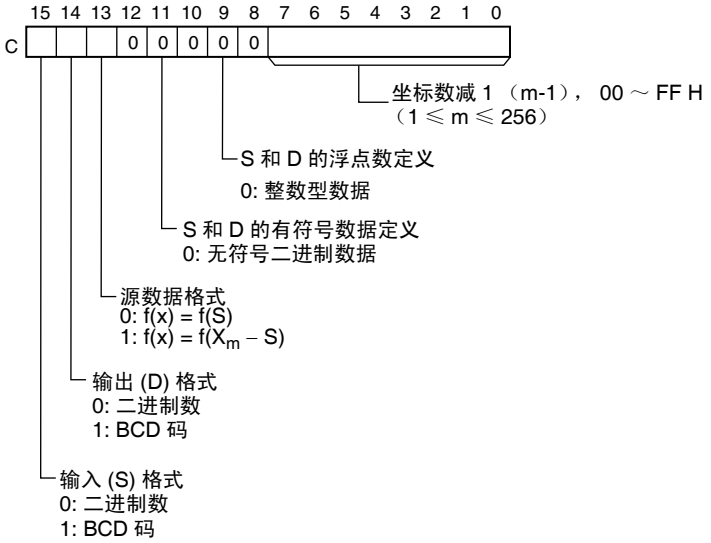
当 C 为 0001，APR(069) 计算 COS （S）并把结果写到 R。 S 的范围为 0000 到 0900BCD （ 0.0° ～ 90.0° ） R 的范围为 0000 到 9999BCD （0.0000 到 0.9999），结果中小数点后第四位以后的数被舍去。

线性归纳

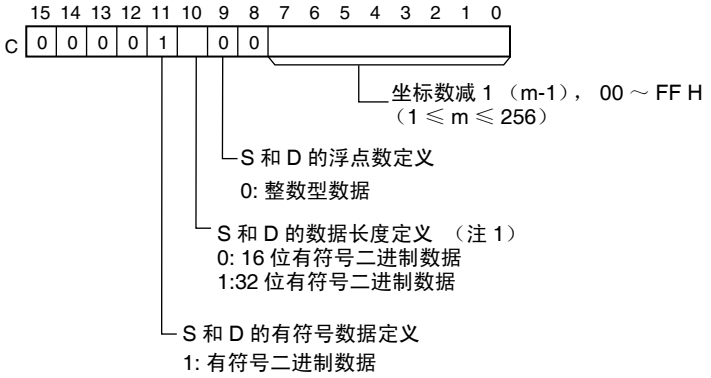
当 C 是一个字地址时，指定 APR(069) 线性归纳。

字 C 的内容指定了从 C+2 开始的数据表中的坐标数，源数据的格式，以及数据是 BCD 码还是二进制数。

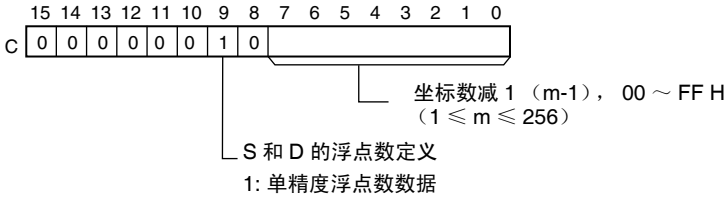
无符号整型数据 （二进制数或 BCD 码）



有符号整数型数据 （二进制数）



单精度浮点数数据



如果使用 16 位二进制数或 BCD 码，线段数据则包含在字 C+1 到 C+2m+2 中。如果使用 32 位二进制数或 BCD 码 （仅 CS1-H,CJ1-H 和 CJ1M CPU 单元），线段数据则包含在字 C+1 到 C+4m+4 中。

位 00 到 07 包含了减去 1 的线性坐标数（二进制数）m-1。位 08 到 12 没有使用。位 13 指定 f(x)=f(s) 或 f(x)=f(xm-s)：OFF 指定 f(x)=f(s)，ON 指定 f(x)=f(xm-s)。位 14 指定输出是 BCD 码还是二进制数：OFF 指定是二进制数， ON 指定

是 BCD 码 。位 15 决定输入是 BCD 码还是二进制数：OFF 指定是二进制数，ON 指定是 BCD 码。

16 位 BCD 码 16 位二进制数 （有符号或无符号）或 16 位 BCD 数据

C+1	X0 (*1)
C+2	Y0
C+3	X1
C+4	Y1
C+5	X2
C+6	Y2
	Xn
	Yn
C+ (2m+1)	Xm
C+ (2m+2)	Ym

注意：当 S 和 D 的输入输出数据包含有符号数据 （C 的 11 位 =0）时，将 Xm （表中 X 的最大值）写进字 C+1。

32 位有符号二进制数据

C+1	X0 （最右 16 位）
C+2	X0 （最左 16 位）
C+3	Y0 （最右 16 位）
C+4	Y0 （最左 16 位）
C+5	X1 （最右 16 位）
C+6	X1 （最左 16 位）
C+7	Y1 （最右 16 位）
C+8	Y1 （最左 16 位）
～	～
C+ (4n+1)	Xn （最右 16 位）
C+ (4n+2)	Xn （最左 16 位）
C+ (4n+3)	Yn （最右 16 位）
C+ (4n+4)	Yn （最左 16 位）
～	～
C+ (4m+1)	Xm （最右 16 位）
C+ (4m+2)	Xm （最左 16 位）
C+ (4m+3)	Ym （最右 16 位）
C+ (4m+4)	Ym （最左 16 位）

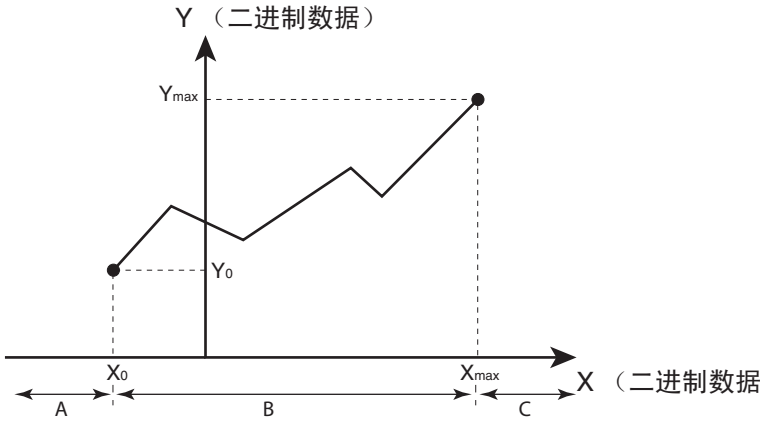
浮点数据

C+1	X0 （最右 16 位）
C+2	X0 （最左 16 位）
C+3	Y0 （最右 16 位）
C+4	Y0 （最左 16 位）
C+5	X1 （最右 16 位）
C+6	X1 （最左 16 位）
C+7	Y1 （最右 16 位）
C+8	Y1 （最左 16 位）
～	～
C+ (4n+1)	Xn （最右 16 位）
C+ (4n+2)	Xn （最左 16 位）
C+ (4n+3)	Yn （最右 16 位）
C+ (4n+4)	Yn （最左 16 位）
～	～
C+ (4m+1)	Xm （最右 16 位）
C+ (4m+2)	Xm （最左 16 位）
C+ (4m+3)	Ym （最右 16 位）
C+ (4m+4)	Ym （最左 16 位）

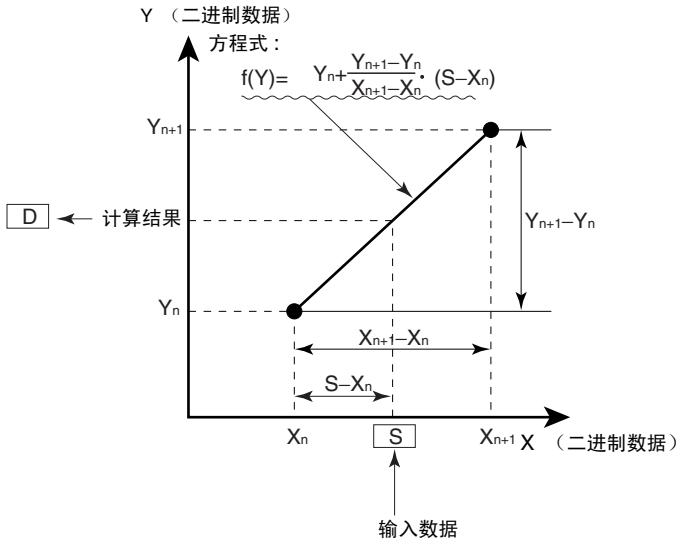
注 X 坐标必须以升序排列：X1< X2<…<Xm。不管控制字 C 指定的数据格式，都以二进制数据输入所有 (Xn， Yn) 值。

线性归纳函数的运算

APR(069) 用下列方程式来处理 S 中指定的输入数据以及表中指定的从 C+1 开始的线段数据 （Xn， Yn）。结果是由 D 决定的目标字的输出。



1. 当 S<X0
   变换后的值 =Y0
2. 当 X0 ≤ S ≤ Xmax，如果 Xn<S<Xn+1，
   变换后的值 =Yn+[{ Yn+1- Yn}/{ Xn+1- Xn }] × [输入数据 S- Xn]



3. $X_{max}<S$
   变换后的值 $=Y_{max}$

等于 256 终点值能存储在从 C+1 开始的线段数据表中。可以使用下面 5 种输入输出数据类型：

- 16 位无符号 BCD 码数据
- 16 位无符号二进制数据
- 16 位有符号二进制数据 （仅限 CS1-H/CJ1-H/CJ1M）
- 32 位有符号二进制数据 （仅限 CS1-H/CJ1-H/CJ1M）
- 单精度浮点数据 （仅限 CS1-H/CJ1-H/CJ1M）

在控制字 C 中设置数据格式

- 16 位无符号 BCD 码数据
  输入数据和 / 或输出数据可为 16 位无符号 BCD 码数据。并且，可设定线性归纳函数直接计算 S 中指定值或 Xm-S 。（Xm 是线段数据 X 的最大值）

设置名	C 中位数	设置
输入数据 （S）格式	15	0: 二进制数 1: BCD 码
输出数据 （D）格式	14	0: 二进制数 1: BCD 码
源数据形式	13	0: 计算 S 1: 计算 X_m–S
S 和 D 的有符号数据定义	11	0: 无符号数据
S 和 D 的数据长度定义	10	无效 （固定 16 位）
浮点数定义	09	0: 整数型数据

• 16 位无符号二进制数据
输入数据和 / 或输出数据可为 16 位无符号二进制数据。并且，可设定线性归纳函数直接计算 S 中指定值或 Xm-S 。（Xm 是线段数据 X 的最大值）

设置名	C 中位数	设置
输入数据 （S）格式	15	0: 二进制数 1: BCD 码
输出数据 （D）格式	14	0: 二进制数 1: BCD 码
源数据形式	13	0: 计算 S 1: 计算 X_m–S
S 和 D 的有符号数据定义	11	0: 无符号数据
S 和 D 的数据长度定义	10	无效 （固定 16 位）
浮点数定义	09	0: 整数型数据

• 16 位有符号二进制数据 （仅限 CS1-H/CJ1-H/CJ1M）

设置名	C 中位数	设置
输入数据 （S）格式	15	0: 二进制数
输出数据 （D）格式	14	0: 二进制数
源数据形式	13	0
S 和 D 的有符号数据定义	11	1: 有符号数据
S 和 D 的数据长度定义	10	0: 16 位有符号二进制数据
浮点数定义	09	0: 整数型数据

• 32 位有符号二进制数据 （仅限 CS1-H/CJ1-H/CJ1M）

设置名	C 中位数	设置
输入数据 （S）格式	15	0: 二进制数
输出数据 （D）格式	14	0: 二进制数
源数据形式	13	0
S 和 D 的有符号数据定义	11	1: 有符号数据
S 和 D 的数据长度定义	10	1: 32 位有符号二进制数据
浮点数定义	09	0: 整数型数据

注 如果 C 的第 10 位 “S 和 D 的数据长度定义” 设为 1 并且 S 的输入为一个 16 位常数，那么输入数据将在线性归纳计算之前被变换为 32 位有符号二进制数。

• 浮点数数据 （仅限 CS1-H/CJ1-H/CJ1M）

设置名	C 中位数	设置
输入数据 （S）格式	15	0: 二进制数
输出数据 （D）格式	14	0: 二进制数
源数据形式	13	0
S 和 D 的有符号数据定义	11	0
S 和 D 的数据长度定义	10	0
浮点数定义	09	1: 浮点数数据

注 如果 C 的第 9 位 “浮点数定义” 设为 1，则 S 的输入不为一个常数。

标志

名称	标志	操作
错误标志	ER	如果 C 是大于 0001 的常数，则置 ON。 如果 C 是一个字地址，但是 X 坐标不是以升序排列（X1<X2<…<Xm）则置 ON。 如果 C 是一个字地址，并且 C 的第 9、 11 和 15 位指定 BCD 输入，但是 S 不是 BCD 码，则置 ON。 如果 C 是一个字地址，并且 C 的第 9 位指定浮点数数据，但是 S 是一个字的常数，则置 ON。 如果 C 是 0000 或 0001，但是 S 不是 0000 到 0900 之间的 BCD 码，则置 ON。 其余情况下置 OFF。
等于标志	=	结果为 0000 时置 ON。 其余情况下置 OFF。
负标志	N	R 的第 15 位为 ON 则置 ON。 其余情况下置 OFF。

注意

SIN（90°）和 COS（0°）的实际结果为 1，但是 9999（0.9999）将输出到 R。

如果 C 是大于 0001 的常数，将出现错误。

如果指定线性归纳，但是 X 坐标未以升序排列（X1<X2<...<Xm），将出现错误。

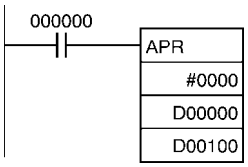
如果指定线性归纳，并且指定 BCD 输入 （C 的第 15 位为 ON），但是 S 不是 BCD 码，将出现错误。

如果指定三角函数（C=0000 或 0001）但是 S 不是 0000 ～ 0900 之间的 BCD 码，将出现错误。

例

正弦函数 (C：#0000)

下例显示了用于计算 30° 正弦值的 APR(069)。



源数据

S: D00000			
0	10^1	10^0	10^{-1}
0	3	0	0

用10^{-1}度设置数据（0000到0900，BCD）

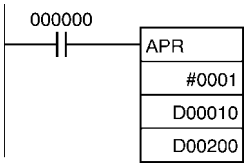
结果

R: D00100			
10^{-1}	10^{-2}	10^{-3}	10^{-4}
5	0	0	0

结果数据为4个数字，第5个及以后数字被忽略。

余弦函数 (C：#0001)

下例显示了用于计算 30° 余弦值的 APR(069)。



源数据

S: D00010			
0	10^1	10^0	10^{-1}
0	3	0	0

用10^{-1}度设置数据（0000到0900，BCD）

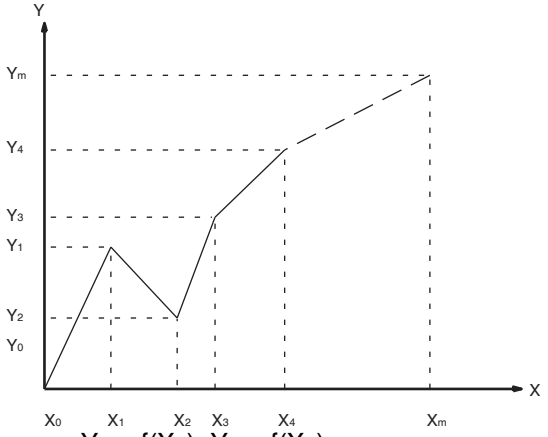
结果

R: D00200			
10^{-1}	10^{-2}	10^{-3}	10^{-4}
8	6	6	0

结果数据为4个数字，第5个及以后数字被忽略。

线性归纳 （C：字地址）
使用 16 位无符号 BCD 码或二进制数据

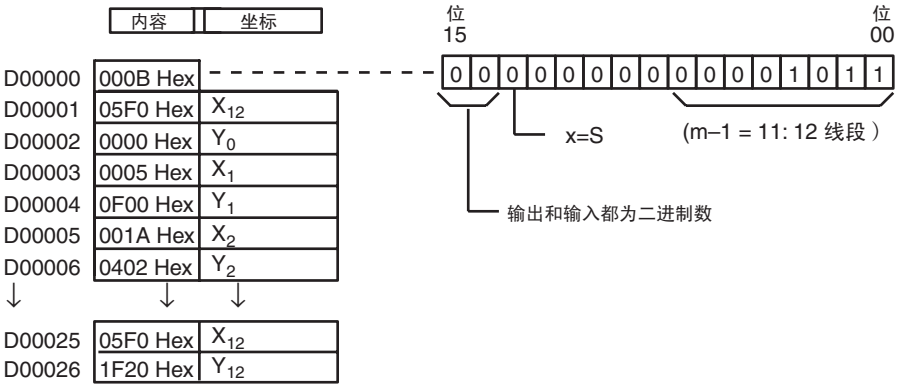
APR（069）是根据 C 中控制数据和表中指定的从 C+1 开始的线段数据来处理 S 中指定的输入数据。结果输出到 D。



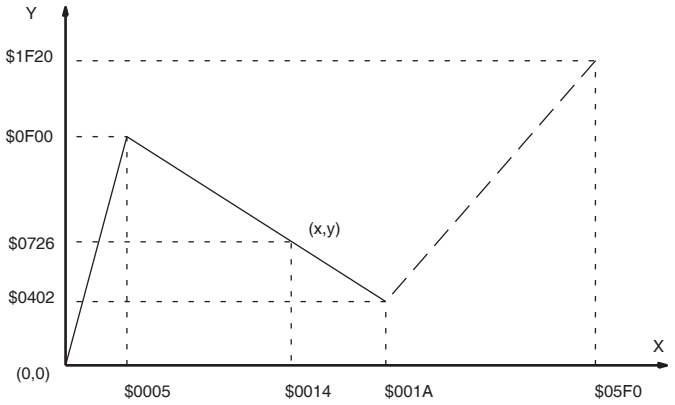
字	坐标
C+1	Xm（最大 X 值）
C+2	Y_0
C+3	X_1
C+4	Y_1
C+5	X_2
C+6	Y_2
↓	↓
C+(2m+1)	X_m （最大 X 值）
C+(2m+2)	Y_m

- $Y_n = f(X_n)$, $Y_0 = f(X_0)$
- 确认在所有情况下 Xn-1< Xn
- 以二进制数据输入所有 （Xn， Yn）值。

下例显示了怎样用 12 个坐标建立一个线性归纳。数据块根据要求是从 D00000 ～ D00026 （C 到 C+ （2 × 12+2））的连续地址。输入数据从 CIO 0010 中取，并且结果输出到 CIO 0011。



在此情况下，源字 CIO 0010 包含 0014，并且 f(0014)=0726 ，输出到 R，CIO 0011。



线性归纳计算如下所示

$$Y = 0F00 + \frac{0402 - 0F00}{001A - 0005} \times (0014 - 0015)$$

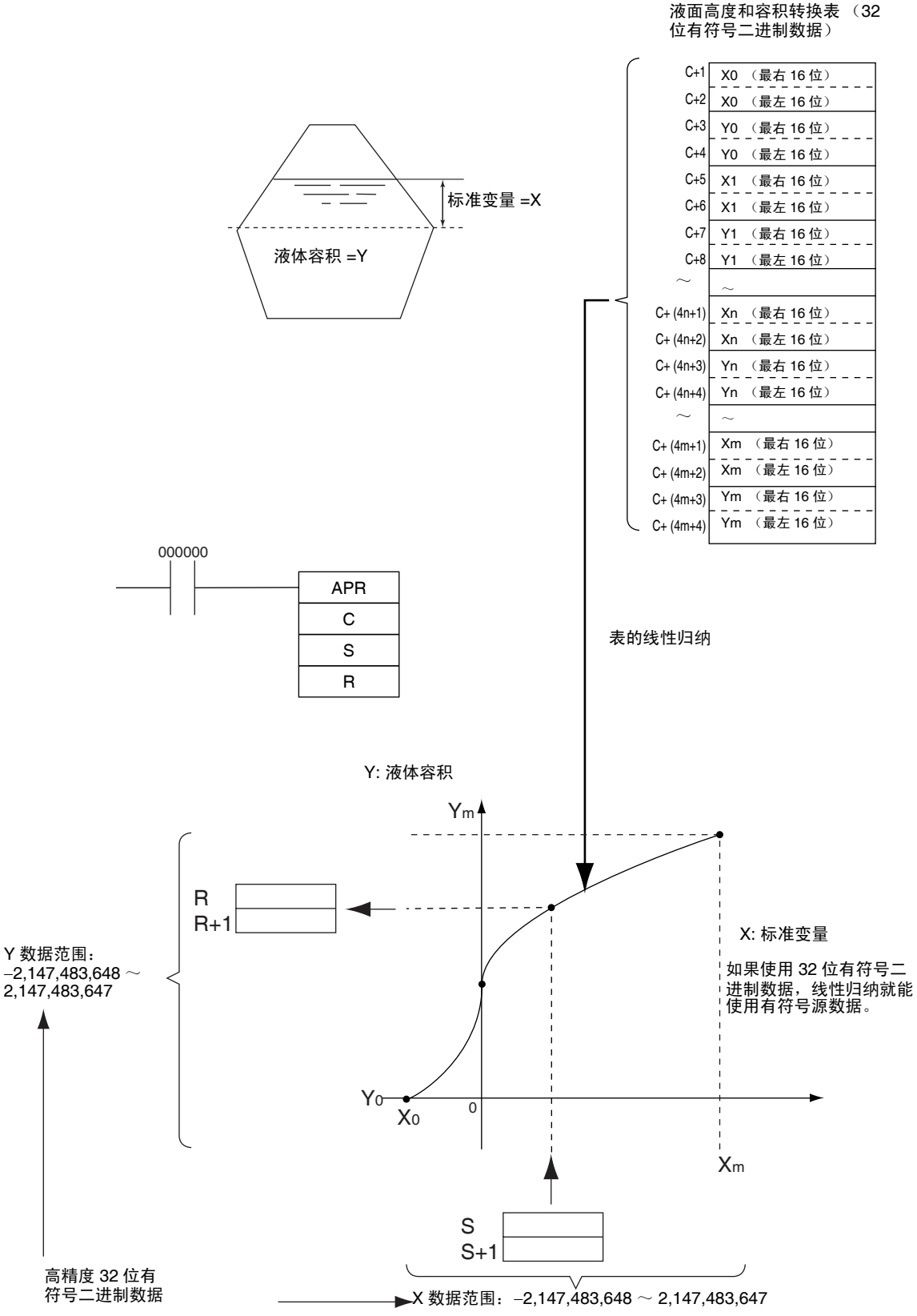
$$= 0F00 - (0086 \times 000F)$$

$$= 0726$$

数值都是16进制。

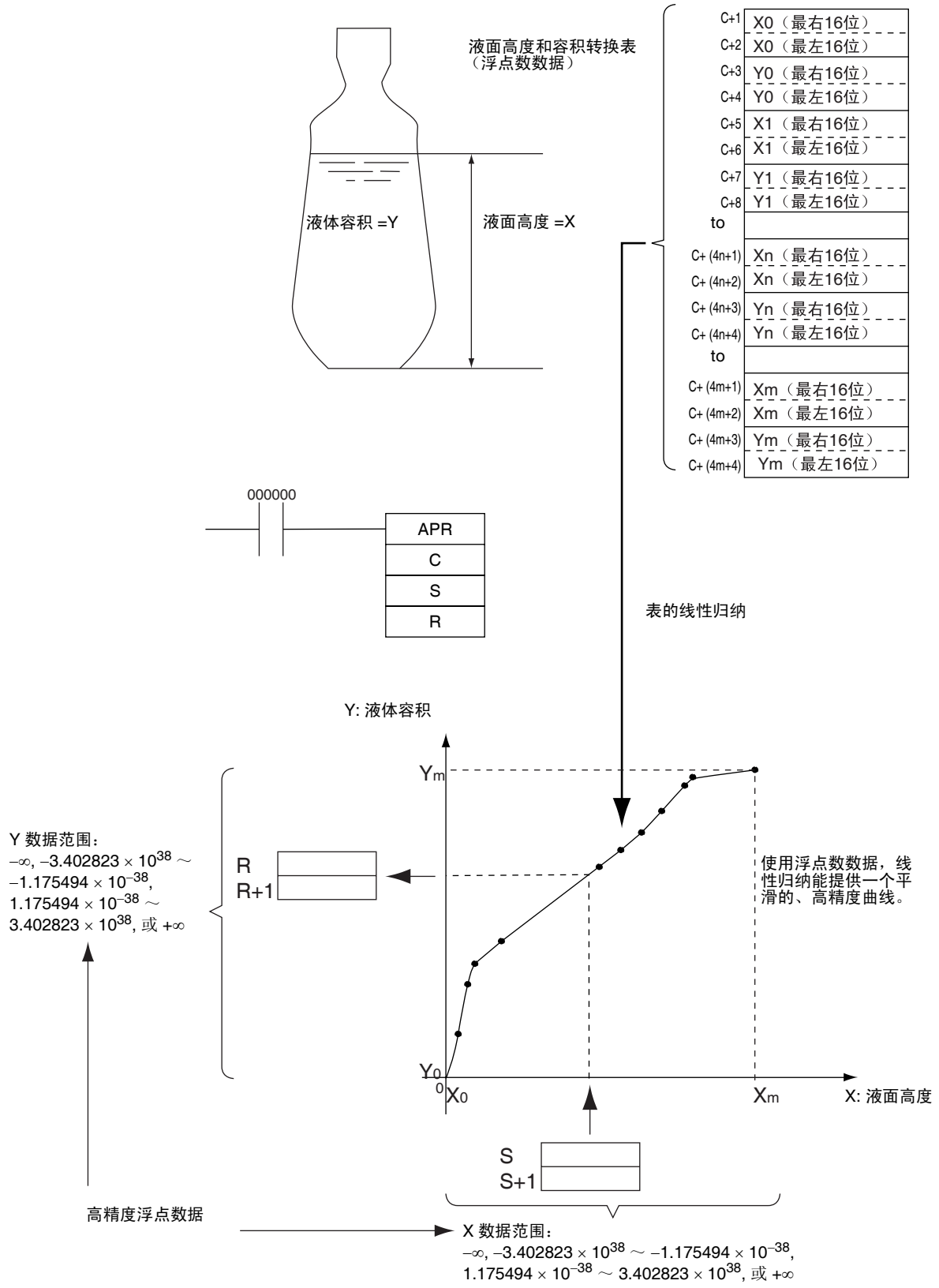
线性归纳 （C：字地址）
使用 32 位有符号二进制数据 （仅限 CS1-H/CJ1-H/CJ1M）

此例中，根据存储容器的形状使用 APR(069) 将容器中液面高度转换为液体容积。



线性归纳 （C：字地址）
使用浮点数数据 （仅限 CS1-H/CJ1-H/CJ1M）

此例中，根据存储容器的形状使用 APR （069）将容器中液面高度转换为液体容积。

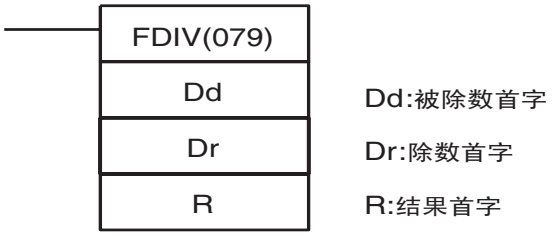


3-14-4 浮点除：FDIV(079)

用途

一个 7 位浮点数除以另一个数。浮点数用科学计数法表示 （7 位尾数和 1 位指数）。

梯形图符号



变化

变化	ON 条件时执行每个循环	FDIV(079)
	沿微分上升时执行一次	@FDIV(079)
	沿微分下降时执行一次	不支持
立即刷新功能		不支持

适用程序区

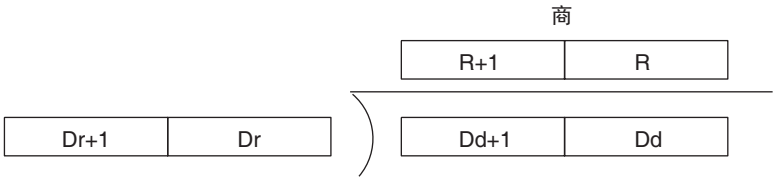
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数定义

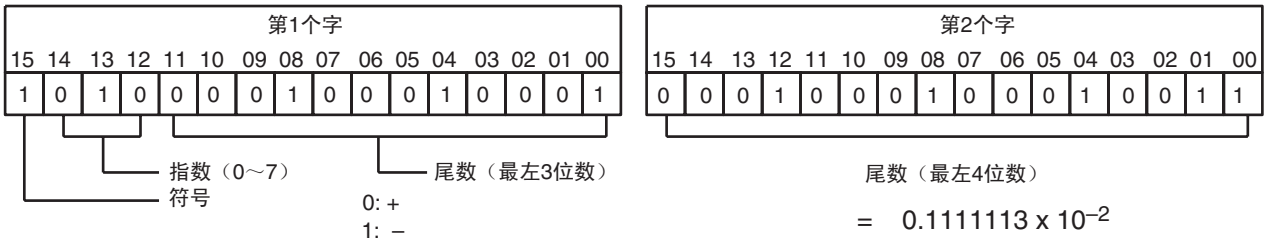
区域	Dd	Dr	R
CIO 区	CIO 0000 ～ CIO 6142		
工作区	W000 ～ W510		
保持位区	H000 ～ H510		
辅助位区	A000 ～ A958		A448 ～ A958
定时器区	T0000 ～ T4094		
计数器区	C0000 ～ C4094		
DM 区	D00000 ～ D32766		
无区号 EM 区	E00000 ～ E32766		
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---		
数据寄存器	---		
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

FDIV(079) 把 Dd 和 Dd+1 中的浮点数除以 Dr 和 Dr+1 中的数，并把结果放到 R 和 R+1。



为表示浮点数，最右的 7 位数字用作尾数，最左位数字用作指数，如下表所示。最左位数字范围从 0 ～ F；正指数范围从 0 ～ 7，负指数范围从 8 ～ F（0 ～ -7）。最右 7 位数字必须为 BCD 码。



两个浮点数例子是：

6123 4567: 0.1234567×10^{6} （6 = 0110 二进制数）

B123 4567: 0.1234567×10^{-3} （B = 1011 二进制数）

下表显示出所允许的最大和最小值

极限	8 位数字十六进制	浮点数
最大值	7999 9999	0.9999999×10^{7}
最大值 （除数和被除数）	F000 0001	0.0000001×10^{-7}
最小值 （结果）	F100 0000	0.1000000×10^{-7}

标志

名称	标记	操作
错误标志	ER	如果 Dd+1 和 Dd 中的尾数 （最右 7 位数字）不是 BCD 码则置 ON。 如果 Dr+1 和 Dr 中的尾数 （最右 7 位数字）不是 BCD 码则置 ON。 如果除数 （Dr+1 和 Dr）为 0 则置 ON。 如果结果不在 0.1000000×10^{-7} 和 0.9999999×10^{-7} 之间则置 ON。 其余情况下则置 OFF。
负标志	=	结果为 0 时置 ON。 其余情况下则置 OFF。

注意

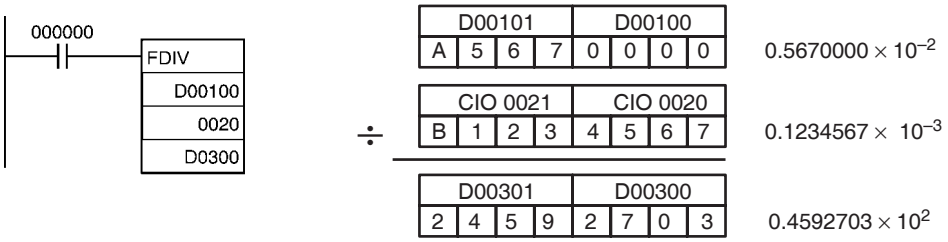
结果以浮点数表示，因此它有 7 位数字。第 8 位及以后数字被舍去。

结果必须在 0.1000000×10^{-7} 和 0.9999999×10^{-7} 之间。

例

基本浮点除法

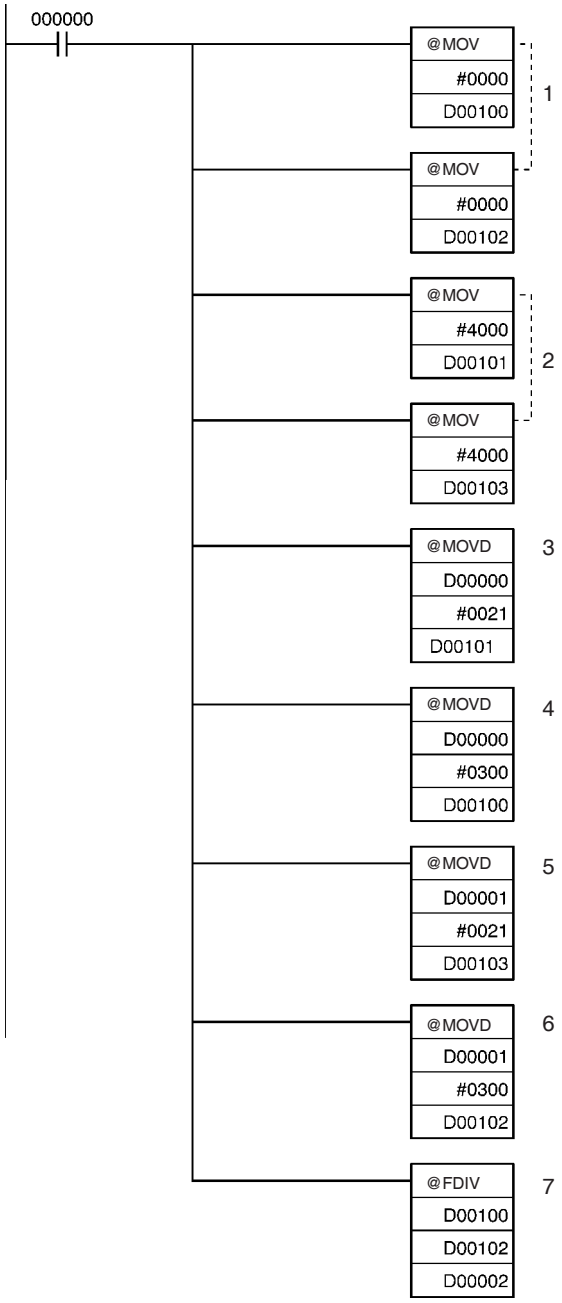
在下例中，当 CIO 000000 为 ON 时， FDIN(079) 把 D00101 和 D00100 中的浮点数除以 CIO 0021 和 CIO 0020 中的浮点数，并把结果写到 D00301 和 D00300。



两个 BCD 数的浮点除法

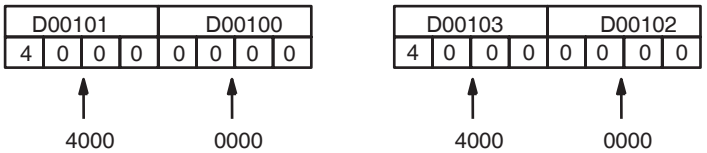
在此例中，D00000 中的 4 位数字 BCD 数除以 D00001 中的 4 位数字 BCD 数，并把浮点结果写到 D00003 和 D00002 中。

为执行浮点除， D00000 中的 BCD 数转换为 D00101 和 D00100 中的浮点格式。 D00001 中的 BCD 数转换为 D00103 和 D00102 中的浮点格式。

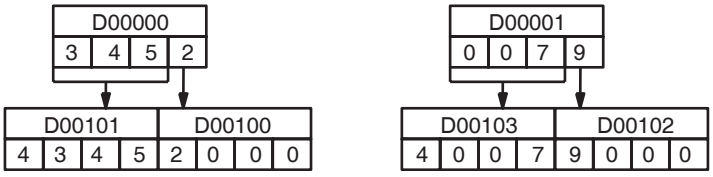


1,2,3... 1. D00100 和 D00102 设为 0000。

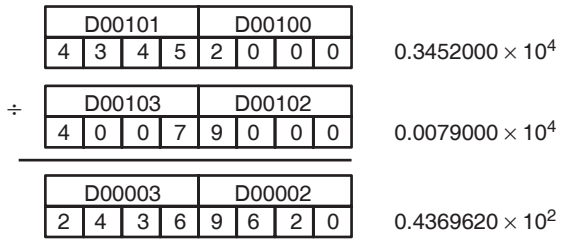
2. D00101 和 D00103 设为 4000。



3. MOVD(083) 用于把原始源字传送到 2 字浮点格式中的适当数字位。



4. FDIV(079) 把 D00101 和 D00100 中的浮点数除以 D00103 和 D00102 中的浮点数。

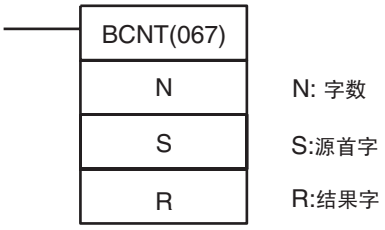


3-14-5 位计数器：BCNT(067)

用途

计算指定字中置 ON 位的总位数。

梯形图符号



变化

变化	ON 条件时执行每个循环	BCNT(067)
	沿微分上升时执行一次	@BCNT(067)
	沿微分下降时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

N：字数

字数必须在 0001 ～ FFFF （1 ～ 65535 字）之间。

S：源首字

S 和 S+(N-1) 必须在同一数据区。

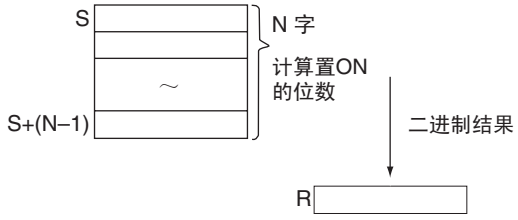
操作数定义

区域	N	S	R
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		

区域	N	S	R
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0001 ～ #FFFF（二进制）或 &1 ～ &65,535	---	
数据寄存器	DR0 ～ DR15	---	DR0 ～ DR15
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

BCNT(067) 计算 S 和 S+(N-1) 之间所有字中置 ON 的总位数，并把结果放进 R。



标志

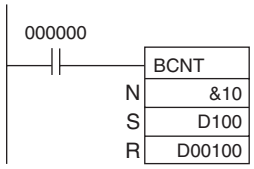
名称	标记	操作
错误标志	ER	N 为 0000 则置 ON。 如果结果超过 FFFF 则置 ON。 其余情况下则置 OFF。
等于标志	=	结果为 0000 时置 ON。 其余情况下则置 OFF。

注意

如果 N=0000 或者结果超过 FFFF，将会出现错误。

例

在下例中，当 CIO 000000 为 ON 时， BCNT(067) 计算从 CIO 0100 ～ CIO 0109 的 10 个字中置 ON 的总位数，并把结果写到 D00100。



3-15 浮点数运算指令

浮点数学指令转换数据并且执行浮点算术操作， CS/CJ 系列 CPU 单元支持下列指令。

指令	助记符	函数代码	页数
浮点到 16 位	FIX	450	519
浮点到 32 位	FIXL	451	522
16 位到浮点	FLT	452	522
32 位到浮点	FLTL	453	525
浮点加	+F	454	526
浮点减	–F	455	529
浮点乘	*F	456	530
浮点除	/F	457	532
角度到弧度	RAD	458	534
弧度到角度	DEG	459	535
正弦	SIN	460	537
余弦	COS	461	540
正切	TAN	462	541
反正弦	ASIN	463	543
反余弦	ACOS	464	545
反正切	ATAN	465	548
平方根	SQRT	466	550
指数	EXP	467	551
对数	LOG	468	554
指数幂	PWR	840	555

除上面所列指令外，CS1-H/CJ1-H CPU 单元还支持下列浮点比较和转换指令。关于双精度浮点指令的详细情况可*参考第 3-16-21 节双精度浮点输入指令*。

指令	助记符	函数代码	页数
单精度浮点符号比较指令（仅限 *CS1-H/CJ1-H/CJ1M）	LD, AND, OR + =F,<>F,<F,<=F, >F，或 >=F	329 ～ 334	556
浮点到 ASCII（仅限 *CS1-H/CJ1-H/CJ1M）	FSTR	448	560
ASCII 到浮点（仅限 *CS1-H/CJ1-H/CJ1M）	FVAL	449	565

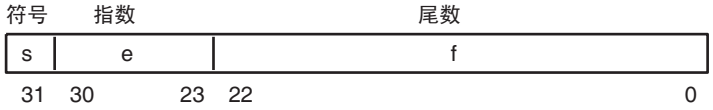
数据格式

浮点数据用符号、指数和尾数表示实数。当数据以浮点格式表示时，采用下列公式。

实数 = $(-1)^S \, 2^{e-127} \, (1.f)$

S：符号
e：指数
f：尾数

浮点数据格式符合 IEEE754 标准，数据用 32 位表示，如下所示：



数据	位数	内容
s：符号	1	0：正； 1：负
e：指数	8	指数 （e）值范围从 0 ～ 255。 实际指数是 e 减去 127 后的余数，结果从 -127 ～ 128。“e=0”和 “e=255”表示特殊数字。
f：尾数	23	二进制浮点数据尾数部分满足形式 2.0>1.f ≥ 1.0。

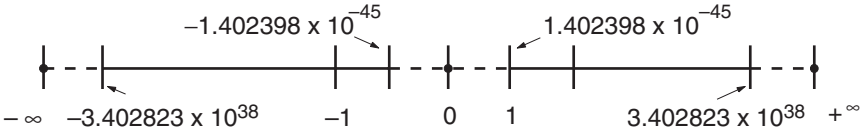
数字个数

浮点数据的有效数字个数是 24 位二进制数 （相当于 7 位数字十进制数）。

浮点数据

下列数据可以用浮点数据表示：

- $-\infty$
- $-3.402823 \times 10^{38} \leq$ 值 $\leq -1.402398 \times 10^{-45}$
- 0
- $1.402398 \times 10^{-45} \leq$ 值 $\leq 3.402823 \times 10^{38}$
- $+\infty$
- 不是一个数字 （NaN）



特殊数字

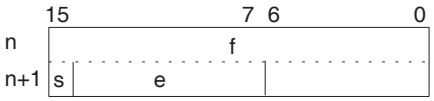
NaN，±∞和 0 的格式如下：

NaN*: e = 255, f ≠ 0
+∞: e = 255, f = 0, s= 0
–∞: e = 255, f = 0, s= 1
0: e = 0

* NaN （不是一个数字）不是一个有效的浮点数。执行浮点计算指令不会导致 NaN。

写浮点数据

当在 CX-Programmer 中把 I/O 内存编辑显示器里的数据格式指定为浮点时，显示器里输入的标准十进制数字自动转换为上面（IEEE754 格式）所示的浮点格式，并且写到 I/O 内存。当在显示器上监控时，用 IEEE754 格式写的数据自动转换为标准十进制格式。



读和写浮点数据时用户不需要知道 IEEE754 数据格式。只需记住每个浮点数占有两个字。

用浮点数表示的数字

下列类型浮点数可使用。

尾数 (f)	指数 (e)		
	0	非 0 和非全 1	全 1(255)
0	0	标准数字	无穷大
非 0	非标准数字		NaN

注 非标准数字是绝对值太小而不能表示为标准数字的数字。非标准数字具有较少的有效数字。如果计算结果是非标准数字 （包括中间结果），有效数字的个数将减少。

标准数字

标准数字表示实数。符号位为 0 表示正数， 1 表示负数。

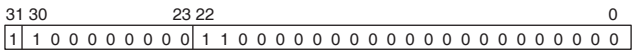
指数 (e) 将从 1 ～ 254 表示，实际指数将小于 127，即 -126 ～ 127。

尾数 (f) 从 0 ～ 2^{23}-1 表示，它假设，在实际尾数中，位 2^{23} 为 1，并且二进制小数点立即跟在它后面。

标准数字如下表示：

$(-1)^{(\text{符号 s})} \times 2^{(\text{指数 e})-127} \times$ （1+ 尾数$\times 2^{-23}$）

例



符号： –
指数： 128 – 127 = 1
尾数： $1 + (2^{22} + 2^{21}) \times 2^{-23} = 1 + (2^{-1} + 2^{-2}) = 1 + 0.75 = 1.75$
值： $-1.75 \times 2^{1} = -3.5$

非标准数字

非标准数字表示绝对值很小的实数。符号位为 0 表示正数， 1 表示负数。

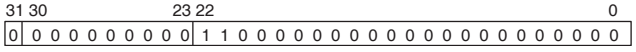
指数 （e）为 0，实际指数为 -126。

尾数 （f）从 0 ～ 2^{23}-1 表示，它假设，在实际尾数中，位 2^{23} 为 0，并且二进制小数点立即跟在它后面。

非标准数字如下表示：

$(-1)^{(\text{符号 s})} \times 2^{-126} \times$ （尾数$\times 2^{-23}$）

例



符号： –
指数： –126
尾数： $0 + (2^{22} + 2^{21}) \times 2^{-23} = 0 + (2^{-1} + 2^{-2}) = 0 + 0.75 = 0.75$
值： -0.75×2^{-126}

零

+0.0 和 -0.0 值可通过设置符号来表示， 0 表示正数， 1 表示负数。指数和尾数都为 0。 +0.0 和 -0.0 都等于 0.0。由 0.0 的符号产生的差别，参考下面的浮点算术结果。

无穷大

+ ∞和 - ∞值可通过设置符号来表示，0 表示正数，1 表示负数。指数为 255(2^{8}-1)，尾数为 0。

NaN

NaN （不是一个数字）在计算如 0.0/0.0，∞ / ∞，或∞ - ∞结果不等于一个数字或无穷大时产生。指数为 255(2^8-1)，尾数为非 0。

注 没有 NaN 符号或尾数区域值 （不是非 0）的定义。

浮点算术结果

四舍五入结果

当浮点算术精确结果中的数字个数超过内部处理表达式中的有效数字，则下面的方法将用于结果的四舍五入。

如果结果近似于两个内部浮点表达式中的一个，使用近似表达式。如果结果介于两个内部浮点表达式的中间，结果将四舍五入，以使尾数的最后一个数字为 0。

上溢出，下溢出和非法计算

上溢出将根据结果符号以正无穷或负无穷输出。下溢出将根据结果符号以正 0 或负 0 输出。

非法计算将导致 NaN，非法计算包括给一个数字加上带相反符号的无穷大，从一个数字减去带相反符号的无穷大，0 和无穷大相乘，0 和 0 相除，或无穷大与无穷大相除。

如果在浮点数转换为整数时出现上溢出，则结果值将不正确。

处理特殊值时的注意事项

下列注意事项适用于处理 0，无穷大和 NaN。

- 正 0 和负 0 的和为正 0。
- 相同符号的 0 之差为正 0。
- 如果任何操作数是 NaN，结果将为 NaN。
- 正 0 和负 0 在比较时以相等处理。
- 对一个或更多的 NaN 进行比较或相等测试时，！ = 总为真，其它指令总为假。

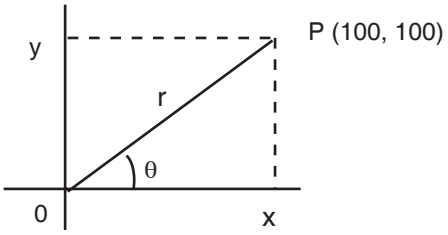
浮点计算结果

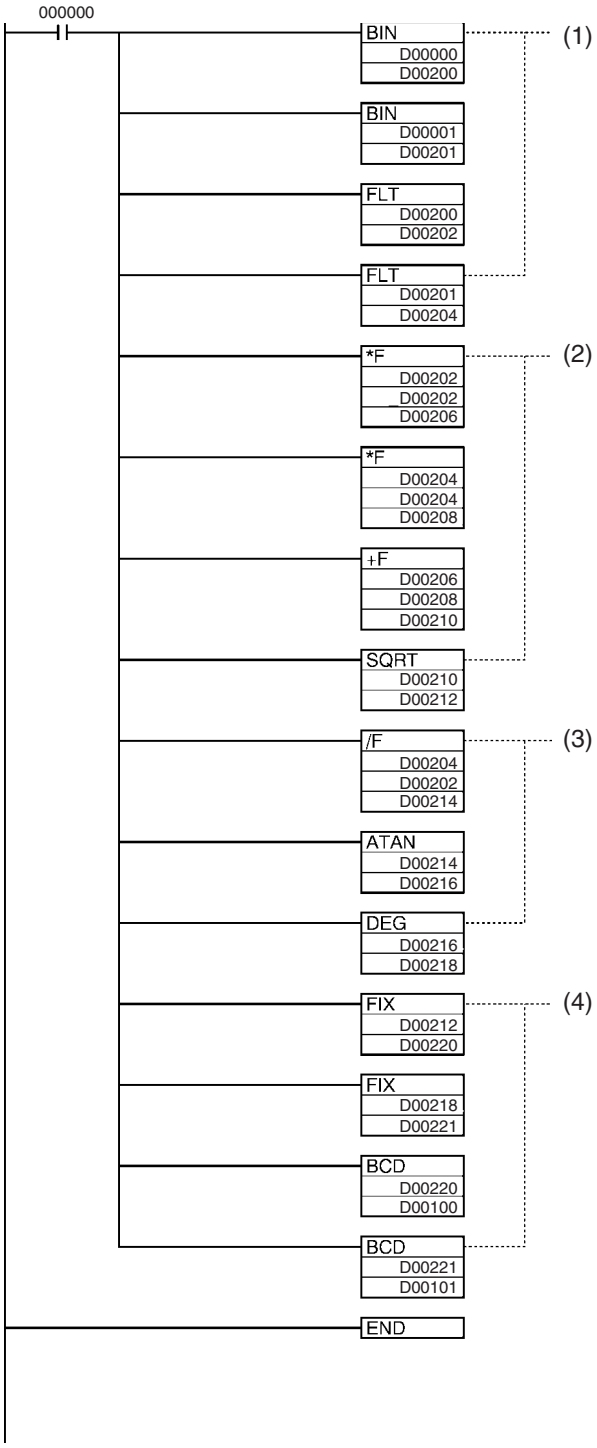
当结果的绝对值大于浮点数据能够表达的最大值，上溢出标志将变为 ON，结果输出±∞。如果结果为正，输出为 + ∞；如果为负，输出为 - ∞。

计算后指数 (e) 和尾数 (f) 都为 0 时，等于标志变 ON，当结果的绝对值小于浮点数据能够表达的最小值，计算结果也输出 0，这时下溢出标志变 ON。

例

在此程序例中，X 轴和 Y 轴坐标 (X，Y) 由 D00000 和 D00001 的 4 位 BCD 数的内容表示。从原点的距离 (r) 和角度（θ，用角度）算出，并输出到 D00100 和 D00101。结果中，小数点后面的数被截去。


y
P (100, 100)
r
θ
0
x


000000
BIN
D00000
D00200
(1)
BIN
D00001
D00201
FLT
D00200
D00202
FLT
D00201
D00204
*F
D00202
D00202
D00206
(2)
*F
D00204
D00204
D00208
+F
D00206
D00208
D00210
SQRT
D00210
D00212
/F
D00204
D00202
D00214
(3)
ATAN
D00214
D00216
DEG
D00216
D00218
FIX
D00212
D00220
(4)
FIX
D00218
D00221
BCD
D00220
D00100
BCD
D00221
D00101
END

计算公式

距离r= $\sqrt{x^2 + y^2}$

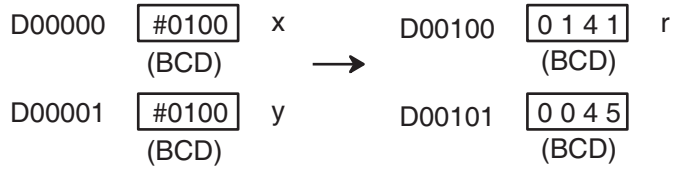
角度 $= \tan^{-1}\left(\frac{y}{x}\right)$

例

距离r=$\sqrt{100^2 + 100^2} = 141.4214$

角度 $= \tan^{-1}\left(\frac{100}{100}\right) = 45.0$

DM 内容



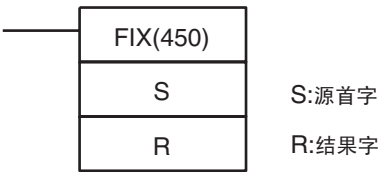
1. 这部分程序把数据从 BCD 码转换成浮点数。
   a) 从 D00200 向上的数据区用作工作区。
   b) 第 1 个 BIN(023) 用于暂时把 BCD 数据转换成二进制数据，而 FLT(452) 用于把二进制数据转换成浮点数据。
   c) 已经转换成浮点数据的 X 值输出到 D00203 和 D00202。
   d) 已经转换成浮点数据的 Y 值输出到 D00205 和 D00204。
2. 为了找到距离 r，浮点数学指令用于计算 X^2+Y^2 的平方根。结果作为浮点数据输出到 D00213 和 D00212。
3. 为了找到角度 θ，浮点数学指令用于计算 $\tan^{-1}(Y/X)$。ATAN(465) 输出结果以弧度表示，因此 DEG(459) 用于转换为度数。结果作为浮点数据输出到 D00219 和 D00218。
4. 数据从浮点数转换回 BCD 数。
   a) 第 1 个 FIX(450) 用于暂时把浮点数据转换成二进制数据，而 BCD(024) 用于把二进制数据转换成 BCD 数据。
   b) 距离 r 输出到 D00100 。
   c) 角度 θ 输出到 D00101。

3-15-1 浮点到 16 位：FIX(450)

用途

把32位浮点值转换为16位有符号二进制数据，并把结果放在指定的结果字中。

梯形图符号



变化

变化	ON 条件时执行每个循环	FIX(450)
	沿微分上升时执行一次	@FIX(450)
	沿微分下降时执行一次	不支持
立即刷新功能		不支持

适用程序区

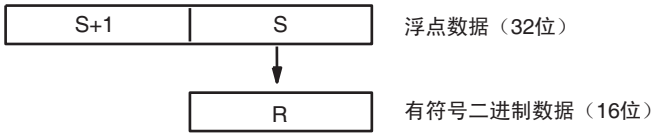
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数定义

区域	S	R
CIO 区	CIO 0000 ～ CIO 6142	CIO 0000 ～ CIO 6143
工作区	W000 ～ W510	W000 ～ W511
保持位区	H000 ～ H510	H000 ～ H511
辅助位区	A000 ～ A958	A448 ～ A959
定时器区	T0000 ～ T4094	T0000 ～ T4095
计数器区	C0000 ～ C4094	C0000 ～ C4095
DM 区	D00000 ～ D32766	D00000 ～ D32767
无区号 EM 区	E00000 ～ E32766	E00000 ～ E32767
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#00000000 ～ #FFFFFFFF （二进制）	---
数据寄存器	---	DR0 ～ DR15
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

说明

FIX(450) 把 S+1 和 S 中的 32 位浮点数 （IEEE754 格式）的整数部分转换为 16 位有符号二进制数据，并把结果放到 R。



只转换了浮点数据的整数部分，小数部分被截去。浮点数据的整数部分必须在 -32768 ～ 32767 之间。

转换例：
浮点值 3.5 转换为 3。
浮点值 -3.5 转换为 -3。

标志

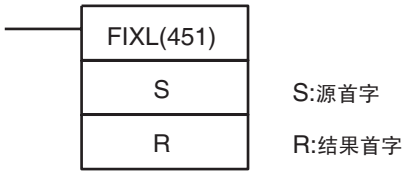
名称	标记	操作
错误标志	ER	如果 S+1 和 S 中的数据不是一个数字 （NaN）则置 ON。 如果 S+1 和 S 的整数部分不在 -32768 到 32767 范围内 F 则置 ON。 其余情况下置 OFF。
等于标志	=	结果为 0000 时置 ON。 其余情况下置 OFF。
负标志	N	结果的第 15 位为 ON 时置 ON。 其余情况下置 OFF。

注意 S+1和S的内容必须为浮点数据，并且整数部分必须在-32768到32767范围内。

3-15-2 浮点到 32 位：FIXL(451)

用途 把32位浮点值转换为32位有符号二进制数据，并把结果放在指定的结果字中。

梯形图符号



变化

变化	ON 条件时每个循环执行	FIXL(451)
	上升沿微分时执行一次	@FIXL(451)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

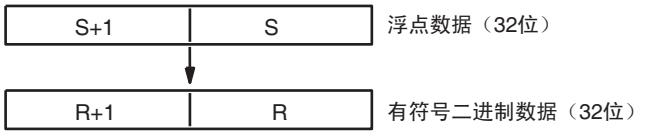
操作数规定

区域	S	R
CIO 区	CIO 0000 ～ CIO 6142	
工作区	W000 ～ W510	
保持位区	H000 ～ H510	
辅助位区	A000 ～ A958	A448 ～ A958
定时器区	T0000 ～ T4094	
计数器区	C0000 ～ C4094	
DM 区	D00000 ～ D32766	
无区号 EM 区	E00000 ～ E32766	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	

区域	S	R
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#00000000 ～ #FFFFFFFF （二进制）	---
数据寄存器	---	
索引寄存器	---	
使用索引寄存器 的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –()IR15	

说明

FIXL(451) 把 S+1 和 S 中的 32 位浮点数 （IEEE754 格式）的整数部分转换为 32 位有符号二进制数据，并把结果放进 R+1 和 R 中。



仅转换浮点数据的整数部分，小数部分被截去，（浮点数据的整数部分必须在 -2147483648 ～ 2147483647 之间）。

转换例：
浮点数 2147483640.5 转换为 2147483640。

浮点数 -2147483640.5 转换为 -2147483640。

标志

名称	标记	操作
错误标志	ER	如果 S+1 和 S 中的数据不是一个数字 （NaN）则置 ON。 如果 S+1 和 S 的整数部分不在 -2147483648 到 2147483647 之间则置 ON。 其余情况下置 OFF。
等于标志	=	如果结果为 00000000 置 ON。 其余情况下置 OFF。
负标志	N	如果执行后 R+1 的第 15 位为 ON 则置 ON。 其余情况下置 OFF。

注意

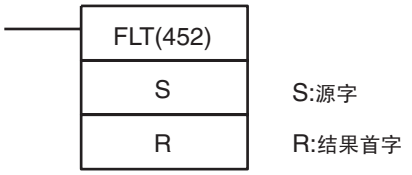
S+1和S的内容必须为浮点数据，并且整数部分必须在-2147483648～2147483647之间。

3-15-3 16 位到浮点：FLT(452)

用途

把16位有符号二进制数据转换为32位浮点数，并把结果放在指定的结果字中。

梯形图符号



变化

变化	ON 条件时每个循环执行	FLT(452)
	上升沿微分时执行一次	@FLT(452)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

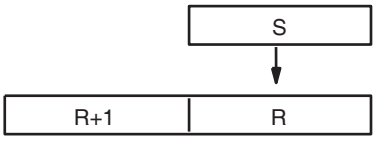
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数定义

区域	S	R
CIO 区	CIO 0000 ～ CIO 6143	CIO 0000 ～ CIO 6142
工作区	W000 ～ W511	W000 ～ W510
保持位区	H000 ～ H511	H000 ～ H510
辅助位区	A000 ～ A959	A448 ～ A958
定时器区	T0000 ～ T4095	T0000 ～ T4094
计数器区	C0000 ～ C4095	C0000 ～ C4094
DM 区	D00000 ～ D32767	D00000 ～ D32766
无区号 EM 区	E00000 ～ E32767	E00000 ～ E32766
有区号 EM 区	En_00000 ～ En_32767 (n= 0 ～ C)	En_00000 ～ En_32766 (n= 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#0000 ～ #FFFF （二进制）	---
数据寄存器	DR0 ～ DR15	---
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

说明

FLT(452) 把 S 中的 16 位有符号二进制值转换为 32 位浮点数（IEEE754 格式），并把结果放进 R+1 和 R 中。浮点结果的小数点后加了一个 0 。



S 只能指定在 -32768 ～ 32767 范围内的数。为转换超出范围的有符号二进制数，使用 FLTL(453)。

转换例：
有符号二进制值 3 转换为 3.0 。
有符号二进制值 -3 转换为 -3.0 。

标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	如果结果为指数和尾数都为 0 则置 ON。 其余情况下置 OFF。
负标志	N	结果为负时置 ON。 其余情况下置 OFF。

注意

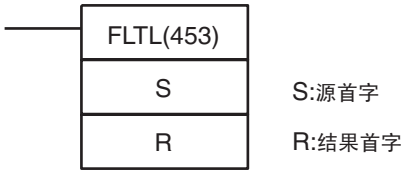
S 的内容必须是包含在范围 （十进制）值 -32768 ～ 32767 之间的有符号二进制数据。

3-15-4 32 位到浮点：FLTL(453)

用途

把32位有符号二进制数据转换为32位浮点数，并把结果放在指定的结果字中。

梯形图符号



变化

变化	ON 条件时每个循环执行	FLTL(453)
	上升沿微分时执行一次	@FLTL(453)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

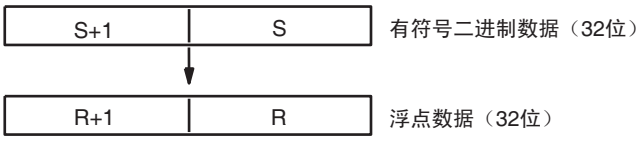
操作数定义

区域	S	R
CIO 区	CIO 0000 ～ CIO 6142	
工作区	W000 ～ W510	
保持位区	H000 ～ H510	
辅助位区	A000 ～ A958	A448 ～ A958
定时器区	T0000 ～ T4094	
计数器区	C0000 ～ C4094	
DM 区	D00000 ～ D32766	
无区号 EM 区	E00000 ～ E32766	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	

区域	S	R
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#00000000 ～ #FFFFFFFF （二进制）	---
数据寄存器	---	
索引寄存器	---	
使用索引寄存器 的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

说明

FLTL(453) 把 S+1 和 S 中的 32 位有符号二进制值转换为 32 位浮点数（IEEE754 格式），并把结果放进 R+1 和 R 中。浮点结果中小数点后加了一个 0。



在 -2147483648 到 2147483647 范围内的有符号二进制数据，可指定在 S+1 和 S 中。浮点数有 24 个有效的二进制数字（位）。如果一个大于 16777215（可用 24 位表示的最大值）的数用 FLTL(453) 转换，结果将不准确。

转换例：

有符号二进制数 16777215 转换为 16777215.0。

有符号二进制值 -16777215 转换为 -15777215.0。

标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	如果结果为指数和尾数都为 0 则置 ON。 其余情况下置 OFF。
负标志	N	结果为负时置 ON。 其余情况下置 OFF。

注意

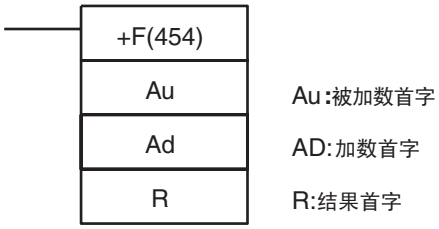
如果一个绝对值大于 16777215（用 24 位表示的最大值）的数被转换，则结果不准确。

3-15-5 浮点加：+F(454)

用途

两个 32 位浮点数相加并把结果放在指定的结果字中。

梯形图符号



变化

变化	ON 条件时每个循环执行	+F(454)
	上升沿微分时执行一次	@+F(454)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

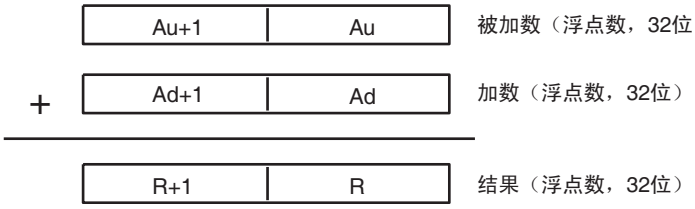
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数定义

区域	Au	Ad	R
CIO 区	CIO 0000 ～ CIO 6142		
工作区	W000 ～ W510		
保持位区	H000 ～ H510		
辅助位区	A000 ～ A958		A448 ～ A958
定时器区	T0000 ～ T4094		
计数器区	C0000 ～ C4094		
DM 区	D00000 ～ D32766		
无区号 EM 区	E00000 ～ E32766		
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#00000000 ～ #FFFFFFFF （二进制）		---
数据寄存器	---		
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

+F(454) 把 Ad+1 和 Ad 中的 32 位浮点数加到 Au+1 和 Au 中的 32 位浮点数中，并把结果放进 R+1 和 R 中。（浮点数必须是 IEEE754 格式）。



如果结果的绝对值大于可用浮点数表示的最大值，上溢出标志将置 ON，并且结果将以±∞输出。

如果结果的绝对值小于可用浮点数表示的最小值，下溢出标志将置 ON，并且结果将以 0 输出。

被加数和加数的各种组合将产生下表所示的结果。

	被加数				
加数	0	数字	+∞	−∞	NaN
0	0	数字	+∞	−∞	
数字	数字	见注 1	+∞	−∞	
+∞	+∞	+∞	+∞	见注 2	
−∞	−∞	−∞	见注 2	−∞	
NaN					见注 2

注 1. 结果可能为 0 （包括下溢出），一个数字， + ∞或 - ∞。

2. 错误标志将置 ON，并且指令不会被执行。

标志

名称	标记	操作
错误标志	ER	如果被加数或加数不是浮点数则置 ON。 如果被加数或加数不是一个数字 （NaN）则置 ON。 + ∞和 - ∞相加时置 ON。 其余情况下置 OFF。
等于标志	=	结果的指数和尾数都为 0 则置 ON。 其余情况下置 OFF。
上溢出标志	OF	如果结果的绝对值太大而不能表示为 32 位浮点数，则置 ON。
下溢出标志	UF	如果结果的绝对值太小而不能表示为 32 位浮点数，则置 ON。
负标志	N	结果为负时置 ON。 其余情况下置 OFF。

注意

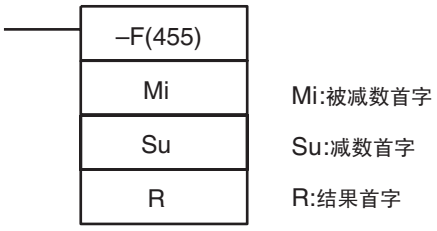
被加数 （Au+1 和 Au）和加数 （Ad+1 和 Ad）必须是 IEEE754 浮点数格式。

3-15-6 浮点减：-F(455)

用途

从一个32位浮点数中减去另一个32位浮点数，并把结果放在指定的结果字中。

梯形图符号



变化

变化	ON 条件时每个循环执行	–F(455)
	上升沿微分时执行一次	@–F(455)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数规定

区域	Mi	Su	R
CIO 区	CIO 0000 ～ CIO 6142		
工作区	W000 ～ W510		
保持位区	H000 ～ H510		
辅助位区	A000 ～ A958		A448 ～ A958
定时器区	T0000 ～ T4094		
计数器区	C0000 ～ C4094		
DM 区	D00000 ～ D32766		
无区号 EM 区	E00000 ～ E32766		
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#00000000 ～ #FFFFFFFF （二进制）		
数据寄存器	---		
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

-F(455) 从 Mi+1 和 Mi 的 32 位浮点数中减去 Su+1 和 Su 的 32 位浮点数，并把结果放在 R+1 和 R 中。（浮点数据必须是 IEEE754 格式）。

如果结果的绝对值大于可用浮点数表示的最大值，上溢出标志将置 ON，并且结果将以±∞输出。

如果结果的绝对值小于可用浮点数表示的最小值，下溢出标志将置 ON，并且结果将以 0 输出。

被减数和减数的各种组合将产生下表所示的结果。

	被减数				
减数	0	**数字**	+∞	–∞	NaN
0	0	数字	+∞	–∞	
数字	数字	见注 1	+∞	–∞	
+∞	–∞	–∞	见注 2	–∞	
–∞	+∞	+∞	+∞	见注 2	
NaN					见注 2

注 1. 结果可能为 0（包括下溢出），一个数字，+ ∞，或 - ∞。

2. 错误标志将置 ON，并且指令不会被执行。

标志

名称	标记	操作
错误标志	ER	如果被减数或减数不是浮点数则置 ON。 如果被减数或减数不是一个数字 （NaN）则置 ON。 + ∞减去 - ∞时置 ON。 - ∞减去 - ∞时置 ON。 其余情况下置 OFF。
等于标志	=	结果的指数和尾数都为 0 则置 ON。 其余情况下置 OFF。
上溢出标志	OF	如果结果的绝对值太大，而不能表示为 32 位浮点数，则置 ON。
下溢出标志	UF	如果结果的绝对值太小，而不能表示为 32 位浮点数，则置 ON。
负标志	N	结果为负时置 ON。 其余情况下置 OFF。

注意

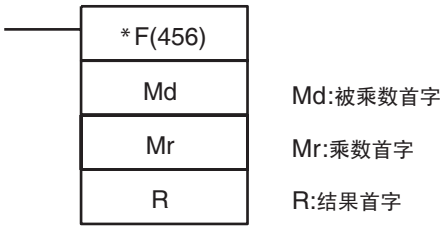
被减数 （Mi+1 和 Mi）和减数 （Su+1 和 Su）必须是 IEEE754 浮点数格式。

3-15-7 浮点乘：*F(456)

用途

两个 32 位浮点数相乘，并把结果放在指定的结果字中。

梯形图符号



变化

变化	ON 条件时每个循环执行	*F(456)
	上升沿微分时执行一次	@*F(456)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数定义

区域	Md	Mr	R
CIO 区	CIO 0000 ～ CIO 6142		
工作区	W000 ～ W510		
保持位区	H000 ～ H510		
辅助位区	A000 ～ A958		A448 ～ A958
定时器区	T0000 ～ T4094		
计数器区	C0000 ～ C4094		
DM 区	D00000 ～ D32766		
无区号 EM 区	E00000 ～ E32766		
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#00000000 ～ #FFFFFFFF （二进制）		
数据寄存器	---		
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

＊ F(456) 将 Md+1 和 Md 中的 32 位浮点数与 Mr+1 和 Mr 中的 32 位浮点数相乘，并把结果放进 R+1 和 R 中。（浮点数必须是 IEEE754 格式）

如果结果的绝对值大于可用浮点数表示的最大值，上溢出标志将置 ON，并且结果将以±∞输出。

如果结果的绝对值小于可用浮点数表示的最小值，下溢出标志将置 ON，并且结果将以 0 输出。

被乘数和乘数的各种组合将产生下表所示的结果。

	被乘数				
乘数	0	数字	+∞	–∞	NaN
0	0	0	见注 2	见注 2	
数字	0	见注 1	+/–∞	+/–∞	
+∞	见注 2	+/–∞	+∞	–∞	
–∞	见注 2	+/–∞	–∞	+∞	
NaN					见注 2

注 1. 结果可能为 0（包括下溢出），一个数字，+∞，或 -∞。

2. 错误标志将置 ON，并且指令不会被执行。

标志

名称	标记	操作
错误标志	ER	如果被乘数或乘数不是浮点数则置 ON。 如果被乘数或乘数不是一个数字 （NaN）则置 ON。 +∞和 0 相乘时置 ON。 -∞和 0 相乘时置 ON。 其余情况下置 OFF。
等于标志	=	如果结果的指数和尾数都为 0 则置 ON。 其余情况下置 OFF。
上溢出标志	OF	如果结果的绝对值太大，而不能表示为 32 位浮点值，则置 ON。
下溢出标志	UF	如果结果的绝对值太小，而不能表示为 32 位浮点值，则置 ON。
负标志	N	结果为负时置 ON。 其余情况下置 OFF。

注意

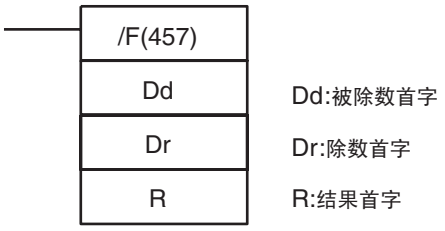
被乘数 （Md+1 和 Md）和乘数 （Mr+1 和 Mr）必须是 IEEE754 浮点数格式。

3-15-8 浮点除：/F(457)

用途

32 位浮点数除以另一个 32 位浮点数，并把结果放在指定的结果字中。

梯形图符号



变化

变化	ON 条件时每个循环执行	/F(457)
	上升沿微分时执行一次	@/F(457)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

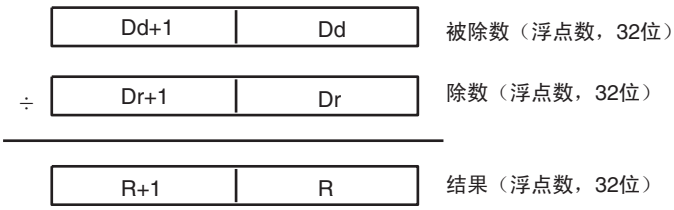
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数定义

区域	Dd	Dr	R
CIO 区	CIO 0000 ～ CIO 6142		
工作区	W000 ～ W510		
保持位区	H000 ～ H510		
辅助位区	A000 ～ A958		A448 ～ A958
定时器区	T0000 ～ T4094		
计数器区	C0000 ～ C4094		
DM 区	D00000 ～ D32766		
无区号 EM 区	E00000 ～ E32766		
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#00000000 ～ #FFFFFFFF （二进制）		---
数据寄存器	---		
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

/F(457) 把 Dd+1 和 Dd 中的 32 位浮点数除以 Dr+1 和 Dr 中的 32 位浮点数，并放进 R+1 和 R 中。（浮点数必须是 IEEE754 格式）



如果结果的绝对值大于可用浮点数表示的最大值，上溢出标志将置 ON，并且结果将以±∞输出。

如果结果的绝对值小于可用浮点数表示的最小值，下溢出标志将置 ON，并且结果将以 0 输出。

被除数和除数的各种组合将产生下表所示的结果。

	被除数				
除数	0	数字	+∞	–∞	NaN
0	见注 3	+/–∞	+∞	–∞	
数字	0	见注 1	+/–∞	+/–∞	
+∞	0	见注 2	见注 3	见注 3	
–∞	0	见注 2	见注 3	见注 3	
NaN					见注 3

注 1. 结果可能为 0（包括下溢出），一个数字，+ ∞或 - ∞。

2. 对下溢出结果将为 0。

3. 错误标志将置 ON，并且指令不会被执行。

标志

名称	标记	操作
错误标志	ER	如果被除数或除数不是浮点数则置 ON。 如果被除数或除数不是一个数字（NaN）则置 ON。 被除数和除数都为 0 时置 ON。 被除数和除数都为 + ∞ /- ∞时置 ON。 其余情况下置 OFF。
等于标志	=	如果结果的指数和尾数都为 0 则置 ON。 其余情况下置 OFF。
上溢出标志	OF	如果结果的绝对值太大，而不能表示为 32 位浮点值，则置 ON。
下溢出标志	UF	如果结果的绝对值太小，而不能表示为 32 位浮点值，则置 ON。
负标志	N	结果为负时置 ON。 其余情况下置 OFF。

注意

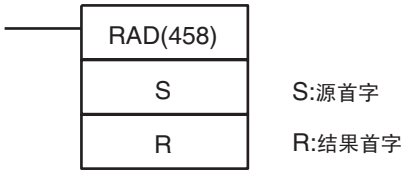
被除数（Dd+1 和 Dd）和除数（Dr+1 和 Dr）必须是 IEEE754 浮点数格式。

3-15-9 角度到弧度：RAD(458)

用途

把 32 位浮点数从角度转换为弧度，并把结果放在指定的结果字中。

梯形图符号



变化

变化	ON 条件时每个循环执行	RAD(458)
	上升沿微分时执行一次	@RAD(458)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

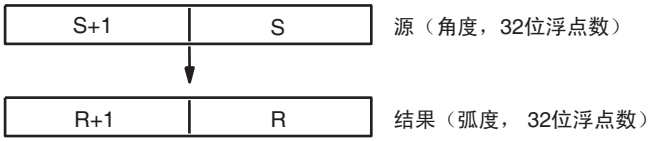
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数规定

区域	S	R
CIO 区	CIO 0000 ～ CIO 6142	
工作区	W000 ～ W510	
保持位区	H000 ～ H510	
辅助位区	A000 ～ A958	A448 ～ A958
定时器区	T0000 ～ T4094	
计数器区	C0000 ～ C4094	
DM 区	D00000 ～ D32766	
无区号 EM 区	E00000 ～ E32766	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#00000000 ～ #FFFFFFFF （二进制）	---
数据寄存器	---	
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

说明

RAD(458) 把 S+1 和 S 中的 32 位浮点数从角度转换为弧度，并把结果放在 R+1 和 R 中。（源浮点数必须表示为 IEEE754 格式）



用下列公式把角度转换为弧度：

角度×π /180= 弧度

如果结果的绝对值大于可用浮点数表示的最大值，上溢出标志将置 ON，并且结果将以±∞输出。

如果结果的绝对值小于可用浮点数表示的最小值，下溢出标志将置 ON，并且结果将以 0 输出。

标志

名称	标记	操作
错误标志	ER	如果源数据不是浮点数则置 ON。 如果源数据不是一个数 （NaN），则置 ON。 其余情况下置 OFF。
等于标志	=	如果结果的指数和尾数都为 0 则置 ON。 其余情况下置 OFF。
上溢出标志	OF	如果结果的绝对值太大，而不能表示为 32 位浮点值，则置 ON。
下溢出标志	UF	如果结果的绝对值太小，而不能表示为 32 位浮点值，则置 ON。
负标志	N	结果为负时置 ON。 其余情况下置 OFF。

注意

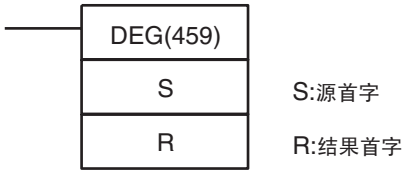
S+1 和 S 中的源数据必须是 IEEE754 浮点数格式。

3-15-10 弧度到角度：DEG(459)

用途

把 32 位浮点数从弧度转换为角度，并把结果放在指定的结果字中。

梯形图符号



变化

变化	ON 条件时每个循环执行	DEG(459)
	上升沿微分时执行一次	@DEG(459)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

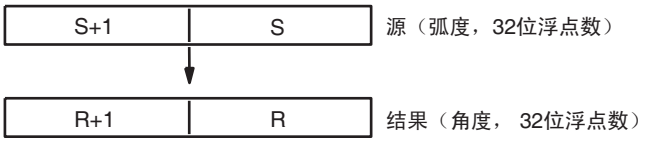
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数规定

区域	S	R
CIO 区	CIO 0000 ～ CIO 6142	
工作区	W000 ～ W510	
保持位区	H000 ～ H510	
辅助位区	A000 ～ A958	A448 ～ A958
定时器区	T0000 ～ T4094	
计数器区	C0000 ～ C4094	
DM 区	D00000 ～ D32766	
无区号 EM 区	E00000 ～ E32766	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#0000000 ～ #FFFFFFFF （常数）	---
数据寄存器	---	
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

说明

DEG(459) 把 S+1 和 S 中的 32 位浮点数从弧度转换为角度，并把结果放在 R+1 和 R 中。（源浮点数必须表示为 IEEE754 格式）



用下列公式把弧度转换为角度：

弧度× 180 / π = 角度

如果结果的绝对值大于可用浮点数表示的最大值，上溢出标志将置 ON，并且结果将以±∞输出。

如果结果的绝对值小于可用浮点数表示的最小值，下溢出标志将置 ON，并且结果将以 0 输出。

标志

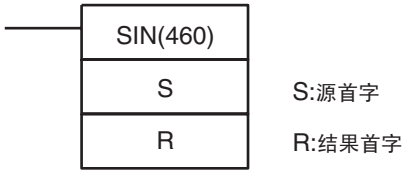
名称	标记	操作
错误标志	ER	如果源数据不是浮点数则置 ON。 如果源数据不是一个数字 （NaN），则置 ON。 其余情况下置 OFF。
等于标志	=	如果结果的指数和尾数都为 0 则置 ON。 其余情况下置 OFF。
上溢出标志	OF	如果结果的绝对值太大，而不能表示为 32 位浮点值，则置 ON。
下溢出标志	UF	如果结果的绝对值太小，而不能表示为 32 位浮点值，则置 ON。
负标志	N	结果为负时置 ON。 其余情况下置 OFF。

注意 S+1 和 S 中的源数据必须是 IEEE754 浮点数格式。

3-15-11 正弦：SIN(460)

用途 计算 32 位浮点数 （用弧度表示）的正弦，并把结果放在指定的结果字中。

梯形图符号



变化

变化	ON 条件时每个循环执行	SIN(460)
	上升沿微分时执行一次	@SIN(460)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

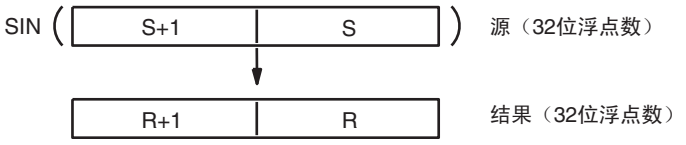
操作数规定

区域	S	R
CIO 区	CIO 0000 ～ CIO 6142	
工作区	W000 ～ W510	
保持位区	H000 ～ H510	
辅助位区	A000 ～ A958	A448 ～ A958
定时器区	T0000 ～ T4094	
计数器区	C0000 ～ C4094	
DM 区	D00000 ～ D32766	
无区号 EM 区	E00000 ～ E32766	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	

区域	S	R
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#00000000 ～ #FFFFFFFF（二进制）	---
数据寄存器	---	
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

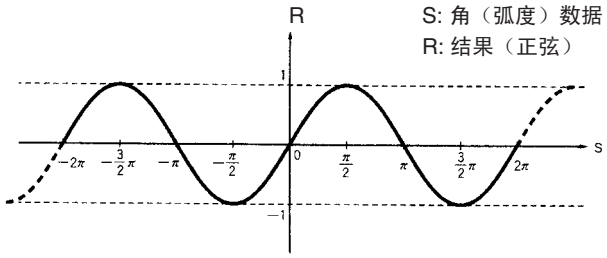
说明

SIN(460) 计算 S+1 和 S 中用 32 位浮点数表示的角（弧度）的正弦，并把结果放在 R+1 和 R 中。（源浮点数必须表示为 IEEE754 格式）



在 S+1 和 S 中用弧度定义所求的角 （-65535 ～ 65535）。如果角的绝对值超过 65535，将会出现错误，并且指令不执行。从角度转换为弧度的信息，*参阅 3-15-9 角度到弧度：RAD(458)*。

下图显示角和结果之间的关系。



标志

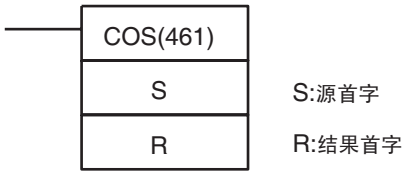
名称	标记	操作
错误标志	ER	如果源数据不是一个数字 （NaN）则置 ON。 如果源数据的绝对值超过 65535，则置 ON。 其余情况下置 OFF。
等于标志	=	如果结果的指数和尾数都为 0 则置 ON。 其余情况下置 OFF。
上溢出标志	OF	OFF
下溢出标志	UF	OFF
负标志	N	结果为负时置 ON。 其余情况下置 OFF。

注意 S+1 和 S 中的源数据必须是 IEEE754 浮点数格式。

3-15-12 余弦：COS(461)

用途 计算 32 位浮点数 （弧度）的余弦，并把结果放在指定的结果字中。

梯形图符号



变化

变化	ON 条件时每个循环执行	COS(461)
	上升沿微分时执行一次	@COS(461)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

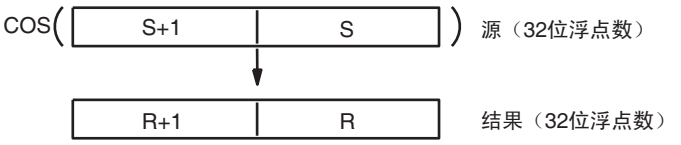
操作数规定

区域	S	R
CIO 区	CIO 0000 ～ CIO 6142	
工作区	W000 ～ W510	
保持位区	H000 ～ H510	
辅助位区	A000 ～ A958	A448 ～ A958
定时器区	T0000 ～ T4094	
计数器区	C0000 ～ C4094	
DM 区	D00000 ～ D32766	
无区号 EM 区	E00000 ～ E32766	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#00000000 ～ #FFFFFFFF （常数）	---
数据寄存器	---	

区域	S	R
索引寄存武器	---	
使用索引寄存器 的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

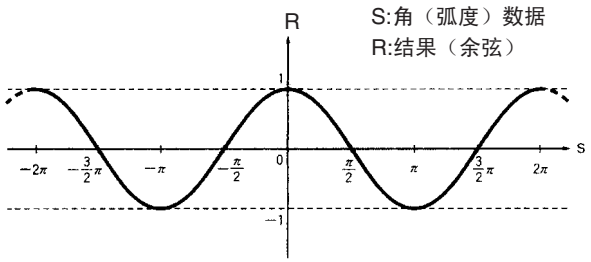
说明

COS(461) 计算 S+1 和 S 中用 32 位浮点数表示的角 （弧度）的余弦，并把结果放在 R+1 和 R 中。（浮点源数据必须是 IEEE754 格式）



在 S+1 和 S 中用弧度指定所求的角 （-65535 ～ 65535）。如果角的绝对值超过 65535，将产生错误，并且指令不执行。从角度转换为弧度的信息，*参阅 3-15-9 角度到弧度：RAD(458)*。

下图显示角和结果之间的关系。



标志

名称	标记	操作
错误标志	ER	如果源数据不是一个数字 （NaN）则置 ON。 如果源数据的绝对值超过 65535，则置 ON。 其余情况下置 OFF。
等于标志	=	如果结果的指数和尾数都为 0 则置 ON。 其余情况下置 OFF。
上溢出标志	OF	OFF
下溢出标志	UF	OFF
负标志	N	结果为负时置 ON。 其余情况下置 OFF。

注意

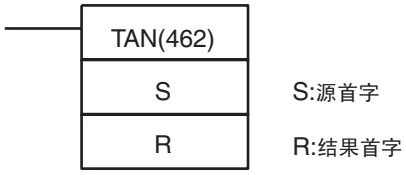
S+1 和 S 中的源数据必须是 IEEE754 浮点数格式。

3-15-13 正切：TAN(462)

用途

计算 32 位浮点数 （弧度）的正切，并把结果放在指定的结果字中。

梯形图符号



变化

变化	ON 条件时每个循环执行	TAN(462)
	上升沿微分时执行一次	@TAN(462)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

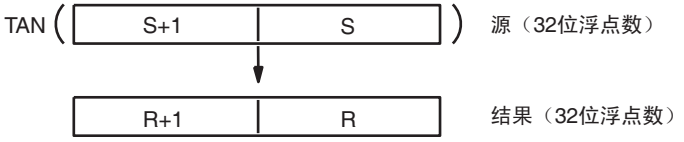
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数规定

区域	S	R
CIO 区	CIO 0000 ～ CIO 6142	
工作区	W000 ～ W510	
保持位区	H000 ～ H510	
辅助位区	A000 ～ A958	A448 ～ A958
定时器区	T0000 ～ T4094	
计数器区	C0000 ～ C4094	
DM 区	D00000 ～ D32766	
无区号 EM 区	E00000 ～ E32766	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#00000000 ～ #FFFFFFFF （二进制）	---
数据寄存器	---	
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

说明

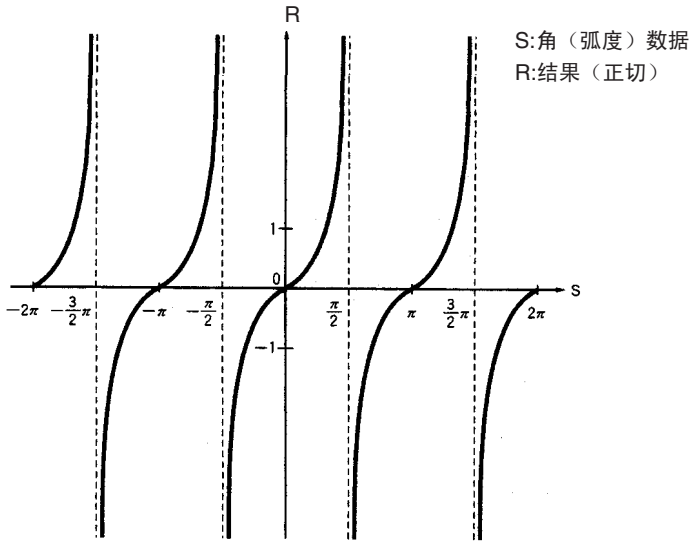
TAN(462) 计算 S+1 和 S 中用 32 位浮点数表示的角 （弧度）的正切，并把结果放在 R+1 和 R 中。（浮点源数据必须是 IEEE754 格式）



在 S+1 和 S 中用弧度指定所求角 （-65535 ～ 65535）。如果角的绝对值超过 65535，将产生错误，并且指令不执行。从角度转换为弧度的信息，*参见 3-15-9 角度到弧度：RAD(458)*。

如果结果的绝对值大于可用浮点数表示的最大值，上溢出标志将置 ON，并且结果将以±∞输出。

下图显示角和结果之间的关系。



标志

名称	标记	操作
错误标志	ER	如果源数据不是一个数字 （NaN）则置 ON。 如果源数据的绝对值超过 65535，则置 ON。 其余情况下置 OFF。
等于标志	=	如果结果的指数和尾数都为 0 则置 ON。 其余情况下置 OFF。
上溢出标志	OF	OFF
下溢出标志	UF	OFF
负标志	N	结果为负时置 ON。 其余情况下置 OFF。

注意

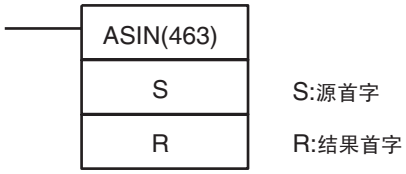
S+1 和 S 中的源数据必须是 IEEE754 浮点数格式。

3-15-14 反正弦：ASIN(463)

用途

计算 32 位浮点数的反正弦，并把结果放在指定的结果字中。（反正弦函数是正弦函数的反函数；它返所给的正弦值为 -1 ～ 1 之间的角）。

梯形图符号



变化

变化	ON 条件时每个循环执行	ASIN(463)
	上升沿微分时执行一次	@ASIN(463)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

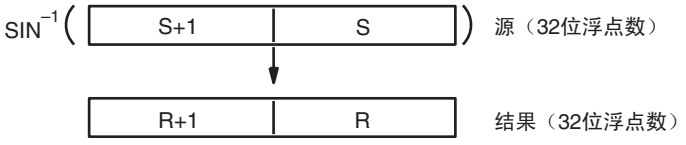
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数规定

区域	S	R
CIO 区	CIO 0000 ～ CIO 6142	
工作区	W000 ～ W510	
保持位区	H000 ～ H510	
辅助位区	A000 ～ A958	A448 ～ A958
定时器区	T0000 ～ T4094	
计数器区	C0000 ～ C4094	
DM 区	D00000 ～ D32766	
无区号 EM 区	E00000 ～ E32766	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#00000000 ～ #FFFFFFFF （二进制）	---
数据寄存器	---	
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

说明

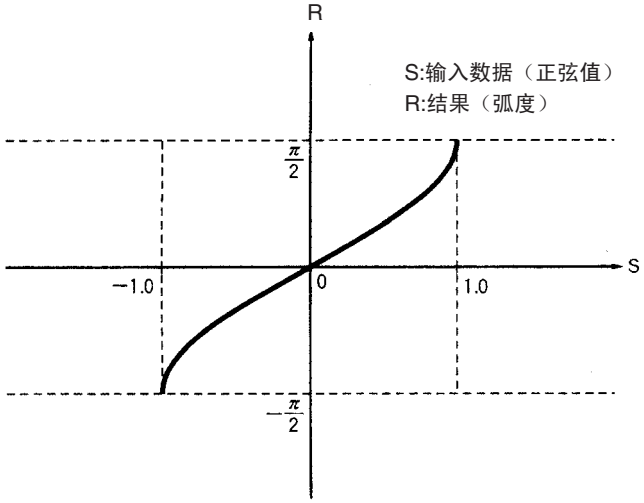
ASIN(463) 计算 S+1 和 S 中用 32 位浮点数表示正弦值的角 （弧度），并把结果放在 R+1 和 R 中。（浮点源数据必须是 IEEE754 格式）



源数据必须在 -1.0 和 1.0 之间。如果源数据的绝对值超过 1.0，将会出现错误，并且指令不执行。

结果以从 - π /2 到 π /2 之间的角 （弧度）形式输出到字 R+1 和 R。

下图显示了输入数据和结果之间的关系。



标志

名称	标记	操作
错误标志	ER	如果源数据不是浮点数则置 ON。 如果源数据不是一个数字 （NaN）则置 ON。 如果源数据的绝对值超过 1.0 则置 ON。 其余情况下置 OFF。
等于标志	=	如果结果的指数和尾数都为 0 则置 ON。 其余情况下置 OFF。
上溢出标志	OF	OFF
下溢出标志	UF	OFF
负标志	N	结果为负时置 ON。 其余情况下置 OFF。

注意

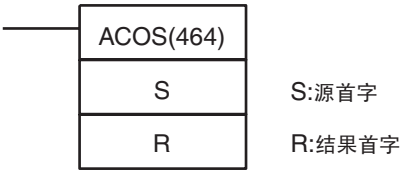
S+1 和 S 中的源数据必须是 IEEE754 浮点数格式。

3-15-15 反余弦：ACOS(464)

用途

计算 32 位浮点数的反余弦，并把结果放在指定的结果字中。（反余弦函数是余弦函数的反函数；它返所给的余弦值为 -1 ～ 1 之间的角）。

梯形图符号



变化

变化	ON 条件时每个循环执行	ACOS(464)
	上升沿微分时执行一次	@ACOS(464)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

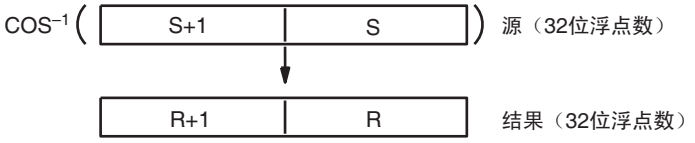
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数规定

区域	S	R
CIO 区	CIO 0000 ～ CIO 6142	
工作区	W000 ～ W510	
保持位区	H000 ～ H510	
辅助位区	A000 ～ A958	A448 ～ A958
定时器区	T0000 ～ T4094	
计数器区	C0000 ～ C4094	
DM 区	D00000 ～ D32766	
无区号 EM 区	E00000 ～ E32766	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#00000000 ～ #FFFFFFFF （二进制）	---
数据寄存器	---	
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

说明

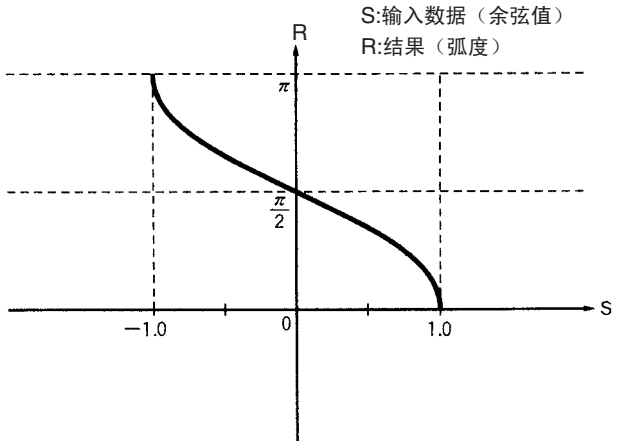
ACOS(464) 计算 S+1 和 S 中余弦值表达为 32 位浮点数的角（弧度），并把结果放在 R+1 和 R 中。（浮点源数据必须是 IEEE754 格式）



源数据必须在 -1.0 和 1.0 之间。如果源数据的绝对值超过 1.0，将会出现错误，并且指令不执行。

结果以从 0 到 π 范围内的角 （弧度）形式输出到字 R+1 和 R 中。

下图显示了输入数据和结果之间的关系。



标志

名称	标记	操作
错误标志	ER	如果源数据不是浮点数据，则置 ON。 如果源数据不是一个数字 （NaN）则置 ON。 如果源数据的绝对值超过 1.0 则置 ON。 其余情况下置 OFF。
等于标志	=	如果结果的指数和尾数都为 0 则置 ON。 其余情况下置 OFF。
上溢出标志	OF	OFF
下溢出标志	UF	OFF
负标志	N	结果为负时置 ON。 其余情况下置 OFF。

注意

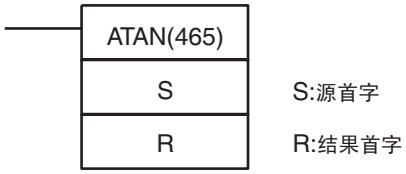
S+1 和 S 中的源数据必须是 IEEE754 浮点数格式。

3-15-16 反正切：ATAN(465)

用途

计算 32 位浮点数的反正切，并把结果放在指定的结果字中。（反正切函数是正切函数的反函数；它返回产生所给的正切值的角）。

梯形图符号



变化

变化	ON 条件时每个循环执行	ATAN(465)
	上升沿微分时执行一次	@ATAN(465)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

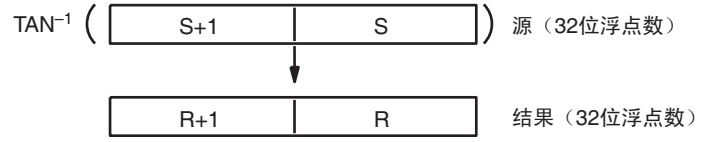
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数规定

区域	S	R
CIO 区	CIO 0000 ～ CIO 6142	
工作区	W000 ～ W510	
保持位区	H000 ～ H510	
辅助位区	A000 ～ A958	A448 ～ A958
定时器区	T0000 ～ T4094	
计数器区	C0000 ～ C4094	
DM 区	D00000 ～ D32766	
无区号 EM 区	E00000 ～ E32766	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#00000000 ～ #FFFFFFFF （二进制）	---
数据寄存器	---	
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

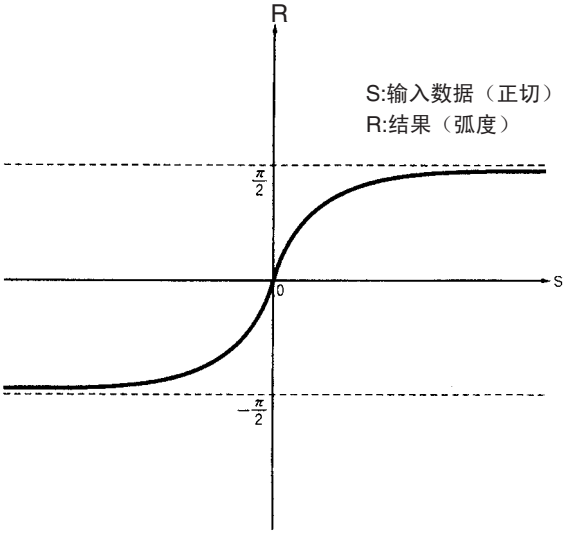
说明

ATAN(465) 计算角 （用弧度表示），它的正切值在 S+1 和 S 中以 32 位浮点数表示，并把结果放在 R+1 和 R 中。（浮点源数据必须是 IEEE754 格式）



结果以在 - π /2 到π /2 的范围内的角度 （用弧度表示）的形式输出到字 R+1 和 R 中。

下图显示了输入数据和结果之间的关系。



标志

名称	标记	操作
错误标志	ER	如果源数据不是浮点数据，则置 ON。 如果源数据不是一个数字 （NaN）则置 ON。 其余情况下置 OFF。
等于标志	=	如果结果的指数和尾数都为 0 则置 ON。 所有其余情况下置 OFF。
上溢出标志	OF	OFF
下溢出标志	UF	OFF
负标志	N	结果为负时置 ON。 所有其余情况下置 OFF。

注意

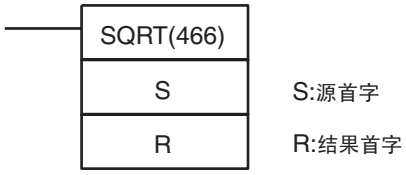
S+1 和 S 中的源数据必须是 IEEE754 浮点数格式。

3-15-17 平方根：SQRT(466)

用途

计算 32 位浮点数的平方根，并把结果放在指定的结果字中。

梯形图符号



变化

变化	ON 条件时每个循环执行	SQRT(466)
	上升沿微分时执行一次	@SQRT(466)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

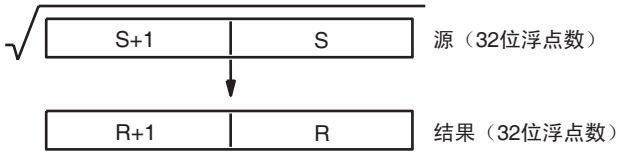
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数规定

区域	S	R
CIO 区	CIO 0000 ～ CIO 6142	
工作区	W000 ～ W510	
保持位区	H000 ～ H510	
辅助位区	A000 ～ A958	A448 ～ A958
定时器区	T0000 ～ T4094	
计数器区	C0000 ～ C4094	
DM 区	D00000 ～ D32766	
无区号 EM 区	E00000 ～ E32766	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#00000000 ～ #FFFFFFFF （二进制）	---
数据寄存器	---	
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

说明

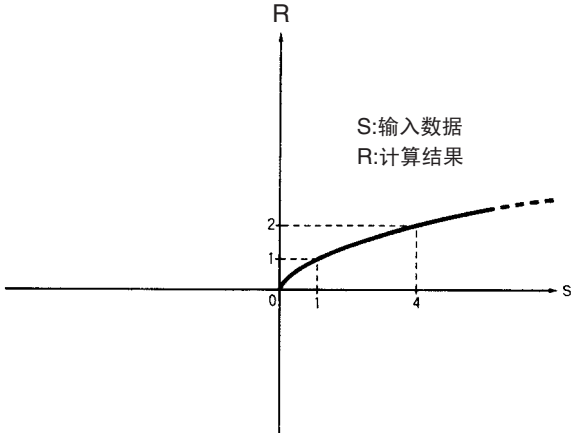
SQRT(466) 计算在 S+1 和 S 中的 32 位浮点数的平方根，并将计算结果存入 R+1 和 R 中。（浮点源数据必须是 IEEE754 格式）



源数据必须为正，如为负则会产生一个错误，并且指令不执行。

如果计算结果的绝对值大于浮点数所能表示的最大值，上溢出标志置为 ON，计算结果输出为 ±∞。

下图显示了输入数据和计算结果的关系。



标志

名称	标记	操作
错误标志	ER	如果源数据不是浮点数据，则置 ON。 源数据为负时则置 ON。 源数据非数字 （NaN）时置 ON。 其余情况下置 OFF。
等于标志	=	计算结果的指数和尾数全为 0 置 ON。 其余情况下置 OFF。
上溢出标志	OF	计算结果的绝对值太大，超出 32 位浮点数表示范围时置 ON。
下溢出标志	UF	OFF
负标志	N	OFF

注意

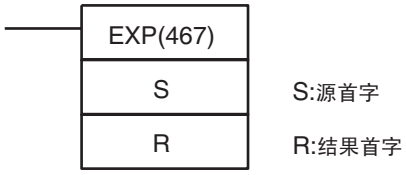
表示为 S+1 和 S 的源数据必须是 IEEE754 浮点数格式。

3-15-18 指数：EXP(467)

用途

计算 32 位浮点数的自然指数 （底为 e），并把结果存入指定的结果字中。

梯形图符号



变化

变化	ON 条件时每个循环执行	EXP(467)
	上升沿微分时执行一次	@EXP(467)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

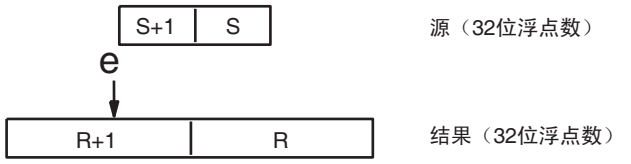
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数规定

区域	S	R
CIO 区	CIO 0000 ～ CIO 6142	
工作区	W000 ～ W510	
保持位区	H000 ～ H510	
辅助位区	A000 ～ A958	A448 ～ A958
定时器区	T0000 ～ T4094	
计数器区	C0000 ～ 4094	
DM 区	D00000 ～ D32766	
无区号 EM 区	E00000 ～ E32766	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#00000000 ～ #FFFFFFFF （二进制）	---
数据寄存器	---	
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

说明

EXP(467) 计算在 S+1 和 S 中的 32 位浮点数的自然（底为 e）指数值，并将计算结果存入 R+1 和 R 中，即 EXP(467) 计算 eX（X 为源数据），并将结果存入 R+1 和 R。

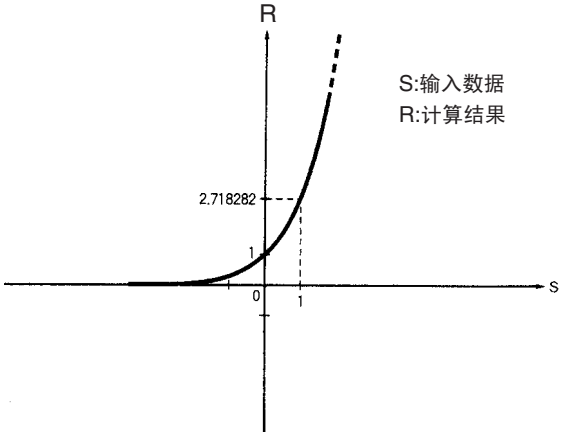


如果计算结果的绝对值大于浮点数所能表示的最大值，上溢出标志置为 ON，计算结果输出为 ±∞。

如果计算结果的绝对值小于浮点数所能表示的最小值，下溢出标志置为 ON，计算结果输出为 0。

注 常数 e 为 2.718282。

下图显示了输入数据和计算结果的关系。



标志

名称	标记	操作
错误标志	ER	如果源数据不是浮点数据，则置 ON。 源数据为负时则置 ON。 其余情况下置 OFF。
等于标志	=	计算结果的指数和尾数全为 0 置 ON。 其余情况下置 OFF。
上溢出标志	OF	计算结果的绝对值太大，超出 32 位浮点数表示范围时置 ON。
下溢出标志	UF	计算结果的绝对值太小，超出 32 位浮点数表示范围时置 ON。
负标志	N	OFF

注意

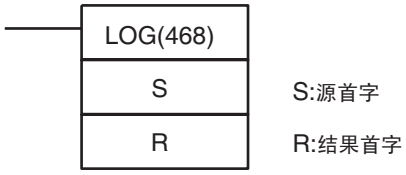
表示为 S+1 和 S 的源数据必须是 IEEE754 浮点数格式。

3-15-19 对数：LOG(468)

用途

计算 32 位浮点数的自然对数 （底为 e），并把结果存入指定的结果字中。

梯形图符号



变化

变化	ON 条件时每个循环执行	LOG(468)
	上升沿微分时执行一次	@LOG(468)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

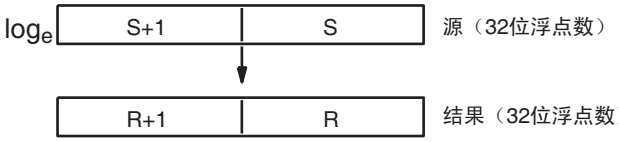
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数规定

区域	S	R
CIO 区	CIO 0000 ～ CIO 6142	
工作区	W000 ～ W510	
保持位区	H000 ～ H510	
辅助位区	A000 ～ A958	A448 ～ A958
定时器区	T0000 ～ T4094	
计数器区	C0000 ～ C4094	
DM 区	D00000 ～ D32766	
无区号 EM 区	E00000 ～ E32766	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#00000000 ～ #FFFFFFFF （二进制）	---
数据寄存器	---	
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

说明

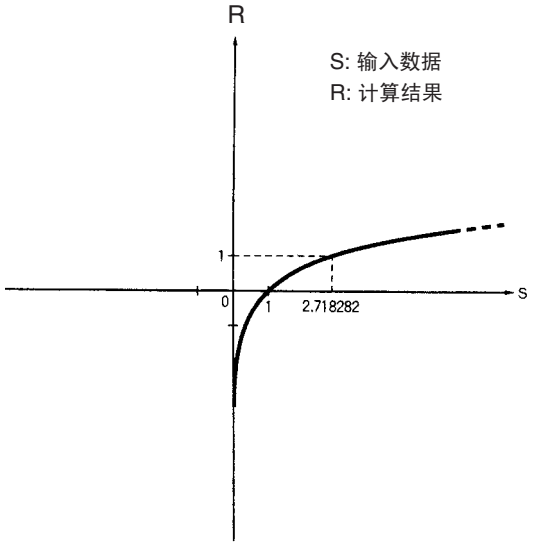
LOG(468) 计算在 S+1 和 S 中的 32 位浮点数的自然对数（底为 e），并将结果存入 R+1 和 R。



源数据必须为正值，如果为负则产生一个错误，并且指令不执行。
如果计算结果的绝对值大于浮点数所能表示的最大值，上溢出标志置为 ON，计算结果输出为 ±∞。

注 常数 e 为 2.718282。

下图显示了输入数据和计算结果的关系。



标志

名称	标记	操作
错误标志	ER	如果源数据不是浮点数据，则置 ON。 源数据为负时则置 ON。 源数据非数字 （NaN）时置 ON。 其余情况下置 OFF。
等于标志	=	计算结果的指数和尾数全为 0 置 ON。 其余情况下置 OFF。
上溢出标志	OF	计算结果的绝对值太大，超出 32 位浮点数表示范围时置 ON。
下溢出标志	UF	OFF
负标志	N	计算结果为负时置 ON。 其余情况下置 OFF。

注意

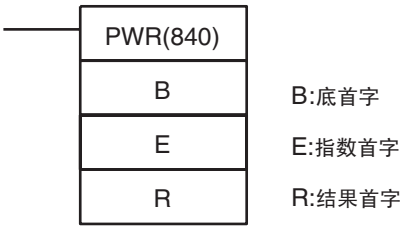
在 S+1 和 S 中的源数据必须是 IEEE754 浮点数据格式。

3-15-20 指数幂：PWR(840)

用途

将一个 32 位浮点数加到另一个 32 位浮点数上作为幂。

梯形图符号



变化

变化	ON 条件时每个循环执行	PWR(840)
	上升沿微分时执行一次	@PWR(840)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

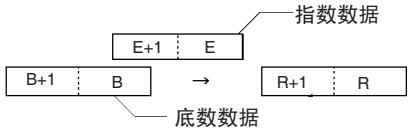
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数规定

区域	B	E	R
CIO 区	CIO 0000 ～ CIO 6142		
工作区	W000 ～ W510		
保持位区	H000 ～ H510		
辅助位区	A000 ～ A958		A448 ～ A958
定时器区	T0000 ～ T4094		
计数器区	C0000 ～ C4094		
DM 区	D00000 ～ D32766		
无区号 EM 区	E00000 ～ E32766		
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#00000000 ～ #FFFFFFFF （二进制）		---
数据寄存器	---		
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

PWR(840) 将以 B+1 和 B 表示的 32 位浮点数自乘到以 E+1 和 E 表示的另一个 32 位浮点数次幂。换言之，PWR(840) 计算 XY（X=B+1 和 B； Y=E+1 和 E）。



例如，当底数 (B+1,B) 为 3.1 而指数 (E+1,E) 为 3 时，计算结果为 3.13 或 29.791。
如果计算结果的绝对值大于浮点数所能表示的最大值，上溢出标志置为 ON。
如果计算结果的绝对值小于浮点数所能表示的最小值，下溢出标志置为 ON。

标志

名称	标记	操作
错误标志	ER	如果底数 (B+1,B) 或指数 (E+1,E) 没有当作浮点数据确认时置 ON。 如果底数 (B+1,B) 或指数 (E+1,E) 不是一个数字 (NaN) 时则置 ON。 如果底数 (B+1,B) 为 0，并且指数 (E+1,E) 小于 0 时置 ON。（除以 0） 如果底数 (B+1,B) 为负，并且指数 (E+1,E) 为非整数时置 ON。（一个负数的根） 其余情况下置 OFF。
等于标志	=	如果结果的指数和尾数全为 0 时置 ON。 其余情况下置 OFF。
上溢出标志	OF	如果结果的绝对值太大而不能用 32 位浮点值表示时置 ON。
下溢出标志	UF	如果结果的绝对值太小而不能用 32 位浮点值表示时置 ON。
负标志	N	如果结果为负时置 ON。 其余情况下置 OFF。

注意

底数 (B+1,B) 和指数 (E+1.E) 必须是 IEEE754 浮点数据格式。

3-15-21 单精度浮点比较指令

用途

这些输入比较指令比较两个单精度浮点数值 （32 位 IEEE754 常数和 / 或指定字的内容）并且当比较条件为真时产生一个 ON 执行条件。
这些指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持。

注 关于有符号和无符号二进制输入比较指令详细情况请参见 *3-7-1 输入比较指令（300 ～ 328）*，并且关于双精度浮点输入比较指令详情可参见 *3-26-21 双精度浮点输入指令*。

梯形图符号

变化

变化	每个循环比较为真时产生 ON	输入比较指令
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数规定

区域	S1	S2
CIO 区	CIO 0000 ～ CIO 6142	
工作区	W000 ～ W510	
保持位区	H000 ～ H510	
辅助位区	A000 ～ A958	
定时器区	T0000 ～ T4094	
计数器区	C0000 ～ C4094	
DM 区	D00000 ～ D32766	
无区号 EM 区	E00000 ～ E32766	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#00000000 ～ #FFFFFFFF （二进制）	
数据寄存器	---	
索引寄存器	IR0 ～ IR15 （仅对无符号数据）	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

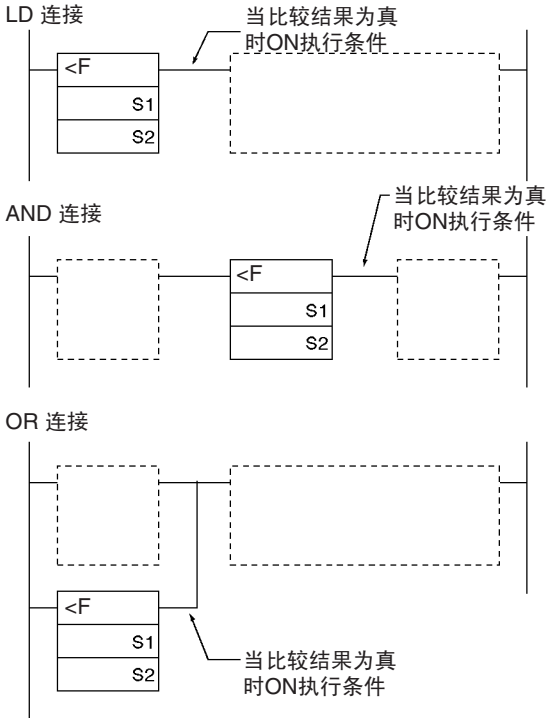
说明

输入比较指令把 S1 和 S2 中指定的数据作为单精度浮点数值 （32 位 IEEE754 数据）进行比较，并且当比较条件为真时产生一个 ON 执行条件。当数据存储在字中时， S1 和 S2 指定了两个包含 32 位数据字中的第一个。也有可能作为一个 8 位 16 进制常数输入浮点数据。

输入指令

输入比较指令被看作如 LD， AND，和 OR 指令用来控制后面指令的执行。

输入类型	操作
LD	指令能直接与左边总线相连
AND	指令不能直接与左边总线相连
OR	指令能直接与左边总线相连



选项

由 3 种输入类型和 6 种符号，有 18 种不同的可能组合。

符号		选项 （数据格式）
=	（等于）	F：单精度浮点数据
< >	（不等于）	
<	（小于）	
<=	（小于或等于）	
>	（大于）	
>=	（大于或等于）	

输入比较指令归纳

下表显示了 18 个单精度浮点数输入比较指令的功能代码。（$C1=S_1+1$， S_1 和 $C2=S_2+1$， S_2）

代码	助记符	名称	函数
329	LD=F	加载浮点等于	如果 C1=C2 则为真
	AND=F	与浮点等于	
	OR=F	或浮点等于	

代码	助记符	名称	函数
330	LD<>F	加载浮点不等于	如果 C1 ≠ C2 则为真
	AND<>F	与浮点不等于	
	OR<>F	或浮点不等于	
331	LD<F	加载浮点小于	如果 C1<C2 则为真
	AND<F	与浮点小于	
	OR<F	或浮点小于	
332	LD<=F	加载浮点小于或等于	如果 C1 ≤ C2 则为真
	AND<=F	与浮点小于或等于	
	OR<=F	或浮点小于或等于	
333	LD>F	加载浮点大于	如果 C1>C2 则为真
	AND>F	与浮点大于	
	OR>F	或浮点大于	
325	LD>=F	加载浮点大于或等于	如果 C1 ≥ C2 则为真
	AND>=F	与浮点大于或等于	
	OR>=F	或浮点大于或等于	

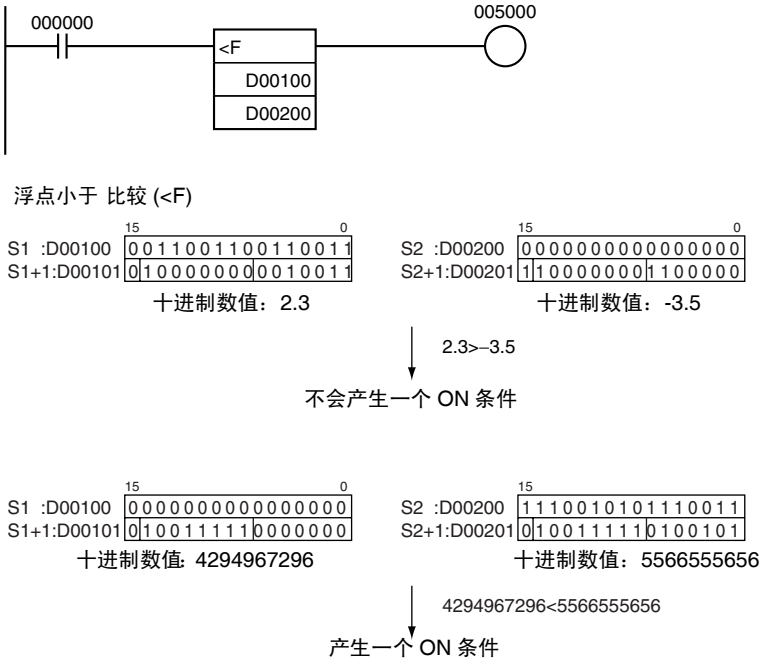
标志

名称	标记	操作
错误标志	ER	如果 S1+1,S1 或 S2+1,S2 不是一个有效的浮点数字（NaN）则置 ON。 如果 S1+1， S1 或 S2+1 是 + ∞则置 ON。 如果 S1+1， S1 或 S2+1 是 - ∞则置 ON。 其余情况下置 OFF。
大于表字	>	如果 S1+1， S1>S2+1， S2 则置 ON。 其余情况下置 OFF。
大于或等于标志	> =	如果 S1+1， S1 ≥ S2+1， S2 则置 ON。 其余情况下置 OFF。
等于标志	=	如果 S1+1， S1=S2+1， S2 则置 ON。 其余情况下置 OFF。
不等于标志	≠	如果 S1+1， S1 ≠ S2+1， S2 则置 ON。 其余情况下置 OFF。
小于标志	<	如果 S1+1， S1<S2+1， S2 则置 ON。 其余情况下置 OFF。
小于或等于标志	< =	如果 S1+1， S1 ≤ S2+1， S2 则置 ON。 其余情况下置 OFF。
负标志	N	不变

注意 输入比较指令不能被用做右手指令，也就是说，在它们与右边总线之间必须使用另一指令。

例 **与浮点小于：AND<F(331)**

在下例中当CIO 000000置ON时，D00101，D00100中的浮点数据与D00201，D00200 中的浮点数据进行比较。如果 D00101， D00100 中的内容小于 D00201， D00200 中的内容，执行进入下一行并且 CIO 005000 变为 ON。如果 D00101， D00100 中的内容不小于 D00201， D00200 中的内容，执行不会处理紧接的指令行。



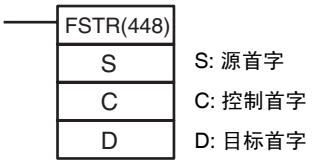
3-15-22 浮点数到 ASCII 码：FSTR(448)

用途

用标准十进制表示法或科学表示法表示一个 32 位浮点值 （IEEE754 格式）并且将这个值转换为 ASCII 文本。

这些指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持

梯形图符号



变化

变化	ON 条件时每个循环执行	FSTR(448)
	上升沿微分时执行一次	@FSTR(448)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断程序
OK	OK	OK	OK

操作数规定

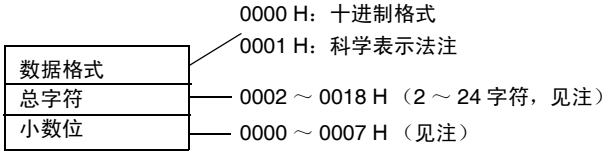
区域	S	C	D
CIO 区	CIO 0000 ～ CIO 6142	CIO 0000 ～ CIO 6141	CIO 0000 ～ CIO 6143
工作区	W000 ～ W510	W000 ～ W509	W000 ～ W511
保持位区	H000 ～ H510	H000 ～ H509	H000 ～ H511
辅助位区	A000 ～ A958	A000 ～ A957	A448 ～ A959
定时器区	T0000 ～ T4094	T0000 ～ T4093	T0000 ～ T4095

区域	S	C	D
辅助位区	C0000 ～ C4094	C0000 ～ C4093	C0000 ～ C4095
DM 区	D00000 ～ D32766	D00000 ～ D32765	D00000 ～ D32767
无区号 EM 区	E00000 ～ E32766	E00000 ～ E32765	E00000 ～ E32767
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	En_00000 ～ En_32765 (n = 0 ～ C)	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	#00000000 ～ #FFFFFFFF （二进制）	---	
数据寄存器	---		
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –()IR15 ,IR0 ～ ,IR15		

说明

FSTR(448) 根据字 C 到 C+2 中的控制数据，把 S+1 和 S（IEEE754 格式）中用十进制表示法或科学表示法表示的 32 位浮点数，转换为 ASCII 文本，并且输出结果到 D 开始的目标字中。

下图显示了 3 个控制字的内容。



注：总字符数和小数位数有限定。详细情况见*第 564 页 ASCII 字符数的限定。*

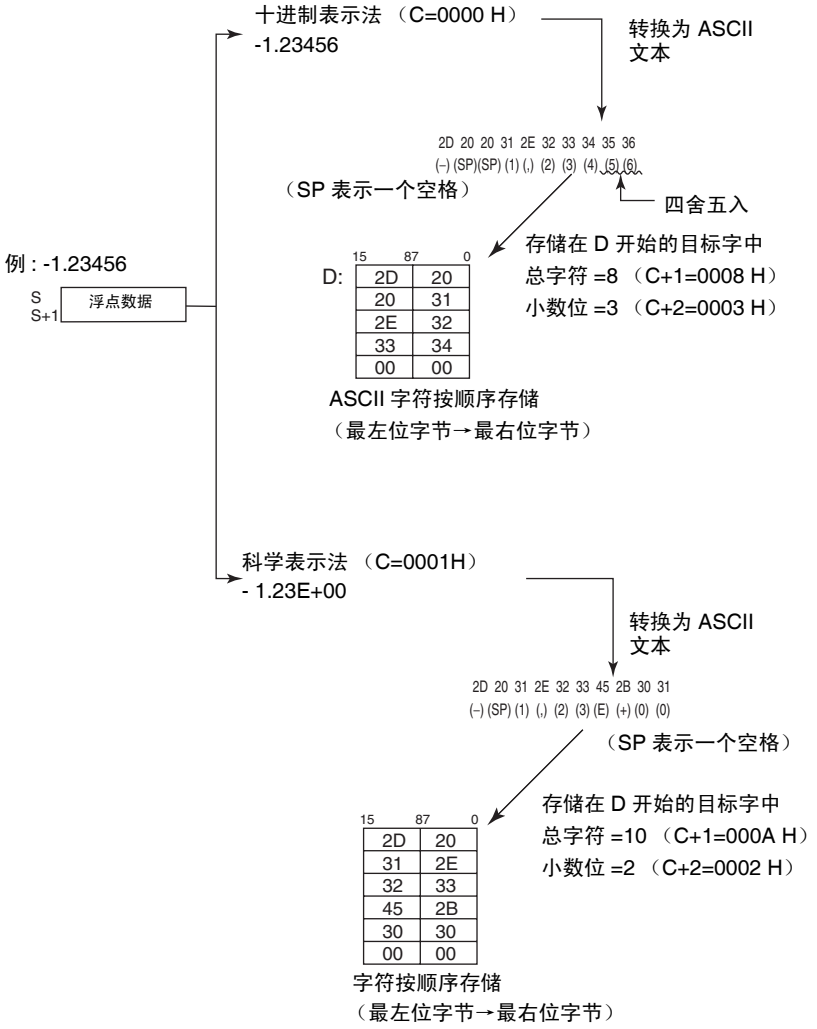
- C（数据格式）的内容说明是否用十进制表示法或科学表示法表示 S+1，S 中的数字。
  - 十进制表示法
    将一个实数表示为一个整数和小数部分。
    例：124.56

• 科学表示法
将一个实数表示为一个整数部分，小数部分，和指数部分。
例：1.2456E-2(1.2456 × 10^{-2})

• C+1 （总字符）的内容说明转换后的 ASCII 字符数，包括正负号符号，数字，小数点和空格。

• C+2 （小数位）的内容说明在小数点后的位 （字符）数。

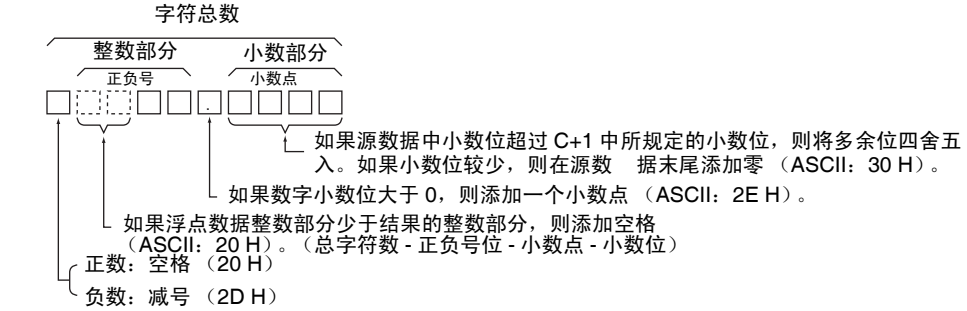
ASCII 文本按如下顺序存储在 D 及其后面的字中：
D 最左位字节， D 最右位字节， D+1 最左位字节， D+1 最右位字节，等等。



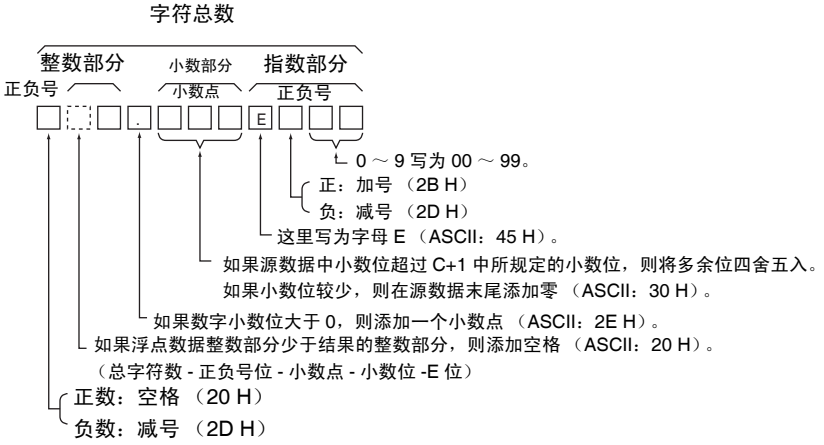
ASCII 文本的存储

浮点数字转换为 ASCII 文本后，ASCII 字符存储在 D 开始的目标字中，
如下图所示。十进制表示法和科学表示法分别采用不同的存储方法。

十进制表示法 （C=0000 H）



科学表示法（C=0001 H）



注 零的一个或两个字节作为一个结束代码加在 ASCII 文本的末尾。
总字符数为奇数：00 H 存储在 ASCII 文本之后。

总字符数为偶数：0000 H 存储在 ASCII 文本之后。

ASCII 字符数的限定

在已转换的数字中 ASCII 字符数有限定。
如果字符数超出容许的最大值，则错误标志将变为 ON。

1. ASCII 总字符数的限定
   a) 十进制表示法 （C=0000 H）
      - 当没有小数部分时 （C+2=0000 H）
        2 ≤总字符≤ 24
      - 当有小数部分时 （C+2=0001 ～ 0007 H）
        （小数位 +3）≤总字符≤ 24

   b) 科学表示法 （C=0001 H）
      - 当没有小数部分时 （C+2=0000 H）
        6 ≤总字符≤ 24
      - 当有小数部分时 （C+2=0001 ～ 0007 H）
        （小数位 +7）≤总字符≤ 24
2. 整数部分位数的限定

a) 十进制表示法 （C=0000 H）

- 当没有小数部分时 （C+2=0000 H）
  1 ≤整数位数≤ 24
- 当有小数部分时 （C+2=0001 ～ 0007 H）
  1 ≤整数位数≤ （24- 小数位 -2）

b) 科学表示法 （C=0001 H）
1 个数位 （固定）

3. 小数部分位数的限定

a) 十进制表示法 （C=0000 H）

- 小数位≤ 7
- 同样：小数位≤ （总 ASCII 字符数 -3）

b) 科学表示法 （C=0001 H）

- 小数位≤ 7
- 同样：小数位≤ （总 ASCII 字符数 -3）

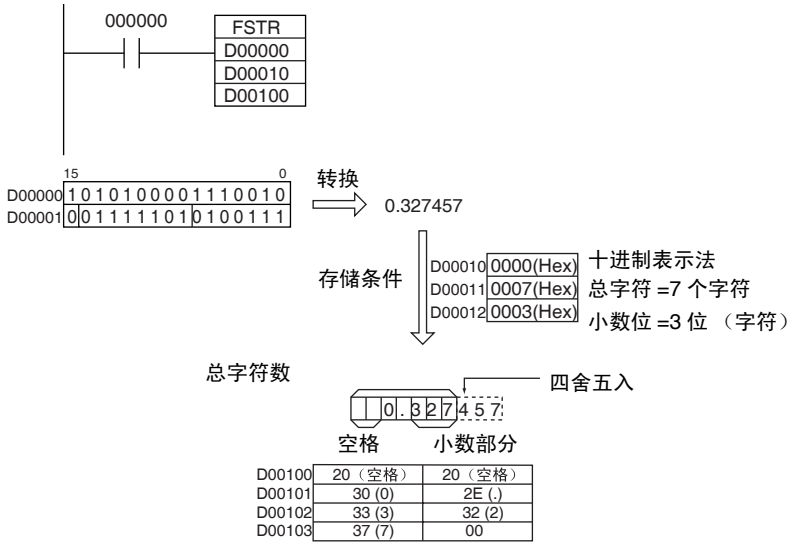
标志

名称	标记	操作
错误标志	ER	如果 S+1 或 S 中的数据不是一个有效的浮点数字（NaN）则置 ON。 如果 S+1 或 S 中的数据是 + ∞或 - ∞则置 ON。 如果 C 中数据格式设置不是 0000 或 0001 则置 ON。 如果 C+1 中总字符设置不在允许范围之内则置 ON。（详情见*上面 1， ASCII 总字符数的限定*）。 如果 C+2 中小数位设置不在允许范围之内则置 ON。（详情见*上面 3，小数部分位数的限定*）。 其余情况下置 OFF。
等于标志	=	如果转换结果为 0 置 ON。 其余情况下置 OFF。

例

转换为十进制表示的 ASCII 文本

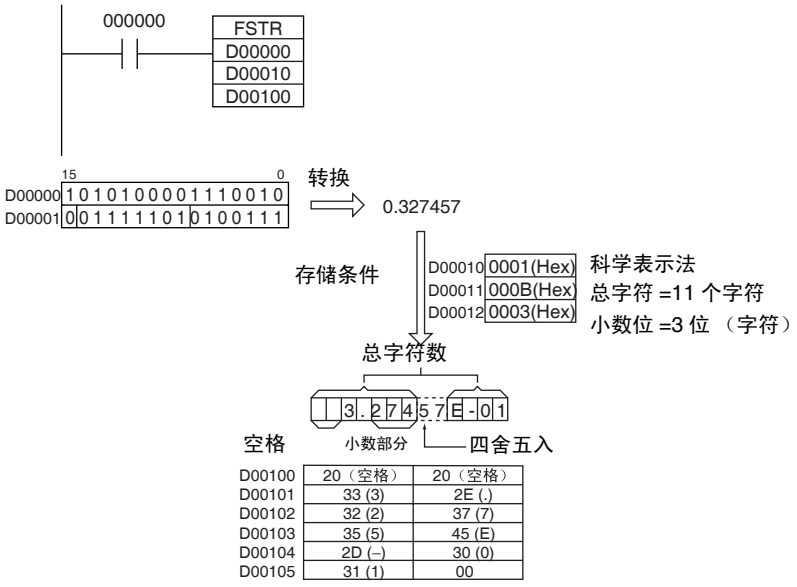
在下例中当 CIO000000 置 ON 时， FSTR(448) 将 D00001 和 D00000 中浮点数据转换为十进制表示的 ASCII 文本，并且将 ASCII 文本写入 D00100 开始的目标字中。控制字 （D00010 和 D00012）的内容说明了数据格式 （十进制表示法，总的 7 个字符， 3 个小数位）的详细情况。



D00100	20（空格）	20（空格）
D00101	30 (0)	2E (.)
D00102	33 (3)	32 (2)
D00103	37 (7)	00

转换为科学表示的 ASCII 文本

在下例中当 CIO000000 置 ON 时， FSTR(448) 将 D00001 和 D00000 中浮点数据转换为科学表示的 ASCII 文本，并且将 ASCII 文本写入 D00100 开始的目的字中。控制字（D00010 和 D00012）的内容说明了数据格式（科学表示法，总的 11 个字符， 3 个小数位）的详细情况。



D00100	20（空格）	20（空格）
D00101	33 (3)	2E (.)
D00102	32 (2)	37 (7)
D00103	35 (5)	45 (E)
D00104	2D (–)	30 (0)
D00105	31 (1)	00

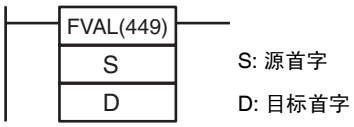
3-15-23 ASCII 码到浮点数：FVAL(449)

用途

把一个用 ASCII 文本 （十进制或科学表示法）表示的数字转换为一个 32 位浮点值 （IEEE754 格式）并且输出浮点值到指定的字中。

这些指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每个循环执行	FVAL(449)
	上升沿微分时执行一次	@FVAL(449)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数规定

区域	S	D
CIO 区	CIO 0000 ～ CIO 6143	CIO 0000 ～ CIO 6142
工作区	W000 ～ W511	W000 ～ W510
保持位区	H000 ～ H511	H000 ～ H510
辅助位区	A000 ～ A959	A448 ～ A958
定时器区	T0000 ～ T4095	T0000 ～ T4094
计数器区	C0000 ～ C4095	C0000 ～ C4094
DM 区	D00000 ～ D32767	D00000 ～ D32766
无区号 EM 区	E00000 ～ E32767	E00000 ～ E32766
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	En_00000 ～ En_32766 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	
数据寄存器	---	
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –( )IR15 ,IR0 ～ ,IR15	

说明

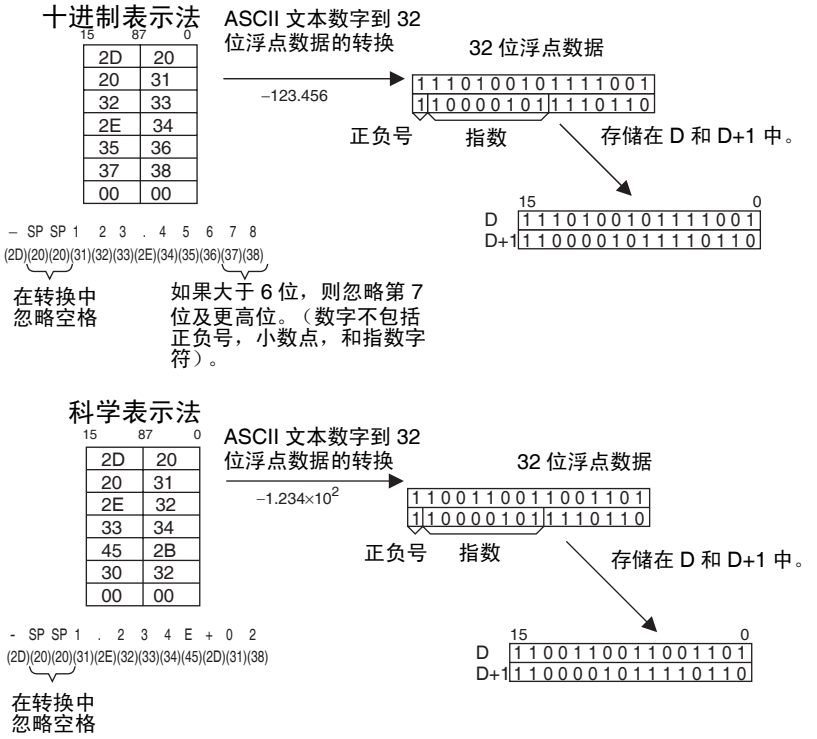
FVAL(449) 将指定的 ASCII 文本数字 （字 S 开始）转换为一个 32 位浮点数字（IEEE754 格式），并且输出结果到 D 开始的目的字中。

FVAL(449) 如果满足下列条件可用十进制或科学表示法转换 ASCII 文本：

- 十进制表示法
  实数表示为一个整数和小数部分。
  例：124.56
- 科学表示法
  实数表示为一个整数，小数，和指数部分。
  例：1.2456E-2 （1.2456×10^{-2}）

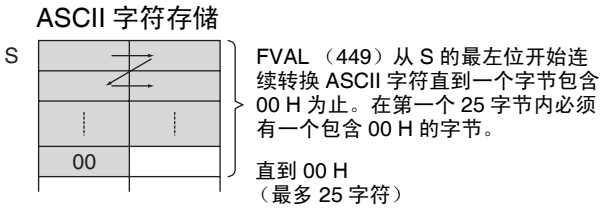
数据格式 （十进制或科学表示法）可自动检测。

ASCII 文本必须按下列顺序存储在 S 及其后面的字中：S 的最左位字节， S 的最右位字节， S+1 的最左位字节， S+1 的最右位字节，等等。

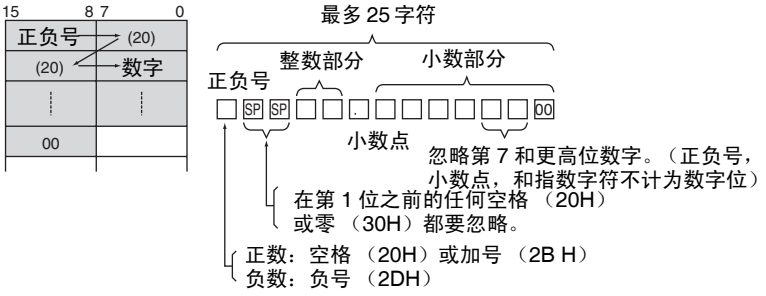


ASCII 文本的存储

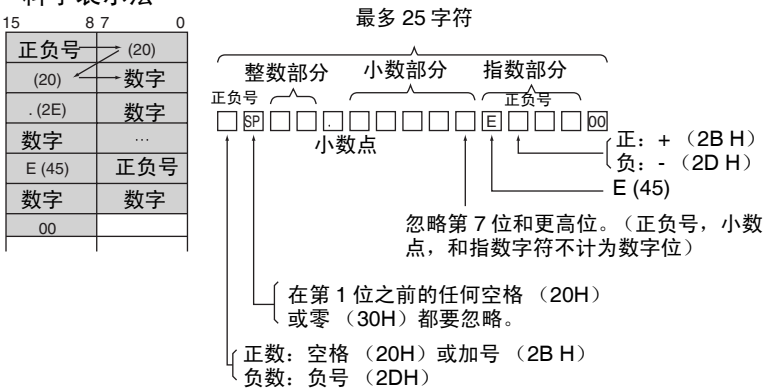
下图表明 ASCII 文本数字怎样转换为浮点数据。用十进制表示法和科学表示法存储的数字分别用不同的转换方法。



十进制表示法



科学表示法



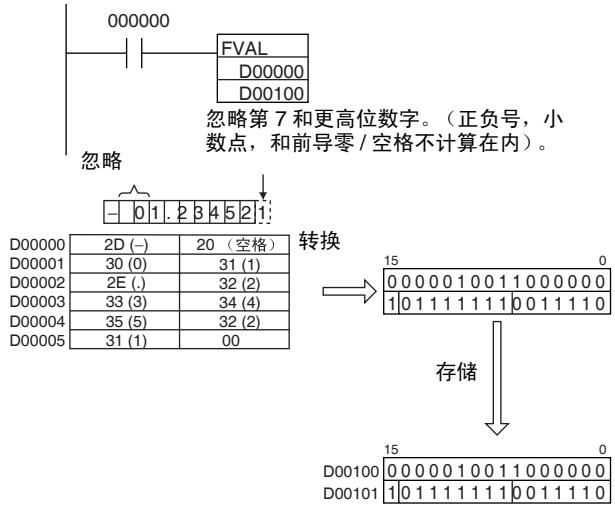
标志

名称	标记	操作
错误标志	ER	如果从 S 开始的源数据中的数字 （整数和小数部分）不是 30 ～ 39 H(0 ～ 9) 则置 ON。 如果指数部分开始的两个数字不为 45 和 2B H(E+) 或 45 和 2D H （E-）则置 ON。（从 S 开始的源数据中的 （整数和小数部分）不是 30 ～ 39 H(0 ～ 9)）。 如果源数据中有两个或更多指数部分则置 ON。 如果转换后数据是 + ∞或 - ∞则置 ON。 如果文本数据中为 0 字符则置 ON。 如果在第一个 25 字符范围中没有找到一个包含 00 H 的字节则置 ON。 其余情况下置 OFF。
等于标志	=	如果转换结果为 0 置 ON。 其余情况下置 OFF。

例

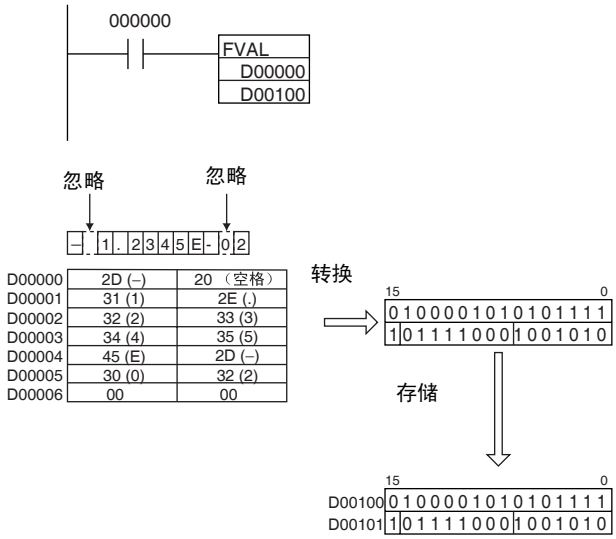
把十进制表示的 ASCII 文本转换为浮点数据

在下例中当 CIO000000 置 ON 时，FVAL(449) 将从 D00000 开始的源字中指定的十进制表示的 ASCII 文本数字转换为浮点数据，并且将结果写入目的字 D00100 和 D00101。



转换科学表示的 ASCII 文本

在下例中当 CIO000000 置 ON 时，FVAL(449) 将从 D00000 开始的源字中指定的科学表示的ASCII文本数字转换为浮点数据，并且将结果写入目标字 D00100 和 D00101。



3-16 双精度浮点数指令（仅限 CS1-H，CJ1-H，CJ1M，或 CS1D）

双精度浮点指令转换数据，并且完成对双精度浮点数据的浮点算术运算。

CS1-H/CJ1-H CPU 单元支持下列 20 条指令。

指令	助记符	函数代码	页数
双精度浮点到 16 位	FIXD	841	576
双精度浮点到 32 位	FIXLD	842	578
16 位到双精度浮点	DBL	843	580
32 位到双精度浮点	DBLL	844	581
双精度浮点加	+D	845	581
双精度浮点减	-D	846	585
双精度浮点乘	*D	847	587

指令	助记符	函数代码	页数
双精度浮点除	/D	848	589
双精度浮点角度到弧度	RADD	849	589
双精度浮点弧度到角度	DEGD	850	593
双精度浮点正弦	SIND	851	593
双精度浮点余弦	COSD	852	596
双精度浮点正切	TAND	853	598
双精度浮点反正弦	ASIND	854	600
双精度浮点反余弦	ACOSD	855	602
双精度浮点反正切	ATAND	856	604
双精度浮点平方根	SQRTD	857	606
双精度浮点指数	EXPD	858	608
双精度浮点对数	LOGD	859	610
双精度浮点指数幂	PWRD	860	612
双精度浮点符号比较指令	LD, AND, OR + =D, <>D, <D, <=D, >D, 或 >=D	335 ～ 340	613

数据格式

浮点数据用一个正负号，指数，和尾数表示实数。当数据被表示为浮点数格式时，可用下列公式。

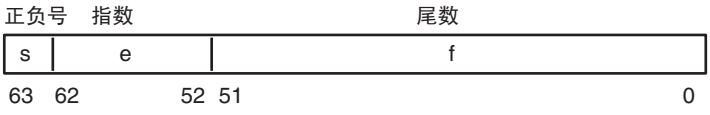
实数 $=(-1)^{s}2^{e-1023}(1.f)$

s：正负号

e：指数

f：尾数

浮点数据格式符合 IEEE754 标准。数据被表示为 64 位，如下示：



数据	位数	内容
s: 正负号	1	0：正； 1：负
e: 指数	11	指数 (e) 值范围从 1 ～ 2047。 实际指数是从 e 中减去 1023 所得的值，所以值范围是从 -1023 ～ 1024。 "e=0" 和 "e=2047" 表示特殊数字。
f: 尾数	52	二进制浮点数据的尾数部分符合格式 2.0>1.f ≥ 1.0。

位数

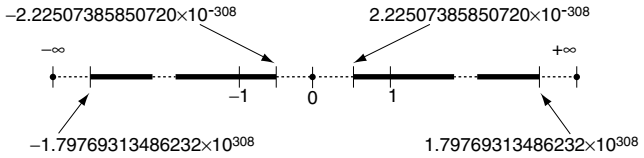
浮点数据的有效位数是二进制 53 位 （相当于十进制 15 位）。

浮点数据

下列数据可由浮点数据表示：

- $-\infty$
- $-1.79769313486232 \times 10^{308} \leqslant$ 值 $\leqslant -2.22507385850720 \times 10^{-308}$
- 0
- $2.22507385850720 \times 10^{-308} \leqslant$ 值 $\leqslant 1.79769313486232 \times 10^{308}$

- +∞
- 非一数字 (NaN)



特殊数字

NaN，±∞，和 0 的格式如下：

NaN*：e=1024 且 f ≠ 0

+ ∞：e=1024, f=0，且 s=0

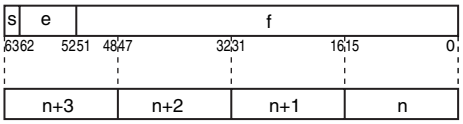
- ∞：e=1024, f=0，且 s=1

0：e=0 且 f=0

*NaN （非一数字）不是一个有效的浮点数字。执行双精度浮点指令不会产生 NaN。

写浮点数据

当 CX-Programmer 中显示的 I/O 内存编辑中的数据格式指定了双精度浮点数，显示出来的标准十进制数字输入自动转换为上面（IEEE754 格式）的双精度浮点格式并写进 I/O 内存。当监视显示时，以 IEEE754 格式写入的数据自动转换为标准十进制格式。



当读和写双精度浮点数据时，用户不用了解 IEEE754 数据格式。只需记住每个双精度浮点数值占 4 个字。

用浮点数值表示数字

下列浮点数字类型可供使用。

尾数 （f）	指数 (e)		
	0	非 0 及非全 1(1024)	全 1(1024)
0	0	标准化数字	无穷大
非 0	非标准化数字		NaN

注 一个非标准化数字是其绝对值小到不能表示为一个标准化数字。非标准化数字有较少的有效位。如果计算结果是一个非标准化数字 （包括中间结果），有效的位数会减少。

标准化数字

标准化数字表示实数。一个正数的正负号位为 0，而一个负数则为 1。

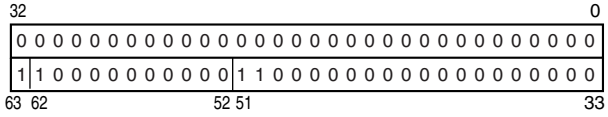
指数 (e) 表示范围 1 ～ 2046，并且实指数则小于等于 1023，也就是，-1022 ～ 1023。

尾数 （f）表示范围 0 ～ （2^{52}-1），并且可以设想，在实尾数中，位 2^{52} 是 1 且它后面立即跟一个小数点。

标准化数字表示如下：

(-1)（正负号 s） × 2（尾数 e） -1023 × （1+ 尾数× 2^{-52}）

例



正负号：-

指数：1024-1023=1

尾数：1+ （$2^{51}+2^{50}$） × 2^{-52}=1+ （$2^{-1}+2^{-2}$） =1+ （0.75） =1.75

值： -1.75 × 2^{1}=-3.5

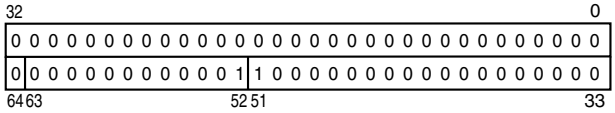
非标准化数字

非标准化数字表示非常小绝对值的实数。一个正数的符号位为 0，而一个负数则为 1。

指数 （e）为 0，并且实指数为 -1022。

尾数 （f）表示范围 1 ～ （2^{52}-1），并且可以设想，在实尾数中，位 2^{52} 是 1 且它后面立即跟一个小数点。

例



正负号：-

指数：-1022

尾数：0+ （$2^{51}+2^{50}$） × 2^{-52}=0+ （$2^{-1}+2^{-2}$） =0+ （0.75） =0.75

值： -0.75 × 2^{-1022}=-1.668805 × 10^{-308}

零

+0.0 和 -0.0 的值可通过设置正符号为 0 或负符号为 1 来表示。指数和尾数都为 0。+0.0 和 -0.0 都等于 0.0。0.0 的正负号所产生的差异可参考下面的浮点算术结果。

无穷大

+ ∞和 - ∞的值可通过设置正符号为 0 或负符号为 1 来表示。指数为 2047（211-1），并且尾数为 0。

NaN

当计算结果，如 0.0/0.0，∞ / ∞，∞ - ∞，不是一个数字或无穷大时，产生 NaN （非一数字）。指数为 255 （28-1），并且尾数不为 0。

注 NaN 的正负号或尾数区域 （除了非 0）没有详细说明。

浮点算术结果

舍入结果

当浮点算术精确结果的位数超过了内部运算表达式的有效位时，可用下面的方法舍入结果。

如果结果接近两个内部浮点表达式中的一个，可使用比较接近的表达式。如果结果在两个内部浮点表达式的中间，结果将被舍去，因此尾数的最末位为 0。

上溢出，下溢出，和非法计算

上溢出是跟据结果的正负号，作为正或负无穷大的输出。下溢出是跟据结果的正负号，作为正或负零的输出。

非法计算将产生 NaN。非法计算包括加无穷大到相反符号一个数字中，从相反符号数字中减去无穷大，把零和无穷大相乘，零除以零，或无穷大除以无穷大。

如果当一个浮点数字转换为一个整数时出现一个上溢出，则结果值可能不正确。

处理特殊值注意事项

处理零，无穷大，和 NaN 应注意下列事项：

- 正零和负零的和是正零。
- 相同符号的零之间的差是正零。
- 如果任意操作数是一个 NaN，结果也是 NaN。
- 比较时正零和负零可看作相等。
- 一个或多个NaN的比较或相等测试对于！=总为真，并且对所有其它指令总为假。

双精度浮点计算结果

当结果的绝对值大于浮点数据所能表示的最大值，上溢出标志将置 ON 并且结果输出±∞。如果结果为正，则输出 + ∞；如果为负，则为 - ∞。

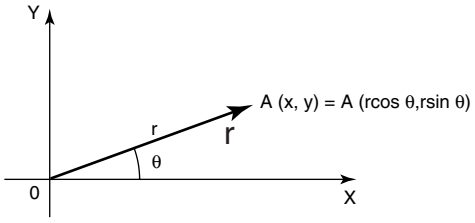
只有当计算后的指数 (e) 和尾数 (f) 都为 0 时，等于标志置为 ON。当结果的绝对值小于浮点数据所能表示的最小值时，一个计算结果也输出为 0。这种情况下，下溢出标志将置 ON。

单精度和双精度计算的比较

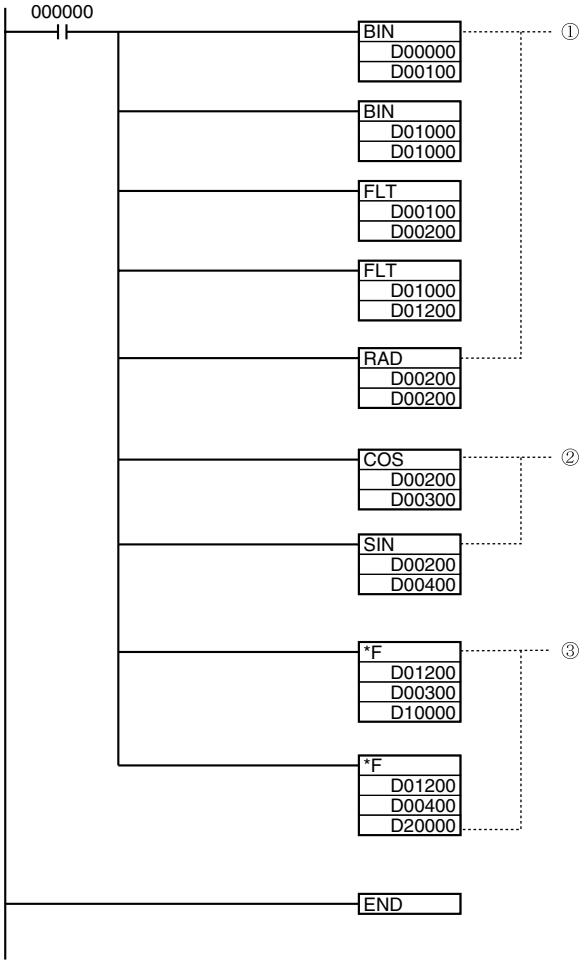
下例说明了当极坐标表示的以下向量转换为直角坐标 A（x， y）时单精度和双精度计算之间的差异。

$$r = re^{j\left(\frac{\pi}{360}\right)\theta}$$

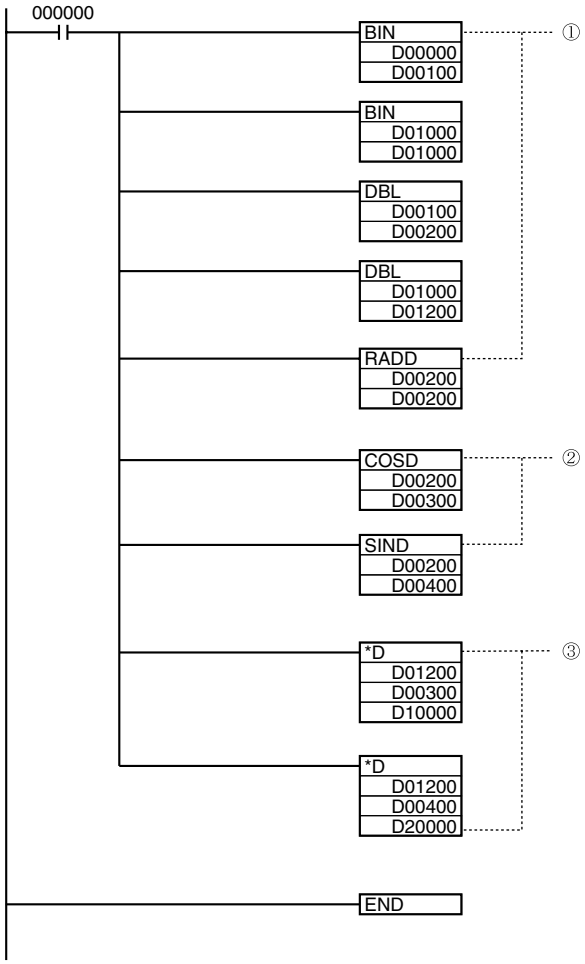
在这个例子中， 4 位 BCD 角 （θ，角度）从 D00000 读出， 4 位 BCD 距离（r）从 D01000 读出。



• 单精度计算梯形图程序



• 双精度计算梯形图程序



1. 这段程序把BCD数据转换为单精度浮点数据（32位， IEEE754 格式）。
   a) BIN(023) 指令把BCD 数据转换为二进制，并且FLT(452)指令把二进制数据转换为单精度浮点数据。
   b) 角 θ 的浮点数据输出到D00200和D00201。
   c) RAD(458) 把 D00200 和 D00201 中的角度数据转换为弧度。
   d) 半径r的浮点数据输出到D01200和D01201。
2. 这段程序计算 sin θ 和cos θ 的单精度浮点数值。
   a) cos θ 值输出到 D00300 和 D00301。
   b) sin θ 值输出到 D00400 和 D00401。
3. 这段程序计算 x （r × cos θ ）和 y(r × sin θ)。
   a) x(r×cos θ)值输出到D10000和D10001。
   b) y(r×sin θ)值输出到D20000和D20001。

坐标	浮点数字	实数
x	4116 59CF	3.4202015399933
y	405A E495	9.3969259262085

1. 段程序把 BCD 数据转换为双精度浮点数据 （64位， IEEE754 格式）。
   a) BIN(023) 指令把BCD 数据转换为二进制，并且 DBL(843) 指令把二进制数据转换为双精度浮点数据。
   b) 角 θ 的浮点数据输出到D00200和D00203。
   c) RADD(849) 把 D00200 和 D00203 中的角度数据转换为弧度。
   d) 半径 r 的浮点数据输出到 D01200 和 D01203
2. 这段程序计算 sin θ 和cos θ 的双精度浮点数值。
   a) cos θ 值输出到 D00300 和 D00303。
   b) sin θ 值输出到 D00400 和 D00403。
3. 这段程序计算 x （r × cos θ ）和 y(r × sin θ)。
   a) x（r×cos θ ）值输出到D10000~D10003。
   b) y(r × sin θ) 值输出到 D20000~D20003。

坐标	浮点数字	实数
x	4022 CB39 E973 5C32	3.4202014332567
y	400B 5C92 91AC 8EEB	9.3969262078591

计算结果的比较

当比较实数结果时，很明显双精度的计算结果更精确些。

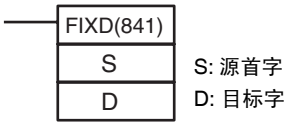
3-16-1 双浮点到 16 位：FIXD(841)

用途

把一个双精度 （64 位）浮点数值转换为 16 位有符号二进制数据，并把结果放在指定的结果字中。

此指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每个循环执行	FIXD(841)
	上升沿微分时执行一次	@FIXD(841)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数规定

区域	S	D
CIO 区	CIO 0000 ～ CIO 6140	CIO 0000 ～ CIO 6143
工作区	W000 ～ W508	W000 ～ W511
保持位区	H000 ～ H508	H000 ～ H511
辅助位区	A000 ～ A956	A448 ～ A959
定时器区	T0000 ～ T4092	T0000 ～ T4095
计数器区	C0000 ～ C4092	C0000 ～ C4095
DM 区	D00000 ～ D32764	D00000 ～ D32767
无区号 EM 区	E00000 ～ E32764	E00000 ～ E32767
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	
数据寄存器	---	DR0 ～ DR15
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

说明

FIXD(841) 把从字 S ～ S+3 （IEEE754 格式）中的双精度 （64 位）浮点数的整数部分转换为 16 位有符号二进制数据，并把结果存入 D 中。

指令仅转换浮点数据的整数部分，小数部分被截去。浮点数据的整数部分必须在 -32768 ～ 32767 之间。

转换例：
浮点数 3.5 转换为 3。
浮点数 -3.5 转换为 -3。

标志

名称	标记	操作
错误标志	ER	如果源数据 （S ～ S+3）不是一个数字 （NaN）则置 ON。 如果源数据 （S ～ S+3）的整数部分不在 -32768 ～ 32767 之间则置 ON。 其余情况下置 OFF。
等于标志	=	如果结果为 0000 置 ON。 其余情况下置 OFF。
负标志	N	如果结果的第 15 位为 ON 则置 ON。 其余情况下置 OFF。

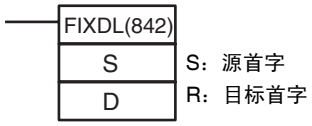
3-16-2 双浮点到 32 位：FIXLD(842)

用途

把一个双精度 （64 位）浮点值转换为 32 位有符号二进制数据，并把结果存入指定的结果字中。

此指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每个循环执行	FIXLD(842)
	上升沿微分时执行一次	@FIXLD(842)
	下降沿微分时执行一次	不支持
立即刷新新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

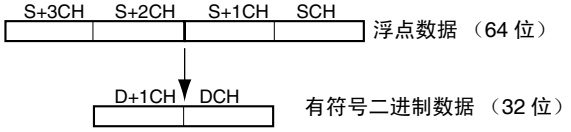
操作数规定

区域	S	D
CIO 区	CIO 0000 ～ CIO 6140	CIO 0000 ～ CIO 6142
工作区	W000 ～ W508	W000 ～ W510

区域	S	D
保持位区	H000 ～ H508	H000 ～ H510
辅助位区	A000 ～ A956	A448 ～ A958
定时器区	T0000 ～ T4092	T0000 ～ T4094
计数器区	C0000 ～ C4092	C0000 ～ C4094
DM 区	D00000 ～ D32764	D00000 ～ D32766
无区号 EM 区	E00000 ～ E32764	E00000 ～ E32766
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	En_00000 ～ En_32766 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	
数据寄存器	---	
索引寄存器	---	
使用索引寄存器 的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

说明

FIXLD(842) 把从字 S ～ S+3（IEEE754 格式）中的双精度（64 位）浮点数的整数部分转换为 32 位有符号二进制数据，并把结果存入 D+1 和 D 中。



指令仅转换浮点数据的整数部分，小数部分被截去。（浮点数据的整数部分必须在 -2147483648 ～ 2147483647 之间）。

转换例：
浮点数 2147483640.5 转换为 2147483640
浮点数 -2147483640.5 转换为 -2147483640

标志

名称	标识	操作
错误标志	ER	如果字 S ～ S+3 中的数据不是一个数字 (NaN) 则置 ON。 如果字 S ～ S+3 中的整数部分不在 -2147483648 ～ 2147483647 之间则置 ON。 其余情况下置 OFF。
等于标志	=	如果结果为 00000000 置 ON。 其余情况下置 OFF。
负标志	N	如果执行后 D+1 的第 15 位为 ON 则置 ON。 其余情况下置 OFF。

注意

字 S ～ S+3 的内容必须是浮点数据且整数部分必须在 -2147483648 ～ 2147483647 范围之内。

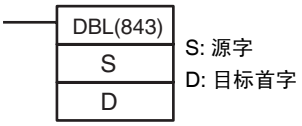
3-16-3 16 位到双浮点：DBL(843)

用途

把一个 16 位有符号二进制数值转换为双精度 （64 位）浮点数据，并把结果放在指定的目的字中。

此指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每个循环执行	DBL(843)
	上升沿微分时执行一次	@DBL(843)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

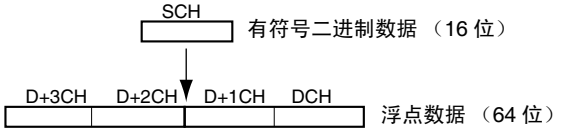
操作数规定

区域	S	D
CIO 区	CIO 0000 ～ CIO 6143	CIO 0000 ～ CIO 6140
工作区	W000 ～ W511	W000 ～ W508
保持位区	H000 ～ H511	H000 ～ H508
辅助位区	A000 ～ A959	A448 ～ A956
定时器区	T0000 ～ T4095	T0000 ～ T4092
计数器区	C0000 ～ C4095	C0000 ～ C4092
DM 区	D00000 ～ D32767	D00000 ～ D32764
无区号 EM 区	E00000 ～ E32767	E00000 ～ E32764
有区号 EM 区	En_00000 ～ En_32767 (n= 0 ～ C)	En_00000 ～ En_32764 (n= 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#0000 ～ #FFFF （二进制）	---
数据寄存器	DR0 ～ DR15	---

区域	S	D
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

说明

DBL(843) 把字 S 中 16 位有符号二进制数据转换为双精度 （64 位）浮点数据（IEEE754 格式）并把结果放进字 D ～ D+3 中。在浮点结果中小数点后需加一个单个的 0。



仅 -32768 ～ 32767 范围之间的数值可由 S 指定。为转换在此范围之外的有符号二进制数据，应使用 DBLL(844)。

转换例：
有符号二进制数值 3 转换为 3.0 。
有符号二进制数值 -3 转换为 -3.0 。

标志

名称	标记	操作
错误标志	ER	OFF
等于标志	=	如果结果的指数和尾数全为 0 时置 ON。 其余情况下置 OFF。
负标志	N	如果结果为负时置 ON。 其余情况下置 OFF。

注意

S 的内容必须包含 -32768 ～ 32767 范围中一个 （十进制）数值的有符号二进制数据。

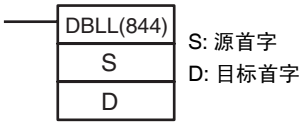
3-16-4 32 位到双浮点：DBLL(844)

用途

把一个 32 位有符号二进制数值转换为双精度 （64 位）浮点数据，并把结果放在指定的目的字中。

此指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持。

梯形图符号



变化

变化		
	ON 条件时每个循环执行	DBLL(844)
	上升沿微分时执行一次	@DBLL(844)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

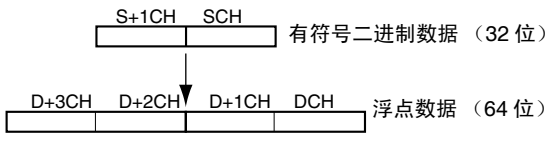
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数规定

区域	S	D
CIO 区	CIO 0000 ～ CIO 6142	CIO 0000 ～ CIO 6140
工作区	W000 ～ W510	W000 ～ W508
保持位区	H000 ～ H510	H000 ～ H508
辅助位区	A000 ～ A958	A448 ～ A956
定时器区	T0000 ～ T4094	T0000 ～ T4092
计数器区	C0000 ～ C4094	C0000 ～ C4092
DM 区	D00000 ～ D32766	D00000 ～ D32764
无区号 EM 区	E00000 ～ E32766	E00000 ～ E32764
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	En_00000 ～ En_32764 (n= 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#00000000 ～ #FFFFFFFF （二进制）	---
数据寄存器	---	
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15	

描述

DBLL(844) 把字 S+1 和 S 中 32 位有符号二进制数据转换为双精度（64 位）浮点数据 （IEEE754 格式）并把结果放进字 D ～ D+3 中。在浮点结果中小数点后需加一个单个的 0 。



S+1 和 S 可表示在 -2147483648 ～ 2147483647 范围内的有符号二进制数据。浮点数值有 24 个有效的二进制数位 （位）。如果 DBLL(844) 转换一个大于 16777215 （能用 24 位表示的最大数值）的数字，其结果将不准确。

转换例：

一个有符号二进制数值 16777215 转换为 16777215.0 。

一个有符号二进制数值 -16777215 转换为 -16777215.0 。

标志

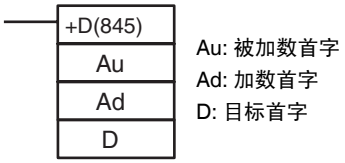
名称	标记	操作
错误标志	ER	OFF
等于标志	=	如果结果的指数和尾数全为 0 时置 ON。 其余情况下置 OFF。
负标志	N	如果结果为负时置 ON。 其余情况下置 OFF。

注意 如果转换一个绝对值大于 16777215（能用 24 位表示的最大数值）的数字，其结果将不准确。

3-16-5 双浮点加：+D(845)

用途 把两个双精度 （64 位）浮点数字相加，并把结果放在指定的目的字中。
此指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每个循环执行	+D(845)
	上升沿微分时执行一次	@+D(845)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

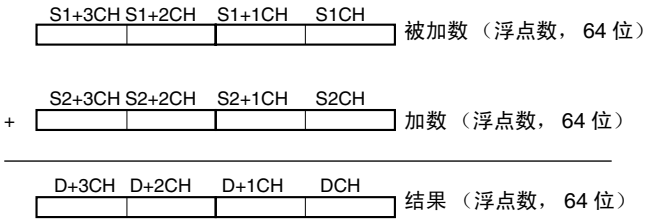
操作数规定

区域	Au	Ad	D
CIO 区	CIO 0000 ～ CIO 6140		
工作区	W000 ～ W508		
保持位区	H000 ～ H508		
辅助位区	A000 ～ A956		A448 ～ A956
定时器区	T0000 ～ T4092		
计数器区	C0000 ～ C4092		
DM 区	D00000 ～ D32764		
无区号 EM 区	E00000 ～ E32764		
有区号 EM 区	En_00000 ～ En_32764 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		

区域	Au	Ad	D
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---		
数据寄存器	---		
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15		

描述

+D(845) 把字 Ad 到 Ad+3 中的双精度（64 位）浮点数加到字 Au ～ Au+3 中的双精度 （64 位）浮点数中，并把结果放进字 D ～ D+3 中。（浮点数必须是 IEEE754 格式）



如果结果的绝对值大于可用浮点数表示的最大值，上溢出标志将置 ON，并且结果将以±∞输出。

如果结果的绝对值小于可用浮点数表示的最小值，下溢出标志将置 ON，并且结果将以 0 输出。

被加数和加数的各种组合将产生下表所示的结果。

	被加数				
加数	0	数字	+∞	–∞	NaN
0	0	数字	+∞	–∞	
数字	数字	见注 1	+∞	–∞	
+∞	+∞	+∞	+∞	见注 2	
–∞	–∞	–∞	见注 2	–∞	
NaN					见注 2

注 1. 结果可能为 0 （包括下溢出），一个数字， + ∞或 - ∞。

2. 错误标志将置 ON，并且指令不会被执行。

标志

名称	标记	操作
错误标志	ER	如果被加数或加数不是浮点数则置 ON。 如果被加数或加数不是一个数字 （NaN）则置 ON。 + ∞和 - ∞相加时置 ON。 其余情况下置 OFF。
等于标志	=	结果的指数和尾数都为 0 则置 ON。 其余情况下置 OFF。
上溢出标志	OF	如果结果的绝对值太大而不能表示为一个双精度浮点数值，则置 ON。
下溢出标志	UF	如果结果的绝对值太小而不能表示为一个双精度浮点数值，则置 ON。
负标志	N	结果为负时置 ON。 其余情况下置 OFF。

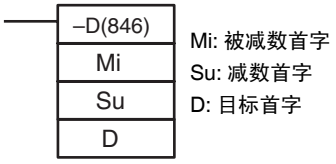
注意 被加数 （Au ～ Au+3）和加数 （Ad ～ Ad+3）必须是 IEEE754 浮点数格式。

3-16-6 双浮点减：*D(846)

用途 从一个双精度 （64 位）浮点数中减去另一个双精度 （64 位）浮点数，并把结果放在指定的目的字中。

此指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每个循环执行	–D(846)
	上升沿微分时执行一次	@–D(846)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

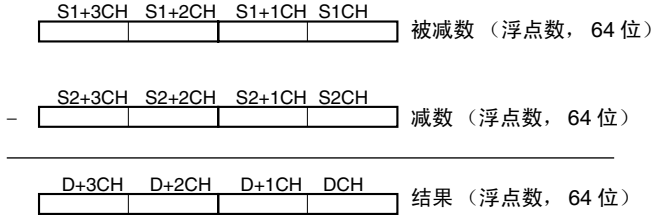
操作数规定

区域	Mi	Su	D
CIO 区	CIO 0000 ～ CIO 6140		
工作区	W000 ～ W508		
保持位区	H000 ～ H508		
辅助位区	A000 ～ A956		A448 ～ A956
定时器区	T0000 ～ T4092		
计数器区	C0000 ～ C4092		
DM 区	D00000 ～ D32764		
无区号 EM 区	E00000 ～ E32764		

区域	Mi	Su	D
有区号 EM 区	En_00000 ～ En_32764 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---		
数据寄存器	---		
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15		

描述

-D(846) 从 Mi ～ Mi+3 中的双精度 （64 位）浮点数中减去字 Su ～ Su+3 中的双精度 （64 位）浮点数，并把结果放在字 D ～ D+3 中。（浮点数据必须是 IEEE754 格式）



如果结果的绝对值大于可用浮点数表示的最大值，上溢出标志将置 ON，并且结果将以±∞输出。

如果结果的绝对值小于可用浮点数表示的最小值，下溢出标志将置 ON，并且结果将以 0 输出。

被减数和减数的各种组合将产生下表所示的结果。

	被减数				
减数	0	数字	+∞	–∞	NaN
0	0	数字	+∞	–∞	
数字	数字	见注 1	+∞	–∞	
+∞	–∞	–∞	见注 2	–∞	
–∞	+∞	+∞	+∞	见注 2	
NaN					见注 2

注 1. 结果可能为 0 （包括下溢出），一个数字， + ∞，或 - ∞。

2. 错误标志将置 ON，并且指令不会被执行。

标志

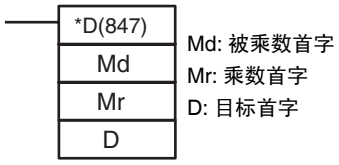
名称	标记	操作
错误标志	ER	如果被减数或减数不是浮点数则置 ON。 如果被减数或减数不是一个数字 （NaN）则置 ON。 + ∞减去 - ∞时置 ON。 - ∞减去 - ∞时置 ON。 其余情况下置 OFF。
等于标志	=	结果的指数和尾数都为 0 则置 ON。 其余情况下置 OFF。
上溢出标志	OF	如果结果的绝对值太大，而不能表示为一个双精度浮点数值，则置 ON。
下溢出标志	UF	如果结果的绝对值太小，而不能表示为一个双精度浮点数值，则置 ON。
负标志	N	结果为负时置 ON。 其余情况下置 OFF。

注意 被减数 (Mi ～ Mi+3) 和减数 (Su ～ Su+3) 必须是 IEEE754 浮点数格式。

3-16-7 双浮点乘 *D(847)

用途 两个双精度 （64 位）浮点数相乘，并把结果放在指定的结果字中。
此指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每个循环执行	*D(847)
	上升沿微分时执行一次	@*D(847)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

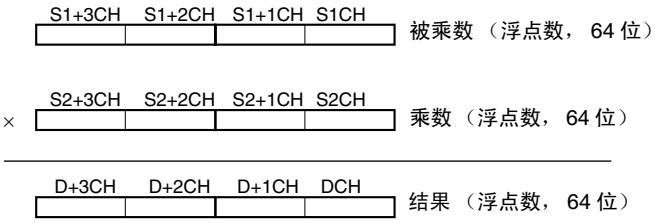
操作数定义

区域	Md	Mr	D
CIO 区	CIO 0000 ～ CIO 6140		
工作区	W000 ～ W508		
保持位区	H000 ～ H508		
辅助位区	A000 ～ A956		A448 ～ A956
定时器区	T0000 ～ T4092		
计数器区	C0000 ～ C4092		
DM 区	D00000 ～ D32764		

区域	Md	Mr	D
无区号 EM 区	E00000 ～ E32764		
有区号 EM 区	En_00000 ～ En_32764 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---		
数据寄存器	---		
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15		

描述

* D(847) 将字 Md ～ Md+3 中的双精度 （64 位）浮点数与字 Mr ～ Mr+3 中的双精度 （64 位）浮点数相乘，并把结果放进字 D ～ D+3 中。（浮点数必须是 IEEE754 格式）



如果结果的绝对值大于可用浮点数表示的最大值，上溢出标志将置 ON，并且结果将以±∞输出。

如果结果的绝对值小于可用浮点数表示的最小值，下溢出标志将置 ON，并且结果将以 0 输出。

被乘数和乘数的各种组合将产生下表所示的结果。

	被乘数				
乘数	0	数字	+∞	–∞	NaN
0	0	0	见注 2	见注 2	
数字	0	见注 1	+/–∞	+/–∞	
+∞	见注 2	+/–∞	+∞	–∞	
–∞	见注 2	+/–∞	–∞	+∞	
NaN					见注 2

注

1. 结果可能为 0 （包括下溢出），一个数字， + ∞，或 - ∞。
2. 错误标志将置 ON，并且指令不会被执行。

标志

名称	标记	操作
错误标志	ER	如果被乘数或乘数不是浮点数则置 ON。 如果被乘数或乘数不是一个数字 （NaN）则置 ON。 + ∞和 0 相乘时置 ON。 - ∞和 0 相乘时置 ON。 其余情况下置 OFF。
等于标志	=	如果结果的指数和尾数都为 0 则置 ON。 其余情况下置 OFF。
上溢出标志	OF	如果结果的绝对值太大，而不能表示为一个双精度浮点数值，则置 ON。
下溢出标志	UF	如果结果的绝对值太小，而不能表示为一个双精度浮点数值，则置 ON。
负标志	N	结果为负时置 ON。 其余情况下置 OFF。

注意 被乘数 （Md ～ Md+3）和乘数 （Mr ～ Mr+3）必须是 IEEE754 浮点数格式。

3-16-8 双浮点除：/D(848)

用途 一个双精度 （64 位）浮点数除以另一个双精度 （64 位）浮点数，并把结果放在指定的目标字中。

此指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持。

梯形图符号

变化

变化	ON 条件时每个循环执行	/D(848)
	上升沿微分时执行一次	@/D(848)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

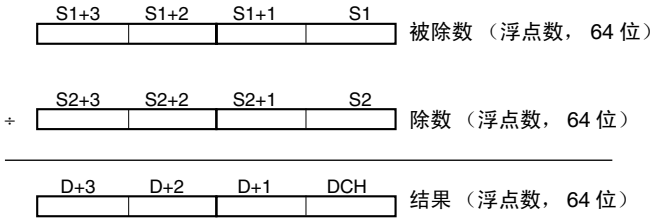
操作数规定

区域	Dd	Dr	D
CIO 区	CIO 0000 ～ CIO 6140		
工作区	W000 ～ W508		
保持位区	H000 ～ H508		
辅助位区	A000 ～ A956		A448 ～ A956
定时器区	T0000 ～ T4092		
计数器区	C0000 ～ C4092		
DM 区	D00000 ～ D32764		

区域	Dd	Dr	D
无区号 EM 区	E00000 ～ E32764		
有区号 EM 区	En_00000 ～ En_32764 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---		
数据寄存器	---		
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15		

说明

/ D(848) 把字 Dd ～ Dd+3 中的双精度 （64 位）浮点数除以字 Dr ～ Dr+3 中的双精度（64 位）浮点数，并把结果放进字 D ～ D+3 中。（浮点数必须是 IEEE754 格式）



如果结果的绝对值大于可用浮点数表示的最大值，上溢出标志将置 ON，并且结果将以±∞输出。

如果结果的绝对值小于可用浮点数表示的最小值，下溢出标志将置 ON，并且结果将以 0 输出。

被除数和除数的各种组合将产生下表所示的结果。

	被除数				
除数	0	数字	+∞	–∞	NaN
0	见注 3	+/–∞	+∞	–∞	
数字	0	见注 1	+/–∞	+/–∞	
+∞	0	见注 2	见注 3	见注 3	
–∞	0	见注 2	见注 3	见注 3	
NaN					见注 3

注

1. 结果可能为 0 （包括下溢出），一个数字， + ∞或 - ∞。
2. 对下溢出结果将为 0 。
3. 错误标志将置 ON，并且指令不会被执行。

标志

名称	标记	操作
错误标志	ER	如果被除数或除数不是浮点数则置 ON。 如果被除数或除数不是一个数字 （NaN）则置 ON。 被除数和除数都为 0 时置 ON。 被除数和除数都为 + ∞ /- ∞时置 ON。 其余情况下置 OFF。
等于标志	=	如果结果的指数和尾数都为 0 则置 ON。 其余情况下置 OFF。
上溢出标志	OF	如果结果的绝对值太大，而不能表示为一个双精度浮点数值，则置 ON。
下溢出标志	UF	如果结果的绝对值太小，而不能表示为一个双精度浮点数值，则置 ON。
负标志	N	结果为负时置 ON。 其余情况下置 OFF。

注意 被除数 （Dd ～ Dd+3）和除数 （Dr 和 Dr+3）必须是 IEEE754 浮点数格式。

3-16-9 双角度到弧度：RADD(849)

用途 把一个双精度（64 位）浮点数从角度转换为弧度，并把结果放在指定的结果字中。

此指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持。

梯形图符号

变化

变化	ON 条件时每个循环执行	RADD(849)
	上升沿微分时执行一次	@RADD(849)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

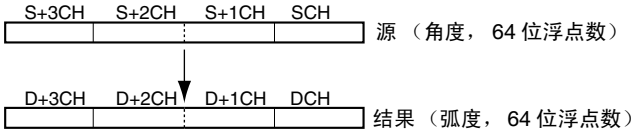
操作数规定

区域	S	D
CIO 区	CIO 0000 ～ CIO 6140	
工作区	W000 ～ W508	
保持位区	H000 ～ H508	
辅助位区	A000 ～ A956	A448 ～ A956
定时器区	T0000 ～ T4092	
计数器区	C0000 ～ C4092	
DM 区	D00000 ～ D32764	
无区号 EM 区	E00000 ～ E32764	

区域	S	D
有区号 EM 区	En_00000 ～ En_32764 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	
数据寄存器	---	
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15	

说明

RADD(849) 把字 S ～ S+3 中的双精度 （64 位）浮点数从角度转换为弧度，并把结果放在字 D ～ D+3 中。（源浮点数必须表示为 IEEE754 格式）



用下列公式把角度转换为弧度：

角度 × π/180 = 弧度

如果结果的绝对值大于可用浮点数表示的最大值，上溢出标志将置 ON，并且结果将以 ± ∞输出。

如果结果的绝对值小于可用浮点数表示的最小值，下溢出标志将置 ON，并且结果将以 0 输出。

标志

名称	标记	操作
错误标志	ER	如果源数据不是浮点数则置 ON。 如果源数据不是一个数 （NaN），则置 ON。 其余情况下置 OFF。
等于标志	=	如果结果的指数和尾数都为 0 则置 ON。 其余情况下置 OFF。
上溢出标志	OF	如果结果的绝对值太大，而不能表示为一个双精度浮点值，则置 ON。
下溢出标志	UF	如果结果的绝对值太小，而不能表示为一个双精度浮点值，则置 ON。
负标志	N	结果为负时置 ON。 其余情况下置 OFF。

注意

字 S 和 S+3 中的源数据必须是 IEEE754 浮点数格式。

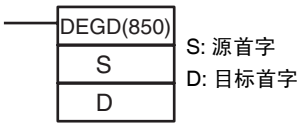
3-16-10 双弧度到角度：DEGD(850)

用途

把一个双精度（64 位）浮点数从弧度转换为角度，并把结果放在指定的结果字中。

此指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每个循环执行	DEGD(850)
	上升沿微分时执行一次	@DEGD(850)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

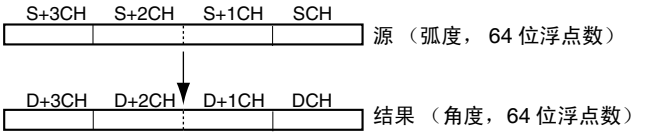
操作数规定

区域	S	D
CIO 区	CIO 0000 ～ CIO 6140	
工作区	W000 ～ W508	
保持位区	H000 ～ H508	
辅助位区	A000 ～ A956	A448 ～ A956
定时器区	T0000 ～ T4092	
计数器区	C0000 ～ C4092	
DM 区	D00000 ～ D32764	
无区号 EM 区	E00000 ～ E32764	
有区号 EM 区	En_00000 ～ En_32764 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	
数据寄存器	---	

区域	S	D
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15	

说明

DEGD(850) 把字 S ～ S+3 中的双精度 （64 位）浮点数从弧度转换为角度，并把结果放在字 D ～ D+3 中。（源浮点数必须表示为 IEEE754 格式）



用下列公式把弧度转换为角度：

弧度 × 180/π = 角度

如果结果的绝对值大于可用浮点数表示的最大值，上溢出标志将置 ON，并且结果将以 ± ∞输出。

如果结果的绝对值小于可用浮点数表示的最小值，下溢出标志将置 ON，并且结果将以 0 输出。

标志

名称	标记	操作
错误标志	ER	如果源数据不是浮点数则置 ON。 如果源数据不是一个数字 （NaN），则置 ON。 其余情况下置 OFF。
等于标志	=	如果结果的指数和尾数都为 0 则置 ON。 其余情况下置 OFF。
上溢出标志	OF	如果结果的绝对值太大，而不能表示为一个双精度浮点值，则置 ON。
下溢出标志	UF	如果结果的绝对值太小，而不能表示为一个双精度浮点值，则置 ON。
负标志	N	结果为负时置 ON。 其余情况下置 OFF。

注意

字 S ～ S+3 中的源数据必须是 IEEE754 浮点数格式。

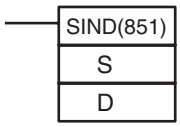
3-16-11 双正弦：SIND(851)

用途

计算一个双精度 （64 位）浮点数 （用弧度表示）的正弦，并把结果放在指定的目标字中。

此指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每个循环执行	SIND(851)
	上升沿微分时执行一次	@SIND(851)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数规定

区域	S	D
CIO 区	CIO 0000 ～ CIO 6140	
工作区	W000 ～ W508	
保持位区	H000 ～ H508	
辅助位区	A000 ～ A956	A448 ～ A956
定时器区	T0000 ～ T4092	
计数器区	C0000 ～ C4092	
DM 区	D00000 ～ D32764	
无区号 EM 区	E00000 ～ E32764	
有区号 EM 区	En_00000 ～ En_32764 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	
数据寄存器	---	
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15	

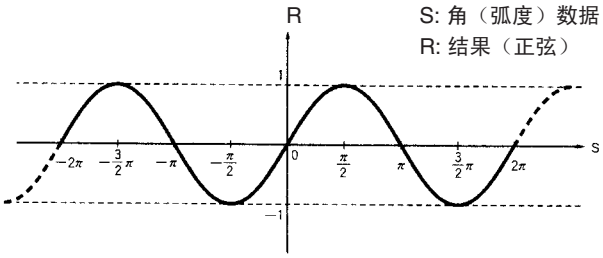
说明

SIND (851) 计算字 S ～ S+3 中用一个双精度 （64 位）浮点数表示的角 （弧度）的正弦，并把结果放在字 D ～ D+3 中。（源浮点数必须表示为 IEEE754 格式）

SIN(S+3 | S+2 | S+1 | S) → D+3 | D+2 | D+1 | D

在字 S ～ S+3 中用弧度定义所求的角 (-65535 ～ 65535)。如果角的绝对值超过 65535，将会出现错误，并且指令不执行。从角度转换为弧度的信息，参阅 *3-16-9 双角度到弧度：RADD (849)* 或 *3-16-10 双弧度到角度：DEGD (850)*。

下图显示角和结果之间的关系。



标志

名称	标记	操作
错误标志	ER	如果源数据不是一个数字 （NaN）则置 ON。 如果源数据的绝对值超过 65535，则置 ON。 其余情况下置 OFF。
等于标志	=	如果结果的指数和尾数都为 0 则置 ON。 其余情况下置 OFF。
上溢出标志	OF	不变
下溢出标志	UF	不变
负标志	N	结果为负时置 ON。 其余情况下置 OFF。

注意

字 S ～ S+3 中的源数据必须是 IEEE754 浮点数格式。

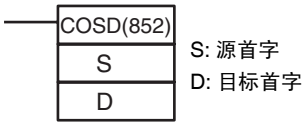
3-16-12 双余弦：COSD(852)

用途

计算一个双精度 （64 位）浮点数 （弧度）的余弦，并把结果放在指定的目标字中。

此指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每个循环执行	COSD(852)
	上升沿微分时执行一次	@COSD(852)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

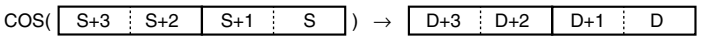
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数规定

区域	S	D
CIO 区	CIO 0000 ～ CIO 6140	
工作区	W000 ～ W508	
保持位区	H000 ～ H508	
辅助位区	A000 ～ A956	A448 ～ A956
定时器区	T0000 ～ T4092	
计数器区	C0000 ～ C4092	
DM 区	D00000 ～ D32764	
无区号 EM 区	E00000 ～ E32764	
有区号 EM 区	En_00000 ～ En_32764 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	
数据寄存器	---	
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15	

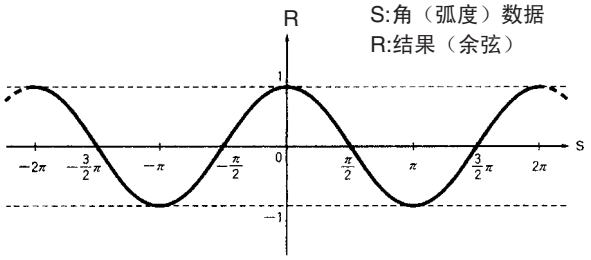
说明

COSD(852) 计算字 S ～ S+3 中用一个双精度 （64 位）浮点数表示的角 （弧度）的余弦，并把结果放在字 D 到 D+3 中。
（浮点源数据必须是 IEEE754 格式）



在字 S ～ S+3 中用弧度指定所求的角 (-65535 ～ 65535)。如果角的绝对值超过 -65535 到 65535 的范围，将产生错误，并且指令不执行。从角度转换为弧度的信息，参阅 *3-16-9 双角度到弧度：RADD(849)* 或 *3-16-10 双弧度到角度：DEGD(850)*。

下图显示角和结果之间的关系。



标志

名称	标记	操作数
错误标志	ER	如果源数据不是一个数字 （NaN）则置 ON。 如果源数据的绝对值超过 65535，则置 ON。 其余情况下置 OFF。
等于标志	=	如果结果的指数和尾数都为 0 则置 ON。 其余情况下置 OFF。
上溢出标志	OF	不变
下溢出标志	UF	不变
负标志	N	结果为负时置 ON。 其余情况下置 OFF。

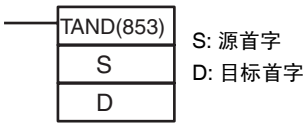
注意 字 S ～ S+3 中的源数据必须是 IEEE754 浮点数格式。

3-16-13 双正切：TAND(853)

用途 计算一个双精度 （64 位）浮点数 （弧度）的正切，并把结果放在指定的目标字中。

此指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每个循环执行	TAND(853)
	上升沿微分时执行一次	@TAND(853)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数规定

区域	S	D
CIO 区	CIO 0000 ～ CIO 6140	
工作区	W000 ～ W508	
保持位区	H000 ～ H508	
辅助位区	A000 ～ A956	A448 ～ A956
定时器区	T0000 ～ T4092	
计数器区	C0000 ～ C4092	
DM 区	D00000 ～ D32764	
无区号 EM 区	E00000 ～ E32764	
有区号 EM 区	En_00000 ～ En_32764 (n = 0 ～ C)	

区域	S	D
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	
数据寄存器	---	
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15	

说明

TAND(853) 计算字 S ～ S+3 中用一个双精度 （64 位）浮点数表示的角 （弧度）的正切，并把结果放在字 D ～ D+3 中。

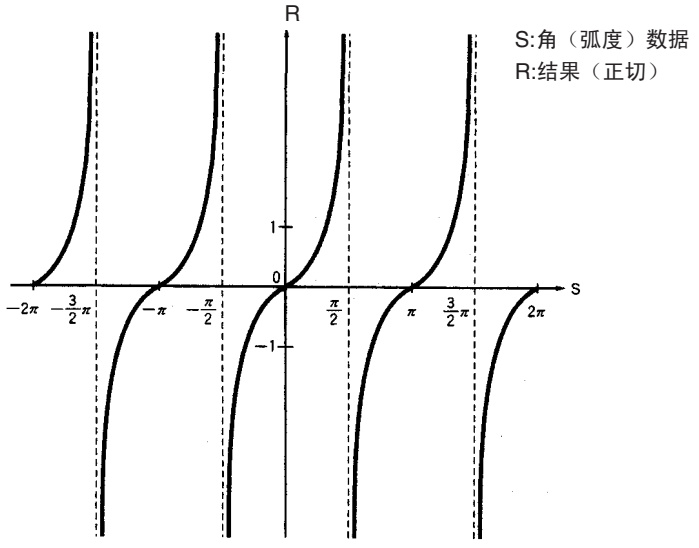
（浮点源数据必须是 IEEE754 格式）

TAN(S+3 | S+2 | S+1 | S) → D+3 | D+2 | D+1 | D

在字 S ～ S+3 中用弧度指定所求角 (-65535 ～ 65535) 。如果角的绝对值超过 -65535 ～ 65535 的范围，将产生错误，并且指令不执行。从角度转换为弧度的信息，参阅 *3-16-9 双角度到弧度：RADD(849)* 或 *3-16-10 双弧度到角度：DEGD(850)* 。

如果结果的绝对值大于可用浮点数表示的最大值，上溢出标志将置 ON，并且结果将以±∞输出。

下图显示角和结果之间的关系。



标志

名称	标记	操作
错误标志	ER	如果源数据不是一个数字 （NaN）则置 ON。 如果源数据的绝对值超过 65535，则置 ON。 其余情况下置 OFF。
等于标志	=	如果结果的指数和尾数都为 0 则置 ON。 其余情况下置 OFF。
上溢出标志	OF	如果结果的绝对值太大，而不能表示为一个双精度 （64 位）浮点值，则置 ON。
下溢出标志	UF	不变
负标志	N	结果为负时置 ON。 其余情况下置 OFF。

注意

字 S ～ S+3 中的源数据必须是 IEEE754 浮点数格式。

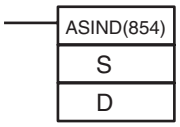
3-16-14 双反正弦：ASIND(854)

用途

计算一个双精度 （64 位）浮点数的反正弦，并把结果放在指定的目标字中。（反正弦函数是正弦函数的反函数；它返回所给的正弦值为 -1 到 1 之间的角）。

此指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每个循环执行	ASIND(854)
	上升沿微分时执行一次	@ASIND(854)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

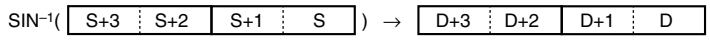
操作数规定

区域	S	D
CIO 区	CIO 0000 ～ CIO 6140	
工作区	W000 ～ W508	
保持位区	H000 ～ H508	
辅助位区	A000 ～ A956	A448 ～ A956
定时器区	T0000 ～ T4092	
计数器区	C0000 ～ C4092	
DM 区	D00000 ～ D32764	
无区号 EM 区	E00000 ～ E32764	

区域	S	D
有区号 EM 区	En_00000 ～ En_32764 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	
数据寄存器	---	
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15	

说明

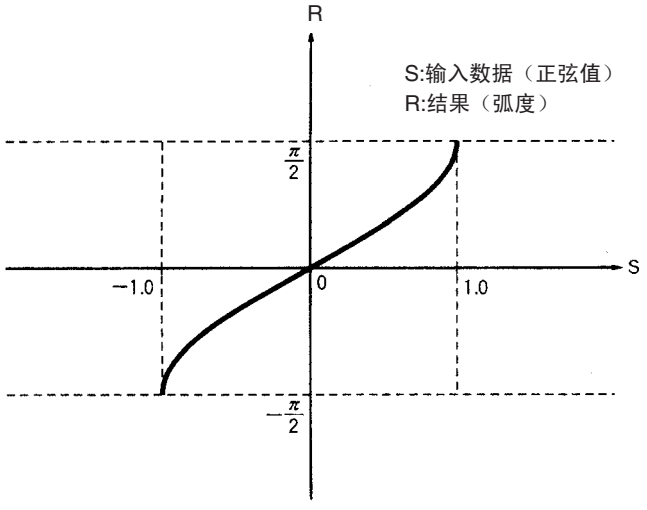
ASIND(854) 计算字 S ～ S+3 中用一个双精度（64 位）浮点数表示正弦值的角（弧度），并把结果放在字 D ～ D+3 中。（浮点源数据必须是 IEEE754 格式）



源数据必须在 -1.0 和 1.0 之间。如果源数据的绝对值超过 1.0，将会出现错误，并且指令不执行。

结果以从 - π /2 ～ π /2 之间的角 （弧度）形式输出到字 D ～ D+3。

下图显示了输入数据和结果之间的关系。



标志

名称	标记	操作
错误标志	ER	如果源数据不是浮点数则置 ON。 如果源数据不是一个数字 （NaN）则置 ON。 如果源数据的绝对值超过 1.0 则置 ON。 其余情况下置 OFF。
等于标志	=	如果结果的指数和尾数都为 0 则置 ON。 其余情况下置 OFF。
上溢出标志	OF	不变
下溢出标志	UF	不变
负标志	N	结果为负时置 ON。 其余情况下置 OFF。

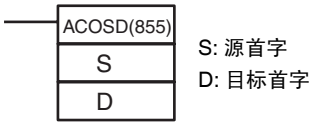
注意 字 S ～ S+3 中的源数据必须是 IEEE754 浮点数格式。

3-16-15 反余弦：ACOSD(855)

用途 计算一个双精度（64 位）浮点数的反余弦，并把结果放在指定的结果字中（反余弦函数是余弦函数的反函数；它返回所给的余弦值为 -1 到 1 之间的角）。

此指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每个循环执行	ACOSD(855)
	上升沿微分时执行一次	@ACOSD(855)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数规定

区域	S	D
CIO 区	CIO 0000 ～ CIO 6140	
工作区	W000 ～ W508	
保持位区	H000 ～ H508	
辅助位区	A000 ～ A956	A448 ～ A956
定时器区	T0000 ～ T4092	
计数器区	C0000 ～ C4092	
DM 区	D00000 ～ D32764	
无区号 EM 区	E00000 ～ E32764	

区域	S	D
有区号 EM 区	En_00000 ～ En_32764 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	
数据寄存器	---	
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15	

说明

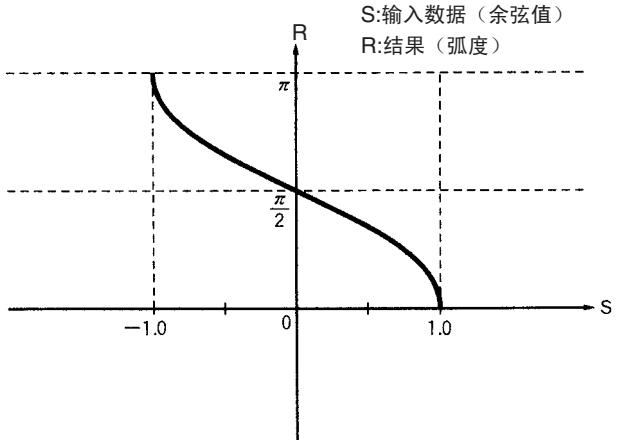
ACOSD(855) 计算字 S ～ S+3 中余弦值表达为一个双精度 （64 位）浮点数的角（弧度），并把结果放在字 D ～ D+3 中。（浮点源数据必须是 IEEE754 格式）

COS^{-1}(S+3 | S+2 | S+1 | S) → D+3 | D+2 | D+1 | D

源数据必须在 -1.0 和 1.0 之间。如果源数据的绝对值超过 1.0，将会出现错误，并且指令不执行。

结果以从 0 ～ π 范围内的角 （弧度）形式输出到字 D ～ D+3 中。

下图显示了输入数据和结果之间的关系。



标志

名称	标记	操作
错误标志	ER	如果源数据不是浮点数据，则置 ON。 如果源数据不是一个数字 （NaN）则置 ON。 如果源数据的绝对值超过 1.0 则置 ON。 其余情况下置 OFF。
等于标志	=	如果结果的指数和尾数都为 0 则置 ON。 其余情况下置 OFF。
上溢出标志	OF	不变
下溢出标志	UF	不变
负标志	N	不变

注意

字 S ～ S+3 中的源数据必须是 IEEE754 浮点数格式。

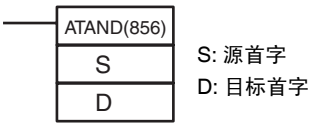
3-16-16 双反正切：ATAND(856)

用途

计算一个双精度 （64 位）浮点数的反正切，并把结果放在指定的结果字中。（反正切函数是正切函数的反函数；它返回产生所给的正切值的角）。

此指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每个循环执行	ATAND(856)
	上升沿微分时执行一次	@ATAND(856)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

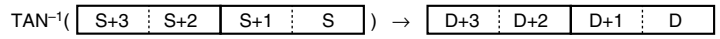
操作数规定

区域	S	D
CIO 区	CIO 0000 ～ CIO 6140	
工作区	W000 ～ W508	
保持位区	H000 ～ H508	
辅助位区	A000 ～ A956	A448 ～ A956
定时器区	T0000 ～ T4092	
计数器区	C0000 ～ C4092	
DM 区	D00000 ～ D32764	
无区号 EM 区	E00000 ～ E32764	
有区号 EM 区	En_00000 ～ En_32764 (n = 0 ～ C)	

区域	S	D
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	
数据寄存器	---	
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15	

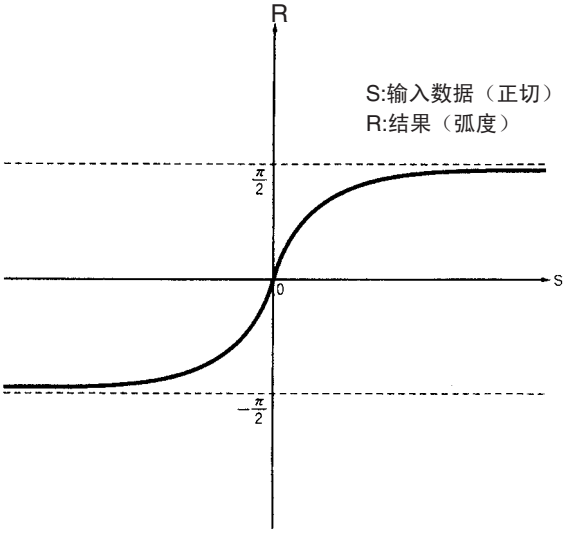
说明

ATAND(856) 计算角 （用弧度表示），它的正切值在字 S ～ S+3 中以一个双精度 （64 位）浮点数表示，并把结果放在字 D ～ D+3 中。（浮点源数据必须是 IEEE754 格式）



结果以在 - π /2 ～π /2 的范围内的角度 （用弧度表示）的形式输出到字 D ～ D+3 中。

下图显示了输入数据和结果之间的关系。



标志

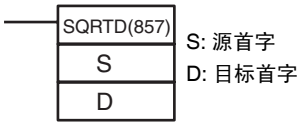
名称	标记	操作
错误标志	ER	如果源数据不是浮点数据，则置 ON。 如果源数据不是一个数字 （NaN）则置 ON。 其余情况下置 OFF。
等于标志	=	如果结果的指数和尾数都为 0 则置 ON。 其余情况下置 OFF。
上溢出标志	OF	不变
下溢出标志	UF	不变
负标志	N	结果为负时置 ON。 其余情况下置 OFF。

注意 字 S ～ S+3 中的源数据必须是 IEEE754 浮点数格式。

3-16-17 双平方根：SQRTD(857)

用途 计算一个双精度 （64 位）浮点数的平方根，并把结果放在指定的结果字中。此指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每个循环执行	SQRTD(857)
	上升沿微分时执行一次	@SQRTD(857)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

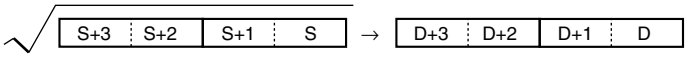
操作数规定

区域	S	D
CIO 区	CIO 0000 ～ CIO 6140	
工作区	W000 ～ W508	
保持位区	H000 ～ H508	
辅助位区	A000 ～ A956	A448 ～ A956
定时器区	T0000 ～ T4092	
计数器区	C0000 ～ C4092	
DM 区	D00000 ～ D32764	
无区号 EM 区	E00000 ～ E32764	
有区号 EM 区	En_00000 ～ En_32764 (n = 0 ～ C)	

区域	S	D
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	
数据寄存器	---	
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15	

说明

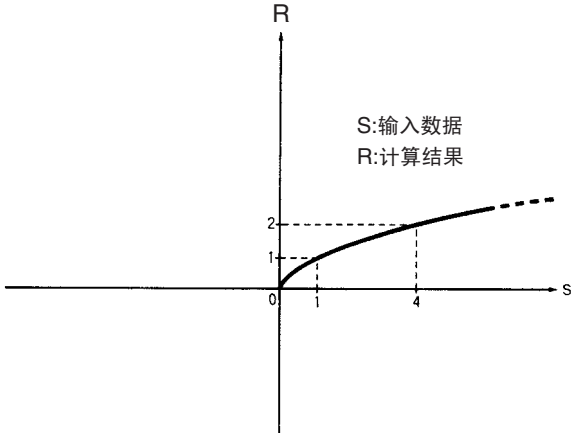
SQRTD(857) 计算在字 S ～ S+3 中的一个双精度 （64 位）浮点数的平方根，并将计算结果存入字 D ～ D+3 中。（浮点源数据必须是 IEEE754 格式）



源数据必须为正，如为负则会产生一个错误，并且指令不执行。

如果计算结果的绝对值大于浮点数所能表示的最大值，上溢出标志置为 ON，计算结果输出为 ±∞。

下图为输入数据和计算结果的关系图。



标志

名称	标记	操作
错误标志	ER	源数据不能被识别为浮点数时置 ON。 源数据为负时则置 ON。 源数据非数字 （NaN）时置 ON。 其余情况下置 OFF。
等于标志	=	计算结果的指数和尾数全为 0 置 ON。 其余情况下置 OFF。
上溢出标志	OF	计算结果的绝对值太大超出一个双精度 （64 位）浮点数表示范围时置 ON。
下溢出标志	UF	不变
负标志	N	不变

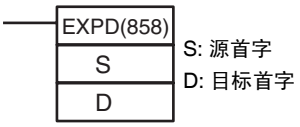
注意 表示为字 S ～ S+3 的源数据必须是 IEEE754 浮点数格式。

3-16-18 双指数：EXPD(858)

用途 计算一个双精度 （64 位）浮点数的自然指数 （底为 e），并把结果存入指定的结果字中。

此指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每个循环执行	EXPD(858)
	上升沿微分时执行一次	@EXPD(858)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

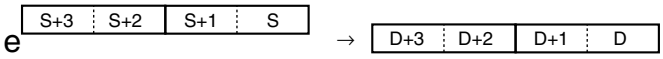
操作数规定

区域	S	D
CIO 区	CIO 0000 ～ CIO 6140	
工作区	W000 ～ W508	
保持位区	H000 ～ H508	
辅助位区	A000 ～ A956	A448 ～ A956
定时器区	T0000 ～ T4092	
计数器区	C0000 ～ C4092	
DM 区	D00000 ～ D32764	
无区号 EM 区	E00000 ～ E32764	
有区号 EM 区	En_00000 ～ En_32764 (n = 0 ～ C)	

区域	S	D
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	
数据寄存器	---	
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15	

说明

EXPD(858) 计算在字 S ～ S+3 中的一个双精度（64 位）浮点数的自然（底为 e）指数值，并将计算结果存入字 D ～ D+3 中，换言之， EXPD(858) 计算 e^X（X 为源数据），并将结果存入字 D ～ D+3 中。

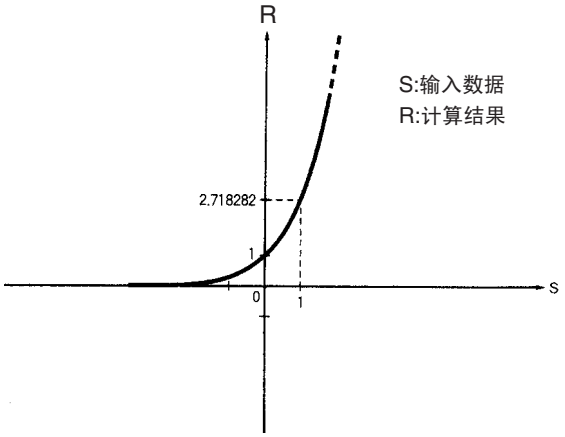


如果计算结果的绝对值大于浮点数所能表示的最大值，上溢出标志置为 ON，计算结果输出为 ±∞。

如果计算结果的绝对值小于浮点数所能表示的最小值，下溢出标志置为 ON，计算结果输出为 0。

注 常数 e 为 2.718282。

下图为输入数据和计算结果的关系图。



标志

名称	标记	操作
错误标志	ER	源数据不能被识别为浮点数时置 ON。 源数据为负时则置 ON。 其余情况下置 OFF。
等于标志	=	计算结果的指数和尾数全为 0 置 ON。 其余情况下置 OFF。
上溢出标志	OF	计算结果的绝对值太大超出一个双精度 （64 位）浮点数表示范围时置 ON。
下溢出标志	UF	计算结果的绝对值太小超出一个双精度 （64 位）浮点数表示范围时置 ON。
负标志	N	不变

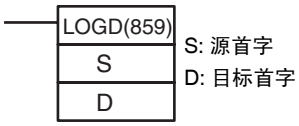
注意 表示为字 S ～ S+3 的源数据必须是 IEEE754 浮点数格式。

3-16-19 双对数：LOGD(859)

用途 计算一个双精度 （64 位）浮点数的自然对数 （底为 e），并把结果存入指定的结果字中。

此指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每个循环执行	LOGD(859)
	上升沿微分时执行一次	@LOGD(859)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数规定

区域	S	D
CIO 区	CIO 0000 ～ CIO 6140	
工作区	W000 ～ W508	
保持位区	H000 ～ H508	
辅助位区	A000 ～ A956	A448 ～ A956
定时器区	T0000 ～ T4092	
计数器区	C0000 ～ C4092	
DM 区	D00000 ～ D32764	
无区号 EM 区	E00000 ～ E32764	
有区号 EM 区	En_00000 ～ En_32764 (n = 0 ～ C)	

区域	S	D
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	
数据寄存器	---	
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15	

说明

LOGD(859) 计算在字 S ～ S+3 中的一个双精度 （64 位）浮点数的自然对数（底为 e），并将结果存入字 D ～ D+3。

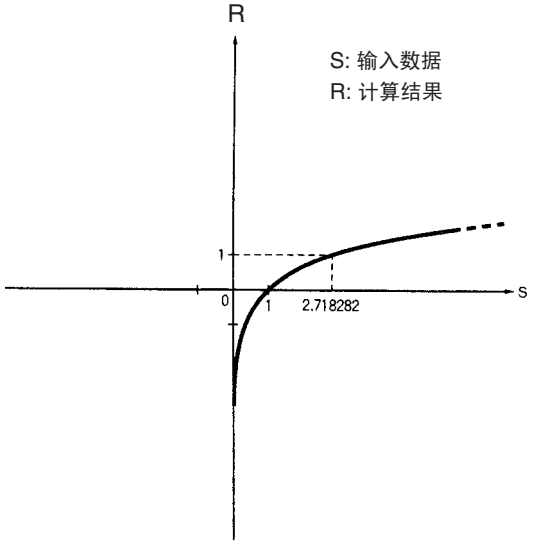
$\log_e$ [S+3 | S+2 | S+1 | S] → [D+3 | D+2 | D+1 | D]

源数据必须为正值，如果为负则产生一个错误，并且指令不执行。

如果计算结果的绝对值大于浮点数所能表示的最大值，上溢出标志置为 ON，计算结果输出为 ±∞。

注 常数 e 为 2.718282。

下图为输入数据和计算结果的关系图。



标志

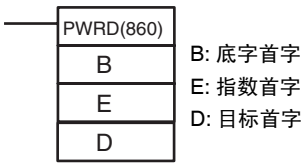
名称	标记	操作
错误标志	ER	源数据不能被识别为浮点数时置 ON。 源数据为负时则置 ON。 源数据非数字 （NaN）时置 ON。 其余情况下置 OFF。
等于标志	=	计算结果的指数和尾数全为 0 置 ON。 其余情况下置 OFF。
上溢出标志	OF	计算结果的绝对值太大超出一个双精度 （64 位）浮点数表示范围时置 ON。
下溢出标志	UF	不变
负标志	N	计算结果为负时置 ON。 其余情况下置 OFF。

注意 在字 S ～ S+3 中的源数据必须是 IEEE754 浮点数据格式。

3-16-20 双指数幂：PWRD(860)

用途 将一个双精度 （64 位）浮点数加到另一个双精度 （64 位）浮点数上作为幂。此指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每个循环执行	PWRD(860)
	上升沿微分时执行一次	@PWRD(860)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

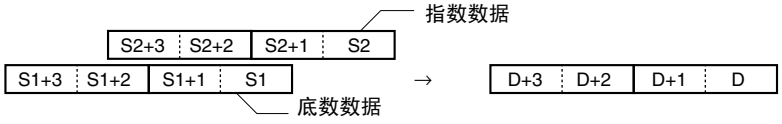
操作数规定

区域	B	E	D
CIO 区	CIO 0000 ～ CIO 6140		
工作区	W000 ～ W508		
保持位区	H000 ～ H508		
辅助位区	A000 ～ A956		A448 ～ A956
定时器区	T0000 ～ T4092		
计数器区	C0000 ～ C4092		
DM 区	D00000 ～ D32764		
无区号 EM 区	E00000 ～ E32764		

区域	B	E	D
有区号 EM 区	En_00000 ～ En_32764 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---		
数据寄存器	---		
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15		

说明

PWRD(860) 将以字 B ～ B+3 表示的双精度 （64 位）浮点数自乘到以字 E ～ E+3 表示的另一个双精度 （64 位）浮点数次幂。换言之， PWRD(860) 计算 X^Y(X= B ～ B+3 ； Y= E ～ E+3)。



例如，当底数(B～B+3)为3.1而指数(E～E+3)为3时，计算结果为3.1^3或29.791。

如果计算结果的绝对值大于浮点数所能表示的最大值，上溢出标志置为 ON。

如果计算结果的绝对值小于浮点数所能表示的最小值，下溢出标志置为 ON。

标志

名称	标记	操作
错误标志	ER	如果底数 （B ～ B+3）或指数 （E ～ E+3）没有当作浮点数据确认时置 ON。 如果底数 （B ～ B+3）或指数 （E ～ E+3）不是一个数字 （NaN）时则置 ON。 如果底数 （B ～ B+3）为 0，并且指数 （E ～ E+3）小于 0 时置 ON。（除以 0） 如果底数 （B ～ B+3）为负，并且指数 （E ～ E+3）为非整数时置 ON。（一个负数的根） 其余情况下置 OFF。
等于标志	=	如果结果的指数和尾数全为 0 时置 ON。 其余情况下置 OFF。
上溢出标志	OF	如果结果的绝对值太大而不能用一个双精度浮点值表示时置 ON。

名称	标记	操作
下溢出标志	UF	如果结果的绝对值太小而不能用一个双精度浮点值表示时置 ON。
负标志	N	如果结果为负时置 ON。 其余情况下置 OFF。

注意 底数 (B ～ B+3t) 和指数 (E ～ E+3t) 必须是 IEEE754 浮点数据格式。

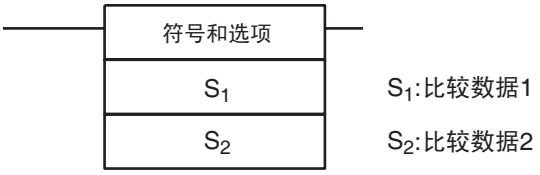
3-16-21 双精度浮点输入指令

用途 这些输入比较指令比较两个双精度浮点数值 （64 位 IEEE754 常数和 / 或指定字的内容）并且当比较条件为真时产生一个 ON 执行条件。

这些指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持。

注 关于有符号和无符号二进制输入比较指令详细情况请参见 *3-7-1 输入比较指令 (300 ～ 328)*，并且关于单精度浮点输入比较指令详情可参见 *3-16-21 单精度浮点比较指令*。

梯形图符号



变化

变化	每个循环比较为真时产生 ON	输入比较指令
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数规定

区域	S_1	S_2
CIO 区	CIO 0000 ～ CIO 6140	
工作区	W000 ～ W508	
保持位区	H000 ～ H508	
辅助位区	A000 ～ A956	
定时器区	T0000 ～ T4092	
计数器区	C0000 ～ C4092	
DM 区	D00000 ～ D32764	
无区号 EM 区	E00000 ～ E32764	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	

区域	S_1	S_2
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	
数据寄存器	---	
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15	

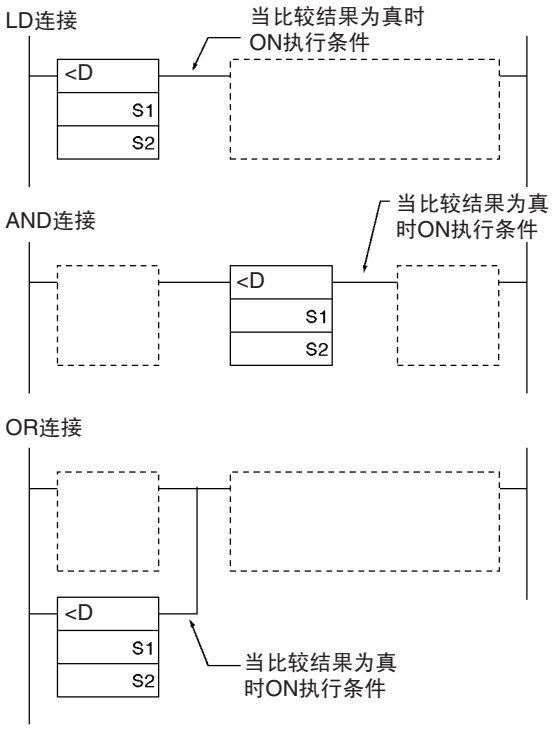
说明

输入比较指令把 S_1 和 S_2 中指定的数据作为双精度浮点数值 （64 位 IEEE754 数据）进行比较，并且当比较条件为真时产生一个 ON 执行条件。当数据存储在字中时， S_1 和 S_2 指定了包含 64 位数据的 4 个字中的第一个。64 位浮点数据不能为常数的输入。

输入指令

输入比较指令被看作如 LD， AND，和 OR 指令用来控制后面指令的执行。

输入类型	操作
LD	指令能直接与左边总线相连
AND	指令不能直接与左边总线相连
OR	指令能直接与左边总线相连



选项

由 3 种输入类型和 6 种符号可知有 18 种不同的可能的连接。

符号	选项 （数据格式）
= （等于） < > （不等于） < （小于） <= （小于或等于） > （大于） >= （大于或等于）	D: 双精度浮点数据

输入比较指令的总结

下表显示了 18 种双精度浮点数输入比较指令的功能码，助记符，名称和功能 (C1=S_1+3, S_1+2, S_1+1, S_1 和 C2=S_2+3, S_2+2, S_2+1, S_2)

代码	助记符	名称	功能
335	LD=D	加载双浮点等于	如果 C1 = C2 则为真
	AND=D	与双浮点等于	
	OR=D	或双浮点等于	
336	LD<>D	加载双浮点不等于	如果 C1 ≠ C2 则为真
	AND<>D	与双浮点不等于	
	OR<>D	或双浮点不等于	
337	LD<D	加载双浮点小于	如果 C1 < C2 则为真
	AND<D	与双浮点小于	
	OR<D	或双浮点小于	
338	LD<=D	加载双浮点小于或等于	如果 C1 ≤ C2 则为真
	AND<=D	与双浮点小于或等于	
	OR<=D	或双浮点小于或等于	
339	LD>D	加载双浮点大于	如果 C1 > C2 则为真
	AND>D	与双浮点大于	
	OR>D	或双浮点大于	
340	LD>=D	加载双浮点大于或等于	如果 C1 ≥ C2 则为真
	AND>=D	与双浮点大于或等于	
	OR>=D	或双浮点大于或等于	

标志

表中， C1= S1 到 S1+3 的内容和 C2= S2 到 S2+3 的内容。

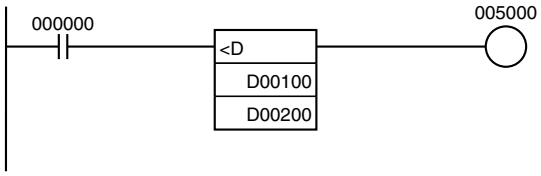
名称	标记	操作
错误标志	ER	如果 C1 或 C2 不是一个有效的浮点数字 (NaN) 则置 ON。 如果 C1 或 C2 是 + ∞则置 ON。 如果 C1 或 C2 是 - ∞则置 ON。 其余情况下置 OFF。
大于标志	>	如果 C1>C2 则置 ON。 其余情况下置 OFF。
大于或等于标志	> =	如果 C1 ≥ C2 则置 ON。 其余情况下置 OFF。
等于标志	=	如果 C1 = C2 则置 ON。 其余情况下置 OFF。
不等于标志	=	如果 C1 ≠ C2 则置 ON。 其余情况下置 OFF。

名称	标记	操作
小于标志	<	如果 C1 < C2 则置 ON。 其余情况下置 OFF。
小于等于标志	< =	如果 C1 ≤ C2 则置 ON。 其余情况下置 OFF。
负标志	N	不变

注意 输入指令不能用作右侧指令，即另一个指令必须用在它们和右母线之间。

例 **与双字浮点小于：AND<D(331)**

在下例当CIO 000000 置ON时，D00100～D00103之间的浮点数据与D00200～D00203之间的浮点数据相比较。如果D00100～D00103的内容小于D00200～D00203的内容，执行进行到紧挨着的行，CIO005000置为ON。如果D00100～ D00103 的内容不小于 D00200 ～ D00203 的内容，执行过程不会执行到紧挨着的行。



双字浮点小于比较 (<D)

	15															0
S1 :D00100	1	0	0	0	1	0	1	1	0	1	0	0	0	1	0	0
S1+1:D00101	1	1	1	0	0	1	1	1	0	1	1	0	1	1	0	0
S1+2:D00102	1	0	1	0	1	0	0	1	1	1	1	1	1	1	1	1
S1+3:D00103	0	1	0	0	0	0	0	0	0	0	0	0	1	0	1	1

十进制值：3.4580

	15															0
S1 :D00100	0	1	1	1	1	0	0	1	0	0	1	1	1	1	1	0
S2+1:D00101	1	0	1	0	1	0	0	0	0	1	0	1	1	0	0	0
S2+2:D00102	1	1	0	0	1	1	0	1	0	0	1	1	0	1	0	1
S2+3:D00103	0	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1

十进制值：-1.4876

34580>14876

未导致 ON 条件

	15															0
S1 :D00100	1	1	0	1	1	1	1	0	1	0	0	1	0	0	0	1
S1+1:D00101	1	0	1	0	1	0	0	1	1	0	1	1	0	1	1	0
S1+2:D00102	1	1	1	0	1	1	0	1	1	0	1	1	0	0	0	0
S1+3:D00103	1	1	0	0	1	0	1	0	0	0	0	0	0	0	1	0

十进制值：−3.4580E+48

	15															0
S1 :D00100	0	1	0	1	0	1	0	0	0	1	0	1	0	0	1	1
S2+1:D00101	1	0	1	0	1	0	0	0	0	0	1	0	1	0	1	1
S2+2:D00102	0	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0
S2+3:D00103	0	1	0	0	1	0	0	1	1	1	1	1	0	0	0	0

十进制值：1.4876E+48

−3.4580E+48<1.4876E+48

导致 ON 条件

3-17 表处理指令

本节介绍了用于处理表数据、堆栈和其它数据范围指令。在表格底端的 5 条指令 （用 * 标志）仅支持 CS1-H,CJ1-H,CJ1M 和 CS1D CPU 单元。

指令	助记符	功能代码	页
设置堆栈	SSET	630	623
推入栈	PUSH	632	626
先进先出	FIFO	633	629

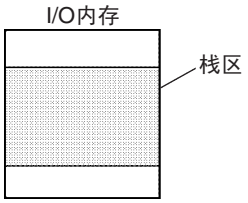
指令	助记符	功能代码	页
后进先出	LIFO	634	632
定维记录表	DIM	631	634
设定记录位置	SETR	635	638
得到记录号	GETR	636	640
数据搜索	SRCH	181	642
交换字节	SWAP	637	643
寻找最大值	MAX	182	646
寻找最小值	MIN	183	649
求和	SUM	184	653
帧校验和	FCS	180	656
堆栈号输出	SNUM	638	659
堆栈数据读	SREAD	639	662
堆栈数据覆盖	SWRIT	640	665
堆栈数据插入	SINS	641	668
堆栈数据删除	SDEL	642	671

上述指令都要定义或操作字组。栈字组由 SSET(630) 定义，记录表字组由 DIM (631) 定义，各范围指令的字组在每个指令中分别进行定义。

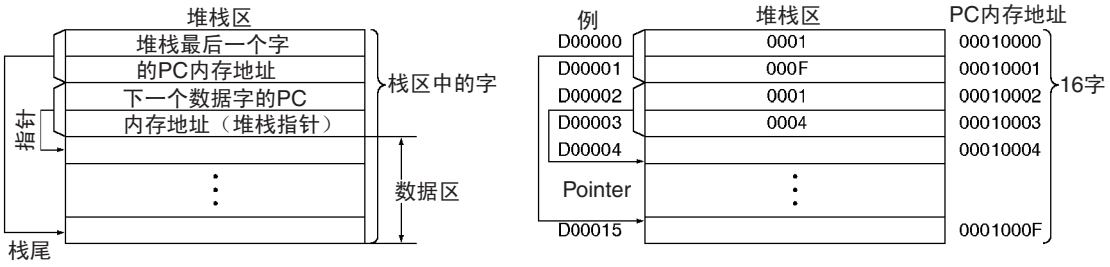
组	用途	指令
栈	对数据表进行 FIFO （先进先出）或 LIFO （后进先出）操作	SSET(630), PUSH(632), FIFO(633), LIFO(634), SREAD(639), SWRIT(640), SINS(641), SDEL(642) 和 SNUM(638)
记录表	对由记录组成的表执行操作 （记录的大小由用户设定）	DIM(631), SETR(635) 和 GETR(636)
范围	对一个范围的字操作，求出范围中的校验和，特殊值，最大值或最小值。	FCS(180), SRCH(181), MAX(182), MIN(183), SUM(184) 和 SWAP(637)

堆栈指令

堆栈指令作用于专门指定的数据表，即堆栈。堆栈的前两个字储存栈最后一个输入字的内部 I/O 储存地址，接下来的两个字储存指针。（字的内部 I/O 内存地址将被下一个 PUSH(632) 指令重写）。



下表表明了堆栈的基本结构。

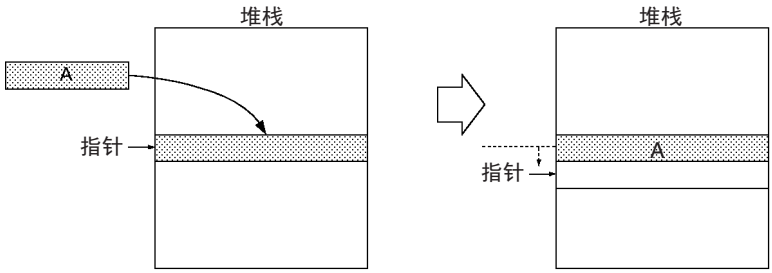


下述指令定义或作用于栈区，基本上，PUSH(632) 将数据存入栈中下一个有效数据字中。FIF0(633) 和 LIF0(634) 从栈中读取数据。FIF0(633) 读出存储的首字，而 LIF0(634) 读出存储的末字。

CS1-H，CJ1-H，CJ1M 和 CS1D CPU 支持后五条指令。SNUM(638) 统计在指定堆栈中数据元素（字）的数目；例如，该指令能用来表明在一个传送器上的条项数。用 SREAD(639)，SWRIT(640)，SINS(641) 和 SDEL(642) 指令来读取，重写，插入和删除在堆栈中的数据元素。例如当器件在传送带上处理时，这些指令能增加，移去或改变在堆栈中的数据元数，这些元数对应于传送带上的器件。

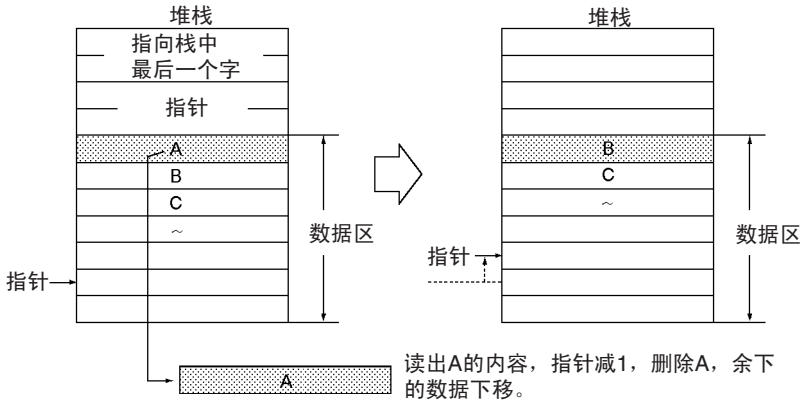
PUSH(632)

将数据存入栈指针指示的地址中，并将指针加 1。



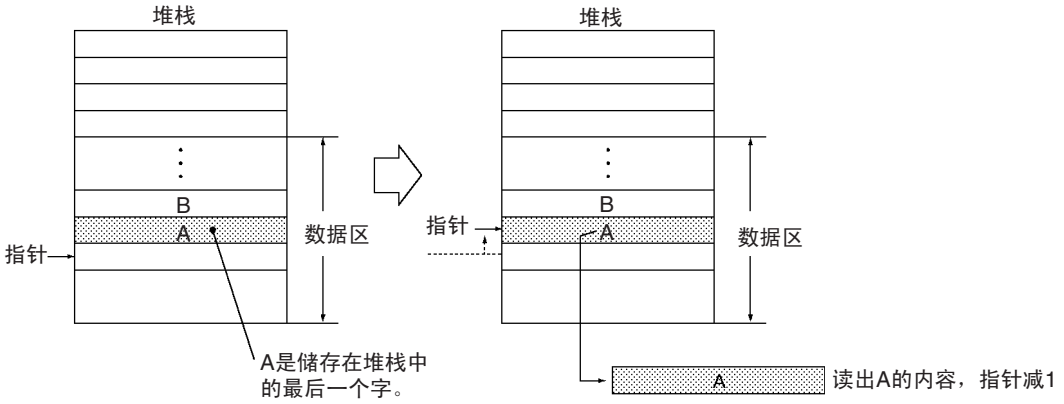
FIFO(633)

读出栈中最早存入的字（最老的），余下的数据下移 1 个，指针减 1。



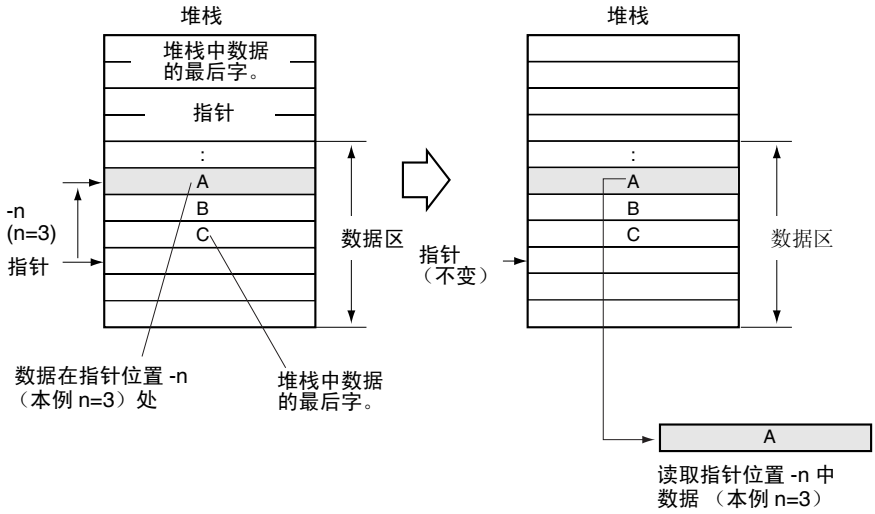
LIFO(634)

读出栈中最晚（最新）存入的字。指针减 1 并读出该地址的数据（栈中最晚存入的数据）。读后数据将不被清除。



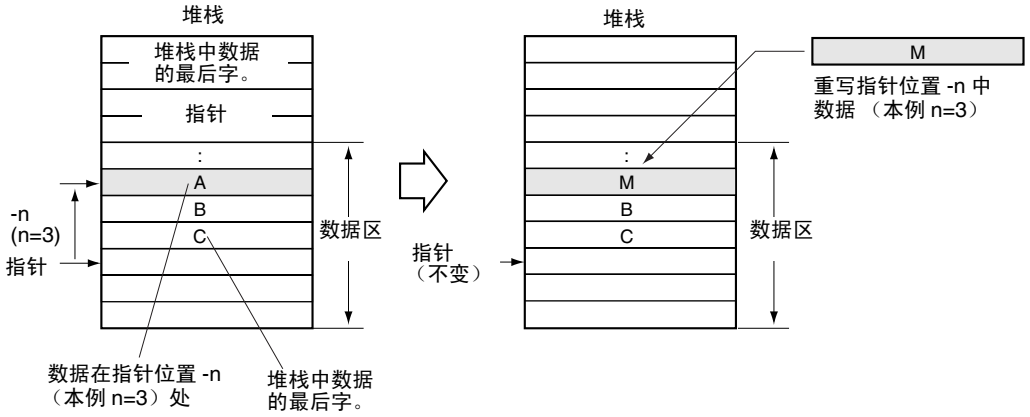
SREAD(639)

从堆栈中读取指定数据元素中的数据。偏移值表明需要字的位置（字数在当前指定指针位置之前）。



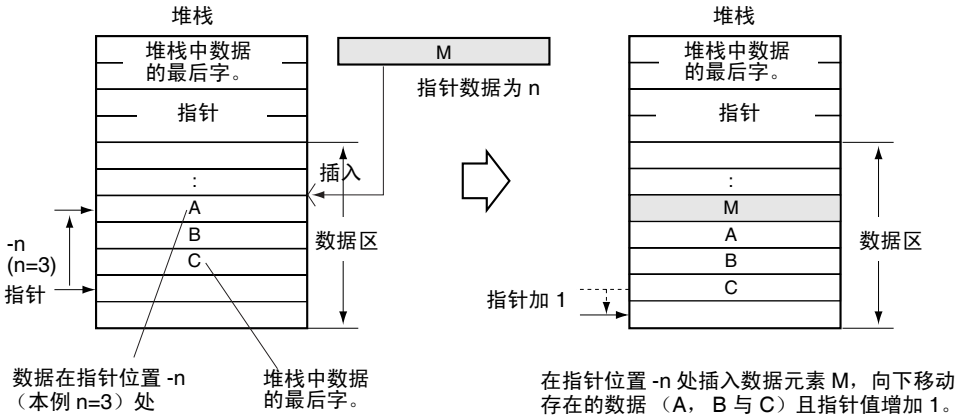
SWRIT(640)

把源数据写入到堆栈中指定数据元数处 （覆盖存在的数据）。偏移值表明需要字的位置 （在当前指针位置之前的字数）。



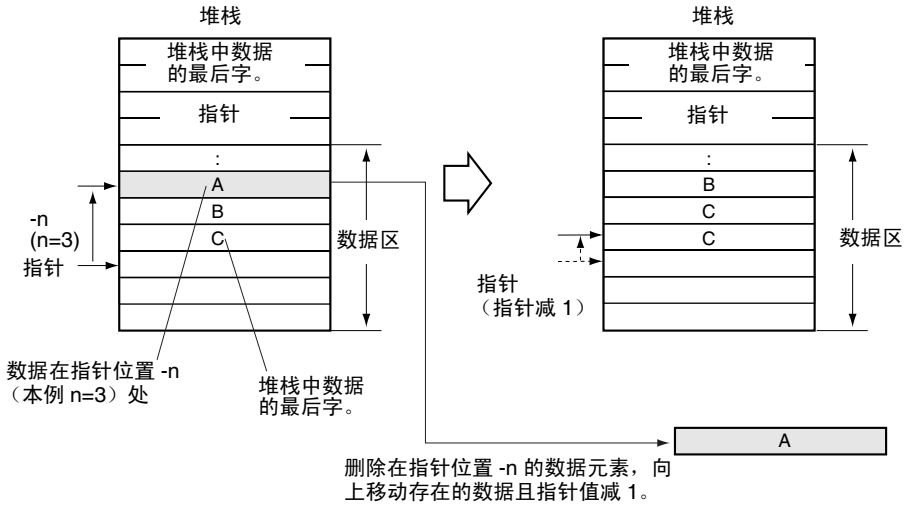
SINS(641)

在堆栈中指定位置处插入源数据。在堆栈中向下移动数据。偏移值表明需要字的位置 （在当前指针位置前的字数数目）。



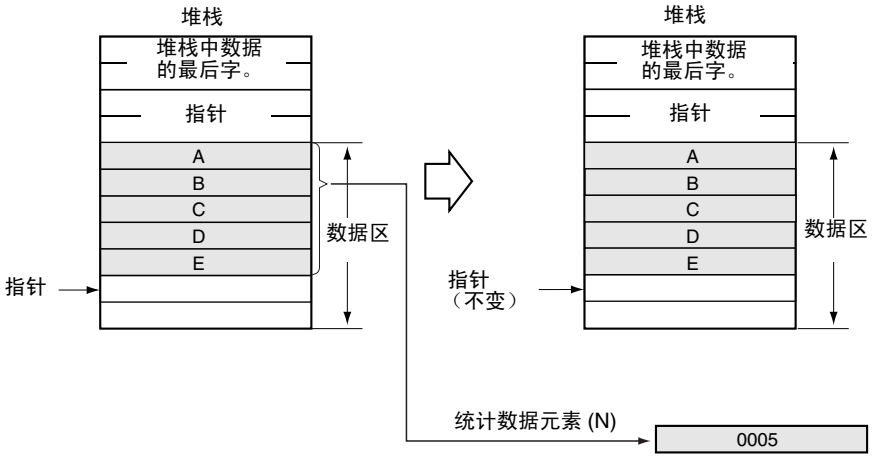
SDEL(642)

在堆栈中删除指定位置的数据元素，向下移动堆栈中余下的数据元素。偏移值表明需要字的位置 （在当前指针位置前字数的数目）。



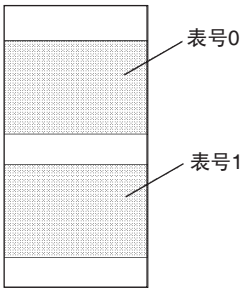
SNUM(638)

从堆栈指针数据范围开始的位置起，统计堆栈数据的数目 （数据字数）。

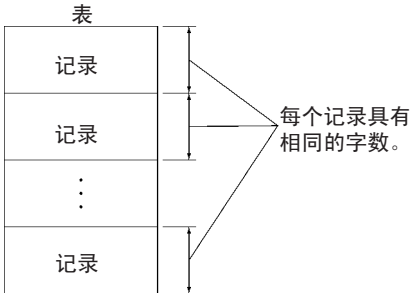


记录表指针

一组由多个具有相同字数的记录组成的数据称为表数据。存储在指定 I/O 内存中的数据表可以用 DIM 指令注册为表区域。最多可有 16 个单独的表被定义为表 0 ～表 15。



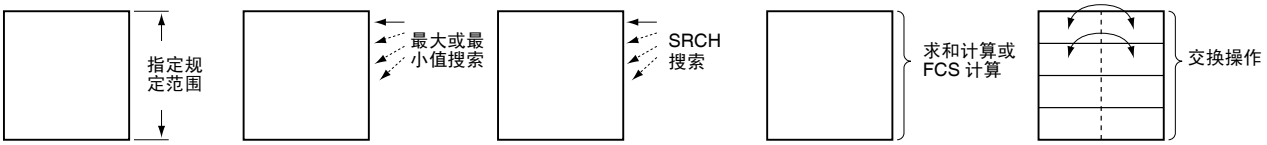
下图展示了一个记录表的基本结构。表中每一个记录具有相同的字数。



索引寄存器 (IR) 可以间接地访问表数据，记录地址的计算可以很容易地通过 SEJT(635) （设置记录号）指令和 GEJT(636) （得到记录号）指令实现。

范围指令

范围指令包括作用于指定字的范围求最大值 (MAX(182)) 或最小值 (MIN(183))，搜索特定值 (SRCH(181))，求和 (SUM(184)) 或 FCS(FCS(180))，交换字的左字节和右字节的内容 (SWAP(637)) 。

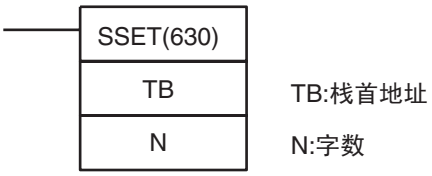


3-17-1 设置堆栈：SSET(630)

用途

定义栈的指定长度并指定起始字。

梯形图符号



变化

变化	ON 条件时每次周期执行	SSET(630)
	上升沿微分执行一次	@SSET(630)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

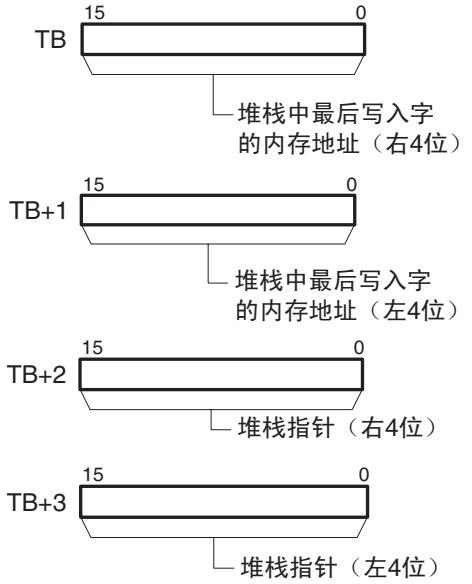
适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

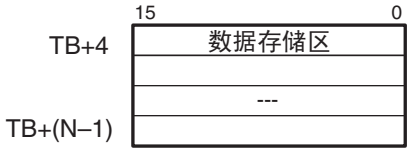
操作

TB ～ TB+3: 栈控制字

栈的前 4 个字包含了最后写入栈的字的内部 I/O 存储地址和栈指针。（PLC 下一个字的内存地址将被 PUSH(632) 重写）。



从 TB+4 ～ TB+(N-1)：数据存贮区
堆栈的其余部分用于存放数据。



注 1. 栈指针的初始化值总是等于 TB+4 的内部 I/O 存储地址。
2. TB 和 TB+(N-1) 必须位于同一数据区。

操作数规定

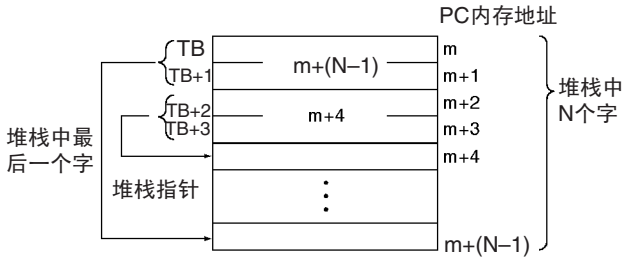
区域	TB	N
CIO 区	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W511	
保持位区	H000 ～ H511	
辅助位区	A448 ～ A959	A000 ～ A959
定时器区	T0000 ～ T4095	
计数器区	C0000 ～ C4095	
DM 区	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	#0005 ～ #FFFF （二进制） 或 &5 ～ &65,535

区域	TB	N
数据寄存器	---	DR0 ～ DR15
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15	

描述

SSET(630) 设定了一个始于 TB 止于 TB+(N-1) 的具有 N 个字的栈。栈的前两个字 （TB 和 TB+1）存贮了栈的最后一个字的 8 位 16 进制的内部 I/O 存储地址。接下来的两个字 （TB+3 和 TB+2）存贮了栈指针。栈指针是栈中下一个将被 PUSH(632) 覆写的内部 I/O 存储地址 , 它的初始值是 TB+4 的地址。

SSET(630) 自动将栈数据区 （从 TB+4 ～ TB+(N-1)）始化为 0 。下图展示了栈的 基本结构。



SSET(630) 只建立和初始化栈 , 应通过使用下述指令向栈中存入或读出数据。

1,2,3...

1. PUSH(632) 每次向栈中存入一个字的数据。
2. FIFO(633) 和 LIF0(634) 从栈中读出数据。 FIFO(633) 读出存储的第一个字， LIFO(634) 读出存储的最后一个字。
3. 栈控制字中的指针值在执行 PUSH(632) ，FIFO(633) 或 LIFO(634) 指令时被动态更新。通常用户不需要关注栈控制字。当通过非前述指令访问堆栈的内容时，可用索引寄存器 (IR) 间接访问来设定指针值。

标志

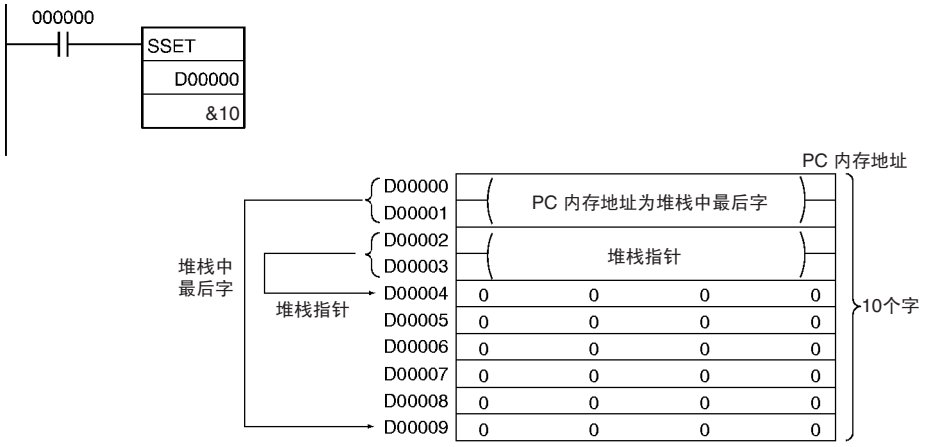
名称	标记	操作
错误标志	ER	如果 N 不是在指定的范围 0005 ～ FFFF 置 ON。 其余为 OFF。

注意

栈的最小字数 (N) 为 5, 因为 N 中有 4 个字用于存储栈最后一个字和栈指针。如果 N 不在 0005 ～ FFFF 范围内 , 将产生一个错误。

例

如下例中当 CIO 000000 为 ON 时， SSET(630) 定义了一个从 D000000 ～ D000009 的 10 字栈。D000000 和 D000001 包含了栈最后一个字的内部 I/O 存储地址。 D00002 和 D00003 存贮栈指针，栈本身的数据区从 D00004 开始。

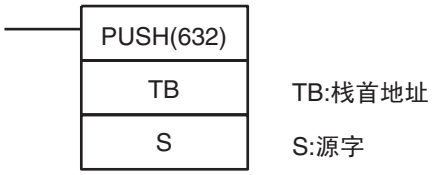


3-17-2 压入堆栈：PUSH(632)

用途

向指定堆栈写入一个数据字。

梯形图符号



变化

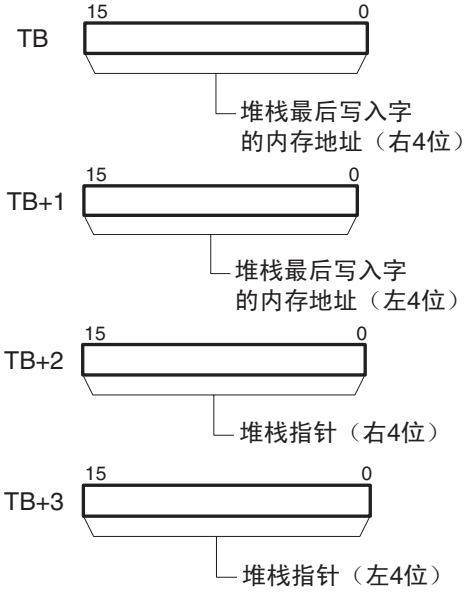
变化	ON 条件时每次周期执行	PUSH(632)
	上升沿微分执行一次	@PUSH(632)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作

从 TB ～ TB+3：栈控制字
栈的前 4 个字包含了最后写入栈的字的 PLC 内存存储地址和栈指针。（被 PUSH(632) 重写的内容在 PLC 下一个字的内存地址中）。



TB+4 ～ TB+(N-1)：数据存贮区
堆栈的其余部分用于存放数据。

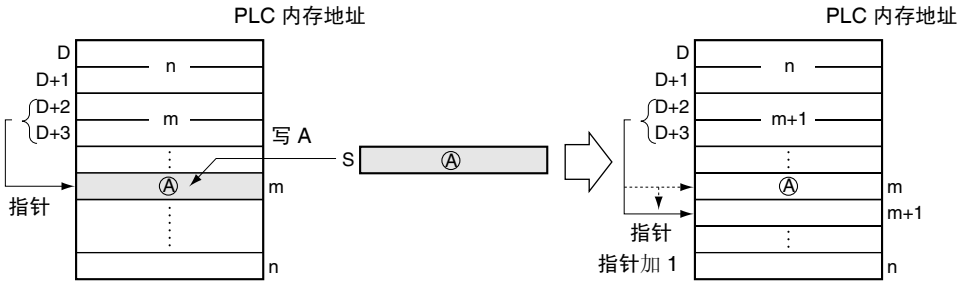
操作数规定

区域	TB	S
CIO 区	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W511	
保持位区	H000 ～ H511	
辅助位区	A448 ～ A959	A000 ～ A959
定时器区	T0000 ～ T4095	
计数器区	C0000 ～ C4095	
DM 区	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	#0000 ～ #FFFF （二进制）
数据寄存器	---	DR0 ～ DR15

区域	TB	S
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15	

描述

PUSH(632) 把 S 中的内容写到由堆栈指针 （TB+3 和 TB+2）指向的地址中，栈指针增加 1。



在 PUSH(632) 将数据写到堆栈后，FIFO(633) 和 LIFO(634) 可以用来从堆栈中读取数据。

标志

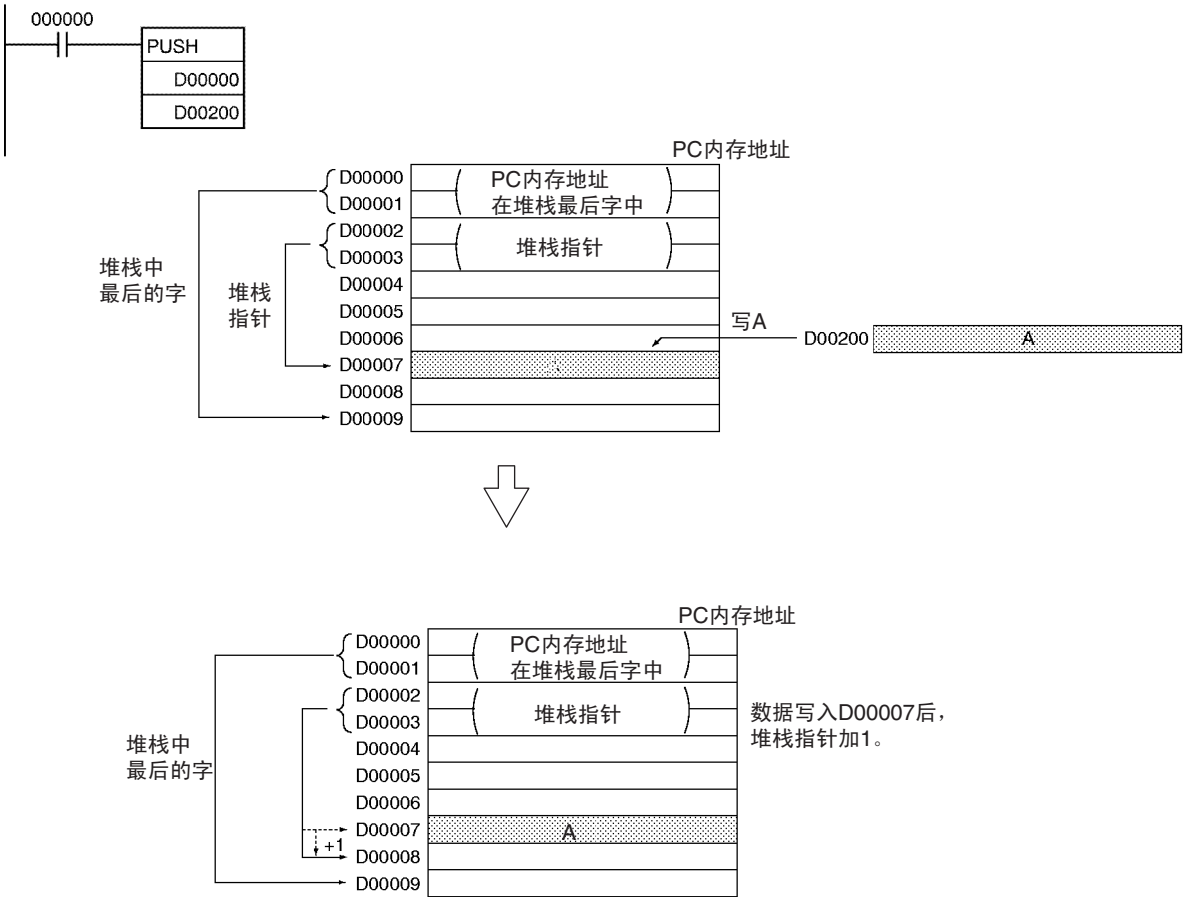
名称	标记	操作
Error Flag	ER	如果堆栈指针 （TB+3 和 TB+2）指定的地址超过堆栈最后一个字时置 ON （这是栈溢础错误） 其余情况为 OFF。

注意

堆栈必须预先用 SSET(630) 定义。

例

在下例中当 CIO 000000 是 ON 时，PUSH(632) 将 D00200 的内容复制到堆栈开始 D00000 处。此时，堆栈指针指向 D00007。

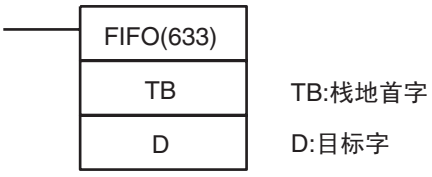


3-17-3 先进先出：FIFO(633)

用途

读出指定栈中最先写入的数据（栈中最老的数据）。

梯形图符号



变化

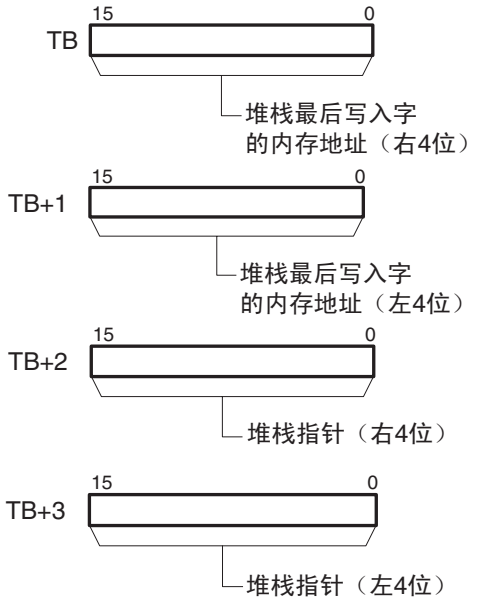
变化	ON 条件时每次周期执行	FIFO(633)
	上升沿微分执行一次	@FIFO(633)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

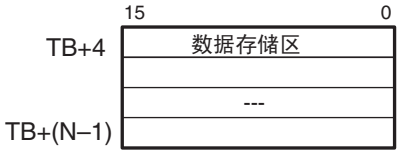
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

从 TB ～ TB+3：栈控制字
栈的前 4 个字包含了最后写入栈的字的 PLC 内存存储地址和栈指针。（被 PUSH(632) 重写的内容在 PLC 下一个字的内存地址中）。



TB+4 ～ TB+(N-l)：数据存贮区
堆栈的其余部分用于存放数据。



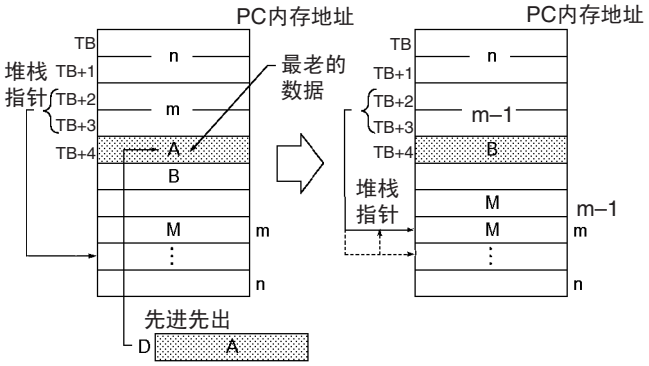
操作数规定

区域	TB	D
CIO 区	CIO 0000 ～ CIO 6143	
工作区	W000 ～ 4W511	
保持位区	H000 ～ H511	
辅助位区	A448 ～ A959	
定时器区	T0000 ～ T4095	
计数器区	C0000 ～ C4095	
DM 区	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	

区域	TB	D
数据寄存器	---	DR0 ～ DR15
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15	

描述

FIFO(633) 读取栈中最老的字 (BT+4) 并将数据输出到 D，然后栈指针 （TB+3 和 TB+2）减 1。栈中余下的数据下移一个字， TB+4 中被读的数据删除，栈尾的数据 （栈指针指向的地址）保持不变。



将 FIF0(633) 和 PUSH(632) 结合使用，在 PUSH(632) 向栈中写入数据后，FIFO(633) 可以用先进先出的方式读取栈中数据。

FIFO(633) 读取栈始点数据并删除该数据 , 然后将下一数据上移。

标志

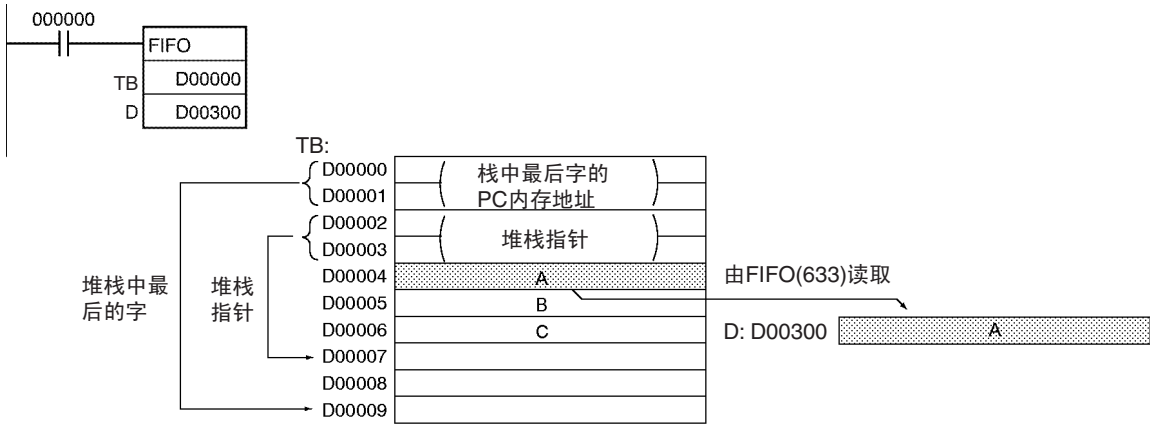
名称	标记	操作
错误标志	ER	如果栈指针 （TB+3 和 TB+2）指定的地址小于或等于栈数据区第一个字 (TB+4) 的地址时置 ON。 （这是一个栈下溢出错误） 其它情况置 OFF。

注意

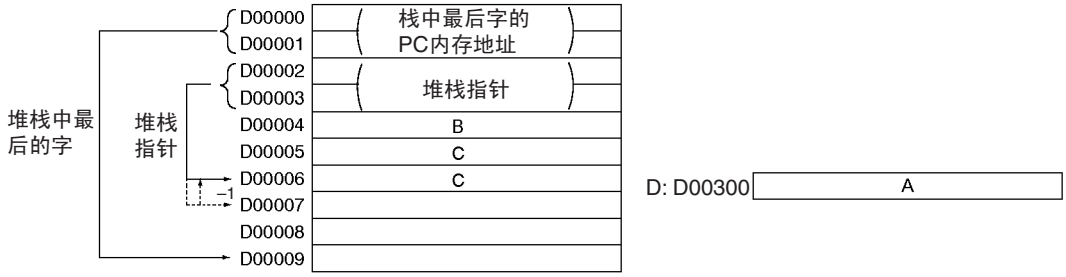
堆栈必须预先用 SSET(630) 定义。

例

当下例中 CIO 000000 为 ON 时， FIFO(633) 读取 D00004 （栈在 D00000 起始时的 TB）中的内容，并将数据写入 D00300 。



数据写入 D00300 后，栈指针减 1，余下的数据下移 （D00005 的内容移至 D00004， D00006 的内容移至 D00005）。

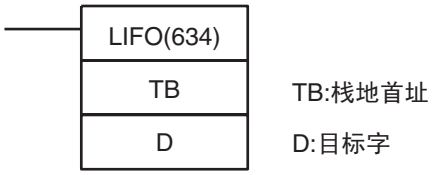


3-17-4 后进先出：LIFO(634)

用途

读取所指定栈的最后写入栈的数据 （栈中最新的数据）。

梯形图符号



变化

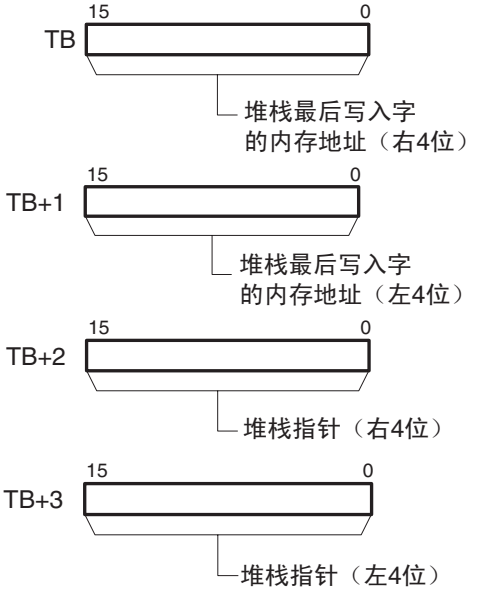
变化	ON 条件时每次周期执行	LIFO(634)
	上升沿微分执行一次	@LIFO(634)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

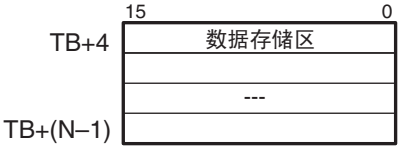
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

TB ～ TB+3：栈控制字
栈的前 4 个字包含了最后写入栈的字的 PLC 内存地址和栈指针。（被 PUSH (632) 重写的内容在 PLC 下一个字的内存地址中）。



TB+4 ～ TB+(N-1)：数据存贮区
堆栈的其余部分用于存放数据。



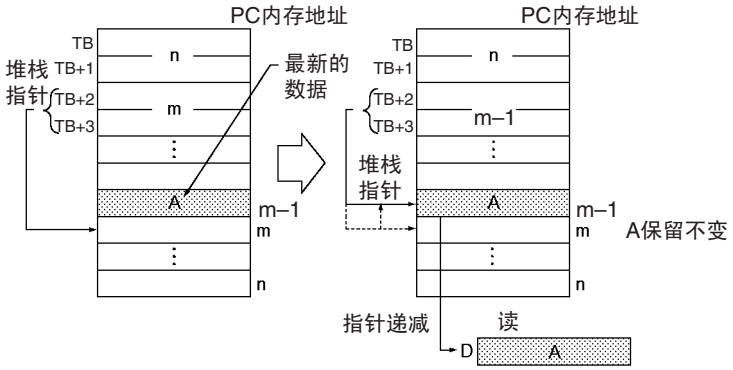
操作数规定

区域	TB	D
CIO 区域	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W511	
保持位区	H000 ～ H511	
辅助位区	A448 ～ A959	
定时器区	T0000 ～ T4095	
计数器区	C0000 ～ C4095	
DM 区	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	

区域	TB	D
数据寄存器	---	DR0 ～ DR15
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15	

描述

LIFO(634) 读取栈指针指向的地址中的数据 （栈中最新字的数据），栈指针减 1，并将数据输出到 D，所读的字保留不变。



将 LIFO(634) 和 PUSH(632) 结合使用，在 PUSH(632) 向栈中写入数据后，LIFO(634) 可以以后进先出的方式读取栈中数据。使用 PUSH(632) 存储数据后，栈指针指向下一个最后数据的地址。

标志

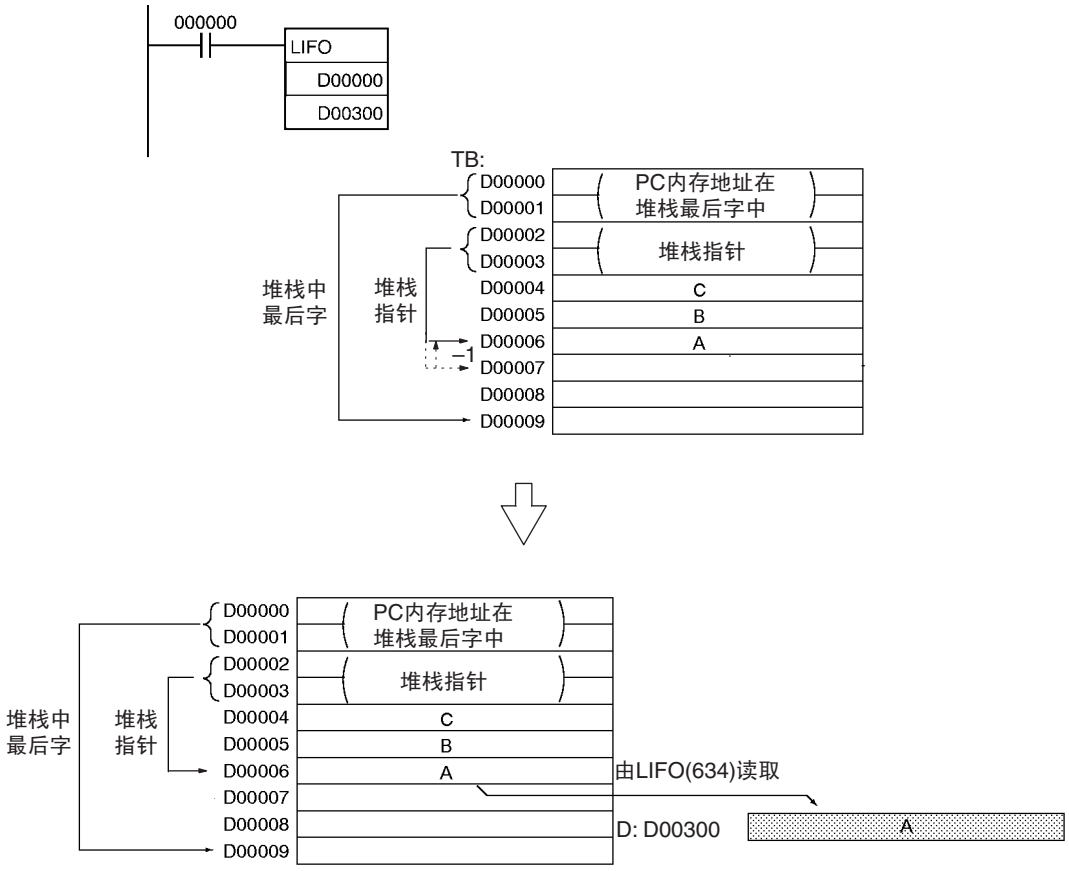
名称	标记	操作
错误标志	ER	栈指针 （TB+3 和 TB+2）指定的地址小于或等于栈数据区第一个字 (JB+4) 地址时置 ON。 （这是一个栈下溢出错误） 其它情况置 OFF。

注意

堆栈必须预先用 SSET(630) 定义。

例

在下例中当 CIO 000000 为 ON 时，LIFO(634) 读出堆栈指针 (D00006) 字的内容，并将数据写入 D00300。



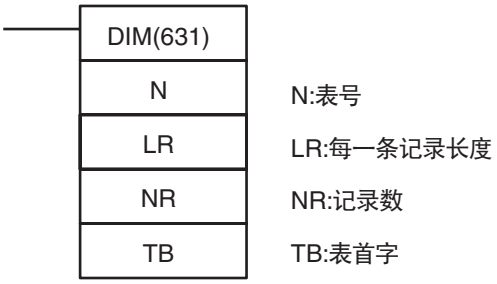
数据写入 D00300 后，堆栈指针减 1,D000006 中的内容不变。

3-17-5 定维记录表 :DIM(631)

用途

通过约定每一个记录的长度和记录数将指定的 I/O 内存区定义为记录表。定义的记录表可达 16 个。

梯形图符号



变化

变化	ON 条件时每次周期执行	DIM(631)
	上升沿微分执行一次	@DIM(631)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作

N: 表号
指示表号， N 为 0 ～ 15 之间。

LR: 每个记录的长度
指示每个记录中的字数， LR 必须为 0001 ～ FFFF 之间的 16 进制数 （1 ～ 65535 个字）

NR: 记录数
指示表中记录的个数，NR必须为0001～FFFF的16进制数（1～65,535个字）。

TB: 表首字
指示表中的第一个字，表中所有的字都必须在同一数据区域内，也就是说 TB 和 TB+LR × NR-1 必须在同一数据区域内。

操作数规定

区域	N	LR	NR	TB
CIO 区	---	CIO 0000 ～ CIO 6143		
工作区	---	W000 ～ W511		
保持位区	---	H000 ～ H511		
辅助位区	---	A000 ～ A959		A448 ～ A959
定时器区	---	T0000 ～ T4095		
计数器区	---	C0000 ～ C4095		
DM 区	---	D00000 ～ D32767		
无区号 EM 区	---	E00000 ～ E32767		
有区号 EM 区	---	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	---	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	---	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	0 ～ 15	#0001 ～ #FFFF （二进制）或 &1 ～ &65,535		---
数据寄存器	---	DR0 ～ DR15		---
索引寄存器	---	---		
使用索引寄存器的间接寻址	---	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0～DR15, IR0～IR15 ,IR0+(++)～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15		

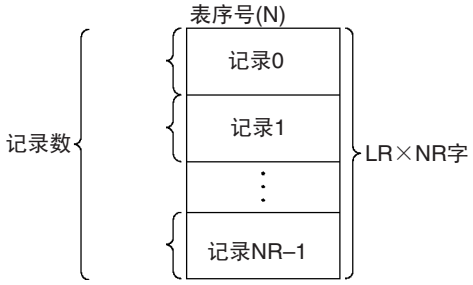
说明

DIM(631) 将 TB ～ TB+LRxNR-1 注册表号为 N，表 N 有 NR 个记录，每个记录有 LR 个字长。一旦某个区域被约定为记录，该范围内的数据将不能改变。

DIM(631) 与 SETR(635)（设置记录号）或 GETR(636)（获得记录号）结合使用以简化数据表中地址的计算。使用 DIM (631) 将数据分割为记录，然后用

SETR(635) 将需要记录的首地址存入索引寄存器，索引寄存器就可以在其它指令诸如读，写，搜索或比较指令中作为指针使用。

例如，如果将温度、压力或其它设置值存入记录中，不同型号的记录被组成一个表，那么就方便地读出指定条件下每种型号的设定值。



SETR(635) 和 GETR(636)（是 DIM(631) 相关联的两个记录表指令，SETR(635) 在指定的索引寄存器中设置指定记录号的第一个 I/O 内存地址。GETR(636) 输出包括指定索引寄存器值 （PLC 内存地址）记录的记录号。

标志

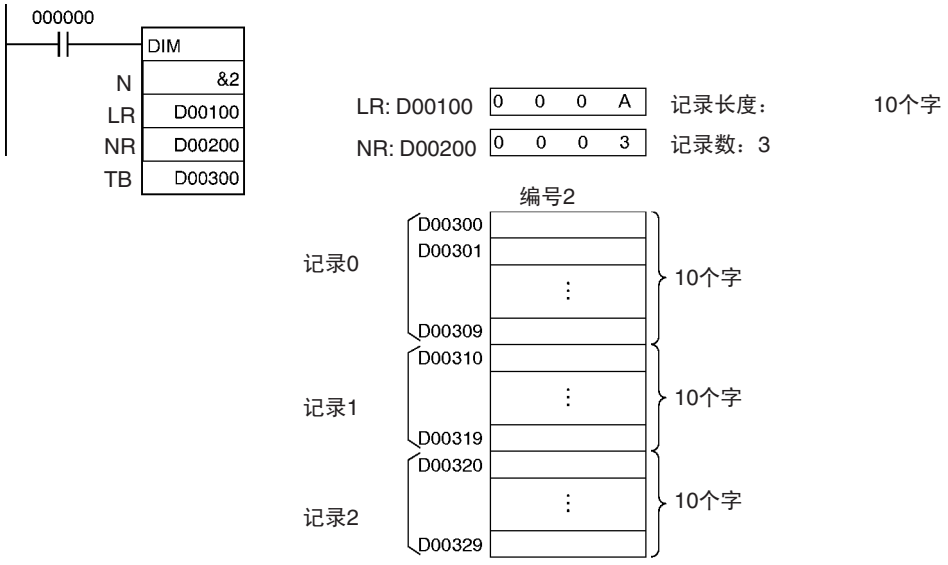
名称	标记	操作
错误标志	ER	LR 或 NR 为 0000 时置 ON。 其它情况置 OFF。

注意

注册表中的记录通过它们的记录号来识别 , 记录号的范围为 0 ～ NR-1。

例

在下例中当 CIO 00000 为 ON 时，DIN(631) 定义了一个表号为 2, 具有三个 10 字记录的记录表。该表从 D00300 开始。

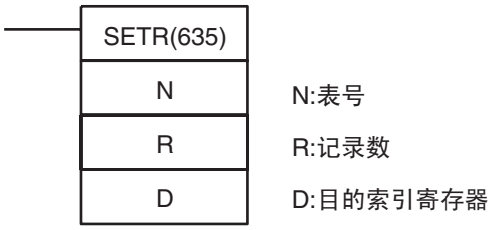


3-17-6 设置记录位置：SETR(635)

用途

将指定记录的位置 （记录开始的 PLC 内存地址）写入指定的索引寄存器。

梯形图符号



变化

变化	ON 条件时每次周期执行	SETR(635)
	上升沿微分执行一次	@SETR(635)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

N: 表号
指示表号， N 为 0 ～ 15。

R: 记录号
指示所需记录的记录号， R 必须为 0000 ～ FFFF 之间的 16 进制 (0 ～ 65 535)。记录号从 0 开始，因此 NR 个记录的有效表的记录号为 0 ～ NR-1。

D: 目标索引寄存器
指示所需的索引寄存器。 D 必须在 IR0 ～ IR15 之间。

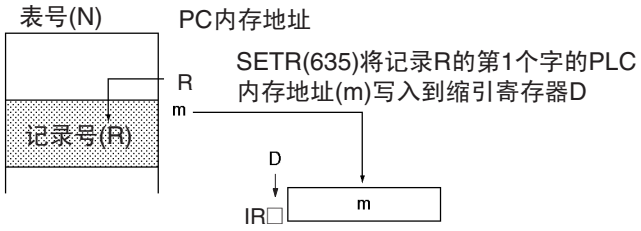
操作数规定

区域	N	R	D
CIO 区	---	CIO 0000 ～ CIO 6143	---
工作区	---	W000 ～ W511	---
保持位区	---	H000 ～ H511	---
辅助位区	---	A000 ～ A959	---
定时器区	---	T0000 ～ T4095	---
计数器区	---	C0000 ～ C4095	---
DM 区	---	D00000 ～ D32767	---
无区号 EM 区	---	E00000 ～ E32767	---
有区号 EM 区	---	En_00000 ～ En_32767 (n = 0 ～ C)	---
二进制间接 DM/EM 地址	---	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767	---
BCD 间接 DM/EM 地址	---	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767	---
常数	0 ～ 15	#0000 ～ #FFFF（二进制）或 &0 ～ 65535	---
数据寄存器	---	DR0 ～ DR15	---

区域	N	R	D
索引寄存器	---		IR0 ～ IR15
使用索引寄存器的间接寻址	---	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,– (– –)IR0 ～ ,– (– –)IR15	---

说明

SETR(635) 将指定记录的第一个字的 PLC 内存地址存入指定的索引寄存器。下图显示了 SETR(635) 的基本操作。



标志

名称	标记	操作
错误标志	ER	指定表号 (N) 未用 DIM (631) 定义时置 ON 指定记录号超过表记录号最大值 (NR-1) 时置 ON。 其它情况置 OFF。

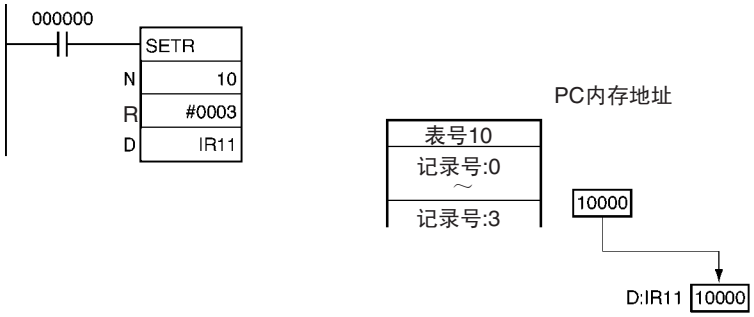
注意

记录表必须用 DIM(631) 顶先定义。

可用的记录号在 0 ～ NR-1 之间，其中 NR 是在表通过 DIM (631) 定义时指定的记录数。

例

在下例中当 CIO000000 为 ON 时，SETR(635) 找出表 10 的记录 3 的第 1 个字的 PLC 内存地址 , 并将它存入索引寄存器 IR11。

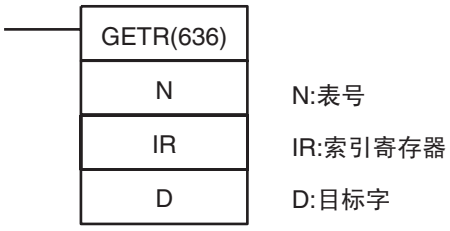


3-17-7 获得记录号：GETR(636)

用途

回收包含在指定索引寄存器中记录 PLC 内存地址的记录号。

梯形图符号



变化

变化	ON 条件时每次周期执行	GETR(636)
	上升沿微分执行一次	@GETR(636)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

N: 表号
指示表号， N 为 0 ～ 15。

IR: 索引寄存器
指示所需索引寄存器， D 必须在 IR0 ～ IR15 范围内。

D: 目标字
指示将记录号写入的字。

操作数规定

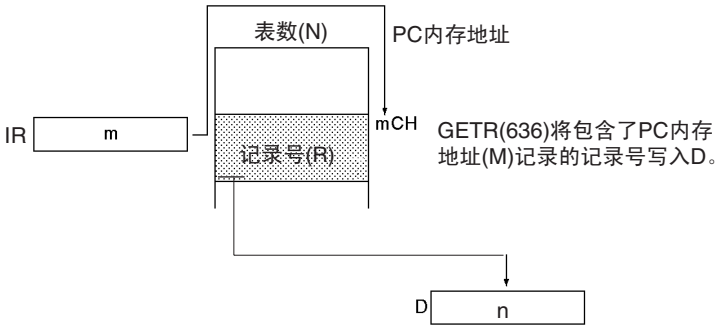
区域	N	IR	D
CIO 区	---		CIO 0000 ～ CIO 6143
工作区	---		W000 ～ W511
保持位区	---		H000 ～ H511
辅助位区	---		A448 ～ A959
定时器区	---		T0000 ～ T4095
计数器区	---		C0000 ～ C4095
DM 区	---		D00000 ～ D32767
无区号 EM 区	---		E00000 ～ E32767
有区号 EM 区	---		En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	---		@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000～@ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 地址	---		*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	0 ～ 15	---	---

区域	N	IR	D
数据寄存器	---		DR0 ～ DR15
索引寄存器	---	IR0 ～ IR15	---
使用索引寄存器的间接寻址	---		,IR0 ～ ,IR15 –2048～+2047 ,IR0～–2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15

说明

GETR(636) 找出在指定的索引寄存器中包含了 PLC 内存地址的记录，并将记录号写入 D。索引寄存器中包含的 PLC 内存地址不必是记录的第一个字，它可以是记录中的任何一个字。

下图显示了 GETR(636) 的基本操作。



标志

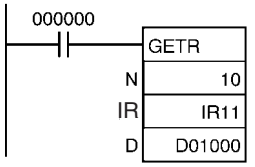
名称	标记	操作
错误标志	ER	指定索引寄存器中的 PLC 内存地址不在指定范围内时置 ON。 指定表号 (N) 没有用 DIM(631) 定义时置 ON。 其它情况置 OFF。

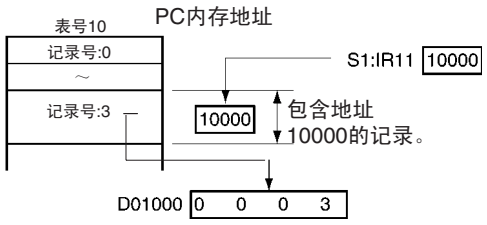
注意

记录表必须由 DIM (631) 预先定义，指定索引寄存器中的 PLC 内存地址必须在指定表范围内。

例

在下例中当 CIO000000 为 ON 时 GETR(636) 找出在索引寄存器 IR11 中包含了 PLC 内存地址记录的记录号，并将它写入 D01000 。



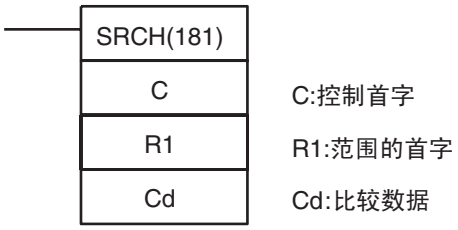


3-17-8 数据搜索：SRCH(181)

用途

从指定了范围的字中查找某字的数据 。
对于 CS1-H,CJ1-H 与 CJ1M CPU 单元，这条指令可以在后台运行，有关后台操作的详细情况，请参考 *CS/CJ 系列可编程控制器编程手册*。

梯形图符号



变化

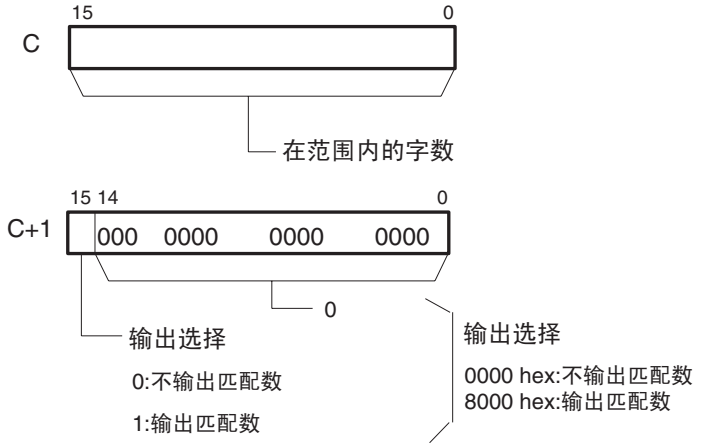
变化	ON 条件时每次周期执行	SRCH(181)
	上升沿微分执行一次	@SRCH(181)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

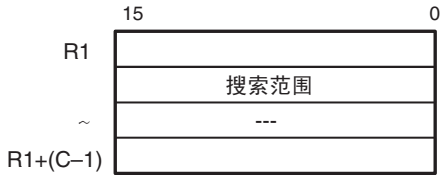
操作数

C 和 C+1：控制字
C 指定了范围的字数 ,C+1 的位 15 指示是否向 DR00 输出匹配的数。



注 C 和 C+1 必须位于同一数据区中。

R1：范围内的第 1 个字
RI 指定搜索范围中的第 1 个字 , 在 R1 到 R1+(C-1) 的范围内搜索所要的数据。（C 为设置在 C 中的字数）。



注 R1 和 R1+C-1 必须在同一数据区域中。

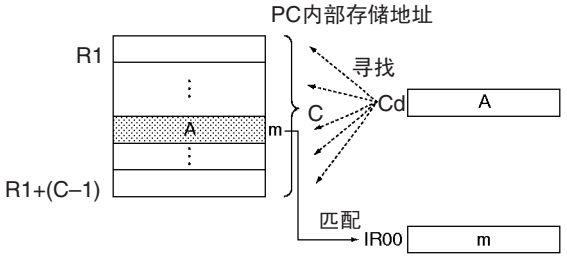
操作数规定

区域	C	R1	Cd
CIO 区	CIO 0000 ～ CIO 6142	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W510	W000 ～ W511	
保持位区	H000 ～ H510	H000 ～ H511	
辅助位区	A000 ～ A958	A000 ～ A959	
定时器区	T0000 ～ T4094	T0000 ～ T4095	
计数器区	C0000 ～ C4094	C0000 ～ C4095	
DM 区	D00000～D32766	D00000 ～ D32767	
无区号 EM 区	E00000～E32766	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	仅指定的值	---	#0000 ～ #FFFF（二进制）
数据寄存器	---		DR0 ～ DR15
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15		

说明

SRCH(181) 在 R1 到 R1+C-1 的内存范围内搜索含有比较数据 (CD) 的字。如果找到一个匹配字 ,SRCH(181) 将该字的 PLC 内存地址写入 IR00, 并将等于标志置为 ON。（如果有 2 个或更多的匹配字 , 只有第一个含有比较数据的字被写入 IR00）。

当 C+1 的位 15 被置为 1 时 SRCH(181) 将匹配的数写入 DR00。如果 C+1 的位 15 为 0， DR00 保留不变。



SRCH(181) 可搜索每个记录含一个字的表数据。对于每个记录包含多个字的数据搜索，用 DIM(631)，SETR(635)，GETR(636)，FOR(512)—NEXT(513) 或 BREAK(514) 配合索引寄存器 (IR) 。

执行指令后可利用等于标志的状态立即检查是否有匹配字。

标志

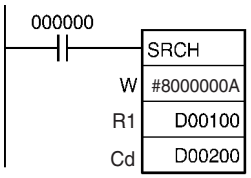
名称	标记	操作
错误标志	ER	如果 C 的内容不在规定的 0001 ～ FFFF 的范围内时置 ON。 其它情况置 OFF。
等于标志	=	如果在搜索范围包含一个或多个含有比较数据的字时置 ON。 其它情况置 OFF。

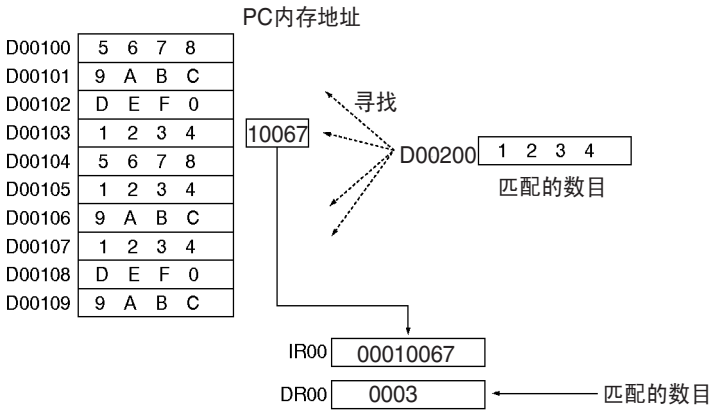
注意

如果找不到匹配字， IR00 和 DR00 的内容不变。

例

在下例中当 CIO000000 为 ON 时 ,SRCH(181) 在从 D00100 开始的 10 字范围中搜索与D00200内容相同的字。第一个包含匹配字的PLC内存地址写入IR00，而匹配字的总数写入 DR00。





如果表格长度指定为 &10（10 进制）或 A 十六进制 , 匹配数将不被输出到数据寄存器 DR00 。

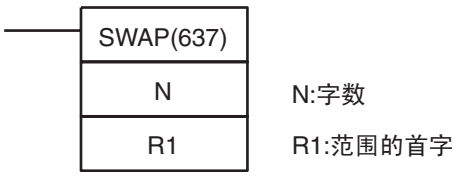
3-17-9 交换字节：SWAP(637)

用途

将范围内的所有字的左字节和右字节交换。

对于 CS1-H,CJ1-H 与 CJ1M CPU 单元这条指令可以在后台运行，有关后台执行详细情况参考 *CS/CJ 系列可编程控制器编程手册*。

梯形图符号



变化

变化	ON 条件时每次周期执行	SWAP(637)
	上升沿微分执行一次	@SWAP(637)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

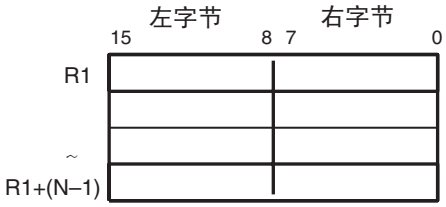
适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

N: 字数
N 指定了范围中的字数 , 它必须是 0001 ～ FFFF 之间的 16 进制数 （或 &1 ～ &65,535）。

R1: 范围内的首字
R1 指定了范围内的第一个字， R1 和 R1+(N-1) 必须在同一数据区内。



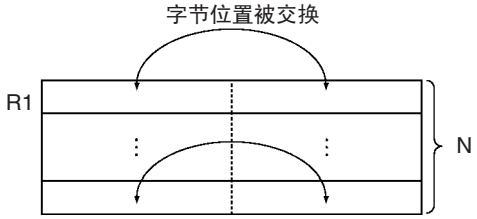
操作数规定

区域	N	R1
CIO 区	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W511	
保持位区	H000 ～ H511	
辅助位区	A000 ～ A959	A448 ～ A959
定时器区	T0000 ～ T4095	
计数器区	C0000 ～ C4095	
DM 区	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#0001 ～ #FFFF （二进制） 或 &1 ～ &65,535	---
数据寄存器	DR00 ～ DR15	---

区域	N	R1
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15	

说明

SWAP(637) 交换从 R1 到 R1+N- 1 内存范围内的所有字的两个位置的字节，该指令可被用来颠倒每个字中的 ASCII 码字符的顺序。

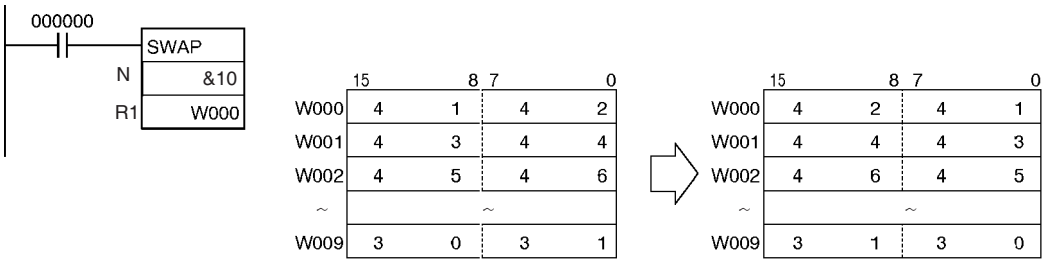


标志

名称	标记	操作
错误标志	ER	N 为 0000 时置 ON。 其它情况置 OFF。

例

在下例中当 CIO 000000 为 ON 时 ,SWAP(637) 交换从 W000 ～ W009 范围内 10 个字的每个字的左字节和右字节的位置。



3-17-10 查找最大值：MAX(182)

用途

查找指定范围内的最大值。
对于 CS1-H,CJ1-H 与 CJ1M CPU 单元这条指令可以在后台运行，后台操作的详 细情况参考 *CS/CJ 系列可编程控制器编程手册*。

梯形图符号

变化

变化	ON 条件时每次周期执行	MAX(182)
	上升沿微分执行一次	@MAX(182)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

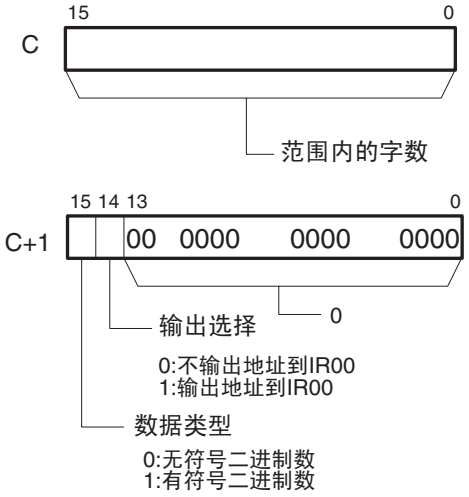
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

C 和 C+1：控制字

C 指定了查找范围内的字数， C+1 的第 15 位指示数据是按有符号二进制数还是按无符号二进制数处理。 C+1 的第 14 位指示是否将含有最大值的字的 PLC 内存地址放入 IR00。

注 C 和 C+1 必须在同一数据区内。

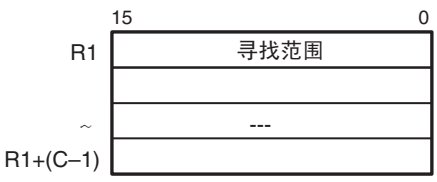


下表为 C 的可选值：

C+1	数据类型	索引寄存器输出
0000	无符号二进制数	No
4000	无符号二进制数	Yes
8000	有符号二进制数	No
C000	有符号二进制数	Yes

R1: 范围的首字

R1 指定了查找范围的第 1 个字，搜索最大值是在从 R1 ～ R1+(C-1) 的范围内的字中进行的 （C 为存放在 C 中的字数）。



注 R1 和 R1+C-1 必须在同一数据区。

操作数规定

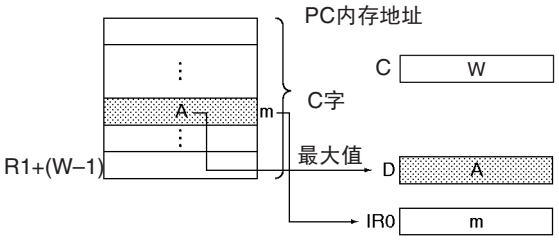
区域	C	R1	D
CIO 区域	CIO 0000～CIO 6142	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W510	W000 ～ W511	
保持位区	H000 ～ H510	H000 ～ H511	
辅助位区	A000 ～ A958	A000 ～ A959	A448 ～ A959
定时器区	T0000 ～ T4094	T0000 ～ T4095	
计数器区	C0000 ～ C4094	C0000 ～ C4095	
DM 区	D00000 ～ D32766	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32766	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	仅指定的值	---	
数据寄存器	---		DR0 ～ DR15
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15		

说明

MAX(182) 从 R1 到 R1+C-1 的内存范围内查找范围内的最大值 , 并将最 大值输出到 D。

当 C+1 的第 14 位被设置为 1 时，MAX(182) 将含有最大值的字的 PLC 内存地址写入 IR00（如果在查找范围内有两个或两个以上的含有最大值的字 , 第一个查找到的含有最大值的字的地址写入 IR00）。

当 C+1 的第 15 位被设置为 1 时 ,MAX(182) 将查找范围内的数据按有符号二进制数据处理。



标志

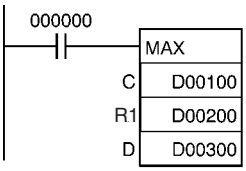
名称	标记	操作
错误标志	ER	如果 C 的内容不在规定的 0001 ～ FFFF 的范围内时置 ON。 其它情况置 OFF。
等于标志	=	如果最大值为 0000 时置 ON。 其它情况置 OFF。
负标志	N	最大值的第 15 位为 ON 时置 ON。 其它情况置 OFF。

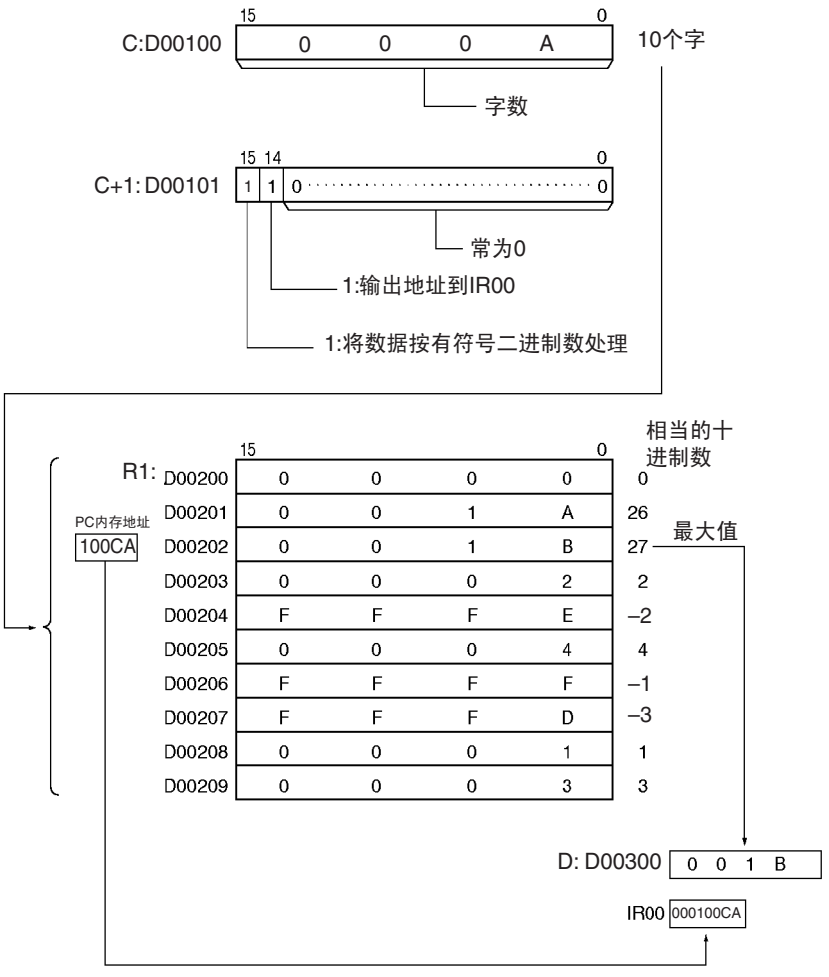
注意

当 C+1 的第 15 位被设置为 1 时，查找范围内的数据按有符号二进制数处理，十六进制值 8000 ～ FFFF 为负。查找结果也因数据类型设置的不同而不同。

例

当下例 CIO00000 为 ON 时，MAX(182) 从 D00200 开始的 10 字范围内搜索最大值。最大值写入 D00300，含有最大值的字的 PLC 内存地址写入 IR00。



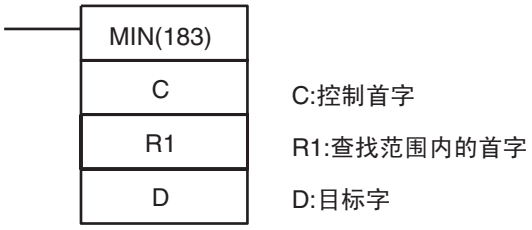


3-17-11 查找最小值：MIN(183)

用途

查找指定范围内的最小值。
对于 CS1-H,CJ1-H 与 CJ1M CPU 单元这条指令可以在后台运行，后台操作的详 细情况参考 *CS/CJ 系列可编程控制器编程手册*。

梯形图符号



变化

变化	ON 条件时每次周期执行	MIN(183)
	上升沿微分执行一次	@MIN(183)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

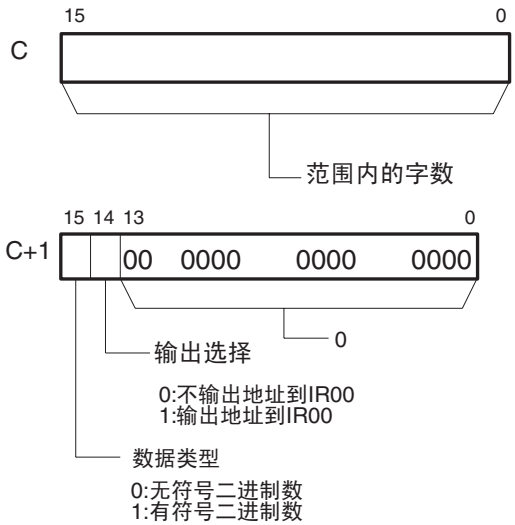
适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

C 和 C+1：控制字
C 指定了查找范围内的字数， C+1 的第 15 位指示数据是按有符号二进制数还是按无符号二进制数处理。 C+1 的第 14 位指示是否将含有最大值的字的 PLC 内存地址存入 IR00。

注 C 和 C+1 必须在同一数据区内。

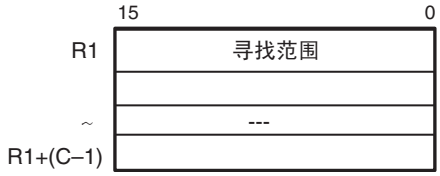


下表显示了 C 的可选值：

C+1	数据类型	索引寄存器输出
0000	无符号二进制数	No
4000	无符号二进制数	Yes

C+1	数据类型	索引寄存器输出
8000	有符号二进制数	No
C000	有符号二进制数	Yes

R1: 范围的首字
R1 指定了查找范围的第 1 个字，搜索最小值是在从 R1 ～ R1+(C-1) 的范围内的字中进行的 （C 为存放在 C 中的字数）。



注 R1 和 R1+C-1 必须在同一数据区。

操作数规定

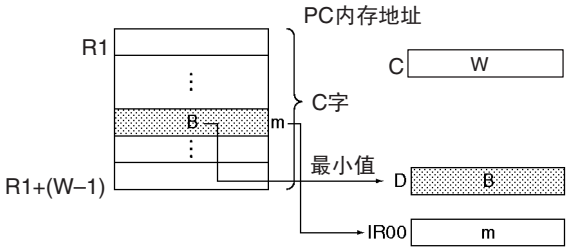
区域	C	R1	D
CIO 区	CIO 0000 ～ CIO 6142	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W510	W000 ～ W511	
保持位区	H000 ～ H510	H000 ～ H511	
辅助位区	A000 ～ A958	A000 ～ A959	A448 ～ A959
定时器区	T0000 ～ T4094	T0000 ～ T4095	
计数器区	C0000 ～ C4094	C0000 ～ C4095	
DM 区	D00000～D32766	D00000 ～ D32767	
无区号 EM 区	E00000～E32766	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D0000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	仅指定的值	---	
数据寄存器	---		DR0 ～ DR15
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

MIN(183) 从 R1 ～ R1+C-1 的内存范围内查找范围内的最小值，并将最小值输出到 D。

当 C+1 的第 14 位被设置为 1 时， MIN(183) 将含有最小值的字的 PLC 内存地址写入 IR00（如果在查找范围内有两个或两个以上的含有最小值的字，第一个查找到的含有最小值的字的地址写入 IR00）。

当 C+1 的第 15 位被设置为 1 时，MIN(183) 将查找范围内的数据按有符号二进制数据处理。



标志

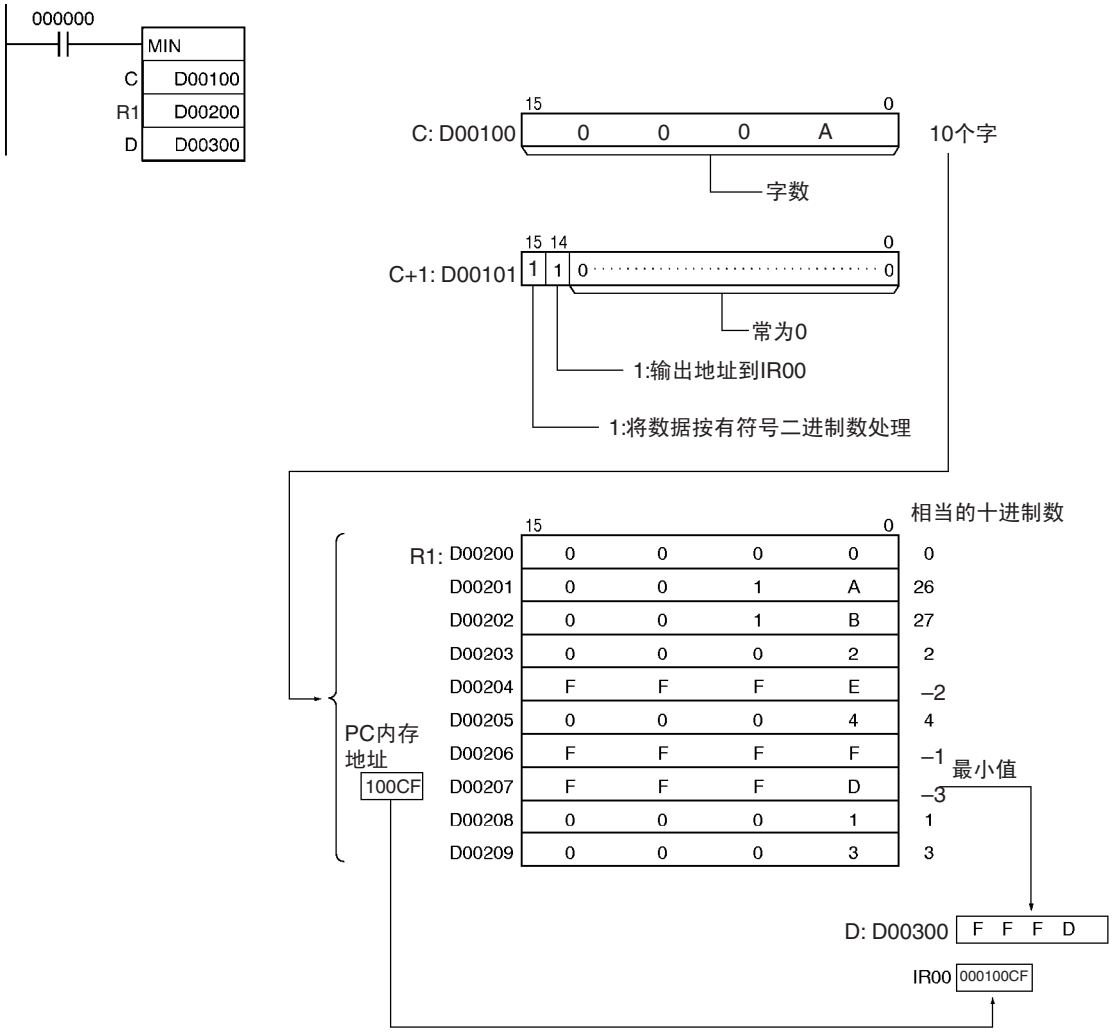
名称	标记	操作
错误标志	ER	如果 C 的内容不在规定的 0001 到 FFFF 的范围内时置 ON。 其它情况置 OFF。
等于标志	=	如果最小值为 0000 时置 ON。 其它情况置 OFF。
负标志	N	最小值的第 15 位为 ON 时置 ON。 其它情况置 OFF。

说明

当 C+1 的第 15 位被设置为 1 时，查找范围内的数据按有符号二进制数处理，十六进制值 8000 ～ FFFF 为负。查找结果也因数据类型设置的不同而不同。

例

当下例 CIO00000 为 ON 时 ,MIN(183) 从 D00200 开始的 10 字范围内搜索最小值。最小值写人 D00300, 含有最小值的字的 PLC 内存地址写入 IR00。

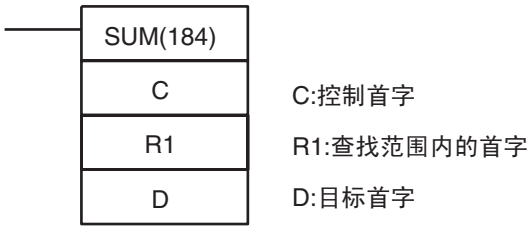


3-17-12 求和：SUM(184)

用途

将范围内的字节或字相加，并将结果输出到两个字中。

梯形图符号



变化

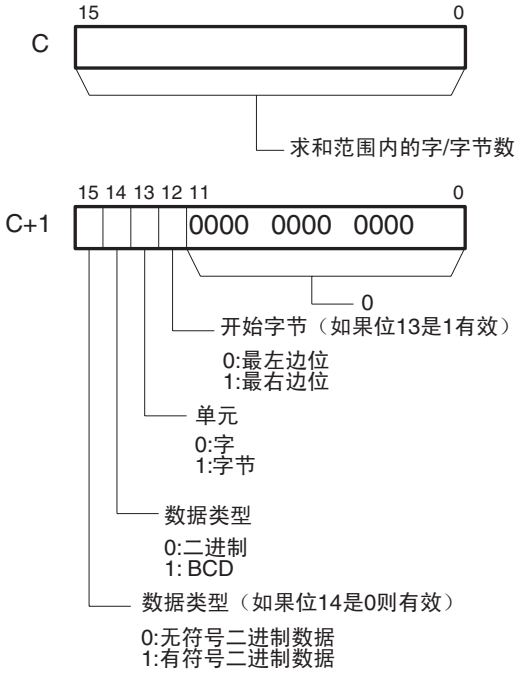
变化	ON 条件时每次周期执行	SUM(184)
	上升沿微分执行一次	@SUM(184)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

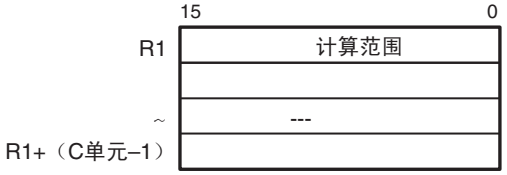
操作数

C 和 C+1：控制字
C 指定求和的单元数（字节或字），（C+1 的第 13 位判断是字节还是字求和）。
C+1 的 12 ～ 15 位指示了求和数据的数据类型，如下图所示。



注 C 和 C+1 必须在同一数据区内。

R1: 求和范围中的首字
R1 指定了求和范围中的第一个字，范围的长度取决于单位数量，如果是字节求和，则还取决于开始字节。



注 计算范围内的所有字必须在同一数据区内。

D: 目标首字
计算结果输出到 D+1 和 D，左边 4 个数存入 D+1，右边 4 个数存入 D。

操作数规定

区域	C	R1	D
CIO 区	CIO 0000 ～ CIO 6142	CIO 0000 ～ CIO 6143	CIO 0000 ～ CIO 6142
工作区	W000 ～ W510	W000 ～ W511	W000 ～ W510
保持位区	H000 ～ H510	H000 ～ H511	H000 ～ H510
辅助位区	A000 ～ A958	A000 ～ A959	A448 ～ A958
定时器区	T0000 ～ T4094	T0000 ～ T4095	T0000 ～ T4094
计数器区	C0000 ～ C4094	C0000 ～ C4095	C0000 ～ C4094
DM 区	D00000～D32766	D00000～D32767	D00000～D32766

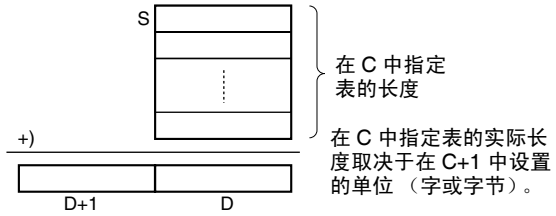
区域	C	R1	D
无区号 EM 区	E00000～E32766	E00000～E32767	E00000～E32766
有区号 EM 区	En_00000～En_32766 (n = 0～C)	En_00000～En_32767 (n= 0～C)	En_00000～En_32766 (n = 0～C)
二进制间接 DM/EM 地址	@ D00000～@ D32767 @ E00000～@ E32767 @ En_00000～@ En_32767 (n = 0～C)		
BCD 间接 DM/EM 地址	*D00000～*D32767 *E00000～*E32767 *En_00000～*En_32767 (n = 0～C)		
常数	仅指定的值	---	
数据寄存器	---		
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0～,IR15 –2048～+2047 ,IR0～–2048～+2047 ,IR15 DR0～DR15, IR0～IR15 ,IR0+(++)～,IR15+(++) ,–(– –)IR0～,–(– –)IR15		

说明

SUM(184) 将从 R1 中的数据开始的 C 单位数相加，并将计算结果输出到 D+1 和 D。由 C+1 的设置决定单位是字还是字节、数据是二进制数（有符号数或无符号数）还是 BCD 码。如果是字节求和，决定是从 R1 的左字节开始还是右字节开始。

当 C+1 的第 14 位置为 0 时， SUM(184) 将数据按二进数处理。此时，第十五位决定数据是有符号数 （第 15 位为 1）还是无符号数 （第 15 位为 0）。

当 C+1 的第 13 位置为 1 时，SUM(184) 将数据的字节相加，此时第 12 决定计算是从 RI 的最右字节 （第 12 位为 1）还是最左字节 （第 12 位为 0）开始。

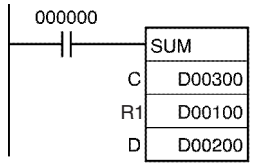


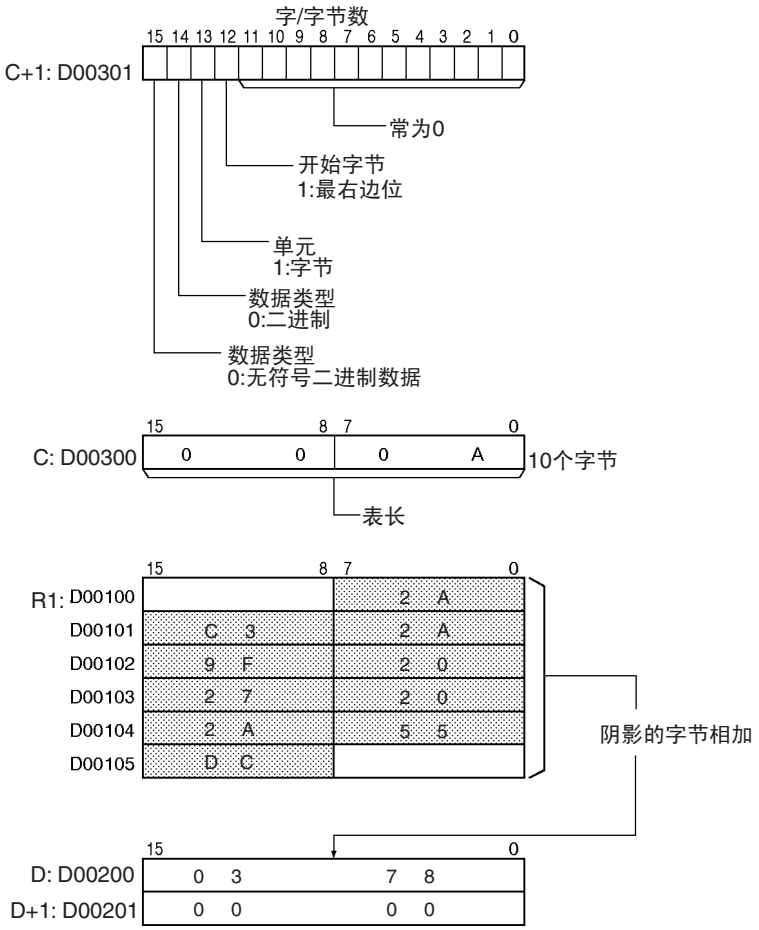
标志

名称	标记	操作
错误标志	ER	如果 C 的内容不在规定的 0001～FFFF 的范围置 ON。 如果指定为 BCD 码，但求和范围内有二进制码数据时置 ON。 其它情况置 OFF。
等于标志	=	计算结果为 0000 时置 ON。 其它情况置 OFF。
负标志	N	计算结果的第 15 位为 ON 时置 ON。 其它情况置 OFF。

例

在下例中当 CIO000000 为 ON 时，SUM(184) 将从 D00100 的最右字节开始的 10 个字节无符号二进制数相加。并将结果写入 D00201 ～ D00200。



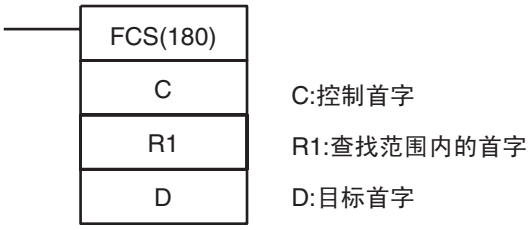


3-17-13 帧校验和：FCS(180)

用途

计算指定范围的 ASCII FCS 值并以 ASCII 输出。

梯形图符号



变化

变化	ON 条件时每次周期执行	FCS(180)
	上升沿微分执行一次	@FCS(180)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

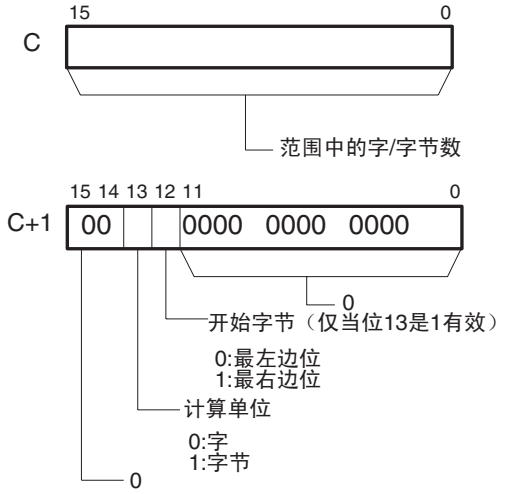
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

C 和 C+1：控制字

C 指定了在 FCS 计算中使用的单位 （字节或字），（C+1 的第 13 位判断是字节还是字）。

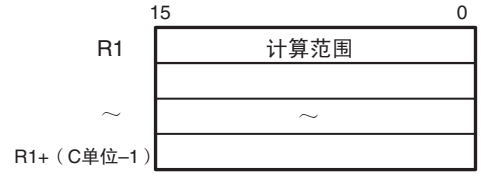
当 C+1 的 13 位被设置为 1 时， FCS(180) 计算数据字节的 FCS 值。此时，第 12 位用来决定计算是从 R1 的最右字节开始 （第 12 位等于 1）还是从 R 的最左字节开始 （第 12 位等于 0）。



注 C 和 C+1 必须在同一数据区内。

R1: 范围中的首字

R1 指定了范围中的第一字，范围的长度取决于单位数量，如果是字节求和，则还取决于开始字节。



注 计算范围内的所有字必须在同一数据区内。

D: 目标首字

如果选择字节方式，结果输出到 D。

如果选择字节方式，计算结果输出到 D+1 和 D。在这种情况下，左边 4 个数字存入 D+1, 右边 4 个数字存入 D 。

操作数规定

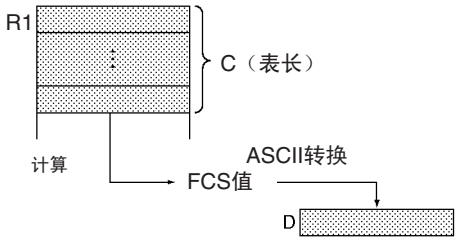
区域	C	R1	D
CIO 区	CIO 0000 ～ CIO 6142	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W510	W000 ～ W511	
保持位区	H000 ～ H510	H000 ～ H511	
辅助位区	A000 ～ A958	A000 ～ A959	A448 ～ A959
定时器区	T0000 ～ T4094	T0000 ～ T4095	

区域	C	R1	D
计数器区	C0000 ～ C4094	C0000 ～ C4095	
DM 区	D00000～D32766	D00000 ～ D32767	
无区号 EM 区	E00000～E32766	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	En_0000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	仅指定的值	---	
数据寄存器	---		
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15		

说明

FCS(180) 计算从 R1 开始的 C 单位数的 FCS 值，并将该值转换成 ASCII 码，然后将计算结果输出到 D（对于字节）或 D+1 和 D（对于字），C+1 的设置决定单位是字还是字节，数据是二进制数 （有符号或无符号）或 BCD 码，如果相加的是字节 , 决定从 R1 的左字节还是右字节开始相加。

当 C+1 的第 13 位被设置为 1 时，FCS(180) 对字节数据操作。此时第 12 位决定是从R1的最右位（第12位等于1）还是从R1的最左位（第12位等于0）开始。

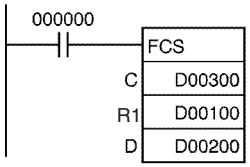


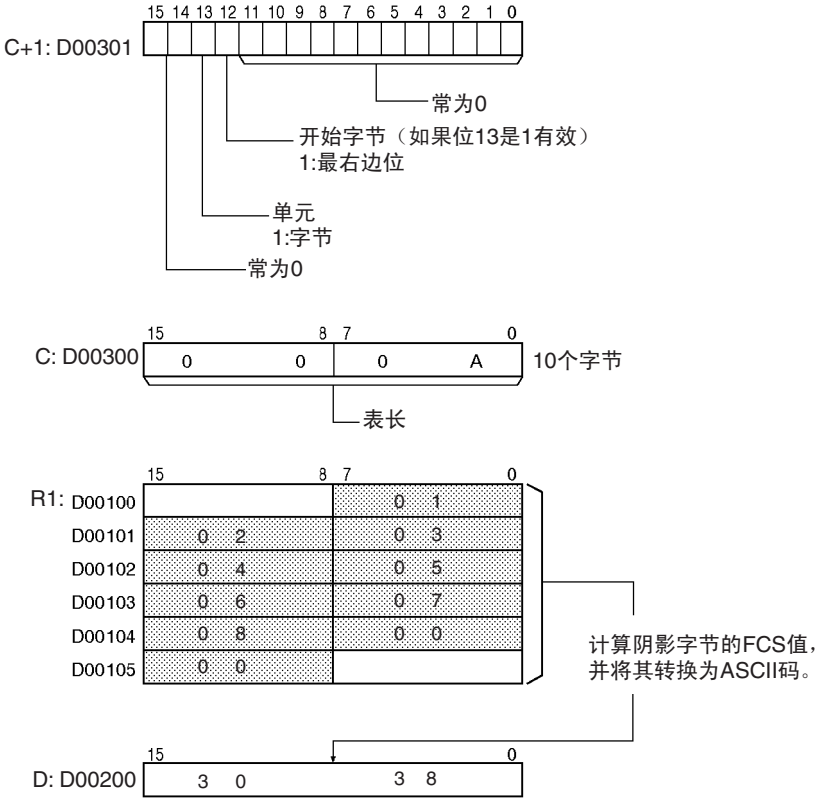
标志

名称	标记	操作
错误标志	ER	如果 C 的内容不在指定的 0001 ～ FFFF 的范围内时置 ON。 其它情况置 OFF。

例

当下例中 CIO00000 为 ON 时， FCS(180) 计算从 D00100 的最右字节开始的 10 字节的 FCS 值，并把结果写入 D00200。





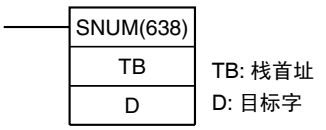
3-17-14 读取堆栈大小：SNUM(638)

用途

计算指定堆栈数据 （字数）数的多少。

该指令仅由 CS1-H， CJ1-H， CJ1M 和 CS1D CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每次周期执行	SNUM(638)
	上升沿微分执行一次	@SNUM(638)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

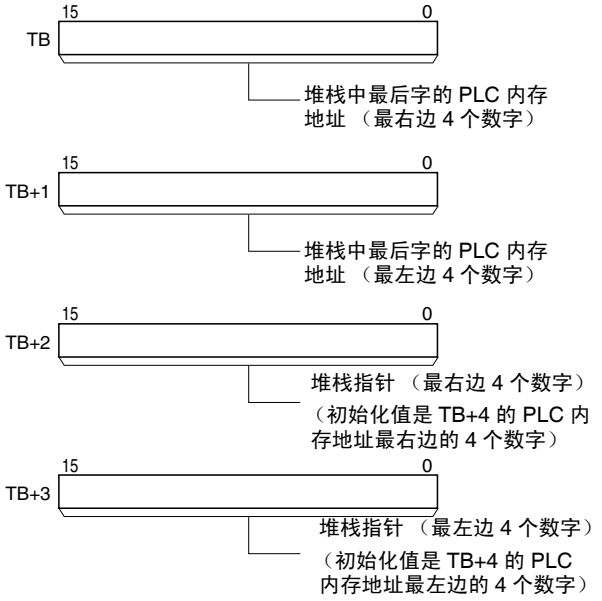
适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

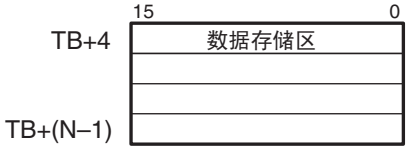
TB ～ TB+3：堆栈控制字

堆栈开始的 4 个字包含了堆栈内最后字的 PLC 内存地址和堆栈指针 （堆栈中下一个有效字的 PLC 内存地址）。



TB+4 ～ TB+(N-1)：数据存储范围

堆栈的余下部分用来存储数据。



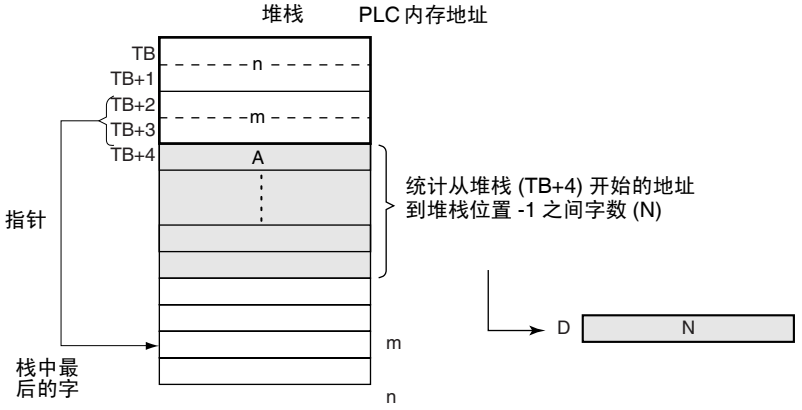
操作数规定

区域	TB	D
CIO 区	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W511	
保持位区	H000 ～ H511	
辅助位区	A448 ～ A959	
定时器区	T0000 ～ T4095	
计数器区	C0000 ～ C4095	
DM 区	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	

区域	TB	D
数据寄存器	---	DR0 ～ DR15
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15	

说明

SNUM(638) 统计指定堆栈中数据的字数，数据字数从 TB+4 数据区开始，一直到由堆栈指针 （TB+2 与 TB+3）指定的数据之前的地址。 SNUM(638) 不改变堆栈的数据或堆栈指针。



标志

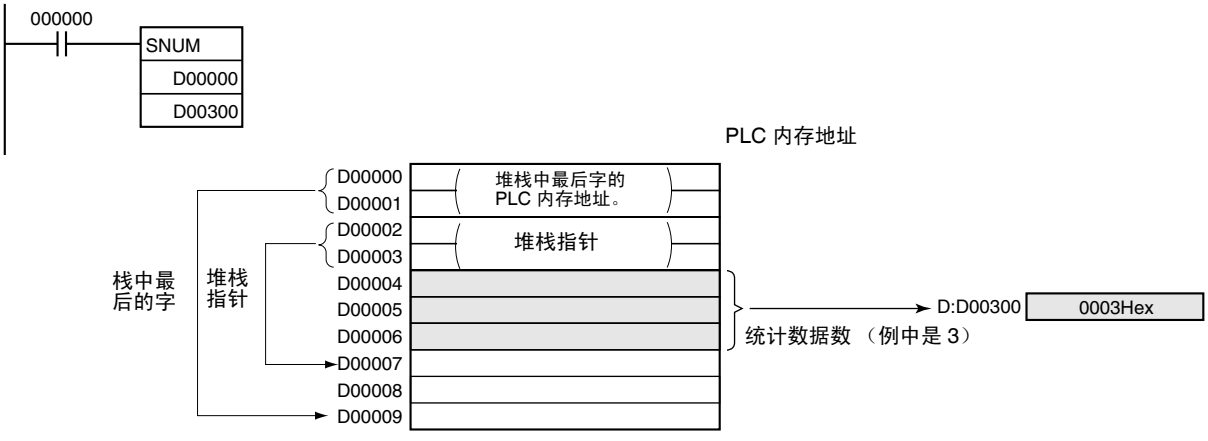
名称	标记	操作
错误标志	ER	如果在堆栈中数据字数是 0 （值输出到 D）时置 ON。 其它情况置 OFF。

注意

堆栈必须预先由 SSET(630) 定义。

例

当下例 CIO00000 为 ON 时，SNUM(638) 统计从 D00004 开始的数据区到堆栈指针位置 -1(D000006) 之间字数，并结果输出到 D00300 （此时堆栈指针指向 D00007）。



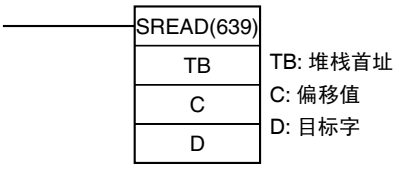
3-17-15 读取堆栈数据：SREAD(639)

用途

从堆栈中指定数据元素处读取数据。偏移值指定了所需数据元素的位置（在当前指针位置前多少数据元素）。

该指令仅由 CS1-H， CJ1-H， CJ1M 和 CS1D CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每次周期执行	SREAD(639)
	上升沿微分执行一次	@SREAD(639)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

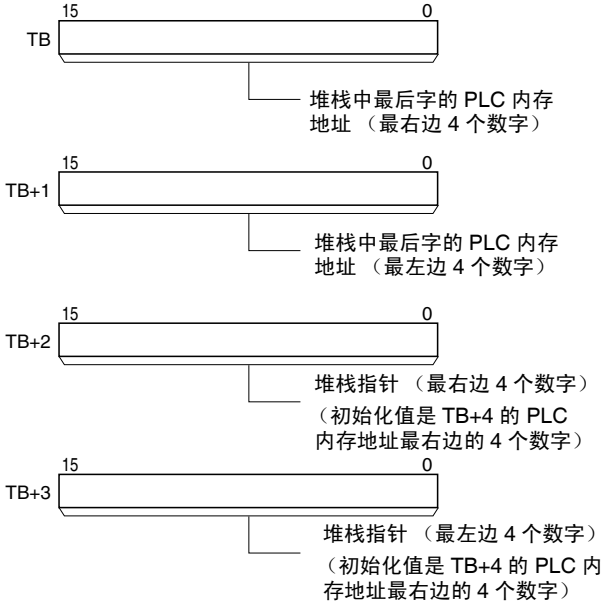
适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

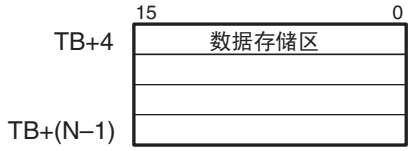
操作数

TB ～ TB+3：堆栈控制字

堆栈开始的 4 个字包含了堆栈内最后字的 PLC 内存地址和堆栈指针（堆栈中下一个有效字的 PLC 内存地址）。



TB+4 ～ TB+(N-1)：数据存储范围
堆栈的余下部分用来存储数据。

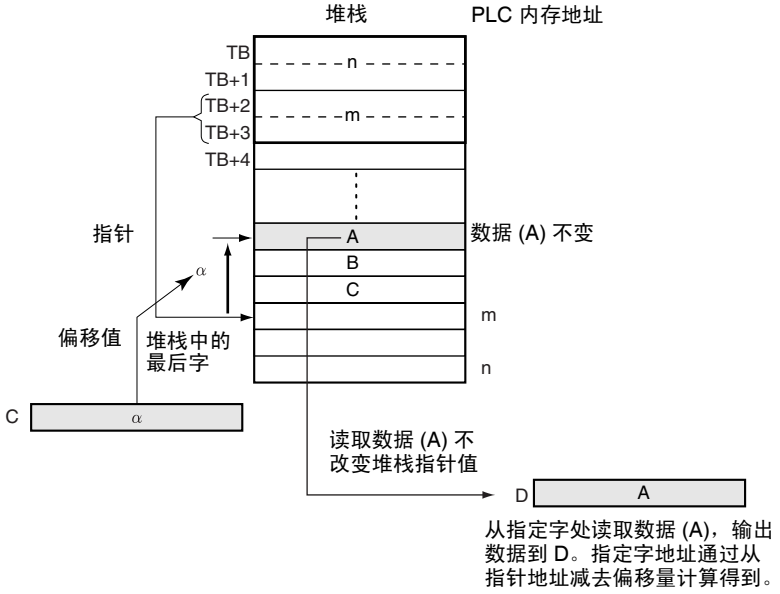


操作数规定

区域	TB	C	D
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A448 ～ A959	A000 ～ A959	A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---	#0001 ～ #FFFB （十六进制）	---
数据寄存器	---	DR0 ～ DR15	
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15		

说明

SREAD(639) 从指定堆栈中地址处读取数据，地址从（TB+3 和 TB+2）堆栈指针减去 C 中偏移量得到。 SREAD(639) 不改变堆栈中的数据或堆栈指针。



SREAD(639) 可通常用来读取传送带上的当前元素数据。所要读取的元素数据位置简单地从最当前加到传送带的元素返回几个元素 （偏移量）。

标志

名称	标记	操作
错误标志	ER	如果指定读取位置不在堆栈区时置 ON。 如果 C 中偏移值是 0 或大于最大数据范围 （FFFB 十六进制）时置 ON。 其它情况置 OFF。
等于标志	=	输出到 D 中数据为 0000 时置 ON。 其它情况置 OFF。

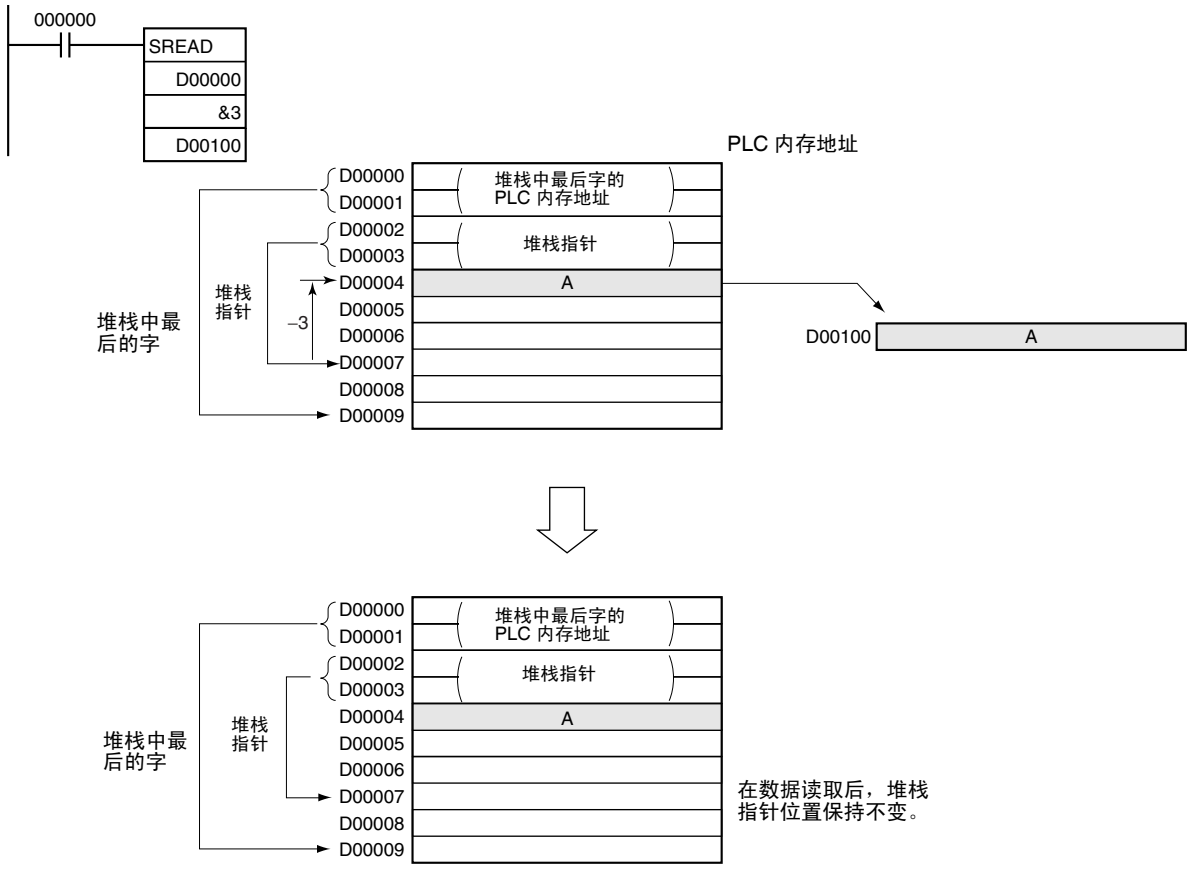
注意

堆栈必须预先由 SSET (630) 定义。

堆栈指针地址必须大于数据区开始 (TB+4) 的 PLC 内存地址。如果堆栈指针小于 TB+4 的 PLC 内存地址，将产生错误，即发生堆栈下溢错误。

例

当下例 CIO00000 为 ON 时， SREAD(639) 读取从 D00000 开始的堆栈中指定数据字并将结果输出到 D00100。此时，堆栈指针指向 D00007，偏移值是 3，因此从 D00004 读取数据。



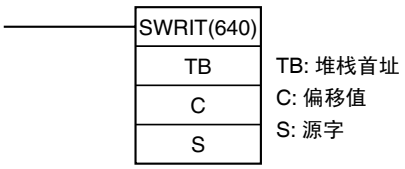
3-17-16 重写堆栈数：SWRIT(640)

用途

将源数据写到堆栈指定的数据元素处（覆盖已存在的数据）。偏移值指定了所需数据元素的位置（在当前指针位置前多少数据元素）。

该指令仅由 CS1-H， CJ1-H， CJ1M 和 CS1D CPU 支持。

梯形图符号



变化

变化	ON 条件时每次周期执行	SWRIT(640)
	上升沿微分执行一次	@SWRIT(640)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

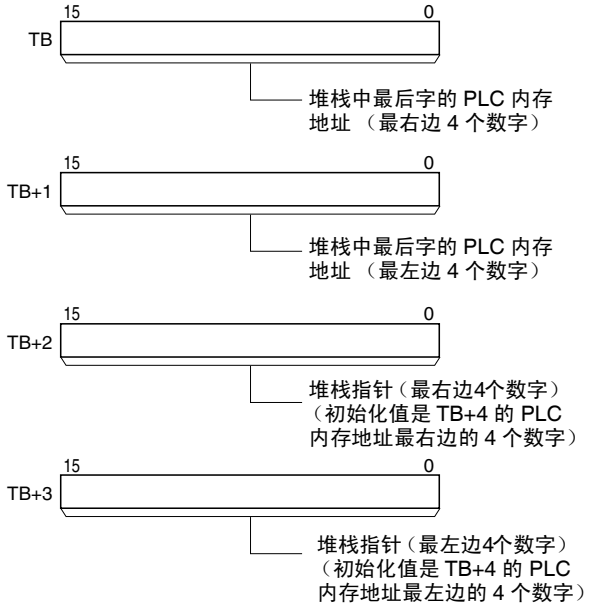
适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

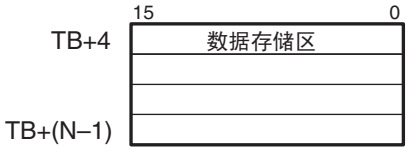
TB ～ TB+3：堆栈控制字

堆栈开始的 4 个字包含了堆栈内最后字的 PLC 内存地址和堆栈指针 （堆栈中下一个有效字的 PLC 内存地址）。



TB+4 ～ TB+(N-1)：数据存储范围

堆栈的余下部分用来存储数据。



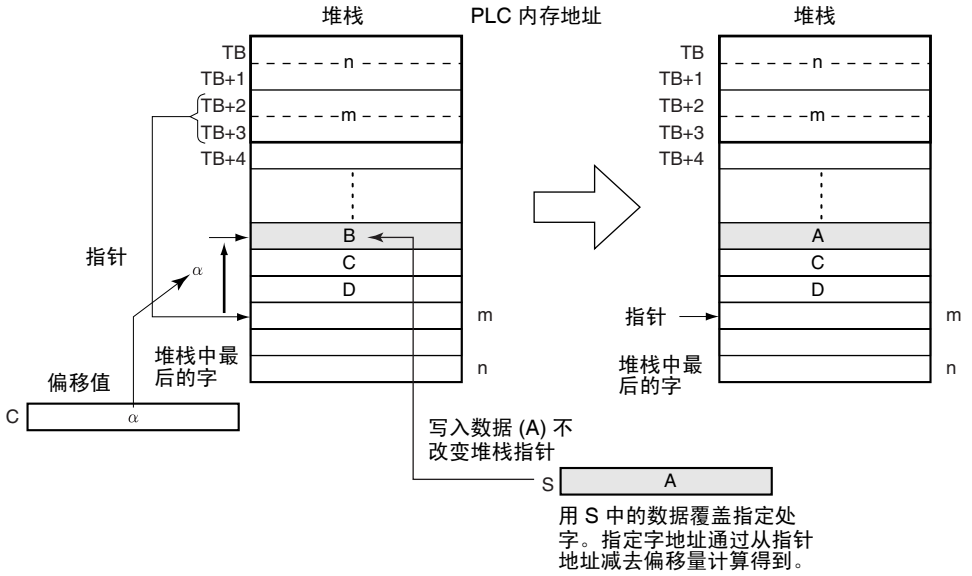
操作数规定

区域	TB	C	S
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A448 ～ A959	A000 ～ A959	
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		

区域	TB	C	S
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---	#0001 ～ #FFFB（十六进制）	#0000 ～ #FFFF（十六进制）
数据寄存器	---	DR0 ～ DR15	
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15		

说明

SWRIT(640) 将数据重写到由 S 中数据指定的字，指定字的位置从堆栈指针（TB+3 和 TB+2）减去 C 中偏移量得到。SWRIT(640) 不改变堆栈中的数据或堆栈指针。



SWRIT(640) 可用于改变在传送带上当前内容。指定器件的位置是从最新添加到传送带上的件向后条目数 （偏移量）。

标志

名称	标记	操作
错误标志	ER	如果指定读取位置不在堆栈区时置 ON。 如果 C 中偏移值是 0 或大于最大数据范围 （FFFB 十六进制）时置 ON。 其它情况置 OFF。

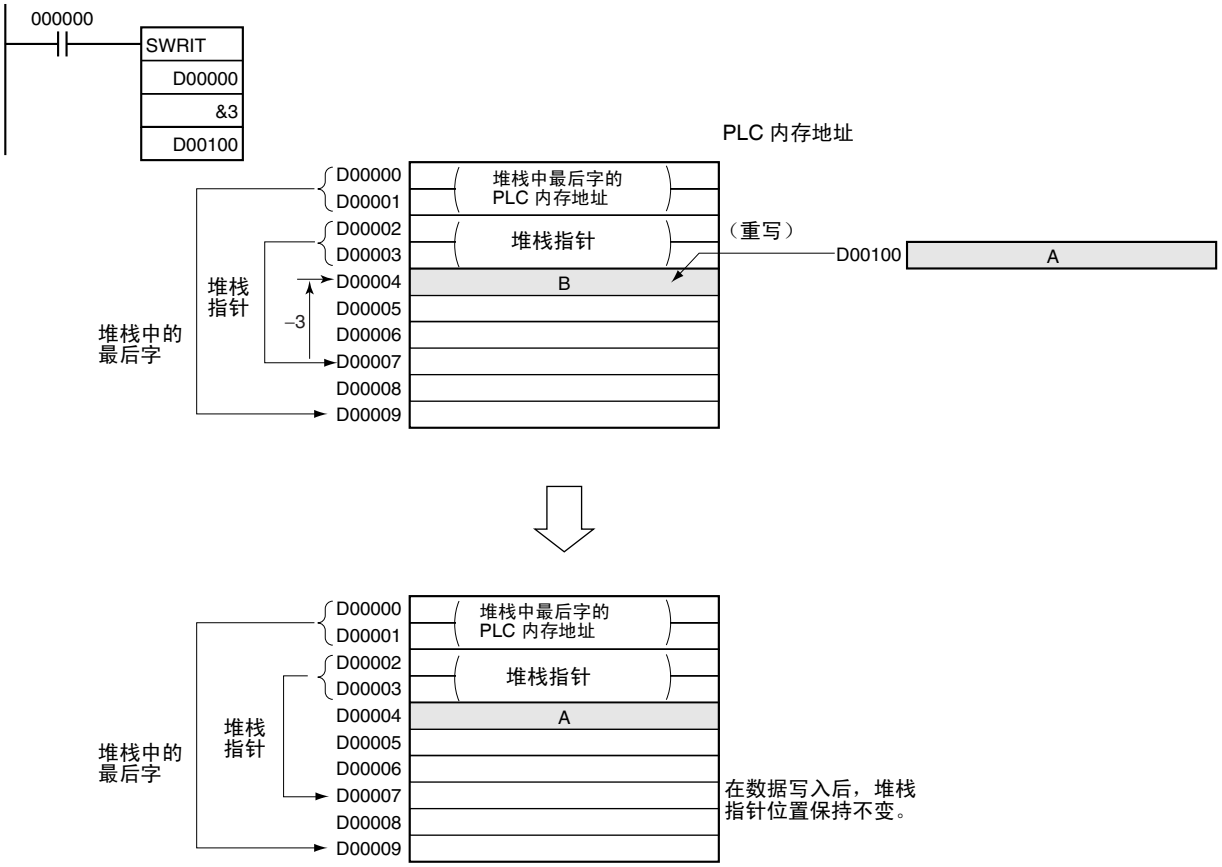
注意

堆栈必须预先由 SSET (630) 定义。

堆栈指针中的地址必须大于数据区开始 (TB+4) 的 PLC 内存地址。如果堆栈指针小于 TB+4 的 PLC 内存地址，将产生错误，即发生堆栈下溢错误。

例

当下例 CIO00000 为 ON 时，SWRIT(640) 将在 D00100 的数据写入到 D00000 开始的指定数据字处。此时，堆栈指针指向 D00007，偏移值是 3，因此，D00004 中的数据被覆盖。



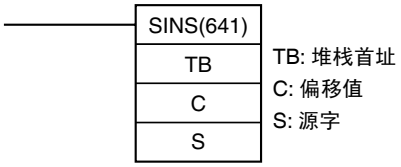
3-17-17 插入堆栈数据：SINS(641)

用途

将源数据插入到堆栈指定的位置，并将堆栈中余下的数据往下移。偏移值指定了所需数据元素处的位置（在当前指针位置前多少数据元数）。

该指令仅由 CS1-H， CJ1-H， CJ1M 和 CS1D CPU 支持。

梯形图符号



变化

变化	ON 条件时每次周期执行	SINS(641)
	上升沿微分执行一次	@SINS(641)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

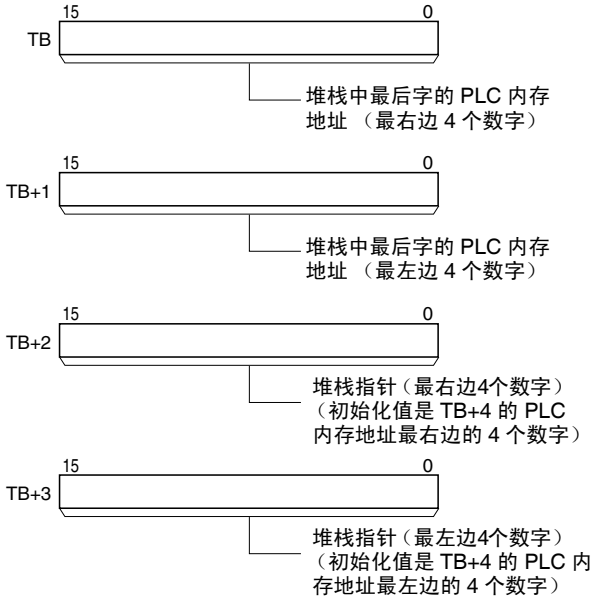
适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

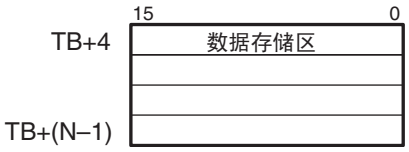
TB ～ TB+3：堆栈控制字

堆栈开始的 4 个字包含了堆栈内最后字的 PLC 内存地址和堆栈指针 （堆栈中下一个有效字的 PLC 内存地址）。



TB+4 ～ TB+(N-1)：数据存储范围

堆栈的余下部分用来存储数据。



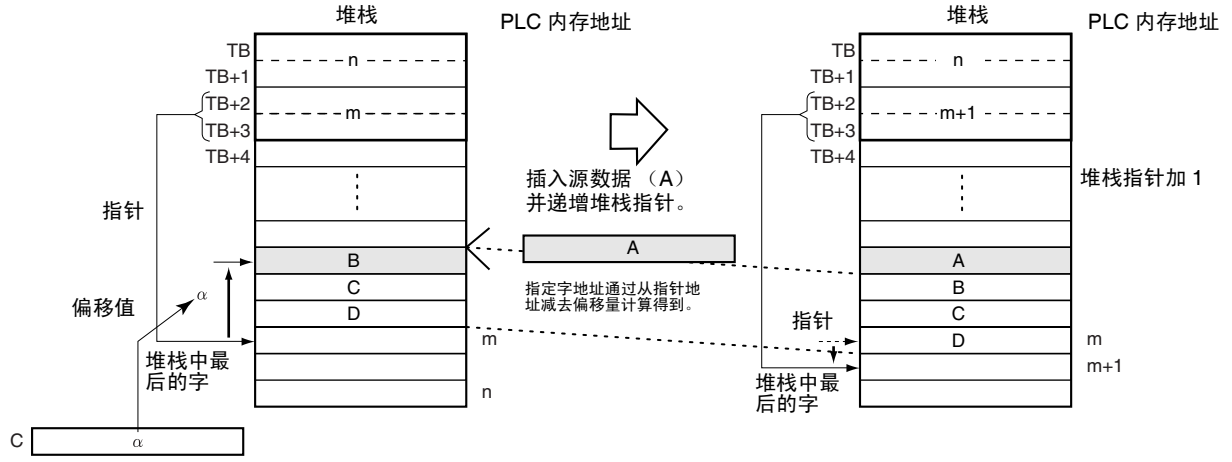
操作数规定

区域	TB	C	S
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A448 ～ A959	A000 ～ A959	
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		

区域	TB	C	S
常数	---	#0001 ～ #FFFB（十六进制）	#0000 ～ #FFFF（十六进制）
数据寄存器	---	DR0 ～ DR15	
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15		

说明

SINS(641) 在指定地址插入源数据，将其它数据往下移动一个字。同时，SINS(641) 使堆栈指针 （TB+3 和 TB+2）增加 1。指定位置从堆栈指针减去 C 中偏移量得到。



SINS(641) 可用来将数据插入到传送带上已存在器件中间。插入点的位置是从最新添加到传送带上的器件向后器件数 （偏移量）。

标志

名称	标记	操作
错误标志	ER	如果由堆栈指针 （TB+3 与 TB+2）指定位置大于堆栈数据区最后字 PLC 内存地址时置 ON。 （这是堆栈上溢错误） 如果指定偏移值大于最大数据范围大小 -1 （FFFA 十六进制）时置 ON。 其它情况置 OFF。

注意

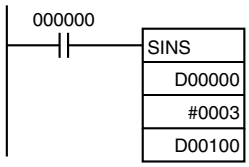
堆栈必须预先由 SSET(630) 定义。

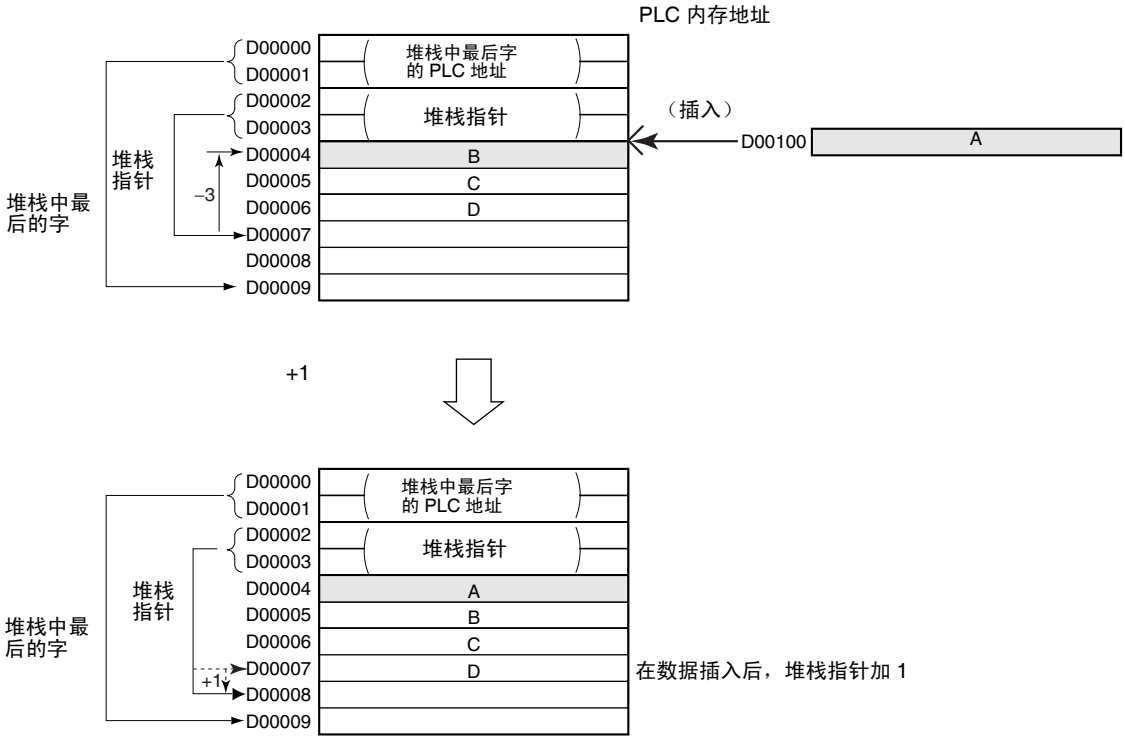
INS(641) 插入一个数据字到堆栈，因此在堆栈尾部必须至少要有一个可插入字的空间。如果堆栈满，将产生错误，并且不能插入数据。

如果由堆栈指针 （TB+3 和 TB+2）指定的堆栈地址大于堆栈 （TB+1 与 TB）最后字的地址，当 SINS(641) 执行时，将产生堆栈上溢错误，源数据不能插入。

例

当下例 CIO00000 为 ON 时 ,SINS(641) 将 D00100 中的数据写入到 D00000 开始的堆栈指定地址处。此时，堆栈指针指向 D00007，偏移值是 3，因此从 D00004 插入数据。原有的数据往下移动一个字，且覆盖 D00007 的数据，同时堆栈指针将从 D00007 增加到 D00008。





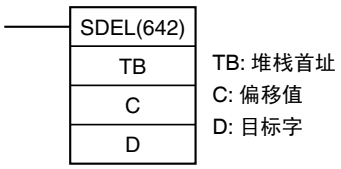
3-17-18 删除堆栈数据：SDEL(642)

用途

在堆栈指定位置处将数据元数删除，将该数据输出到指定位置处，堆栈中余下的数据往下移。偏移值指定了所需数据元素处的位置（在当前指针位置前多少数据元素）。

该指令仅由 CS1-H,CJ1-H,CJ1M 和 CS1D CPU 支持。

梯形图符号



变化

变化	ON 条件时每次周期执行	SDEL(642)
	上升沿微分执行一次	@SDEL(642)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

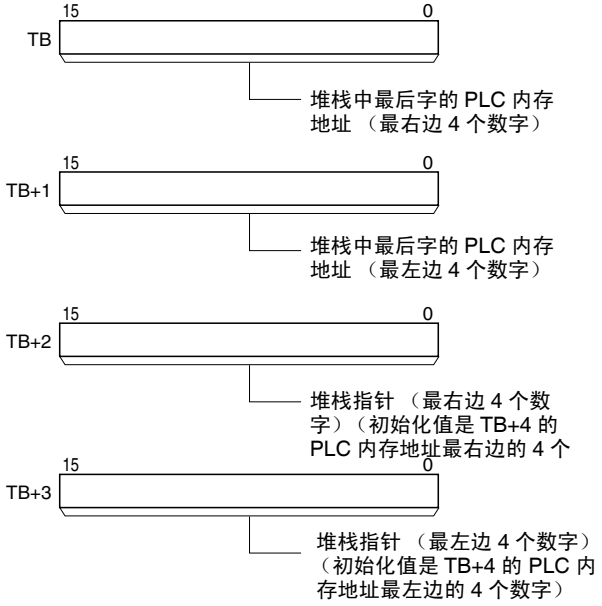
适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

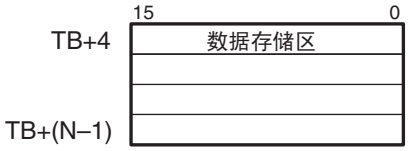
TB ～ TB+3：堆栈控制字

堆栈开始的 4 个字包含了堆栈内最后字的 PLC 内存地址和堆栈指针 （堆栈中下一个有效字的 PLC 内存地址）。



TB+4 ～ TB+(N–1): 数据存储范围

堆栈的余下部分用来存储数据。



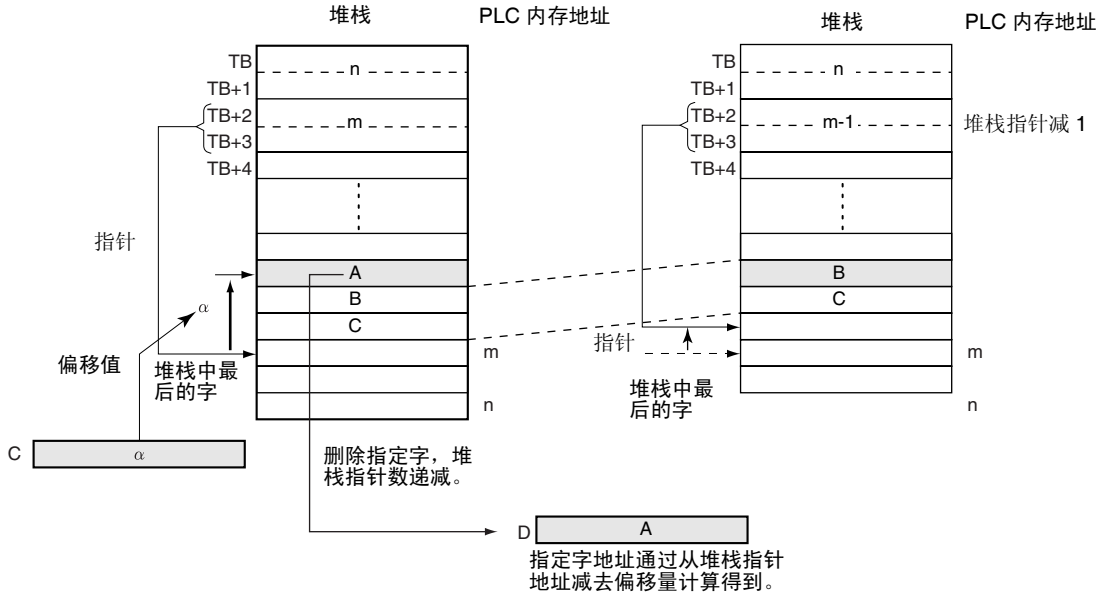
操作数规定

区域	TB	C	D
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A448 ～ A959	A000 ～ A959	A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		

区域	TB	C	D
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---	#0001 ～ #FFFB（十六进制）	---
数据寄存器	---	DR0 ～ DR15	
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15		

说明

SDEL(642) 在堆栈指定地址删除数据，输出该数据到指定目标字，将堆栈中其它数据往上移动。同时，SDEL(642) 使堆栈指针（TB+3 和 TB+2）减 1。指定位置从堆栈指针减去 C 中偏移量得到。



SDEL(642) 可用来删除传送带上不需要的数据。删除位置是从最新添加到传送带上的器件向后器件数（偏移量）。

标志

名称	标记	操作
错误标志	ER	如果堆栈指针的内容 （TB+3 与 TB+2）小于或等于堆栈数据范围中第一个字的 PLC 内存地址 （TB+4）时置 ON。 （这是堆栈下溢错误） 如果 C 中指定偏移值等于 0，或大于最大数据范围（FFFB 十六进制）时置 ON。 其它情况置 OFF。
等于标志	=	输出到 D 中数据为 0000 时置 ON。 其它情况置 OFF。

注意

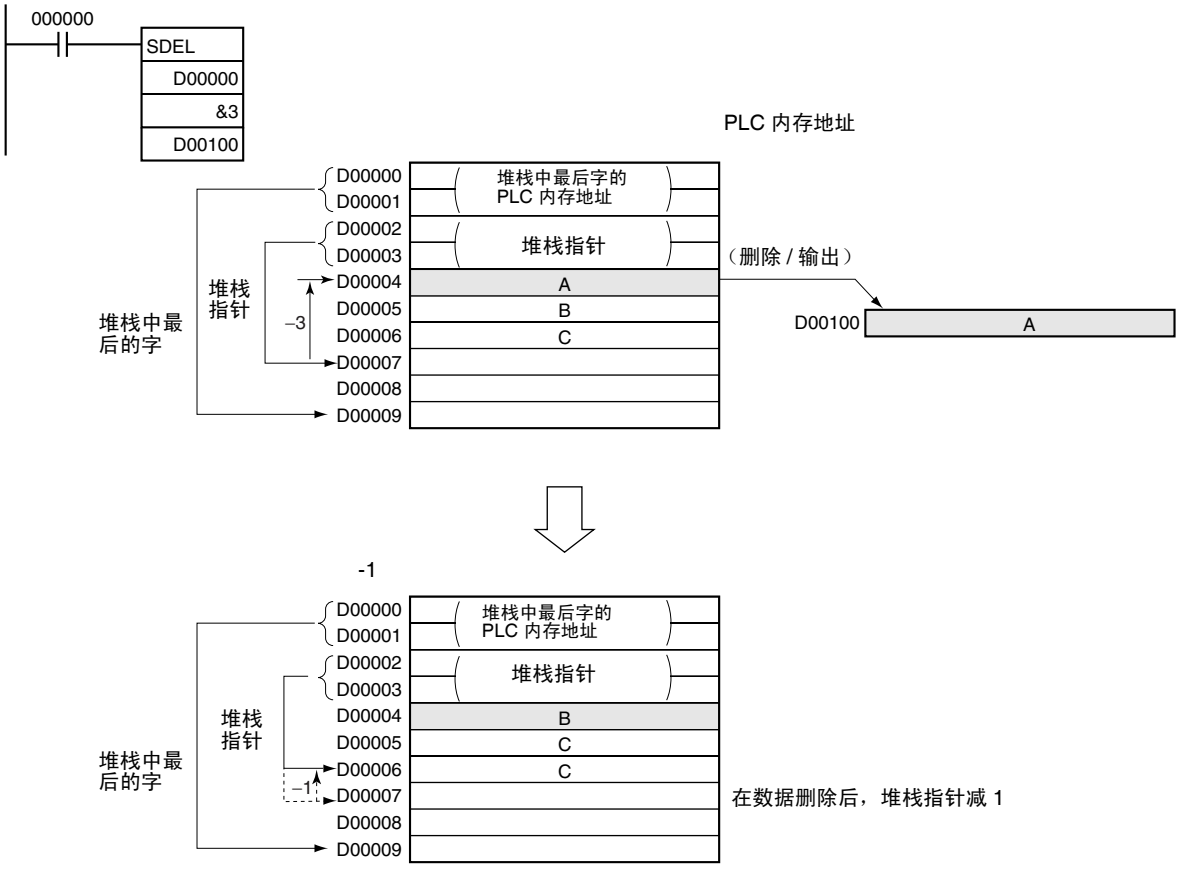
堆栈必须预先由 SSET(630) 定义。

堆栈指针中的地址必须大于开始数据区的 PLC 内存地址 (TB+4)。如果堆栈指针小于 (TB+4) 的 PLC 内存地址将出现一个错误，即产生堆栈下溢错误。

例

当下例 CIO00000 为 ON 时，SDEL (642) 将删除从 D00000 开始的堆栈中的指定字，并将删除的数据输出到 D00100，余下的数据往上移，堆栈指针递减。

此时，堆栈指针指向 D00007，偏移值是 3，因此从 D00004 删除数据。保留的数据往上移动一个字，同时堆栈指针将从 D00007 减到 D00006。



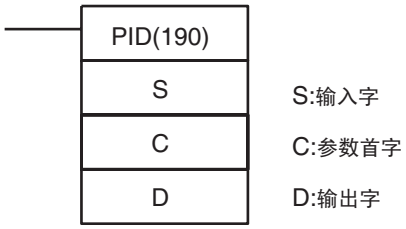
3-18 数据控制指令

3-18-1 PID 控制：PID(190)

用途

根据设定的参数执行 PID 控制。

梯形图符号



变化

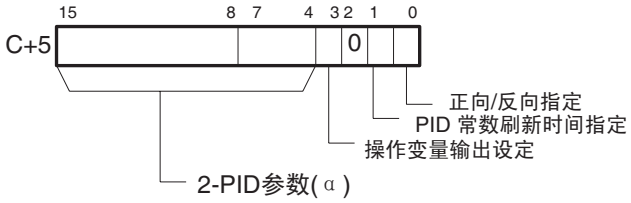
变化	ON 条件时每次周期执行	PID(190)
	上升沿微分执行一次	不支持
	下降沿微分执行一次	不支持
立即刷新定义		不支持

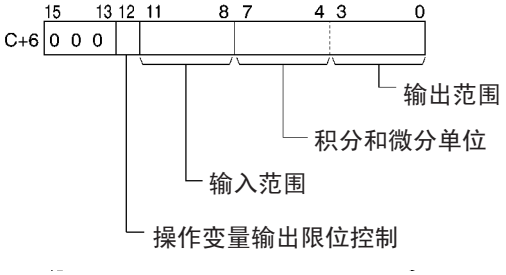
适用程序区

块程序区	步程序区	子程序	中断任务
不允许	OK	OK	不允许

参数

下图展示了参数数据的位置。若需这些参数的详细资料，可参见本章中 *PID 参数的设置部分*。





操作数规定

区域	S	C	D
CIO 区	CIO 0000～CIO 6143	CIO 0000～CIO 6105	CIO 0000～CIO 6143
工作区	W000～W511	W000～W473	W000～W511
保持位区	H000～H511	H000～H473	H000～H511
辅助位区	A000～A959	A000～A921	A448～A959
定时器区	T0000～T4095	T0000～T4057	T0000～T4095
计数器区	C0000～C4095	C0000～C4057	C0000～C4095
DM 区	D00000～D32767	D00000～D32729	D00000～D32767
无区号 EM 区	E00000～E32767	E00000～E32729	E00000～E32767
有区号 EM 区	En_00000～En_32767 (n = 0～C)	En_00000～En_32729 (n = 0～C)	En_00000～En_32767 (n = 0～C)
二进制间接 DM/EM 地址	@ D00000～@ D32767 @ E00000～@ E32767 @ En_00000～@ En_32767 (n = 0～C)		
BCD 间接 DM/EM 地址	*D00000～*D32767 *E00000～*E32767 *En_00000～*En_32767 (n = 0～C)		
常数	DR0～DR15	---	DR0～DR15
数据寄存器	---		
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0～,IR15 –2048～+2047 ,IR0～–2048～+2047 ,IR15 DR0～DR15, IR0～IR15		

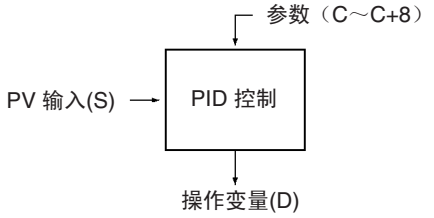
说明

当执行条件为 ON 时， PID(190) 按照 C 中设置的参数（设定值， PID 常量等等）在两个自由度上对目标值执行 PID 控制，从输入字 S 的内容中得到指定输入范围的二进制数据，并根据设定参数执行 PID 动作。计算结果以操作变量的形式存入输出字 D。

在执行条件由 OFF 变为 ON 时读取设定参数，如果设置超出了许可范围，错误标志置 ON。

如果设置值在许可范围内， PID 处理按初始值执行，此时未运行缓冲操作，该操作在执行 PID 过程后作用于操作变量。（缓冲操作过程指的是为了避免突然变化造成相反效果，而渐近地、持续地改变操作变量的过程）。

当执行条件变 ON 时，计入设定采样周期的 PV 值并执行过程。



C+6 的第 8 ～ 11 位中设置的输入范围指定了 16 位以内的 PV 输入 (S) 的有效输入数据位数。例如，如果指定输入范围为 12 位（16 进制 4），那么从 0000H ～ OFFFH 的范围对 PV 是有效的。（大于 0FFFH 的值作为 OFFFH 处理），设定值的范围也依赖于输入范围。

测量值(PV)和设定值(SV)为从0000H到输入范围的最大值的二进制无符号数。

C+6 的第 0 ～ 3 位中设置的输出范围指定了 16 位以内的操作变量有效输出数据位数。例如，如果指定输出范围为 12 位 （16 进制 4），那么从 0000H ～ OFFFH 的范围将被输出为操作变量。

对只是比例操作 , 当 PV 等于 SV 时，变量可被指定为：

0：输出 0%

1：输出 50%

比例操作方向可被设定为正向或反向。

可以指定操作变量的上限和下限。

采样周期可以设定为以 10ms(0.01 ～ 99.99s) 为单位的值 , 但实际的 PID 动作由采样周期和 PID(190) 指令执行时间 （每个周期）共同决定。

使 PID 常量变化的时刻可以设置成下列之一：

1） PID 指令执行开始。

2） PID 指令执行和每个采样周期开始。

在每个采样周期 （也就是 PID 指令执行期间）仅比列带 (P)，积分常量 (TiK)，微分常量 (TdK) 可以变化。时刻由 C+5 的位 1 设置。

注 设置 C+5 的位 1 仅由 CJ1,CS1-H,CJ1-H CPU 和 CS1 CPU（有 001201 □□□□ 或更大的产品批号）（生产日期在 2001 年 12 月 1 日或以后）支持。

在 PID 参数 (C ～ C+38) 中，当执行条件为 ON 时 , 只有设定值 (SV) 可变。改变其它值后，一定要将执行条件由 OFF 置为 ON。

标志

名称	标记	操作
错误标志	ER	C 数据超出范围时置 ON 实际采样周期大于设定采样周期的两倍时置 ON 其它情况置 OFF
大于标志	>	执行 PID 动作后操作变量超出上限时置 ON 其它情况置 OFF
小于标志	<	执行 PID 动作后操作变量低于下限时置 ON 其它情况置 OFF
进位标志	CY	PID 控制正在执行时置 ON 其它情况置 OFF

注意

执行条件象一个停止 - 运行信号一样控制着 PID(190) 的执行。在 C+9 ～ C+38 被初始化后 , 若下一个周期执行条件仍为 ON，则执行 PID 计算。因此 , 当使用常 ON 标志 (ON) 作为 PID(l90) 的执行条件时 , 在操作开始时要有一个单独的 C+9 和 C+38 初始化过程。

如果 C 数据超出范围 , 将产生一个错误 , 错误标志置 ON。

如果实际采样周期大于指定采样周期的两倍,将产生一个错误,错误标志置ON，而 PID 控制继续执行。

PID 控制被执行时进位标志置 ON。

执行 PID 动作后，操作变量超过上限，大于标志置 ON，此时计算结果以上限值输出。

执行 PID 动作后，操作变量低于下限，小于标志置 ON，此时计算结果按下限值输出。

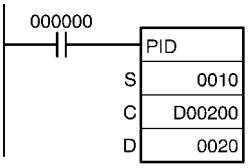
在 PID 参数 (C ～ C+38) 中 , 输入条件为 ON 时只有 C 中的 SV 值可变。如果改变了其它值 , 一定要将输入条件由 OFF 置为 ON, 以使新值有效。

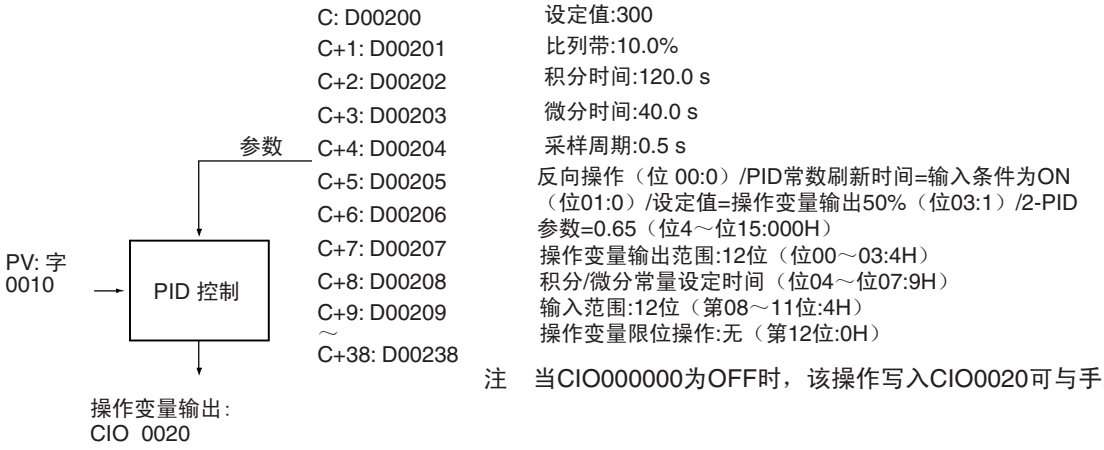
例

在 CIO000000 的上升沿 （从 OFF ～ ON），工作区 D00209 ～ D00238 按 D00200 ～ D00208 中的参数（如下所示）进行了初始化。工作区被初始化后，将执行 PID 控制，并将操作变量输出到 CIO00200。

当 CIO000000 为 ON 时，PID 控制依据 D00200 ～ D00208 中的设置，以采样周期间隔执行，操作变量输出到 CIO0020。

在 CIP000000 置为 ON 后，如果改变比列带 (P)、积分常量 (TiK)、微分常量，用在 PID 计算中的 PID 常量将不会改变。





注 当CIO000000为OFF时，该操作写入CIO0020可与手工操作相同。

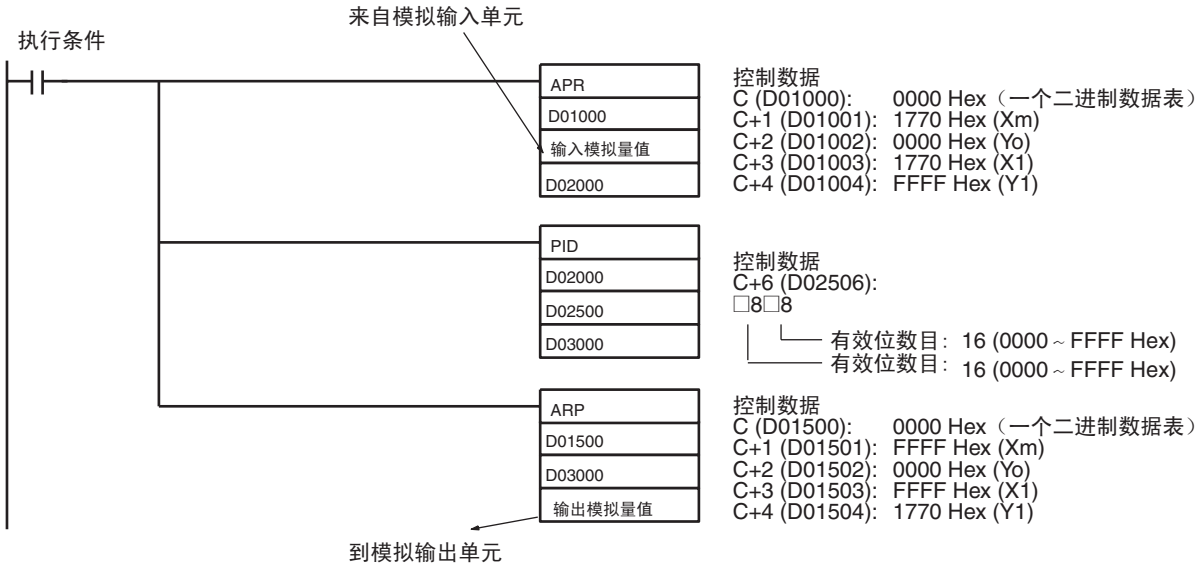
输入值和操作变量范围

用于测量值有效输入数据位数由输入范围设定 （C+8，位 08 ～ 11）指定，操作变量有效输出位数由输出范围设定 （C=6，位 0 ～ 3）指定。下表显示了该范围。

C+6, 位 08 ～ 11 或 C+6, 位 00 ～ 03	有效位数	范围
0	8	0000 ～ 00FF Hex
1	9	0000 ～ 01FF Hex
2	10	0000 ～ 03FF Hex
3	11	0000 ～ 07FF Hex
4	12	0000 ～ 0FFF Hex
5	13	0000 ～ 1FFF Hex
6	14	0000 ～ 3FFF Hex
7	15	0000 ～ 7FFF Hex
8	16	0000 ～ FFFF Hex

如果由模拟输入单元和模拟输出单元处理的数据范围不能由有效位数准确设定，在 PID(190) 前或后，可用 APR(069) （算术处理）以转换合适范围。

下面程序部分显示了一个作为 COMPOBUS/D 从操作的 DRT1-AD04 模拟输入元和 DRT1-DA02 模拟输出单元例子。两个单元的数据范围是 0000 ～ 1770H，它不能仅由设置有效数位来设置。因而， APR(069) 用来把 0000 ～ 1770H 模拟输入数据范围转变为输入到 PID(190) 的 0000 ～ FFFFH，接着 PID(190) 操作变量输出再用 APR(069) 转换回 0000 ～ 1770H，从模拟量输出单元输出。



执行规定

项目			规定
PID 控制方法		---	目标值过滤类型两个自由度 PID 方法 （正向 / 反向）
PID 控制循环		---	无限制 （每个指令 1 个循环）
采样周期		t	0.01 ～ 99.99 s
PID 常量	比列带	P	0.1 ～ 999.9%
	积分常数	Tik	1 ～ 8191, 9999 （对采样周期倍数 9999 时，无积分动作）
	微分常数	Tdk	0 ～ 8191 （对采样周期倍数 0 时，无微分动作）
设定值		SV	0 ～ 65535 （有效至输入范围最大值）
测量值		PV	0 ～ 65535 （有效至输入范围最大值）
操作变量		MV	0 ～ 65535 （有效至输入范围最大值）

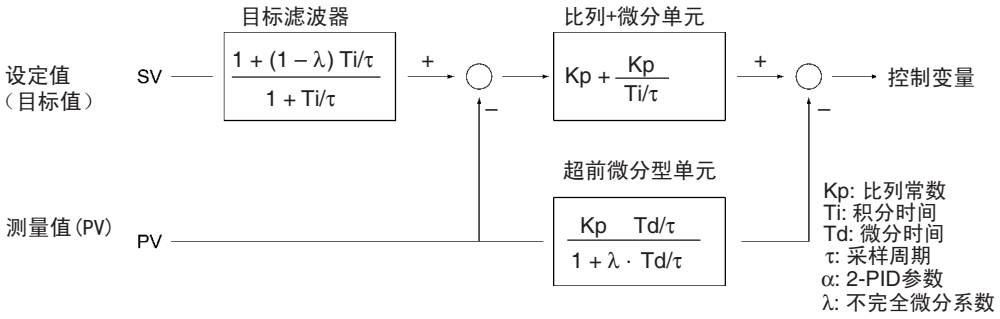
计算方法

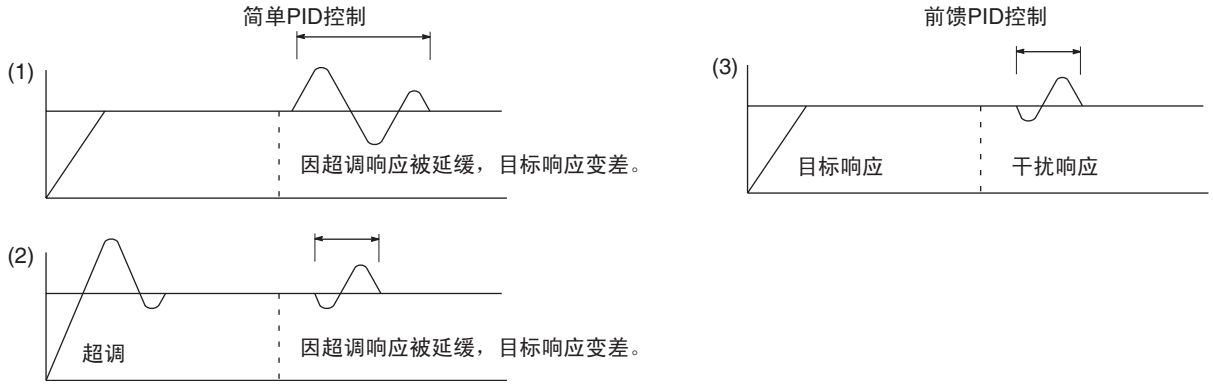
PID 控制计算由在两个自由度上的目标值过滤控制进行。

两个自由度的目标值 PID 方框图

简单 PID 控制存在超调时，干扰的稳定作用被延缓 (1)。另一方面，如果干扰的稳定作用被加速，将产生超调并延缓目标值响应 (2)。

当使用两个自由度的目标值 PID 控制时，不存在超调，并可同时加速目标值和干扰稳定作用的响应 (3)。





PID 参数设置

控制数据	项目	内容	设定范围	置 ON 输入条件变化
C	设定值 (SV)	控制过程的目标值	二进制数据 （与指定输入范围位数相同）	不允许
C+1	比例带	表示比例控制范围 / 总控制范围的 P 作用参数。	0001 ～ 270FH(1 ～ 9999)；（0.1% ～ 999.9%，以 0.1% 为单位）	如果 C+5 的位 1 是 1，输入条件 ON 可以改变。
C+2	Tik	表示积分作用强度的一个常量，数值增加，积分强度减小。	0001 ～ 1FFFH(1 ～ 8191)（9999= 不执行积分操作）（参见注 1）	
C+3	Tdk	表示微分作用强度的一个常量，数值增加 , 微分强度减小。	0001 ～ 1FFFH(1 ～ 8191)；（ 0000= 不执行微分操作）（参见注 1）	
C+4	采样周期	设定执行 PID 作用的周期。	0001 ～ 270FH(1 ～ 9999)；（0.01 ～ 99.99s，以 10ms 为单位）	不允许
C+5 中位 04 ～ 15	2-PID 参数 (α)	输入滤波系数 , 通常为 0.65 （即 000 设定），过滤效率随系数向 0 靠近而递减。	000H: α =0.65 设置从 100 ～ 163H 表示最右边的两位的值被设定为从 α =0.00 ～ α =0.99 （参见注 2）	
C+5 中位 03	操作变量输出设定	设定 PV 等于 S Ⅴ时的输出变量控制	0: 输出 0% 1: 输出 50%	
C+5 中位 01	PID 常量变化设定	在 PID 计算中，可以改变比例带 (P)，积分常数 (Tik) 与微分常数 (Tdk) 的时刻设定	0:PID 指令开始执行 1:PID 指令开始执行和每一个采样周期	允许

控制数据	项目	内容	设定范围	置 ON 输入条件变化
C+5 （位 00）	PID 前向 / 逆向设定	决定比例作用的方向	0: 反向 1: 正向	不允许
C+6 （位 12）	操作变量输出限位控制	决定输出控制变量是否应用限位操作	0: 无效 （无限位控制） 1: 有效 （有限位控制）	
C+6 （位 08 ～ 11）	输入范围	输入数据位数	0:8 位　5:13 位 1:9 位　6:14 位 2:10 位　7:15 位 3:11 位　8:16 位 4:12 位	
C+6 （位 04 ～ 07）	积分和微分单位	决定积分和微分常量的单位	1: 采样周期倍数 9: 时间 （单位：100ms）	
C+6 （位 00 ～ 03）	输出范围	输出数据位数 （输出位数自动等于输入位数）	0:8 位　5:13 位 1:9 位　6:14 位 2:10 位　7:15 位 3:11 位　8:16 位 4:12 位	
C+7	输出变量下限	操作变量输出限位有效时的下限	0000 ～ FFFF （二进制） （参见注 3）	
C+8	输出变量上限	操作变量输出限位有效时的上限	0000 ～ FFFF （二进制） （参见注 3）	

注 1. 单位设为 1 时，范围为从 1 ～ 8,191 倍周期。单位设定为 9 时，范围为从 0.1 ～ 819.1s。设为 9 时，将积分和微分时间设在 1 ～ 8,191 倍采样周期的范围内。

2. 设定 2-PID 参数 (α) 为 000(0.65)，即标准值。

3. 当输出控制变量限位控制有效 （即设为 1）时 , 设定如下值：
输出范围最大值≥ MV 输出上限≥ MV 输出下限≥ 0000

采样周期和循环时间

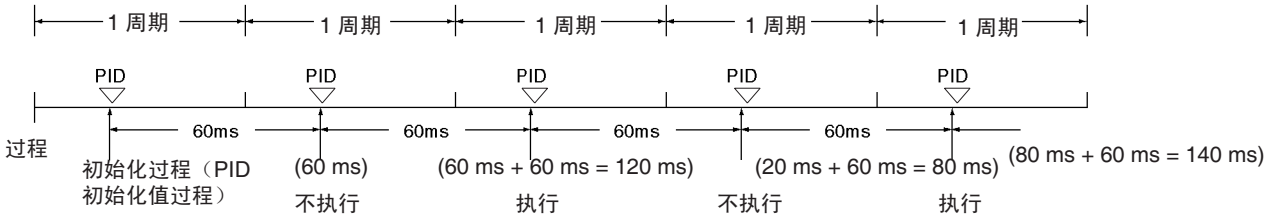
采样周期可以以 10ms 为单位设定 (0.01 ～ 99.99s), 但实际 PID 作用是由采样周期和 PID 指令执行时间对每个周期共同决定的。采样周期和循环时间的关系如下所述。

- 如果采样周期小于循环时间， PID 控制按每个循环而不是按每个采样周期执行。
- 如果采样周期大于或等于循环时间，PID 控制并不是每次循环执行 , 而是当循环时间累积值 （PID 指令间的时间）大于或等于采样周期时执行 PID(190)。累积值乘余部分 （即循环时间累积值减采样周期）转入下一次累积值。

  例如 , 设采样周期为 100ms, 而循环时间为 60ms, 在初始化后的第一个周期 , 因为60ms小于100ms,故不执行PID(190)。在第二个循环,60ms+60ms大于 100ms,PID(190) 被执行。余值 20ms （即 120ms-100ms=20ms）转入下一循环累积值。

  在第三个循环 , 余值 20ms 与 60ms 相加。因为和 80ms 小于 100ms, 故不执行 PID(190)。

  在第四个循环 ,80ms 与 60ms 相加。因为 140ms 大于 100ms,PID(190) 被执行 , 剩余的 40ms （即 120ms-100ms=20ms）转入下一周期，在下面的周期中重复执行该过程。



读入测量时间　小于100ms，故不执行PID　大于100msPID被执行并将20ms转入下一循环　小于100ms，故不执行PID　大于100ms PID被执行并将40ms转入下一循环

控制作用

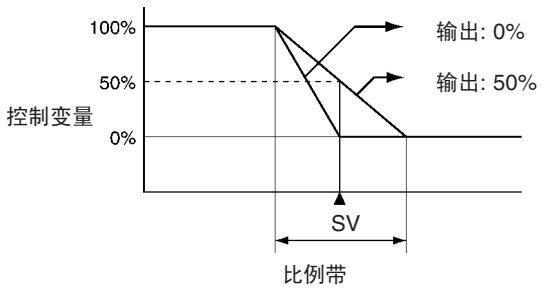
比例作用 (P)

比例作用指建立在设定值 (SV) 上的比例带操作 , 在此带内控制变量 (MV) 与偏差成比例，下面的例子为反向操作实例。

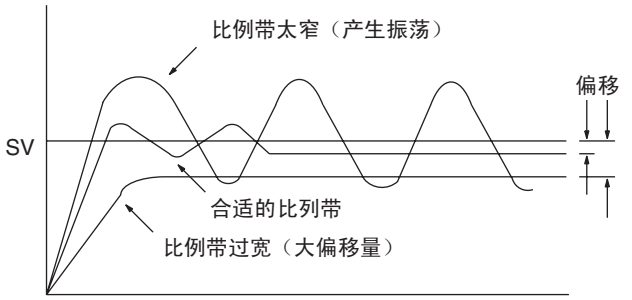
如果使用了比例作用，且当前值 (PV) 变得小于比例带 , 控制变量为 100%，（即最大值）。在比例带内， MV 与偏差 （SV 与 PV 之差）成比例并逐渐减小，直至 SV 与 PV 匹配 （即直到偏差为 0），此时 MV 为最小值 0% （或 50%, 根据控制变量输出定义参数的设定而定）。当 PV 大于 SV 时 MV 也为 0%。

比例带用整个输入范围的百分值来表示。比例带越小，比例常数越大 , 校正作用也越强。在比例作用下产生一个偏移量 （剩余偏差），减小比例带可减少偏移量，而若比例带太小则会产生振荡。

比例作用（反向作用）

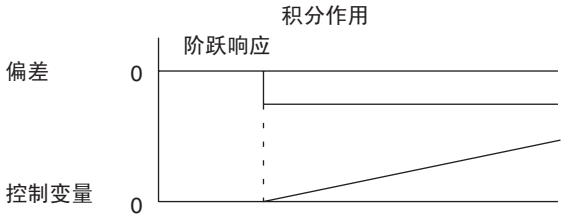


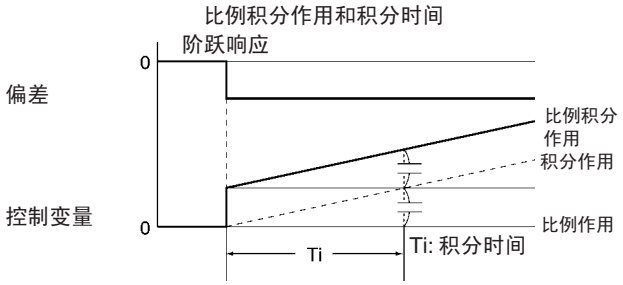
调整比例带



积分作用 (I)

结合比例和积分作用，随着过去的时间会减少偏移量，使 PV 和 SV 相匹配，积分作用的强度由积分时间指定。如下例所示，积分时间指对阶跃偏差，使积分作用的控制变量与比例作用的控制变量达到同一水平所需的时间。





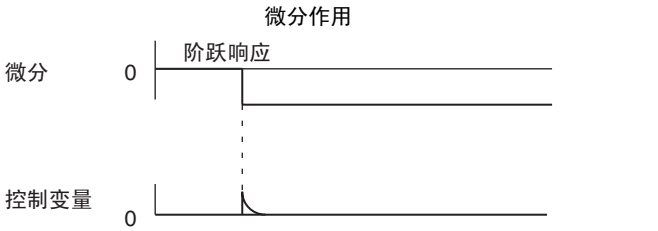
微分作用 (D)

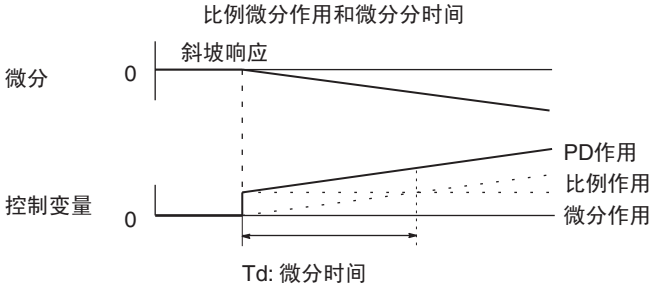
比例作用和积分作用都用通过控制结果进行校正，因此不可避免地会产生响应滞后。微分作用弥补了这一缺陷。为了响应突然干扰，微分作用产生一个大的控制变量，并迅速恢复在原始状态。校正的执行是通过操作变量与偏差导致的斜坡 （微分系数）成比例进行的。

微分作用的强度由微分时间指定。如下例所示，微分时间是指对阶跃偏差。

使微分作用的控制变量与比例作用的控制变量达到同一水平所需的时间。

微分时间越长，微分作用的校正作用越强。

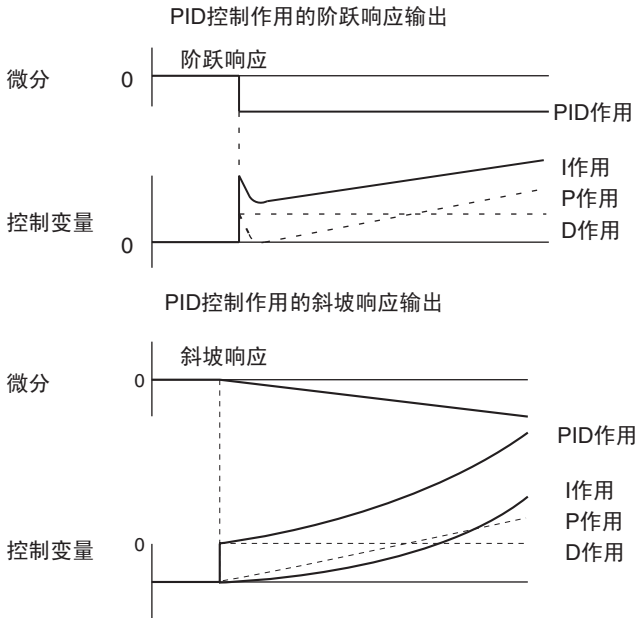




PID 作用

PID 作用由比例作用 (P), 积分作用 (I)，和微分作用 (D) 共同组成。 PID 作用对有死区的控制对象也能进行优化控制。

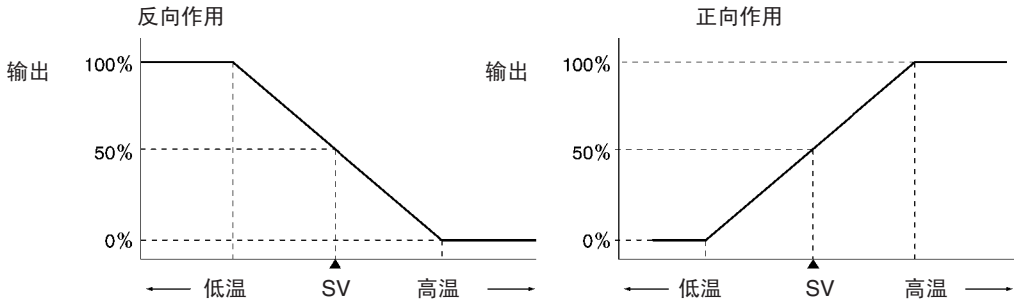
PID 作用中的比例作用是提供无振荡的平滑控制，积分作用是自动校正任何偏移，微分作用加速干扰响应。



作用方向

使用 PID 控制时 , 要在下面两种控制方向中选择一个。对每一个方向 ,MV 都随着 SV 与 PV 之差的增长而增长。

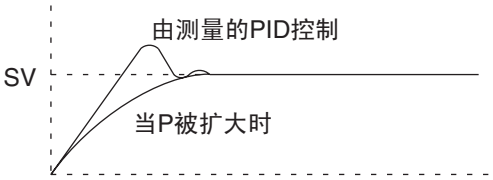
- 正向作用：当 PV 大于 SV 时增大 MV。
- 反向作用：当 PV 小于 SV 时增大 MV。



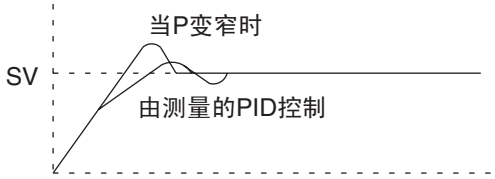
调整 PID 参数

PID 参数与控制状态之间的大致关系如下所示。

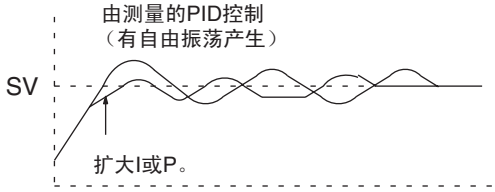
- 当达到稳定作用所需要的时间量（安定时间）没有问题 , 而稳定不引起超调很重要时 , 应扩大比例带。



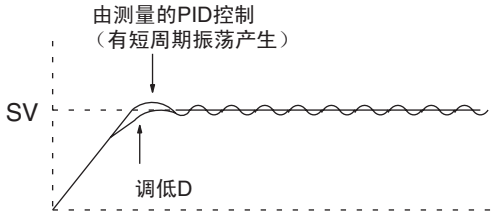
- 如果没有超调问题，但需要快速稳定控制，则调窄比例带，而如果比例带过窄，则会产生振荡。



- 如果有宽幅振荡或者操作受到超调和欠调的困扰，可能是因为积分作用太强了。如果增加积分时间或扩大比例带，则可减少振荡。



- 如果产生了短周期振荡，可能是由控制系统响应快和微分作用过强造成的，此时调低微分作用。

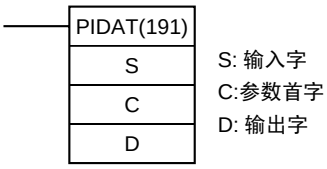


3-18-2 PID 自动整定：PIDAT(191)

用途

根据设定的参数执行 PID 控制。 PID 常数可以自动调节。
该指令仅由 CS1-H， CJ1-H， CJ1M 和 CS1D CPU 支持。

梯形图符号



变化

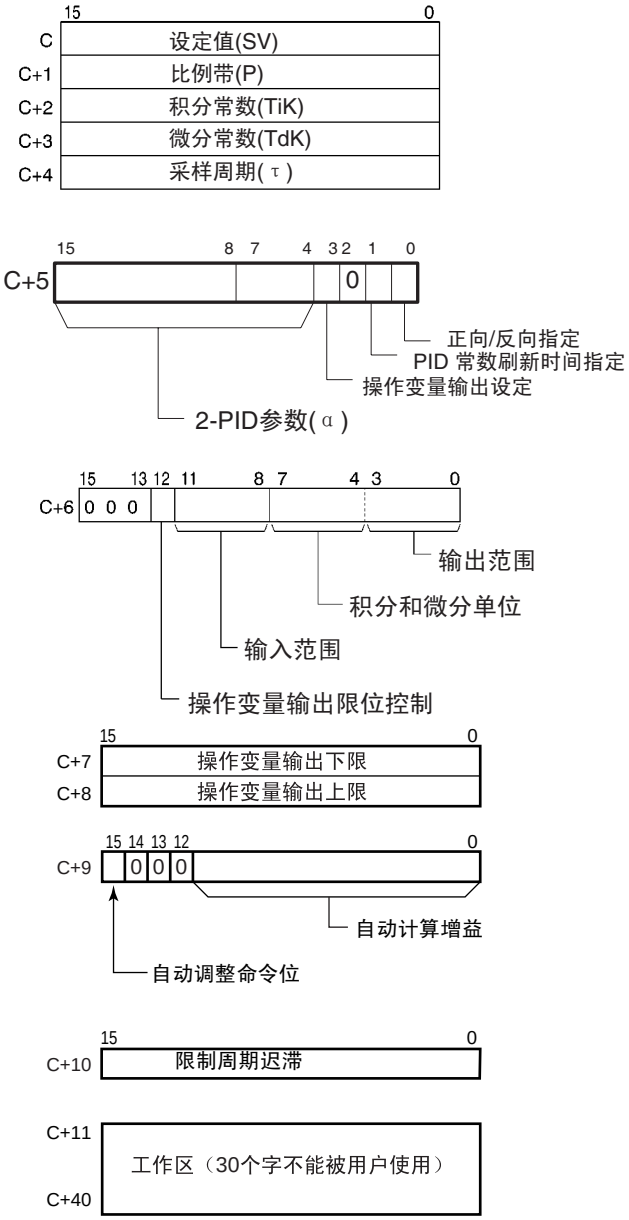
变化	ON 条件时每次周期执行	PIDAT(191)
	上升沿微分执行一次	不支持
	下降沿微分执行一次	不支持
立即刷新定义		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
不允许	OK	OK	不允许

参数

下图显示参数数据的位置。有关参数的详细资料，可参见*本章中 PID 参数的设置部分。*



操作数规定

区域	S	C	D
CIO 区	CIO 0000～CIO 6143	CIO 0000～CIO 6105	CIO 0000～CIO 6143
工作区	W000～W511	W000～W473	W000～W511
保持位区	H000～H511	H000～H473	H000～H511
辅助位区	A000～A959	A000～A921	A448～A959
定时器区	T0000～T4095	T0000～T4057	T0000～T4095
计数器区	C0000～C4095	C0000～C4057	C0000～C4095
DM 区	D00000～D32767	D00000～D32729	D00000～D32767
无区号 EM 区	E00000～E32767	E00000～E32729	E00000～E32767

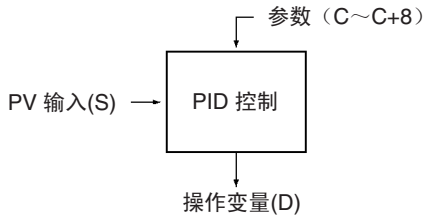
区域	S	C	D
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	En_00000 ～ En_32729 (n = 0 ～ C)	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	DR0 ～ DR15	---	DR0 ～ DR15
数据寄存器	---		
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ 2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15		

说明

当执行条件为 ON 时， PIDAT(191) 按照 C 中设置的参数 （设定值， PID 常量等等）在两个自由度上对目标值执行 PID 控制，从输入字 S 的内容中得到指定输入范围的二进制数据，并根据设定参数执行 PID 动作。计算结果以控制变量的形式存入输出字 D。

在执行条件由 OFF 交为 ON 时读取设定参数，如果设置超出了许可范围，错误标志置 ON。如果设置值在许可范围内，PID 处理按初始值执行 , 此时未运行缓冲操作，该操作在执行 PID 过程后作用于控制变量（缓冲操作过程指为了避免突然变化造成相反效果，而渐近地、持续地改变操作变量的过程）。

当执行条件变 ON 时，计入设定采样周期的 PV 值并完成处理。



自动调整功能

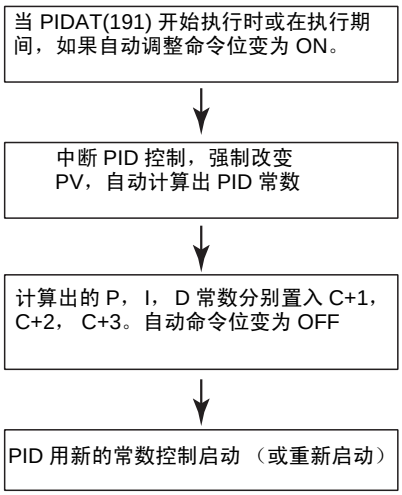
在每一个周期都会检查自动命令位 （C+9 的位 15）的状态。在给定周期，如果控制位置 ON， PIDAT(191) 将开始自动调整 PID 常量（当自动调整时， SV 的变化将不会被自动反映出来）。

自动调整用到限制周期方法。 PIDAT(191) 强制使控制变量发生变化

（最大值 控制变量←→最小值 控制变量），监视控制系统的特性。从观察到的特性中计算出 PID 常数，新的 P， I 和 D 常数自动存储到 C+1， C+2 和 C+3。此刻，自动命令位 （C+9 的位 15）置为 OFF，在 C+1， C+2， C+3 中，用新的常数恢复 PID 控制。

- 当PIDAT(191)开始执行时，如果自动调整命令位置ON，自动调整将首先被执行，然后 PID 控制将开始 PID 常数计算。
- 在 PIDAT(191) 执行期间，如果自动调整命令位置 ON，PIDAT(191) 中断用户设置常数的 PID 控制，完成自动调整，然后用计算好的 PID 常数恢复 PID 控制。

下列流程表显示了自动调整过程：



注 1. 在自动调整期间，如果通过将 AT 命令位置 OFF 来中断自动调整， PID 控制将启动在自动调整之前的 PID 常数。

2. 如果发生自动执行错误，PID 控制将启动在开始调整前使用过的 PID 常数。

在上述注 1 和注 2 中，当中断自动调整时， PID 常数已经计算出来，它们将自动有效。

PID 控制

C+6 的第 8 ～ I1 位中设置的输入范围，指定了 16 位以内的 P Ⅴ输入 (S) 的有效输入数据位数。例如，如果指定输入范围为 12 位 （16 进制 4），那么从 0000H 到 OFFFH 的范围对 PV 是有效的。（大于 0FFFH 的值作为 OFFFH 处理）

设定值的范围也依赖于输入范围。

测量值 (PV) 和设定值 (SV) 是从 0000H 到输入范围的最大值的二进制无符号数。

C+6 的第 0 ～ 3 位中设置的输出范围，指定了 16 位以内的操作变量有效输出数据位数。例如，如果指定输出范围为 12 位 （4 十六进制），那么从 0000H ～ 0FFFH 的范围将被输出为控制变量。

对只是比例操作，当 PV 等于 SV 时，变量可被指定为如下：

0：输出 0%

1：输出 50%

比例操作方向可设定为正向或反向。

控制变量输出可以指定为上限制和下制限。

采样周期可以设定为以 10ms(0.01 ～ 99.99s) 为单位的值，但实际的 PID 运行由采样周期和 PID(190) 指令执行时间 （每个周期）共同决定。

使 PID 常量变化的时刻可以设置成下列之一：

1）PID 指令执行开始。

2） PID 指令执行和每个采样周期开始。

在每个采样周期 （即 PID 指令执行期间）仅比例带 (P)、积分常量 (Tik)、微分常量 (Tdk) 可以变化。时刻由 C+5 的位 1 设置。

当手动改变 PID 常数时，设置 PID 常量使能位（C+5 的位 1）为 1，这样 C+1，C+2， C+3 的值在 PID 计算的每个采样期间可被刷新。这种设置在自动调整后也可以手动调整 PID 常数。

在 PID 参数 （C 到 C+38）中，当执行条件为 ON 时，下列参数可以被改变。当任何其它的值被改变后，确保变化条件从 OFF 到 ON，以便新的设置允许。

- 在C中设置值(SV)仅在PID控制可以被改变。在自动调整期间，SV变化不会被反映出来。
- PID 常数变化允许设置 （C+5 的位 1）
- P，I，D 常数在 C+1，C+2，C+3 中。（仅当 PID 常数变化使能位（C+5 的位 1）设置为 1，变化的常数在采样期间被刷新）。
- 自动调整命令位 （C+9 的位 15）
- 自动调整计算增益（C+9 的位 0）和限制周期迟滞 (C+10)（在自动调整开始时，这些值可被读取）。

注 PIDAT(191) 指令和 PID(190) 指令类同，新增的自动调整 (AT) 功能。因此 PID 控制操作相同。详细的 PID(190) 控制操作和例子可参考 *3-18-1 PID 控制*。

标志

名称	标记	操作
错误标志	ER	C 数据超出范围时置 ON 实际采样周期大于设定采样周期的两倍时置 ON 其它情况置 OFF
大于标志	>	执行 PID 操作后，控制变量超出上限时置 ON 其它情况置 OFF
小于标志	<	执行 PID 操作后，控制变量小于下限时置 ON 其它情况置 OFF
进位标志	CY	PID 控制正在执行时置 ON 其它情况置 OFF

注意

执行条件象一个停止－运行信号一样控制着 PIDAT(191) 的执行。在 C+9 ～ C+38 被初始化后，若下一个周期执行条件仍为 ON 则执行 PID 运算。因此，当使用常 ON 标志 (Al) 作为 PIDAT(l91) 的执行条件时，在操作开始时要有一个单独的 C+9 和 C+38 初始化过程。如果 C 数据超出范围，将产生一个错误，错误标志置 ON。

如果实际采样周期大于指定采样周期的两倍，将产生一个错误，错误标志置 ON，而 PID 控制继续执行。

PID 控制被执行时进位标志置 ON。

执行 PID 运算后，控制变量超过上限，大于标志置 ON，此时什算结果以上限制值输出。

执行 PID 运算后，控制变量小于下限，小于标志置 ON，此时计算结果按下限值输出。

PID 参数设置

控制数据	项目	内容	设定范围	ON 输入条件的变化
C	设定值 (SV)	控制过程的目标值。	二进制数据 （与指定输入范围位数相同）	允许
C+1	比例带	P 作用参数， 表示比例控制范围 / 总控制。	0001 ～ 270FH(1 ～ 9999)； （0.1% ～ 999.9%，以 0.1% 为单位）	如果 C+5 的位 1 是 1，输入条件 ON 可以改变。
C+2	Tik	表示积分作用强度的一个常量，数值增加，积分强度减小。	0001 ～ 1FFFH (1 ～ 8191) （9999= 积分操作不执行）（见注 1）	
C+3	Tdk	表示微分作用强度的一个常量，数值增加，微分强度减小。	0001 ～ 1FFFH(1 ～ 8191)； （0000= 不执行微分操作） （见注）	
C+4	采样周期	设定执行 PID 运行的周期。	0001 ～ 270FH(1 ～ 9999)； （0.01 ～ 99.99s，以 10ms 为单位）	不允许
C+5 中位 04 ～ 15	2-PID 参数 (α)	输入滤波系数，通常为 0.65 （即 000 设定），滤波效率随系数向 0 靠近而递减。	000H: α =0.65 设置从 100 ～ 163H 表示最右边的两位的值被设定为从 α =0.00 到 α =0.99 （见注 2）	
C+5 中位 03	控制变量输出设定	设定 PV 等于 SV 时的控制变量输出。	0 ：输出 0% 1：输出 50%	
C+5 中位 01	PID 常量变化设定	PID 运算的比例带 (P)，积分常数 (Tik) 与微分常数 (Tdk) 允许变化的时刻。	0:PID 指令开始执行 1:PID 指令开始执行和每一个采样周期	允许

控制数据	项目	内容	设定范围	ON 输入条件的变化
C+5 中位 00	PID 前向 / 逆向设定	决定比例作用的方向。	0: 正向 1: 反向	不允许
C+6 中位 12	操作变量输出限制控制	决定操作变量输出是否应用限制操作。	0: 无效 （无限制控制） 1: 有效 （有限制控制）	
C+6 中位 08 ～ 11	输入范围	输入数据位数。	0:8 位 3:11 位 6:14 位 1:9 位 4:12 位 7:15 位 2:10 位 5:13 位 8:16 位	
C+6 中位 04 ～ 07	输出范围	输出数据位数 （输出位数自动等于输入位数）。		
C+6 中位 00 ～ 03	积分和微分单位	决定积分和微分常量的单位。	1: 采样周期倍数 9: 时间 (100ms)	
C+7	输出变量下限	操作变量输出限制有效时的下限值。	0000 ～ FFFF （二进制） （见注 3）	
C+8	输出变量上限	操作变量输出限制有效时的上限值。	0000 ～ FFFF （二进制） （见注 3）	
C+9 位 15	AT 命令位	控制位开始自动调整 • 设置 AT 命令位为 1，执行自动调整。（当 PIDAT(191) 执行时，自动调整启动）。 • 自动调整完成，该位自动置 OFF。 如果 AT 命令位手动置 OFF，自动调整中断。在这种情况下，当自动调整中断，如果参数已经计算出来， PID 常数将允许改变。	作为控制位： 0 → 1 执行自动调整 1 → 0 中断自动调整 （当自动调整完成时， PID(191) 自动将位置 OFF） 作为标志： 0：自动调整没有执行 1：自动调整正在执行 .	允许
C+9 位 00 ～ 11	AT 计算增益	设置该参数是调整 PID 计算结果基值到存储值。通常将该参数设置成默认值 （0000） • 强调稳定性，增加该值 • 强调响应，减少该值	0000H：1.00 （缺省值） 0001 ～ 03E8H （1 ～ 1000）；（0.01 ～ 10.00，以 0.01 为单位）	允许 （自动调谐启动时，可以读取这些参数）
C+10	限制周期迟滞	当限制周期产生时设置迟滞。对反向操作确省设置置 ON， MV 有 SV-20% 迟滞。因为 PV 不稳定，而不能产生正确的限制周期，增加该设置。然而如果限制周期迟滞高于常值， AT 的准确性将下降。	0000H：0.20% （缺省值） 0001 ～ 03E8H （0.01 ～ 10.00%，以 0.01 为单位） FFFH:0.00% 注 百分数与输入范围相对应。	

注 1. 单位设为 1 时，范围为从 1 ～ 8,191 倍周期。单位设定为 9 时，范围为 从 0.1 ～ 8,191s。设为 9 时，将积分和微分时间设在 1 ～ 8,191 倍采样周期的范围内。

2. 设定 2-PID 参数 (α) 为 000 (0.65)，即标准值。

当控制变量输出限位控制有效 （即设为 “1”）时，设定如下值：

输出范围最大值≥ MV 输出上限≥ MV 输出下限≥ 0000

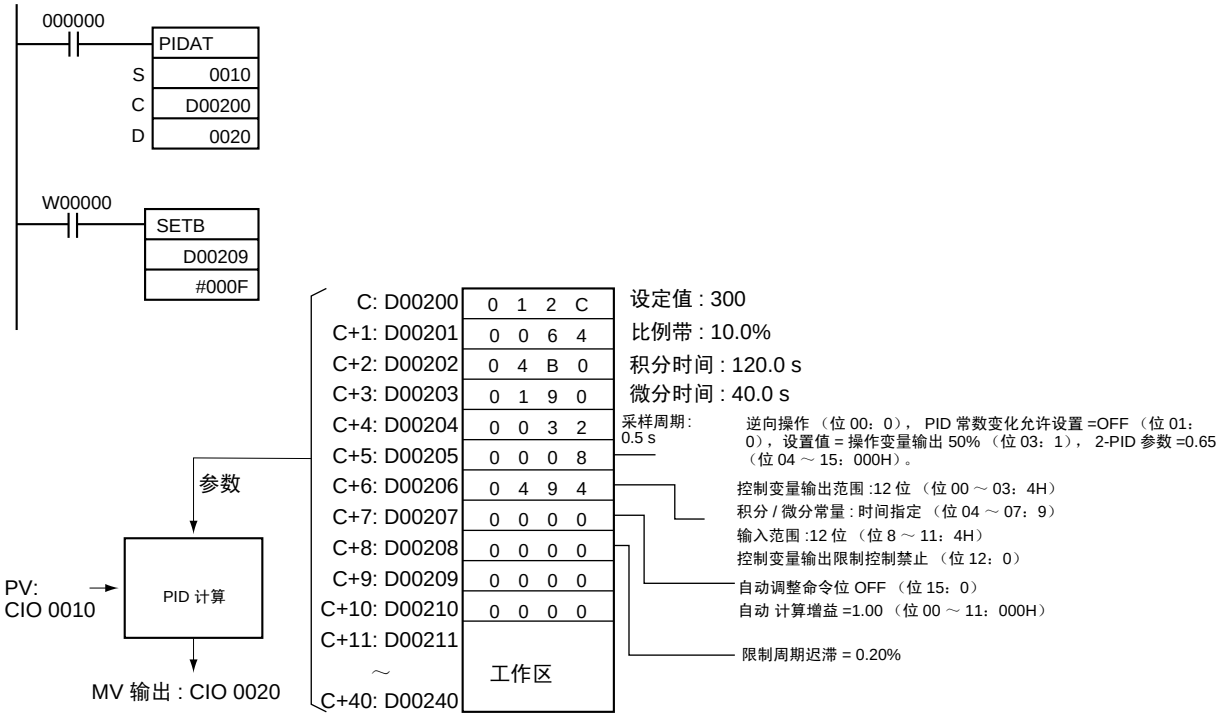
例 1：中断 PID 控制执行自动调整

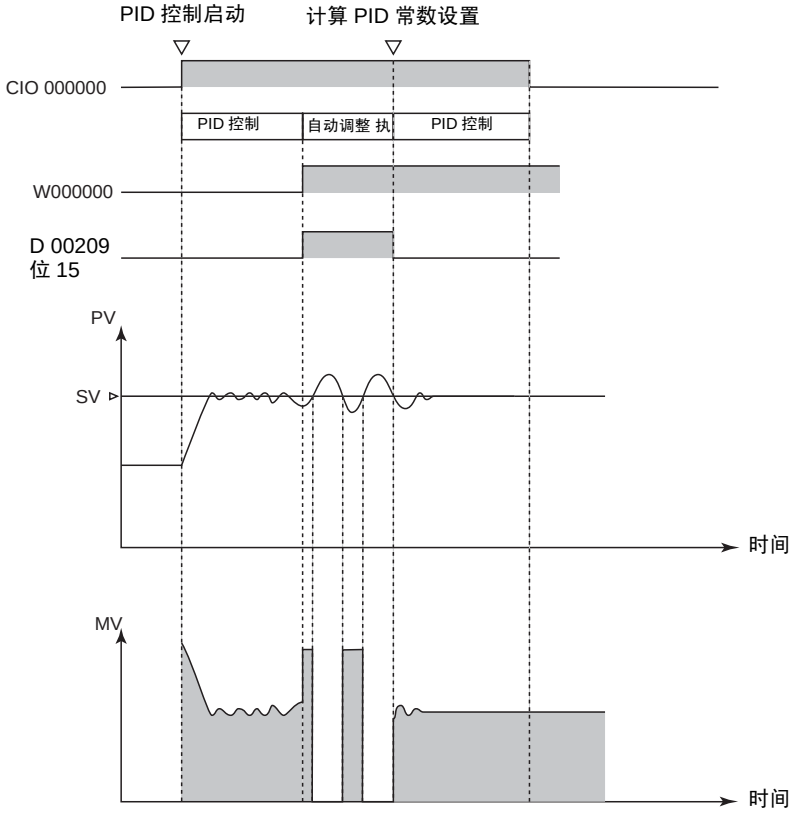
在 CIO 000000(OFF → ON) 上升沿， D00211 ～ D00240 工作区按照设置在 D00200 ～ D00208 的参数（如下所示）初始化，在工作区已经初始化后，PID 控制执行，控制变量输出到 CIO 00200。

当 CIO 000000 置 ON，按照在 D00200 ～ D00210 的参数设置，在采样周期期间 PID 控制执行，将控制变量输出到 CIO00200。

在 CIO00200 置 ON 后，即使改变比例带 (P)，积分常量 (TiK)，微分常量，用于 PID 运算的常数将不会改变。

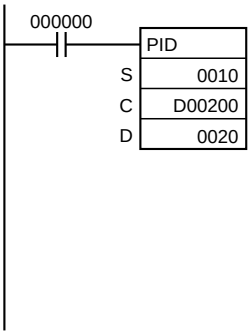
在 W 000000(OFF → ON) 上升沿，SETB(532) 置 D00209(C+9) 位 15 为 ON，开始自动调整。当自动调整完成，计算 P、I、D 常数写到 C+1，C+2，C+3 中。在新常数下， PID 控制重新启动。

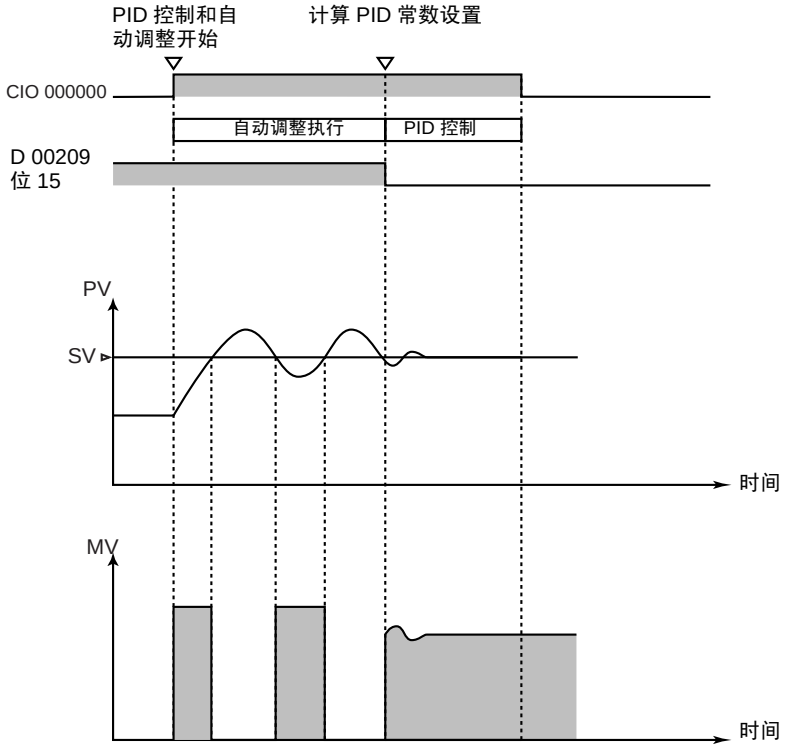



PID 控制启动
计算 PID 常数设置
CIO 000000
PID 控制
自动调整 执
PID 控制
W000000
D 00209
位 15
PV
SV
时间
MV
时间

例 2：启动 PIDAT(191) 自动调整

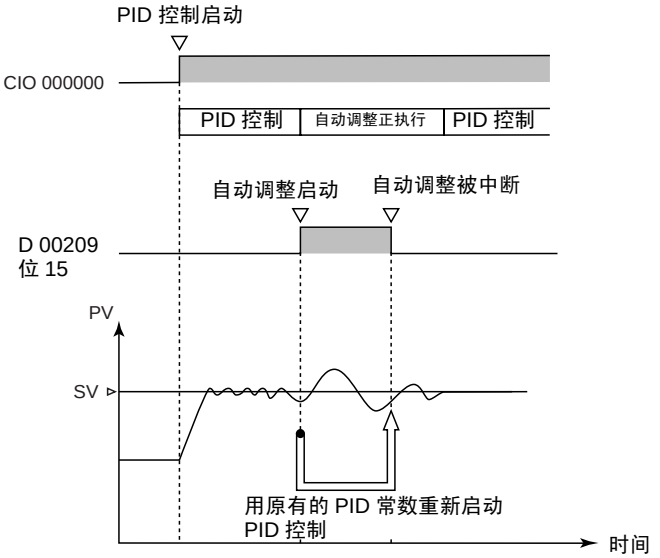
在 W 000000(OFF → ON) 上升沿， SET(532) 置 D00209 位 15(C+9) 为 ON，开始自动调整。当自动调整完成，计算 P、I、D 常数写到 C+1，C+2，C+3 中。在新常数下， PID 控制重新启动。





例 3：在完成前中断自动调整

通过设置 D00209(C+9) 位 15 从 ON ～ OFF，自动调整中断。在自动调整开始前， P， I 和 D 常数有效， PID 控制重新启动。

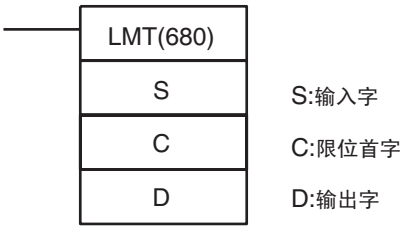


3-18-3 限位控制 :LMT(680)

用途

根据输人数据是否在上、下限之间控制输出数据。

梯形图符号



变化

变化	ON 条件时每次周期执行	LMT(680)
	上升沿微分执行一次	@LMT(680)
	下降沿微分执行一次	不支持
立即刷新定义		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数规定

区域	S	C	D
CIO 区	CIO 0000 ～ CIO 6143	CIO 0000 ～ CIO 6142	CIO 0000 ～ CIO 6143
工作区	W000 ～ W511	W000 ～ W510	W000 ～ W511
保持位区	H000 ～ H511	H000 ～ H510	H000 ～ H511
扶助位区	A000 ～ 959	A000 ～ A958	A448 ～ A959
定时器区	T0000 ～ T4095	T0000 ～ T4094	T0000 ～ T4095
计数器区	C0000 ～ C4095	C0000 ～ C4094	C0000 ～ C4095
DM 区	D00000 ～ D32767	D00000 ～ D32766	D00000 ～ D32767
无区号 EM 区	E00000 ～ E32767	E00000 ～ E32766	E00000 ～ E32767
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	En_00000 ～ En_32766 (n = 0 ～ C)	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #FFFF （二进制）	---	
数据寄存器	DR0 ～ DR15	---	DR0 ～ DR15

区域	S	C	D
索引寄存器	---		
使用索引寄存器 的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

当执行条件为 ON 时， LMT(680) 将根据具体输入数据（有符号 16 位二进制）是否在上、下限之间而控制输出数据。字 C 和 C+1 的内容如下表所示。

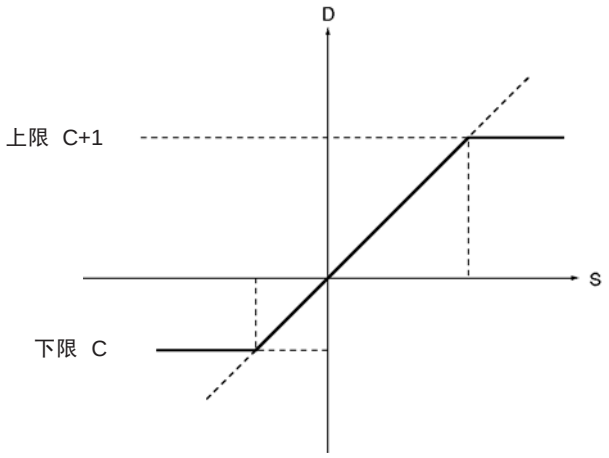
C	下限 （最小输出数据）
C+1	上限 （最大输出数据）

C 和 C+I 必须具有相同的区域类型。

如果输入数据 (S) 小于下限 (C)，下限数据将输出到 D, 并将小于标志置 ON。

如果输入数据 (S) 大于上限 (C+I)，上限数据将输出到 D, 并将大于标志置 ON。

如果输入数据 (S) 大于等于下限 (C)，小于等于上限 (C+I), 输入数据 (S) 将输出到 D。



标志

名称	标记	操作
错误标志	ER	如果上限小于下限时置 ON。 其它情况置 OFF。
大于标志	>	如果输入数据 (S) 大于上限置 ON。 其它情况置 OFF。
等于标志	=	如果结果为 0 置 ON。 其它情况置 OFF。
小于标志	<	如果输入数据 (S) 小于下限置 ON。 其它情况置 OFF。
负标志	N	如果最左位结果是 “1” 置 ON。 其它情况置 OFF。

注意

如果上限小于下限，将产生错误并将错误标志置 ON。

如果输入数据 (S) 大于上限，大于标志将置 ON。

如输出字 D 是 0000 hex，等于标志将置 ON。

如输入数据 (S) 小于下限，小于标志将置 ON。

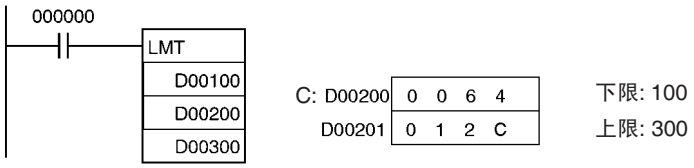
如果输出数据 D 的最左位是 “1”，负标志将置 ON。

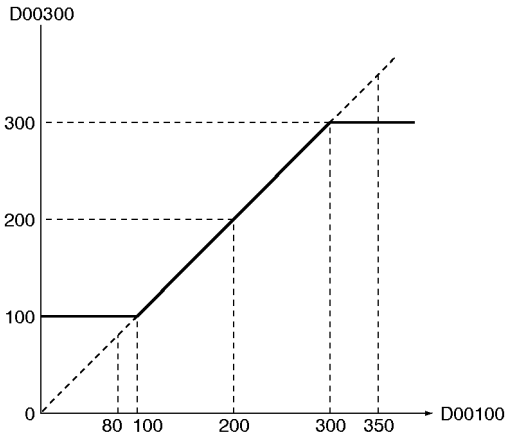
例

如果 D00100 是 0050H (80)，那么 0064H (100) 将输出到 D00300，因为 80 小于下限 100。

如果 D00800 是 00c8h(200)，那么 00C8H(200) 将输出到 D00300，因为 200 在上、下限之间。

如果 D00100 是 015EH(350)，那么 012CH (300) 将输出到 D00300，因为 350 大于上限 300。



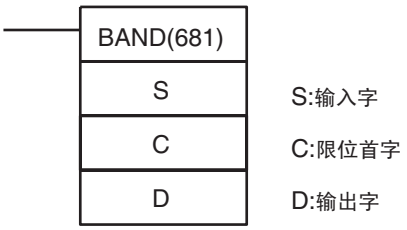


3-18-4 静带控制 :BAND(681)

用途

根据输人数据是否在静带范围内控制输出数据。

梯形图符号



变化

变化	ON 条件时每次周期执行	BAND(681)
	上升沿微分执行一次	@BAND(681)
	下降沿微分执行一次	不支持
立即刷新定义		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数规定

区域	S	C	D
CIO 区	CIO 0000～CIO 6143	CIO 0000～CIO 6142	CIO 0000～CIO 6143
工作区	W000 ～ W511	W000 ～ W510	W000 ～ W511
保持位区	H000 ～ H511	H000 ～ H510	H000 ～ H511
辅助位区	A000 ～ A959	A000 ～ A958	A448 ～ A959
定时器区	T0000 ～ T4095	T0000 ～ T4094	T0000 ～ T4095
计数器区	C0000 ～ C4095	C0000 ～ C4094	C0000 ～ C4095
DM 区	D00000 ～ D32767	D00000 ～ D32766	D00000 ～ D32767
无区号 EM 区	E00000 ～ E32767	E00000 ～ E32766	E00000 ～ E32767
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	En_00000 ～ En_32766 (n = 0 ～ C)	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #FFFF（二进制）	---	
数据寄存器	DR0 ～ DR15	---	DR0 ～ DR15
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

当执行条件为 ON 时，BAND(681) 将根据具体输入数据（有符号 16 位二进制）是否在上、下限之间而控制输出数据（静带）。字 C 和 C+1 的内容如下表所示。

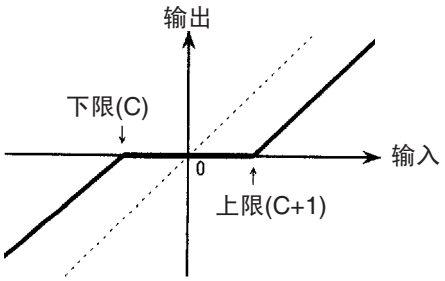
C	下限数据 （静带下限）
C+1	上限数据 （静带上限）

C 和 C+I 必须具有相同的区域类型。

如果输入数据 (S) 大于或等于下限 (C) 而小于或等于上限 (C+I)， 0000(H) 将输出到 D 并将等于标志置 ON。

如果输人数据 (S) 小于下限 (C)，则输人数据减下限的差值将输出到 D，并将小于标志置 ON。

如果输入数据 (S) 大于上限 (C+I)，则输人数据减上限数据的差值输出到 D，并将大于标志置 ON。



如果输出数据小于 8000(H) 或者大于 7FFF，那么符号变号。例如，若下限为 0100(H) 而输入数据为 8000(H)，那么输出数据为如下所示：
8000 (hex) [–32768] – 0100 (hex) [256] = 7F00 (hex) [32512]

标志

名称	标记	操作
错误标志	ER	如果上限小于下限时置 ON。 其它情况置 OFF。
大于标志	>	如果输入数据 (S) 大于上限置 ON。 其它情况置 OFF。
等于标志	=	如果结果为 0 置 ON。 其它情况置 OFF。
小于标志	<	如果输入数据 (S) 小于下限置 ON。 其它情况置 OFF。
负标志	N	如果最左位结果是 “1” 置 ON。 其它情况置 OFF。

注意

如果上限小于下限 , 将产生错误并将错误标志置 ON。

如果输入数据 (S) 大于上限，大于标志将置 ON。

如输出字 D 是 0000 H，等于标志将置 ON。

如输入数据 (S) 小于下限，小于标志将置 ON。

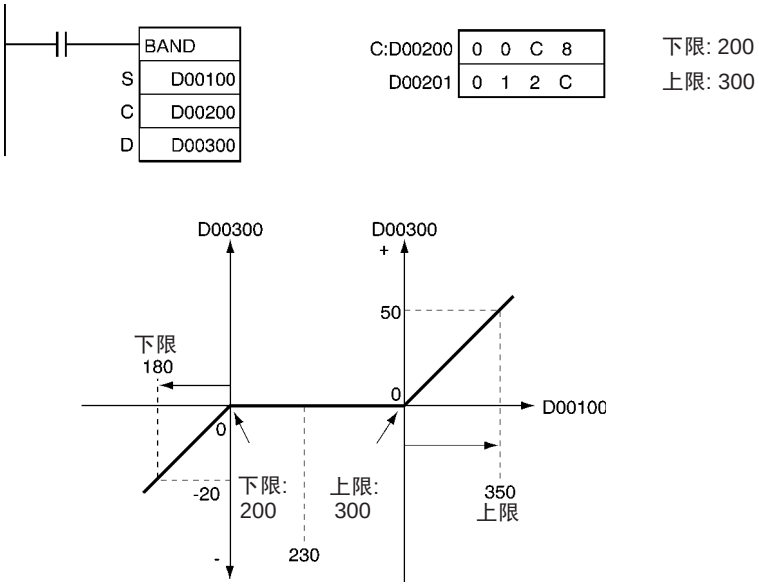
如果输出数据 D 的最左位是 “1”，负标志将置 ON。

例

如果 D00100 是 00B4H(180), 那么 180-200=FFECH（-20）将输出到 D00300, 因为 180 小于下限 200。

如果D00100是00E6H(230),那么0将输出到D00300,因为230在上、下限之间。

如果 D00100 是 015EH(350), 那么 350-300=0032H(50) 将输出到 D00300, 因为 350 大于上限 300。

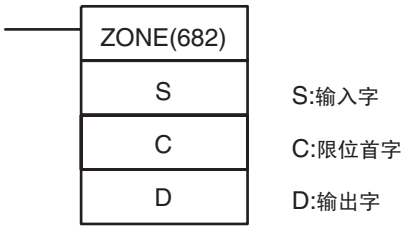


3-18-5 静域控制 :ZONE(682)

用途

将指定偏移量加入输入数据并输出结果。

梯形图符号



变化

变化	ON 条件时每次周期执行	ZONE(682)
	上升沿微分执行一次	@ZONE(682)
	下降沿微分执行一次	不支持
立即刷新定义		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数规定

区域	S	C	D
CIO 区	CIO 0000～CIO 6143	CIO 0000～CIO 6142	CIO 0000～CIO 6143
工作区	W000 ～ W511	W000 ～ W510	W000 ～ W511
保持位区	H000 ～ H511	H000 ～ H510	H000 ～ H511
辅助位区	A000 ～ A959	A000 ～ A958	A448 ～ A959
定时器区	T0000 ～ T4095	T0000 ～ T4094	T0000 ～ T4095
计数器区	C0000 ～ C4095	C0000 ～ C4094	C0000 ～ C4095
DM 区	D00000 ～ D32767	D00000 ～ D32766	D00000 ～ D32767
无区号 EM 区	E00000 ～ E32767	E00000 ～ E32766	E00000 ～ E32767

区域	S	C	D
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	En_00000 ～ En_32766 (n = 0 ～ C)	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #FFFF （二进制）	---	
数据寄存器	DR0 ～ DR15	---	DR0 ～ DR15
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

当执行条件为 ON 时，ZONE(682) 将根据具体输入数据（有符号 16 位二进制）是否在上、下限之间而控制输出数据。字 C 和 C+1 的内容如下表所示。

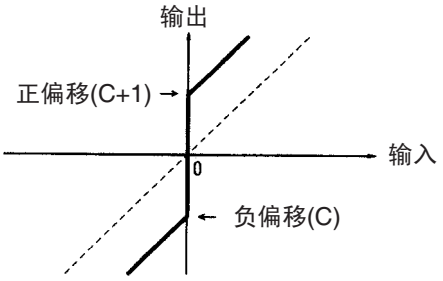
C	负偏移
C+1	正偏移

C 和 C+I 必须具有相同的区域类型。

如果输入数据 (S) 小于 0，输入数据加上负偏移输出到 D 并将小于标志置 ON。

如果输人数据 (S) 大于 0，输入数据加上正偏移输出到 D 并将大于标志置 ON。

如果输入数据 (S) 等于 0， 0000 将输出到 D，并将等于标志置 ON。



如果输出数据小于 8000(H) 或者大于 7FFF，那么符号反号。例如，对于负偏移量 FF00H 和输入数据 8000(H)，那么输出数据为如下所示：

8000 (hex) [–32768] – FF00 (hex) [–256] = 7F00 (hex) [32512]

标志

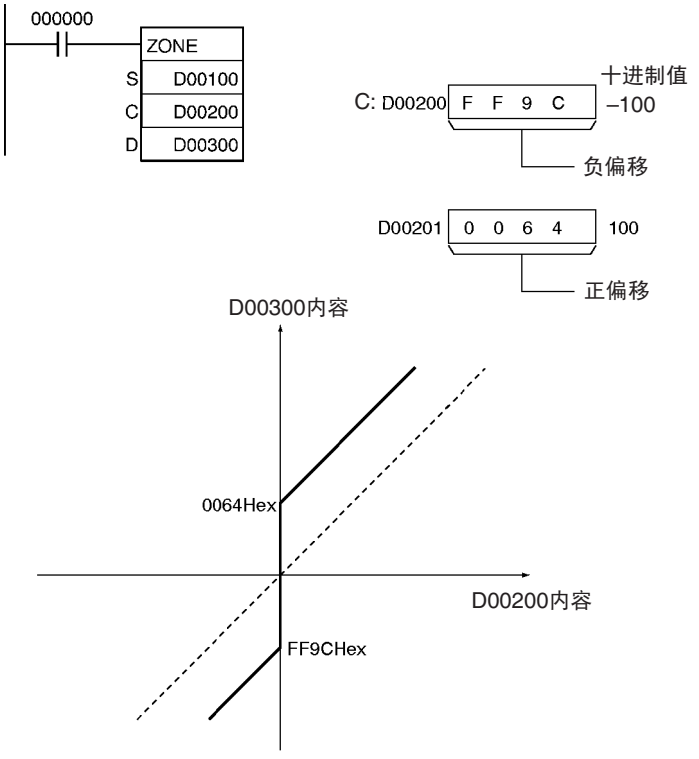
名称	标记	操作
错误标志	ER	如果上限小于下限时置 ON。 其它情况置 OFF。
大于标志	>	如果输入数据 (S) 大于上限置 ON。 其它情况置 OFF。
等于标志	=	如果结果为 0 置 ON。 其它情况置 OFF。
小于标志	<	如果输入数据 (S) 小于下限置 ON。 其它情况置 OFF。
负标志	N	如果最左位结果是 “1” 置 ON。 其它情况置 OFF。

注意

如果上限小于下限，将产生错误并将错误标志置 ON。
如果输入数据 (S) 大于上限，大于标志将置 ON。
如输出字 D 是 0000 H，等于标志将置 ON。
如输入数据 (S) 小于下限，小于标志将置 ON。
如果输出数据 D 的最左位是 “1”，负标志将置 ON。

例

当 CIO00000 为 ON 时，若 D00100 的值小于 0，偏移 -100 将赋给 D00100，并将经果存入 D00300。
如果结果 D00100 的值是 0，那么 0000H 将被存入 D00300。
如果D00100的值大于0，则偏移量+100赋给D00100，并将结果存入D00300。

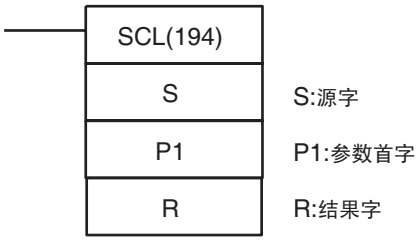


3-18-6 标度：SCL(194)

用途

根据指定的线性函数将无符号二进制数据转换为无符号 BCD 码。

梯形图符号



变化

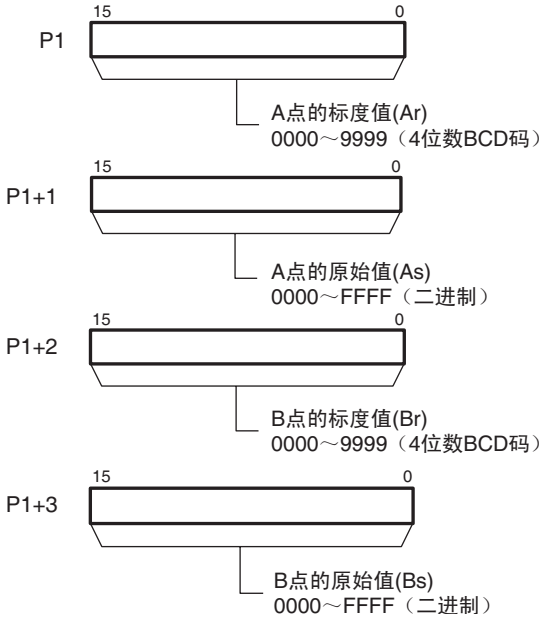
变化	ON 条件时每次周期执行	SCL(194)
	上升沿微分执行一次	@SCL(194)
	下降沿微分执行一次	不支持
立即刷新定义		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

从第 1 个参数字 (PI) 开始的 4 个字的内容如下图所示。



注 P1 到 P1+3 必须在同一区域内。

操作数规定

区域	S	P1	R
CIO 区	CIO 0000～CIO 6143	CIO 0000～CIO 6140	CIO 0000～CIO 6143
工作区	W000 ～ W511	W000 ～ W508	W000 ～ W511
保持位区	H000 ～ H511	H000 ～ H508	H000 ～ H511
辅助位区	A000 ～ A959	A000 ～ A956	A448 ～ A959
定时器区	T0000 ～ T4095	T0000 ～ T4092	T0000 ～ T4095

区域	S	P1	R
计数器区	C0000 ～ C4095	C0000 ～ C4092	C0000 ～ C4095
DM 区	D00000 ～ D32767	D00000 ～ D32764	D00000 ～ D32767
无区号 EM 区	E00000 ～ E32767	E00000 ～ E32764	E00000 ～ E32767
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	En_00000 ～ En_32764 (n = 0 ～ C)	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---		
数据寄存器	DR0 ～ DR15	---	DR0 ～ DR15
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++ ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

SCL(194) 用于将源字 S 中的无符号二进制数转换为无符号 BCD 数 , 并根据由点 (As,Ad) 和 (Bs,Bd) 定义的线性函数将结果存入结果字 R。包含了点 (As, Ar) 和 (Bs,Br) 的坐标的第一个字的地址是由第一个参数字 PI 指定的 , 这些点在标度前由 (As 和 Bs) 两值定义 , 标度后由 (Ar,Br) 两值定义。

转换公式如下：

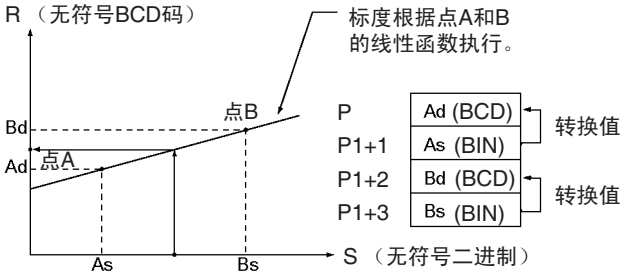
$$R = Bd - \frac{(Bd - Ad)}{(Bs\text{-}As)\text{的转换BCD码}} \times (Bs\text{-}S)\text{的转换BCD码}$$

直线的斜率如下：

$$R = Bd - \frac{(Bd - Ad)}{(Bs\text{-}As)\text{的转换BCD码}}$$

点 A 和 B 可以用正斜率或负斜率定义一条线。使用负斜率可以使转换反向。

计算结果舍入为最近的整数。如果结果小于 0000，结果输出为 0000。若结果大于 9999，则输出为 9999。



SCL(194) 可用于标度模拟量信号，根据用户定义的标度参数，将模拟输入单元输入的模拟信号转换。例如，如果输入为 I 到 5V 的模拟量输入单元以 0000 ～ 0FA0 的十六进制数输入到内存，那么内存中的值可用 SCL(194) 标度为 50 ～ 200 C。

SCL(194) 将无符号二进制数转换为无符号 BCD 码。为了转换负值，在程序中使用 SCL(194) 前，应先加上最大负值。

SCL(194) 不能输出负值到结果字 R，如果结果为负值则输出 0000 到 R。

标志

名称	标记	操作
错误标志	ER	C(Ar) 或 C+I(Br) 的内容非 BCD 码时置 ON。 C+I(As) 和 C+3(Bs) 的内容相等时置 ON。其它情况置 OFF。
等于标志	=	结果为 0 置 ON。 其它情况置 OFF。

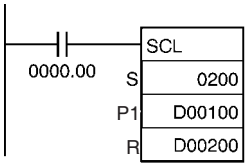
注意

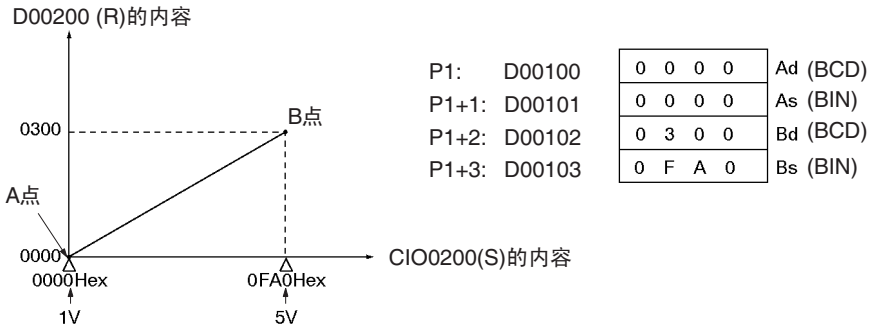
如果 Ar(C) 和 Br(C+2) 不是 BCD 码或者 As(C+1) 和 Bs(C+3) 的值相等，则会产生错误，错误标志置 ON。如果结果字 D 的内容为 0000，相等标志置 ON。

例

在下例中，假定要转换从 1 ～ 5V 的模拟量信号，并以 0000 ～ 0FA0 间的十六进制数输入到 CIO0200。 SCL(194) 用于将 CIO0200 的值转换 （标度）为从 0000 ～ 0300 的 BCD 码。

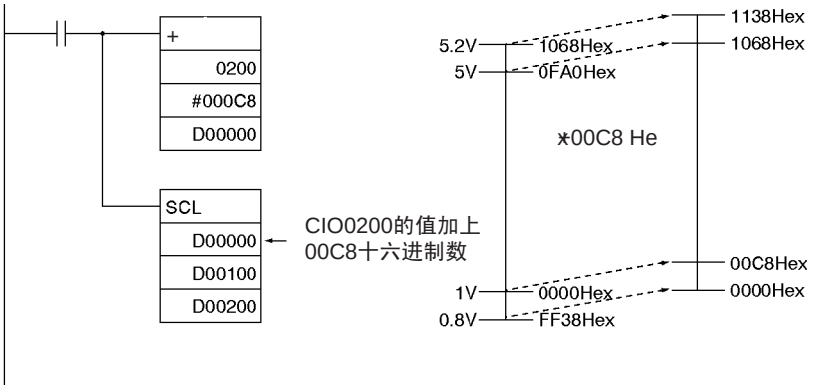
当 CIO 000000 为 ON 时，CIO0200 的内容用由点 A(0000,0000) 和点 B(0FA0, 0300) 定义的线性函数进行标度。这些点的坐标放在 D00100 ～ D00103 中，结果输出到 D00200。

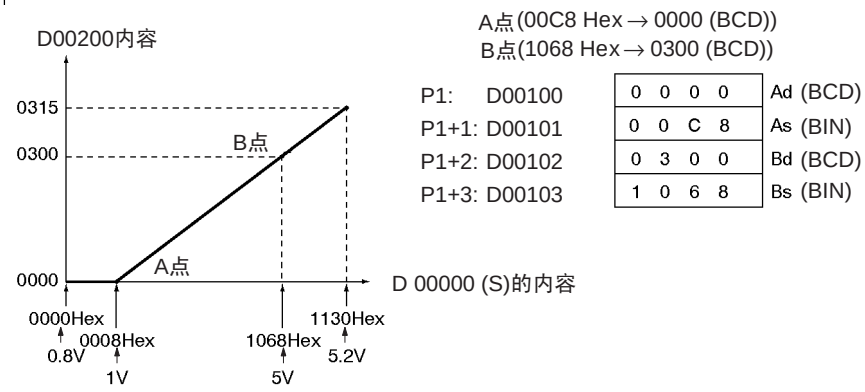




负值

一个模拟量输入单元实际输入值是用从 FF38 ～ 1068 的十六制数代表 0.8 ～ 5.2V，而 SCL(194) 只能处理从 0000 ～ FFFF 的十六进制的无符号二进制数，使 SCL(194) 不能直接处理低于 1V（即 (0000H) 值的有符号二进制数，即 FF38 ～ FFFF 十六进制数。在实际应用中，在使用 SCL(194) 之前给所有值加上 00C8，这样十六进制数 FF38 就相当于十六进制数 0000，示例如下：

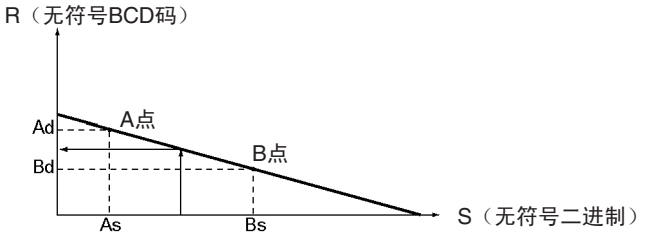




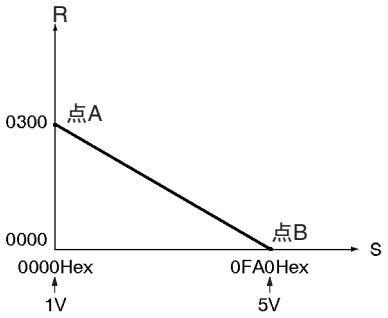
在此例中，从 0000 ～ 00C8 的十六进制数值将被转换为负值。而 SCL(194) 只能输出从 0000 ～ 9999 的无符号 BCD 码，因此只要 D0000 的内容是在 0000 到 00C8 的十六进制数之间，输出均为 BCD 码 0000。

反向标度

也可以通过设定 As<Bs 和 Ar>Br 来实现反向标度，这样将产生如下关系：



例如，如下图所示，反向标度可用于分别转换（反向标度）1 ～ 5v（0000 ～ 0FAO 的十六进制数）为 0300 ～ 0000。

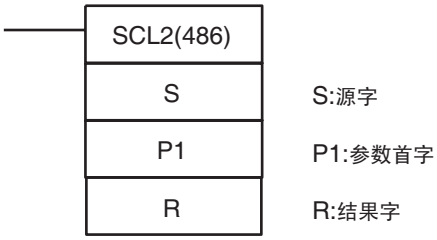


3-18-7 标度 2：SCL2(486)

用途

根据指定的线性函数将带符号二进制数据转化为带符号 BCD 码数据。

梯形图符号



变化

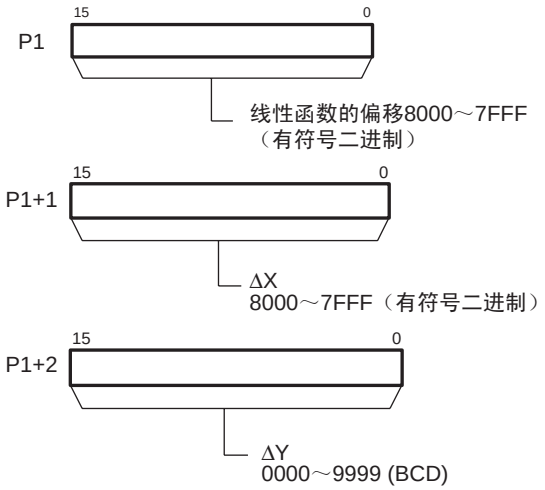
变化	ON 条件时每次周期执行	SCL2(486)
	上升沿微分执行一次	@SCL2(486)
	下降沿微分执行一次	不支持
立即刷新定义		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

从第 1 个参数字 (P1) 开始的 3 个字的内容如下图所示。



注 P1 ～ P1+2 必须在同一区域内。

操作数规定

区域	S	P1	R
CIO 区	CIO 0000～CIO 6143	CIO 0000～CIO 6141	CIO 0000～CIO 6143
工作区	W000 ～ W511	W000 ～ W509	W000 ～ W511
保持位区	H000 ～ H511	H000 ～ H509	H000 ～ H511
辅助位区	A000 ～ A959	A000 ～ A957	A448 ～ A959
定时器区	T0000 ～ T4095	T0000 ～ T4093	T0000 ～ T4095
计数器区	C0000 ～ C4095	C0000 ～ C4093	C0000 to C4095
DM 区	D00000 ～ D32767	D00000 ～ D32765	D00000 ～ D32767
无区号 EM 区	E00000 ～ E32767	E00000 ～ E32765	E00000 ～ E32767
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	En_00000 ～ En_32765 (n = 0 ～ C)	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---		
数据寄存器	DR0 ～ DR15	---	DR0 ～ DR15
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

SCL2(486)用来将源字S中的有符号二进制数转化为有符号BCD码数据（BCD码包含其绝对值，符号由进位标志表示），并根据斜率 (△ X, △ Y) 和偏移确定的线性函数，将结果存入结果字 R 中。包含△ X，△ Y 和偏移的第一个字的地址是由第一个参数字 P1 确定的，结果的符号由进位标志指示（ON：负，OFF：正）。

转换公式如下：

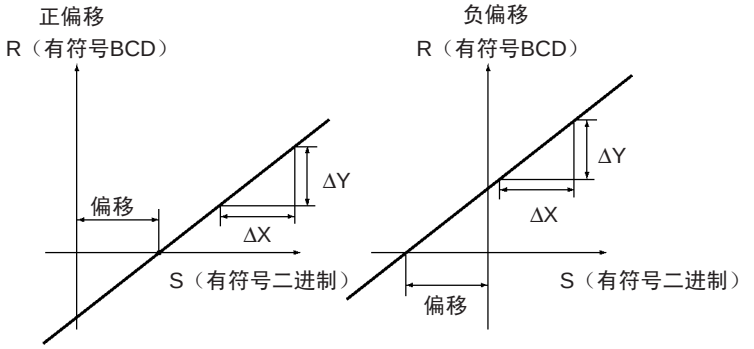
$$R = \frac{\Delta Y}{\Delta X\text{的BCD码}} \times （S\text{的BCD转换码}）-（\text{偏移的BCD转换码}）$$

直线的斜率是ΔY/ΔX。

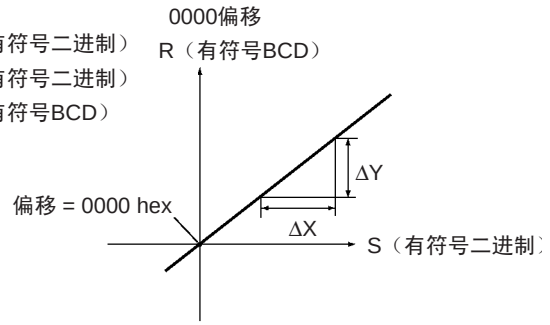
偏移和斜率可以是正数， 0 或负数，使用负斜率可以使标度反向。

计算结果舍入为最近的整数。

R 中的结果将是绝对值 BCD 转换，并且结果符号由进位标志指示。结果因此可以在 -9999 ～ 9999 之间。



P1	偏移	（有符号二进制）
P1+1	ΔY	（有符号二进制）
P1+2	ΔX	（有符号BCD）



SCL2(486) 可用于根据用户定义的标度参数从模拟量输入单元标度模拟量信号转换值的结果。例如，如果输入为 1 ～ 5V 的模拟量输入单元以 0000 ～ 0FA0 的十六进制数输入内存，那么内存中的值可用SCL2(486)标度为-100～20° C。

SCL2(486) 将有符号二进制数转化为有符号 BCD 数。因此 S 的负值可以直接处理， SCL2(486) 也输出负值。

标志

名称	标记	操作
错误标志	ER	C+1(ΔY) 的内容是 0000 时置 ON C+2(ΔY) 的内容不是 BCD 置 ON。 其它情况置 OFF。
等于标志	=	结果为 0 置 ON。 其它情况置 OFF。
进位标志	CY	结果为负数置 ON。 结果为 0 或正置 OFF。

注意

如果△ X(C+I) 是 0000 或△ Y(C+2) 的值不是 BCD，将产生错误，并将错误标志置 ON。

当结果字 D 的内容是 0000 时，等于标志将置 ON。

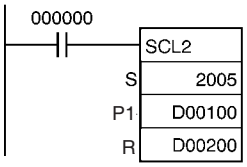
如果存入结果字中的数是负数，进位标志将置 ON。

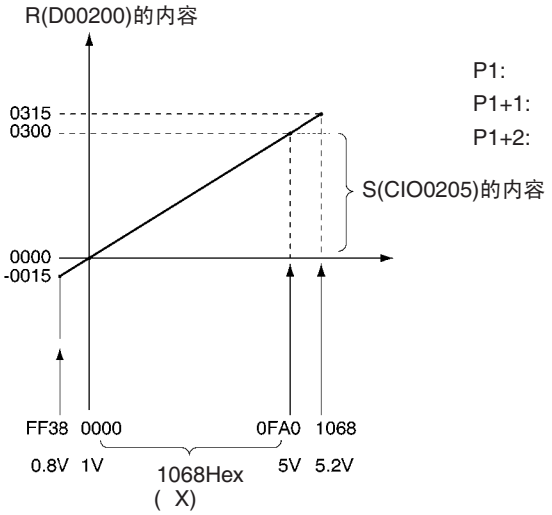
例

标度 1 ～ 5V 模拟量输入为 0 ～ 300

在下例中 , 假定要转换从 1 ～ 5V 的模拟量信号，并以 0000 ～ 0FA0 间的十六进制数输入到 CIO00205， SCL2(486) 用于将 CIO00205 的值转换（标度）为从 0000 ～ 0300 间的 BCD 码。

当 CIO000000 是 ON 时，CIO00205 的内容用由△ X(0FA0)，△ Y(0300) 和偏移 (0) 定义的线性函数进行标度。上述值放在 D00100 ～ D000102 之间，并将结果输出到 D00200。



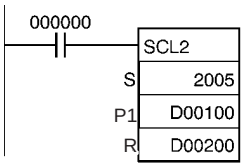


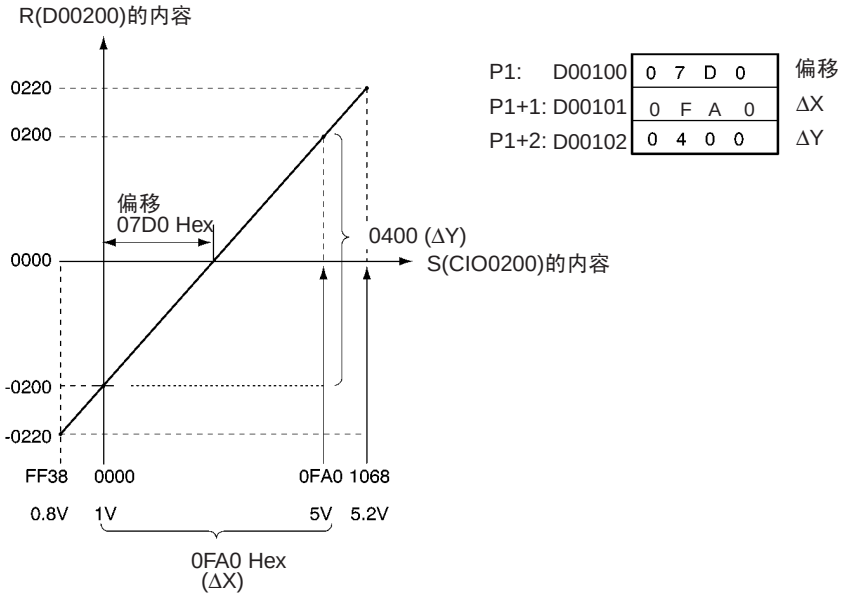
P1:	D00100	0	0	0	0	偏移
P1+1:	D00101	0	F	A	0	X
P1+2:	D00102	0	3	0	0	Y

标度 1 ～ 5V 模拟量输入为 -200 ～ 200

在下例中 , 假定要转换从 1 ～ 5V 的模拟量信号，并以 0000 ～ 0FA0 之间的十六进制数输入到 CIO2005。 SCL2(486) 用于将 CIO2005 的值转换（标度）为从 -0200 ～ 0200 间的 BCD 码。

当 CIO000000 为 ON 时， CIO2005 中的内容利用由△ X(0FA0)，△ Y(0400)和偏移(0700)定义的线性函数进行标度。上述值存贮在 D00100～D00102 中，结果输出到 D00200。



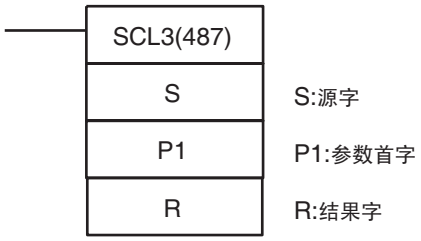


3-18-8 标度 3：SCL3(487)

用途

根据指定线性函数，将有符号 BCD 数转化为有符号二进制数，定义线性函数中可输人偏移。

梯形图符号



变化

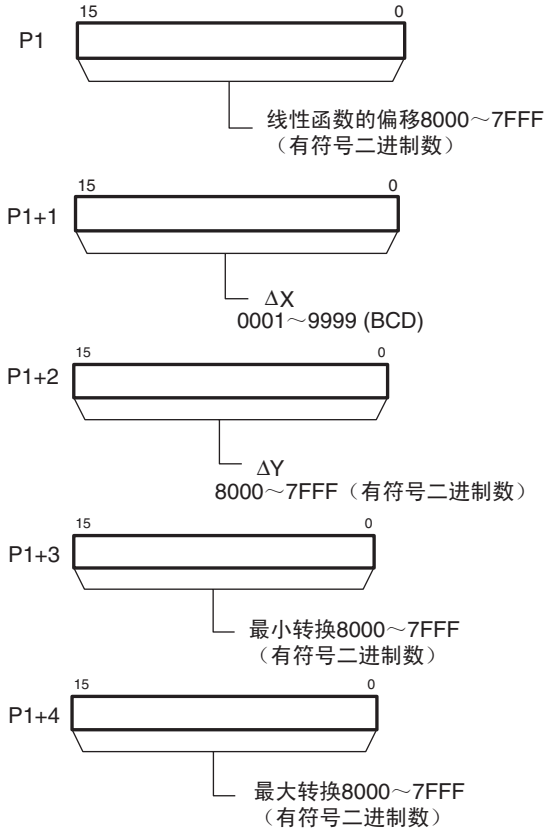
变化	ON 条件时每次周期执行	SCL3(487)
	上升沿微分执行一次	@SCL3(487)
	下降沿微分执行一次	不支持
立即刷新定义		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

从第 1 个参数字 (P1) 开始的 5 个字的内容如下图所示。



注　P1 ～ P1+4 必须在同一区域内。

操作数规定

区域	S	P1	R
CIO 区	CIO 0000～CIO 6143	CIO 0000～CIO 6139	CIO 0000～CIO 6143
工作区	W000 ～ W511	W000 ～ W507	W000 ～ W511
保持位区	H000 ～ H511	H000 ～ H507	H000 ～ H511
辅助位区	A000 ～ A447 A448 ～ A959	A000 ～ A443 A448 ～ A955	A448 ～ A959
定时器区	T0000 ～ T4095	T0000 ～ T4091	T0000 ～ T4095
计数器区	C0000 ～ C4095	C0000 ～ C4091	C0000 ～ C4095
DM 区	D00000 ～ D32767	D00000 ～ D32763	D00000 ～ D32767
无区号 EM 区	E00000 ～ E32767	E00000 ～ E32763	E00000 ～ E32767
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	En_00000 ～ En_32763 (n = 0 ～ C)	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		

区域	S	P1	R
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---		
数据寄存器	DR0 ～ DR15	---	DR0 ～ DR15
索引寄存器	---		
使用索引寄存器的 间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

SCL3(487) 用来将源字 S 中的有符号 BCD 数根据斜率 (△ X, △ Y) 和偏移定义的线性函数转换为有符号二进制数，将结果存入结果字 R 中。最大和最小转换值也是指定的，包含△ X，△ Y 偏移，最大转换和最小转换的第一个字的地址由第一个参数字 PI 确定，结果的符号由进位标志状态 （ON ：负、 OFF：正）指示。

转换公式如下：

$$R = \frac{\Delta Y}{\Delta X\text{的BCD码}} \times （S\text{的二进制转换码}）+（偏移）$$

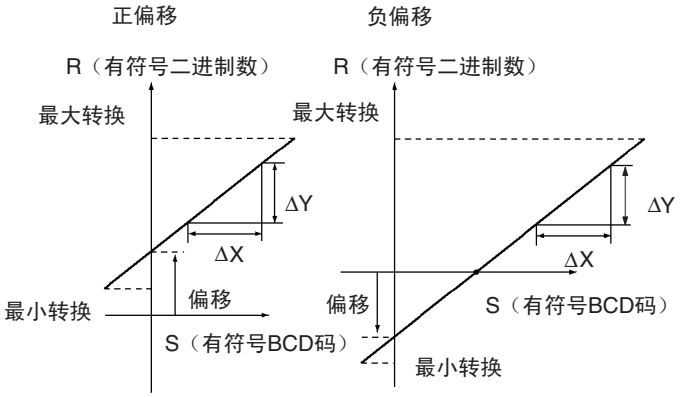
直线的斜率是ΔY/ΔX。

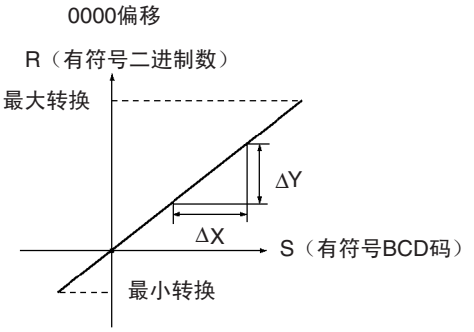
偏移和斜率可以是正数， 0 或负数，使用负斜率可以使标度反向。

计算结果舍入为最近的整数。

R 中的结果将是绝对值 BCD 转换，并且结果符号由进位标志指示。结果因此可以在 -9999 ～ 9999 之间。

如果结果小于最小转换值，最小转换值将被作为结果输出。如果结果大于最大转换值，最大转换值将被输出。





SCL3(487) 用于使用户定义标度将数据转换为模拟量输出单元用的有符号二进制数。例如，SCL3(487) 可以将 0 ～ 20° C 转换为 0000 ～ 0FA0H，并从模拟量输出单元输出一个 1 ～ 5V 的模拟量输出信号。

标志

名称	标记	操作
错误标志	ER	S 的内容不是 BCD 码置 ON C+1(ΔX) 的内容不是 0001 ～ 9999 之间 BCD 码时 ON。 其它情况置 OFF。
等于标志	=	结果为 0 置 ON。 其它情况置 OFF。
负标志	N	R （结果）的 MSB 为 1 时置 ON。 结果为 0 或正数置 OFF。

注意

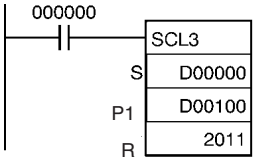
如果 S 的内容不是 BCD 数或 ΔX(C+1) 的值不在 0001 ～ 9999BCD 之间，则将产生错误 , 错误标志将置 ON。

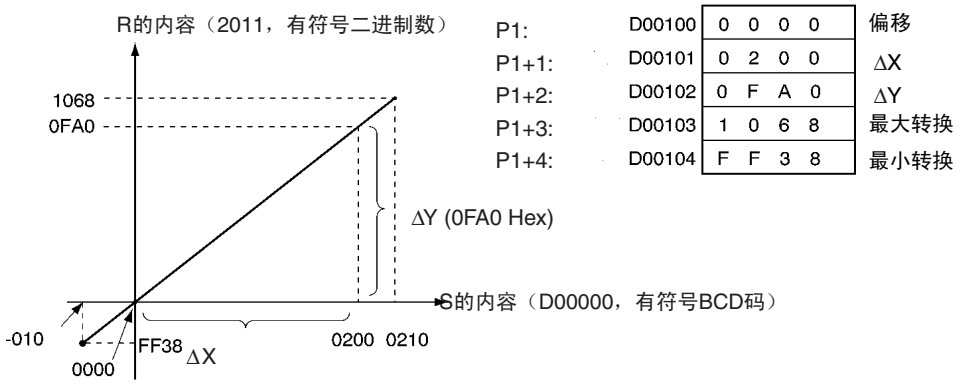
当结果字 D 的内容为 0000 时，等于标志将置 ON。

如果 R 中结果的 MSB 是 1，即结果是负数，负标志将置 ON。

例

当一个 0 ～ 200 之间的值被标度为一个模拟信号 （例如 1 ～ 5V）时，一个从 0000～0200 的有符号 BCD 码被转换为模拟量输出单元用的有符号二进制数值。在下例中当 CIO000000 为 ON 时，D00000 的内容利用由 ΔX(0200)，ΔY(0FA0) 和偏移 (0) 定义的线性函数被标度。这些值存贮在 D00100～D00102 中。D00000 中的 BCD 码值的符号由进位标志指示。结果输出到 CIO 02011。



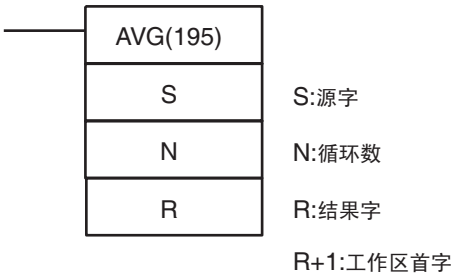


3-18-9 平均值：AVG(195)

用途

计算指定循环数中输入字的平均值。

梯形图符号



变化

变化	ON 条件时每次周期执行	AVG(195)
	上升沿微分执行一次	不支持
	下降沿微分执行一次	不支持
立即刷新功能		不支持

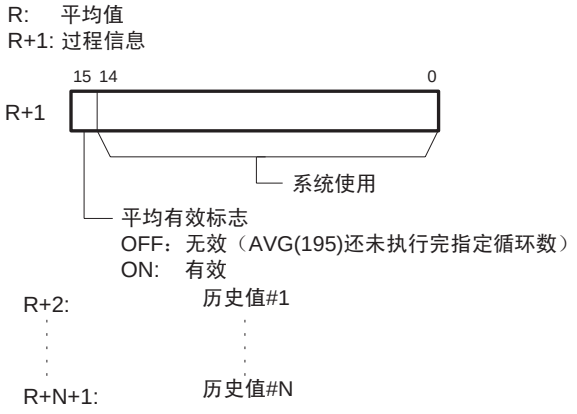
适用程序区

块程序区	步程序区	子程序	中断任务
不允许	OK	OK	OK

操作数

N：循环数
循环数必须为 0001 或 0040 之间的 16 进制数 （0 ～ 64 个循环）。

R：结果字和 R+1：工作区首字
R 包含了指定循环之后的平均值。R+1 提供了有关求均值过程的信息，R+2 ～ R+N+1 包含了如下图所示的 S 的历史值。



注 R ～ R+N+1 必须在同一区域内。

操作数规定

区域	S	N	R
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #FFFF（二进制）	#0001 ～ #0040（二进制）	---
数据寄存器	DR0 ～ DR15		---
索引寄存机	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15		

说明

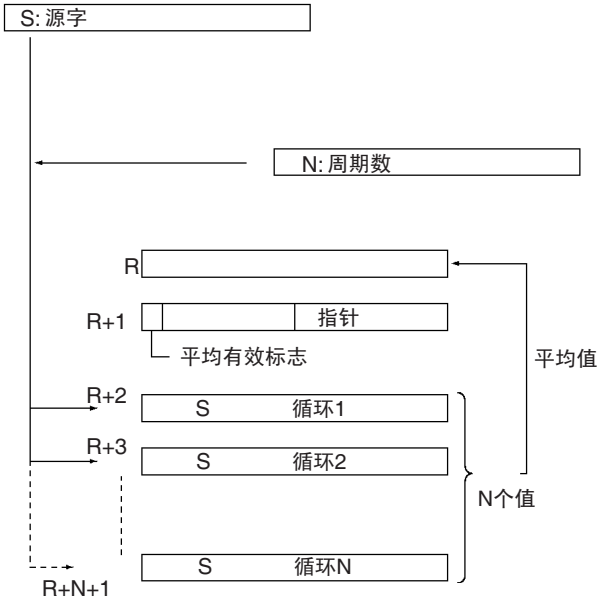
当执行条件为 ON 时，在前 N-1 个循环，AVG(195) 按次序将 S 的值写入从 R+2 开始的字中。历史值指针 （R+1 的第 00 ～ 07 位）在每写入一个字后加 1。 S 的内容按原值输出到 R 且平均值标志（R+1 的第 15 位）保持 OFF 直至写入第 N 个字。

当第 N 个值写入 R+N+1 后，计算已写入的所有的值的平均值，将平均值以无符号二进制数输出到 R，并将平均值标志（R+1 的第 15 位）置 ON。在下面的循环中， R 的值按最新的 N 个 S 值更新。

N 的最大值为 64。

历史指针值在写入 N-1 个值后将被复位为 0。

输出到 R 的平均值将被舍入为最近的整数。



标志

名称	标记	操作
错误标志	ER	N 的内容是 0 置 ON。 其它情况置 OFF。

注意

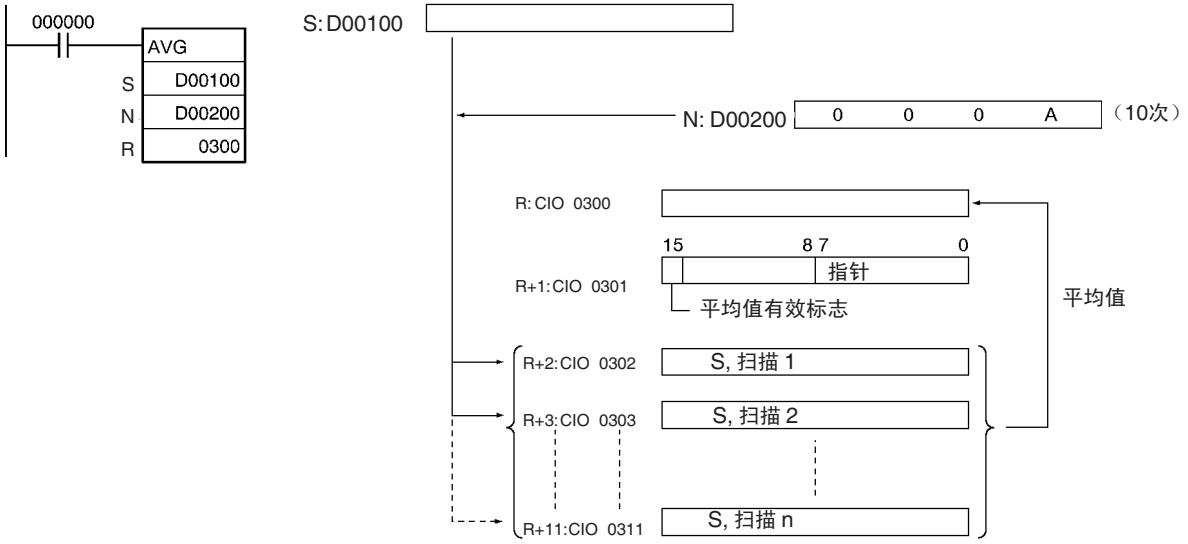
工作区首字 (R+1) 的内容在每次执行条件由 OFF 变为 ON 时清零。

工作区首字 (R+1) 的内容在操作开始第一次执行程序时不清零。

如果在第一个程序区扫描执行 AVG(195)，在程序中清除工作区首字。

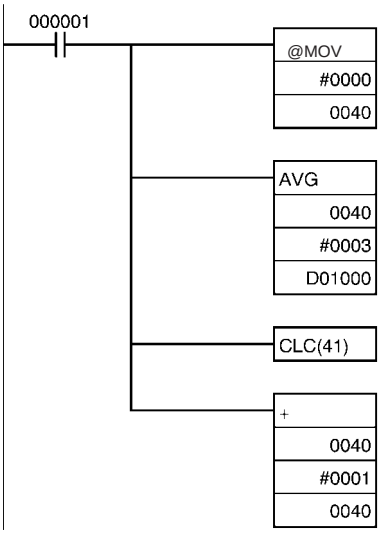
如果 N （循环次数）为 0000，将产生一个错误，错误标志置 ON。

在下例中当 CIO 000000 为 ON 时，D00100 的内容在 D00200 中指定扫描数的每次扫描时存储一次。内容被依次存储在从 CIO 00302 ～ CIO 00311 的 10 个字中，这 10 个字的内容的平均值被放入 CIO 00300，然后 CIO 00301 的第 15 位置 ON。



例

在下例中， CIO 0040 中的内容被设置为 0，然后在执行时每个周期加 1。在前两个周期中，AVG(195) 将 CIO 0040 的内容移至 D 01002 和 D 01003。D 01001 的内容也发生了变化 （它可用于确认 AVG(195) 的结果已改变）。在第三个和随后的循环中，AVG(195) 计算从 D 01002 和 D 01004 中的内容的平均值并将平均值写入 D 01001。



	第一个循环	第二个循环	第三个循环	第四个循环
CIO 0040	0000	0001	0002	0003

D01000	0000	0001	0001	0002	平均值指针
D01001	0001	0002	8000	8001	
D01002	0000	0000	0000	0003	IR40的3个历史值
D01003	---	0001	0001	0001	
D01004	---	---	0002	0002	

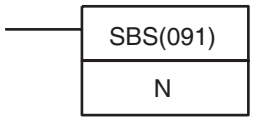
3-19 子程序

3-19-1 子程序调用：SBS(091)

用途

调用指定编号的子程序并执行该程序。

梯形图符号



变化

变化	ON 条件时每次周期执行	SBS(091)
	上升沿微分执行一次	@SBS(091)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

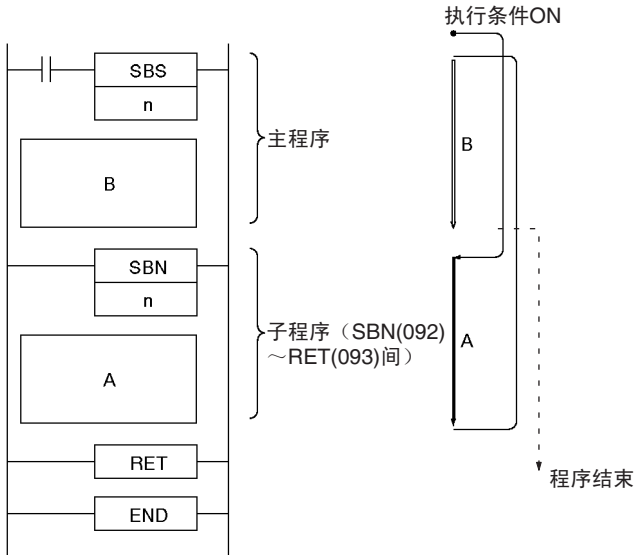
N：子程序号
指定 0 ～ 1023 之间的子程序编号。

操作数规定

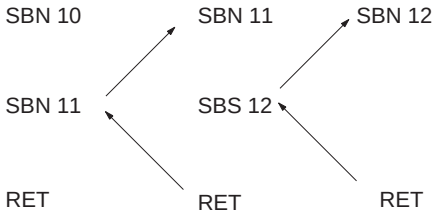
区域	N
CIO 区	---
工作区	---
保持位区	---
辅助位区	---
定时器区	---
计数器区	---
DM 区	---
无区号 EM 区	---
有区号 EM 区	---
二进制间接 DM/EM 寻址	---
BCD 间接 DM/EM 寻址	---
常数	0 ～ 1023 （十进制）
数据寄存器	---
索引寄存器	---
使用索引寄存器的间接寻址	---

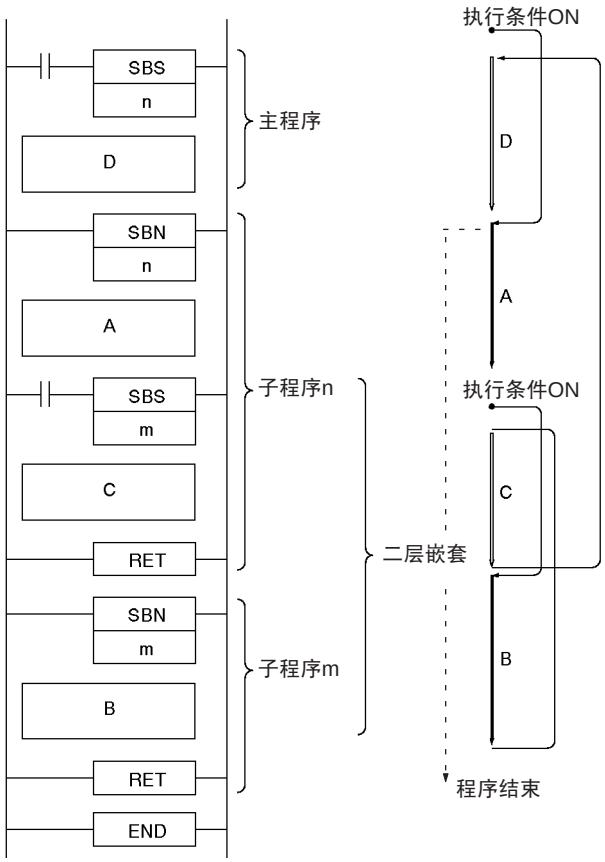
说明

SBS(091) 调用指定编号的子程序，子程序指 SBN(092) 和 RET(093) 之间的程序段，当子程序执行完成后，程序从 SBS(091) 的下一个指令继续执行。



子程序可被嵌套到 16 层。嵌套是在一个子程序内有另一个子程序被调用，如下图所示，子程序被嵌套到 3 层。



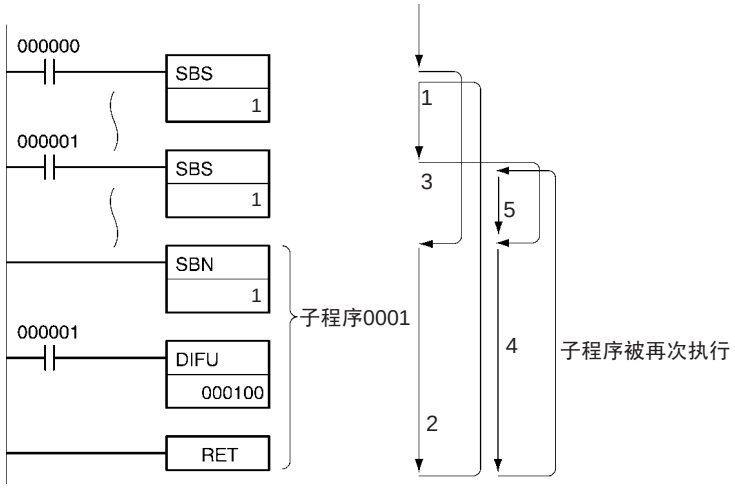


注 同一子程序可以在程序中调用多次。

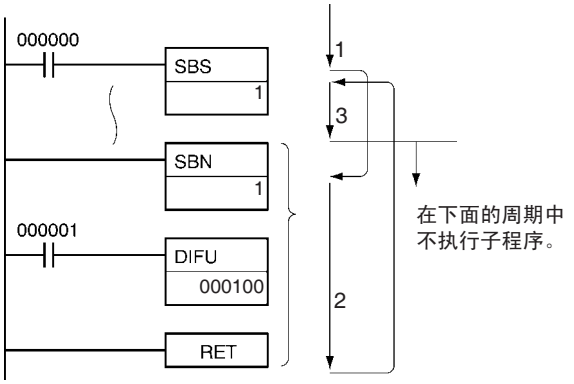
子程序和微分

在子程序中使用微分指令（DIFU(013)，DIFU(014) 或上升 / 下降沿微分指令）时必须注意下列事项。

如果一个子程序在同一个周期中执行多次，那么子程序中的微分指令是不可预见的。在下例中 CIO 000000 为 ON 时执行子程序 0001，而当 CIO 00001 由 OFF 变时 ON，CIO 00100 由 DIFU(013) 置 ON。如果在同一周期中 CIO 00001 为 ON，则将再次执行子程序 0001，而此时 DIFU(013) 将不检验 CIO 00001 的状态，并将 CIO 00001 置 OFF。



相反，如果子程序中微分指令 （DIFU(013) 或 DIFU(014)）执行后使输出变为 ON，但相同子程序没有被第 2 次调用，则微分输出保持为 ON。



在下例中若 CIO 000000 为 ON，则执行子程序 0001。当 CIO 000001 由 OFF 变为 ON 时， DIFU(013) 将输出 CIO00100 置为 ON。在下面的周期中若 CIO 00000 为 OFF，将不再执行子程序 0001，输出 CIO 000100 保持 ON。

标志

名称	标记	操作
错误标志	ER	嵌套超过 16 层时置 ON。 指定的子程序号不存在时置 ON。 子程序调用自身时置 ON。 调用一个正在执行的子程序时置 ON。 指定的子程序未在当前任务中定义过时置 ON。 其它情况置 OFF。

注意

SBS(091) 相应的 SBN(092) 程序必须编在同一任务中。如果任务中无相应的 SBN(092) 将产生一个错误。

如果 SBS(091) 在由 IL(002) 和 ILC(003) 联锁的程序段中，将作为 NOP(000) 处理。

在下述情况下即使执行了 SBS(091)，也不能调用子程序：

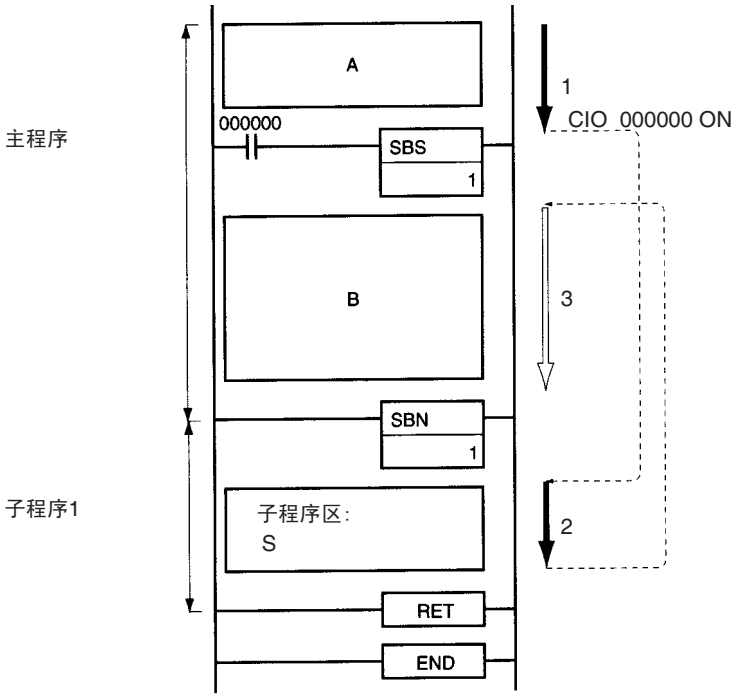
1,2,3...

1. 指定子程序未在当前任务中定义。
2. 子程序调用自身。
3. 子程序嵌套超出了 16 层。
4. 指定子程序正在执行。

例

例 1：顺序 （无嵌套）子程序

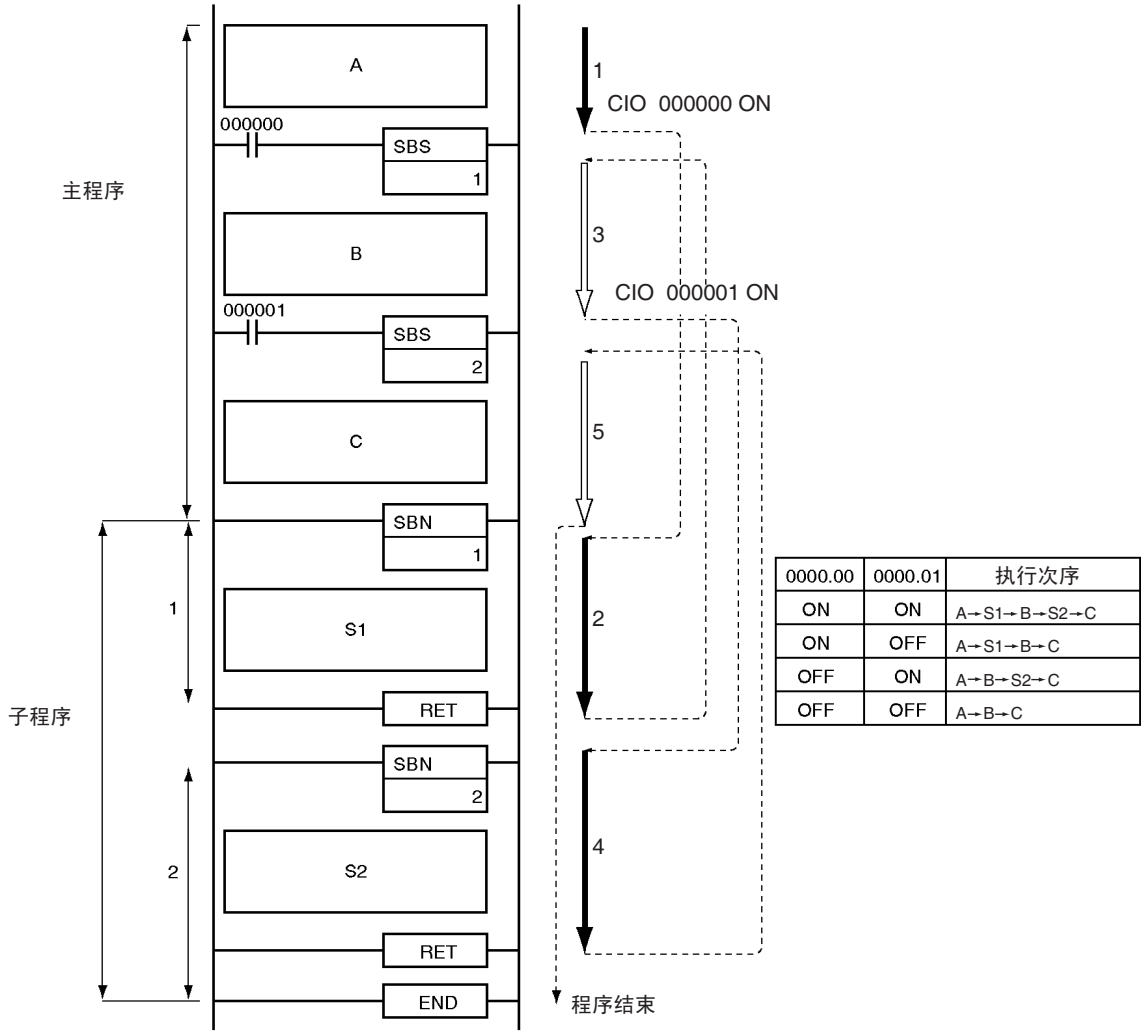
在下例中当 CIO 000000 为 ON 时，执行子程序 1，并返回到 SBS(091) 后的下一个指令，然后执行主程序的剩余部分 （完成 SBN(092)#1 前的指令）。



0000.00	执行次序
ON	A→S→B
OFF	A→B

例 2：顺序 （无嵌套）子程序

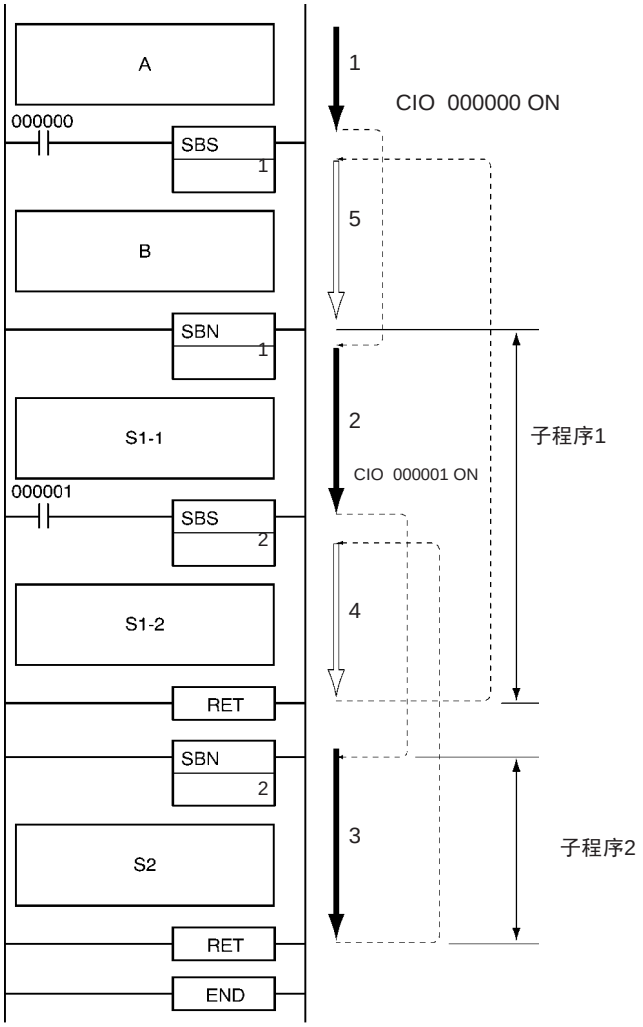
在下例中当 CIO 000000 为 ON 时，执行子程序 1 并返回到 SBS(091)#1 后的下一个指令。当 CIO 000001 为 ON 时，执行子程序 2 并返回到 SBS(091)#2 后的下一个指令。



0000.00	0000.01	执行次序
ON	ON	A→S1→B→S2→C
ON	OFF	A→S1→B→C
OFF	ON	A→B→S2→C
OFF	OFF	A→B→C

例 3：嵌套子程序

在下例中当 CIO 000000 为 ON 时执行子程序 1。当 CIO 000001 为 ON 时，则在子程序 #1 中执行子程序 2，子程序 2 执行完后返回到 SBS(091)#2 后的下一个指令。程序 1 继续执行，在子程序 1 执行完后并返回到 SBS(091)#1 后的下一个指令。



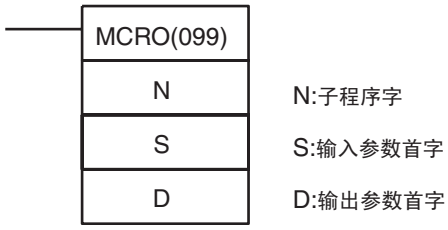
0000.00	0000.01	执行次序
ON	ON	A→S1-1→S2→S1-2→B
ON	OFF	A→S1-1→S1-2→B
OFF	ON	A→B
OFF	OFF	A→B

3-19-2 宏：MCRO(099)

用途

调用指定子程序号的子程序，并用 S ～ S+3 中的输入参数和 D ～ D+3 中的输出参数执行程序。

梯形图符号



变化

变化	ON 条件时每次周期执行	MCRO(099)
	上升沿微分执行一次	@MCRO(099)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

N：子程序编号
指定 0 ～ 1023 之间的十进制数子程序编号。

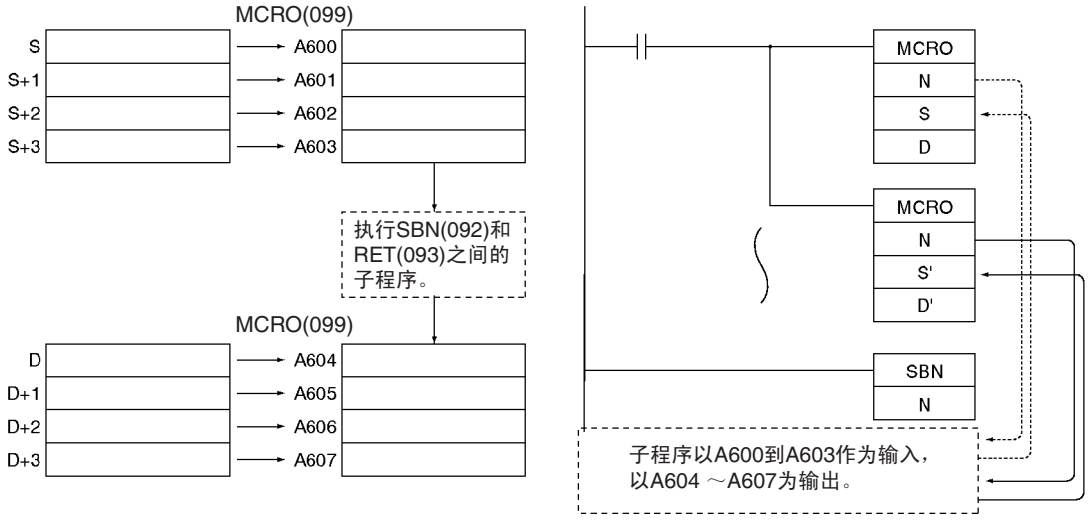
操作数规定

区域	N	S	D
CIO 区	---	CIO 0000 ～ CIO 6140	
工作区	---	W000 ～ W508	
保持位区	---	H000 ～ H508	
辅助位区	---	A000 ～ A444 A448 ～ A956	A448 ～ A956
定时器区	---	T0000 ～ T4092	
计数器区	---	C0000 ～ C4092	
DM 区	---	D00000 ～ D32764	
无区号 EM 区	---	E00000 ～ E32764	
有区号 EM 区	---	En_00000 ～ En_32764 (n = 0 ～ C)	
二进制间接 DM/EM 寻址	---	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 寻址	---	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	0 ～ 1023 （十进制）	---	
数据寄存器	---		
索引寄存器	---		
使用索引寄存器的间接寻址	---	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 to +2047, IR15 DR0 ～ DR15, IR0 ～ IR15, IR0+(++) ～ IR015+(++) ,–(– –)IR0 ～ , –(– –)IR15	

说明

MCRO(099) 对指定子程序号的子程序调用与 SBS(091) 类似，MCRO(099) 与 SBS(091) 的不同在于 MCRO(099) 可以使用 S 和 D 中的参数改变子程序中位和字地址，而不改变子程序的结构。

当执行 MCRO(099) 时， S ～ S+3 的内容复制到 A600 ～ A603（宏区输入），并执行指定子程序。当子程序执行完成时， A604 ～ A607（宏区输出）被复制到 D ～ D+3，且程序将继续执行 MCRO(099) 后的下一个指令。



MCRO(099) 可用于两个或两个以上的结构相同但输入和输出地址不同的子程序合并为一个子程序。在执行 MCRO(099) 时，指定的输入输出数据被传入指定子程序中。

标志

名称	标记	操作
错误标志	ER	嵌套超过 16 层时置 ON。 指定的子程序号不存在时置 ON。 子程序调用自身时置 ON。 调用正在运行的子程序时置 ON。 指定的子程序未在当前任务中定义过时置 ON。 其它情况置 OFF。

下表为辅助区中相应字。

名称	地址	操作
宏区输入字	A600 ～ A603	当执行 MCRO(099) 时， S 到 S+3 四字被复制到 A600 ～ A603，这些输入字被传送到子程序。
宏区输出字	A604 ～ A607	在 MCRO(099) 中定义的子程序被执行后，这些输出字中的输出数据被复制到 D 到 D+3 中。

注意

A600 ～ A603 的四个字中的输入数据 （字或位）和 A604 ～ A607 的四个字中的输出数据 （字或位）必须在由 MCRO(099) 调用的子程序中使用，它不能传递超过四个字的数据。

MCRO(099) 指令不能嵌套，但宏区输入和输出字 （A604 ～ A607）中的数据必须在调用其它子程序前进行存储，因为所有的 MCRO(099) 指令使用相同的 8 个字。

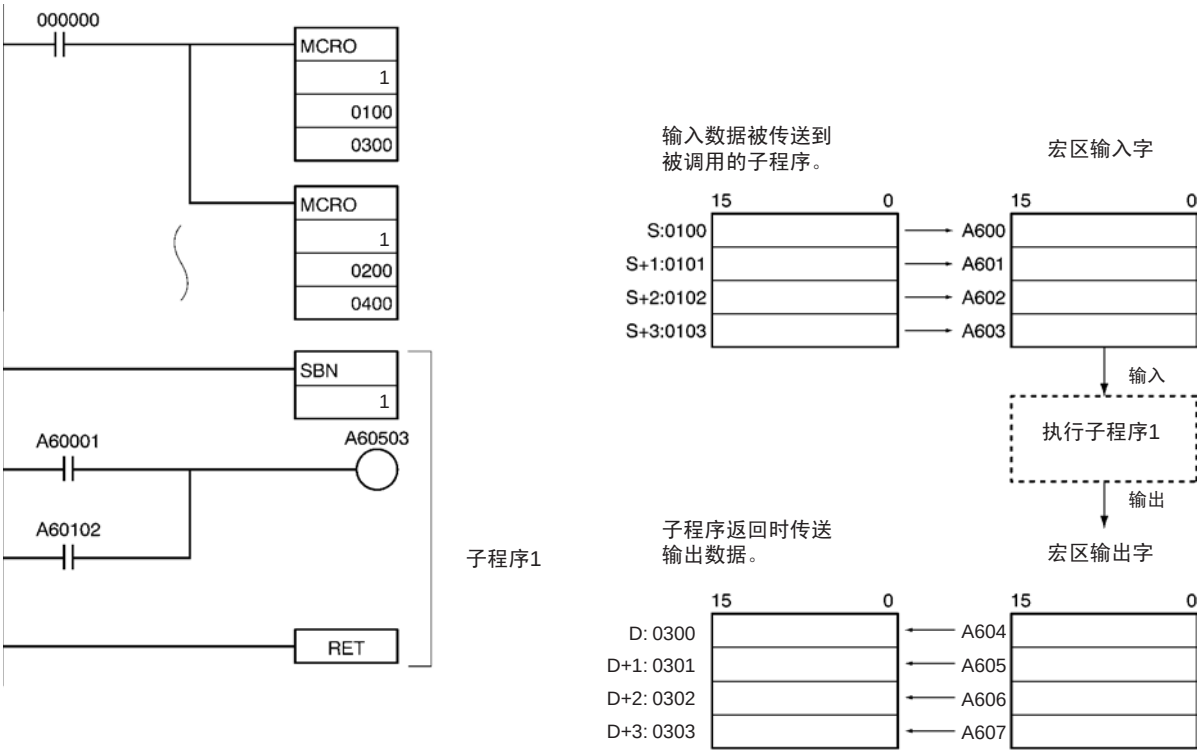
例

在下例中当 CIO 0000 为 ON 时，两个 MCRO(099) 指令传送不同的输入和输出数据到子程序 1。

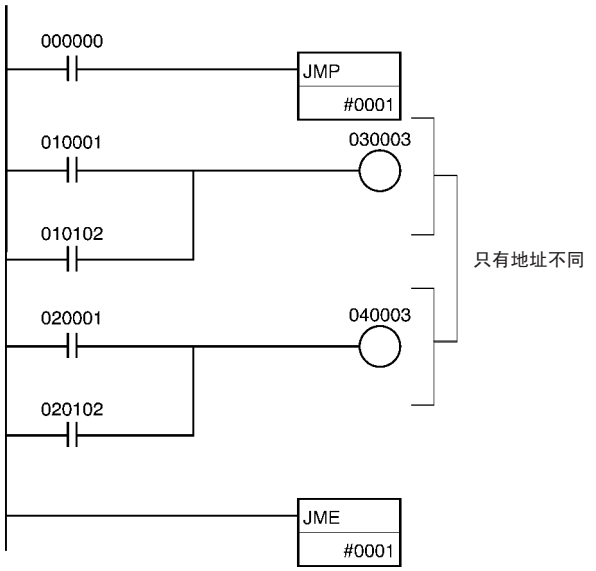
1,2,3...

1. 第一个 MCRO(099) 指令将 CIO 0100 到 CIO 0103 中输入数据传入，并执行子程序。子程序执行完毕后，输出数据输入存入 CIO 0300 ～ CIO 0303。

2. 第二个 MCRO(099) 指令将 CIO 0200 ～ CIO 0203 中输入数据传入，并执行子程序。子程序执行完毕后，输出数据输入存入 CIO 0400 ～ CIO 0403。



第二个 MCRO(099) 指令执行同样的操作，但 CIO 0200 ～ CIO 0203 中的输入数据被传送到 A600 ～ A603，A604 ～ A607 中的输出数据被传送到 CIO 0400 ～ CIO 0403。

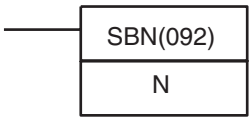


3-19-3 子程序入口：SBN(092)

用途

用指定子程序编号来指示子程序的开始。与 RET(093) 一起使用，定义一个子程序的范围。

梯形图符号



N:子程序编号

变化

变化	ON 条件时每次周期执行	SBN(092)
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
不允许	不允许	OK	OK

操作数

N：子程序号

指定子程序编号为 0 ～ 1023 之间的十进制数。

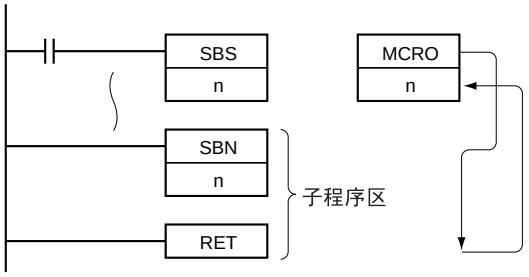
操作数规定

区域	N
CIO 区	---
工作区	---
保持位区	---
辅助位区	---
定时器区	---
计数器区	---
DM 区	---
无区号 EM 区	---
有区号 EM 区	---
二进制间接 DM/EM 寻址	---
BCD 间接 DM/EM 寻址	---
常数	0 ～ 1023 （十进制）
数据寄存器	---
索引寄存器	---
使用索引寄存器的间接寻址	---

说明

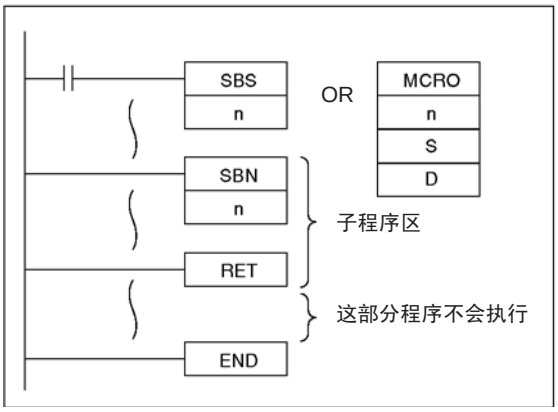
SBN(092) 指示子程序编号的子程序的开始。 RET(093) 指示子程序结束。

从第一个 SBN(092) 指令开始的程序区为子程序区。子程序只有被 SBS(091) 或 MCRO(099) 调用时才能执行。



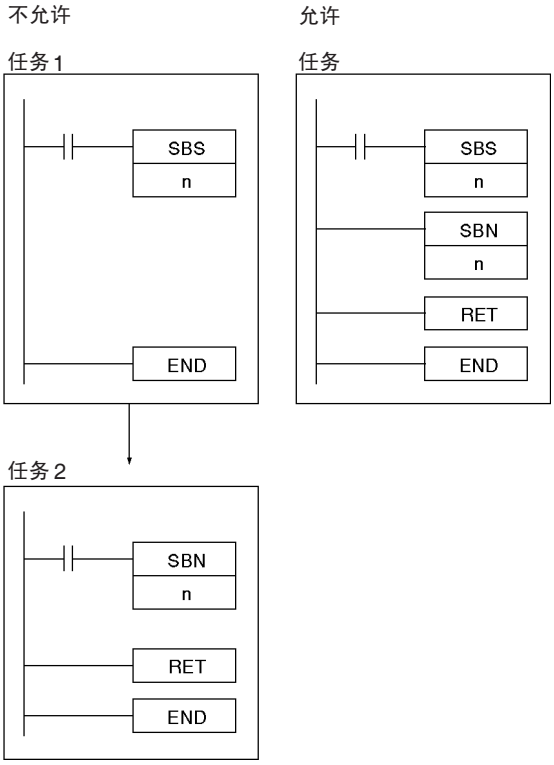
注意 当子程序未处于执行状态时，指令被当作 NOP(000) 处理。

对每个任务，将子程序放在主程序之后和程序的 END(001) 指令之前。如果部分主程序放在了子程序之后，这部分程序将被忽略。

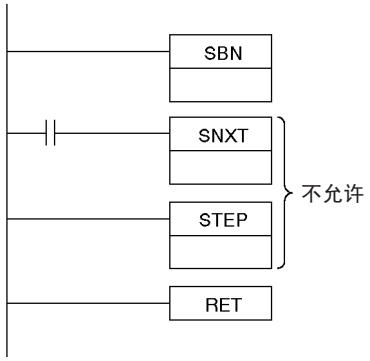


注 CX-Programmer和编程器对子程序编号N的输入方法是不同的。在CX-Programmer中输入 #0 ～ #1023，而在编程器中输入 0000 ～ 1023。

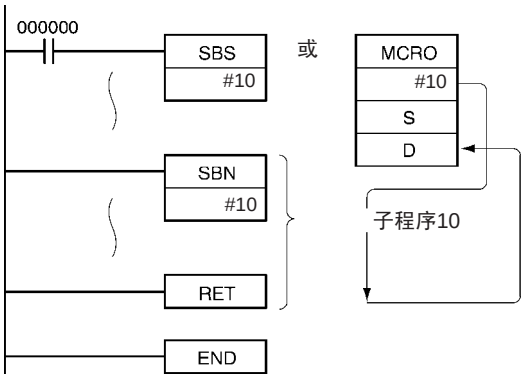
确认将每个子程序放在与对应 SBS(091) 或 MCRO(099) 指令相同的程序 （任务）中。一个任务中的子程序不能被另一个任务所调用。可以在中断任务中编写子程序。



步指令 （STEP(008) 和 SNXT(009)）不能用在子程序中。



例 在下例中当 CIO 000000 为 ON 时，执行子程序 10 并返回到调用该子程序的 SBS(091) 或 MCRO(099) 指令后的下一个指令。



3-19-4 子程序返回：RET(093)

用途

表示一个子程序的结束。与 SBN(092) 一起使用定义一个子程序区

梯形图符号

变化

变化	ON 条件时每次周期执行	RET(093)
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
不允许	不允许	OK	OK

说明

RET(093) 表示子程序的结束，而 SBN(092) 表示子程序开始。子程序操作的详细内容见 *3-19-3 子程序入口：SBN(092)*。

当程序执行到 RET(093) 时自动返回调用该子程序的 SBS(091) 或 MCRO(099) 指令的下一条指令。当子程序被 MCRO(099) 调用时，在程序返回前，A604 到 A607 中的输出数据被写入 D 到 D+3 中。

注意

当子程序未处于执行状态时，指令被作为 NOP(000) 处理。

例

RET(093) 操作的实例参见 *3-19-3 子程序入口：SBN(092)*。

3-19-5 全局子程序调用：GSBS(750)

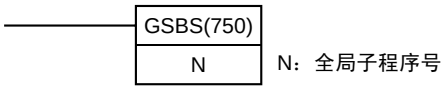
用途

调用指定编号的全局子程序并执行该程序。同一全局子程序能被 2 到多个任务调用。

这项指令仅适用于 CS1-H, CJ1H, CJ1M 和 CS1D CPU 单元。

GSBS(750)与全局子程序入口指令GSBN(751)、全局子程序返回指令GRET(752)一起使用。

梯形图符号



变化

变化	ON 条件时每次周期执行	GSBS(750)
	上升沿微分执行一次	@GSBS(750)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

N：全局子程序号
指定全局子程序号 （为 0 ～ 1023 间的十进制数）。

操作数规定

区域	N
CIO 区	---
工作区	---
保持位区	---
辅助位区	---
定时器区	---
计数器区	---
DM 区	---
无区号 EM 区	---
有区号 EM 区	---
二进制间接 DM/EM 寻址	---
BCD 间接 DM/EM 寻址	---
常数	0 ～ 1023 （十进制）
数据寄存器	---
索引寄存器	---
使用索引寄存器的间接寻址	---

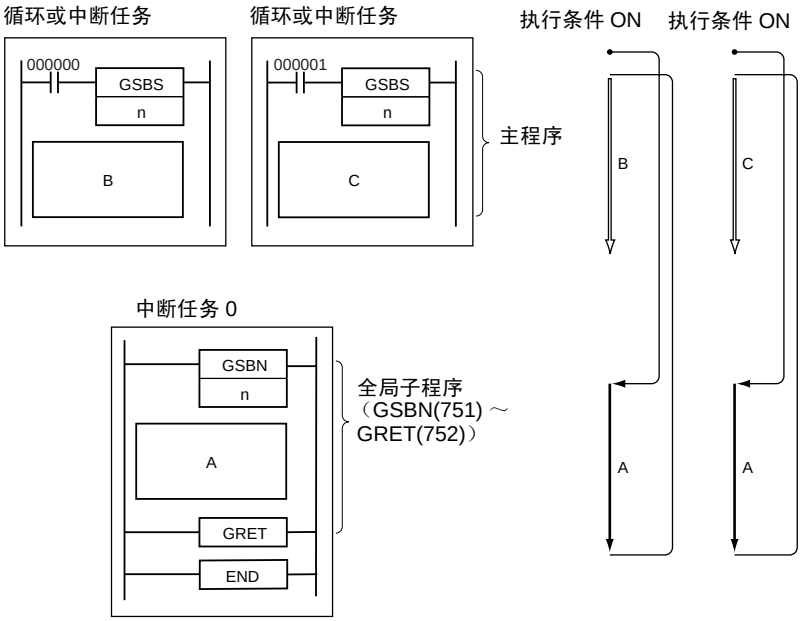
说明

GSBS(750) 调用指定编号的全局子程序，全局子程序指GSBN(751) 和 GRET(752) 之间的程序段，当全局子程序执行完成后，程序从 GSBS(750) 后的下一个指令继续执行。

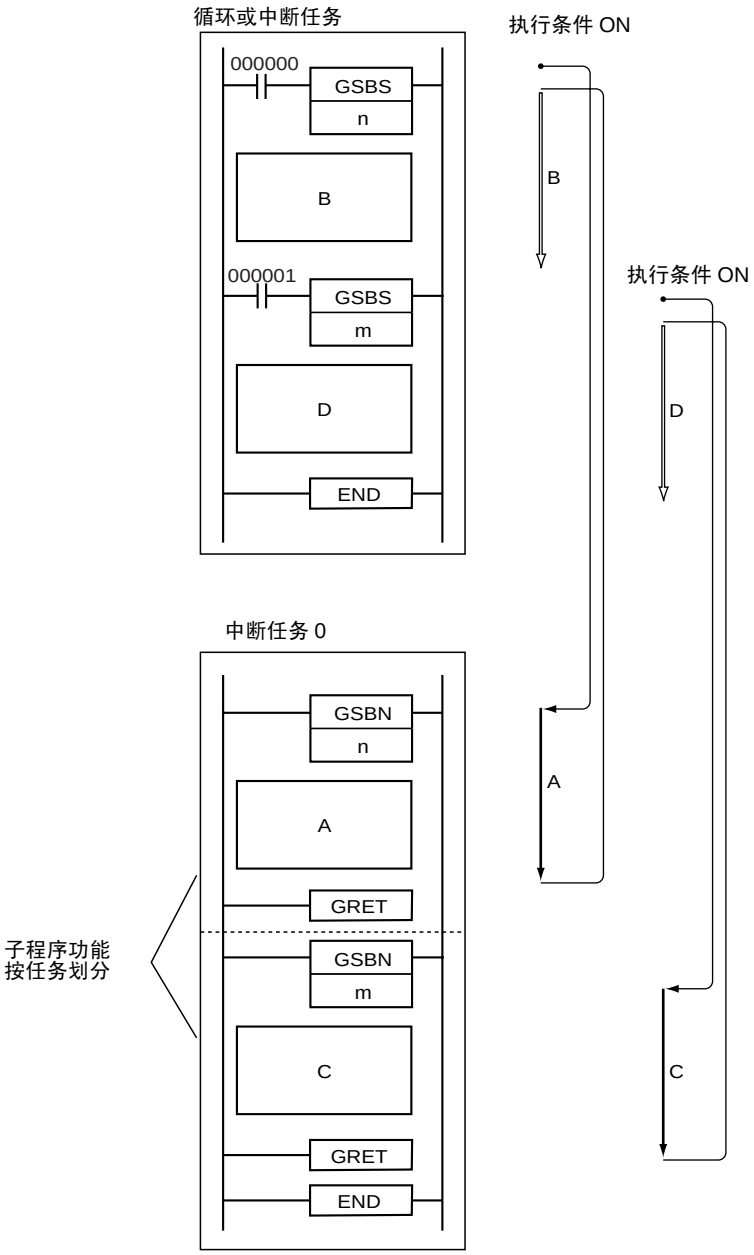
这项指令能写入带同一全局子程序号的多任务来调用来自不同任务的程序。通过把全局子程序变成对很多任务相同的标准全局子程序可以使程序模块化。

全局子程序区 （在 GSBS(751) 和 GRET(752) 之间）必须在中断任务 0 中定义。当它在另一个任务中定义时，会出现错误，并且当 GSBS(750) 指令执行时错误标志将变 ON。

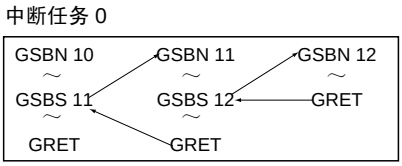
GSBS(750) 指令能被同时写入循环任务 （包括附加循环任务）和中断任务中。



在中断任务 0 中可以定义多个全局子程序区 (GSBN(751) ～ GRET(752))。



一个 SBS(091) 或 GSBS(750) 指令可以写入子程序区 (SBN(092) ～ RET(093)) 或全局子程序区 (GSBN(751) ～ GRET(752)) 来“嵌套”子程序。子程序嵌套可达 16 层。

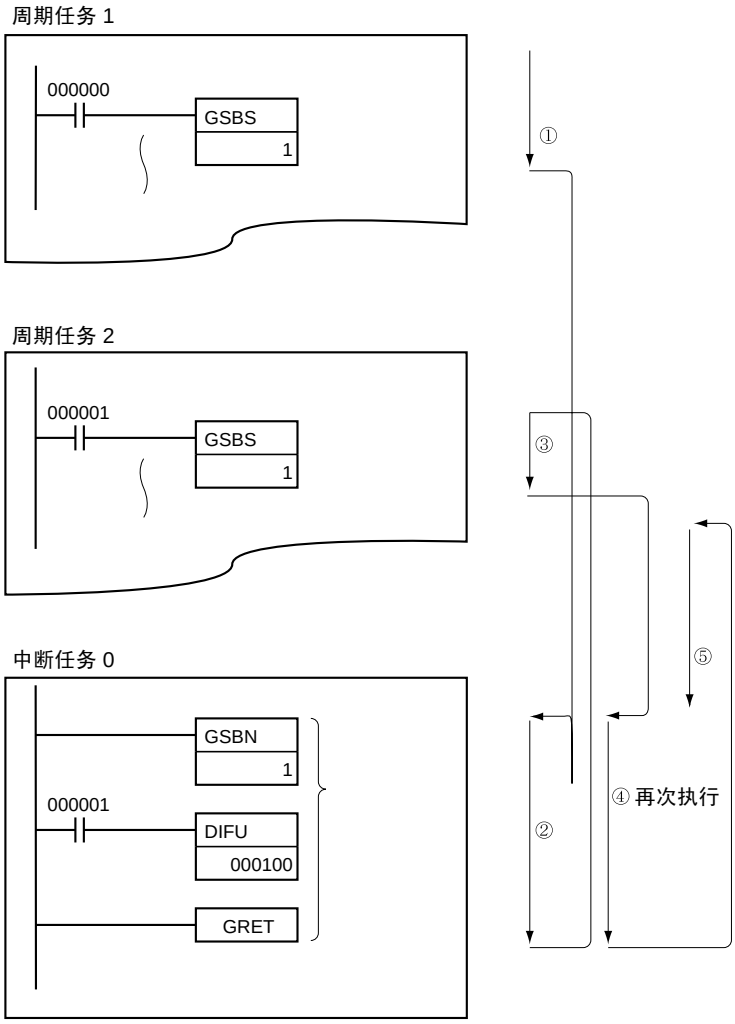


全局子程序和微分

在程序中使用微分指令 （DIFU(013)， DIFU(014) 或上升 / 下降沿微分指令）时必须注意下列事项。

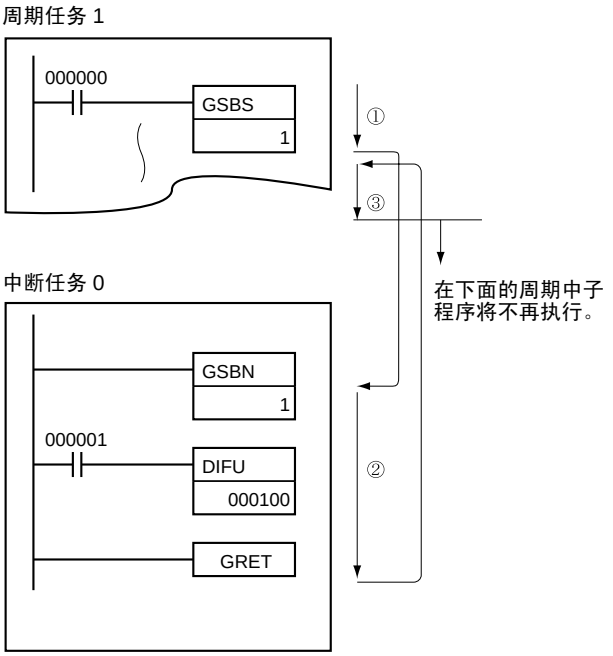
如果一个全局子程序在同一个周期中执行多次，那么子程序中的微分指令是不可预见的。在下例中 CIO 000000 为 ON 时执行子程序 0001，而当 CIO 000001 由 OFF 变时 ON，CIO 000100 由 DIFU(013) 置 ON。如果在同一周期中 CIO 000001

为 ON，则将再次执行子程序 0001，而此时 DIFU(013) 将不检验 CIO 00001 的上生沿状态，并将 CIO 000100 置 OFF。



相反，如果全局子程序中微分指令（DIFU(013) 或 DIFU(014) 执行后使输出变为 ON，但相同全局子程序没有被第 2 次调用，则微分输出保持为 ON。

在下例中若 CIO 000000 为 ON，则执行全局子程序 0001。当 CIO 000001 由 OFF 变为 ON 时， DIFU(013) 将输出 CIO00100 置为 ON。在下面的周期中若 CIO 000000 为 OFF，子程序 0001 将不再被执行，输出 CIO 000100 保持 ON。



标志

名称	标记	操作
错误标志	ER	嵌套超过 16 层时置 ON （计数包括常规子程序和全局子程序在内）。 指定的全局子程序号不存在时置 ON。 全局子程序调用自身时置 ON。 调用一个正在执行的全局子程序时置 ON。 指定的全局子程序未在当前任务中定义过时置 ON。 其它情况置 OFF。

说明

全局子程序入口指令 GSBN(751) 和相应的全局子程序返回指令 GRET(752) 必须在中断任务 0 中编程。如果全局子程序区不在中断任务 0 中编程，将会产生一个错误并且当 GSBS(750) 指令执行时错误标志将会变成 ON。

常规子程序调用指令 SBS(091) 不能调用全局子程序区 (GSBN(751) ～ GRET(752)) 内的子程序。

如果 GSBS(750) 在由 IL(002) 和 ILC(003) 联锁的程序段中，将作为 NOP(000) 处理。

同一全局子程序区 (GSBN(751) ～ GRET(752)) 可允许多次调用。

在下述情况下，即使执行了 GSBS(750) 指令，也不能调用全局子程序。

1,2,3...

1. 指定全局子程序未在当前任务中定义。
2. 子程序嵌套超过 16 层 （计数包括常规子程序和全局子程序在内）。
3. 全局子程序调用自身。
4. 指定的全局子程序正被执行。
5. 指定全局子程序未在中断任务 0 中定义。

例

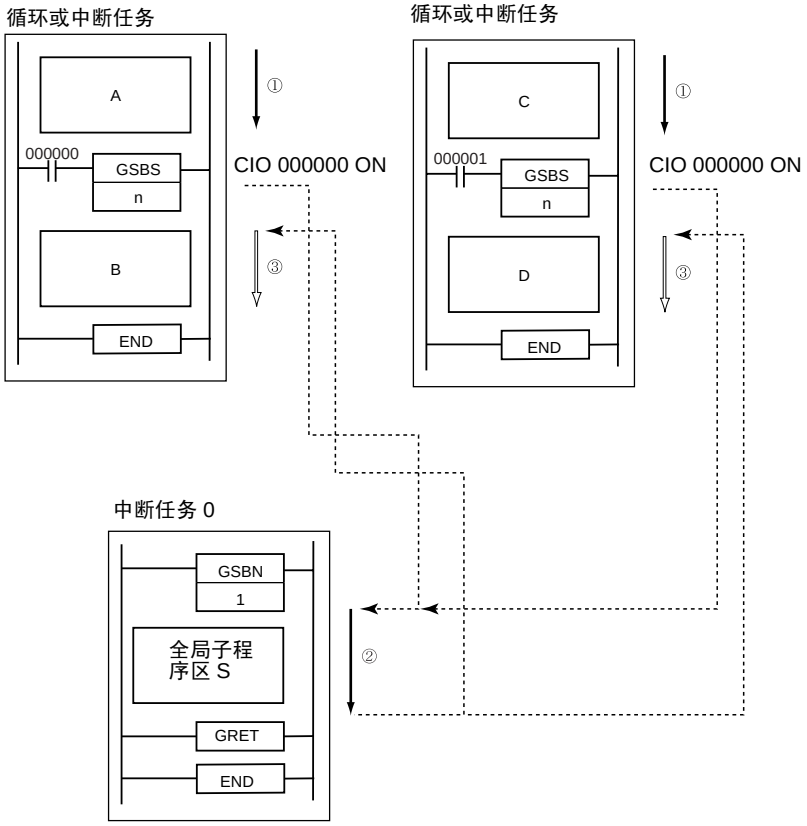
例 1

在下例中当CIO 000000为ON时，执行子程序1，并且程序执行返回到GSBS(750)后的下一条指令。

CIO 000000 的状态	执行程序的次序
ON	A → S → B
OFF	A → B

在下例中当CIO 000000为ON时，执行子程序1，并且程序执行返回到GSBS(750)后的下一个指令。

CIO 000000 的状态	执行程序的次序
ON	C → S → D
OFF	C → D

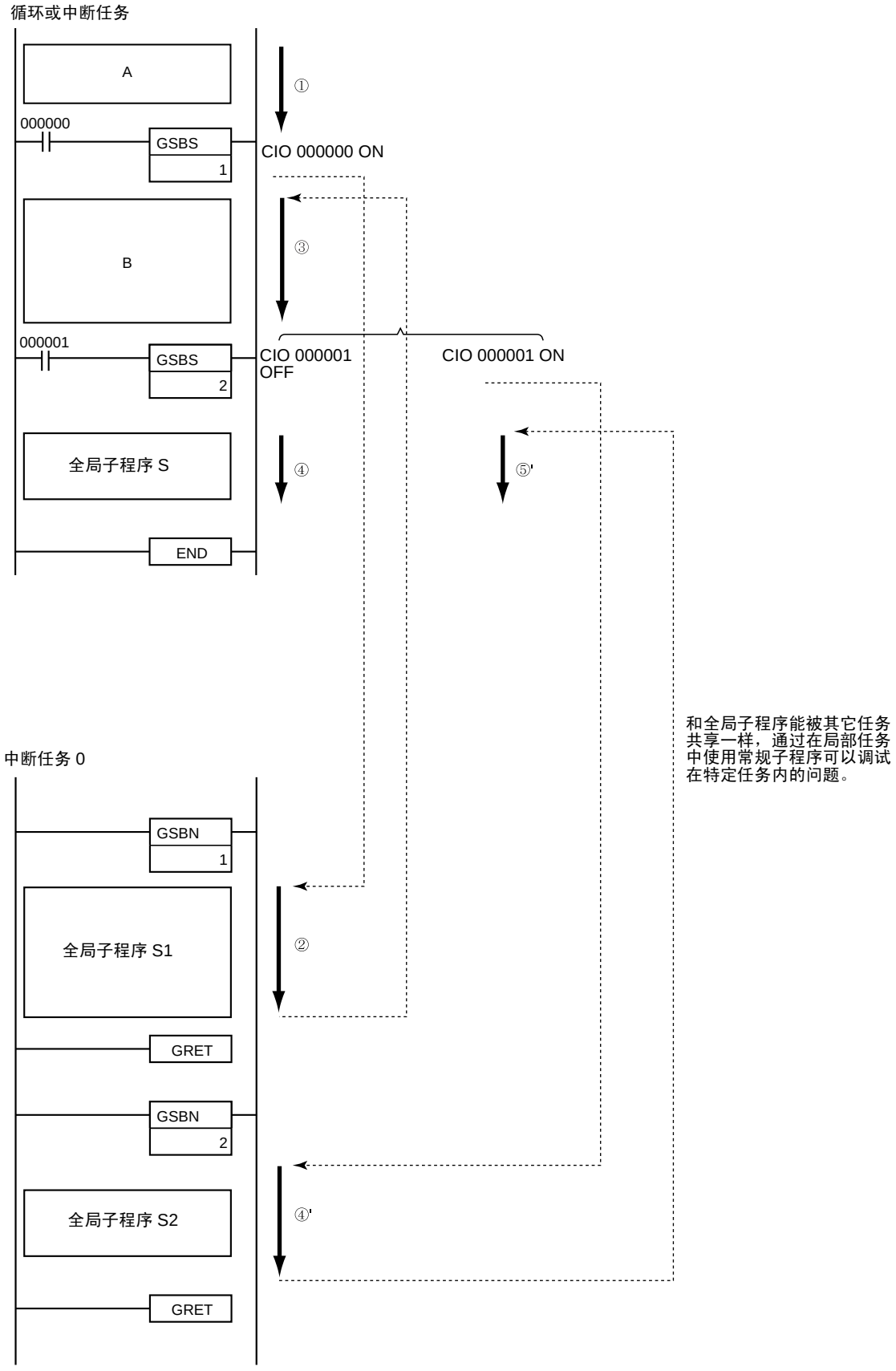


例 2

两个或更多全局子程序可以在中断任务 0 中编程。

在下例中，中断任务 0 可以被划分，并用作子程序功能的任务。

当 CIO 000000 为 ON 时，执行全局子程序 1。
当 CIO 000001 为 ON 时，执行全局子程序 2。



3-19-6 全局子程序入口：GSBN(751)

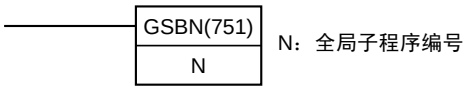
用途

用指定子程序编号来表示全局子程序的开始。与 GRET(752) 一起使用，定义一个全局子程序号的范围。

这项指令仅适用于 CS1-H,CJ1-H,CJ1M 和 CS1D CPU 单元。

GSBN(751)和全局子程序调用指令GSBS(750)、全局子程序返回指令GRET(752)一起使用。

梯形图符号



变化

变化	ON 条件时每次周期执行	GSBN(751)
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
不允许	不允许	---	OK

操作数

N：全局子程序编号

指定全局子程序编号为 0 ～ 1023 之间的十进制数。

操作数规定

区域	N
CIO 区	---
工作区	---
保持位区	---
辅助位区	---
定时器区	---
计数器区	---
DM 区	---
无区号 EM 区	---
有区号 EM 区	---
二进制间接 DM/EM 寻址	---
BCD 间接 DM/EM 寻址	---
常数	0 ～ 1023 （十进制）
数据寄存器	---
索引寄存器	---
使用索引寄存器的间接寻址	---

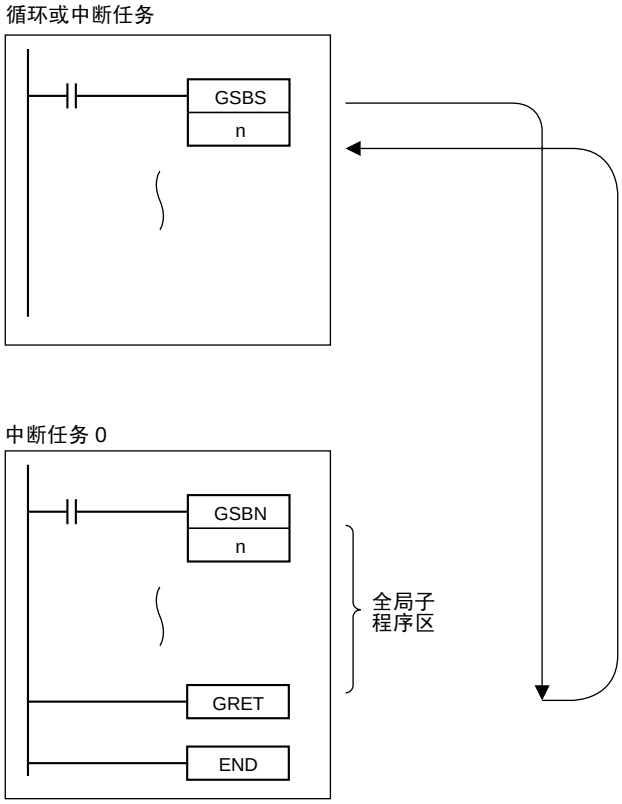
说明

GSBN(751) 指示全局子程序编号的全局子程序的开始。 GRET(752) 指示全局子程序结束。

从第一个 GSBN(751) 指令开始的程序区为全局子程序区。全局子程序只有被 GSBS(750) 调用时才能执行。

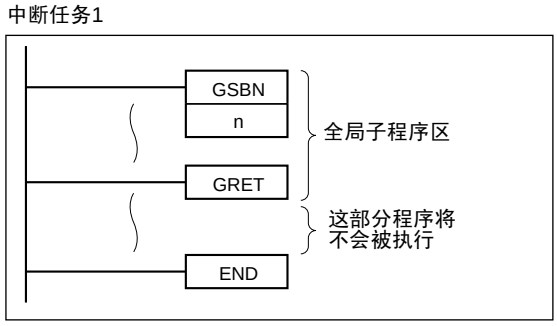
全局子程序区 （在 GSBN(751) ～ GRET(752) 之间）必须在中断任务 0 中定义。当它在另一个任务中定义时，会出现错误，并且当 GSBS(750) 指令执行时错误标志会变 ON。

GSBS(750) 指令能被同时写入循环任务（包括附加循环任务）和中断任务中。

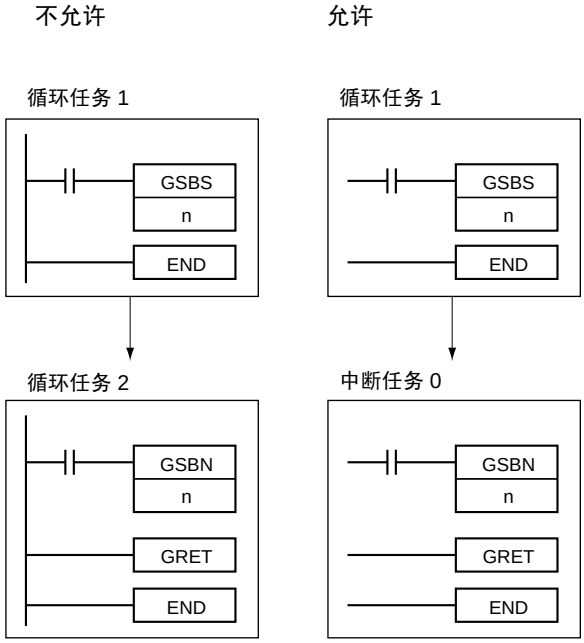


注意

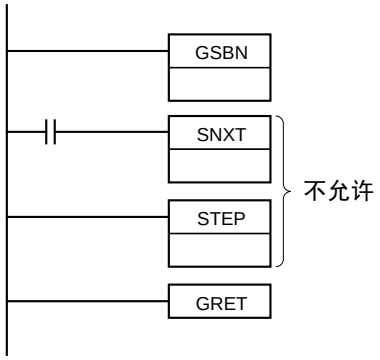
- 当子程序未处于执行状态时，这项指令被当作 NOP(000) 处理。
- 将中断任务 0 中全局子程序区 （GSBN(751) ～ GRET(752)）程序放在 END(001) 指令之前。当有两个或更多全局子程序被使用时，在主程序结尾处把在中断任务 0 中程序组合在一起。如果部分主程序放在了全局子程序之后，这部分程序将被忽略。



- CX-Programmer 和编程器对全局子程序编号 N 的输入方法是不同的。在 CX-Programmer 中输入 #0 ～ #1023，而在编程器中输入 0000 ～ 1023。
- 总是把全局子程序放在中断任务0中。如果一个全局子程序被调用并且子程序不在中断任务 0 中将会出现一个错误。

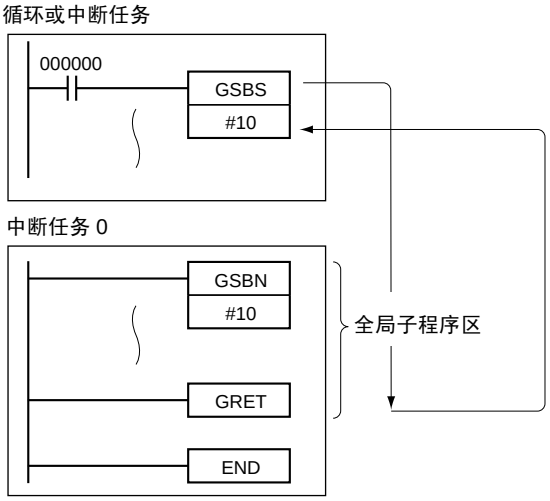


• 步指令 （STEP(008) 和 SNXT(009)）不能在全局子程序中使用。



例

在下例中当 CIO 000000 为 ON 时，执行全局子程序 10 并且程序执行返回到调用子程序的 GSBS(750) 指令后下一条指令。



3-19-7 全局子程序返回：GRET(752)

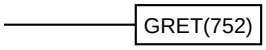
用途

表示子程序的结束。与 GSBN(751) 一起使用定义子程序区。

此指令仅由 CS1-H, CJ1-H, CJ1M 和 CS1D CPU 单元支持。

GRET(752) 指令与 GSBS(750) 和 GSBN(751) 组合相同，GSBS(750) 为全局子程序调用， GSBN(751) 为全局子程序进入。

梯形图符号



变化

变化	ON 条件时每次周期执行	GRET(752)
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
不允许	不允许	不允许	OK

说明

GRET(752) 表示全局子程序的结束，而 GSBN(751) 表示全局子程序开始。全局子程序操作的详细内容见 *3-19-6 子程序入口：GSBN(751)*。

当程序执行到 GRET(752) 时自动返回调用该子程序的 GSBS(750) 指令的下一条指令。

注意

当子程序未处于执行状态时，指令被作为 NOP(000) 处理。

例

GRET(752) 操作的实例参见 *3-19-6 全局子程序入口：GSBN(752)*。

3-20 中断控制指令

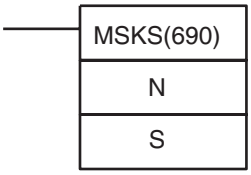
3-20-1 中断屏蔽：MSKS(690)

用途

当 PC 进入运行 (RUN) 方式时， I/O 中断任务和定时中断任务均被屏蔽 （禁止）。 MSKS(690) 可用于对 I/O 中断屏蔽或取消屏蔽和为定时中断设定时间间隔。

CS1D CPU 单元不支持 MSKS(690) 指令。

梯形图符号



变化

变化	ON 条件时每次周期执行	MSKS(690)
	上升沿微分执行一次	@MSKS(690)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

CS1W-INT01/CJ1W-INT01

指定 I/O 中断处理和屏蔽处理。

操作数	内容
N	指定中断输入单元的单元号 0：单元号 0 1：单元号 1
S	中断屏蔽 设置为 16 进制数 0000~FFFF （16 位 / 单元） 0：允许中断 1：屏蔽中断

注 1. CS1W-INT01 和 C200HS-INT01 不能被同时使用。

2. 当中断屏蔽被清除时所有被检测到的中断输入将被清除。

3. CJ1W-INT01 中断输入单元不能和 CJ1 CPU 单元一起使用。同样，I/O 中断任务也不能被执行。

中断输入单元的单元号和中断任务号之间的关系如下表所示。

单元号	中断任务号	
0	100 ～ 115	S 的第 00 ～ 15 位对应于输入中断任务。
1	116 ～ 131	

指定 I /O 中断处理上升 / 下降沿标志

操作数	内容
N	指定中断输入单元的单元号 2：单元号 0 3：单元号 1
S	指定中断输入信号的上升或下降沿 设置为 16 进制数 0000~FFFF （16 位 / 单元） 特殊位表示如下： 0：上升沿 1：下降沿

中断输入单元的单元号和中断任务号之间的关系如下表所示。

单元号	中断任务号	
2	100 ～ 115	S 的第 00 ～ 15 位对应于输入中断任务。
3	116 ～ 131	

注 当上升 / 下降沿指定标志改变时所有检测到的中断输入将被清除。

C200HS-INT01

指定 I/O 中断处理和屏蔽处理。

操作数	内容
N	指定中断输入单元的单元号 0：单元号 0 1：单元号 1 2：单元号 2 3：单元号 3
S	中断屏蔽 设置为 16 进制数 0000~FFFF （8 位 / 单元） 特殊位表示如下： 0：允许中断 1：屏蔽中断

注 1. CS1W-INT01 和 C200HS-INT01 不能被同时使用。

2. 当中断屏蔽被清除时，所有被检测到的中断输入将被清除。

中断输入单元的单元号和中断任务号之间的关系如下表所示。

单元号	中断任务	
0	100 ～ 107	S 的第 00 ～ 07 位对应于输入中断任务。
1	108 ～ 115	
2	116 ～ 123	
3	124 ～ 131	

注 当上升 / 下降沿指定标志改变时，所有检测到的中断输入将被清除。

CJ1M CPU 单元内置中断输入

指定 I/O 中断处理和屏蔽处理。

单元号	内容
N	指定中断输入号 6：中断输入 0 7：中断输入 1 8：中断输入 2 9：中断输入 3
S	中断屏蔽 0000：中断允许 （直接模式） 0001：中断屏蔽 （直接模式） 0002：递减计数器启动和中断允许 （计数器模式） 0003：递增计数器启动和中断允许 （计数器模式）

注 当中断屏蔽被清除时，所有被检测到的输入中断将清除。

中断输入单元的单元号和中断任务号之间的关系如下表所示。

中断输入号	中断任务号	
中断输入 0	140	CIO 296000
中断输入 1	141	CIO 296001
中断输入 2	142	CIO 296002
中断输入 3	143	CIO 296003

指定 I/O 中断处理上升 / 下降沿标志

操作数	内容
N	指定中断输入号 10：中断输入 0 11：中断输入 1 12：中断输入 2 13：中断输入 3
S	指定中断输入信号的上升或下降沿。 0000：上升沿 0001：下降沿

中断输入单元的单元号和中断任务号之间的关系如下表所示。

中断输入号	中断任务号	
中断输入 0	140	CIO 296000
中断输入 1	141	CIO 296001
中断输入 2	142	CIO 296002
中断输入 3	143	CIO 296003

注 当上升 / 下降沿指定标志改变时，所有检测到的中断输入将被清除。

指定定时中断

操作数	内容 s
N	指定定时中断号 4：中断任务 2 5：中断任务 3
S	0000：禁止定时中断 0001 ～ 270F：定时中断时间间隔 (1 ～ 9999) 注 定时中断时间间隔单位可以在 PC 设置的中断环境中设为 10ms 或 1ms，对于 CJ1M CPU 单元，还可以设为 0.1ms，其设定范围为 0005 ～ 270F(5 ～ 9999)

重新设定 / 启动定时中断 （仅 CJM1 允许 ）

操作数	内容		
N	指定定时中断号 14：定时中断 0 （中断任务 2） 15：定时中断 1 （中断任务 3）		
S	禁止定时中断		0000
	设置定时中断时间和启动定时中断	10~99,990ms 或 1~9,999ms （中断时间间隔单位为 10ms 或 1ms）	0001 ～ 270F
		0.5~999.9ms （中断时间间隔单位为 0.1ms）	0005 ～ 270F （设置 0001 ～ 0004Hex 不能被使用；当其使用时，将出现指令错误）

操作数规定

区域	N	S
CIO 区	---	CIO 0000 ～ CIO 6143
工作区	---	W000 ～ W511
保持位区	---	H000 ～ H511
辅助位区	---	A000 ～ A447 A448 ～ A959
定时器区	---	T0000 ～ T4095
计数器区	---	C0000 ～ C4095
DM 区	---	D00000 ～ D32767
无区号 EM 区	---	E00000 ～ E32767
有区号 EM 区	---	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 寻址	---	@ D00000 ～ @ 32767 @ E00000 ～ @ 32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 寻址	---	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	只有指定值	
数据寄存器	---	DR0 ～ DR15

区域	N	S
索引寄存器	---	
使用索引寄存器 的间接寻址	---	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –) IR0 ～ , –(– –) IR15

说明

MSKS(690) 控制 I/O 中断和定时中断。 N 值用于识别中断。

单元	N	含义
CS1W-INT01 或 CJ1W-IN01	0 ～ 1	N 对应于中断输入任务。 S 的第 0 ～ 7 位对应于指定单元中中断输入号 0 ～ 7。 MSKS(690) 当中断输入对应位是 ON 时屏蔽 （禁止），当中断输入对应位是 OFF 时取消屏蔽 （允许）。
C200HS-INT01	0 ～ 3	
CJ1M CPU 单元内置中断输入	6 ～ 9	

单元	N	含义
CS1W-INT01 或 CJ1W-IN01	2 或 3	N 对应于中断输入任务。 S 把上升或下降沿指定为触发器 （缺省值为下降沿）。
CJ1M CPU 单元内置中断输入	10 ～ 13	

注 1. MSKS(690) 可用于使一个特定 I/O 中断任务仅在特定循环中允许，并使该任务在其它循环中禁止。

2. 单元号按照安装的次序从左到右赋给中断输入单元。

N= 4 或 5

值 4 和 5 对应于定时中断 2 和 3。

当 N= 4 或 5 时， S 的内容禁止中断任务 (S=0000)，或者用于指定时间间隔来设置中断任务。定时中断间隔的单位可在 PC 设置中设定为 10ms， 1ms，和 0.1ms。

N=14 或 15 （仅 CJ1M CPU 单元）

当 N=14 或 15 时，由于定时中断任务由 N 指定，在 S 中设置指定的定时中断时间并且重新设置定时器间隔。重新设置启动首次中断时间将保留。

注 1. 定时中断的时间单位在 PC 中设置。

2. 确保时间间隔长于执行定时中断任务的时间。

3. 对于定时中断， MSKS(690) 仅用作设置定时中断时间间隔，而不能设置首次定时中断的时间。为了准确控制首次中断和中断时间间隔，设置首次定时中断时间的程序 CLI(691) 应刚好放在程序 MSKS(690) 之前。如果 MSKS(690) 用于重新启动 CJ1M CPU 单元的首次定时中断，即使不使用 CLI(691)，首次定时中断的时间仍然很准确。

A440 包含了中断任务的最大处理时间，A441 的最右字节包含了最长处理时间任务的中断任务号。

标志

名称	标记	操作
错误标志	ER	若 N 不在指定范围 0~5 时置位 ON。（对于 CJ1M 内置中断输入为 0 ～ 15）。 当 N 是 0 ～ 3 时 （当使用一个 C200HS-INT01 和指定 I/O 中断处理时）， S 不在指定范围 0000 ～ 00FF 时置 ON。 若 S 不在指定范围 0000 ～ 00FF 时置 ON （当使用 CJ1M 内置中断输入和指定 I/O 中断处理时）。 当 N 是 4 或 5 （对于定时时间间隔单位为 0.1ms 的 CJ1M 内置中断输入， 0005 ～ 270F）时， S 不在指定范围 0000 ～ 270F 时置 ON。 若指令在一个中断任务中执行时置 ON。 其它情况置 OFF。
等于标志	=	OFF
负标志	N	OFF

下表为辅助区中的相应标志。

名称	地址	操作
中断任务错误标志	A40213	在下述情况置 ON： 1）在 I/O 刷新 C200H 特殊 I/O 单元或远程 I/O 从站机架时执行了大于 10ms 的中断任务 （仅 CS 系列）。 2）在未使特殊 I/O 循环刷新无效时，在一个中断任务中执行了 IORF(097)。
导致中断任务出错标志	A42615	指明是产生了中断任务错误 1 还是 2。
中断任务错误任务号	A42600 ～ A42611	对错误 1： 指明中断任务号。 对错误 2： 在多重 I/O 刷新产生时指明特殊 I/O 单元的单元号。

注意

支持标准 CS/CJ 系列中断输入单元和 C200H 中输入单元的中断输入。不支持插入板和特殊的 I/O 单元的中断输入。

在 CPU 的的站机架上安装中断输入单元。如果使用 CJ1-H CPU 单元，把这个单元安装在 0 ～ 4 槽，如果用的是 CJ1M CPU 单元，则把这个单元安装在 0 ～ 2 槽。除非中断输入单元安装到这些插槽中的一个，PC 是不可能启动 I/O 中断任务的。

字地址按照中断输入单元从左到右的顺序指派给输入单元。

中断具有不同的优先级别。电源 OFF 中断具有最高优先权，其次是定时中断，最后是 I/O 中断。低编号的 I/O 中断级别优先于高编号的 I/O 中断。

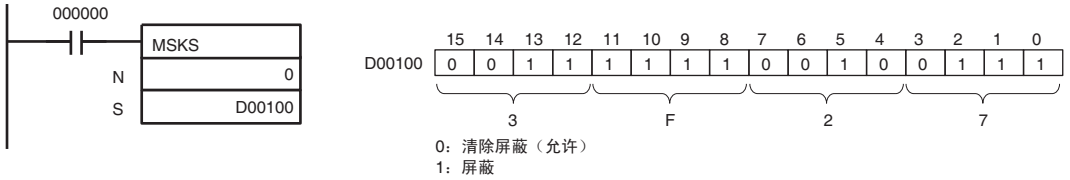
如果与一个 C200H 特殊 I/O 单元或 SYSMAC BUS 远程 I/O 从站机架相连，应确保中断任务所需时间不大于 10ms。如果在 I/O 刷新特殊 I/O 单元或从站机架时执行了大于 10ms 的中断任务，将产生一个非致命错误，中断任务错误标志 (A40213) 置 ON。

当在一个中断任务中，执行 IORF(097) 来刷新特殊 I/O 单元中的 I/O 时，必须在 PC 设置中禁止特殊 I/O 单元的刷新。如果没有禁止特殊 I/O 单元的循环刷新，在循环刷新时执行 IORF(097) 将导致一个非致命的重复刷新错误，中断任务标志 (A40213) 置 ON。

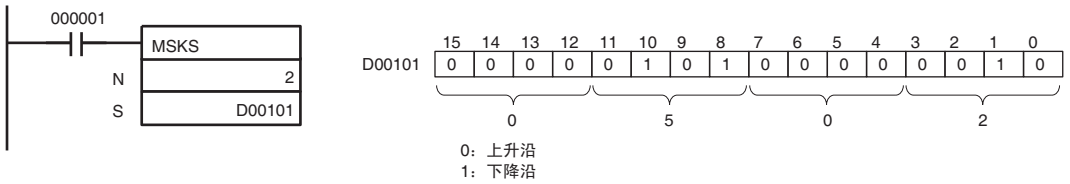
例

CS1W-INT01/CJ1W-INT01 实例

下例中 CIO 000000 置 ON 时，MSKS(690) 将中断输入单元 0 的中断清除屏蔽（允许）。

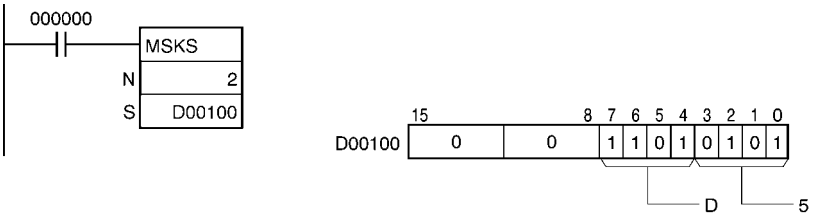


下例中 CIO 000001 置 ON 时，MSKS(690) 给中断输入单元 0 设置上升 / 下降沿指定标志。



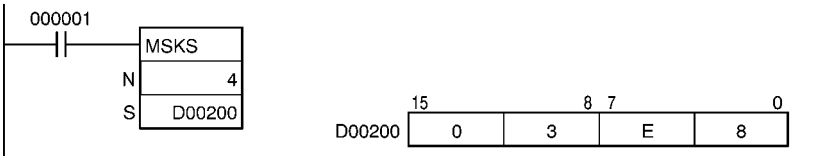
C200HS-INT01 实例

下例中 CIO 000000 置 ON 时， MSKS(690) 将中断输入单元 2 中的 1、 3 和 5 中断输入清除屏蔽 （允许）。



定时中断实例

下例中 CIO 000001 置 ON 时，MSKS(690) 将给定时中断 2 设置一个 10s 时间间隔 （此例中，在 PC 设置时设定定时时间间隔为 10ms）。



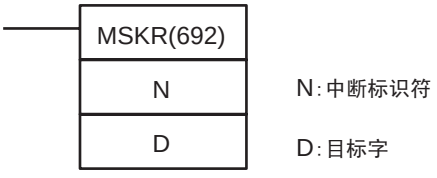
3-20-2 读中断屏蔽：MSKR(692)

用途

读取 MSKS(690) 中设定的当前中断处理设置。

CS1D CPU 单元不支持 MSKR(692) 指令。

梯形图符号



变化

变化	ON 条件时每次周期执行	MSKR(692)
	上升沿微分执行一次	@MSKR(692)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

CS1W-INT01/CJ1W-INT01

读屏蔽

操作数	内容
N	指定中断输入单元的单元号 0：单元号 0 1：单元号 1
D	中断屏蔽状态。 16 进制数 0000~FFFF （16 位 / 单元） 特殊位表示如下： 0：中断允许 1：中断屏蔽

中断输入单元的单元号和中断任务号之间的关系如下表所示。

单元号	中断任务号	
0	100 ～ 115	S 的 00 ～ 15 位对应于输入中断任务。
1	116 ～ 131	

读上升 / 下降沿指定标志

操作数	内容
N	指定中断输入单元的单元号 2：单元号 0 3：单元号 1
D	指定中断输入信号的上升或下降沿。 16 位进制 0000~FFFF （16 位 / 单元） 特殊位表示如下： 0：上升沿 1：下降沿

注 1. CS1W-INT01 和 C200HS-INT01 不能被同时使用。

2. CJ1W-INT01 中断输入单元不能与 CJ1 CPU 单元一起使用，且 I/O 中断任务不能执行。

中断输入单元的单元号和中断任务号之间的关系如下表所示。

单元号	中断任务号	
2	100 ～ 115	S 的第 00 ～ 15 位对应于输入中断任务。
3	116 ～ 131	

C200HS-INT01

读屏蔽

操作数	内容
N	指定中断输入单元的单元号 0：单元号 0 1：单元号 1 2：单元号 2 3：单元号 3
D	中断屏蔽状态 0000~00FF （8 位 / 单元） 特殊位指示如下： 0：中断允许 1：中断屏蔽

注 CS1W-INT01 和 C200HS-INT01 不能被同时使用。

中断输入单元的单元号和中断任务号之间的关系如下表所示。

单元号	中断任务号	
0	100 ～ 107	S 的 00 ～ 07 位对应于输入中断任务。
1	108 ～ 115	
2	116 ～ 123	
3	124 ～ 131	

CJ1M CPU 单元内置式中断输入

读屏蔽

操作数	内容
N	指定中断输入号 6：中断输入 0 7：中断输入 1 8：中断输入 2 9：中断输入 3
D	中断屏蔽 0000 Hex：中断允许 （直接模式） 0001 Hex：中断屏蔽 （直接模式） 0002 Hex：递减计数器启动和中断允许 （计数器模式） 0003 Hex：递增计数器启动和中断允许 （计数器模式）

中断输入单元单元号和中断任务号之间的关系如下表所示。

中断输入号	中断任务号	
中断输入 0	140	CIO 296000
中断输入 1	141	CIO 296001
中断输入 2	142	CIO 296002
中断输入 3	143	CIO 296003

读 I/O 中断处理上升 / 下降沿指定标志

操作数	内容
N	指定中断输入号 10：中断输入 0 11：中断输入 1 12：中断输入 2 13：中断输入 3
D	指定中断输入信号的上升或下降沿。 0000：上升沿 0001：下降沿

中断输入号和中断任务号之间的关系如下表所示。

中断输入号	中断任务号	
中断输入 0	140	CIO 296000
中断输入 1	141	CIO 296001
中断输入 2	142	CIO 296002
中断输入 3	143	CIO 296003

读定时中断时间间隔

操作数	内容
N	指定定时中断号 4：定时中断 0 （中断任务 2） 5：定时中断 1 （中断任务 3）
D	0000：屏蔽定时中断 0001 ～ 270F：定时中断时间间隔 （1 ～ 9999ms） **注** 定时中断间隔单位可以在 PC 设置中断环境中设为 10ms 或 1ms。对于 CJ1M CPU 单元可设 0.1ms 为单位，在此情况下时间将为 0005 ～ 270F(5 ～ 9999)ms。

读定时中断当前值

操作数	内容	
N	指定定时中断号 14：定时中断 0 （中断任务 2） 15：定时中断 1 （中断任务 3）	
D	计时从本定时中断处理启动或前一定时中断历史值开始。 10ms 为单位：0 ～ 99,990 1ms 为单位：1 ～ 9,999 0.1ms 为单位：0 ～ 999.9 （仅 CJ1M）	0000 ～ 270F **注** 即使定时中断当前是停止，它停止前消耗的时间仍然可以读出。如果定时中断根本没开始执行，将读得 0000 Hex。

操作数规定

区域	N	D
CIO 区	---	CIO 0000 ～ CIO 6143
工作区	---	W000 ～ W511
保持位区	---	H000 ～ H511
辅助位区	---	A448 ～ A959
定时器区	---	T0000 ～ T4095
计数器区	---	C0000 ～ C4095

区域	N	D
DM 区	---	D00000 ～ D32767
无区号 EM 区	---	E00000 ～ E32767
有区号 EM 区	---	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 寻址	---	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 寻址	---	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	只有指定值	---
数据寄存器	---	DR0 ～ DR15
索引寄存器	---	
使用索引寄存器 的间接寻址	---	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –) IR0 ～ , –(– –) IR15

说明

MSKR(692) 读取 MSKS(690) 中的中断设置

N= 0 或 1 （对于 C200HS-INT01，为 0 ～ 3）
值 0 和 1(0 ～ 3) 对应于中断输入单元 0 和 1(0 ～ 3)。D 的第 0 ～ 7 位对应于指定单元中的中断输入号 0 ～ 7。若有一位是 ON，对应中断输入屏蔽 （禁止）。若有一位是 OFF，对应中断输入清除屏蔽 （允许）。

N= 2 或 3 （仅 CS1W-INT01/CJ1W-INT0/CJ1M 内置式中断输入）
值 2 和 3 对应于中断输入单元 0 和 1。由 N 指定的中断输入单元的输入中断的上升 / 下降沿标志输出给 D。

N= 4 或 5
值 4 和 5 对应于定时中断输入单元 2 和 3。

当 N 是 4 或 5 时， D 的内容表示了已设置的时间间隔。 0000 的设置表示中断已禁止。定时中断间隔单位可以在 PC 中设置 （00：10ms， 01：1.0ms）。因此，时间间隔的范围可在 10ms ～ 99. 99 或 1ms ～ 9.999s 之间。

N=14 或 15
当 N 是 14 或 15 时，把由定时中断任务 N 指定的定时中断定时器的 （当前） PV 值存入 D。

标志

名称	标记	操作
错误标志	ER	N 不在指定区域 0 ～ 5 （对于 CJ1M， 0 ～ 15）间置 ON。 其它情况置 OFF。

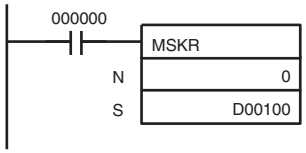
注意

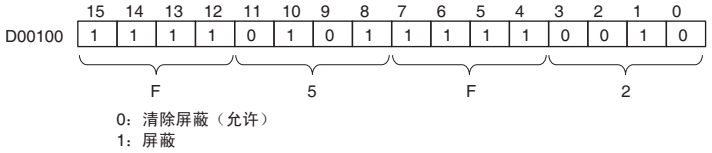
MSKR(692) 可在主程序或中断任务中执行。

例

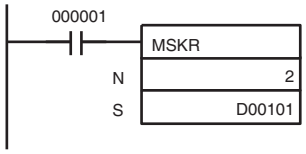
CS1W-INT01/CJ1M-INT01 实例

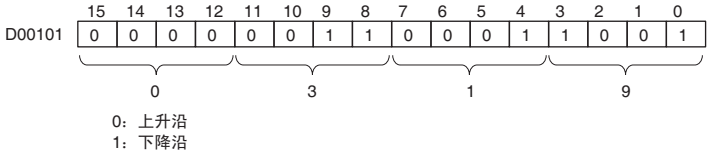
下例中 CIO 000000 置 ON 时，MSKR(692) 读中断输入单元 2 的当前屏蔽状态并把它存入 D00100。





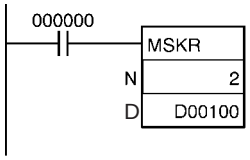
下例中 CIO 000001 置 ON 时， MSKS(690) 读中断输入单元 0 的上升 / 下降沿标志并把它存入 D00100。

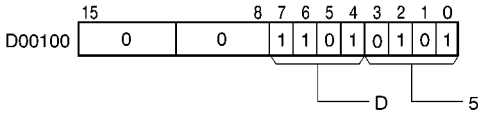




C200H-INT01 实例

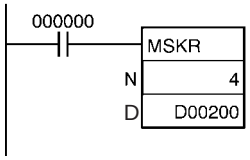
下例中 CIO 000000 置 ON 时，MSKR(692) 读中断输入单元 2 的当前屏蔽状态，这时中断 1,3 和 5 允许中断。

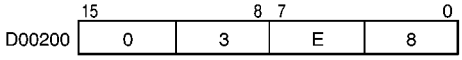




定时中断实例

下例中 CIO 000000 置 ON 时， MSKS(690) 读定时中断 2 的设置。此时如果 PC 设置中设置定时时间间隔单位为一个 10ms，时间间隔设置为 1,000(3E8H)，等效于 10s。





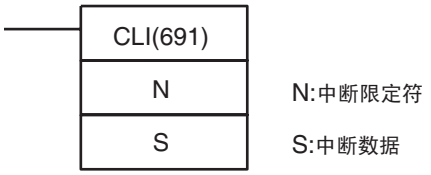
3-20-3 清除中断：CLI(691)

用途

清除或保持 I/O 中断的中断输入记录，或者设定定时中断的首次定时中断的时间。

CS1D CPU 单元不支持 CLI(691) 指令。

梯形图符号



变化

变化	ON 条件时每次周期执行	CLI(691)
	上升沿微分执行一次	@CLI(691)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

CS1W-INT01/CJ1W-INT01 的 I/O 中断

操作数	内容
N	指定中断输入单元的单元号 0：单元号 0 1：单元号 1
S	清除中断屏蔽分类（16 位 / 单元） 0000~FFFF 特殊位含义： 0：保持已记录的中断输入 1：清除已记录的中断输入

注 1. CS1W-INT01 和 C200HS-INT01 不能被同时使用。

2. CJ1W-INT01 中断输入单元不能与 CJ1 CPU 单元一起使用，并且 I/O 中断任务不能执行。

中断输入单元单元号和中断任务号之间的关系如下表所示。

单元号	中断任务号	
0	100 ～ 115	S 的 00 ～ 15 位对应于输入中断任务。
1	116 ～ 131	

C200H-INT01 的 I/O 中断

操作数	内容
N	指定中断输入单元的单元号 0：单元号 0 1：单元号 1 2：单元号 2 3：单元号 3
S	清除中断屏蔽分类（16 位 / 单元） 0000 ～ FFFF 特殊位含义： 0：保持已记录的中断输入 1：清除已记录的中断输入

注 CS1W-INT01 和 C200HS-INT01 不能被同时使用。

中断输入单元单元号和中断任务号之间的关系如下表所示。

单元号	中断任务号	
0	100 ～ 107	S 的 00 ～ 07 位对应于输入中断任务。
1	108 ～ 115	
2	116 ～ 123	
3	124 ～ 131	

CJ1M CPU 单元内置式中断输入

操作数	内容
N	指定中断输入号 6：中断输入 0 7：中断输入 1 8：中断输入 2 9：中断输入 3
S	清除中断屏蔽规定。 0000：保持已记录的中断输入 0001：清除已记录的中断输入

中断输入单元单元号和中断任务号之间的关系如下表所示。

中断输入号	中断任务号	
中断输入 0	140	CIO 296000
中断输入 1	141	CIO 296001
中断输入 2	142	CIO 296002
中断输入 3	143	CIO 296003

清除高速计数器中断 （仅 CJ1M）

操作数	内容
N	指定高速计数器输入 10：高速计数器输入 0 11：高速计数器输入 1
S	清除中断屏蔽规定。 0000：保持已记录的中断输入 0001：清除已记录的中断输入

首次定时中断的时间设定

操作数	内容
N	指定定时中断号 4：定时中断 0 （中断任务 2） 5：定时中断 1 （中断任务 3）
S	0000~270F：首次定时中断时间 （0 ～ 9999） 注 定时中断间隔单位可以在 PC 设置的中断设置中设定为 10ms 或 1ms，对于 CJ1M CPU 单元还可以设为 0.1ms。

操作数规定

区域	N	S
CIO 区	---	CIO 0000 ～ CIO 6143
工作区	---	W000 ～ W511
保持位区	---	H000 ～ H511
辅助位区	---	A000 ～ A959

区域	N	S
定时器区	---	T0000 ～ T4095
计数器区	---	C0000 ～ C4095
DM 区	---	D00000 ～ D32767
无区号 EM 区	---	E00000 ～ E32767
有区号 EM 区	---	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 寻址	---	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 寻址	---	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---	DR0 ～ DR15
数据寄存器	只有指定值	
索引寄存器	---	
使用索引寄存器的间接寻址	---	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –) IR0 ～ , –(– –) IR15

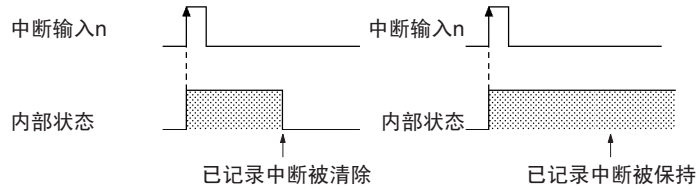
说明

根据 N 的值，CLI(691) 清除指定记录的 I/O 中断或者在首次定时中断发生之前设置时间。对于 CJ1M，它还可以用来清除高速计数器中断

N=0 ～ 3（对 C200HS-INT01 为 0 ～ 3，对 CJ1M CPU 单元内置式中断输入为 6 ～ 9）

值 0 和 1(0 ～ 3) 对应于中断输入单元 0 和 1(0 ～ 3)。

S 的位 0 到 7 对应于指定单元中的中断输入号 0 ～ 7。若有一位是 ON，CLI(691) 清除已记录的中断输入。若有一位是 OFF，CLI(691) 保持对应已记录的中断输入。



若正在执行一个 I/O 中断任务，且一个有不同输入号的中断输入被接受，那个中断号记录于内部。记录的 I/O 中断稍后按次序（从低号到高号）执行。CLI(691) 可用于在它们执行之前清除这些记录中断。

注 1. MSKS(690) 可在特定的循环中允许特定 I/O 中断任务，并且在其它循环中禁止这个任务。

2. 单元号按照中断输入单元的安装从左到右按序指派给输入单元。

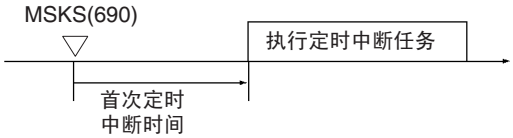
N=4 或 5

值 4 和 5 对应于定时中断 2 和 3。

N为4或5时，S的内容指定执行MSKS(690)后的首次定时中断任务的时间间隔。

时间间隔可设置为 0000 到 270F(0 ～ 9,999)。定时中断时间间隔单位在 PC 设置中设置 (00：10ms， 01：1.0ms)，因此有效的时间间隔范围可以从 10ms ～ 99.99s 或 1ms ～ 9.999s。

注 设置首次定时中断的时间间隔为 10ms 或更长。



N = 10 或 11 （仅 CJ1M）

值 10 和 11 对应高速计数器中断，它们可用于清除或保留中断（目标值比较或区域比较）。

标志

名称	标记	操作
错误标志	ER	若 N 不在指定范围 0~5 时置位 ON （对于 CJ1M 为 0， 1 或 4 ～ 11）。 当 N 是 0 ～ 3， S 不在指定范围 0000 ～ 00FF 间时置 ON （仅对于 I/O 中断和 C200H-INT01）。 若 S 不为 0000 或 0001 时置 ON （仅对于高速计数器中断和 CJ1M 内置式中断输入）。 对于定时中断，当 S 不在指定范围 0000 ～ 270F 时置 ON。 其它情况置 OFF。

注意

对于可记录的中断输入数没有限制，可以记录所有 I/O 中断输入，但如果新的中断已经被记录，则被忽略。另外，记录的中断直到中断任务结束才清除。因此如果一个新中断输入单元的中断输入在它的中断任务在执行时接收到，则新中断被忽略。

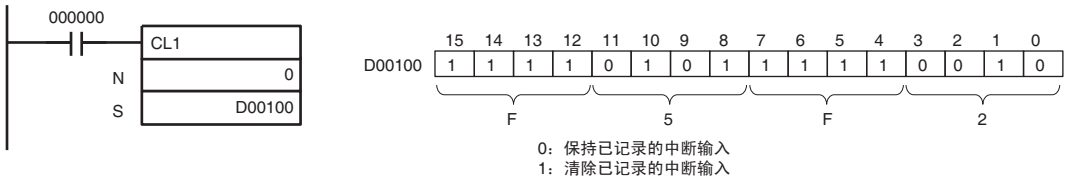
支持标准 CS/CJ 系列中断输入单元和 C200H 中输入单元的中断输入。不支持插入板和特殊的 I/O 单元的中断输入。

中断具有不同的优先级别。电源 OFF 中断具有最高优先权，其次是定时中断，最后是 I/O 中断。低编号的 I/O 中断优先于高编号的 I/O 中断。

例

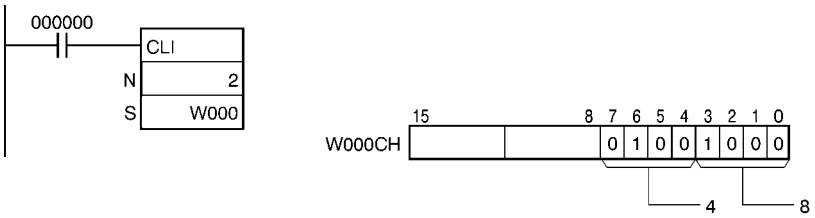
CS1W-INT01/CJ1W-INT01 实例

下例中 CIO 000000 置 ON 时，CLI(691) 清除中断输入单元 0 中的中断 1,4,8,10 和 12 ～ 15 的中断记录。



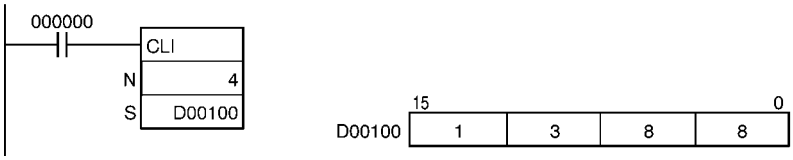
C200HS-INT01 实例

下例中 CIO 000000 置 ON 时， CLI(691) 清除中断输入单元 2 中的中断 3 和 6 的中断记录。



实例：设置首次定时中断时间

下例中 CIO 000001 置 ON 时，CLI(691) 将定时中断 2 的第一次时间设定为 50 秒 （在本例中，定时时间间隔的单位在 PC 设置中设定为 10ms）。



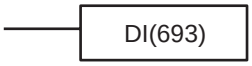
3-20-4 禁止中断：DI(693)

用途

禁止除电源 OFF 中断以外的中断任务执行。

当使用的是 CS1-H， CJ1-H 和 CJ1M CPU 单元，且电源 OFF 中断任务被禁止，则也可同时禁止电源 OFF 中断处理。

梯形图符号



变化

变化	ON 条件时每次周期执行	DI(693)
	上升沿微分执行一次	@DI(693)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	不允许

注意

主程序中执行 DI(693) 将暂时禁止除电源 OFF 中断之外的所有中断任务 （I/O 中断，定时中断和外部中断）。

所有的中断任务被禁止，直至执行 EI(694) 指令，中断被重新允许。

CS1-H， CJ1-H 和 CJ1M CPU 单元与电源 OFF 中断

当使用 CS1-H， CJ1-H 和 CJ1M CPU 单元，A503 （电源 OFF 中断的禁止设置）被设置为 A5A5 Hex 时，电源 OFF 中断处理被同时禁止。在程序指令按次序执行直至最后一项任务中的 EI(694) 或 END(001) 指令之后，即使电源中断在 DI(693) 执行后被检测到，CPU 单元将会重新启动。

如果电源 OFF 中断任务被激活， CPU 单元将在电源 OFF 中断执行后重新设置。参照 *CS/CJ 系列编程手册详细了解关于电源 OFF 中断任务的信息。*

标志

名称	标记	操作
错误标志	ER	若 DI(693) 由一个中断任务中执行时为 ON。 其它情况置 OFF。

相关标志和字

下表中的字是在辅助寄存区中的相应标志。

名称	地址	操作
电源 OFF 的中断禁止设置	A530	A5A5： 允许电源 OFF 中断的禁止设置。电源 OFF 处理 （包括执行电源 OFF 中断任务）在 DI(693) 和 EI(694) 之间被屏蔽，因此，到 EI(694) 的指令都被执行。

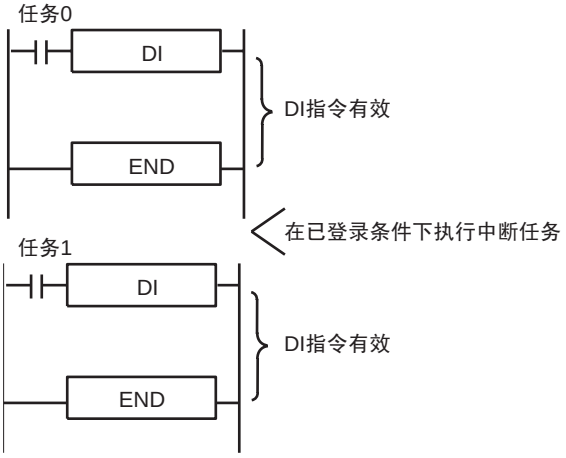
注意

所有中断任务将保持被禁止，直到 EI(694) 指令执行。

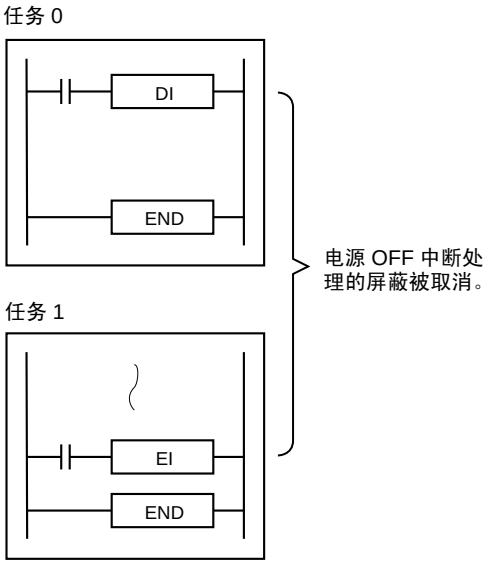
DI(693) 不能在一个中断任务中执行。

DI(693) 不能为多个任务执行。为使多个循环执行任务禁止，应在每个循环任务中插入 DI(693)，如下例所示，任何一个循环执行任务正在执行时产生的中断，会在该循环执行任务执行完毕后执行，除非它们已被 CLI(691) 指令禁止。

当使用 DI(693) 在一个 CS1-H，CJ1-H，CJ1M CPU 单元中禁止电源 OFF 中断处理时，可以禁止循环任务处理。（在所有启动任务完成后禁止条件被释放）

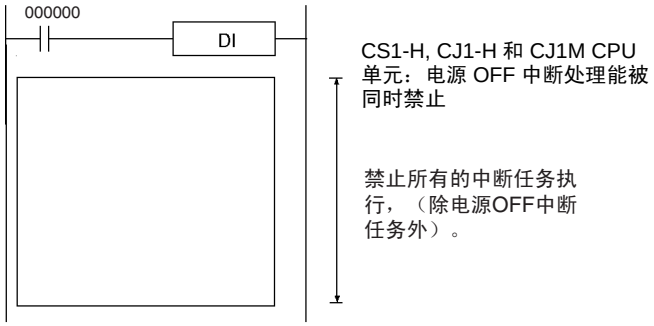


当使用 CS1-H，CJ1-H，CJ1M CPU 单元时，电源 OFF 中断任务被禁止且 A530 被设置为 A5A5，如果执行在 DI(693) 和 EI(694) 之间指令期间电源中断被检测到， CPU 单元将在 EI(694) 执行后被重新设置。



例

在下例中 CIO 00000 为 ON 时，DI(693) 禁止除电源 OFF 中断任务外所有的中断任务。



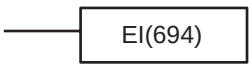
3-20-5 允许中断：EI(694)

用途

使 DI(693) 禁止的所有中断任务允许执行。

当 CS1-H,CJ1-H 和 CJ1M CPU 单元被使用，且电源 OFF 中断任务被禁止，EI(694) 将同时释放被禁止的电源 OFF 中断处理。

梯形图符号



变化

变化	ON 条件时每次周期执行	EI(694)
	上升沿微分执行一次	不支持
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	不允许

说明

主程序中执行EI(694)使已由DI(693)禁止所有中断任务暂时允许执行。DI(693)将使除电源 OFF 中断之外的所有中断任务 （I/O 中断，定时中断和外部中断）被禁止。

CS1-H， CJ1-H， CJ1M CPU 单元与电源 OFF 中断

当 CS1-H，CJ1-H，CJ1M CPU 单元被使用，且电源 OFF 中断处理被 DI(693) 禁止，EI(694) 将会释放对电源 OFF 中断处理的保持。在 DI(693) 已被执行后，即使电源中断被检测到， CPU 单元也不会被重新设置。 CPU 单元将在所有在 EI(694) 和 DI(693) 之间的指令执行完之后才会重新设置。参照 *3-20-4 禁止中断：DI(693)* 详细了解关于使用 DI(693) 禁止电源 OFF 中断处理。

标志

名称	标记	操作
错误标志	ER	EI(694) 由中断任务中执行时置 ON。 其它情况置 OFF。

相关标志和字

下表中的字是在辅助寄存区中的相应标志。

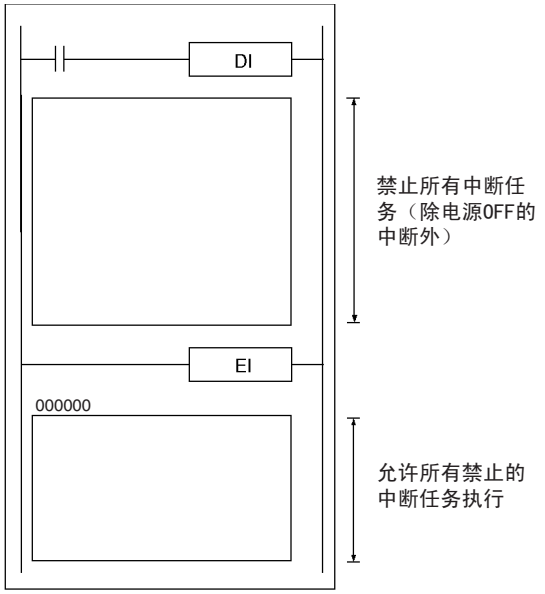
名称	地址	内容
电源 OFF 的禁止设置	A530	A5A5： 允许电源 OFF 中断的禁止设置。电源 OFF 处理 （包括执行电源 OFF 中断任务）在 DI(693) 和 EI(694) 之间被屏蔽，因此，所有到 EI(694) 的指令被执行。 其它值：禁止电源 OFF 处理屏蔽

注意

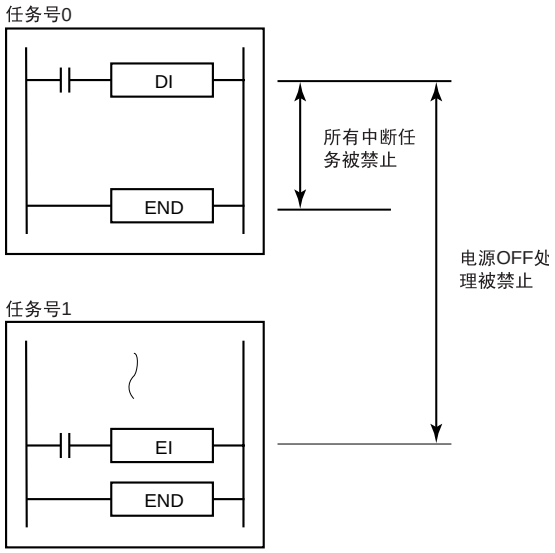
EI(694) 不需要执行条件，它一直在 ON 执行条件下执行，EI(694) 使被 DI(693) 禁止的中断任务允许执行。它不能使没有用 MSKS(690) 清除屏蔽的 I/O 中断清除屏蔽或没有用 MSKS(690) 设置的定时中断设置定时中断。EI(694) 不能在中断任务中执行。

例

在下例中， EI(694) 使所有被 DI(693) 禁止的中断任务允许执行。



注 对于 CS1-H、 CJ1-H、 CJ1M CPU 单元，当电源 OFF 的中断任务被禁止时，电源 OFF 处理在此同时也被允许。



3-20-6 中断控制综述

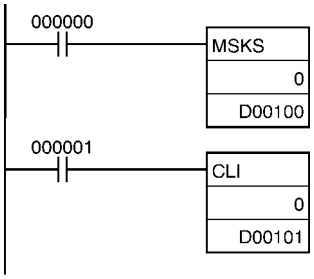
中断控制指令控制或读取 I/O 中断和定时中断的设置（DI(693) 和 EI(694) 控制外部中断操作同 I/O 中断和定时中断一样）。

作用于个别中断的指令有一个标记中断源的操作数 N，数 0 ～ 3 表示中断输入单元 0 ～ 3，数 4 和 5 表示定时中断 2 和 3。

I/O 中断处理 (N=0~3)

I/O 中断是由中断输入单元中的输入信号产生的。PC 可连接不超过四个的中断输入单元，单元号 0 ～ 3 按照单元在 PC 中的位置从左到右分配给它们。

下例说明了 MSKS(690) 和 CLI(691) 用于控制 I/O 中断时的操作。



D00100	0 0 F 0
D00101	0 0 F 2

MSKS(690) 的操作

在 PC 第一次通电时， I/O 中断任务和定时中断任务均被屏蔽（禁止）。MSKS(690) 可用于对 I/O 中断屏蔽或取消屏蔽，为定时中断设定时间间隔。

在本例中，MSKS(690) 使用 D00100 中的内容对中断输入单元 0 的中断输入 0 ～ 3 清除屏蔽，屏蔽中断输入 4 ～ 7。

	F				0			
单元0的中断输入	7	6	5	4	3	2	1	0
中断屏蔽设置	1	1	1	1	0	0	0	0

1=屏蔽（禁止） 0=清除屏蔽（允许）

当中断输入 3 由 OFF 变为 ON 时，主程序的执行被中断，并执行 I/O 中断任务 3（中断任务 103）， I/O 中断任务 3 执行完毕后，主程序在中断点恢复执行。

I/O 中断任务优先级别

若同时接收到两个或两个以上的中断输入，将按照它们的中断号由低到高(100～131) 顺序执行中断。

单元	中断任务
中断输入单元 0	输入 0 ～ 7 相应于 I/O 中断任务 100 ～ 107。
中断输入单元 1	输入 0 ～ 7 相应于 I/O 中断任务 108 ～ 115。
中断输入单元 2	输入 0 ～ 7 相应于 I/O 中断任务 116 ～ 123。
中断输入单元 3	输入 0 ～ 7 相应于 I/O 中断任务 124 ～ 131。

如果在一个中断任务正在执行时接收到多个中断任务输入，当前中断任务完成后，被记录过的中断按优先级次序将被执行。

如果是一个时间间隔中断发生，时间间隔中断任务将由优先级更字的 I/O 中断任务接管。

CLI(691) 的操作

如果在一个 I/O 中断任务正在执行时接收到一个不同的中断输入，输入的中断号在内部被记录，直至当前任务和其它较高优先级别的任务执行完毕。

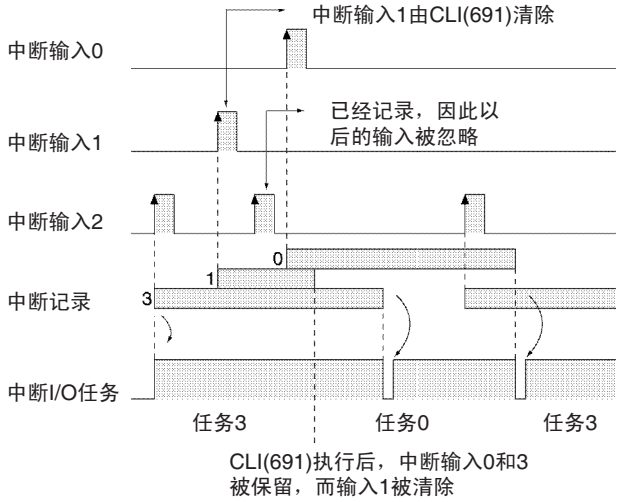
CLI(691) 可用于被记录的中断在其执行前将它们清除，但不能清除一个正在执行的中断任务。

在本例中，CLI(691) 用 D00101 中的内容，将中断输入单元 0 中除了输入 0，2 和 3 之外的所有记录的中断输入清除。

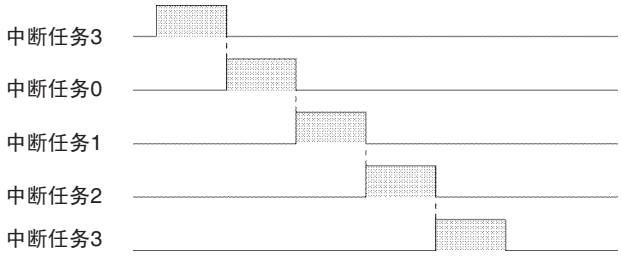
	F				2			
单元0的中断输入	7	6	5	4	3	2	1	0
中断清除/保留设置	1	1	1	1	0	0	1	0

1=清除记录的输入 0=保留记录的输入

在中断任务 3 执行完毕后，按优先级顺序执行已记录的中断。因为中断输入 0 记录了一个输入。所以将在任务 3 执行完毕后执行 I/O 中断任务 0 （中断任务 100）。因为 CLI(691) 未保留中断输入 1，所以这一中断输入记录被清除。



如中断输入 0 ～ 3 全变为 ON，且未执行 CLI(691)，所有输入将被记录，并在中断任务 3 执行完毕后顺序执行之。（中断任务按优先权从最低中断号到最高中断号顺序执行）。



注 1. 不一定要使用 CLI(691)。

2. 未执行 CLI(691) 时，正在执行一个中断任务时接收到的所有的 I/O 中断输入都被记录。如果一个记录的输入再次接收到，后面的输入将被忽略。

3. 若记录了两个或两个以上的 I/O 中断输入，它们将根据优先级别顺序执行，与记录输入的接收次序无关。

定时中断处理 （N=4 或 5）

定时中断按照 MSKS(690) 中设定的间隔重复进行，而不依赖于 PC 周期时间。编号 N 的 4 和 5 分别对应于定时中断号 2 和 3。

定时中断处理

定时中断处理的主要性能如下所示：

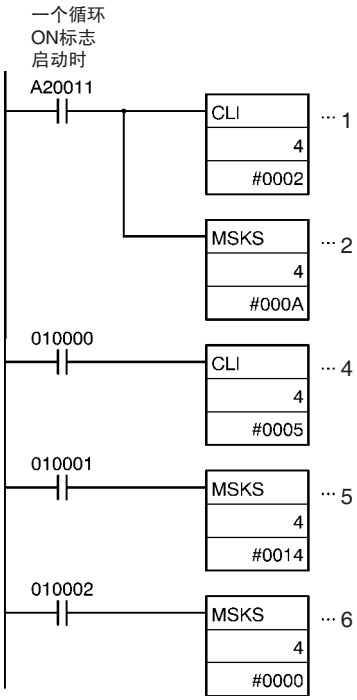
1,2,3...

1. 在 PC 第一次通电时定时中断被屏蔽 （禁止）。
2. 用 CLI(691) 为首次定时中断设定时间 （在执行 MSKS(690) 之后）。若未用 CLI(691) 设定，首次定时中断的时间是不能预料的。

3. 定时时间间隔的设定和中断处理
   - 用 MSKS(690) 设定定时时间间隔。
   - 在执行了 MSKS(690)，且首次定时中断时间已过，正在处理的任务被中断，并执行定时中断任务。
   - 当定时中断任务执行到 END(001) 指令时，程序在定时中断产生的断点处恢复执行。
   - 经过定时时间间隔后，程序执行将中断并再次执行定时中断任务。定时中断任务将重复执行直至被禁止。
4. 禁止定时中断
   - 可以用 MSKS(690) 将定时时间间隔设为 0000，使定时中断任务禁止。
   - 在重新允许定时中断任务时，要确保在使用 MSKS(690) 再次设定定时时间间隔之前，用 CLI(691) 设定首次定时中断时间。

定时中断操作

在下例中，在 PC 设置中将定时时间间隔单位设定为 10ms。

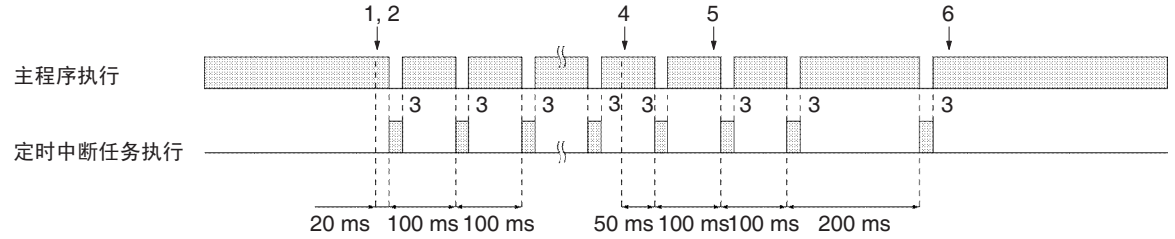


1,2,3...

1. 用 CLI(691) 将首次定时中断时间设定为 20ms。
2. 用 MSKS(690) 将定时时间间隔设为 100ms，并允许定时中断 2 执行。
3. MSKS(690) 执行 20ms 后执行定时中断 2，此后每 100ms 执行一次。
4. 在定时中断处理开始后，可用 CLI(690) 来改变下一个定时中断的时间，但此设置只有效一次。

5. 在定时中断处理开始后，可通过执行 MSKS(690) 来改变定时时间间隔。本例中，时间间隔由 100ms 变为 200ms。
6. 通过执行 MSKS(690) 设定时间间隔为 0000，使定时中断处理禁止。

下面的时间图显示了上面所列实例的操作。

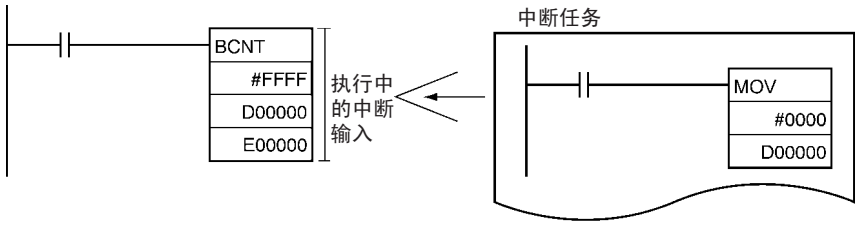


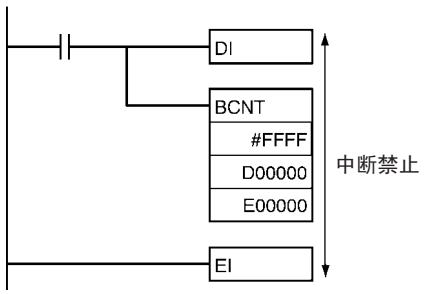
注意

应确保定时时间间隔大于执行定时中断任务所需的时间。如果定时时间间隔太短，中断任务将连续执行并产生一个循环时间太长错误。（长定时中断任务会严重影响主程序的整个执行时间）。

在经过指定时间间隔加一条指令执行时间之后执行定时中断。通常执行一条指令所需的时间可以忽略的，但当使用一条长时间指令时会导致错误，这同样也会导致定时器 （TIM 和 TIMH）和数据跟踪出错。在 PC 设置中将定时时间间隔单位设定为 0.5ms 或 1ms 时要特别小心。

即使在一条指令正在执行时也能接收中断。因此，如果正在执行一条需要长处理时间的指令时接收了中断，则可能得不到正确处理结果，因为中断任务和指令可能访问了同一数据。在这种情况下，用 DI(693) 和 EI(694) 来禁止或允许中断。





3-21 高速计数器 / 脉冲输出指令

本节介绍控制高速计数器 / 脉冲输出的指令。

指令	助记符	功能代码	页号
控制模式	INI	880	768
读高速计数器 PV 值	PRV	881	772
登记比较表	CTBL	882	776
速度输出	SPED	885	780
脉冲设定	PULS	886	785
脉冲输出	PLS2	887	784
加速度控制	ACC	888	791
原点搜索	ORG	889	798
可变占空比脉冲	PWM	891	801

3-21-1 控制模式：INI(880) （仅 CJ1M-CPU22/23）

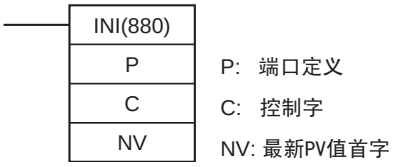
用途

INI(880) 指令用于 CJ1M CPU 单元的内置式 I/O 执行下列操作：

- 启动带高速计数器比较表的比较
- 停止带高速计数器比较表的比较
- 改变高速计数器的 PV 值
- 改变计数器模式下中断输入的 PV 值
- 改变脉冲输出的 PV 值 （原始匹配于 0）
- 停止脉冲输出

这项指令仅由 CJ1M-CPU22/CPU23 CPU 单元支持。

梯形图符号



变化

变化		
	ON 条件时每次周期执行	INI(880)
	上升沿微分执行一次	@INI(880)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

P：端口定义

P 指定操作适用的端口

端口定义	端口
0000 hex	脉冲输出 0
0001 hex	脉冲输出 1
0010 hex	高速计数器 0
0011 hex	高速计数器 1

端口定义	端口
0100 hex	计数器模式下的中断输入 0
0101 hex	计数器模式下的中断输入 1
0102 hex	计数器模式下的中断输入 2
0103 hex	计数器模式下的中断输入 3
1000 hex	PWM(891) 输出 0
1001 hex	PWM(891) 输出 1

C：控制数据

INI(880) 的功能由控制数据 C 决定

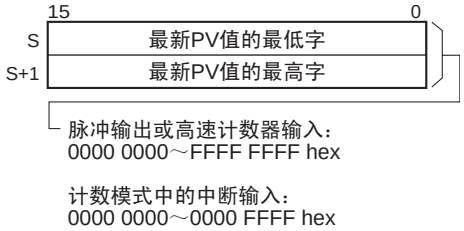
控制数据	INI(880) 的功能
0000 hex	启动比较
0001 hex	停止比较
0002 hex	改变 PV 值
0003 hex	停止脉冲输出

NV：最新 PV 值首字

当 PV 值改变时， NV 和 NV+1 包含最新 PV 值。

如果 C 是 0002（也就是说，当 PV 值改变时）， NV 和 NV+1 包含最新 PV 值。

当 C 不是 0002 时， NV 和 NV+1 中的值将被忽略。



操作数规定

区域	P	C	NV
CIO 区	---	---	CIO 0000 ～ CIO 6142
工作区	---	---	W000 ～ W510
保持位区	---	---	H000 ～ H510
辅助位区	---	---	A448 ～ A958
定时器区	---	---	T0000 ～ T4094
计数器区	---	---	C0000 ～ C4094
DM 区	---	---	D00000 ～ D32766
无区号 EM 区	---	---	---
有区号 EM 区	---	---	---
二进制间接 DM/EM 寻址	---	---	@ D00000 ～ @ D32767
BCD 间接 DM/EM 寻址	---	---	*D00000 ～ *D32767
常数	见操作数说明	见操作数说明	---
数据寄存器	---	---	---

区域	P	C	NV
索引寄存器	---	---	---
使用索引寄存器的间接寻址	---	---	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

说明

对于 P 中指定端口，INI(880) 执行 C 中的指定操作。操作与端口的可能组合如下表所示。

P：端口定义	C：控制字			
	0000：启动比较	0001：停止比较	0002：改变 PV 值	0003：停止脉冲输出
0000 或 0001 Hex：脉冲输出	不允许	不允许	OK	OK
0010 或 0011 Hex：高速计数器输入	OK	OK	OK	不允许
0100， 0101， 0102 或 0103 Hex：计数器模式下的中断输入	不允许	不允许	OK	不允许
1000 或 1001 Hex：PWM(891) 输	不允许	不允许	不允许	OK

■ 启动比较 (C=0000 Hex)

如果 C 是 0000 Hex， INI(880) 启动一个高速计数器的 PV 值与用 CTBL(882) 存入的比较表的比较。

注 目标值比较表必须用 CTBL(882) 先登记。如果在表没有登记的情况下执行 INI(880)，错误标志将变为 ON。

■ 停止比较 (C=0001 Hex)

如果 C 是 0001 Hex， INI(880) 停止一个高速计数器的 PV 值与用 CTBL(882) 写入的比较表的比较。

■ 改变一个 PV 值 (C=0002 Hex)

如果 C 是 0002 Hex， INI(880) 改变一个 PV 值如下表所示。

端口与模式			操作	设置范围
脉冲输出 （P=0000 或 0001 Hex）			脉冲输出的当前值被改变。最新值被指定在 NV 和 NV+1 中。 注 指令仅当脉冲输出被停止时执行。如果它在脉冲输出期间执行，一个错误将会出现。	8000 0000 ～ 7FFF FFFF hex (-2,147,483,648 ～ 2,147,483,647)
高速计数器输入 （P=0010 或 0011 Hex）	线性模式	微分输入，递增 / 递减脉冲，或脉冲 + 输入方向	高速计数器的当前值被改变。最新值被指定在 NV 和 NV+1 中。 注 如果指定端口未被设置为高速计数器，一个错误将因这个指令而出现。	8000 0000 ～ 7FFF FFFF hex (-2,147,483,648 ～ 2,147,483,647)
		递增脉冲输入		0000 0000 ～ FFFF FFFF hex (0 ～ 4,294,967,295)
	环形模式			0000 0000 ～ FFFF FFFF hex (0 ～ 4,294,967,295)
计数器模式下的中断输入 （P=0100， 0101， 0102 或 0103 Hex）			中断输入的当前值被改变。 最新值被指定在 NV 和 NV+1 中。	0000 0000 ～ 0000 FFFF hex (0 ～ 65,535) 注 如果被指定的值在此范围之外，将会出现一个错误。

■ 停止脉冲输出 (P=1000 或 1001 和 C=0003 Hex)

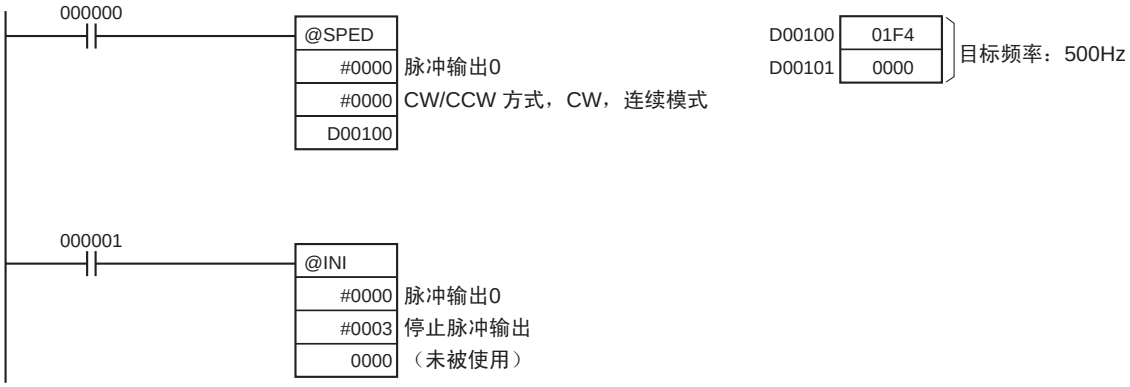
如果 C 是 0003 Hex，INI(880) 立刻停止指定端口的脉冲输出。当脉冲输出已被停止时，如果这个指令被执行，脉冲数设置将被清除。

标志

名称	标记	操作
错误标志	ER	P,C 或 NV 超出指定范围时置 ON。 P 和 C 的组合不允许时置 ON。 比较表尚未登记但指定比较已启动时置 ON。 PV 的最新值被指定给一个正输出脉冲的端口时置 ON。 高速计数器的 PV 值的改变被指定给一个尚未被指定为高速计数器输出的端口时置 ON。 指定的值超出了计数器模式下的中断输入的 PV 值指定的范围时置 ON。 INI(880) 为高速计数器执行一个中断任务并且在 CTBL(882) 执行期间出现一个中断时置 ON。 已执行任务的端口未被设置为计数器模式下的中断输入时置 ON。

例

在下例中当 CIO 000000 变为 ON 时，在连续模式下 SPED(885) 从脉冲输出 0 启动 500HZ 的脉冲输出。当 CIO 000001 变为 ON 时，INI(880) 停止脉冲输出。



3-21-2 读高速计数器 PV 值：PRV(881) （仅 CJ1M-CPU22/-CPU23）

用途

PRV(881) 读 CJ1M CPU 单元的内置式 I/O 中的下列数据：

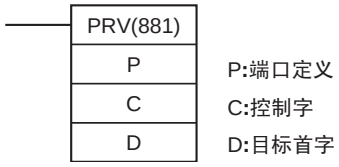
- PV：高速计数器 PV 值，脉冲输出 PV 值，计数器模式下中断输入 PV 值
- 状态信息如下：

状态类型	内容
脉冲输出状态	脉冲输出状态标志 PV 值上溢 / 下溢标志 脉冲输出量设置标志 脉冲输出完成标志 脉冲输出标志 无原点标志 在原点标志 停止脉冲输出错误标志
高速计数器输入状态	比较进行中标志 PV 值上溢 / 下溢标志
PWM(891) 输出状态	脉冲输出进行中标志

- 范围比较结果
- 高速计数器输入 0 的高速计数器频率

这项指令仅由 CJ1M-CPU22/CPU23 CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每次周期执行	PRV(881)
	上升沿微分执行一次	@PRV(881)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

P：端口定义

P 指定操作适用的端口

端口定义	端口
0000 hex	脉冲输出 0
0001 hex	脉冲输出 1
0010 hex	高速计数器 0
0011 hex	高速计数器 1
0100 hex	计数器模式下的中断输入 0
0101 hex	计数器模式下的中断输入 1
0102 hex	计数器模式下的中断输入 2
0103 hex	计数器模式下的中断输入 3
1000 hex	PWM(891) 输出 0
1001 hex	PWM(891) 输出 1

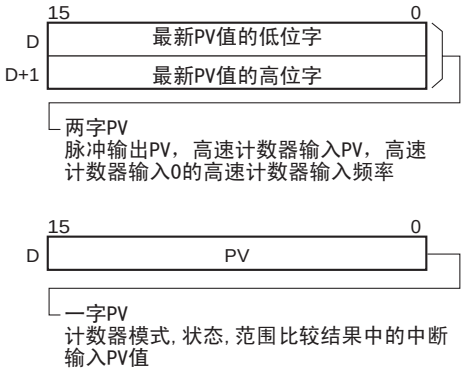
C：控制数据

PRV(881) 的功能由控制数据 C 决定

控制数据	PRV(881) 的功能
0000 hex	读 PV 值
0001 hex	读状态
0002 hex	读范围比较结果
0003 hex	读高速计数器输入 0 的高速计数器频率

D：目标首字

PV 值输出到 D 或 D 和 D+1 中。



操作数规定

区域	P	C	D
CIO 区	---	---	CIO 0000 ～ CIO 6143
工作区	---	---	W000 ～ W511
保持位区	---	---	H000 ～ H511
辅助位区	---	---	A448 ～ A959
定时器区	---	---	T0000 ～ T4095
计数器区	---	---	C0000 ～ C4095
DM 区	---	---	D00000 ～ D32767

区域	P	C	D
无区号 EM 区	---	---	---
有区号 EM 区	---	---	---
二进制间接 DM/EM 寻址	---	---	@ D00000 ～ @D32767
BCD 间接 DM/EM 寻址	---	---	*D00000 ～ *D32767
常数	见操作数说明	见操作数说明	---
数据寄存器	---	---	---
索引寄存器	---	---	---
使用索引寄存器的间接寻址	---	---	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

说明

PRV(881)读P中指定端口C中指定的数据。数据和端口的可能组合如下表所示。

P：端口定义	C：控制数据			
	0000：读 PV 值	0001：读状态	0002：范围比较结果	0003：读高速计数器频率
0000 或 0001 Hex：脉冲输入	OK	OK	不允许	不允许
0010 或 0011 Hex：高速计数器输出	OK	OK	OK	OK （仅高速计数器 0）
0100， 0101， 0102 或 0103 Hex：计数器模式下的中断输入	OK	不允许	不允许	不允许
1000 或 1001 Hex：PWM(891) 输出	不允许	OK	不允许	不允许

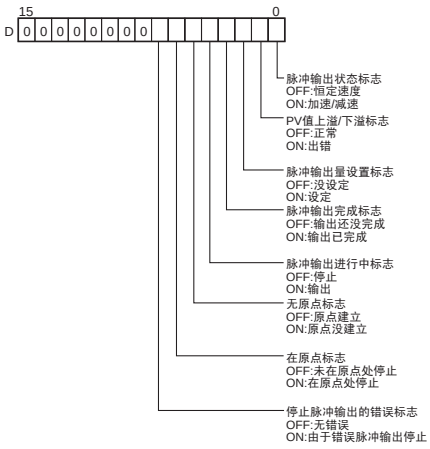
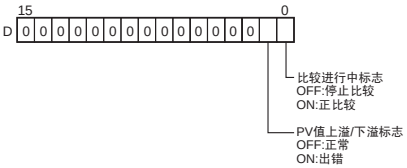
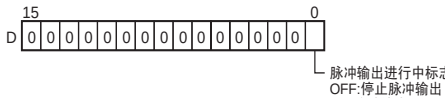
■ 读一个 PV 值 (C=0000 Hex)

如果 C 是 0000 Hex， PRV(881) 读一个 PV 值如下表所示。

端口与模式		操作	设置范围
脉冲输出 （P=0000 或 0001 Hex）		脉冲输出的当前值被存入 D 和 D+1 中。	8000 0000 ～ 7FFF FFFF hex (-2,147,483,648 ～ 2,147,483,647)
高速计数器输入 （P=0010 或 0011Hex）	线性模式	高速计数器的当前值被存入 D 和 D+1 中。	8000 0000 ～ 7FFF FFFF hex (-2,147,483,648 ～ 2,147,483,647)
	环形模式		0000 0000 ～ FFFF FFFF hex (0 ～ 4,294,967,295)
计数器模式下的中断输入 （P=0100，0101，0102 或 0103 Hex）		中断输入的当前值被存入 D 中。	0000 ～ FFFF hex (0 ～ 65,535)

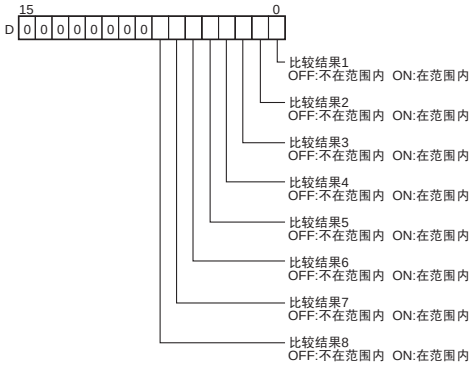
■ 读状态 （C=0001 Hex）

如果 C 是 0001 Hex， PRV(881) 读状态如下表所示。

端口与模式	操作	读的结果
脉冲输出	脉冲输出状态被存储在 D 中	
高速计数器输入	高速计数器状态被存入 D 中	
PWM(891) 输出	PWM(891) 输出被存入 D 中	

■ 读范围比较结果 (C=0002 Hex)

如果C是0002 Hex，PRV(881)读范围比较的结果并把它存入D中，如下图所示。



■ 读高速计数器输出频率 (C=0003 Hex)

如果 C 是 0003 Hex，PRV(881) 读以 Hz 为单位的高速计数器 0 的输出频率并存入 D 和 D+1 中。它的值在 0000 0000 和 0001 86A0 之间（0 ～ 100,000）。即使输出频率超过 100KHz，将会把最大值 0001 86A0 存入结果中。

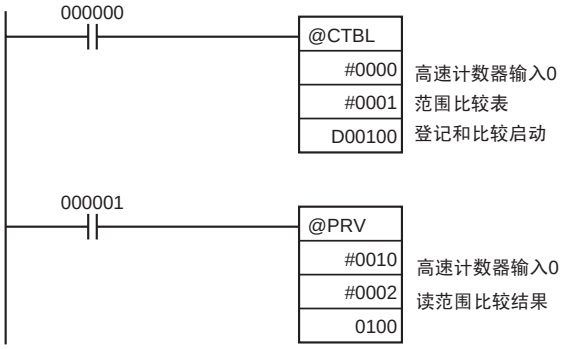
标志

名称	标记	操作
错误标志	ER	P 或 C 的值超出指定范围时置 ON。 P 和 C 的组合不允许时置 ON。 读范围比较结果被指定但范围比较尚未执行时置 ON。 读除高速计数器 0 以外的输出频率时置 ON。 指定的端口尚未被设置为高速计数器时置 ON。 执行的端口尚未被设置为计数模式下的中断输入时置 ON。

例

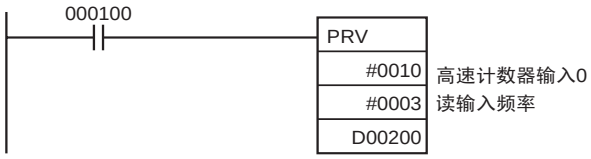
■ 例 1

下面程序实例中当 CIO 000000 变为 ON 时，CTBL(882) 为高速计数器 0 登记一个范围比较表中并启动比较。当 CIO 000001 变为 ON 时，PRV(881) 读当时的范围比较结果并把它们存入 CIO 0100。



■ 例 2

下面程序实例中当 CIO 000000 变为 ON 时，PRV(881) 读当时输入高速计数器 0 脉冲的频率并把它以十六进制数形式存入 D00200 和 D00201 中。

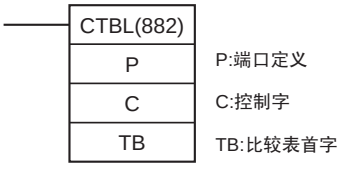


3-21-3 登记比较表：CTBL(882) （仅 CJ1M-CPU22/-CPU23）

用途

CTBL(882) 指令用于登记一个比较表并完成对一个高速计数器 PV 值比较。比较可对目标值或范围进行。当指定条件达到时执行中断任务。这项指令仅由 CJ1M-CPU22/-CPU23 CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每次周期执行	CTBL(882)
	上升沿微分执行一次	@CTBL(882)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

P：口定义

P 指定其脉冲将要被计数的端口，如下表所示。

端口定义	端口
0000 hex	高速计数器 0
0001 hex	高速计数器 1

C：控制数据

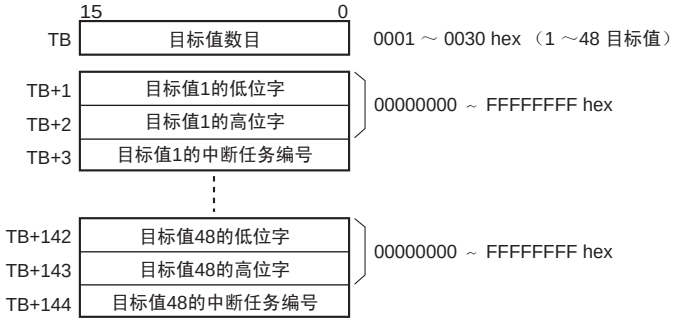
CTBL(882) 的功能由如下表中控制数据 C 决定

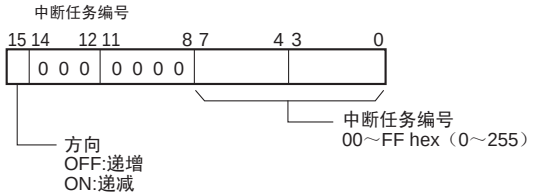
控制数据	CTBL(882) 的功能
0000 hex	登记一个目标值比较表并启动比较
0001 hex	登记一个范围比较表并执行一项比较
0002 hex	登记一个目标值比较表。 INI(880) 启动比较
0003 hex	登记范围比较表。 INI(880) 启动比较

TB：比较表首字

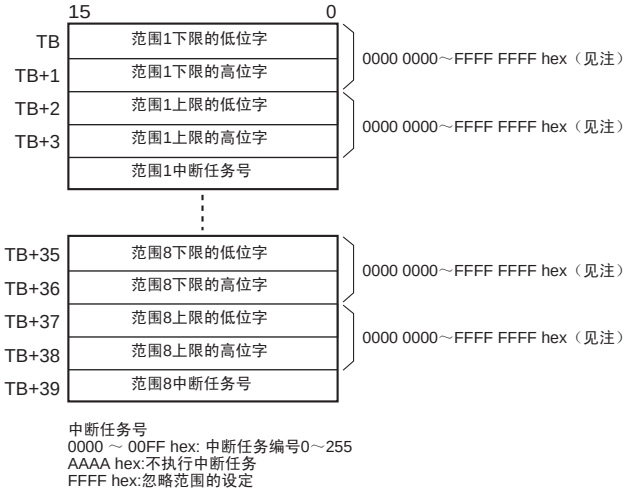
TB 是比较表的首字。比较表的结构依赖于执行的比较的类型。

对于目标值比较，比较表的长度由 TB 中指定的目标值的数目决定。表长可在 4 ～ 145 字间，如下所示。





对于范围比较，比较表通常包含 8 个范围。表是 40 字长，如下所示。若没必要设定 8 个范围，把所有没未使用范围的中断任务编号设为 FFFF hex。



注 任何一个范围通常都设置为上限大于或等于下限。

操作数规定

区域	P	C	TB
CIO 区	---	---	CIO 0000 ～ CIO 6143
工作区	---	---	W000 ～ W511
保持位区	---	---	H000 ～ H511
辅助位区	---	---	A448 ～ A959
定时器区	---	---	T0000 ～ T4095
计数器区	---	---	C0000 ～ C4095
DM 区	---	---	D00000 ～ D32767
无区号 EM 区	---	---	---
有区号 EM 区	---	---	---
二进制间接 DM/EM 寻址	---	---	@ D00000 ～ @ D32767
BCD 间接 DM/EM 寻址	---	---	*D00000 ～ *D32767
常数	见操作数说明	见操作数说明	---
数据寄存器	---	---	---
索引寄存器	---	---	---
使用索引寄存器的间接寻址	---	---	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

说明

CTBL(882) 登记一个比较表或登记比较表并启动按 P 中指定端口和 C 中指定方法的比较。一旦登记比较表，它就有效直至一个不同表被登记或 CPU 单元被切换为编程 (PROGRAM) 模式。

每次 CTBL(882) 执行，比较在指定条件下启动。当使用 CTBL(882) 来启动比较时，一般使用微分形式 (@CTBL(882)) 的指令或仅因一次扫描而置 ON 的执行条件。

注 若一个尚未登记中断任务被指定，第一次中断产生时将会出现一个致命的程序错误。

■ 登记比较表 (C=0002 或 0003 Hex)

如果 C 被设为 0002 或 0003 Hex，一个比较表将被登记，但比较将不会被启动。比较由 INI(880) 启动。

■ 比较表登记和比较启动 (C=0000 或 0001 Hex)

如果 C 被设为 0000 或 0001 Hex，比较表将被登记，且启动比较。

■ 停止比较

用 INI(880) 停止比较。它与用来启动比较的指令没差别。

■ 目标值比较

当 PV 值与一个目标值相符时，相应的中断任务将被调用并执行。

- 同一中断任务编号能被指定给一个或多个目标值。
- 当PV值被递增或递减时，方向能被设置成指定目标值是否有效。如果用来指定中断任务编号范围字中的第 15 位是 OFF，仅当 PV 递增时， PV 值将与目标值比较。当 00 位是 ON 时，仅当 PV 值递减时， PV 值将与目标值比较。
- 比较表能包含48个目标值，且目标值的数量在TB中指定。（也就是说表的长度依赖于指定目标值的数量）
- 对表中登记的所有目标值进行比较。

注
1. 若带相同的比较方向的同一目标值在同一表中登记超过一次，将会出现一个错误。
2. 如果高速计数器被设置为递增脉冲模式，递减被设置为表的比较方向时将出现错误。
3. 如果当PV值等于与设置的比较方向相反的目标值时计数方向改变，则那个目标值的比较条件将不会达到。不要设定目标值为计数值的峰值和谷值。

<u>范围比较</u>

当 PV 值落在某一个设定范围时，相应的中断任务被调用并执行。

- 相同的中断任务编号可被指定给多个目标值。
- 范围比较表包含8个范围，每个均由下限和上限定义。如果范围未被使用，中断任务编号设为 FFFF 来禁止范围。
- 仅当 PV 值落在其范围内时，执行中断任务。

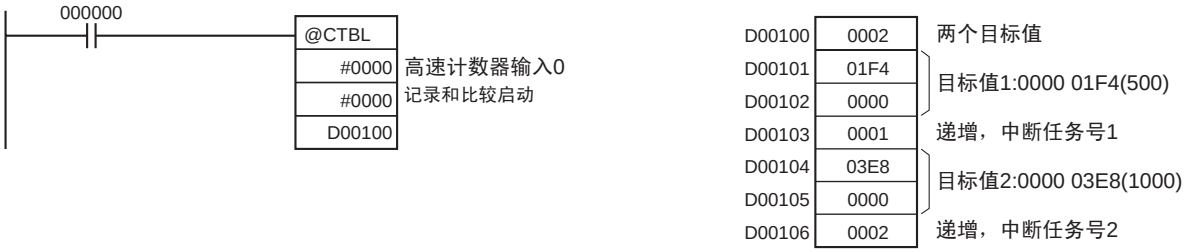
- 如果比较后 PV 值在不止一个范围内，中断任务的范围接近于表的起始位置将获优先权，而其它中断任务则在下一循环中执行。
- 如果不需要执行中断任务，可指定 AAAA Hex 为中断任务编号。可用 PRV(881) 读范围比较结果或使用范围比较进行中标志。

注 任何范围的上限比下限小时会出现一个错误标志。

标志

名称	标记	操作
E 错误标志	ER	P 或 C 的值超出指定范围时置 ON。 在同一比较方向上用于目标值比较的相同比较值被多次指定时置 ON。 任何一个范围内上限值小于下限值时置 ON。 高速计数器被设置为递增脉冲模式而表中比较方向设置为递减时置 ON。 指定一个尚未设为高速计数器的端口时置 ON。 在比较正在进行期间执行一个不同的比较方法时置 ON。

例 下面程序实例中当 CIO 000000 变为 ON 时， CTBL(882) 登记一个范围比较表并启动高速计数器 0 的比较。当高速计数器的 PV 值计数递增，且增至 500 时，PV 值等于目标值 1，则执行中断任务 1。当 PV 递增至 1000 时，等于目标值 2，中断任务 2 执行。



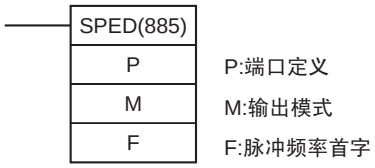
3-21-4 速度输出：SPED(885) （仅 CJ1M-CPU22/CPU23）

用途 SPED(885) 用于设定指定端口的输出脉冲频率和启动不带加 / 减速度的脉冲输出。无论独立模式定位或者连续模式速度控制都可适用。对于独立模式位置控制，使用 PULS(886) 设定脉冲数。

可以用 SPED(885) 在脉冲输出时改变输出频率，而使速度逐步地变化。

此指令仅由 CJ1M-CPU22/CPU23 CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每次周期执行	SPED(885)
	上升沿微分执行一次	@SPED(885)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

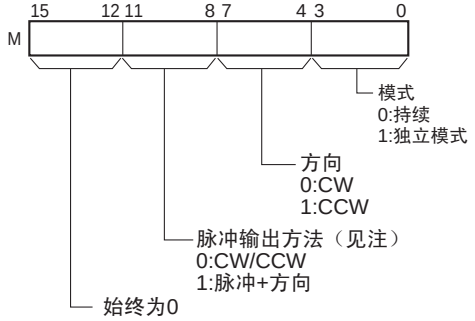
P：端口定义

P 指定将要输出脉冲的端口

端口定义	端口
0000 hex	脉冲输出 0
0001 hex	脉冲输出 1

M：输出模式

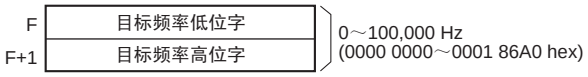
M 的值决定输出模式。



注 当同时使用脉冲输出0和1时，使用相同脉冲输出方法。

F：脉冲频率首字

F 和 F+1 的值设定脉冲频率 (Hz)。



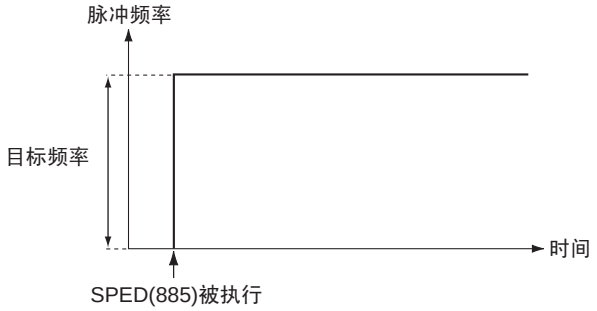
操作数规定

区域	P	M	F
CIO 区	---	---	CIO 0000 ～ CIO 6142
工作区	---	---	W000 ～ W510
保持位区	---	---	H000 ～ H510

区域	P	M	F
辅助位区	---	---	A448 ～ A958
定时器区	---	---	T0000 ～ T4094
计数器区	---	---	C0000 ～ C4094
DM 区	---	---	D00000 ～ D32766
无区号 EM 区	---	---	---
有区号 EM 区	---	---	---
二进制间接 DM/EM 寻址	---	---	@ D00000 ～ @ D32767
BCD 间接 DM/EM 寻址	---	---	*D00000 ～ *D32767
常数	见操作数说明	见操作数说明	见操作数说明
数据寄存器	---	---	---
索引寄存器	---	---	---
使用索引寄存器的间接寻址	---	---	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

说明

SPED(885) 在 F 中指定的频率下，使用 M 中指定的方法及在 P 中指定端口启动脉冲输出。每次 SPED(885) 执行时，将启动脉冲输出。因此，它通常使用微分形式 (@SPED(885)) 的指令或一个仅因一次扫描而置 ON 的执行条件。



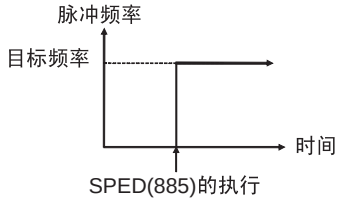
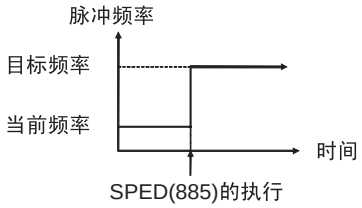
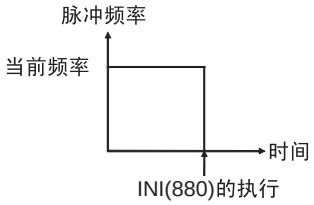
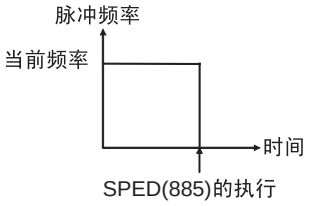
在独立模式下，当用 PULS(886) 预先设置的脉冲数已输出时，脉冲输出将会自动停止。在连续模式下，脉冲输出将持续直至被程序停止。

当脉冲正在输出时，如果模式在独立模式和连续模式之间改变，将出现一个错误。

■ 连续模式速度控制

当启动连续模式操作时，脉冲输出将持续直至程序命令停止。

注 如果 CPU 单元切换为编程 (PROGRAM) 模式，脉冲输出将立刻停止。

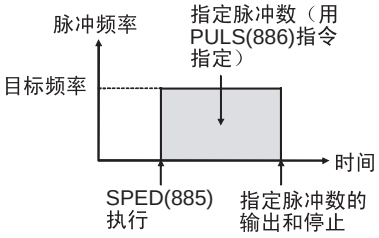
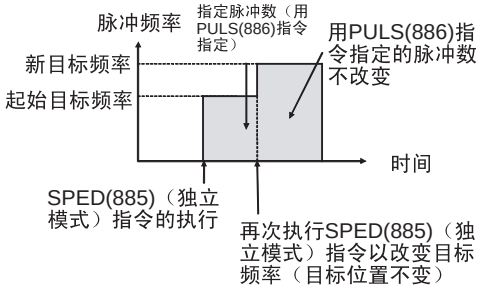
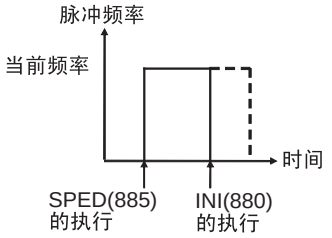
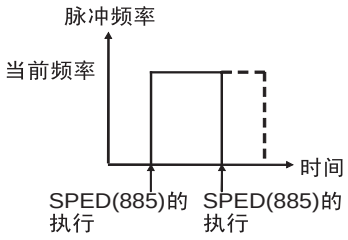
操作	用途	应用	频率改变	说明	操作顺序 / 指令
启动脉冲输出	以指定速度输出	以阶跃方式改变速度（频率）		以指定的频率输出脉冲。	SPED(885) （连续模式）
改变设置	以阶跃方式改变速度	操作时改变速度		以阶跃方式改变脉冲输出频率（高或低）。	SPED(885) （连续模式） ↓ SPED(885) （连续模式）
停止脉冲输出	停止脉冲输出	立刻停止		立刻停止脉冲输出。	SPED(885) （连续模式） ↓ INI(880)
	停止脉冲输出	立刻停止		立刻停止脉冲输出。	SPED(885) （连续模式） ↓ SPED(885) （连续模式，目标频率为 $0Hz$）

■ 独立模式位置控制

当启动独立模式操作时，脉冲输出将持续直至指定的脉冲数全部输出。

注
1. 如果 CPU 单元切换为 PROGRAM 模式，脉冲输出将立刻停止。
2. 每次输出重新启动时，输出脉冲数必须设定。
3. 输出脉冲数必须用 PULS(881) 指令预先设定。如果不首先执行 PULS(881) 指令，SPED(885) 指令将不会输出脉冲。

4. 如果用 PULS(881) 指令把脉冲数设定为绝对值，SPED(885) 指令操作数的方向设置将被忽略。

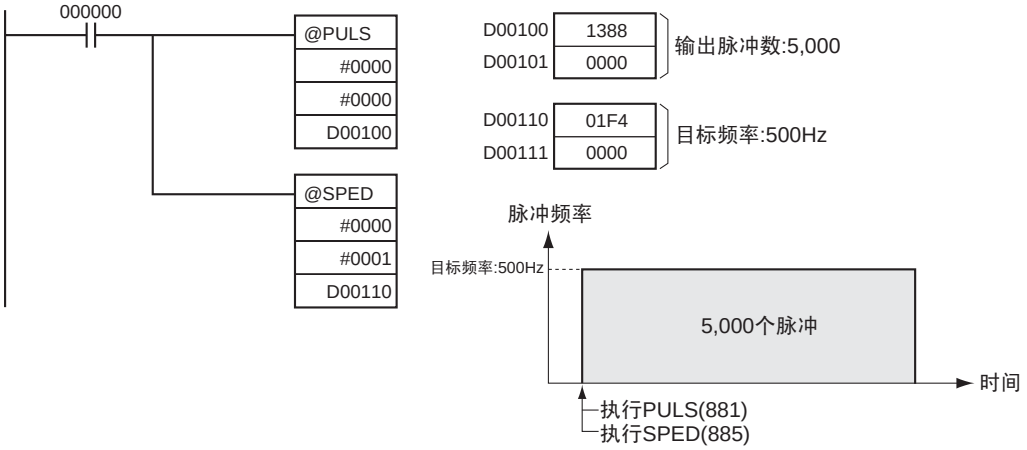
操作	用途	应用	频率改变	说明	操作顺序 / 指令
启动脉冲输出	以指定速度输出	不带加速和减速的位置控制		启动指定频率脉冲输出和当指定脉冲数已经输出时立刻停止脉冲输出。 **注** 目标位置 （指定脉冲数）在定位时不改变。	PULS(886) ↓ SPED(885) （独立模式）
改变设置	以阶跃方式改变速度	操作时以阶跃方式改变速度		在定位以阶跃方式改变脉冲输出频率 （升或降）时执行 SPED(885)。目标位置 （指定脉冲数）不改变	PULS(886) ↓ SPED(885) （独立模式） ↓ SPED(885) （独立模式）
停止脉冲输出	停止脉冲输出（输出脉冲数未保存）	立刻停止		立刻停止脉冲输出和清除输出脉冲数设置。	PULS(886) ↓ SPED(885) （独立模式） ↓ INI(880) PLS2(887) ↓ INI(880)
	停止脉冲输出（输出脉冲数未保存）	立刻停止		立刻停止脉冲输出和清除输出脉冲数设置。	PULS(886) ↓ SPED(885) （独立模式） ↓ SPED(885), （独立模式，目标频率为0Hz）

标志

名称	标记	操作
错误标志	ER	P， M 或 F 的值超出指定范围时置 ON。 如果 PLS2(887) 或 ORG(889) 正用于控制指定端口的脉冲输出时置 ON。 脉冲输出期间用 SPED(885) 或 INI(880) 在连续模式和独立模式间切换输出时置 ON。 在循环任务中执行控制脉冲输出指令期间用 SPED(885) 执行中断任务时置 ON。 独立模式下 SPED(885) 执行绝对脉冲数输出且原点尚未建立时置 ON。

例

下面程序实例中当 CIO 000000 变为 ON 时，PULS(886) 设置脉冲输出 0 的输出脉冲数。设置绝对值为 5000 的脉冲数。执行 SPED(885) 指令，在独立模式下沿顺时针方向用 CW/CCW 方法立刻启动目标频率为 500Hz 的脉冲输出。



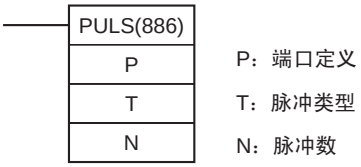
3-21-5 脉冲设定：PULS(886) （仅 CJ1M-CPU22/CPU23 适用）

用途

PULS(886) 用于设定脉冲输出的脉冲数量 （输出脉冲的数），在独立模式下，这些脉冲的输出在随后的程序中用 SPED(885) 或 ACC(888) 启动。

这项指令仅由 CJ1M-CPU22/CPU23 CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每次周期执行	PULS(886)
	上升沿微分执行一次	@PULS(886)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

P：端口定义

P 指明端口。D 和 N 中参数设置将用于下一条指定相同端口输出位置的 SPED(885) 或 ACC(888) 指令。

端口定义	端口
0000 hex	脉冲输出 0
0001 hex	脉冲输出 1

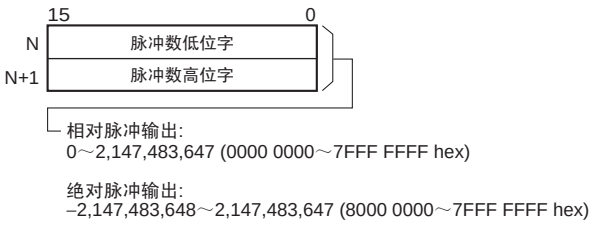
T：脉冲类型

T 指定输出脉冲类型，如下所示：

T	脉冲类型
0000 hex	相对
0001 hex	绝对

N 和 N+1：脉冲数

N 和 N+1 指定相对脉冲输出的脉冲数或 8 数字十六进制绝对脉冲的绝对目标位置。



将要输出的实际移动脉冲数如下：

对于相对脉冲输出，移动脉冲数 = 设定脉冲数。对于绝对脉冲输出，移动脉冲数 = 设定脉冲数 -PV 值。

操作数规定

区域	P	T	N
CIO 区	---	---	CIO 0000 ～ CIO 6142
工作区	---	---	W000 ～ W510
保持位区	---	---	H000 ～ H510
辅助位区	---	---	A448 ～ A958
定时器区	---	---	T0000 ～ T4094
计数器区	---	---	C0000 ～ C4094
DM 区	---	---	D00000 ～ D32766
无区号 EM 区	---	---	---
有区号 EM 区	---	---	---
二进制间接 DM/EM 寻址	---	---	@ D00000 ～ @ D32767

区域	P	T	N
BCD 间接 DM/EM 寻址	---	---	*D00000 ～ *D32767
常数	见操作数说明	见操作数说明	见操作数说明
数据寄存器	---	---	---
索引寄存器	---	---	---
使用索引寄存器的间接寻址	---	---	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

说明

PULS(886) 设置 P 中指定端口的 T 和 N 中的指定脉冲类型和脉冲数。在随后程序中使用 SPED(885) 或 ACC(888) 启动独立模式下的实际脉冲输出。

标志

名称	标记	操作
错误标志	ER	P， T， N 的值超出指定范围时置 ON。 执行 PULS(886) 的端口正在执行输出脉冲时置 ON。 循环任务中，在执行控制脉冲输出指令期间用 PULS(886) 执行中断任务时置 ON。

注意

t脉冲正被输出期间执行PULS(886)将出现错误。使用微分形式(@ PULS(886))指令或 变为 ON 仅为一次扫描时间的执行条件，以防止错误的出现。

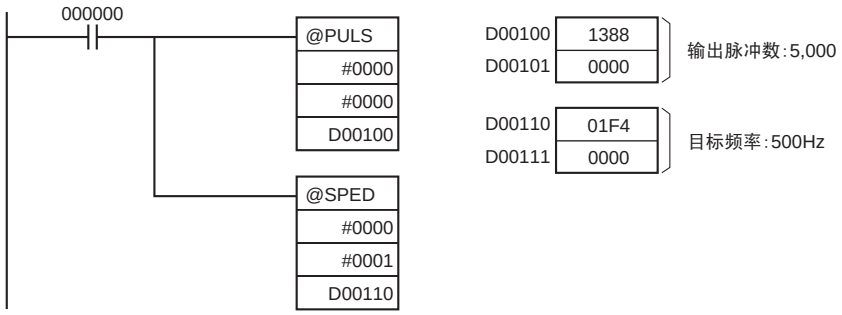
- 即使INI(880)用来改变脉冲输出的PV值，PULS(886)计算的输出脉冲数将不会改变。
- 如果 PULS(886) 把脉冲数设为绝对值，SPED(885) 或 ACC(888) 的方向设置将被忽略。

t 脉冲输出数量的 PV 值的范围可以移出它的范围 (-2,147,483,648~2,147,483,647)。

例

下面程序实例中当 CIO 000000 变为 ON 时， PULS(886) 设定脉冲输出 0 的输出脉冲数。

设置 5000 个脉冲的绝对值， SPED(885) 在独立模式下沿顺时针方向用 CW/CCW 方式立刻启动目标频率为 500Hz 的脉冲输出。



3-21-6 脉冲输出：PLS2(887)（仅 CJ1M-CPU22/CPU23 适用）

用途

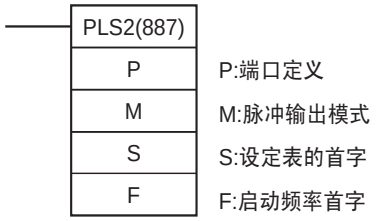
PLS2(887) 指令把指定脉冲数输出到指定端口。脉冲在指定启动频率处启动输出，以指定加速率增加到目标频率，以指定减速率减小到目标频率，停止在正好与启动频率相同处。指令仅适用于定位控制。

在脉冲输出来期间，PLS2(887) 指令也可用于改变输出脉冲数、目标频率、加 / 减速率的执行。

因此， PLS2(887) 指令用于带不同加 / 减速率、目标定位改变、目标与速度改变或方向改变的有坡度的速度改变。

此指令仅由 CJ1M-CPU22/CPU23 CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每次周期执行	PLS2(887)
	上升沿微分执行一次	@PLS2(887)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

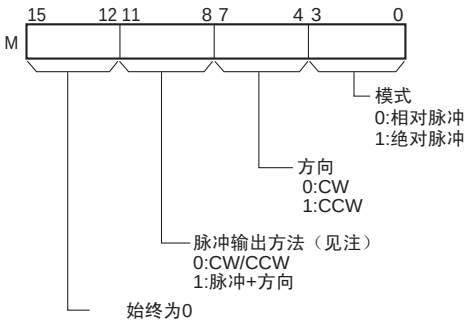
P：端口定义

P 指明端口。

端口定义	端口
0000 hex	脉冲输出 0
0001 hex	脉冲输出 1

M：输出模式

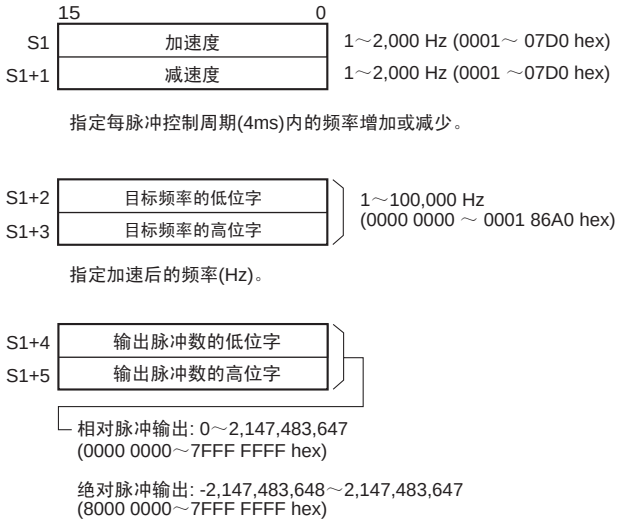
M 的内容指定脉冲输出参数，如下所示：



注 当同时使用脉冲输出0和1时，请使用相同脉冲输出方法。

S：设定表的首字

S 和 S+5 的内容控制脉冲输出如下图所示。

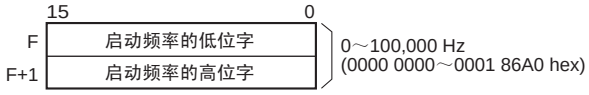


将要输出的实际移动脉冲数如下：

对于相对脉冲输出，移动脉冲数 = 设定脉冲数。对于绝对脉冲输出，移动脉冲数 = 设定脉冲数 -PV 值。

F：启动频率首字

在 F 和 F+1 中给出启动频率。



操作数规定

区域	P	M	S	F
CIO 区	---	---	CIO 0000 ～ CIO 6138	CIO 0000 ～ CIO 6142
工作区	---	---	W000 ～ W506	W000 ～ W510
保持位区	---	---	H000 ～ H506	H000 ～ H510
辅助位区	---	---	A448 ～ A954	A448 ～ A958
定时器区	---	---	T0000 ～ T4090	T0000 ～ T4094
计数器区	---	---	C0000 ～ C4090	C0000 ～ C4094
DM 区	---	---	D00000 ～ D32762	D00000 ～ D32766
无区号 EM 区	---	---	---	---
有区号 EM 区	---	---	---	---
二进制间接 DM/EM 寻址	---	---	@ D00000 ～ @ D32767	@ D00000 ～ @ D32767
BCD 间接 DM/EM 寻址	---	---	*D00000 ～ *D32767	*D00000 ～ *D32767
常数	见操作数说明	见操作数说明	---	见操作数说明
数据寄存器	---	---	---	---

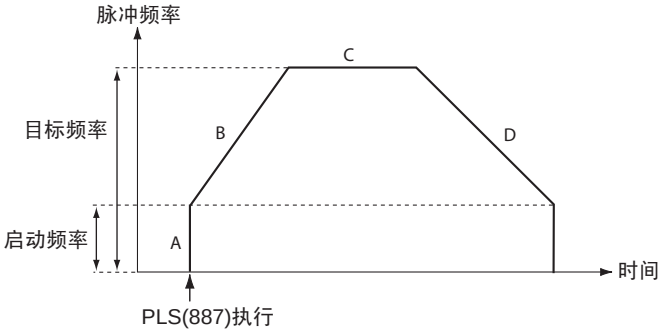
区域	P	M	S	F
索引寄存器	---	---	---	---
使用索引寄存器 的间接寻址	---	---	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

说明

在 M 中的指定模式下，PLS2(887) 在 P 中指定端口启动 F 中指定启动频率的脉冲输出（图中 1）。频率在每个脉冲控制周期 (4ms) 内以 S 中指定加速率增加，直至频率值达到 S 中指定目标频率 （图中 2）。当目标频率达到时，加速度停止并且脉冲以恒定速度输出 （图中 3）。

减速度点通过脉冲数和设置于 S 减速率来计算，到达减速率点，频率在每个脉冲控制周期 (4ms) 内以 S 中指定减速率减小，直至频率值达到 S 中指定的启动频率。当启动频率达到时，脉冲输出停止 （图中 4）。

每次PLS2(887)执行时启动脉冲输出。因此，一般使用微分形式指令(@PLS2(887))或变为 ON 仅为一次扫描时间的执行条件。



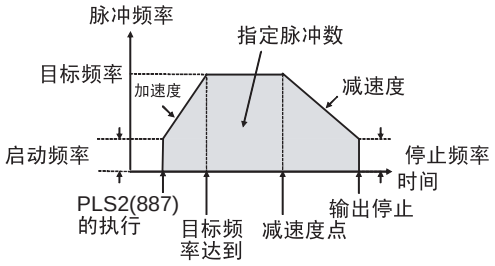
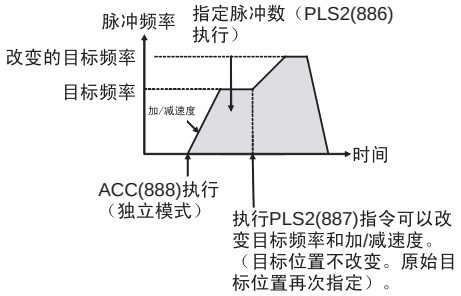
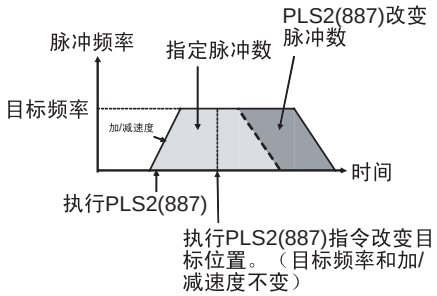
PLS(887) 指令仅适用于定位控制。

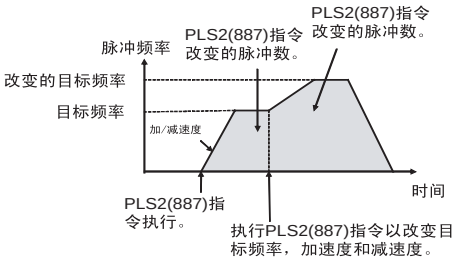
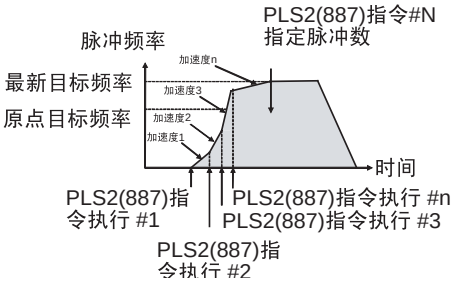
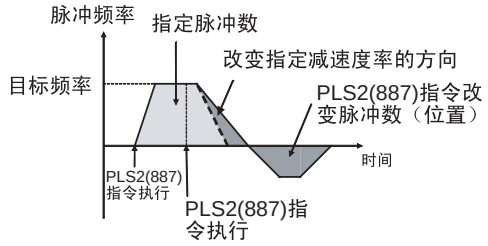
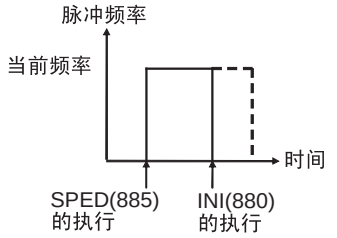
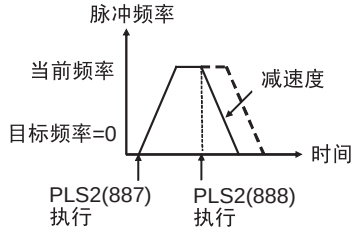
对于 CJ1M CPU 单元，在独立模式或连续模式下，在 ACC(888) 进行脉冲输出以及加速度，恒定速度或减速度期间， PLS2(887) 指令可以执行 （见注）。在用 PLS2(887) 指令进行脉冲输出以及加速度，恒定速度或减速度期间， ACC(888) 指令也可以执行。

注 在使用 ACC(888) 指令 （连续模式）进行速度控制期间 , 执行与 ACC(888) 指令相同的目标频率和一个 0 加速率的 PLS2(887) 指令，可用于获得一个定距离的中断馈给。

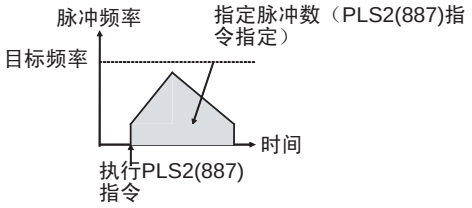
■ 独立模式位置控制

注　如果 CPU 单元切换为编程 (PROGRAM) 模式，脉冲输出将立刻停止。

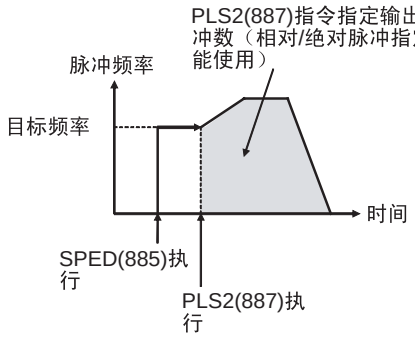
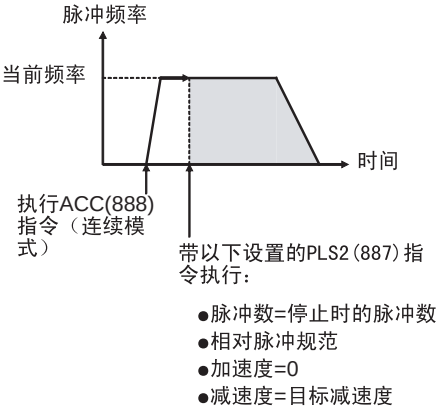
操作	用途	应用	频率改变	说明	操作顺序 / 指令
启动脉冲输出	复杂的梯形控制	带梯形加速度和减速度定位（使用独立的加减速率；启动速度）定位时脉冲数可以改变。		以固定比例进行加速和减速。当指定脉冲数已输出时，脉冲输出停止。（见注） 注　定位期间目标位置（指定脉冲数）能改变。	PLS2(887)
改变设置	平缓的改变速度（带不同加 / 减速率）	定位（不同的加 / 减速率）时改变目标速度（频率）		定位控制期间，可执行 PLS2(887) 指令改变加 / 减速率和目标频率。 注　为防止目标位置改变，原始目标位置必须在绝对坐标系中指定。	PLS2(887) ↓ PLS2(887) PULS(886) ↓ ACC(888)（独立模式） ↓ PLS2(887)
	改变目标位置	位置控制期间改变目标位置（多路启动功能）		在位置控制期间，执行 PLS2(887) 指令以改变目标位置（脉冲数）定位、加 / 减速率和目标频率）。 注　如果改变设置后恒定速度不能维持，一个错误将出现，且原来的操作将持续到原来的目标位置。	PLS2(887) ↓ PLS2(887) PULS(886) ↓ ACC(888)（独立模式） ↓ PLS2(887)

操作	用途	应用	频率改变	说明	操作顺序 / 指令
改变设置（续）	平缓的改变速度和目标位置	位置控制期间平缓的改变目标速度（频率）和目标位置（多路启动功能）		在位置控制期间，执行 PLS2(887) 指令以改变目标位置（脉冲数）定位、加 / 减速率和目标频率）。 注 如果改变设置后恒定速度不能维持，一个错误将出现，且原来的操作将持续到原来的目标位置。	PULS(886) ↓ ACC(888)（独立模式） ↓ PLS2(887)
		位置控制期间（多路启动功能）改变加 / 减速率		位置控制期间（加速度或减速度）可执行 PLS2(887) 指令以改变加 / 减速率。	PLS2(887) ↓ PLS2(887) PULS(886) ↓ ACC(888)（独立模式） ↓ PLS2(887)
	改变方向	位置控制期间改变方向		以绝对脉冲要求的位置控制期间，可执行 PLS2(887) 指令以改变绝对脉冲数和方向。	PLS2(887) ↓ PLS2(887) PULS(886) ↓ ACC(888)（独立模式） ↓ PLS2(887)
停止脉冲输出	停止脉冲输出（脉冲数设置未保存）。	立刻停止		立刻停止脉冲输出并清除输出脉冲数。	PLS2(887) ↓ INI(880)
	平缓的停止脉冲输出（脉冲数设置未保存）。	减速到停止		脉冲输出减速，直至停止。	PLS2(887) ↓ ACC(888)（独立模式，目标频率 0Hz）。

注 三角形控制
如果指定脉冲数小于所要达到的目标频率的脉冲数并返回 0，其功能将自动减小加速 / 减速时间并执行三角形控制 （仅加速与减速）。错误将不会出现。



■ 从连续模式速度控制到独立模式定位转换

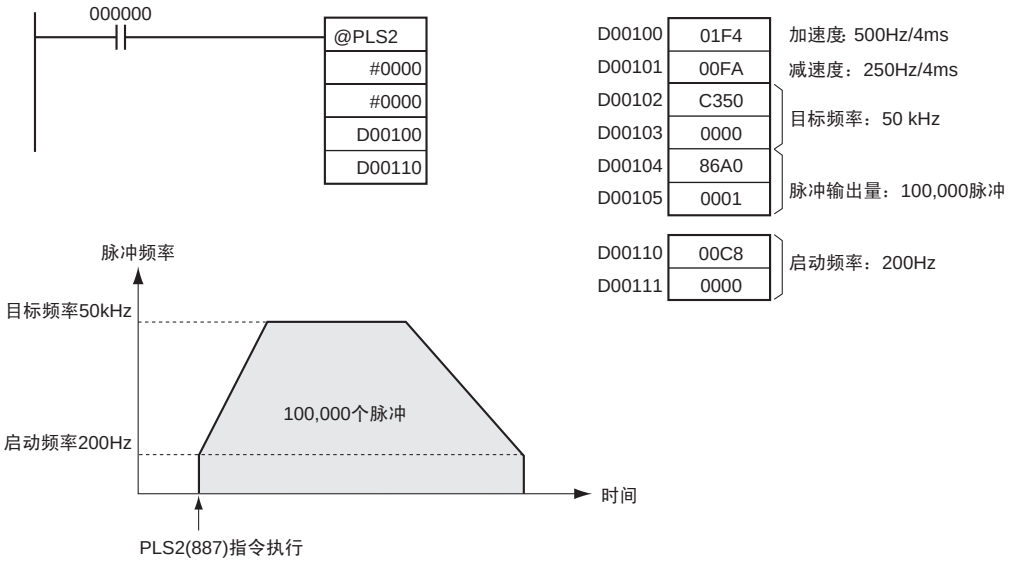
应用实例	频率变化	说明	错做程序 / 指令
操作期间从速度控制到定距离位置控制的改变		由 ACC(888) 启动的速度控制操作期间可执行 PLS2(887) 指令以改变到位置控制操作。	ACC(888)（连续模式） ↓ PLS2(887)
定距离馈给中断			

标志

名称	标记	操作
错误标志	ER	P， M， S 或 F 的值超出指定范围时置 ON。 执行 PLS2(887) 的端口已由 SPED(885) 或 ORG(889) 指令脉冲输出时置 ON。 在循环任务中执行控制脉冲输出指令期间用 PLS2(887) 执行中断任务时置 ON。 PLS2(887) 执行绝对脉冲输出但原点还没被设置时置 ON。

例

在下列程序实例中当 CIO 000000 变为 ON， PLS2(887) 启动脉冲输出 0 以绝对脉冲要求的 100,000 个脉冲的脉冲输出。脉冲输出以启动频率为 200Hz 加速率为 500Hz/4ms 增加，直至达到目标速度 50 KHz。从减速度点看，脉冲以减速率 250Hz/4ms 输出，直至达到启动速度 200Hz，在此点脉冲输出停止。



3-21-7 加速度控制：ACC(888) （仅 CJ1M-CPU22/CPU23 适用）

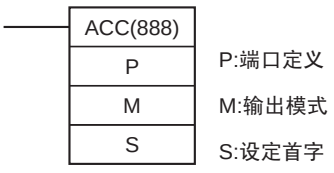
用途

ACC(888) 指令在使用指定加 / 减速率（加速率和减速率相同），以指定的频率向指定的输出口输出脉冲。独立模式位置控制或恒定速度模式控制都可使用。对于位置控制， ACC(888) 与 PULS(886) 结合一起使用。

脉冲输出时也可执行 ACC(888) 指令以改变目标频率或加 / 减速率及使速度平缓 （带斜度的）变化。

这条指令仅由 CJ1M-CPU22/CPU23 CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每次周期执行	ACC(888)
	上升沿微分执行一次	@ACC(888)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

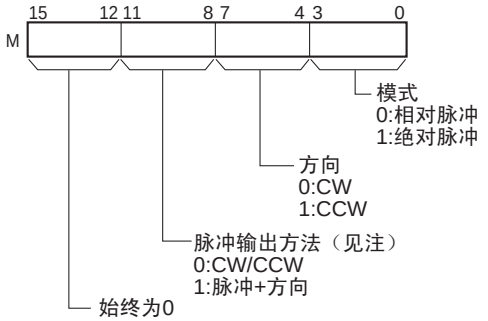
P：端口定义

P 指定将要被计数的脉冲输出端口。

端口定义	端口
0000 hex	脉冲输出 0
0001 hex	脉冲输出 1

M：输出模式

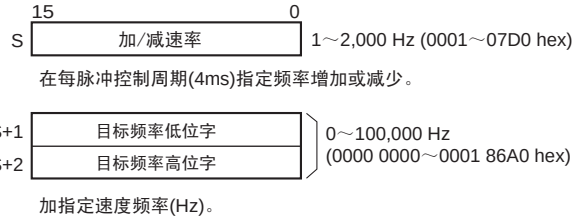
M 的内容指定脉冲输出参数，如下所示：



注 当同时使用脉冲输出0和1时，使用相同脉冲输出方法。

S：设定表的首字

S ～ S+2 的内容控制脉冲输出如下图所示。



操作数规定

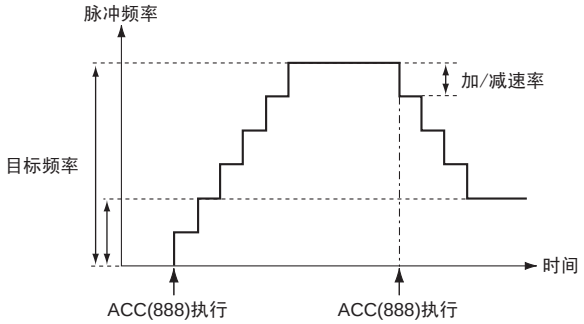
区域	P	M	S
CIO 区	---	---	CIO 0000 ～ CIO 6141
工作区	---	---	W000 ～ W509
保持位区	---	---	H000 ～ H509
辅助位区	---	---	A448 ～ A957
定时器区	---	---	T0000 ～ T4093
计数器区	---	---	C0000 ～ C4093
DM 区	---	---	D00000 ～ D32765
无区号 EM 区	---	---	---
有区号 EM 区	---	---	---

区域	P	M	S
二进制间接 DM/EM 寻址	---	---	@ D00000 ～ @ D32767
BCD 间接 DM/EM 寻址	---	---	*D00000 ～ *D32767
常数	见操作数说明	见操作数说明	---
数据寄存器	---	---	---
索引寄存器	---	---	---
使用索引寄存器的间接寻址	---	---	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

说明

ACC(888) 在 M 中指定模式下从 P 中指定端口启动使用指定目标频率和 S 中加 / 减速率脉冲输出。频率以 S 中指定加速率 / 脉冲控制周期 (4ms) 增加，直至达到 S 中指定目标频率。

每次执行ACC(888)指令时，启动脉冲输出。因此，一般使用微分形式指令@ACC(888)或变为 ON 仅为一次扫描时间的执行条件。



在独立模式下，当指定的脉冲数全部输出时，脉冲输出会自动停止。在连续模式下，在程序命令停止前，脉冲连续输出。

脉冲输出期间，如果在独立模式和连续模式之间切换将会出错。

CJ1M CPU 单元在独立模式或连续模式下用 ACC(888) 以加速、恒速或减速进行脉冲输出时，可以执行 PLS2(887) 指令。（见注）

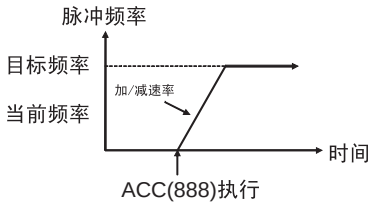
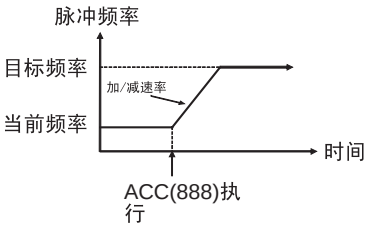
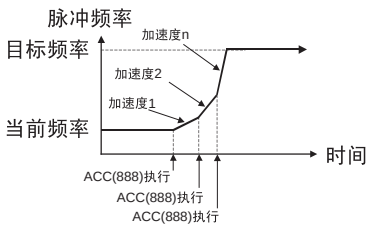
在 PLS2(887) 加速、恒速或减速输出脉冲时，也能执行 ACC(888) 指令。

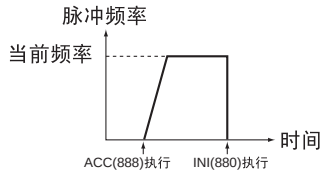
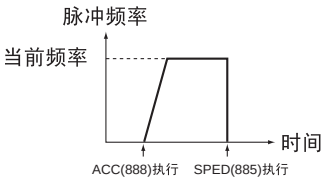
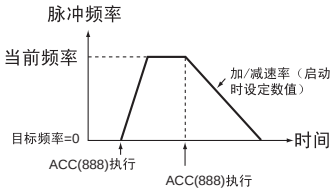
注 在使用 ACC(888) 指令 （连续模式）进行速度控制期间 , 执行与 ACC(888) 指令相同的目标频率和一个 0 加速率的 PLS2(887) 指令，可用于获得一个定距离的中断反馈。

■ 连续模式速度控制

在程序命令停止前，脉冲持续输出。

注 CPU 单元切换为编程 (PROGRAM) 模式时，脉冲输出将立刻停止。

操作	用途	应用	频率改变	说明	操作次序 / 指令
启动脉冲输出	以指定加速度和速度输出	以固定速率增加速度（频率）		输出脉冲并以一个固定的速率改变频率	ACC(888)（连续模式）
改变设置	缓慢改变速度	操作期间缓慢改变速度		从当前频率开始以一个固定的速率改变频率。频率可以加速或减速	ACC(888)或SPED(885)（连续模式） ↓ ACC(888)（连续模式）
		操作时沿多折线改变速度		加速或减速期间改变加 / 减速率	ACC(888)（连续模式） ↓ ACC(888)（连续模式）

操作	用途	应用	频率改变	说明	操作次序 / 指令
停止脉冲输出	停止脉冲输出	立刻停止		立刻停止脉冲输出	ACC(888)（连续模式） ↓ INI(880)（连续模式）
	停止脉冲输出	立刻停止		立刻停止脉冲输出	ACC(888)（连续模式） ↓ SPED(885)（连续模式，目标频率为 0）
	缓慢停止脉冲输出	减速至停止		脉冲减速输出直至停止。 注 如果用 ACC(888) 启动操作，原始加 / 减速率仍然有效。如用 SPED(885) 启动操作，加 / 减速率无效，且脉冲输出将立刻停止。	ACC(888)（连续模式） ↓ ACC(888)（连续模式，目标频率为 0）

■ 独立模式位置控制

当独立模式启动时，脉冲输出将持续直至指定脉冲数全部输出。

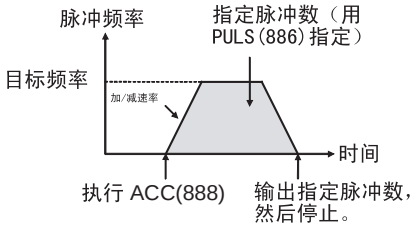
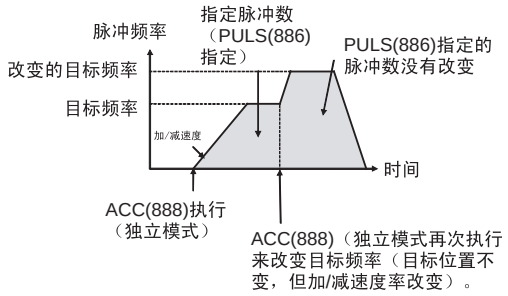
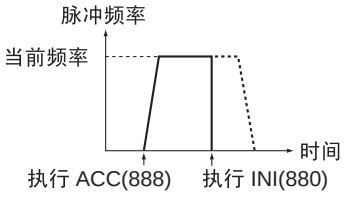
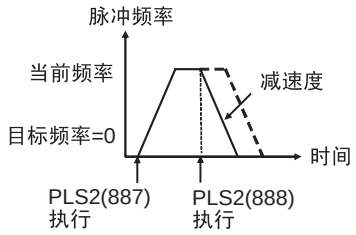
由输出脉冲数和 S 中设置的减速度来计算减速点，并且在该点已经到达时，频率在每个脉冲控制周期 (4ms) 内以 S 中指定减速率减少直至指定脉冲数全部输出，并在该点脉冲输出停止。

注 1. CPU 单元切换为编程 (PRAGRAM) 模式时，脉冲输出立刻停止。

2. 每次输出重新启动时，输出脉冲数必须设定。

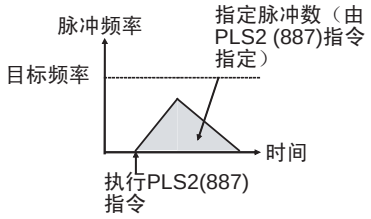
3. 输出脉冲数必须先用 PULS(881) 指令设定。如果不首先执行 PULS(881) 指令， ACC(888) 将不会输出脉冲。

4. 如果用PULS(881)指令把脉冲数设置为绝对值，ACC(888)操作数方向设置会被忽略。

操作	用途	应用	频率改变	说明	操作次序 / 指令
启动脉冲输出	简单梯形控制	带梯形加 / 减速度位置控制（相同的加 / 减速率；无启动速度）。位置控制期间脉冲数不能被改变。		以相同固定的加 / 减速率运行，当指定脉冲数全部输出时，立即停止。（见注） 注 位置控制期间目标位置（指定脉冲数）不能被改变。	PULS(886) ↓ ACC(888) （独立模式）
改变设置	缓慢改变速度（带相同加 / 减速率）	位置控制期间改变目标速度（频率）（加速率 = 减速率）		在位置控制期间，可执行 ACC(888) 指令来改变加 / 减速度率和目标频率。目标位置（指定脉冲数）不变。	PULS(886) ↓ ACC(888) 或 SPED(885) （独立模式） ↓ ACC(888) （独立模式）
停止脉冲输出	停止脉冲输出（脉冲数设置未保存	立刻停止		脉冲输出立刻停止并且把剩余的脉冲数清除。	PULS(886) ↓ ACC(888) （独立模式） ↓ INI(880)
	缓慢停止脉冲输出（脉冲数设置未保存）	减小至停止		脉冲减速输出直至停止。 注 如果 ACC(888) 启动操作，原始加 / 减速率仍然有效。如果 SPED(885) 启动操作，加 / 减速率将无效并且脉冲输出将立刻停止。	PULS(886) ↓ ACC(888) 或 SPED(885) （独立模式） ↓ ACC(888) （独立模式，目标频率为0） PLS2(887) ↓ ACC(888) （独立模式，目标频率为0）

注 三角形控制

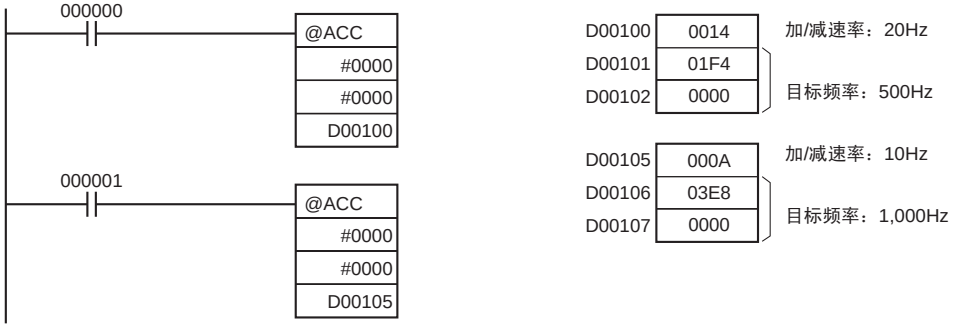
如果指定脉冲数小于所要达到的目标频率的脉冲数并返回 0，其功能将自动减小加速 / 减速时间并执行三角形控制 （仅加速与减速）。错误将不会出现。

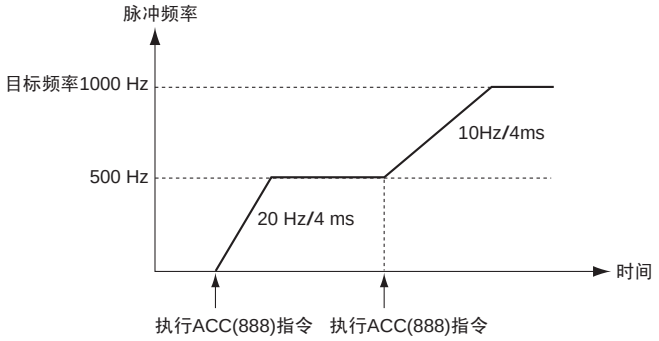


标志

名称	标记	操作
错误标志	ER	P， M 或 S 的值超出指定范围时置 ON。 如果脉冲正输出 , 使用 ORG(889) 指定端口时置 ON。 已正在为 SPED(885)、ACC(888) 或 PLS2(887) 输出脉冲的端口执行 ACC(888) 以实现在独立模式和连续模式间切换时置 ON。 在循环任务中执行控制脉冲输出指令期间，用 ACC(888) 执行绝对脉冲输出时置 ON。 ACC(888) 指令在独立模式下执行绝对脉冲输出，但原点尚未建立时置 ON。

例 在以下程序实例中当 CIO 000000 变为 ON 时，脉冲输出 0 中 ACC(888) 使用 CW/CCW方式沿顺时针方向在连续模式下启动脉冲输出。脉冲输出以20Hz /4ms 速率加速，直至达到目标频率 500Hz。当 CIO 000001 变为 ON 时， ACC(888) 改变为加速率 10Hz/4ms 直至达到目标频率 1,000Hz。





3-21-8 原点搜索：ORG(889) （仅 CJ1M-CPU22/CPU23 适用）

用途

ORG(889) 执行原点搜索或原点返回操作。

这个指令仅由 CJ1M-CPU22/CPU23 CPU 单元支持。

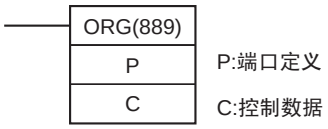
■ **原点搜索**

脉冲采用指定方法输出，以实际地驱动电机和建立基于原点接近输入和原点输入信号的原点。

■ **原点返回**

位置控制系统返回先前建立的原点。

梯形图符号



变化

变化	ON 条件时每次周期执行	ORG(889)
	上升沿微分执行一次	@ORG(889)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

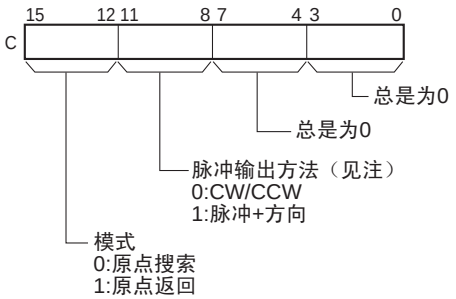
P：端口定义

P 指定将要输出脉冲的端口。

端口定义	端口
0000 hex	脉冲输出 0
0001 hex	脉冲输出 1

C：控制数据

C 的内容决定原点搜索方法。



注 当同时使用脉冲输出0和1时，请使用相同的脉冲输出方法。

操作数规定

区域	P	C
CIO 区	---	---
工作区	---	---
保持位区	---	---

区域	P	C
辅助位区	---	---
定时器区	---	---
计数器区	---	---
DM 区	---	---
无区号 EM 区	---	---
有区号 EM 区	---	---
二进制间接 DM/EM 寻址	---	---
BCD 间接 DM/EM 寻址	---	---
常数	见操作数说明	见操作数说明
数据寄存器	---	---
索引寄存器	---	---
使用索引寄存器的间接寻址	---	---

说明

ORG(889) 使用 C 中指定方法执行 P 中指定端口的原点搜索或原点返回操作。

在执行 ORG(889) 前，下列参数必须在 PC 设置中设定。详细情况参照 *CJ 系列内置式 I/O 操作手册*。

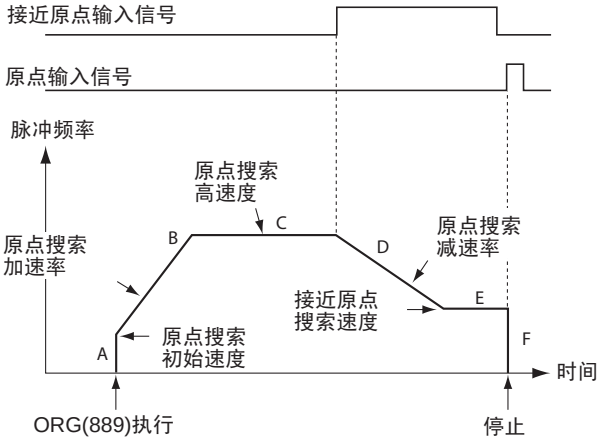
原点搜索	原点返回
原点搜索功能允许 / 禁止	原点搜索 / 返回初始速度
原点搜索操作模式	原点返回目标速度
原点搜索操作设置	原点返回加速率
原点检测方法	原点返回减速率
原点搜索方向设置	
原点搜索 / 返回初始速度	
原点搜索高速度	
接近原点搜索速度	
原点补偿	
原点搜索加速率	
原点搜索减速率	
限止输入信号类型	
接近原点输入信号类型	
原点输入信号类型	

每次执行 ORG(889) 指令时启动原点搜索或原点返回。因此，一般使用微分形式指令 (ORG(889)) 或变为 ON 仅为一次扫描时间的执行条件。

■ **原点搜索** （C 的 12 ～ 15 位 =0）

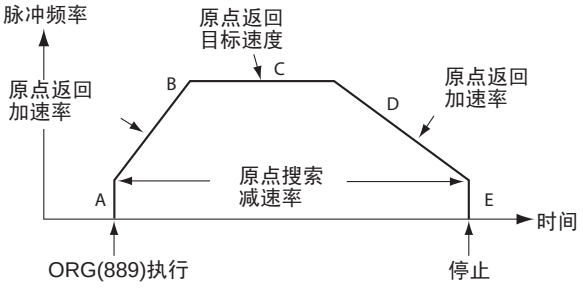
ORG(889) 使用指定的方法以原点初始速度启动脉冲输出 （图中 A）。使用原点加速率把脉冲输出加速到原点搜索高速度 （图中 B）。然后脉冲持续以恒定速度输出，直至接近原点输入信号变为 ON （图中 C），从该点使用原点减速率把脉冲输出减速至接近原点搜索速度 （图中 D）。然后脉冲以恒速输出，直至原点输入信号变 ON （图中 E）。原点输入信号变 ON 时脉冲输出停止 （图中 F）。

当原点搜索操作完成时，错误计数器复位输出将会变 ON。然而以上操作皆依赖于操作模式、原点检测方法以及其它参数。详细情况参照 *CJ 系列内置 I/O 操作手册*。



■ 原点返回 （C 的 12 ～ 15 位 =1）

ORG(889) 采用指定的方式，以原点初始速度 （图中 A）启动输出脉冲。脉冲使用原点返回加速率增加到原点返回目标速度 （图中 B）然后脉冲以恒定速度输出 （图中 C）。减速度点由原点剩余脉冲数、减速率以及到达该点的时间计算，脉冲输出以原点返回减速率减速 （图中 D）直至达到原点返回启动速度为止，在该点脉冲输出停止于原点 （图中 E）。

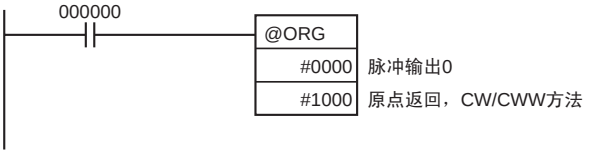


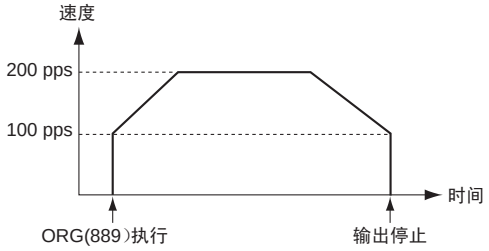
标志

名称	标记	操作
错误标志	ER	P 或 C 的值超出指定范围时置 ON。 已正在为 SPED(885)、ACC(888) 或 PLS2(887) 指令输出脉冲的指定端口执行 ORG(889) 时置 ON。 在循环任务中执行控制脉冲输出指令期间用 ORG(889) 执行中断任务时置 ON。 PC 设置中设定的原点搜索或原点返回参数超出范围时置 ON。 原点搜索高速度小于或等于近原点搜索速度或者近原点搜索速度小于或等于原点搜索初始速度时置 ON。 原点返回目标速度小于或等于原点返回初始速度时置 ON。 原点尚未建立期间试图执行原点返回操作时置 ON。

例

在以下程序实例中，CIO 000000 变为 ON，ORG(889) 对脉冲输出 0 采用 CW/CCW 输出脉冲方法启动原点返回操作。根据 PC 设置，初始速度是 100 pps，目标速度是 200 pps，加 / 减速率为 50 Hz/ms。





PC 设置参数如下：

参数	设置
脉冲输出 0 原点搜索和原点返回的启动速度	0000 0064 : 100 pps
脉冲输出 0 原点返回目标速度	0000 00C8: 200 pps
脉冲输出 0 原点返回加速率	0032 : 50 hex/4 ms
脉冲输出 0 原点返回减速率	0032 :50 hex/4 ms

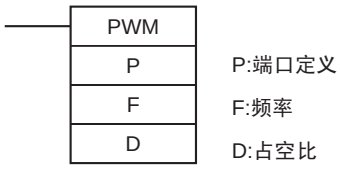
3-21-9 占空比可变的脉冲 PWM(891) （仅 CJ1M-CPU22/CPU23 适用）

用途

PWM(891) 用于在指定端口输出占空比可变的脉冲。

这个指令仅由 CJ1M-CPU22/CPU23 CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每次周期执行	PWM(891)
	上升沿微分执行一次	@PWM(891)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

P：端口定义

P 指定将要脉冲输出的端口。

端口定义	端口
0000 hex	脉冲输出 0
0001 hex	脉冲输出 1

F：频率

F 指定脉冲输出的频率在 0.1 和 6,553.5Hz （以 0.1Hz 为单位， 0001 ～ FFFF Hex）之间。 PWM(891) 实际输出 （占空比 ON +5%/-0%）波形的准确性由于受输出电路的限制，仅适用于 0.1 ～ 1,000.0 Hz 范围。

D：占空比

D 指定脉冲输出的占空比，也就是说，输出 ON 的百分比时间。 D 必须在 0% 到 100% 之间 （0000 ～ 0064 Hex）。

操作数规定

区域	P	F	D
CIO 区	---	CIO 0000 ～ CIO 6143	CIO 0000 ～ CIO 6143
工作区	---	W000 ～ W511	W000 ～ W511
保持位区	---	H000 ～ H511	H000 ～ H511
辅助位区	---	A448 ～ A959	A448 ～ A959
定时器区	---	T0000 ～ T4095	T0000 ～ T4095
计数器区	---	C0000 ～ C4095	C0000 ～ C4095
DM 区	---	D00000 ～ D32767	D00000 ～ D32767
无区号 EM 区	---	---	---
有区号 EM 区	---	---	---
二进制间接 DM/EM 寻址	---	@ D00000 ～ @ D32767	@ D00000 ～ @ D32767
BCD 间接 DM/EM 寻址	---	*D00000 ～ *D32767	*D00000 ～ *D32767
常数	见操作数说明	0000 ～ FFFF hex	0000 ～ 0064 hex
数据寄存器	---	DR0 ～ DR15	DR0 ～ DR15
索引寄存器	---	---	---
使用索引寄存器的间接寻址	---	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

说明

PWM(891) 指令以 D 中指定占空比从 P 中指定端口输出 F 中指定频率的脉冲。占空比脉冲输出期间可执行 PWM(891) 指令来改变占空比输出，而不停止脉冲输出。任何改变频率的试图都将忽略。

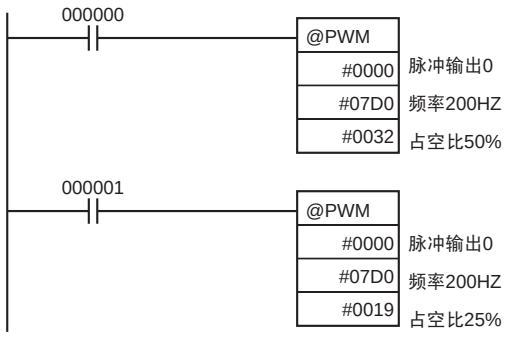
每次执行 PWM(891) 时脉冲启动。因此，一般使用微分形式指令 (@PWM(891)) 或变为 ON 仅为一次扫描时间的执行条件。脉冲将持续输出，直至执行 INI(880) 指令或 CPU 单元切换到编程 (PROGRAM) 模式。

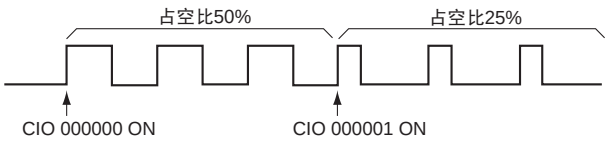
标记

名称	标记	操作
错误标志	ER	如果 P、 F 或 D 超出允许范围置 ON 用 ORG(889) 指令指定脉冲正在输出的口时置 ON 当循环任务中脉冲输出指令正在执行时，在一个中断任务中执行 PWM （891）时置 ON

例

在下例中，当 CIO 000000 为 ON，PWM(891) 从脉冲输出 0 用占空比 50%，启动频率为 200Hz 的脉冲输出。当 CIO 000001 为 ON ， 占空比改为 25%。





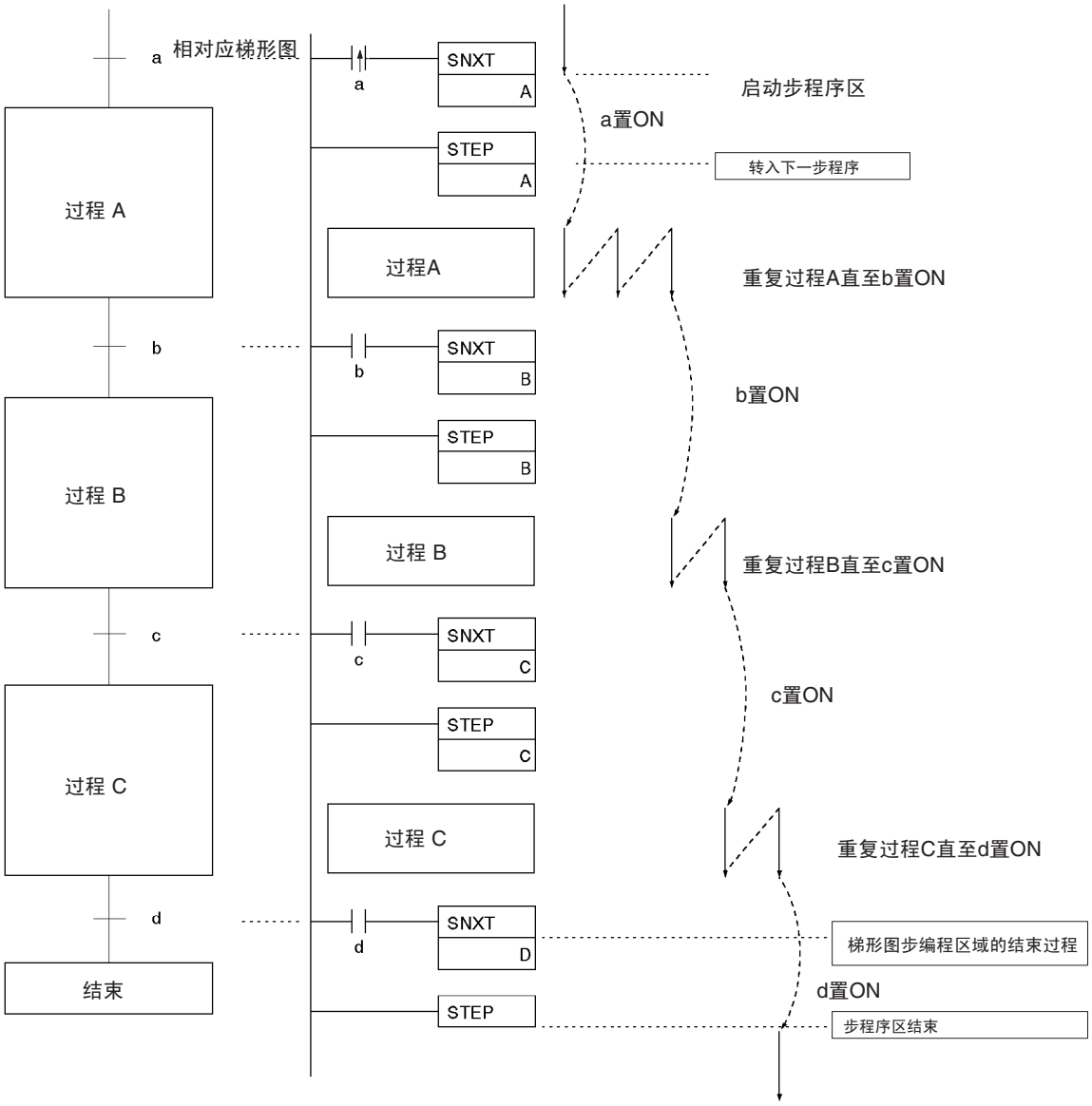
3-22 步指令

本章描述步指令，这些指令用于在一个大型程序的程序段之间设立断点，这样这些程序段可以按单元执行，并在完成后复位。

指令	助记符	函数代码	页
步定义	STEP	008	808
步开始	SNXT	009	808

在CS/CJ系列PLC中，STEP(008)/SNXT(009)指令可以一起使用，建立步程 序。

指令	操作	梯形图
SNXT(009) 步开始	控制程序转入下一步程序。	相对应
STEP(008) 步定义	指示步的开始。重复同一步程序，直至形成转入下一步的条件。	相对应



注　工作位用作 A， B， C 和 D 的控制位。

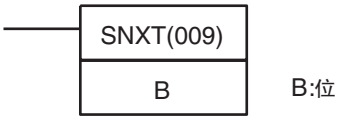
3-22-1 步定义和步开始：STEP(008)/SNXT(009)

用途

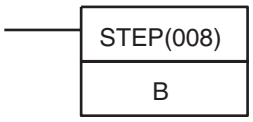
SNXT(009) 就放在 STEP(008) 指令之前，并由置指定控制位为 ON 来控制步程序的执行。如果就在 SNXT(009) 前存在另一个步，它也能将该处理的控制位置 OFF。

STEP(008) 置于每个过程之前并紧随 SNXT(009) 指令之后，它定义了每个过程的开始，并为其制定控制位。 STEP(008) 也可放在最后一个 SNXT(009) 之后的步程序区结尾，以表示步程序区结束， STEP(008) 出现在步程序结尾处时，不带控制位。

梯形图符号



在定义一个步程序的开始，定义控制位如下：



如下所示，在定义步的结尾时不指定控制位。

变化

变化	ON 条件时每个循环执行	STEP(008)/ SNXT(009)
	上升沿微分执行一次	不支持
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
不允许	OK	不允许	不允许

操作数规定

区域	B
CIO 区	---
工作区	W00000 ～ W51115
保持位区	---
辅助位区	---
定时器区	---
计数器区	---
DM 区	---
无区号 EM 区	---
有区号 EM 区	---
二进制间接 DM/EM 寻址	---
BCD 间接 DM/EM 寻址	---
常数	---
数据寄存器	---
索引寄存器	---
使用索引寄存器 的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

说明

SNXT (009)

SNXT (009) 可通过以下三种方式使用：

1,2,3...

1. 启动步程序执行。
2. 转入下一个步控制位。
3. 结束步程序执行。

步程序区从第一个 STEP(008) 指令（它通常带有一个控制位）开始，到最后一个 STEP(008) 指令 （不带控制位）结束。

启动步程序

SNXT (009) 位于步程序的始端以启动步程序。它将下一个 STEP(008) 指定的 B 的控制位置 ON，并转入执行步 B （STEP(008) B 之后的所有指令）。必须对 SNXT (009) 使用一个微分执行条件来启动步程序区执行，或使步执行只能持续一个周期的执行条件。

转入下一步

步程序区中间的 SNXT(009) 指令用于转入下一步，它将当前控制位置 OFF，并将下一个控制位 B 置 ON，从而为下一步启动步 B （STEP(008) B 之后的所有指令）。

结束步程序区

当 SNXT(009) 出现在步程序区的结尾时，它结束步执行，并将当前控制位置 OFF，为 B 指定的控制位是一个虚位，该位也会变 ON，因此一定要选择一个不会产生问题的位。

STEP (008)

STEP(008) 根据其位置和是否已指定了一个控制位，有两种使用方法

1,2,3... 1. 启动一个指定步。
2. 结束步程序区 （即步执行）。

启动一个步

STEP(008) 位于每一步的始端，并带有一个操作数 B，作为该步的控制位。

控制位 B 有 SNXT 置 ON，并从紧随 STEP(008) 之后的指令开始执行步程序中的指令。在步开始执行时，也将 A20010 （步标记）置 ON。

执行完第一个周期后，继续该步执行直至变换步条件形成，即直至 SNXT(009) 将下一个 STEP(008) 的控制位置 ON。

当SNXT(009) 将一个步的控制位置ON时，当前指令的控制位B被复位（置OFF），由位 B 控制的步程序被互锁。

根据控制位 B 的 ON/OFF 状态改变对步程序中输出和指令的处理。（控制位的状态由 SNXT(009) 控制）。当控制位 B 被置 OFF 时，步程序中的指令被复位且互锁。参见下面的表。

控制位状态	处理
ON	执行步程序中的指令。
ON→OFF	如下一个表所示步程序中的位和指令是互锁的。
OFF	步程序中所有指令被作为 NOP 处理。

互锁状态 (IL)

指令输出		状态
OUT， OUT NOT 指定位		全 OFF
定时器指令（TIM,TIMX(551),TIMH(015), TIMHX(551),TMHH(540), TIMHHX(552), TIML(542), 和 TIMLX(553))	PV	0000 hex （复位）
	完成标记	OFF （复位）
其它指令指定的位或字 （见注）		保持前面的状态 （但不执行这些指令）

注 指明诸如函数代码 TTIM(087)，TTIMX(555)，MTIM(543),MTIMX(554),SET,REST，CNT，CNTX(546)，CNTR(012)，CNTRX(548)，SFT(010) 和 KEEP(011) 之类所有其它指令。

STEP(008) 指令必须置于每一步的始端。STEP(008) 被放在一个部区域的始端来定义步的开始。

始止步程序区

当在 SNXT(009) 指令之前的控制位被置 OFF 时，不带操作数的 STEP(008) 被置于步程序区的末端来定义步程序的终止， SNXT(009) 终止步执行。

标记：STEP(008)

名称	标记	操作
错误标志	ER	指定位 B 不在 WR 区中时置 ON。 在一个中断程序中使用了 STEP(008) 时置 ON。 其他情况置 OFF。

标记：SNXT(009)

名称	标记	操作
错误标志	ER	指定位 B 不在 WR 区中时置 ON。 在一个中断程序中使用了 SNXT(009) 时置 ON。 其他情况置 OFF。

注意

STEP(008)/ SNXT(009) 的控制位 B 必须在工作区。

STEP(008)/ SNXT(009) 的控制位不能用于梯形图的其它地方。若同一位用了两次，将产生重复位错误。

如果 SBS(091) 被用来从一个步程序中调用子程序，当控制位置 OFF 时，子程序输出和指令将不被互锁。

一段步程序中的控制位必须是连续的并取自同一字。

SNXT(009) 只能执行一次，即在上升沿执行条件下。

在步程序区末端输入 SNXT(009) 并确保控制位是工作区中的虚位。如果一个步的控制位被用在步程序区的最后一个 SNXT(009) 中， SNXT(009) 被执行时将启动相应步程序。

如果 SNXT(009) 或 STEP(008) 指定的操作数不在工作区内或者步程序未置入一个循环任务中，将产生一个错误，并将错误标记置 ON。

执行 STEP(008) 时， A20012（步标记）被置 ON 一个循环。该标记可用于在启动步执行时进行一次初始化。

步程序区的使用条件 （STEP B ～ STEP）

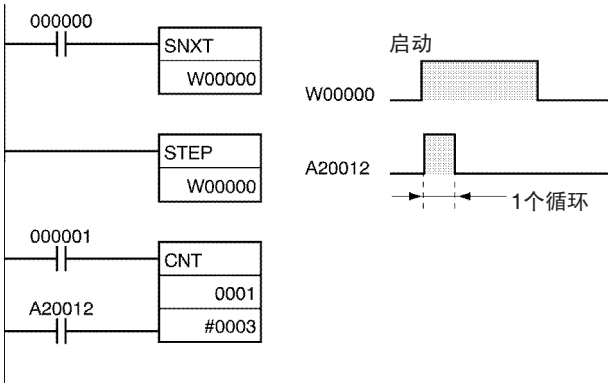
STEP(008) 和 SNXT(009) 不能在子程序、中断程序或块程序中使用。

不能在同一循环中执行两个步。

不能用于步程序中的指令

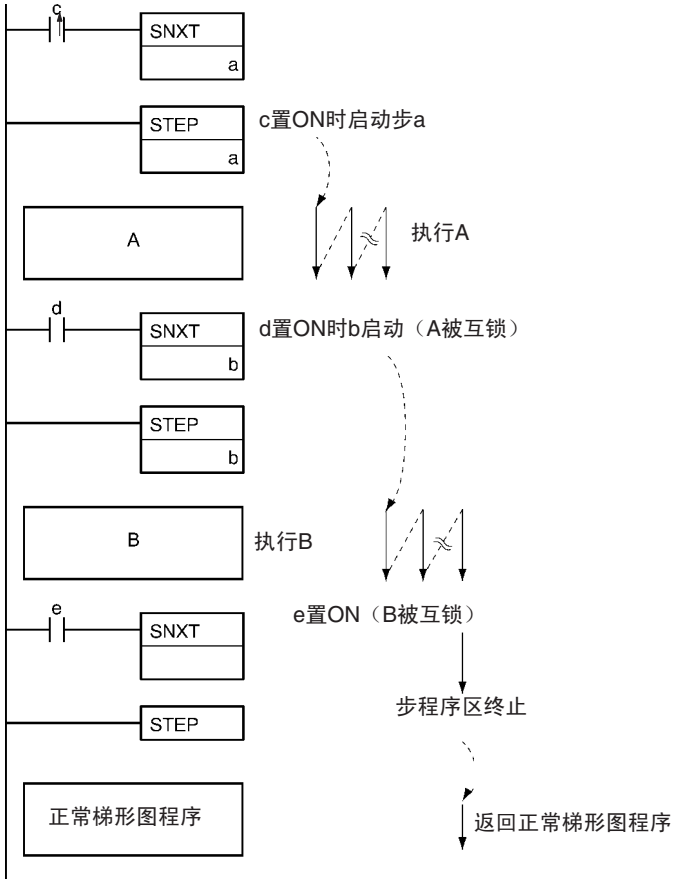
在下表中列明了不能用于步程序中的指令。

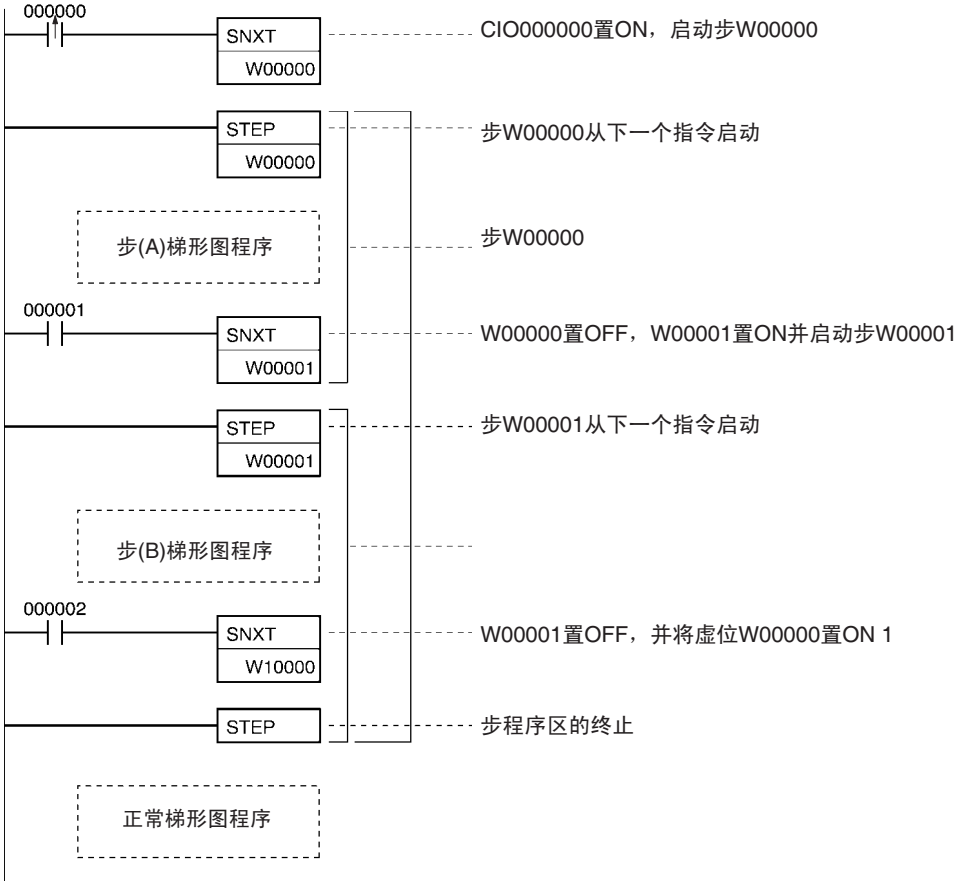
函数	助 记 符	名称
顺序控制指令	END(001)	结束
	IL(002)	互锁
	ILC(003	互锁清除
	JMP(004)	跳转
	JME(005)	跳转结束
	CJP(510)	条件跳转
	CJPN(511)	条件跳转非
	JMP0(515)	多路跳转
	JME0(516)	多路跳转结束
子程序指令	SBN(092)	子程序入口
	RET(093)	子程序返回



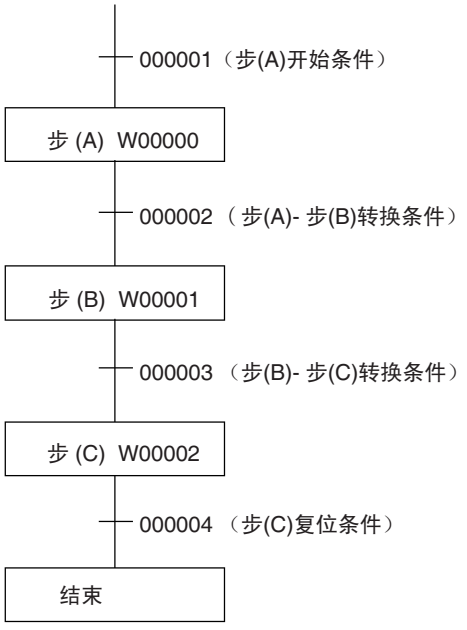
相关位

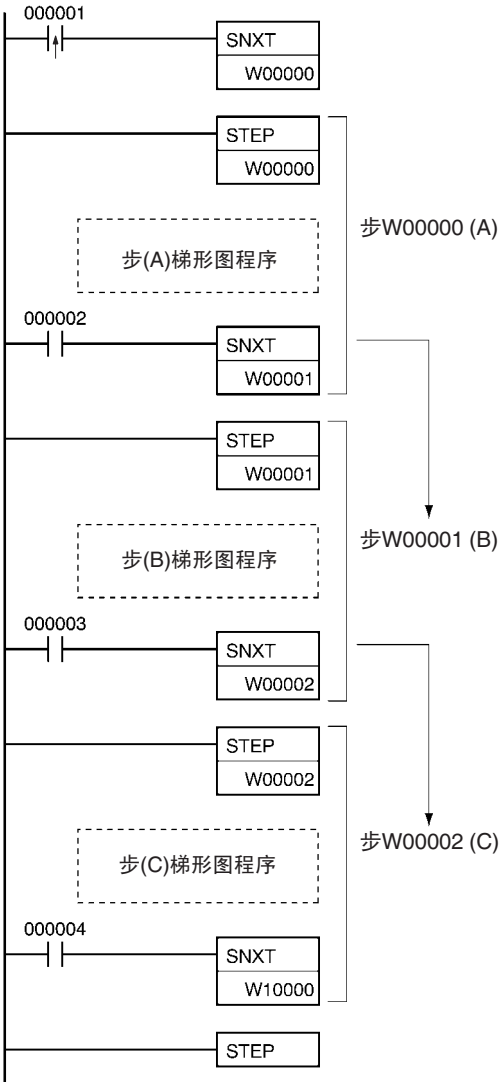
名称	地址	详细说明
步标记	A20012	当步程序用 STEP(008) 启动时置 ON 一个循环。可用于定时器复位，并在开始一个新步时执行其它过程。


c
SNXT
a
STEP
a
c置ON时启动步a
A
执行A
d
SNXT
b
d置ON时b启动（A被互锁）
STEP
b
B
执行B
e
SNXT
e置ON（B被互锁）
步程序区终止
STEP
正常梯形图程序
返回正常梯形图程序

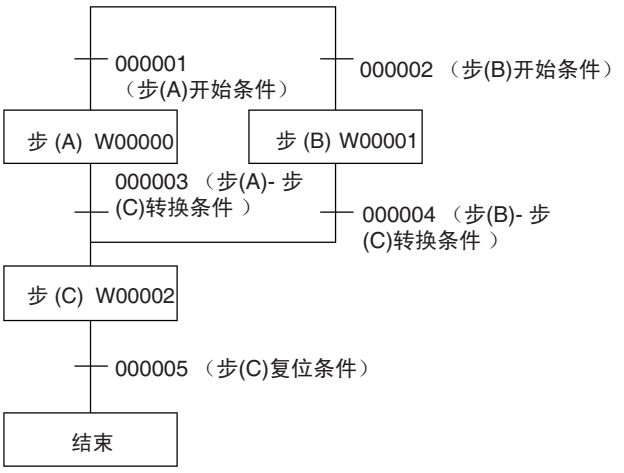

000000
SNXT
W00000
CIO000000置ON，启动步W00000
STEP
W00000
步W00000从下一个指令启动
步(A)梯形图程序
步W00000
000001
SNXT
W00001
W00000置OFF，W00001置ON并启动步W00001
STEP
W00001
步W00001从下一个指令启动
步(B)梯形图程序
000002
SNXT
W10000
W00001置OFF，并将虚位W00000置ON 1
STEP
步程序区的终止
正常梯形图程序

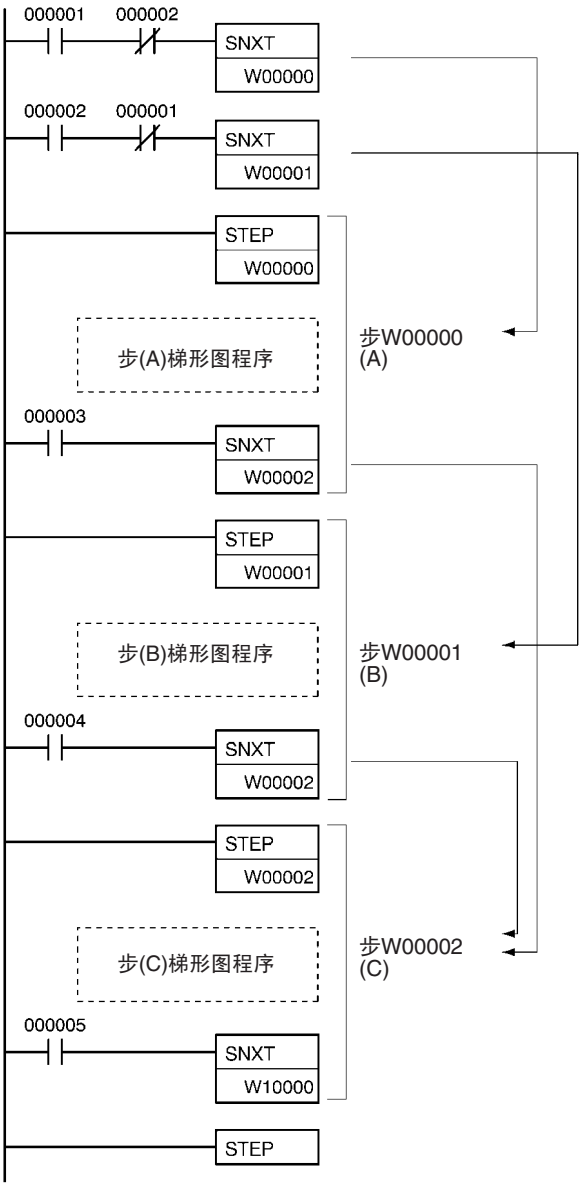
例 顺序控制


000001（步(A)开始条件）
步 (A) W00000
000002（步(A)- 步(B)转换条件）
步 (B) W00001
000003（步(B)- 步(C)转换条件）
步 (C) W00002
000004（步(C)复位条件）
结束

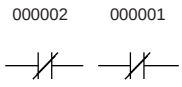

000001
SNXT
W00000
STEP
W00000
步W00000 (A)
步(A)梯形图程序
000002
SNXT
W00001
STEP
W00001
步W00001 (B)
步(B)梯形图程序
000003
SNXT
W00002
STEP
W00002
步W00002 (C)
步(C)梯形图程序
000004
SNXT
W10000
STEP

分支控制


000001
（步(A)开始条件）
000002 （步(B)开始条件）
步 (A) W00000
步 (B) W00001
000003 （步(A)- 步(C)转换条件 ）
000004 （步(B)- 步(C)转换条件 ）
步 (C) W00002
000005 （步(C)复位条件）
结束

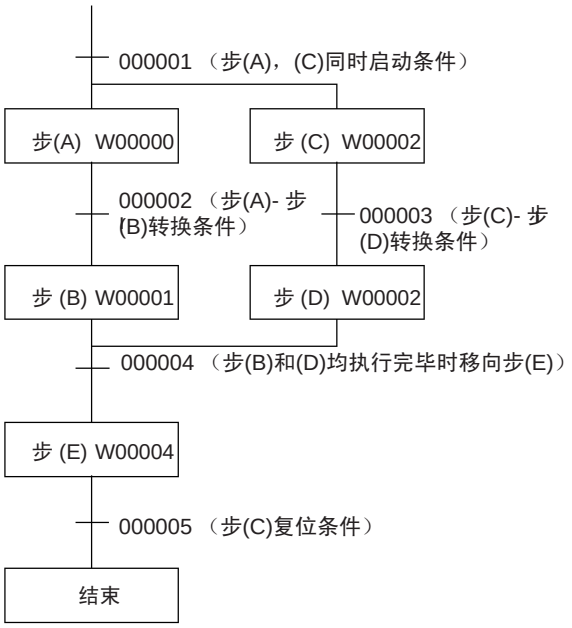


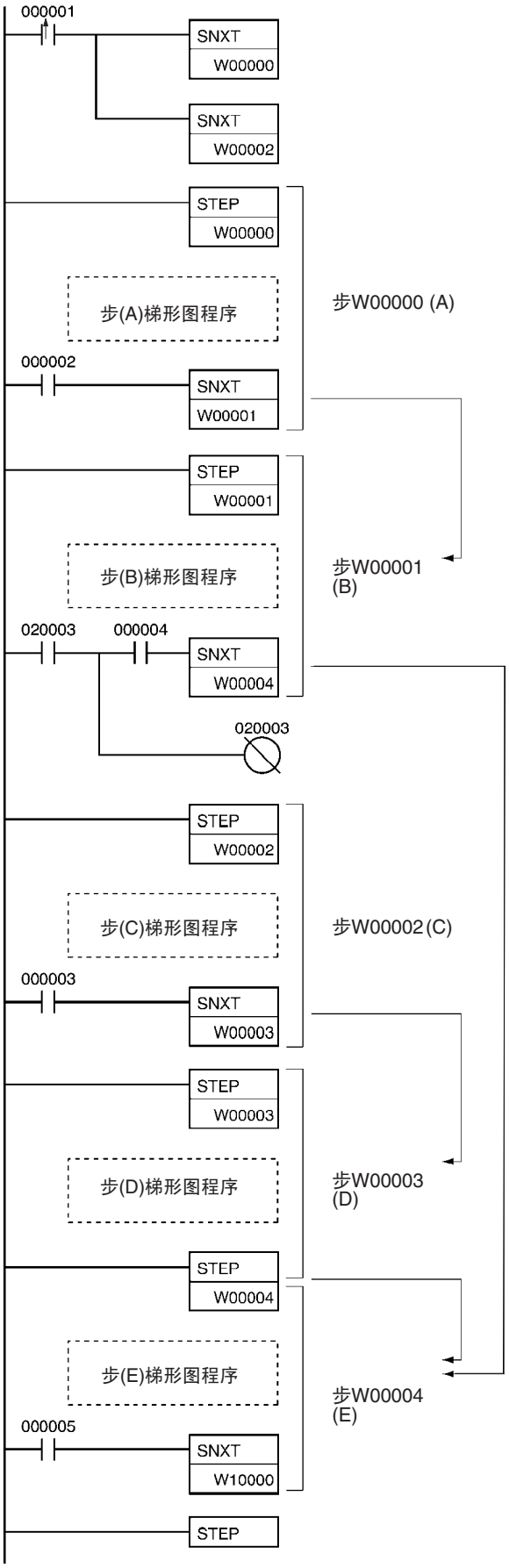
在上述程序中不能同时执行步 A 和步 B。若需同时执行 A 和 B，将下列执行条件删除。



注 在上例中，对 W00002 执行 SNXT(009) 时，即使同一控制位被使用了两次，分支也会移至下一步。CX-Programmer 检测不认为这是一个错误。步梯形图程序的重复位错误只发生在下一个步指令的控制位被正常梯形图使用时。

并行控制


000001 （步(A)，(C)同时启动条件）
步(A) W00000
步 (C) W00002
000002 （步(A)- 步(B)转换条件）
000003 （步(C)- 步(D)转换条件）
步 (B) W00001
步 (D) W00002
000004 （步(B)和(D)均执行完毕时移向步(E)）
步 (E) W00004
000005 （步(C)复位条件）
结束

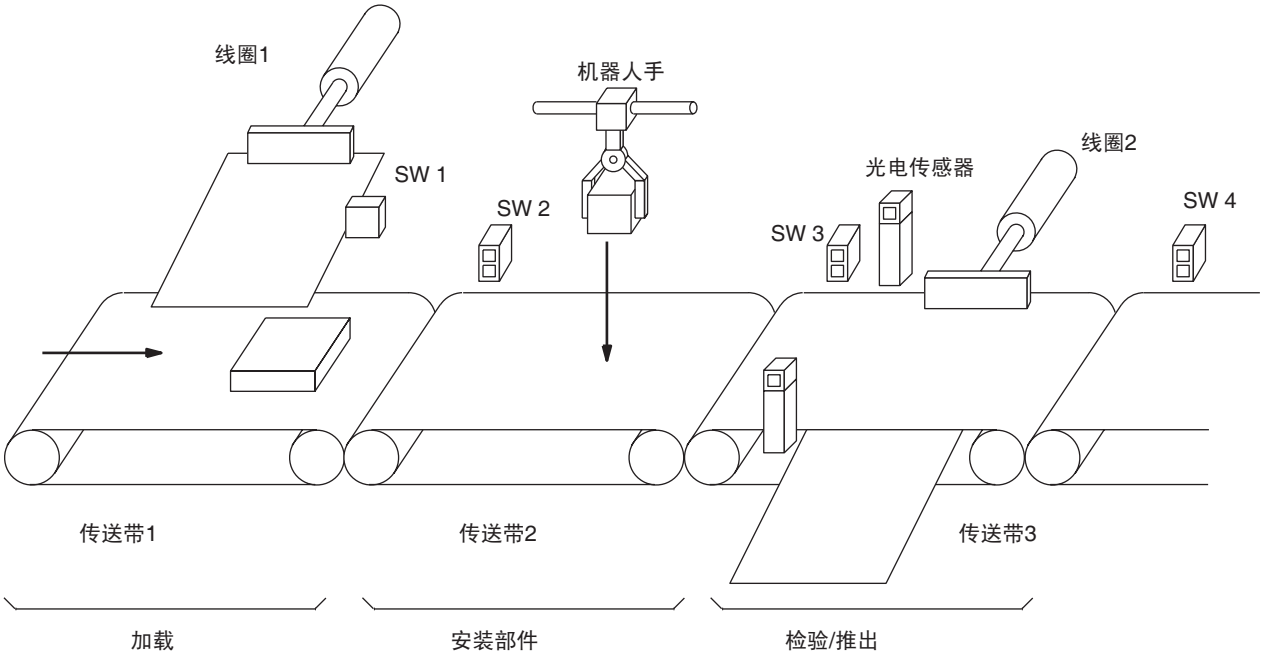

000001
SNXT
W00000
SNXT
W00002
STEP
W00000
步(A)梯形图程序
步W00000 (A)
000002
SNXT
W00001
STEP
W00001
步(B)梯形图程序
步W00001
(B)
020003
000004
SNXT
W00004
020003
STEP
W00002
步(C)梯形图程序
步W00002(C)
000003
SNXT
W00003
STEP
W00003
步(D)梯形图程序
步W00003
(D)
STEP
W00004
步(E)梯形图程序
步W00004
(E)
000005
SNXT
W10000
STEP

应用举例

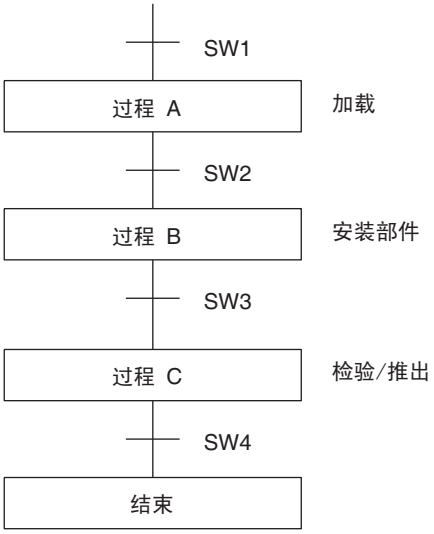
下面 3 个例子说明了步程序中三种不同执行控制的类型：*例 1* 是顺序执行；*例 2* 是分支执行；*例 3* 是并行执行。

例 1：顺序执行

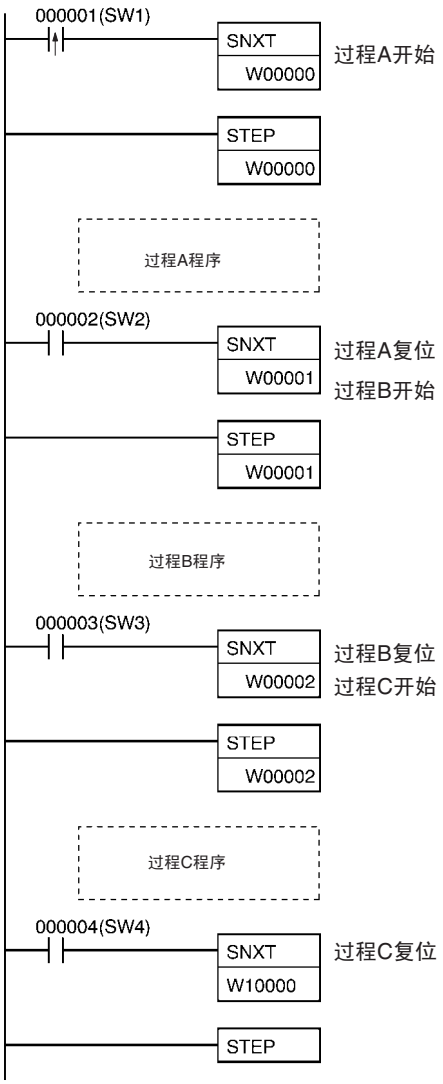
下列过程要求 3 个步骤：加载、安装部件和检验推出，3 个步骤按顺序执行，在开始下一个过程前上一过程被复位。当过程开始或结束时，设置在不同位置的一组传感器（SW1、SW2、SW3 和 SW4）发出信号。



下图表示流程和用于控制执行的开关。



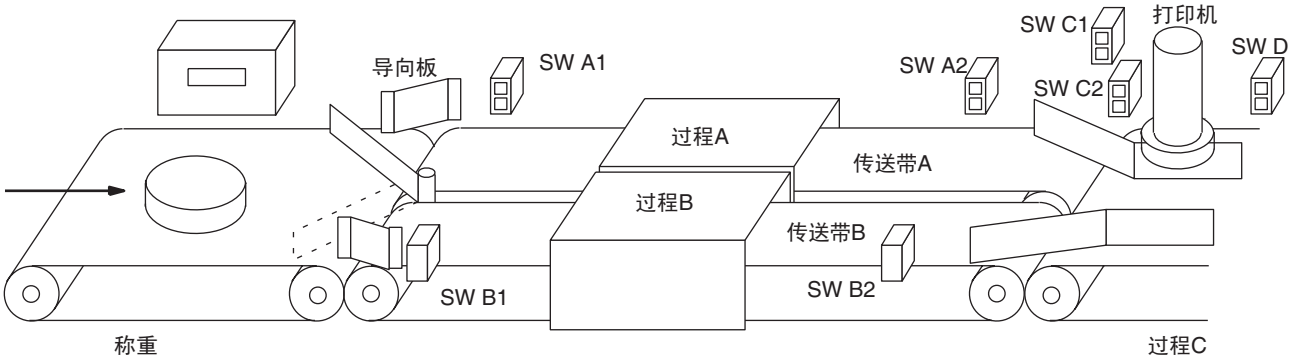
该控制过程的程序如下，它使用步编程的最基本类型：每一步用唯一的 SNXT(009) 来完成并开始下一步。只有当表示前一步完成的开关接通后，下一步才开始



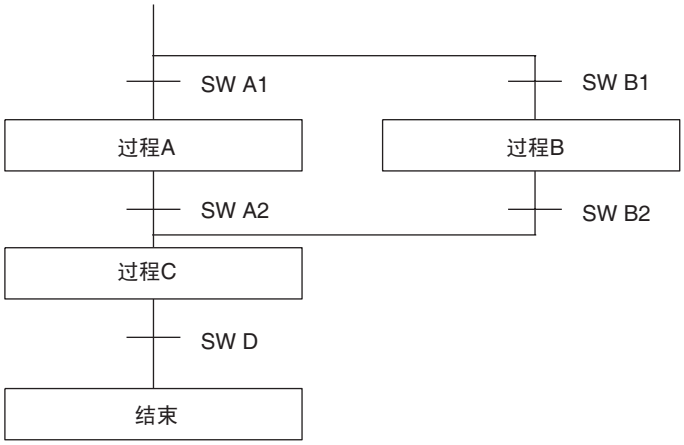
地址	指令	操作
000000	@LD	000001
000001	SNXT(009)	W00000
000002	STEP(008)	W00000
	过程A	
000100	LD	000002
000101	SNXT(009)	W00001
000102	STEP(008)	W00001
	过程B	
000100	LD	000003
000101	SNXT(009)	W00002
000102	STEP(008)	W00002
	过程C	
000200	LD	000004
000201	SNXT(009)	W00003
000202	STEP(008)	W00003

例 2：
分支执行

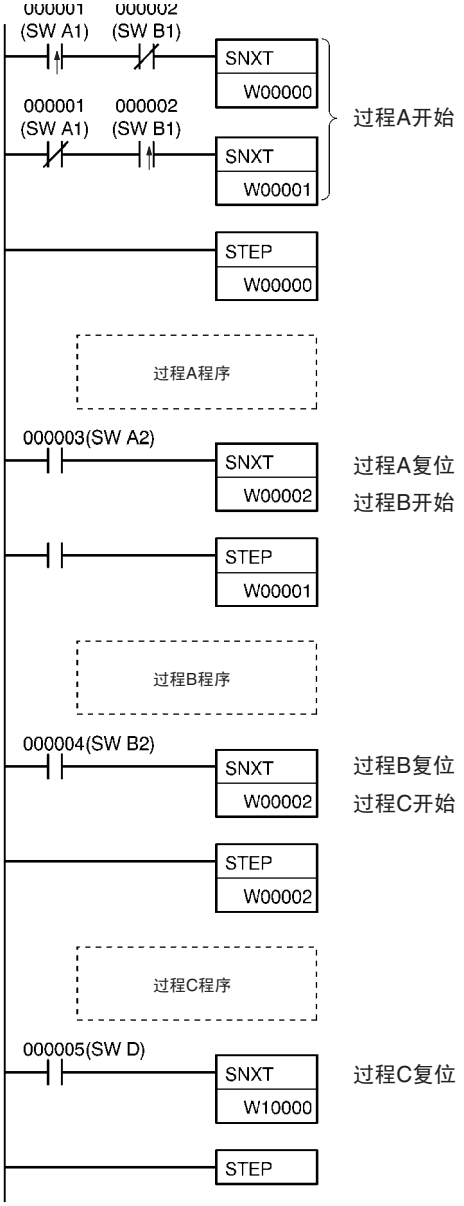
下面的过程要求某种产品在打印前，按照其重量在二种处理方式中选一种。但无论做哪种处理，均需打印。当过程开始和结束时，位于不同位置的各种传感器将发出信号。



下图表示执行控制的流程及用于执行控制的开关。这里用开关 SW A1 和 SW A2 的状态决定过程 A 或过程 B。



下图是该控制过程的程序，用两条 SNX(009) 指令开始过程 A 或 B。由于 CIO000001(SWA1) 和 CIO000002(SWA2) 的编程方法，只有一个具有 ON 执行条件而执行，启动过程 A 或过程 B。A 或 B 两个过程用同一条 SNXT(009) 指令来结束，并开始过程 C。



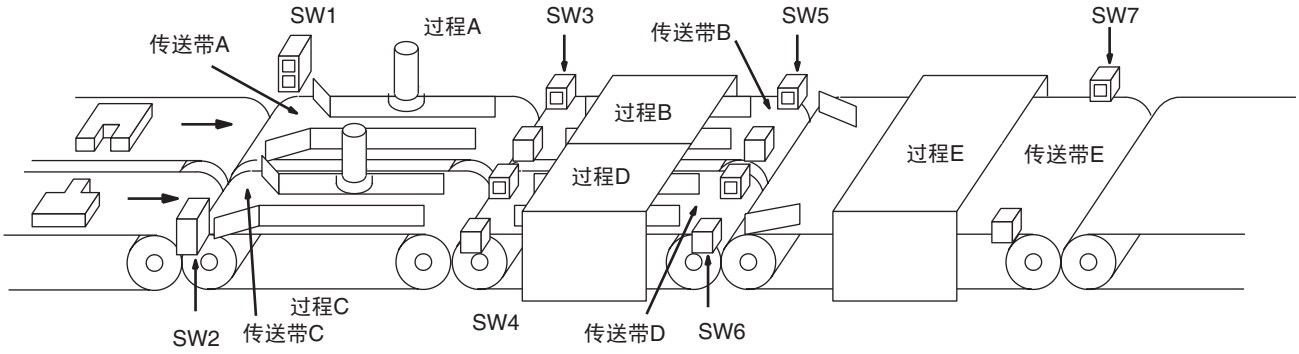
地址	指令	操作
000000	@LD	000001
000001	AND NOT	000002
000002	SNXT(009)	010000
000003	LD NOT	000001
000004	@AND	000002
000005	SNXT(009)	010001
000006	STEP(008)	010000
过程A		
000100	LD	000003
000101	SNXT(009)	010002
000102	STEP(008)	010001
过程B		
000100	LD	000004
000101	SNXT(009)	010002
000102	STEP(008)	010002
过程C		
000200	LD	000005
000201	SNXT(009)	024614
000202	STEP(008)	---

注 在上面的程序中，CIO010002用于两个 SNXT(009)指令中，在检查程序时不会产生重复错误。

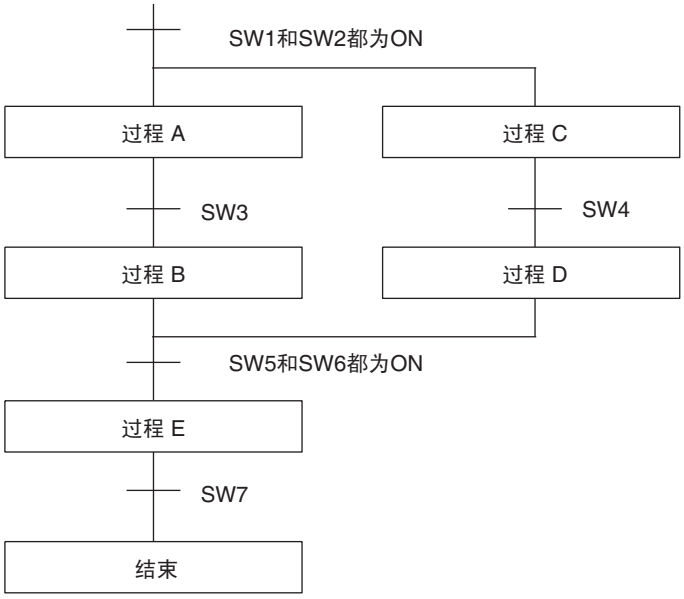
例 3：

并行执行

下面的过程要求某种产品的两个零件同时进行两种不同加工处理，然后在第五道工序中会合在一起。当过程开始或结束时，一组位于不同位置的传感器将发出信号。

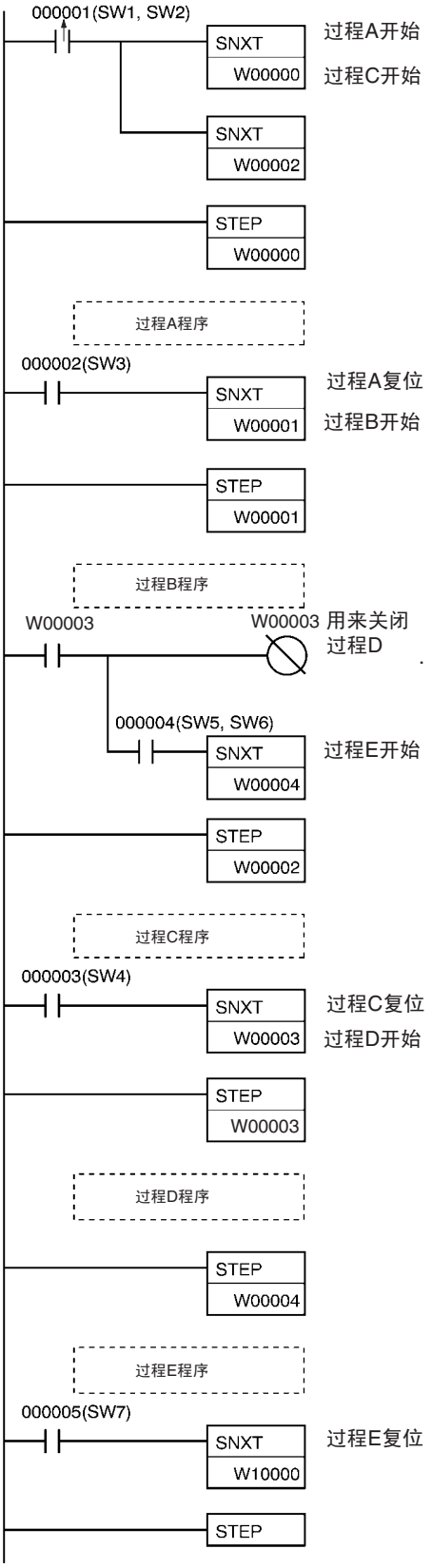


下图表示上面控制过程的流程及用作执行控制的开关。这里过程 A 和过程 C 一起开始，过程 A 结束时，过程 B 开始；过程 C 结束时，过程 D 开始。当过程 B 和过程 D 都结束时，过程 E 开始。



该生产过程的程序如下图所示。程序从启动过程 A 和过程 C 的两条 SNXT(009) 指令开始，它们从同一指令行分开，同时执行，开始过程 A 和过程 C 的步。当过程 A 和过程 C 都结束时，过程 B 和过程 D 立即开始。

当过程 B 和过程 D 都结束时 （即 SW5 和 SW6 都为 ON 时），在过程 B 的程序结束时，过程 B 和过程 D 一起被 SNXT(009) 指令复位。虽然在过程 D 结束时没有 SNXT(009)，但它的控制位通过执行 SNXT(009)W00004 指令而关闭为 OFF，这是因为该步中的 W00003 的输出是由 SNXT(009)W00004 指令进行复位的。即当执行 SNXT(009)W00004 指令时，W00003 被关闭，这样过程 B 直接复位，而过程 D 在过程 E 执行前被间接复位。



地址	指令	操作数
000000	@LD	000001
000001	SNXT(009)	W00000
000002	SNXT(009)	W00002
000003	STEP(008)	W00000
	过程A	
000100	LD	000002
000101	SNXT(009)	W00001
000102	STEP(008)	W00001
	过程B	
000100	LD	000003
000101	OUT	W00003
000101	AND	000004
000101	SNXT(009)	W00004
000102	STEP(008)	W00002
	过程C	
000200	LD	000003
000201	SNXT(009)	W00003
000202	STEP(008)	W00003
	过程D	
000300	STEP(008)	W00004
	过程E	
000400	LD	000005
000401	SNXT(009)	024613
000402	STEP(008)	---

3-23 基本 I/O 单元指令

本节介绍了用于 I/O 单元的指令。

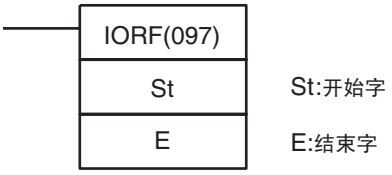
指令	助记符	函数代码	页
I/O 刷新	IORF	097	819
7 段译码器	SDEC	078	822
智能 I/O 单元读	IORD	222	825
智能 I/O 单元写	IOWR	223	834

3-23-1 I/O 单元刷新：IORF(097)

用途

刷新指定 I/O 字。

梯形图符号



变化

变化	ON 条件时每个循环执行	IORF(097)
	上升沿微分执行一次	@IORF(097)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

St：开始字
CIO 0000 ～ CIO 0999 （I/O 位区）或
CIO 2000 ～ CIO 2959 （特殊 I/O 单元位区）
E：结束字
CIO 0000 ～ CIO 0999 （I/O 位区）或
CIO 2000 ～ CIO 2959 （特殊 I/O 单元位区）

注 St 和 E 必须在同一存储区内。

操作数规定

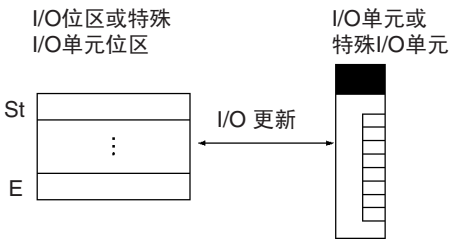
区域	St	E
CIO 区	CIO 0000 ～ CIO 0999	CIO 2000 ～ CIO 2959
辅助位区	---	
保持位区	---	
专用位区	---	
定时器区	---	
计数器区	---	
DM 区	---	
无区号 EM 区	---	
有区号 EM 区	---	
二进制间接 DM/EM 寻址	---	

区域	St	E
BCD 码间接 DM/EM 寻址	---	
常数	---	
数据寄存器	---	
索引寄存器	---	
使用索引寄存器 的间接寻址	,IR0 ～ 15 –2048 ～ +2047, IR0 ～ 15 DR0 ～ 15, IR0 ～ 15, IR0 ～ 15+(++) ,–(– –) IR0 ～ 15	---

说明

IORF(097) 刷新 St 和 E 之间的所有的 I/O 字。IORF(097) 用于刷新安装在 CPU 机架或扩充机架上的基本 I/O 单元或特殊 I/O 单元。IORF(097) 不能用于同时刷新两个区中的字（即用同一指令）。基本 I/O 单元分配从 CIO 0000 ～ CIO 0999 之间的字，特殊 I/O 单元分配从 CIO 2000 ～ CIO 2959 之间的字。

当指定刷新特殊 I/O 单元位区中的字时，只要分配给该单元的 10 个字中的第一个字被包含在指定字范围中，所有的 10 个字都将被刷新。



如果存在于 St 和 E 之间的字没有安装单元，那么将不对这些字进行任何操作，只有分配给单元的字才能被刷新。

可以用同一指令刷新C200H特殊I/O单元和CS特殊I/O单元。（仅适于CS系列）

所有分配给 C200H 组 2 高密度 I/O 单元的字必须同时刷新。如果单元的第一个分配字在特殊 I/O 字范围内，那么该单元 I/O 字将被刷新。（如果开始字在第一个分配字之后，单元字不被刷新，但即使结束字在最后一个分配字之前，它们也将被刷新）。（仅适于 CS 系列）

IORF(097) 可用在中断任务中，允许对中断任务中的特殊 I/O 字刷新以高速响应。（参见注意）

适用单元

以下单元可用 IORF(097) 刷新。这些单元只有在 CPU 机架或扩展机架上时才能被刷新，在从机架上时不能被刷新。

CS 系列基本 I/O 单元，C200H 基本 I/O 单元（仅适于 CS 系列），C200H 组 -2 高密度 I/O 单元（仅适于 CS 系列），CJ 系列基本 I/O 单元和特殊 I/O 单元（包括高密度单元，这些单元的所有分配字都能被刷新）。

注 可以被 IORF(097) 刷新的单元不需要与可被立即刷新功能 (!) 刷新的单元相同。

标志

名称	标记	操作
错误标志	ER	St 大于 E 时置 ON 若地址超出了 St 的间接 IR 规定的范围置 ON。 若地址超出了 E 的间接 IR 规定的范围置 ON。 St 和 E 在不同内存区域内时置 ON。 对于 CS1D CPU 单元：若主 CPU 单元和备用 CPU 单元不同步置 ON。 其它情况置 OFF。

注意

若 I/O 位区 (CIO 0000 ～ CIO 0999) 和特殊 I/O 单元位区 (CIO2000 ～ CIO2959) 的字由同一指令指定，将产生一错误。

I/O 刷新不能在产生了 I/O 表错误的单元中运行。（仅适于 CS 系列）

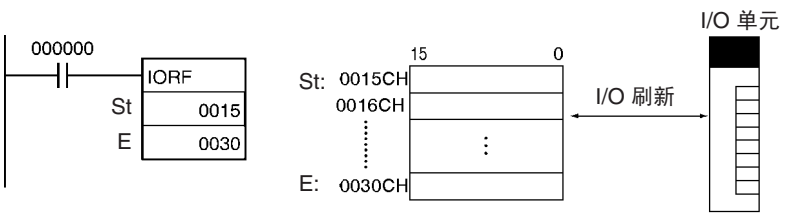
若在 I/O 刷新时产生了 I/O 总线错误，IORF(097) 进行的 I/O 刷新将中途停止。

在一个中断任务中使用 IORF(097) 时，一定要在 PLC 设定中禁止特殊 I/O 单元循环刷新。若特殊 I/O 单元刷新允许，且由 IORF(097) 再次执行刷新，则会产生一个非致命的重复刷新错误，中断任务错误标记 (A40213) 置 ON。

例

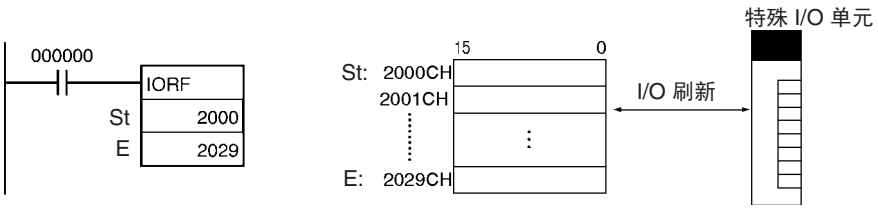
刷新 I/O 位区的字

下列说明了当 CIO 000000 置 ON 时，如何刷新从 CIO 0015 ～ CIO 0030 中的 16 个字。



刷新特殊 I/O 单元位区中的字

下列说明了当 CIO 000000 置 ON 时，如何刷新从 CIO 2000 ～ CIO 2029 中的 30 个字。

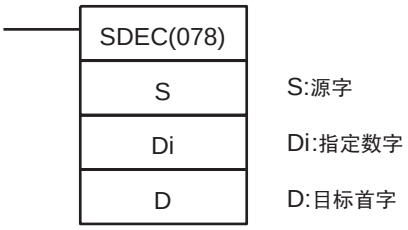


3-23-2 7 段译码器：SDEC(078)

用途

将指定数字的十六进制内容转换为 8 位， 7 段显示代码，并将其放入指定目标字的高 8 位或低 8 位中。

梯形图符号



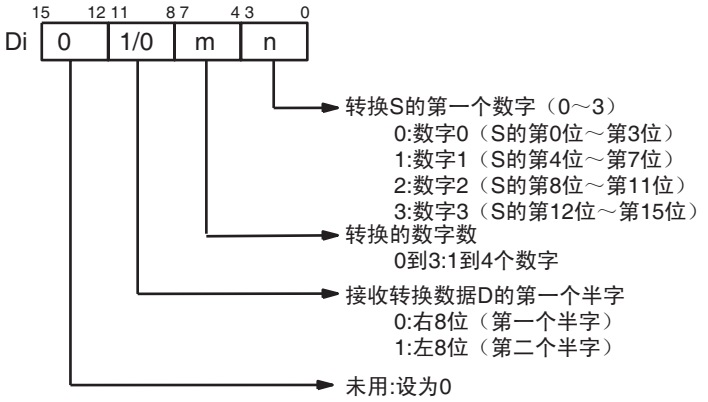
变化

变化	ON 条件时每个循环执行	SDEC(078)
	上升沿微分执行一次	@SDEC(078)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数：数字指示器



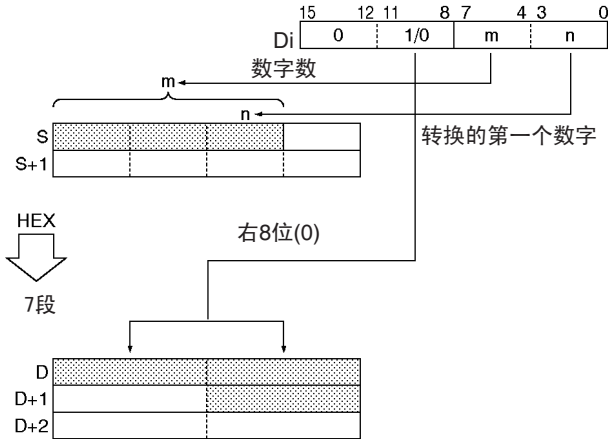
操作数规定

区域	S	Di	D
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		

区域	S	Di	D
二进制间接 DM/EM 寻址	@D00000 ～ @D32767 @E00000 ～ @E32767 @En_00000 ～ @En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---	仅指定值	---
数据寄存器	DR0 ～ DR15		---
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

SDEC(078) 将 S 指定的数据作为 4 个 16 进制数字，把由 Di（第一个数字和数字数）指定的 S 中的数字转换位 7 段数据，并将结果按 Di 位中的指定输出到 D。



标记

名称	标记	操作
错误标志	ER	若 Di 中的设置不在指定范围内时置 ON。 其它情况置 OFF。

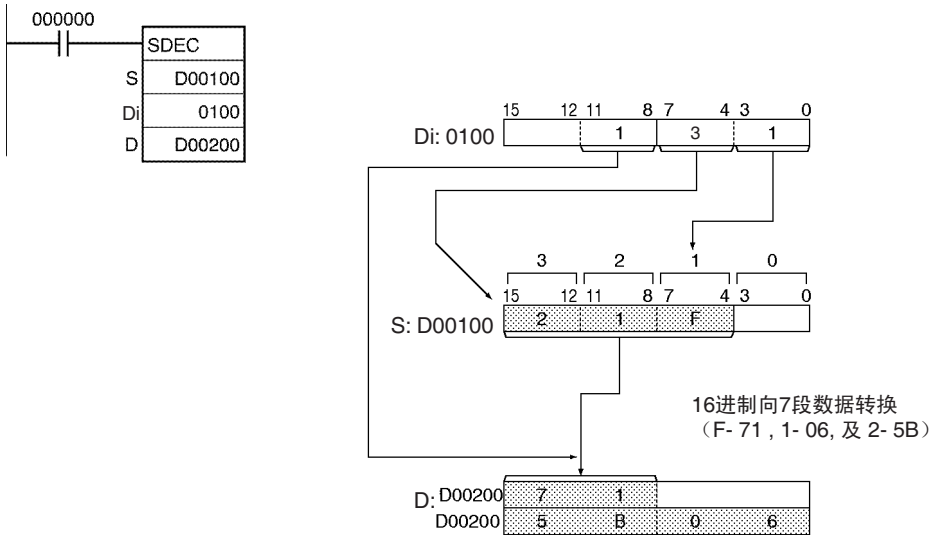
注意

若 Di 指定了多个转换数字，则数字转换按向高位数字的次序进行。数字 0 是数字 3 的下一个数字。

结果按从指定的部分向高地址字的顺序存入 D。如果目标字只有一个字节接收转换数据，另一字节保持不变。

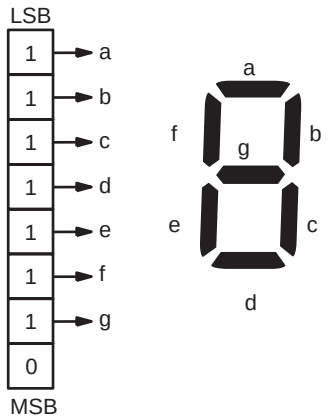
例

当下例中当 CIO 000000 置 ON 时，从 D00 100 中的第 1 个数字开始的 3 个数字的内容将被从 16 进制数据转换成 7 段数据，并将结果输出到 D00 200 的高字节和 D00 201 的两个字节。转换的字节和输出字节的规定在 CIO 1000 中。



7 段数据　　下表显示了从一个 16 进制数字 （4 位）转换位 7 段代码 （8 位）数据。

原始数据					转换码 （段）									显示原始数据
数据	位				–	g	f	e	d	c	b	a	Hex	
0	0	0	0	0	0	0	1	1	1	1	1	1	3F	0
1	0	0	0	1	0	0	0	0	0	1	1	0	06	1
2	0	0	1	0	0	1	0	1	1	0	1	1	5B	2
3	0	0	1	1	0	1	0	0	1	1	1	1	4F	3
4	0	1	0	0	0	1	1	0	0	1	1	0	66	4
5	0	1	0	1	0	1	1	0	1	1	0	1	6D	5
6	0	1	1	0	0	1	1	1	1	1	0	1	7D	6
7	0	1	1	1	0	0	1	0	0	1	1	1	27	7
8	1	0	0	0	0	1	1	1	1	1	1	1	7F	8
9	1	0	0	1	0	1	1	0	1	1	1	1	6F	9
A	1	0	1	0	0	1	1	1	0	1	1	1	77	A
B	1	0	1	1	0	1	1	1	1	1	0	0	7C	b
C	1	1	0	0	0	0	1	1	1	0	0	1	39	C
D	1	1	0	1	0	1	0	1	1	1	1	0	5E	d
E	1	1	1	0	0	1	1	1	1	0	0	1	79	E
F	1	1	1	1	0	1	1	1	0	0	0	1	71	F

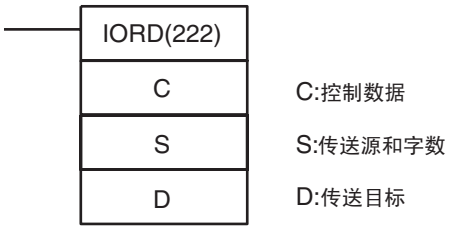


3-23-3 智能 I/O 读：IORD(222)

用途

读取 I/O 单元内存区的内容。

梯形图符号



变化

变化	ON 条件时每个循环执行	IORD(222)
	上升沿微分执行一次	@IORD(222)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

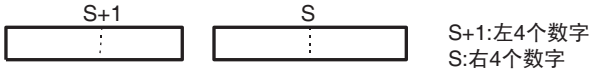
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

C：根据特殊 I/O 单元变化

S：单元号 0000 到 005F （0 ～ 95）

S ＋ 1：传送的字数 （0001 ～ 0080H，由特殊 I/O 单元决定）



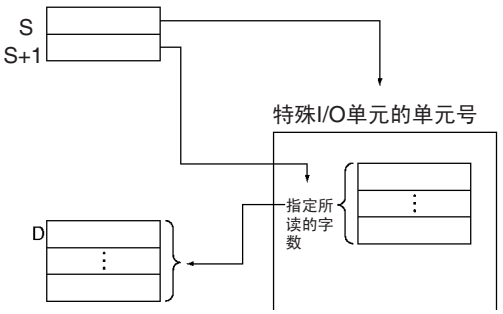
操作数规定

区域	C	S	D
CIO 区	CIO 0000 ～CIO 6143	CIO 0000 ～CIO 6142	CIO 0000 ～CIO 6143
工作区	W000 ～ W511	W000 ～ W510	W000 ～ W511
保持位区	H000 ～ H511	H000 ～ H510	H000 ～ H511
辅助位区	A000 ～ A959	A000 ～ A958	A448 ～ A959
定时器区	T0000 ～ T4095	T0000 ～ T4094	T0000 ～ T4095
计数器区	C0000 ～ C4095	C0000 ～ C4094	C0000 ～ C4095
DM 区	D00000 ～ D32767	D00000 ～ D32766	D00000 ～ D32767
无区号 EM 区	E00000 ～ E32767	E00000 ～ E32766	E00000 ～ E32767
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	En_00000 ～ En_32766 (n = 0 ～ C)	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		

区域	C	S	D
BCD 码间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #FFFF （二进制）	仅指定值	---
数据寄存器	DR0 ～ DR15	---	
索引寄存器	---		
使用索引寄存器 的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

IORD(222) 从被 S 指定了单元号的特殊 I/O 单元的内存区中读取 S+1 中指定的字数，并将结果输出到 D。只能指定安装在 CPU 机架或扩展机架上的特殊 I/O 单元。每一单元的被读数据的详细资料参见特殊 I/O 单元操作。



标志

名称	标记	操作
错误标志	ER	传送字数 (S) 在 0001 ～ 0080H 的范围外时置 ON。 若单元号 (S) 在 0000 ～ 005FH 范围之外置 ON。 指定的特殊 I/O 单元在 SYSMAC BUS 上时置 ON。 指定了一个不受 IORD(222) 影响的特殊 I/O 单元时置 ON。 指定的特殊 I/O 单元有特殊 I/O 单元设定错误或特殊 I/O 单元错误时置 ON。 对 CS1D CPU 单元：若主 CPU 和备用 CPU 不同步时置 ON。 其它情况置 OFF。
等于标志	=	读操作正常完成时置 ON。 读操作没有正常完成时置 OFF。

注意

读入操作正常完成时等于标记置 ON。

由于特殊 I/O 单元忙而使读入操作不能正常完成时，等于标记置 OFF。

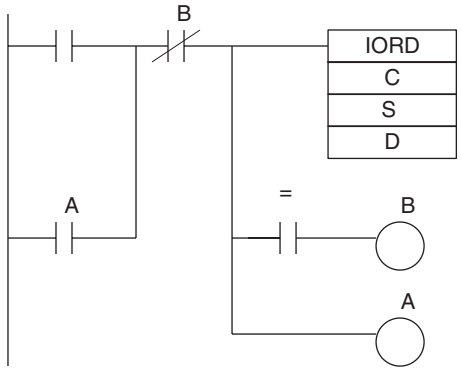
以下任一情况均会产生一个错误，并将错误标记置 ON。

- 传送字数 (S) 在 0001 ～ 0080H 的范围外。
- 单元号 (S) 在 0000 ～ 005F 的范围外。

- 指定的特殊 I/O 单元在 SYSMAC BUS 上。
- 指定了一个不受 IORD(222) 影响的特殊 I/O 单元。
- 指定的特殊 I/O 单元有特殊 I/O 单元设定错误或特殊 I/O 单元错误。

IORD(222) 的执行结果反映在条件标记中，读入完成时等于标记置 ON。在与 IORD(222) 指令相同的输入条件后，对输出分支加条件标记如等于标记。

如果特殊 I/O 单元忙，读操作不执行。使用等于标记创建如下所示的自保持程序，用于循环执行 IORD(222)，直至读入操作执行完成。

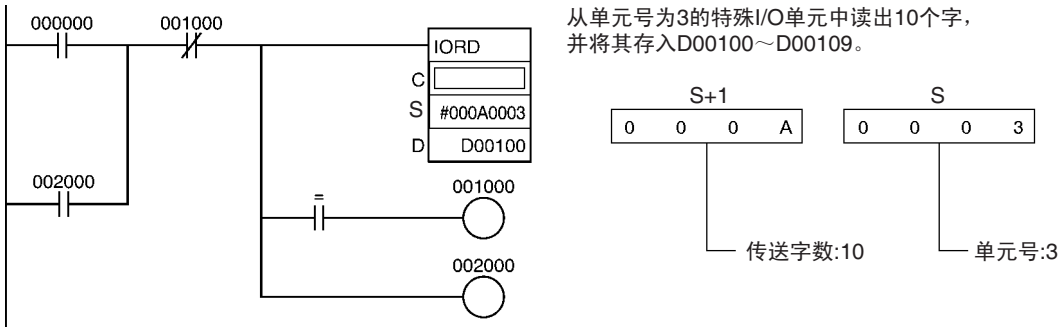


输入条件满足时，用输出 A 作为自保持，IORD(222) 每个循环执行，直至等于标记置 ON。读入完成后等于标记置 ON，输出 B 置 ON，清除自保持。

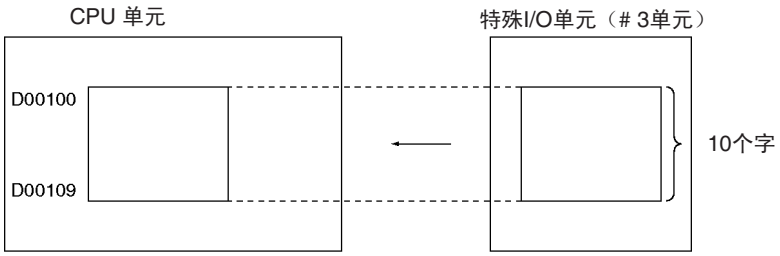
一定要在 IORD(222) 指令而非其它指令之后直接用条件标记。若在其它指令之后用条件标记，它将被该指令的执行结果所影响

例

本例中 IORD(222) 用于读取数据。



控制代码(C)根据特殊I/O单元而变化。CPU Unit CPU单元

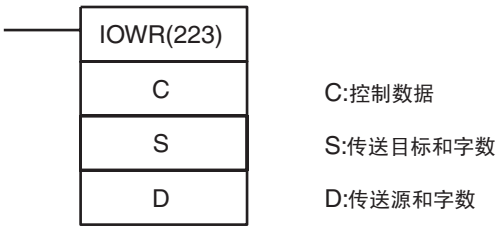


3-23-4 智能 I/O 单元写：IOWR(223)

用途

输出 CPU 单元的 I/O 内存区的内容到特殊 I/O 单元。

梯形图符号



变化

变化	ON 条件时每个循环执行	IOWR(223)
	上升沿微分执行一次	@IOWR(223)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

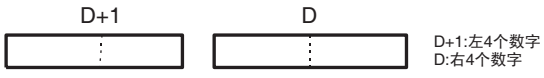
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

C：根据特殊 I/O 单元变化

D：单元号 0000 ～ 005F （0 ～ 95）

D ＋ 1：传送字数 （0001 ～ 0080H，由特殊 I/O 单元决定）

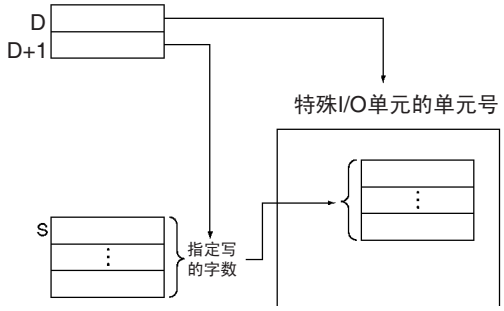


操作数规定

区域	C	S	D
CIO 区	CIO 0000 ～ CIO 6143		CIO 0000 ～ CIO 6142
工作区	W000 ～ W511		W000 ～ W510
保持位区	H000 ～ H511		H000 ～ H510
辅助位区	A000 ～ A959		A000 ～ A958
定时器区	T0000 ～ T4095		T0000 ～ T4094
计数器区	C0000 ～ C4095		C0000 ～ C4094
DM 区	D00000 ～ D32767		D00000 ～ D32766
无区号 EM 区	E00000 ～ E32767		E00000 ～ E32766
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		En_00000 ～ En_32766 (n = 0 ～ C)
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	#0000 ～ #FFFF （二进制）		仅指定值
数据寄存器	DR0 ～ DR15	---	---
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

IOWR(223) 将从第一个源字 （由 S 指定）开始的指定字数 (D) 输出到由 D 指定单元号的特殊 I/O 单元中。只能指定安装 CPU 机架或扩展 I/O 机架上的特殊 I/O 单元。



标志

名称	标记	操作
错误标志	ER	传送字数 (D) 在 0001 ～ 0080H 的范围外时置 ON。 若单元号 (D) 在 0000 ～ 005FH 范围之外置 ON。 S 由常数指定，传送字数 (D ＋ 1) 不是 0001H 时置 ON。 指定的特殊 I/O 单元在 SYSMAC BUS 上时置 ON。 指定了一个不受 IOWR(223) 影响的特殊 I/O 单元时置 ON。 指定的特殊 I/O 单元有特殊 I/O 单元设定错误或特殊 I/O 单元错误时置 ON。 对 CS1D CPU 单元：若主 CPU 和备用 CPU 不同步时置 ON。 其它情况置 OFF。
等于标志	=	读操作正常完成时置 ON。 读操作没有正常完成时置 OFF。

注意

传送字数 (D+1) 指定为 “0001” 时，数据 S 用常数指定。如果 S 指定为一常数，传送字数不是 0001 时，将出错，且错误标记置 ON。

写操作正常完成时，等于标记置 ON。

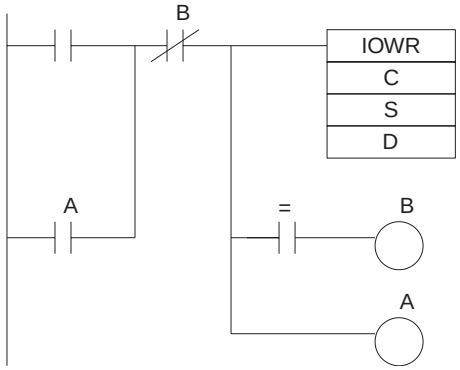
由于特殊 I/O 单元忙而写操作不能正常完成时，等于标记置 OFF。

以下任一情况均会产生一个错误，并将错误标记置 ON。

- 特殊I/O单元有I/O单元校验错误，特殊I/O单元设定错误，或特殊I/O单元错误。
- 传送字数 (D) 在 0001 ～ 0080H 的范围外。
- 单元号 (D) 在 0000 ～ 005F 的范围外。
- 指定的特殊 I/O 单元在 SYSMAC BUS 上。
- 指定了一个不受 IOWR(223) 影响的特殊 I/O 单元。
- 指定的特殊 I/O 单元有特殊 I/O 单元设定错误或特殊 I/O 单元错误。

IOWR(223) 的执行结果反映在条件标记中，写完成时等于标记置 ON。在与 IOWR(223) 指令相同的输入条件后，对输出分支加条件标记如等于标记。

如果特殊 I/O 单元忙，写操作不执行。使用等于标记创建如下所示的自保持程序，用于循环执行 IOWR(223)，直至写操作执行完成。

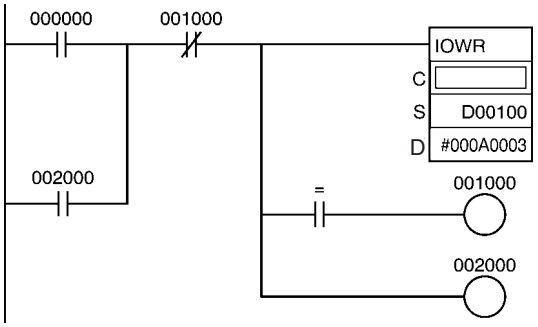


输入条件满足时，用输出 A 作为自保持，IOWR(223) 每个循环执行，直至等于标记置 ON。写完成后等于标记置 ON，输出 B 置 ON，清除自保持。

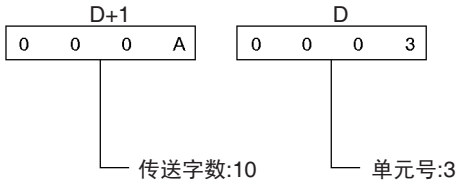
一定要在 IOWR(223) 指令而非其它指令之后直接用条件标记。若在其它指令之后用条件标记，它将被该指令的执行结果所影响。

例

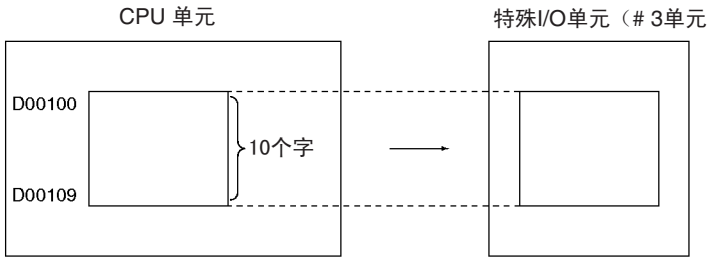
本例中， IOWR(223) 用于写数据。



当CIO000000置ON时，D00100到D00109中的10个字被写到特殊I/O单元。



控制代码(C)根据特殊I/O单元而变化。



3-23-5 CPU 总线单元 I/O 刷新：DLNK(226)

用途

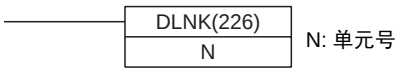
立即对有指定单元号的 CPU 总线单元执行 I/O 刷新。

下列数据将被刷新：

- 分配给PLC中CPU总线单元区（CIO区为25字，DM区为100字）的CPU总线单元的字。
- 诸如支持数据链接等单元的指定数据刷新。

该指令仅由 CS1-H,CJ1-H,CJ1M 和 CS1D CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每个循环执行	DLNK(226)
	上升沿微分执行一次	@DLNK(226)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

N：单元号
指定 CPU 总线单元的单元号 （0000 ～ 000F Hex 或 0 ～ 15 十进制）。

操作数规定

区域	N
CIO 区	CIO 0000 ～ CIO 6143
工作区	W000 ～ W511
保持位区	H000 ～ H511
辅助位区	A448 ～ A959
定时器区	T0000 ～ T4095
计数器区	C0000 ～ C4095
DM 区	D00000 ～ D32767
无区号 EM 区	E00000 ～ E32767
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 码间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	#0000 ～ #000F （二进制）或 0 ～ 15 （二进制）
数据寄存器	DR0 ～ DR15
索引寄存器	---
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

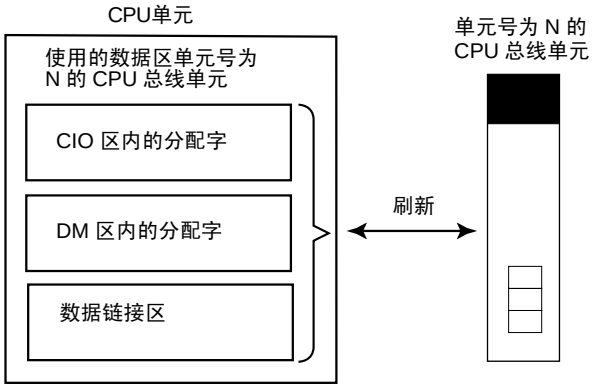
说明

DLNK(226) 对指定单元号的 CPU 总线单元执行立即 I/O 刷新。下面列出的数据被刷新。用于立即刷新的执行条件的细节参见下面的*注意*。

1. PLC 的 CPU 总线单元区内分配给 CPU 总线单元的字（CIO 区为 25 字，DM 区 100 字）

2. CPU 总线单元的数据，诸如数据链接数据，或 NeviceNet 网远程 I/O 通信数据（和 CPU 总线单元区的数据一起刷新）

CPU 总线单元	指定单元的数据刷新
控制器链接单元或 SYSMAC 链接单元	数据链接刷新
NeviceNet 网单元 （不包括 C200H NeviceNet 网主单元）。	远程 I/O 通信刷新



下表表示 DLNK(226) 和 IORF(097) 不同之处。

指令	操作
DLNK(226)	• CIO 区内 CS1 CPU 总线单元区的 I/O 刷新 （25 字） • DM 区内 CS1 CPU 总线单元区的 I/O 刷新 （100 字） • CPU总线单元的数据刷新，诸如数据链接数据，或设备网远程I/O通信数据
IORF(097)	• 基本 I/O 单元使用的字的 I/O 刷新 • 分配给特殊 I/O 单元的 10 个 CIO 字的 I/O 刷新

DLNK(226) 刷新 CPU 单元和指定 CPU 总线单元之间的数据。使用 DLNK(226) 时，有两个特殊因素需要考虑：

1,2,3...

1. 通过数据链接或设备网远程 I/O 通信进行数据交换时，在执行 DLNK(226) 的同时，其它单元不能执行数据交换。在网络通信周期到达时，查寻单元及与其交换数据的单元，执行数据交换。因此，实际数据交换可能根据网络通信周期时间发生相应延迟。
2. 如果该单元正在进行数据交换，DLNK(226) 不能对 CPU 总线单元执行 I/O 刷新。如果 DLINK(226) 执行过为频繁， I/O 刷新将不能完成。建议允许 DLNK(226) 指令执行之间的延迟长于通信周期时间。

标志

名称	标记	操作
错误标志	ER	若指定单元号在 0000 ～ 000FH(0 ～ 15) 的范围外时置 ON。 PLC 没有指定 CPU 总线单元的单元号置 ON。 对 CS1D CPU 单元：若主 CPU 和备用 CPU 不同步时置 ON。 其它情况置 OFF。
等于标志	=	若由于 CPU 总线单元刷新数据， I/O 刷新不能进行置 OFF。 若在指定 CPU 总线单元出现 CPU 总线单元错误或 CPU 总线单元设置错误时，置 OFF。 若 DLNK(226) 在一中断任务中执行，与常规 I/O 刷新冲突，产生交迭刷新，置 OFF。 若 I/O 刷新正常完成置 ON。

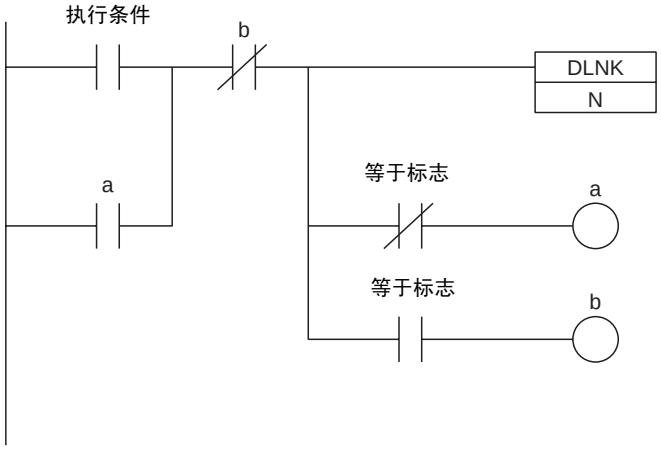
注意

如果在指定 CPU 总线单元中已经产生 CPU 总线单元错误 (A40207) 或 CPU 总线单元设置错误 (A40203)， I/O 刷新不会执行。

如果当 DLNK(226) 执行 I/O 刷新时产生 I/O 总线错误， I/O 刷新终止。

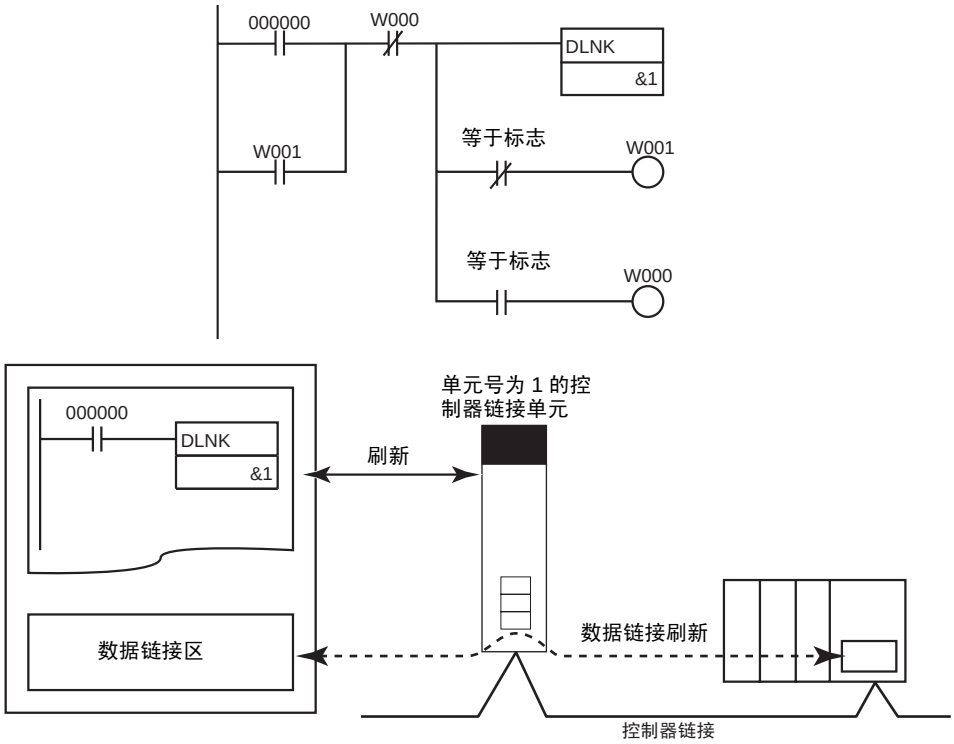
DLNK(226) 刷新 CPU 单元和指定 CPU 总线单元之间的数据。 CPU 总线单元的数据交换需要一段时间 （例如，控制器链接单元的数据链接）。

如果指定的 CPU 总线单元进行数据交换，不执行 DLNK(226)，且等于标记置 OFF。建议对执行条件编程如下， DLNK(226) 的执行将自动重试。



例

当下例中的 CIO 000000 置 ON 时， DLNK(226) 对单元号为 1 （在本情况下，为一控制器链接单元）的 CPU 总线单元执行立即 I/O 刷新 （在本情况下，在 PLC 内刷新数据链接）。如果由于控制器链接单元正刷新数据 I/O 刷新不能执行，等于标记置 OFF，使得 W001 置 ON，指令的执行会在下一循环中重试。当 I/O 刷新正常完成时，等于标记置 ON，指令不会在下一循环中重试。

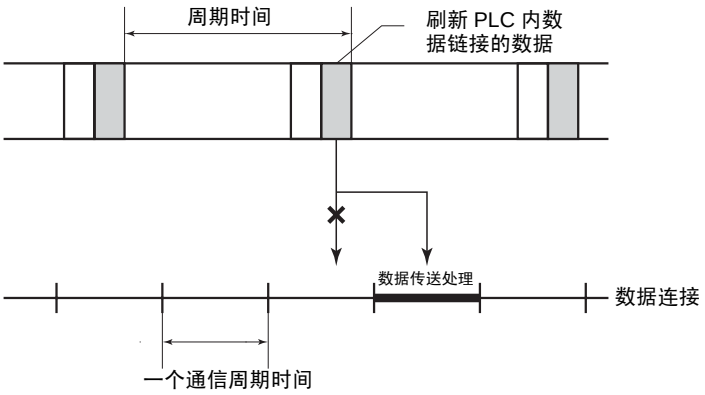


本例中数据链接区刷新的实际计时如下：

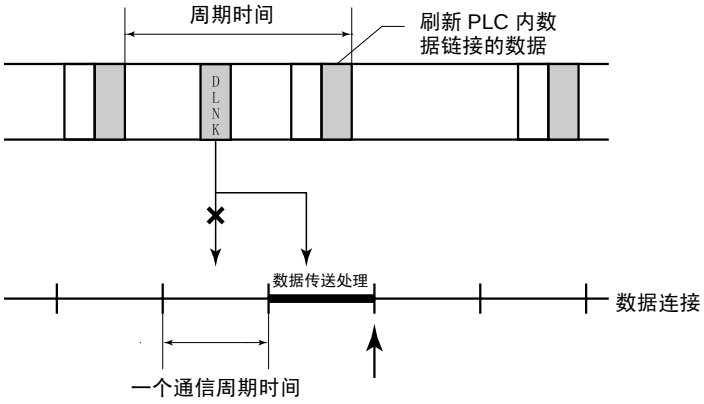
- 传送：在得到正常标志的下一时间，数据通过网络传送。（数据传送最大延迟一个通信周期时间）。
- 接收：在得到正常标记的最后时间，从网络接收数据。（数据接收最大延迟一个通信周期时间）。

数据传送处理示例：

- 传送从先前 I/O 刷新的数据。



- 执行 DLNK(226) 转换数据。



3-24 串行通信指令

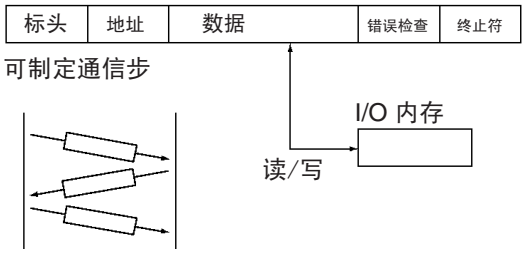
本节介绍了用于串行通信的指令。

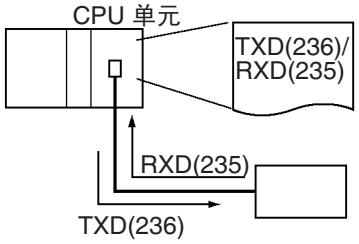
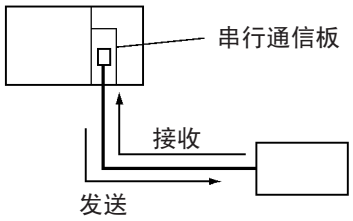
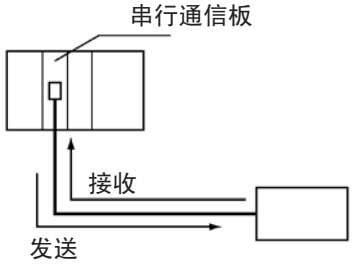
指令	助记符	函数代码	页
协议宏	PMCR	260	838
传送	TXD	236	853
接收	RXD	235	851
改变串行端口设置	STUP	237	856

3-24-1 串行通信

串行通信指令有两种类型。 TXD(236) 和 RXD(235) 通过无协议 （定制）通信向外部设备发送或接收数据。PMCR(260) 使用用户定义的协议向外部设备发送或接收数据。下面的表格说明了二者的区别。

注 PMCR(260)协议宏函数包括TXD(236)和RXD(235)的无协议通信函数。TXD(236)和 RXD(235) 只用在 CPU 单元上的串行端口中。 PMCR(260) 只用在串行通信单元或串行通信板 （仅适于 CS 系列）中的串行端口。 .

指令	通信帧	功能
TXD(236) 和 RXD(235)	可使用下列任一种形式。 无开始或结束代码：数据 开始或结束代码：ST | 数据 | ED 只有开始代码：ST | 数据 CR＋LF结束代码：数据 | CR | LF 只有结束代码：数据 | ED 开始和CR＋LF结束代码：ST | 数据 | CR | LF	只能单向发送或接收数据。 可设置一个发送延迟。
PMCR(260)	可根据外设需要创建如下类型的帧（信息） 	可定义多达 16 步的发送和接收。 可基于响应修改步并再次执行。 可设定通信监视时间。 可向 PLC 读 / 写符号。 可使用重复符号。 其他。

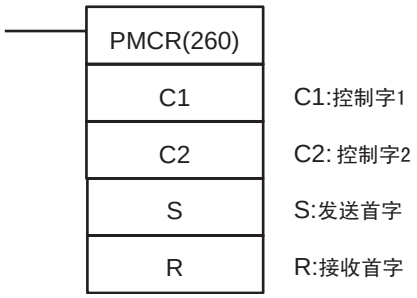
指令	模式	通信端口	
TXD(236) 和 RXD(235)	无协议 （定制）	CPU单元中内置的串行端口 	TXD(236)和RXD(235)不能在串行通信板或其它单元的串行端口上使用。
PMCR(260)	协议宏	串行通信板（仅适用于CS系列） 	串行通信单元 

3-24-2 协议宏指令：PMCR(260)

用途

调用和执行一个已在串行通信板 （仅适于 CS 系列）或串行通信单元中注册的通信序列。

梯形图符号



变化

变化	ON 条件时每个循环执行	PMCR(260)
	上升沿微分执行一次	@PMCR(260)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

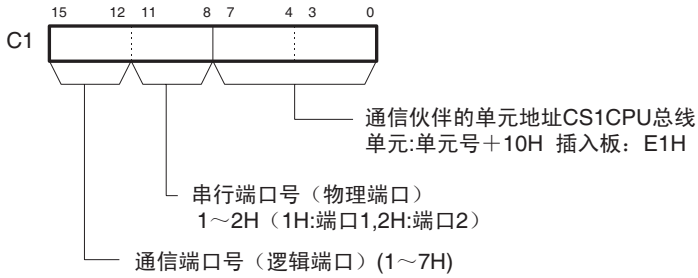
适用程序区

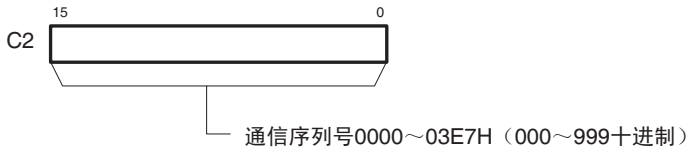
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

C1：控制字 1 和 C2：控制字 2

两个控制字的内容如下所示：



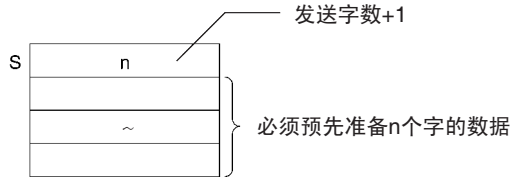


注 使用通信端口号 （逻辑端口）自动分配的细节参见 *872 页通信端口自动分配*。

S：发送首字和发送区

指定要发送的字的首字。S 含有传送字数＋ 1（即包括 S 字），且从 S+1 开始发送数据。 0000 和 00FAH(0 ～ 250) 之间的字可被发送。

如果无发送数据，将 S 的内容设定为常数 0000。若设为其它常数或某字的地址，则会导致错误，并且错误标志置 ON， PMCR(260) 不执行。



R：接收首字和接收区

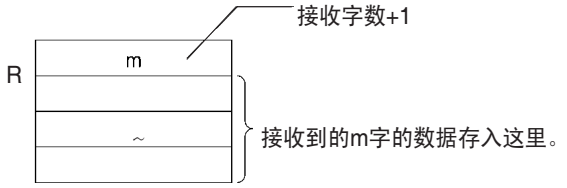
接受到的数据自动存入从 R+1 开始的字中，接收的字数 +1 （即包括 R）自动写到 R， R 在 0000 ～ 00FAH 之间 （0 ～ 250 十进制）。

执行 PMCR 前的设置

设定有 m 指定的数据 （从 D 开始）为接收缓冲器的初始数据 （接收失败时的备份数据。数据 m 可设在 0002 ～ 00FAH（2 ～ 255）之间。若 m 设定为 0000 或 0001H，接收缓冲器的初始值清零。

即使无接收数据也要为 R 设定一个字地址。若设为常数则会出错，错误标记置 ON，并且不会执行 PMCR(260)。若无接收数据，则不会使用 R，就可用于其它用途。

若无接收数据，一定要设为 # 0000 ～ # FFFF 之间的常数。



操作数规定

区域	C1	C2	S	R
CIO 区	CIO 0000 ～ CIO 6143			
工作区	W000 ～ W511			
保持位区	H000 ～ H511			
辅助位区	A000 ～ A447 A448 ～ A959			A448 ～ A959
定时器区	T0000 ～ T4095			
计数器区	C0000 ～ C4095			
DM 区	D00000 ～ D32767			
无区号 EM 区	E00000 ～ E32767			
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)			
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)			
BCD 码间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)			

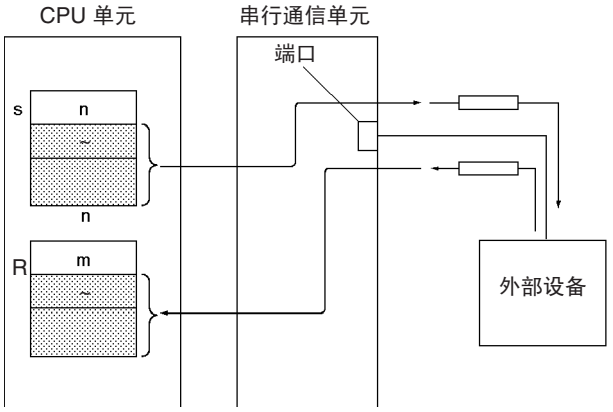
区域	C1	C2	S	R
常数	仅指定值	0000 ～ 03E7Hex (0 ～ 999)	#0000 （二进制）	
数据寄存器	DR0 ～ DR15		---	
索引寄存器	---			
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15			

说明

PMCR(260) 使用 C1 第 12 ～ 15 位指定的逻辑端口和 C1 第 0 ～ 7 位指定的单元地址，在 C1 第 8 ～ 11 位指定的物理端口执行 C2 指定的通信序列。

如果对发送信息指定一个符号为操作数，S 中指定发送字数，并从 S+1 开始作为发送区。如果对接收信息指定符号为操作数，若传送成功，接收数据存入从 R+1 开始的内存，接受到的字数被自动写入 R。

若传送失败，从接收缓冲器中读出 PMCR(260) 执行前设置的数据 （R+1 之后），重新存入 R+1 之后的区域。



标志

名称	标记	操作
错误标志	ER	执行 PMCR(260) 时，指定逻辑端口的通信端口允许标志为 OFF 时置 ON。 C1 不在指定范围内时置 ON。（C2 的数据不在指定范围内时错误标志不置 ON。结束码存贮在辅助区的通信端口完成码 (A203 ～ A210) 中） S 或 R 的字数超出 249 （指定字时）时置 ON。 其它情况置 OFF。

注意

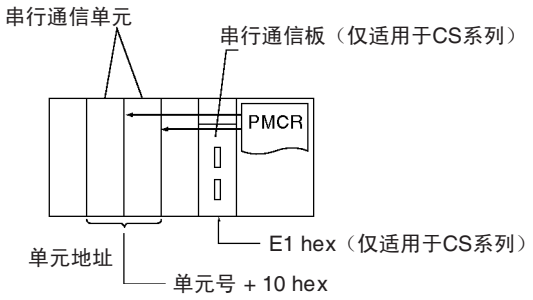
由 S 指定的发送区中数据，实际上是用发送信息中的符号读选项 (R( )) 来发送的。

数据实际上是用接收信息中的符号写选项 (W( )) 来接收的。

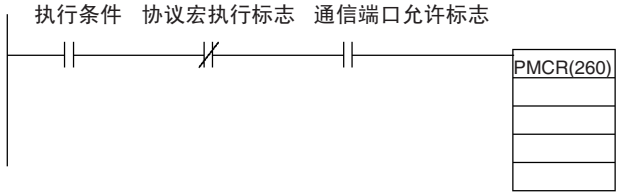
符号 R() 和 W() 的指定过程参见 *CX-Protocol 操作手册 (W344)*。

PMCR(260) 可用在串行通信板（仅适于 CS 系列）或串行通信单元的串行通信端口上。可在 CPU 机架和扩展 I/O 机架上安装多达 16 个串行通信单元。通信伙伴的单元地址在 C1 的 0 ～ 7 位中设定，以指定所用的单元 / 板，串行端口必须在第 8 ～ 11 位中指定。单元地址指定如下表所示。

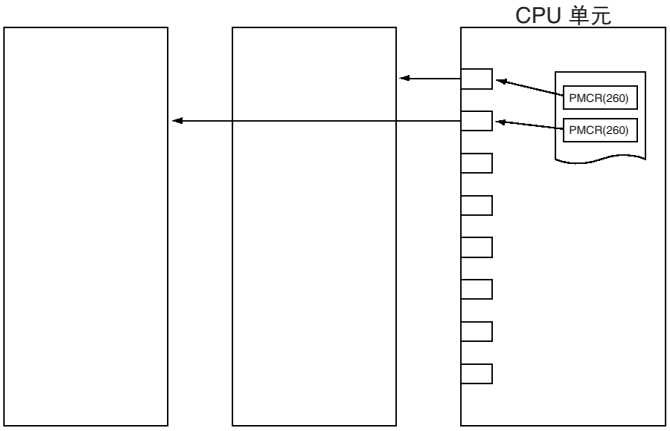
单元 / 板	单元地址
串行通信板 （仅适用于 CS 系列）	E1 hex
串行通信单元	单元号 + 10 hex



在 PMCR(260) 开始执行时，对应的协议宏执行标记置 ON，通信序列执行完毕且数据已被写入指定接收区之后置 OFF。相应协议宏执行标记的输入 (A.N.C.) 是执行条件的一部分，以保证在执行 PMRC(260) 时，在同一时间同一物理端口仅有一个通信序列正在执行。具体实例如下。



SEND(090)、RECV(098) 和 CMND(490) 也通过串行通信单元和串行通信板（内部使用 FINS 命令），也使用逻辑端口 0 ～ 7 执行通信序列。 PMCR(260) 不能在已被 SEND(090)、RECV(098)、CMND(490) 或 PMCR(260) 使用的逻辑端口上执行。如上图所示，为防止几个通信序列同时使用一个逻辑端口，使用相应的通信端口允许标记 (A20200 ～ A20207)，作为 PMCR(260) 的执行条件中的常开输入。



下述情况时错误标志置 ON。

- 执行 PMCR(260) 时，指定逻辑端口 (0 ～ 7) 相应的通信端口允许标志为 OFF。
- C1 不在指定范围时。

接收区的指定

在执行 PMCR(260) 之前，为防止接收过程失败，用户应在接收区设置备份数据。执行 PMCR(260) 时，接收缓冲区中的数据被自动存入接收区。备份数据应用例子如下：预先设定某一值 （备份数据），以保证在执行协议读取控制器的当前值产生传送错误时，读入的当前值不是零。

相关标志和字

在执行 PMCR(260) 时可使用以下标志和字。

辅助区

名称	地址	内容
通信端口允许标志	A20200 ～ A20207	网络通信允许时 （包括 PMCR(260)）置 ON。 第 00 ～ 07 位分别对应于逻辑端口 0 ～ 7。 启动网络通信时，网络通信指令允许标志置 OFF，执行完毕后置 ON （无论通信是正常结束还是有错误）。

名称	地址	内容
通信端口错误标记	A21900 ～ A21907	网络通信出现错误时置 ON。 第 00 ～ 07 位分别对应与逻辑端口 0 ～ 7。 在启动下一次网络通信前，标志状态不变。即使前次执行产生了错误，再次启动通信时，标志也将置 OFF。
通信端口完成码	A203 ～ A210	网络通信完成时保存完成码。 字 A203 ～ A210 分别对应于逻辑端口 0 ～ 7。 执行通信指令时完成码位 00。 执行完毕后存储新的响应代码。 在操作开始时这些字的内容被清除。

通信响应

代码	内容
1106 (hex)	无相应程序号。 指定未注册的发送 / 接收序列号。 使用 CX-Programmer 改变或添加发送 / 接收序列号。
2201 (hex)	由执行协议造成的不可操作性。 因已执行一个协议宏，不接收进一步执行。 向程序添加协议宏执行标志的常闭条件。
2202 (hex)	由故障导致的不可操作性。 因协议已被切换，不再接收进一步执行。 向程序添加串行设置改变标志的常闭条件。
2401 (hex)	无注册表。 协议宏数据或传送的数据出错。 使用 CX-Programmer 传送协议宏数据
其他	其它响应代码参见 *CS/CJ 系列通信命令参考手册 (W342)*。

插入板区 （仅适于 CS 系列）

名称	地址	内容
端口 1 协议宏执行标记	CIO 190915	执行 PMCR(260) 时置 ON。
端口 2 协议宏执行标记	CIO 191915	执行失败该标志保持 OFF。 通信序列完成时置 OFF （结束或退出）。

CPU 总线单元区

n = 1500 + 25 x 单元号

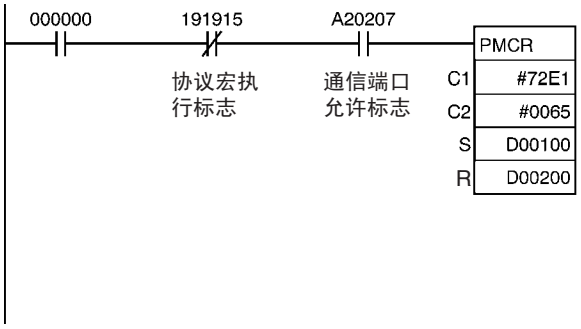
名称	地址	内容
端口 1 协议宏执行标记	CIO n + 9 的第 15 位	执行 PMCR(260) 时置 ON。 执行失败该标志保持 OFF。 通信序列完成时置 OFF （结束或退出）。
端口 2 协议宏执行标记	CIO n + 19 的第 15 位	

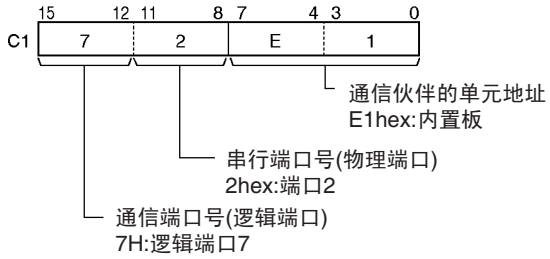
例

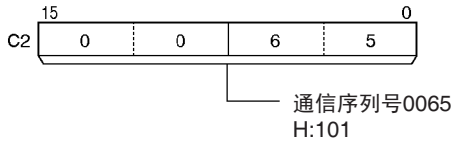
当下例中 CIO 000000 为 ON 时，只要端口 7(A20207) 的通信端口允许标记为 ON，且端口 1 协议宏执行标记 (CIO 190915) 为 OFF，就执行通信序列 101(0065H)。

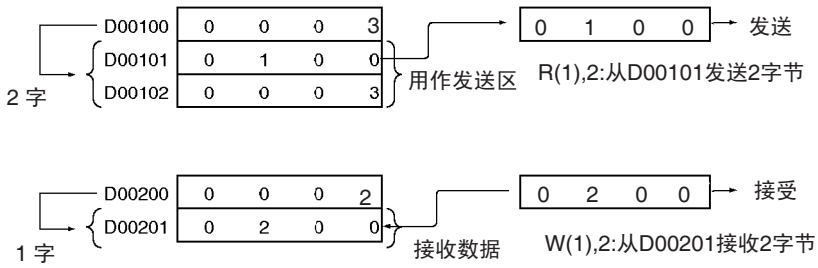
如果在发送信息中对符号指定操作数，从 D00101 开始的数据的 2 个字节用作发送区 （由于 D00101 的内容为 # 0003）。

如果在接收信息中对符号指定操作数，数据的 2 个字节从 D00201 开始存入，接收字数 +1 被写入 D00200。









注 如上所示，发送信息中的符号读选择R(),
或符号写选择W(),实际发送/接收数据。

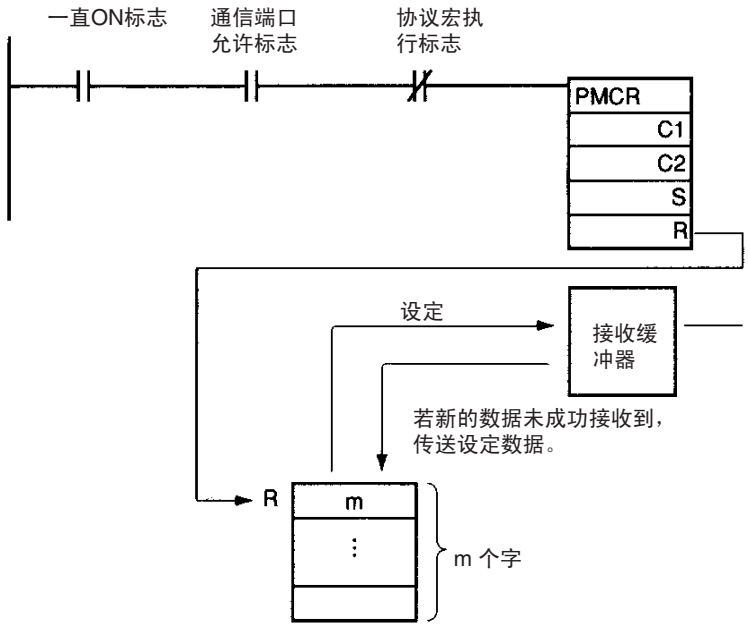
保持接收区

在 PMCR(260) 执行通信序列之前，立即将接收缓冲区清零。如果使用如下所示的程序周期性读取PV数据或其它值，且由于接收错误或其它原因不能读出数据，读入的数据将清除，直至下一次读取成功。

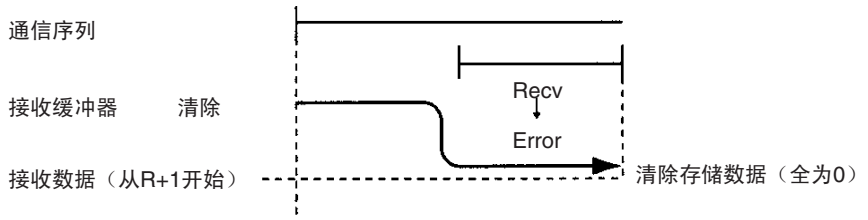
在产生接收错误时，可以用函数保持接收区中的数据。如果使用该函数，在执行通信序列之前缓冲区清零之后，数据从接收区的开始 m 字传送至接收缓冲区。这样就可以避免在未得到新的接收数据时，因写入当前接收数据而使接收区暂时清零。

m 值为保持接收区的指定字数。若指定为 0 或 1，保持函数被禁止，接收区被清零。

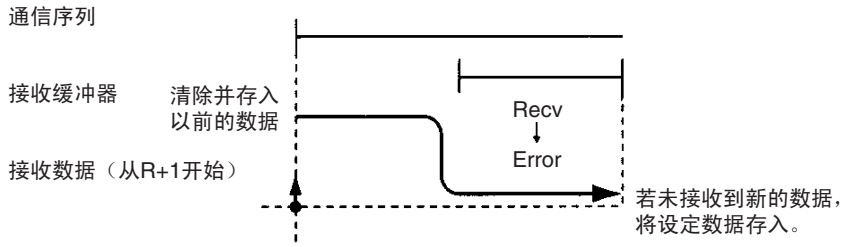
下面的程序示例显示了用于通过单个接收操作，持续地或周期性地执行 PMCR(260) 读取数据的指令。



接收区不保持



接收区保持

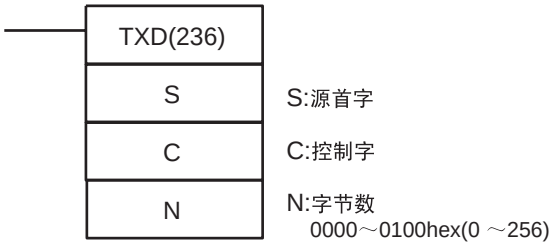


3-24-3 传送：TXD(236)

用途

从 CPU 单元中内置的 RS-232C 端口输出指定字节数的数据。

梯形图符号



变化

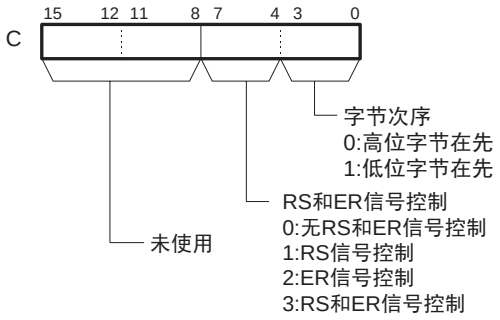
变化	ON 条件时每个循环执行	TXD(236)
	上升沿微分执行一次	@TXD(236)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

控制字 C 的内容如下所示。



操作数规定

区域	S	C	N
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A447 A448 ～ A959		
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		

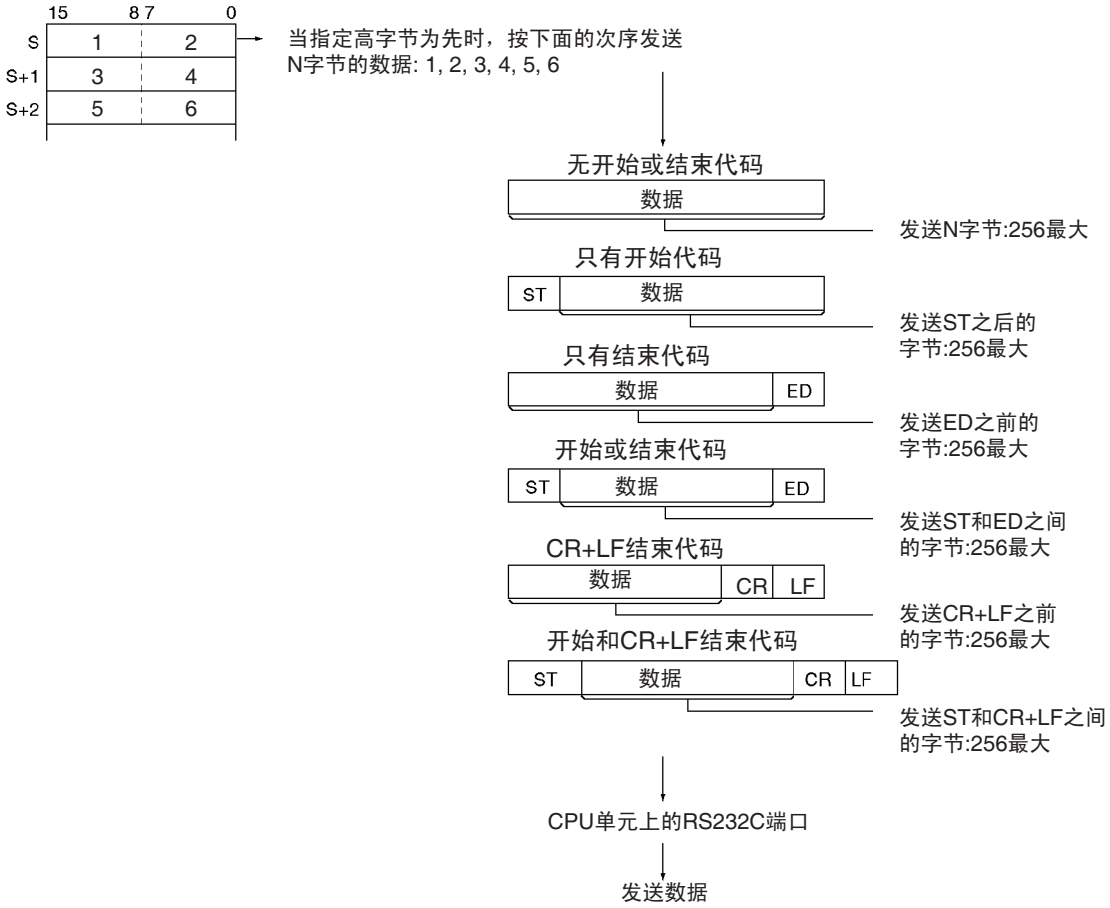
区域	S	C	N
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---	仅指定值	#0000 ～ #0100 （二进制）
数据寄存器	---	DR0 ～ DR15	
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

TXD(236) 从字 S ～ S+(N ÷ 2)-1 读出 N 字节数据，并按无协议模式将数据无变换地从 CPU 单元内置的 RS-232C 端口输出。输出数据时，在 PLC 设置中为无协议模式加入起始和结束码。

只有发送准备号标记 (A39205) 为 ON 时才能发送数据。

下图为数据的发送次序及发送帧不同设定的内容。



标志

名称	标记	操作
错误标志	ER	在 PLC 设置中未设定无协议模式时置 ON。 C 不在范围时置 ON。 N 值不在 0000 ～ 0100H 时置 ON。 当发送准备好标志为 OFF 时试图发送，置 ON。 其它情况置 OFF。

注意

TXD(236) 只能用于 CPU 单元的 RS-232C 端口，且端口为无协议模式。

在 PLC 设置中可设定如下帧格式传递信息。

- 起始码：无或 00 ～ FFH
- 结束码：无、 CR+LR 或 00 ～ FFH

可以在 PLC 设置中任意指定起始 / 或结束码来发送数据，如果指定了起始和结束码，可指定的最大字节数 N 降为 254 字节。

只有当发送准备好标记 (A39205) 为 ON 时，才能发送数据。

数据依照 C 中指定的次序发送。

指定 N 为 0 时不发送。

若在 C 中指定了 RS 信号控制，则 S 的第 15 位被用作 RS 信号。

若在 C 中指定了 ER 信号控制，则 S 的第 15 位被用作 ER 信号。

若在 C 中指定了 RS 和 ER 信号控制，则 S 的第 15 位被用作 RS 信号，S 的第 14 位被用作 ER 信号。

若在 C 中指定了 RS 和 ER 的信号控制为 1、 2 或 3Hex，则在执行 TXD(236) 时不考虑发送准备好标记 (A39205) 的状态。

以下情况会导致错误并使错误标记置 ON。

- 未在 PLC 设置中设定无协议模式时。
- C 不在范围内。
- N 值不在 0000 ～ 0100Hex 之间。
- 当发送准备好标记为 OFF 时试图发送。

在向某些设备发送数据时可能需要一个通信延迟，根据外设需要设定延迟。

相关标志和字

在执行 TXD(236) 时可使用以下 PLC 设置的设定和辅助区标志。

PLC 设置

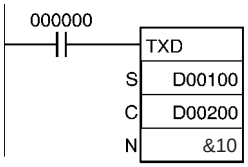
名称	内容	设置
无协议模式起始码	为无协议模式通信指定是否在帧格式中使用起始码。	0：无（缺省） 1：是（设置 00 ～ FFH）
无协议模式结束码	为无协议模式通信指定是否在帧格式中使用结束码。	0：无（缺省） 1：是（设置 00 ～ FFH 或 CR+LF）
无协议模式发送延时	只有在梯形图中执行 TXD(236) 已经过了指定延迟时间后，才从端口发送指定数据。	0 ～ 99990ms（以 10ms 为单位） 缺省：0 ms

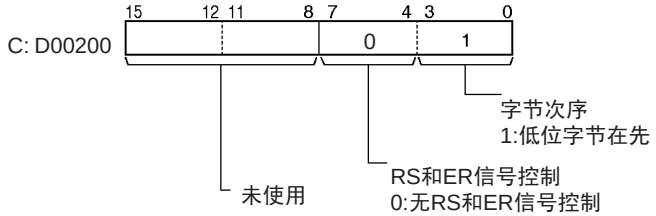
辅助区

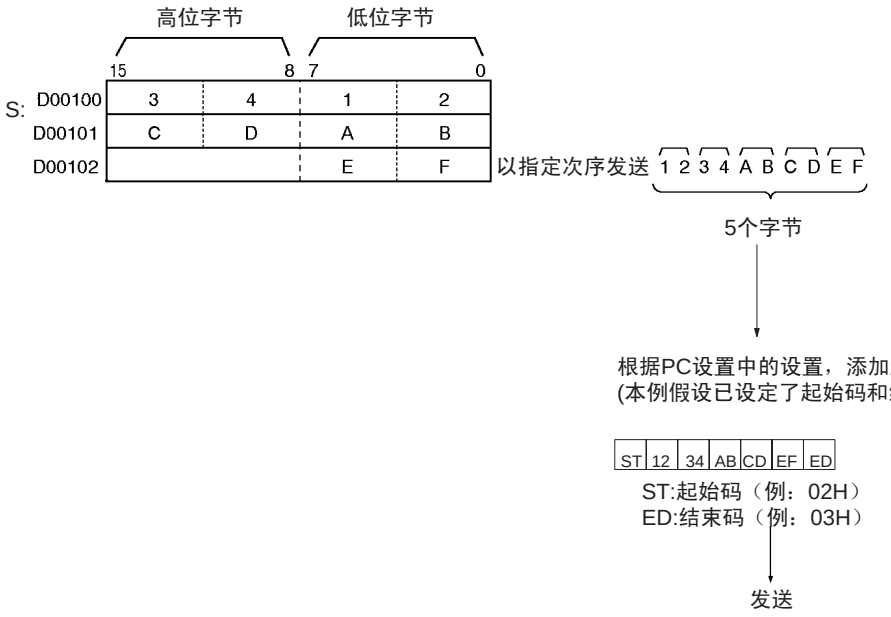
名称	地址	内容
RS-232C 端口发送准备好标记	A39205	数据可用无协议模式发送时置 ON。

例：发送数据

在下例中，当 CIO000000 置 ON 时，不进行任何转换，发送从 D00100 开始的 10 字节数据。

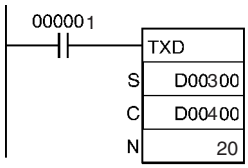


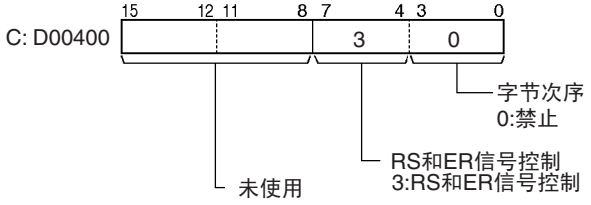


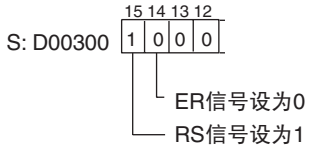


例： 执行信号控制

下例中 CIO 000001 置 ON 时， RS 信号和 ER 信号分别根据 D00300 第 15 为和第 14 位的状态设置。





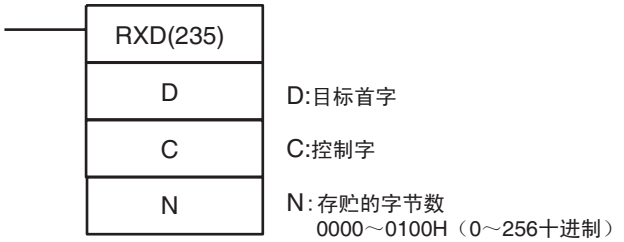


3-24-4 接收：RXD(235)

用途

从指定端口读入指定字节数的数据。

梯形图符号



变化

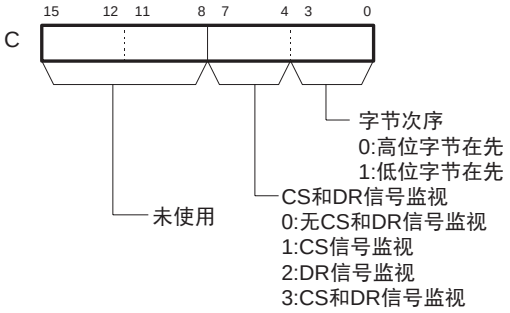
变化	ON 条件时每个循环执行	RXD(235)
	上升沿微分执行一次	@RXD(235)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

控制字 C 的内容如下所示。



操作数规定

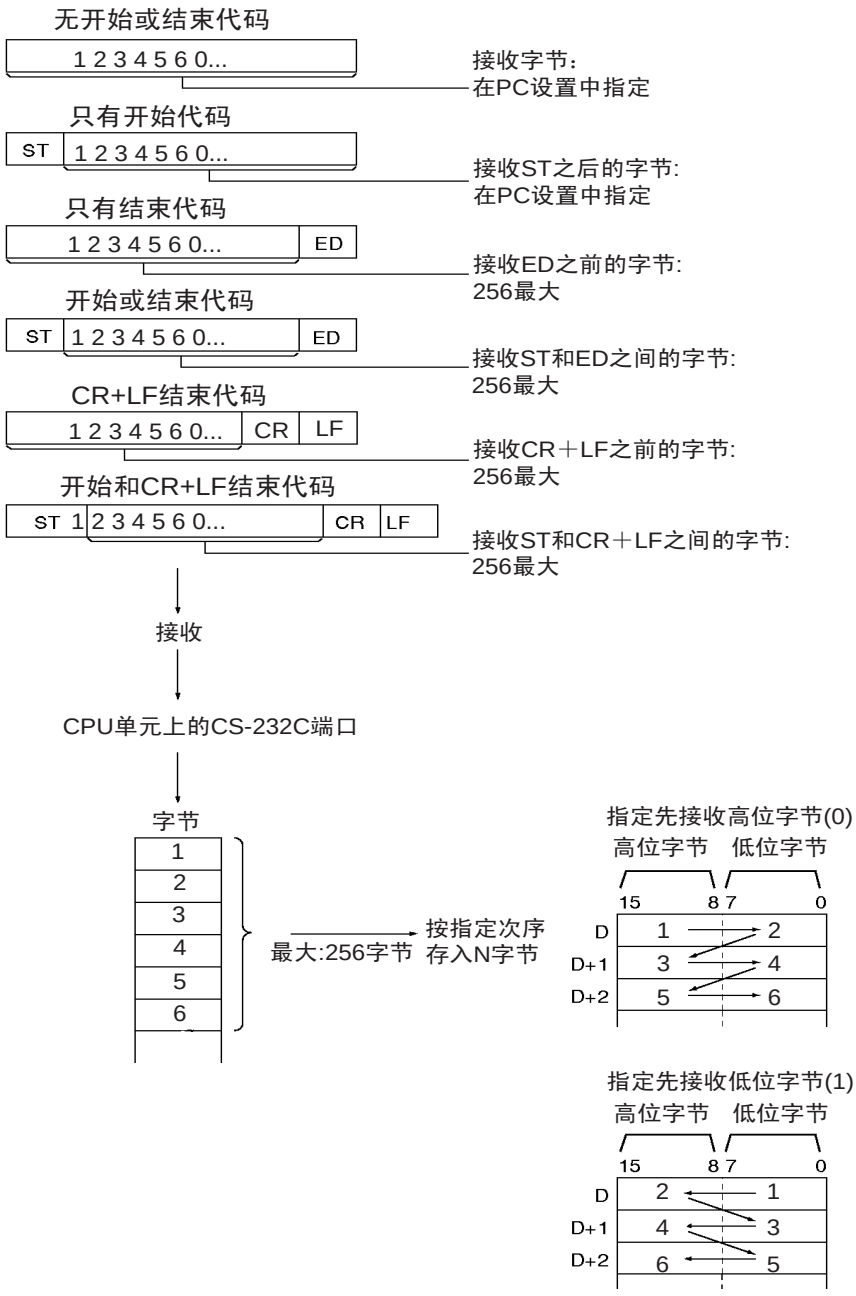
区域	D	C	N
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A448 ～ A959	A000 ～ A447 A448 ～ A959	
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---	仅指定值	#0000 ～ #0100 （二进制）
数据寄存器	---	DR0 ～ DR15	
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

RXD(235) 读入从 CPU 单元内置的 CS-232C 端口接收到的数据，并按无协议模式将 N 字节数据存入字 D ～ D+(N ÷ 2)-1 中。接收数据时，也接收了在 PLC 设置中为无协议模式指定的起始和结束代码。若端口接收到不足 N 字节的数据，则只存入已接收的数据。

只有当接收准备好标记 (A39206) 为 ON 时才能接收数据。

下图显示了接收数据的次序及不同设置时接收帧的内容。



标志

名称	标记	操作
错误标志	ER	在 PLC 设置中未设定无协议模式时置 ON。 C 不在范围时置 ON。 N 值不在 0000 ～ 0100H 时置 ON。 对 CS1D CPU 单元：若主 CPU 和备用 CPU 不同步时置 ON。 其它情况置 OFF。

注意

RXD(235)只能用于CPU单元的CS-232C端口，且端口只能设定为无协议模式。接收信息时，在 PLC 设置中可设定如下帧格式。

- 起始码：无或 00 ～ FFH
- 结束码：无，CR+LR 或 00 ～ FFHex。若为指定结束码，将接收的数据设在 00 ～ FFH （1 ～ 256 十进制； 00 用于指定 256 个字节）。

接收到 PLC 设置中指定的字节数后，接收完成标记 (A39206) 置 ON。在接收完成标记置 ON 时，接收计数器 (A393) 中的字节数与 PLC 设置中指定的接收字节数相同。若接收了超出指定的字节，接收溢出标记 (A39207) 置 ON。

若在 PLC 设置中指定了结束码，则当接收到结束码或已接收了 256 字节的数据时，接收完成标记 (A39206) 置 ON。若在接收完成标记置 ON 后，接收了更多的数据，接收溢出标记 (A39207) 置 ON。

执行 RXD(235) 后，数据被存入从 D 开始的内存，接收完成标记 (A39206) 置 OFF （甚至在接收溢出标记 (A39207) 为 ON 时），接收计数器 (A393) 清零。

若 RS-232C 端口重启动位 (A52600) 置 ON，数据被存入从 D 开始的内存，接收完成标记 (A39206) 置 OFF（甚至在接收溢出标记 (A39207) 为 ON 时），接收计数器 (A393) 清零。

数据依照 C 中指定的次序存入内存。

若在 PLC 设置中指定了起始和终止码，指定字节数 N 被忽略，在 RXD(235) 执行时，从起始码到结束码的数据被接收，并由 D 开始存入。

若 N 指定为 0，接收完成标记 (A39206) 将置 OFF，接收计数器 (A393) 清零，没有数据存入内存。

若在 C 中指定了 CS 信号监视，则 CS 信号的状态被存入 D 的第 15 位。

若在 C 中指定了 DR 信号监视，则 DR 信号的状态被存入 D 的第 15 位。

若在 C 中指定了 CS 和 DR 信号监视，则 CS 信号的状态被存入 D 的第 15 位，DR 信号的状态被存入 D 的第 14 位。

若指定了 CS 或 DR 信号监视，则不存贮接收数据。

若在 C 中指定了 RS 和 ER 的信号控制为 1、2 或 3H，则在执行 RXD(235) 时不考虑接收准备好标记 (A39206) 的状态。

以下情况会导致错误，并将错误标记置 ON。

- 未在 PLC 设置中设定无协议模式。
- C 值超出范围。
- N 值不在 0000 ～ 0100H 之间。

未读出用 RXD(235) 接收的数据，就不能接收后面更多的数据。当接收完成标记置 ON 时读出数据，并执行下一个 RXD(235)。

相关标记和字

在执行 RXD(235) 时，可根据需要，使用以下 PLC 设置的设定和辅助区标志。

PC 设置

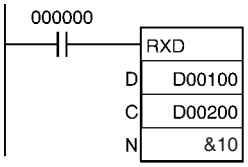
名称	内容	设置
无协议模式起始码	为无协议模式通信指定是否在帧格式中使用起始码。	0：无（缺省） 1：是（设置 00 ～ FFH）
无协议模式结束码	为无协议模式通信指定是否在帧格式中使用结束码。	0：无（缺省） 1：是（设置 00 ～ FFH 或 CR+LF）
无协议模式接受数据字节	在指定无结束码时指定读入的字节数。	01 ～ FF（1 ～ 255 十进制） 缺省：00（256 字节）

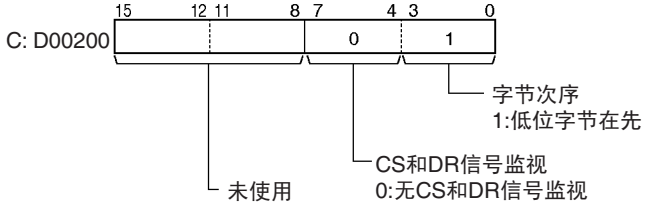
辅助区

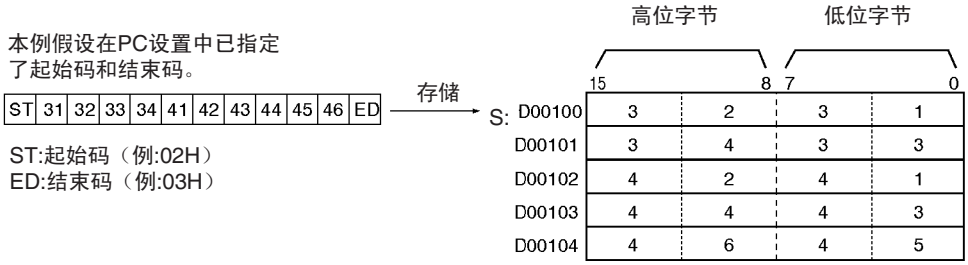
名称	地址	内容
RS-232C 端口接收完成标志	A39206	无协议接收完成时置 ON。 指定接收字节数： 已接收到指定字节数时标记置 ON。 指定结束码：接收到结束码或已接收了 256 个字节时标记置 ON。
RS-232C 端口接收溢出标志	A39207	接收到多于指定字节数时置 ON。 指定接收字节数：当接收完成之后和下一个 RXD(235) 执行前，接收了任何数据时，标记置 ON。 指定结束码：当接收到结束码后和下一个 RXD(235) 执行前，或在接收到结束码前接收了第 257 个字节时，标记置 ON。
RS-232C 计数器	A393	以十六进制记录以无协议模式接收到的字节数。

例

在下例中，当 CIO000000 置 ON 时，从 RS-232C 端口接收数据，并将 10 个字节数据从 D00100 开始存入。





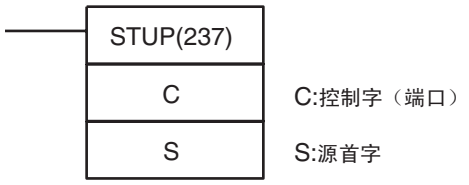


3-24-5 改变串行口设置：STUP(237)

用途

改变在 CPU 单元、串行通信板（仅适于 CS 系列）、串行通信单元（CPU 总线单元）上的串行端口的通信参数。STUP(237) 允许在 PLC 操作中改变协议模式。

梯形图符号



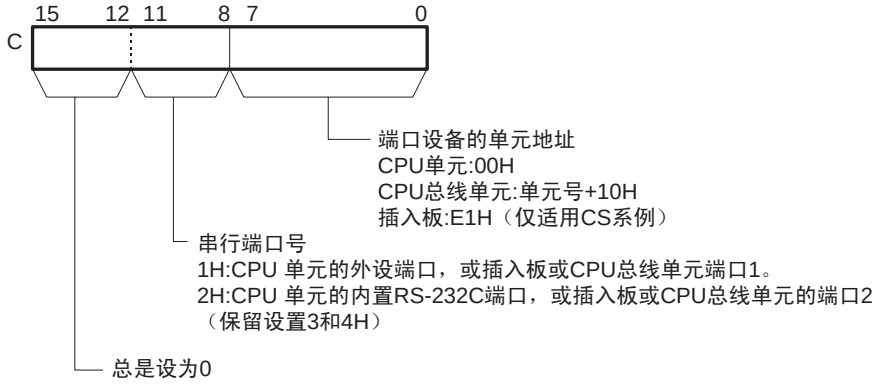
变化

变化	ON 条件时每个循环执行	STUP(237)
	上升沿微分执行一次	@STUP(237)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	不允许

操作数 控制字 C 的内容如下所示



操作数规定

区域	C	S
CIO 区	CIO 0000 ～ CIO 6143	CIO 0000 ～ CIO 6134
工作区	W000 ～ W511	W000 ～ W502
保持位区	H000 ～ H511	H000 ～ H502
辅助位区	A000 ～ A438 A448 ～ A959	A000 ～ A438 A448 ～ A950
定说器区	T0000 ～ T4095	T0000 ～ T4086
计数器区	C0000 ～ C4095	C0000 ～ C4086
DM 区	D00000 ～ D32767	D00000 ～ D32758
无区号 EM 区	E00000 ～ E32767	E00000 ～ E32758
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	En_00000 ～ En_32758 (n = 0 ～ C)
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 码间接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	仅指定值	#0000
数据寄存器	DR0 ～ DR15	---
索引寄存器	---	
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

说明

如下表所示，STUP(237) 将 10 个字数据从 S ～ S+9 写到指定单元地址的单元通信设置区。如下表所示，当 S 指定为常数 # 0000 时，相应端口的通信设定设置为缺省值。

单元地址	单元	端口号	串行端口	串行端口通信设置区
00 Hex	CPU 单元	1 hex	端口 1	PLC 设置中的外围端口的通信参数。
		2 hex	端口 2	PLC 设置中的 RS-232C 端口的通信参数。
单元号 + 10 Hex	串行通信单元（CPU 总线单元）	1 hex	端口 1	从 D30000+100 ×单元号开始的 10 字
		2 hex	端口 2	从 D30000+100 ×单元号 +10 开始的 10 字
E1 Hex	串行通信板 （插入板）（仅适用于 CS 系列）	1 hex	端口 1	从 D32000 开始的 10 字
		2 hex	端口 2	从 D32010 开始的 10 字

执行 STUP(237) 时，相应的端口参数改动标志 (A61901 ， A61902 或 A619 ～ A636) 置 ON。这些标志保持 ON，直至完成参数修改。

根据指定条件在操作中用 STUP(237) 修改端口的通信参数。例如通过调制解调器联接执行通信序列，当满足指定条件时，可使用 STUP(237) 从上位机改变上位机链接通信的监视和编程。

CPU 单元之间的差别

如果在使用 STUP(237) 改变通信参数后，PLC 置 OFF，然后又重新置 ON，新参数会根据不同 CPU 单元保持，或回复到原参数。

CPU 单元	通信参数状态
CS1-H, CJ1-H, CJ1M, 或 CS1D	如果 PLC 置 OFF，然后又重新置 ON，通信参数回复到被 STUP(237) 改变之前的设置。
CS1	如果 PLC 置 OFF，然后又重新置 ON，被 STUP(237) 改变的通信参数保持不变。

标志

名称	标记	操作
错误标志	ER	C 值不在范围时置 ON。 如果对一个通信参数改变标志已经置 ON 的端口执行 STUP(237)，置 ON。 如果在一中断任务中执行 STUP(237)，置 ON。 其它情况置 OFF。

注意

通信参数包括协议模式、波特率、数据格式（协议宏传送方法和协议宏最大通信数据长度）和其他参数。串行端口设计的详细资料参见 *CS/CJ 系列可编程控制器操作手册 (W339) 或 CS/CJ 系列串行通信板和串行通信单元操作手册 (W336)*。

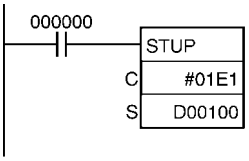
相关标记和字

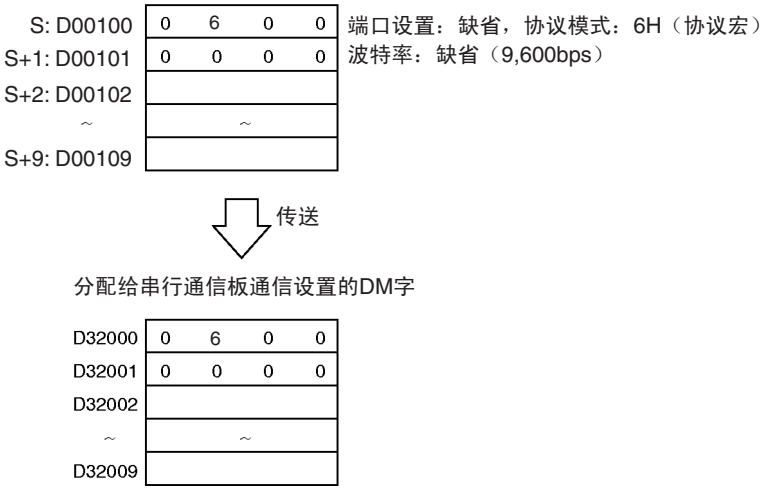
在执行 STUP(237) 时，可根据需要使用以下标志。这些标志在辅助区内。

名称	地址	内容
外设端口参数改动标志	A61901	外围端口通信参数改变时，置 ON。
RS-232C 参数改动标志	A61902	RS-232C 端口通信参数改变时，置 ON。
串行通信单元 1 ～ 15 上的 1 ～ 4 端口参数改动标志	A620 第 1 位～第 4 位～ A635 第 1 位～第 4 位	串行通信单元的端口通信参数改变时，置 ON。
串行通信板上的 1 ～ 4 端口参数改动标志 （仅适用于 CS 系例）	A63601 ～ A63604	串行通信板上的端口通信参数改变时，置 ON。

例

在下例中，当 CIO000000 置 ON 时，串行通信板 （插入板）的串行端口 1 的通信参数变为从 D00100 ～ D00109 的 10 字中包含的设置。在本例中，将协议模式改变位协议宏模式。





3-25 网络指令

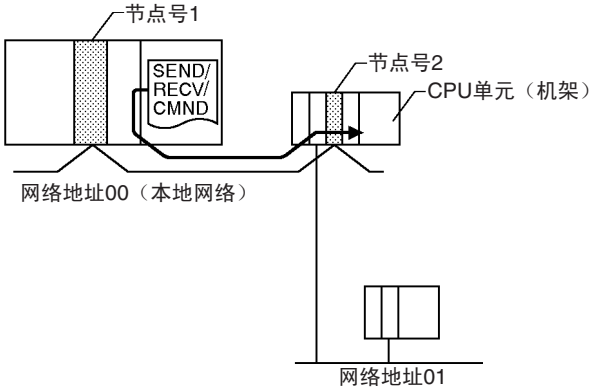
3-25-1 关于 SYSMAC NET 链接 /SYSMAC LINK 操作

网络指令可分为两类，SEND(090)/RECV(098) 和 CMND(490) 。这些指令用在网络中的各单元（CPU 单元， CS1 CPU 总线单元和计算机）之间传送数据或控制操作，例如修改操作模式。

指令	消息内容	操作
SEND(090)/ RECV(098)	数据传送 / 接收命令（FINS 命令）	CPU单元：SEND(090)或 RECV(098)；数据传输；数据接收；其它设备：CPU单元，CS1 CPU总线单元或计算机
CMND(490)	任意命令（FINS 命令）	CPU单元：CMND(490)；命令发送；响应返回；其它设备：CPU单元，CS1 CPU总线单元或计算机

由网络指令执行的命令，及“FINS”命令，用在 FA 控制设备之间通信。（FINS 命令的细节参考 *CS/CJ 系列通信命令参考手册*）。只要指定网络地址、节点号和目标单元的单元号，就可以用 FINS 命令与任意网络上的任何单元或在本身 CPU 机架上单元通信 （用命令 / 响应格式）。

在下例中，通过在网络地址 00 中的节点号 2，把 FINS 命令传送到 CPU 单元。



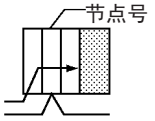
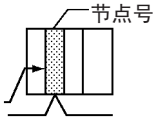
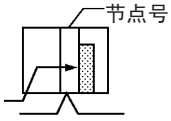
1,2,3... 1. 网络地址：
网络的地址 （本地网络 = 00）

2. 节点号
网络中的逻辑地址

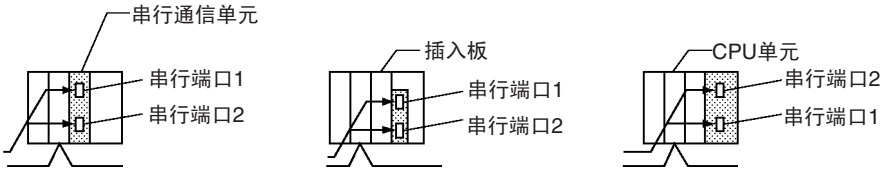
3. 单元号
目标单元的单元号
   a) CPU 单元：00
   b) CPU 总线单元： 单元号＋ 10 （十六进制）
   c) 特殊 I/O 单元 （除 C200H 系列特殊 I/O 单元外）：
   单元号＋ 20 十六进制

d) 插入板 （仅适于 CS 系列）：
E1 十六进制

e) 计算机： 01

单元号 （十六进制）	目标设备
00	
单元号 +10	
E1	
01	

注 也可以在目标设备内指定一个串行端口 (1 ～ 4)。（端口 0 指设备本身。）

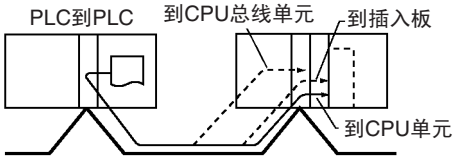


网络通信类型

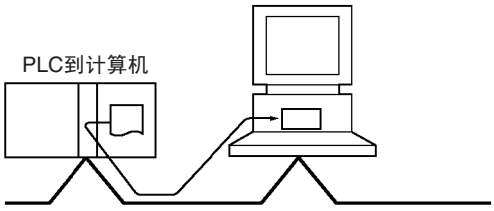
下面的例子描述了三类网络通信：PLC 与网络中其它设备的通信，PLC 与网络中其它设备上的串行端口通信，及与通过上位链接连接的上位机通信。

与网络中其它设备通信

下例为 PLC 与其它 PLC 上的设备 （CPU 单元， CPU 总线单元或插入板）通信。详情参见网络操作手册 （控制器链接或以太网）。

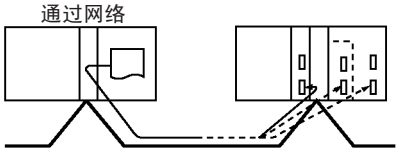


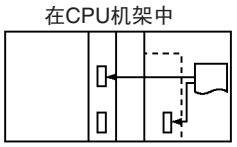
本例显示了 PLC 到个人计算机的通信。



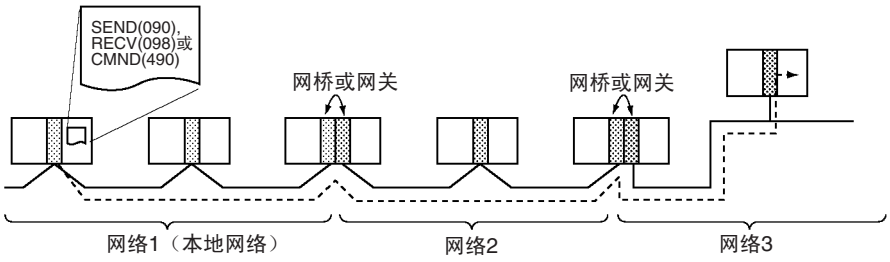
与网络中一个串行端口通信

下面的例子说明了 PLC 与网络中设备的串行端口的通信。第一个例子为与其它 PLC 设备（CPU 单元，CPU 总线单元或插入板）的串行口通信，第二个例子为与本 CPU 机架中串行端口的通信。





注 通信跨越包括本地网络在内的三层网络 （本地网络为通信发起的网络）。



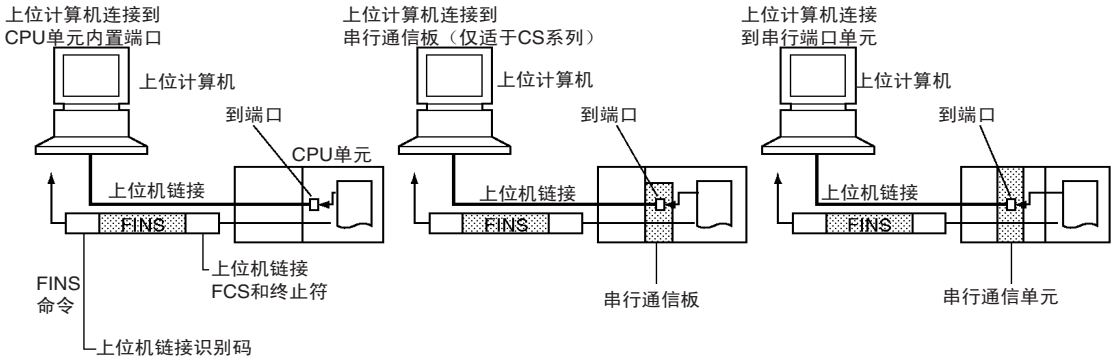
为了在网络中通信，需要在每个 PLC 的 CPU 单元中登录路由表，指明数据传输到所需节点的路由。每一个路由表由一个本地网络和一个中继网络表组成。

1,2,3...

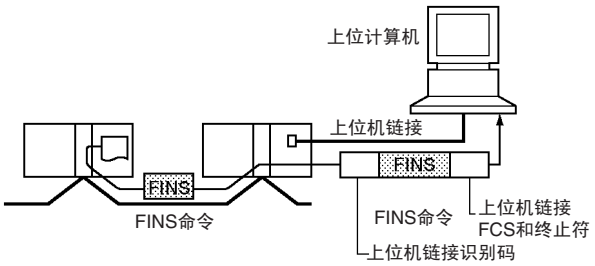
1. 本地网络表
   本表显示了与本地 PLC 连接的节点的单元号和网络地址。
2. 中继网络表
   本表显示了未连接到本地 PLC 的目标网络的第一个中继节点的节点号和网络地址。

与上位机的通信 （上位机链接）

通过发布 SEND(090)/RECV(098) 或 CMND(490) 指令到设定为上位机链接模式的串行端口，将必要的上位机链接识别码和终止符附于 FINS 命令上，然后将命令送到上位计算机。

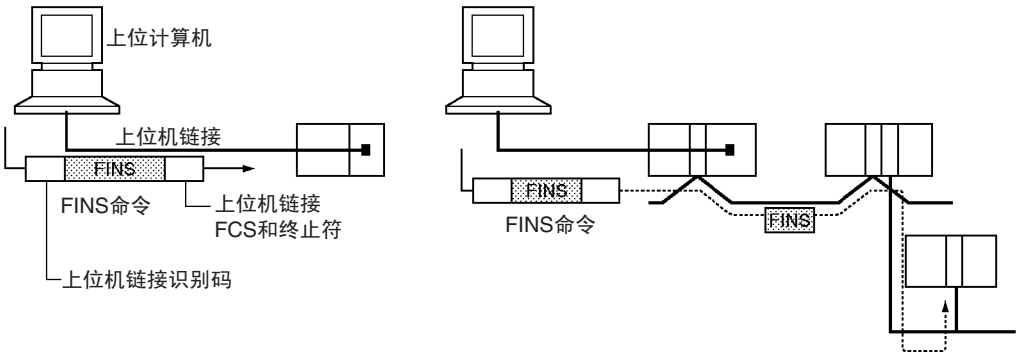


注 上位机链接通信可以通过网络发送。此时，FINS 命令正常穿越网络。当命令到达上位机链接系统时，将必要的上位机识别码和终止符附于 FINS 命令上，然后将命令传送到上位计算机。



与上位计算机的通信 （上位机链接）

可以从上位机发布 FINS 命令到与之相连的 PLC 及网络中的其它设备 （CPU 单元，特殊 I/O 单元，计算机等），此时，将必要的上位机识别码和终止符附于 FINS 命令中。



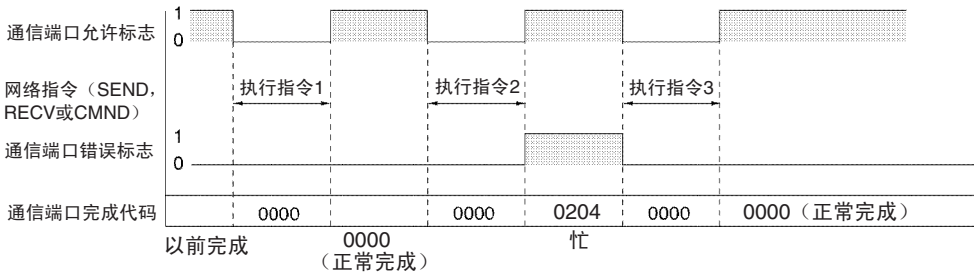
通信标志

通信标志的操作概括如下：

t正在通信时通信端口允许标志复位为0，通信完成时（正常或异常）置1。

t 通信端口错误标志的状态保留至数据的下一次传送或接收。

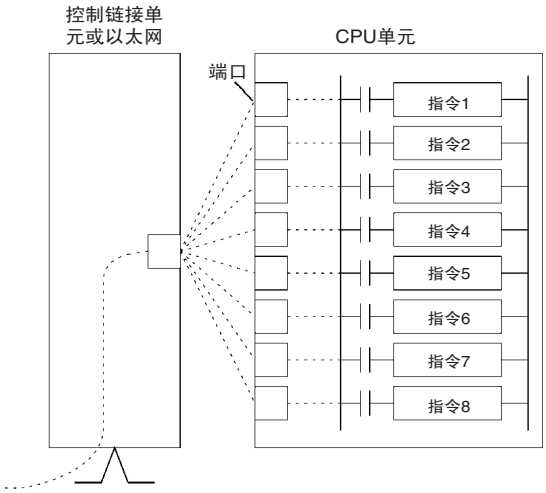
t 即使前面的操作有错误产生，在下一次传送或接收数据时也会将通信端口错误标志复位为 0 。



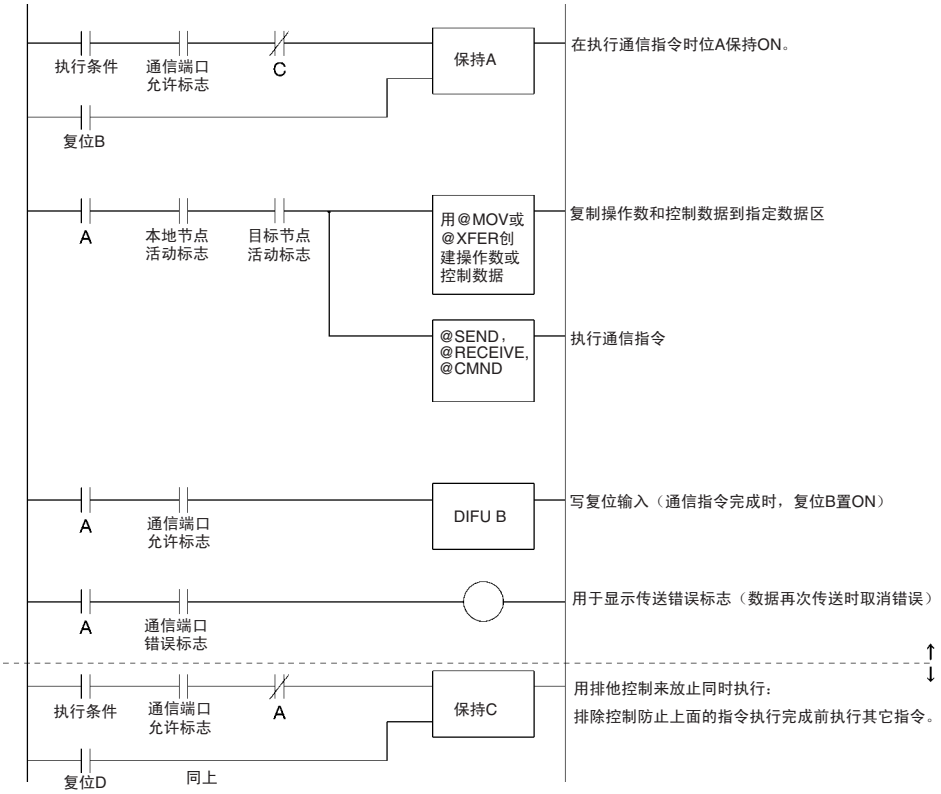
关于通信端口号

由于有 8 个逻辑通信端口，因此可以同时执行 8 条通信指令。每个通信端口 一次只能执行一个指令。当执行 8 条以上的指令时，必须使用排他控制。

网络指令（SEND(090)/RECV(098) 和 CMND(490)）和协议宏指令 (PMCR(260)) 共享 8 个通信端口。不要在同一时间为两条指令指令指定相同的端口号。



下图为排除控制实例。



通信端口的自动分配

■ 概述

下列指令全部使用 0 ～ 7 中的一个通信端口 （逻辑端口）

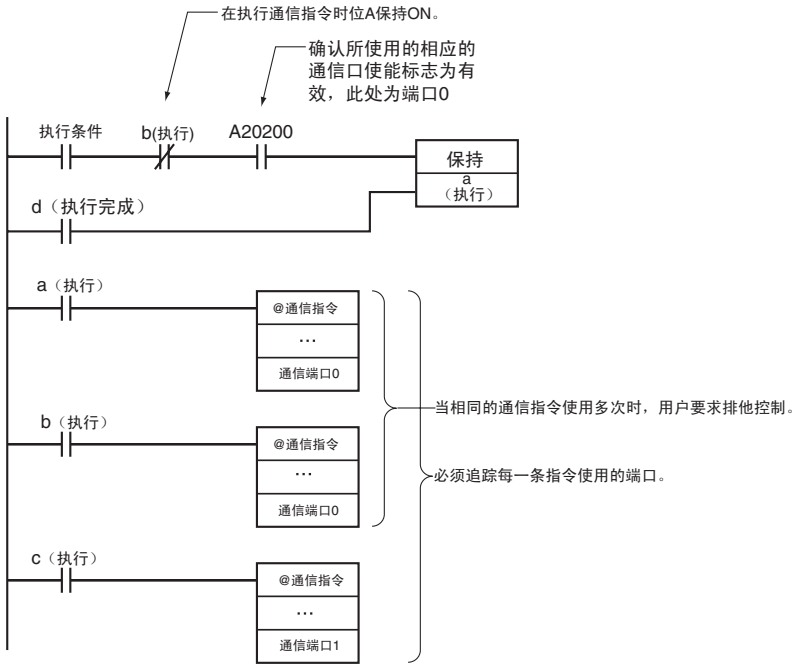
- 网络通信指令：SEND(090)， RECV(098) 和 CMND(490)
- 协议宏：PMCR(260)

本节中，上述所有指令参见通信指令。

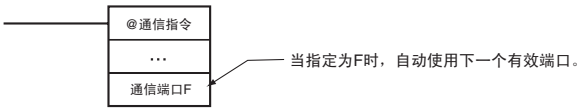
同一时间每个通信端口仅有一条指令使用。在使用通信端口前，必须先执行下列步骤：

- 编程时，必须保持用于操作数指定的通信端口跟踪。
- 在梯形图中，在使用端口前必须确保通信端口的有效性。

前面编程要求实例



批号 020601 或其后（2002 年 6 月 1 日以后生产）的 CS10H，CJ1-H，CS1D CPU 单元而言，端口号可以指定为“F”代替 0 ～ 7 来自动分配通信端口，即自动使用下一个打开的端口。



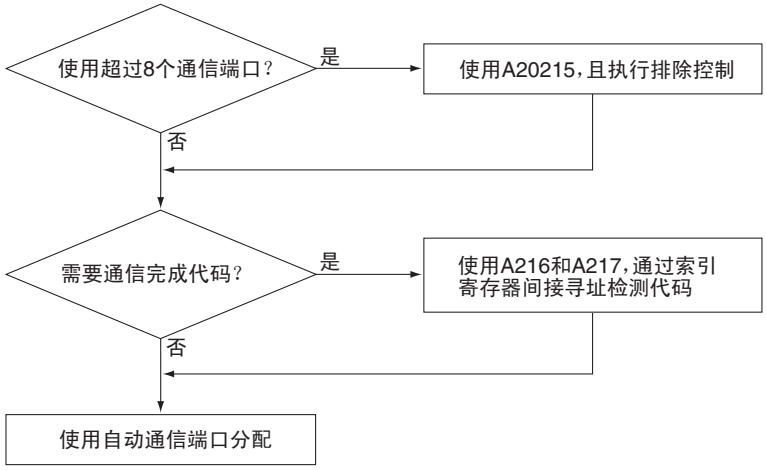
这就使得编程人员编程时不必跟踪通信端口。指定特定端口号和自动分配端口号之间的差别如下表所示。

项目	指定特定端口号	自 动 分 配
在控制数据中说明通信端口号	0 ～ 7	F
排除控制	需要	除非同时需要 8 个以上的通信端口，否则不需要。
标志应用	使用加载或加载非，具有对应与指定通信端口的标志。	用 A218 （使用的通信端口号）使用 TST(350) 或 TST(351)。
网络通信完成代码	访问用户指定通信端口完成代码。	通过使用存贮于 A216 和 A217 （网络通信完成代码存贮地址）中的 I/O 内存地址和索引寄存器间接寻址得到完成代码。

■ 自动分配通信端口时辅助区域使用的位和字。

地址	位	名称	说明
A202	15	网络通信端口分配允许标志	如果有用于自动分配的通信端口时置 ON。该标志可用于确认，是否在执行通信指令前，所有 8 个通信端口已经被分配。
A214	00 ～ 07	网络通信结束后第一个循环标志	通信完成后每个标志只置 ON 一个循环。位 00 ～ 07 对应于端口 0 ～ 7。使用存贮在 A218 中的所用端口号来确认得到的标志。 注： 直到通信指令执行后的下一个循环，这些标志才有效。取得标志至少延迟一个循环。
	08 ～ 15	不使用	
A215	00 ～ 07	网络通信错误后第一个循环标志	通信错误后每个标志只置 ON 一个循环。位 00 ～ 07 对应于端口 0 ～ 7。使用存贮在 A218 中的所用端口号来确认得到的标志。 注： 直到通信指令执行后的下一个循环，这些标志才有效。取得标志至少延迟一个循环。
	08 ～ 15	不使用	
A216 和 A217	---	网络通信完成代码存贮地址	通信指令的完成代码自动存贮在这些字给出的 I/O 内存地址的地址单元内。将该地址置于索引寄存器中，通过索引寄存器使用间接寻址，来得到通信完成代码。
A218	---	使用的通信端口号	当通信指令执行时，使用的通信端口号存贮在该字中。 0000 ～ 0007H 对应于通信端口 0 ～ 7。

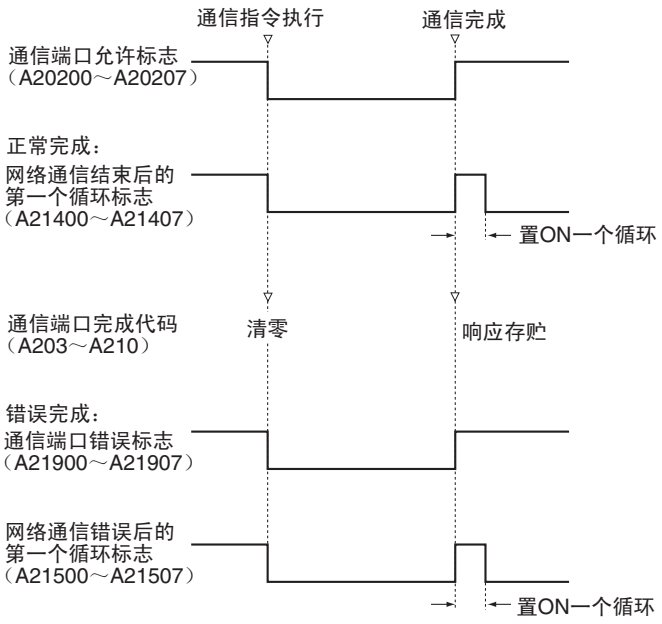
注 1. 使用如下所示的流程图来确定是否使用网络通信端口分配允许标志 (A20215) 和网络通信完成代码存贮地址 （A216 和 A217）。



2. 用于用户指定通信端口的辅助区域位和字如下表所示。

地址	位	名称	说明
A202	00 ～ 07	通信端口允许标志	当在相应的端口号可以执行通信指令时，置 ON。位 00 ～ 07 对应于通信端口 0 ～ 7。 通信完成可以通过监视标志置 ON 的时间来确认。通信指令开始执行时标志置 OFF。
A203 ～ A210	---	通信端口完成代码	当通信指令已经执行，这些字包含相应端口号的完成代码。字 A203 ～ A210 对应于通信端口 0 ～ 7。
A219	00 ～ 07	通信端口错误标志	通信指令执行过程中产生错误时置 ON。当标志置 ON 时，检查 A203 ～ A210 中的完成代码以检测排除故障。 执行正常完成，则置 OFF。位 00 ～ 07 对应于通信端口 0 ～ 7。

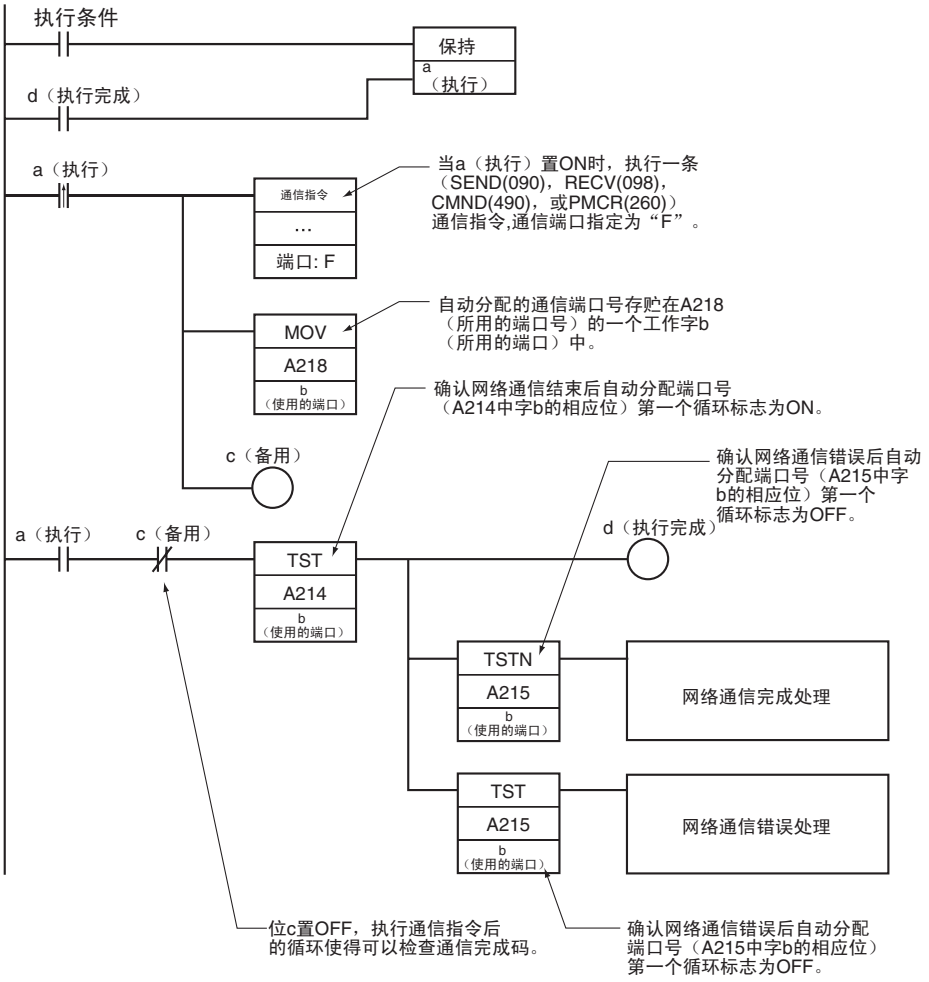
标志 / 字 操作



■ 应用方法

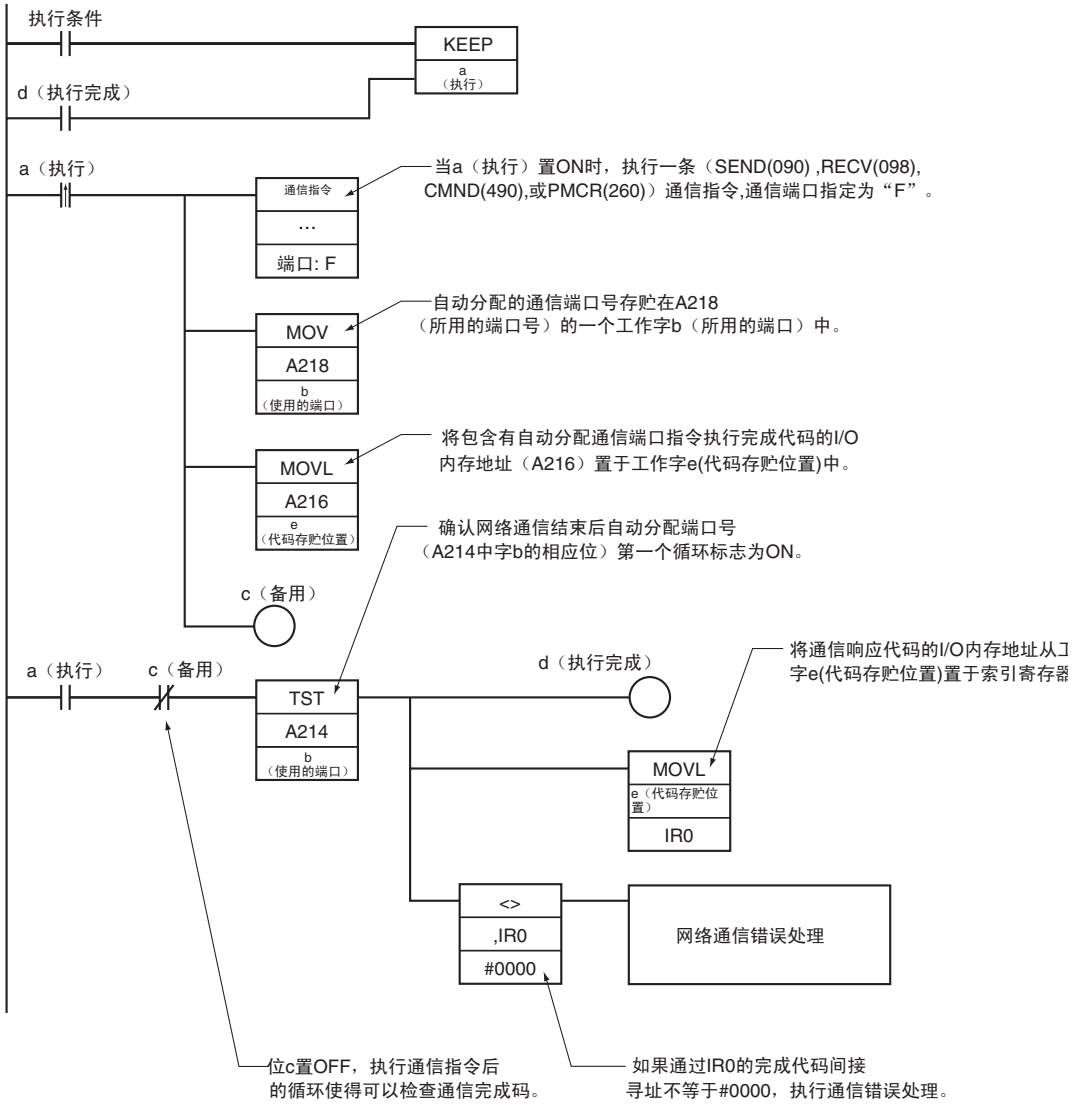
为了使用自动端口分配，将通信端口号设置为 “F”，则程序如下所示。

执行通信指令后的完成和处理错误



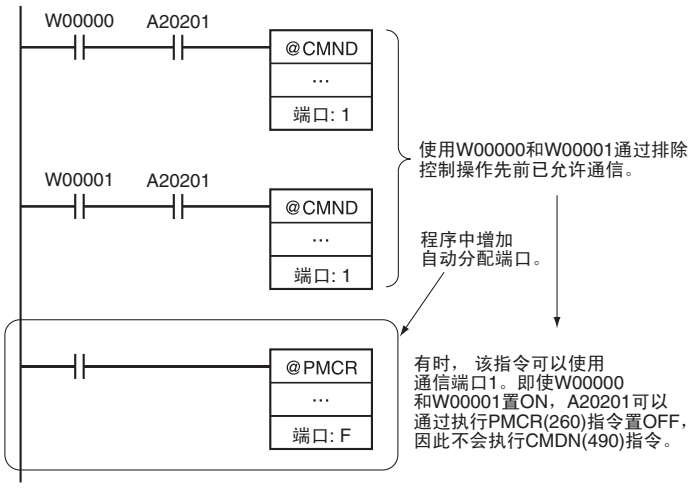
执行通信指令后获取完成代码

发生错误时，完成代码通常用于检测排除这些错误。然而，完成代码 0000H 也可以用于确认通信已经正常完成。



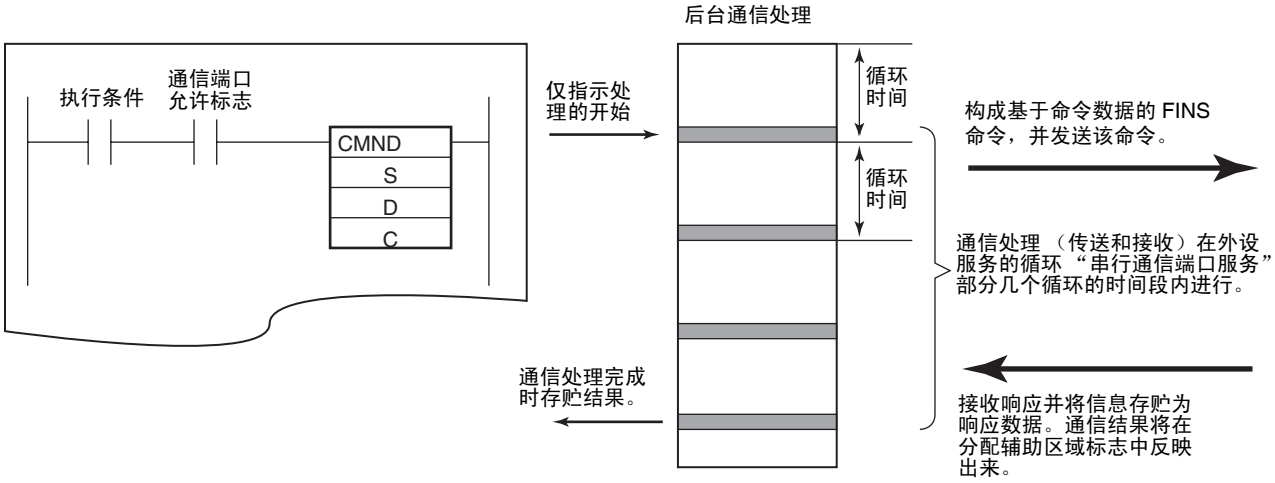
注 用户指定通信端口号和自动指定通信端口号都可以用在同一程序中。然而，用户指定的通信端口号可以用于自动分配。因此当在现行程序中增加使用自动端口分配的通信指令时 （如下例所示），对程序仔细检查是非常重要的。

编程示例



对网络指令的执行计时

网络指令在其执行条件建立时开始通信处理。实际通信处理的执行以外设服务的 “串行通信端口服务”部分为基础。

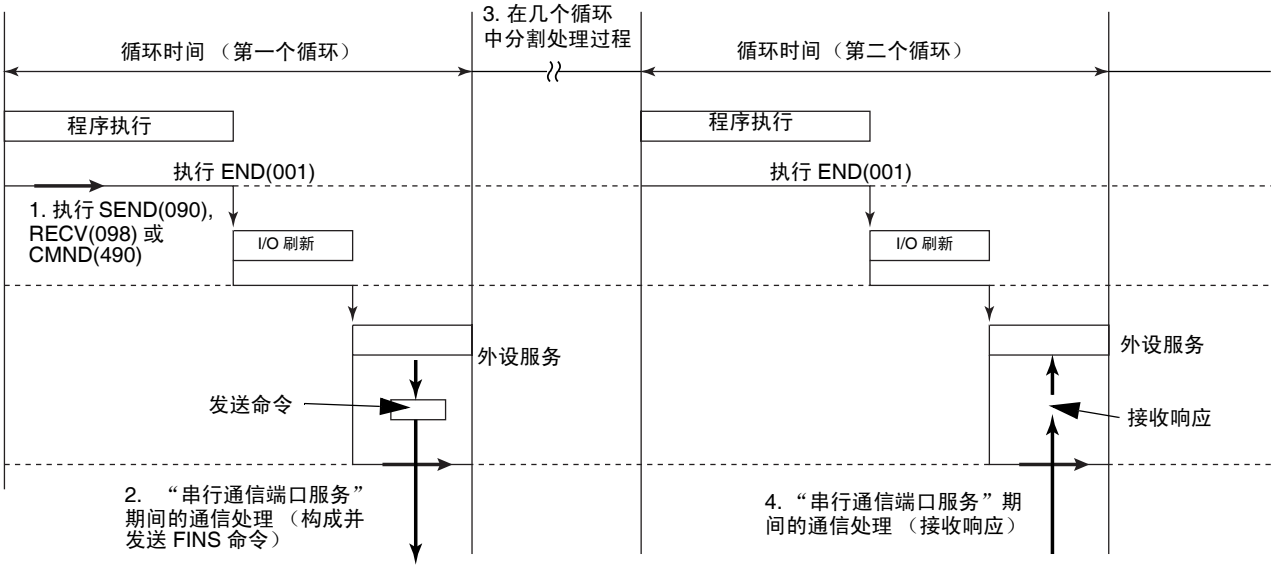


通信处理执行如下：

1. 执行条件建立后，如果相应的通信端口允许标志（A20200 ～ A20207）置 ON，系统执行下列处理：
   - 端口的通信端口允许标志和通信端口错误标志（A21900～A21907）置 OFF。
   - 端口的通信端口完成代码 （A203 ～ A210）设为 0000 。
   - 读入控制字（始于C），且启动通信处理（发送FINS命令或接收响应）。
2. 在循环中外设服务的 " 串行通信端口服务 " 部分，系统构成基于操作数（见注）的 FINS 命令，并且将 FINS 命令发送到通信单元或其它目标节点。

注 执行SEND(090)时，读入S和D的内容，构成数据传送的FINS命令。
执行 RECV(098) 时，读入 S 的内容，构成数据接收的 FINS 命令。
执行 CMND(490) 时，读入 S 的内容，构成响应 FINS 命令。

3. 如果发送处理在“串行通信端口服务”周期的可用时间内不能完成，该处理会在下一个循环的串行通信端口服务周期继续进行。
4. 当返回响应时，系统执行下列处理过程：
   - 用响应数据刷新网络指令给定的目标字。
   - 端口的通信端口允许标志置 ON。
   - 刷新通信端口错误标志 （A21900 ～ A21907）和通信端口完成代码（A203 ～ A210）。

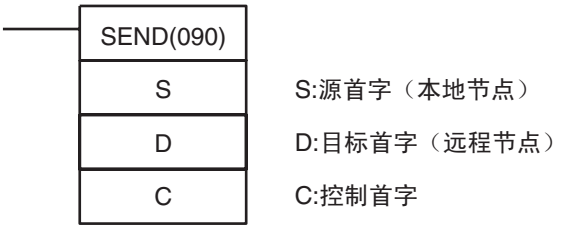


3-25-2 网络发送：SEND(090)

用途

发送数据到网络的节点。

梯形图符号



变化

变化	ON 条件时每个循环执行	SEND(090)
	上升沿微分执行一次	@SEND(090)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

C: 控制首字

C ～ C ＋ 4 这 5 个控制字指定传送字数、目标和其它设置，如下表所示。

字	位 00 ～ 07	位 08 ～ 15
C	字数：0001 ～允许最大值[1] （4 位 16 进制数字）	
C+1	目标网络地址： 00 ～ 7F (0 ～ 127)[2]	位 08 ～ 11: 串行端口号[3] （物理端口） 1 hex: 端口 1 2 hex: 端口 2 （不设置 0， 3 或 4） 第 12 ～ 15 位：总为 0
C+2	目标单元地址：00 ～ FE[4]	目标节点地址：00 ～允许最大值[5]
C+3	重复次数：00 ～ 0F (0 ～ 15)	位 08 ～ 11: 通信端口号 （内部逻辑端口）：0 ～ 7，自动分配 F[6] 第 12 ～ 15 位：响应设置 0: 响应请求 8: 无响应请求[7]
C+4	响应监视时间：0001 ～ FFFF （0.1 ～ 6553.5 秒） （缺省设置 0000 设定监视时间位 2 秒）	

注

1. 允许的最大字数由所用网络决定。对Controller Link而言，允许范围为0001 ～ 03DE （1 ～ 990 字）。
2. 在本地网内传送时，将目标网络地址设定为00。当安装两个或两个以上CPU总线单元时，网络地址为具有最小单元号的单元的单元号。
3. 端口号 1 和 2 指明了目标设备的串行端口 1 和 2 （物理端口）。

   当将数据传送到通过上位机链接连接的计算机时，指定与计算机连接的串行端口。该设置用于将 C ＋ 2 的位 0 ～ 7 目标单元地址设置的组合。

单元地址	单元	串行端口号	串行端口
00 Hex	CPU 单元	1 Hex	内置 RS-232C 端口
		2 Hex	外设端口
10 Hex + 单元号	串行通信单元 （CPU 总线单元）	1 Hex	端口 1
		2 Hex	端口 2
E1 Hex	串行通信板 （插入板）（仅适于 CS 系列）	1 Hex	端口 1
		2 Hex	端口 2

4. 单元地址指示了单元，如下表所示。

单元	单元地址设置
CPU 单元	00 hex
CPU 总线单元	10 hex + 单元号
特殊 I/O 单元 （C200H-系列特殊 I/O 单元除外）	20 hex + 单元号
插入板 （仅适于 CS 系列）	E1 hex
计算机	01 hex
连接到网络的单元 （不必指定单元）	FE hex

5. 最大节点号由所用网络决定。对 Controller Link 而言，允许范围为 00 ～ 20H（0 ～ 32）。在向所有节点广播时，将目标节点号设为 FF ；在向本地节点传送时设为 00。
6. 通信端口号 （逻辑端口）自动分配的使用详情，参见*通信端口的自动分配 第 872 页*。
7. 目标节点号被设定为 FF（广播传送）时，即使第 12 ～ 15 位被设为 0，也没有响应。

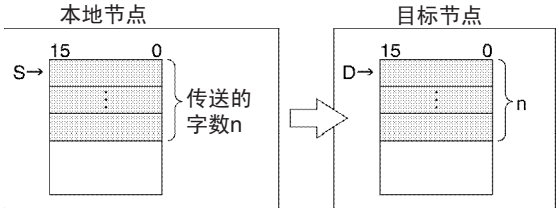
操作数规定

区域	S	D	C
CIO 区	CIO 0000 ～ CIO 6143		CIO 0000 ～ CIO 6139
工作区	W000 ～ W511		W000 ～ W507
保持位区	H000 ～ H511		H000 ～ H507
辅助位区	A000 ～ A959		A000 ～ A955
定时器区	T0000 ～ T4095		T0000 ～ T4091
计数器区	C0000 ～ C4095		C0000 ～ C4091
DM 区	D00000 ～ D32767		D00000～D32763
无区号 EM 区	E00000 ～ E32767		E00000～E32763
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		En_00000 ～ En_32763 (n = 0 ～ C)
二进制间接 DM/EM 寻址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 码直接 DM/EM 寻址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---		
数据寄存器	---		

区域	S	D	C
索引寄存器	---		
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15		

说明

SEND(090) 通过 PLC 的 CPU 总线或网络将从字 S 开始的数据传送到指定设备中从 D 开始的地址中。传送的字数在 C 中指定。



如果目标节点号被设为 FF，数据将向指定网络的所有节点广播。这就是广播传送。

如果需要响应 （C+3 的为 12 ～ 15 设置为 0），但在响应监视时间内未收到响应，数据可最多传输 15 次 （在 C+3 的位 0 ～ 3 中设置重试次数）。广播传输没有响应或重试。

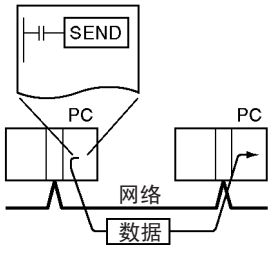
SEND(090) 可以用来把数据传输到目标设备的特定串行端口，还可以把数据传输到设备本身。

数据可传输到与 PLC 串口相连的上位机上 （当设置成 Host Link 模式），也可以传输到通过 Controller Link 或以太网相连的 PLC 或计算机上。

如果在 C ＋ 3 中指定的通信端口的允许标志是 ON，当 SEND(090) 执行时，响应的通信端口允许标志（00 ～ 07 端口：A20200 ～ A20207）和通信端口错误标志 （00 ～ 07 端口：A21900 ～ A21907）将变为 OFF， 000 将写入到包含完成代码的字 （00 ～ 07 端口：A203 ～ A210）中。一旦设置了标志，数据将传输到目标节点。

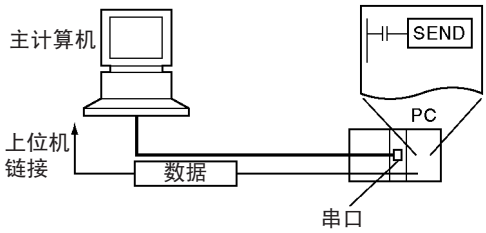
通过网络传输

SEND(090) 可以用来将数据从 PLC 传输到通过 Controller Link 或以太网相连的 PLC 或计算机中指定数据区域中。



通过 Host Link 传输

如果 CPU 单元的内置串口、串行通信板或串行通信单元是 Host Link 模式，并且一对一与主计算机连接，当下次 PLC 有权传输数据时， SEND(090) 可以将数据从 PLC 传输到主计算机。它也可以把数据传输到与网络中别处的 PLC 相连的其它主计算机上。



如果把 SEND(090) 发布给 CPU 单元的串口、串行通信板 （仅适于 CS 系列）或串行通信单元，一个命令将从串口发送到主计算机。该命令是一个封装在 Host Link 数据头和数据尾之间的 FINS 消息。发布的 FINS 命令是一个内存区写命令 （命令代码为 0102），上位机链接头代码是 0FH。

必须在计算机中编写程序来处理接收到的命令 （封装在上位机链接数据头和数据尾之间的 FINS 命令）。

如果目标串口在本地 PLC 上，在 C+1 中设置网络地址为 00 （本地网络），在 C+2 中设置节点地址为 00（本地 PLC），并设置单元地址为 00（CPU 单元），E1 （插入板 （仅适于 CS 系列））或单元号 +10H （串口单元）。

标志

名称	标记	操作
错误标志	ER	如果 C+1 指定的串口号不在 00 ～ 04 的范围内，置 ON。 如果 C+3 指定的通信端口号的通信端口允许标志为 OFF，置 ON。 其它情况，置 OFF。

下表显示了辅助区域的有关位和标志。

名称	地址	操作
通信端口允许标志	A20200 ～ A20207	这些标志变 ON，说明可以在相应端口上 (00 ～ 07) 执行网络指令，包括 PMCR(260) 当网络指令被执行时，相应端口的允许标志变 OFF，当指令完成时，它又变 ON。
通信端口错误标志	A21900 ～ A21907	这些标志如果变 ON，说明在网络指令执行过程中相应的端口发生了错误。 这些标志的状态将保持到下一个网络执行时为止。当下一个指令执行时，即使以前发生了错误，这些标志仍将变 OFF。
通信端口完成代码	A203 ～ A210	这些字包含有相应端口 (00 ～ 07) 网络指令执行的完成代码。 当网络指令被执行时，相应的字内容为 0000，当指令结束时将写入完成代码。这些字在指令执行时被清除。

注意

如果 C+3 中指定端口号的通信端口允许标志为 OFF，指令将等同于 NOP(000) 且不被执行。这种情况下错误标志将变 ON。

当 EM 区当前区号的地址被指定为 D，被传输的数据将写入到目标节点的当前 EM 区。

当数据将要传输到本地网络之外时，用户必须在每个网络的 PLC（CPU 单元）中注册路由表。（路由表指明到连接目标节点的其它网络的路径）。

关于网络通信的完成代码详细情况参阅在*CS/CJ串行通信命令参考手册(W342)* 中的 FINS 命令响应代码。

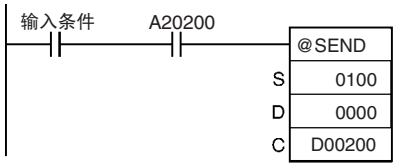
一个通信端口一次只能执行一个网络指令。为了确保在端口忙时 SEND(090) 不被执行，编程中将端口的通信允许标志(A20200～A20207)作为一个常开条件。

通信端口的号码 00 ～ 07 被网络指令和 PMCR(260) 所共享，因此如果 SEND(090) 和 PMCR(260) 指令正在使用相同的端口号时，它们不能被同时执行。

噪音和其它的因素能导致传输或响应破坏和丢失，所以我们建议把重试次数设成非 0 值，这将使得当在响应时间内没收到应答时再次执行 SEND(090)。

例 1

在下例中，当输入条件和 A20200 （端口 0 的通信端口允许标志）为 ON，从 CIO 100 ～ CIO 109 的十个字传输给网络 0 中节点 3 串行通信单元地址 10（十六进制）的单元端口 1 相连的主计算机。

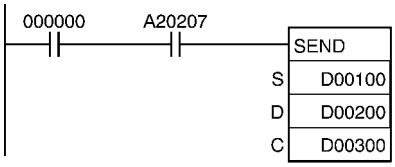


C	D00200	0 0 0 A	要传送的字数：10个字
C+1	D00201	0 1 0 0	传输到网络0和串行通信板的端口1
C+2	D00202	0 0 1 0	节点号0，单元地址10
C+3	D00203	0 0 0 0	请求应答，端口号0，无重试
C+4	D00204	0 0 0 0	应答监视时间：2秒（0000：缺省值）

可以在主机编程接收数据并发送响应信号。

例 2

在下例中，当 CIO 000000 和 A20207（端口 07 的的通信端口允许标志）为 ON 时， D00100 ～ D00109 的 10 个字被传输到本地网的节点 3 D00200 ～ D00209 的 10 个字中。如果在 10 秒之内未收到响应，数据将传输 3 次。



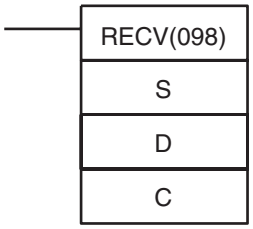
C:	D00300	0 0 0 A	要传送的字数：10个字
C+1:	D00301	0 0 0 0	传输到本地网和设备本身
C+2:	D00302	0 3 0 0	节点号3，单元地址00（CPU单元）
C+3:	D00303	0 7 0 3	请求应答，端口号7，重试3次
C+4:	D00304	0 0 6 4	应答监视时间：十六进制0064（10秒）

3-25-3 网络接收：RECV(098)

用途

请求网络中的一个节点传输数据并接收数据。

梯形图符号



S:源首字（远程节点）

D:目标首字（本地节点）

C:控制首字

变化

变化	ON 条件时每次循环执行	RECV(098)
	上升沿微分时执行一次	@RECV(098)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

C: 控制首字

C ～ C+4 这五个控制字指定了要接收的字的数目、传输源和其它的设置，如下表所示。

字	位 00 ～ 07	位 08 ～ 15
C	字数：0001 ～允许最大值[1]（4 位十六进制）	
C+1	源网络地址： 00 ～ 7F (0 ～ 127)[2]	位 08 ～ 11：串行端口号（物理地址） 1hex: 端口 1 2hex: 端口 2 （端口 0， 3 或 4 不设置） 位 12 ～ 15：总是 0。
C+2	源单元地址[4]	源节点地址： 00 ～允许最大值[5]
C+3	重复次数：00 ～ 0F (0 ～ 15)	端口号：00 ～ 07 （F：自动分配）[6] 应答固定为“必需”
C+4	应答监视时间：0001 ～ FFFF（0.1 ～ 6553.5 秒） （缺省设置 0000，监视时间 2 秒）	

注

1. 允许的最大字数由使用的网络决定，对于控制器链接，允许范围是 0001 ～ 03DE（1 ～ 990 字）。
2. 把传输源地址设置为 00 时指定了一个本地网络的传输源。当安装了 2 个或 2 个以上的 CPU 总线单元时，网络地址将是最低的单元号。
3. 端口号 1 和 2 指明了目标设备的串口 1 和 2（物理端口）。

单元地址	单元	串口号	串口
00 Hex	CPU 单元	1 Hex	内置 RS-232C 端口
		2 Hex	外部端口
10 Hex + 单元号	串行通信单元（CPU 总线单元）	1 Hex	端口 1
		2 Hex	端口 2
E1 Hex	串行通信板（内置板）（仅 CS 系列）	1 Hex	端口 1
		2 Hex	端口 2

4. 单元地址指明了单元，如下表所示。

单元	单元地址设置
CPU 单元	00 hex
CPU 总线单元	10 hex + 单元号
特殊 I/O 单元（除 C200H 系列特殊 I/O 单元）	20 hex + 单元号
内置板（仅适用 CS 系列）	E1 hex
计算机	01 hex
连接到网络的单元（不必指定单元）	FE hex

5. 最大节点号由使用的网络决定。对于控制器链接，允许范围是 (00 ～ 20) hex(0 ～ 32)，把源节点号设为 00 将在本地节点内传输数据。

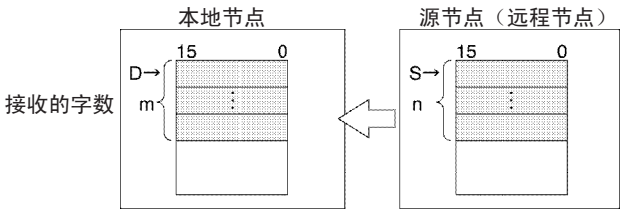
6. 详细情况参考*872*页的*通信端口的自动分配*中关于通信端口号（本地端口）的自动分配。

操作数规定

区域	S	D	C
CIO 区	CIO 0000 ～ CIO 6143		CIO 0000 ～ CIO 6139
工作区	W000 ～ W511		W000 ～ W507
保持位区	H000 ～ H511		H000 ～ H507
辅助位区	A000 ～ A447 A448 ～ A959	A448 ～ A959	A000 ～ A443 A448 ～ A955
定时器区	T0000 ～ T4095		T0000 ～ T4091
计数器区	C0000 ～ C4095		C0000 ～ C4091
DM 区	D00000 ～ D32767		D00000～D32763
无区号 EM 区	E00000 ～ E32767		E00000～E32763
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		En_00000 ～ En_32763 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---		
数据寄存器	---		
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15		

描述

RECV(098) 请求把以字 S 开始的 C 中指定数目的字从指定的设备传输到本地 PLC。数据通过 PLC 的 CPU 总线或通过网络接收，并写入到 PLC 中以 D 开始的数据区。



RECV(098) 要求有响应，因为响应包含要接收的数据。如果在 C+4 中设置的应答监视时间内没收到应答，数据传输请求重复达 15 次 （重试次数在 C+3 的 0 位～ 3 位设置）。

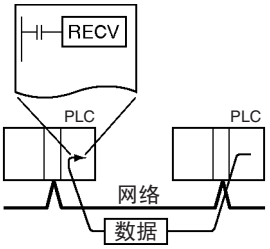
RECV(098) 可以用于请求从源设备的特定串口传输数据，以及设备本身。

可以从连接到 PLC 串口的主机上接收（数据当设置成上位机链接模式），也可从通过控制器链接或以太网相连的 PLC 或计算机上接收数据。

当 SEND(090) 执行时，如果在 C+3 中的指定的通信端口的允许标志为 ON，响应通信端口允许标志（端口 00 ～ 07：A20200 ～ A20207）和通信端口错误标志（端口 00 ～ 07：A21900 ～ A21907）将变为 OFF，并将 0000 写到存储完成代码的字中 （端口 00 ～ 07：A203 ～ A210）。一旦标志被设定，数据将从目标节点接收。

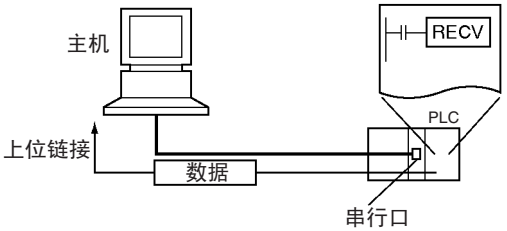
通过网络传输

RECV(098)可接收从通过控制器链接网或以太网连接的PLC或计算机的指定数据区发送的数据，并将该数据写入到本地 PLC 的指定数据区中。



通过 Host Link 传输

当 CPU 的内置串口、串行通信板 （仅 CS 系列）或串行通信单元是上位机链接模式，并且与主机一对一连接，当下次 PLC 有权传输数据时，可以执行 RECV(098) 从主机接收数据，它也能从与网络中别处的PLC相连的其它主机上接收数据。



如果 CPU 单元的串口，串口通信板 （仅 CS 系列）或串行通信单元执行 RECV(098)，串口将发送一个命令到主计算机。该命令是一个封装在上位机链接数据头和数据尾之间的 FINS 消息。FINS 命令是一个内存区读命令（命令代码 0101），上位机链接数据头代码是十六进制 0F。

主计算机中必须创建程序以处理发送命令 （封装在上位机链接数据头和数据尾之间的 FINS 命令）。

如果目标串口在本地 PLC 中，在 C+1 中设置网络地址为 00 （本地网），在 C+2 中设置节点地址为 00（本地 PLC），设置单元地址 00（CPU 单元），E1（内置板仅 CS 系列），或单元号 +10H （串口单元）。

标志

名称	标记	操作
错误标志	ER	如果 C+1 中指定的串口号不在 00 ～ 04 的范围内时置 ON。 如果 C+3 中指定的通信端口号通信端口允许标志是 OFF 时置 ON。 其它情况时 OFF。

下表表示辅助区中的相关位和标志。

名称	地址	操作
通信端口允许标志	A20200 ～ A20207	这些标志变 ON 说明可以在相应端口 (00 ～ 07) 执行包括 PMCR(260) 的网络指令。 当网络指令执行时，相应端口的允许标志变 OFF，当指令完成时，它又变 ON。
通信端口错误标志	A21900 ～ A21907	这些标志如果变 ON，说明在网络指令执行过程中相应的端口发生了错误。 这些标志的状态将保持到下一个网络执行时为止。当下一个指令执行时，即使以前发生了错误，这些标志仍将变 OFF。
通信端口完成代码	A203 ～ A210	这些字包含有相应端口 (00 ～ 07) 网络指令执行的完成代码。 当网络指令被执行时，相应的字包含有 0000，当指令结束时将写入完成代码。这些字在程序开始执行时被清除。

注意

如果 C+3 中指定端口号的通信端口允许标志为 OFF，指令将等同于 NOP(000) 且不被执行。这种情况下错误标志将变 ON。

当 EM 区当前区号的地址被指定为 D，被传输的数据将写入到目标节点的当前 EM 区。

当数据将要传输到本地网络之外时，用户必须在每个网络的 PLC（CPU 单元）中注册路由表。（路由表指明到连接目标节点的其它网络的路径）。

关于网络通信的完成代码详细情况参阅在*CS/CJ串行通信命令参考手册(W342)* 中的 FINS 命令响应代码。

一个通信端口一次只能执行一个网络指令。为了确保在端口忙时 RECV(098) 不被执行，编程中将端口的通信允许标志(A20200～A20207)作为一个常开条件。

通信端口的号码 00 ～ 07 被网络指令和 PMCR(260) 所共享，因此如果 RECV(098) 和 PMCR(260) 指令正在使用相同的端口号时，它们不能被同时执行。

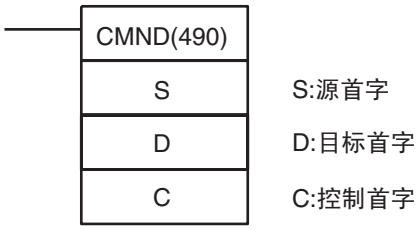
噪音和其它的因素能导致传输或响应破坏和丢失，所以我们建议把重试次数设成非 0 值，这将使得当在响应时间内没收到应答时再次执行 RECV(098)。

3-25-4 传送命令：CMND(490)

用途

发送一个 FINS 命令并接收应答。FINS 命令的详细情况参考 *CS/CJ 系列通信命令参考手册*。

梯形图符号



变化

变化	ON 条件时每次循环执行	CMND(490)
	上升沿微分时执行一次	@CMND(490)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

C: 控制首字

C ～ C+5 这五个控制字指定了要接收的字的数目、传输源和其它的设备，如下表所示。

字	位 00 ～ 07	位 08 ～ 15
C	命令数据的字节：0002 ～允许最大值[1]（4 位十六进制）	
C+1	应答数据的字节：0000 ～允许最大值[1, 2]（4 位十六进制）	
C+2	目标网络地址：00 ～ 07[3]	位 08 ～ 11：串行端口号（物理地址） 1 hex: 端口 1 2 hex: 端口 2 （端口 0，3 或 4 不设置） 位 12 ～ 15：总是 0。
C+3	目标单元地址：00 ～ FE[5]	目标节点号： 00 ～允许最大值[6]
C+4	重试次数：00 ～ 0F (0 ～ 15)	位 08 ～ 11: 端口号（内部逻辑口）： 0 ～ 7 （F：自动分配）[7] 位 12~15：应答设置 0: 应答需要 8: 无应答需要[8]
C+5	应答监视时间：0001 ～ FFFF（0.1 ～ 6553.5 秒） （缺省设置 0000，监视时间 2 秒）	

注

1. 命令和应答数据的允许最大字节数由使用的网络决定，对于控制器链接，最大设置是 07C6（1,990 字节）。
2. 如果实际的应答数据字节数超过 C+1 中的设置，应答数据将不存储。如果实际的应答数据字节数少于 C+1 中的设置，将存储接收到的数据，设置为响应字的其它部分将保持不变。

3. 把目标网络地址设置为 00 时，将在本地网络中进行传输。当安装了 2 个或 2 个以上的 CPU 总线单元时，网络地址将是最低的单元号。
4. 当通过Host Link网将FINS命令发送给主机时，端口号1和2指明了目标设备的串口 1 和 2 （物理端口）。

单元地址	单元	串口号	串口
00 Hex	CPU 单元	1 Hex	内置 RS-232C 端口
		2 Hex	外部端口
10 Hex + 单元号	串行通信单元（CPU 总线单元）	1 Hex	端口 1
		2 Hex	端口 2
E1 Hex	串行通信板 （内置板）（仅 CS 系列）	1 Hex	端口 1
		2 Hex	端口 2

5. 单元地址指明了单元，如下表所示。

单元	单元地址设置
CPU 单元	00 hex
CPU 总线单元	10 hex + 单元号
特殊 I/O 单元 （除 C200H 系列特殊 I/O 单元）	20 hex + 单元号
内置板 （仅适用 CS 系列）	E1 hex
计算机	01 hex
连接到网络的单元（不必指定单元）	FE hex

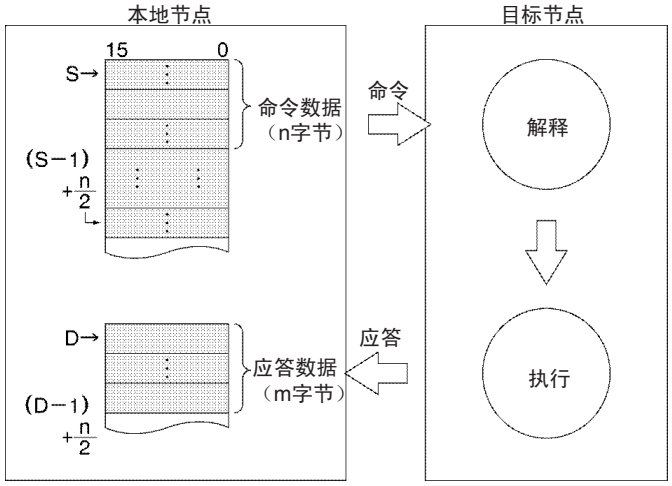
6. 最大节点号由使用的网络决定。对于控制器链接，允许范围是十六进制 00 ～ 20(0 ～ 32)。目标节点设为 FF 将在所有的节点进行广播，目标节点设为 00 将在本地节点内传输。
7. 详细情况参考 *872 页的通信端口的自动分配*中关于通信端口号（逻辑端口）的自动分配。
8. 当目标节点号设置成 FF（广播传输）时，即使位 12 ～ 15 设置为 0，也不会有应答。

区域	S	C	D
CIO 区	CIO 0000 ～ CIO 6143		CIO 0000 ～ CIO 6138
工作区	W000 ～ W511		W000 ～ W506
保持位区	H000 ～ H511		H000 ～ H506
辅助位区	A000 ～ A447 A448 ～ A959	A448 ～ A959	A000 ～ A442 A448 ～ A954
定时器区	T0000 ～ T4095		T0000 ～ T4090
计数器区	C0000 ～ C4095		C0000 ～ C4090
DM 区	D00000 ～ D32767		D00000～D32762

区域	S	C	D
无区号 EM 区	E00000 ～ E32767		E00000～E32762
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		En_00000 ～ En_32763 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767		
常数	---		
数据寄存器	---		
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047, IR0 ～ –2048 ～ +2047, IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15		

描述

CMND(490) 通过 PLC 的 CPU 总线或网络传输以字 S 为起始地址的指定字节数的 FINS 命令数据到指定的设备。应答数据存储到以 D 开始的存储区中。



CMND(490) 能用于把命令数据传输到目标设备的特定串口，以及设备本身。当 FINS 命令代码是 0102 时 （存储区写）， CMND(490) 如同执行 SEND(090)，当代码是 0101 时 （存储区读）， CMND(490) 如同执行 RECV(098)。

执行 CMND(490) 的 CPU 单元能发送一个 FINS 命令到它本身 （在 V1 □以前的 CS 系列的 CS1 CPU 单元除外）。可用下面控制数据设置接收。

- 目标网络地址 （C+2 中的位 00 ～ 07）：00 十六进制 （本地网）
- 串口号 （C+2 中的位 08 ～ 11）：0 十六进制 （不用）
- 目标单元地址 （C+3 中的位 00 ～ 07）：00 十六进制 （CPU 单元）
- 目标节点地址 （C+3 中的位 08 ～ 15）：00 十六进制 （本地节点）
- 重试次数 （C+4 中的位 00 ～ 03）：（这个设置是无效的；设为 0）

- 应答监视时间（C+5中的位00～15）：0000～FFFF 十六进制（但0000将指定为 6553.5 秒，而不是一般的 2 秒）

如果目标节点号被设为 FF，命令数据将广播到指定网络的所以节点上，这就是我们所熟悉的广播传输。

如果要求应答 （C+4 的位 12 ～ 15 设置为 0），但在应答监视时间被没有接收到应答，命令数据可最多重复 15 次 （在 C+3 的位 0 ～ 3 中设置重试次数）。广播传输不存在应答和重试。对于那些不要求应答的指令，把应答设置为 “不需要”。

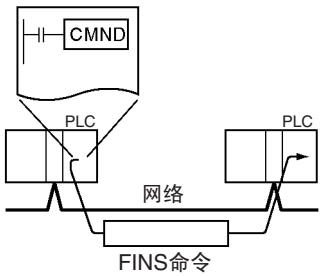
如果应答数据总数超出了 C+1 中设置的应答数据，将发生错误。

FINS 命令数据能够传输到连接到 PLC 串口的主机上 （当设置成上位机链接模式），也可从通过控制器链接或以太网相连的 PLC （CPU 单元，内置板 （仅 CS 系列）， CPU 总线单元）或计算机上传输。

执行 CMND(490) 时，如果在 C+3 中的指定的通信端口的允许标志为 ON，响应通信端口允许标志（端口 00 ～ 07：A20200 ～ A20207）和通信端口错误标志（端口 00 ～ 07：A21900 ～ A21907）将变为 OFF，并将 0000 写到存放完成代码的字中 （端口 00 ～ 07：A203 ～ A210）。一旦标志被设定，命令数据将传输到目标节点。

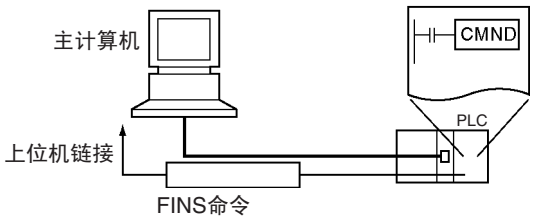
通过网络传输

CMND(490) 可用于把 FINS 命令传输到通过控制器链接网或以太网连接的 PLC （CPU 单元，内置板 （仅 CS 系列）， CPU 总线单元）或计算机。



通过 Host Link 传输

当 CPU 的内置串口、串行通信板 （仅 CS 系列）或串行通信单元是上位机链接模式，并且与主机一对一连接，当下次 PLC 有权传输数据时，可以执行 CMND(490) 把 FINS 命令从 PLC 传输到主计算机，它也能从与网络传输到与网络中别处的 PLC 相连的其它主计算机上。



CMND(490) 能够把命令从 CPU 单元串口、串口通信板或串行通信单元传送到与之相连接的主计算机上。该命令是一个封装在上位机链接数据头和数据尾之间的 FINS 消息（指定串口为 C+2 中的位 08 ～ 11 的 1（十六进制）或 2（十六进制））。任何 FINS 命令都能够发布。上位机链接头代码是十六进制 0F。

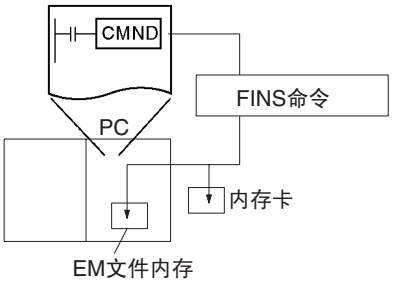
主计算机中必须创建程序以处理发送命令（封装在上位机链接头和尾之间的 FINS 命令）。

如果目标串口在本地 PLC 中，在 C+2 中设置网络地址为 00（本地网），在 C+3 中设置节点地址为 00（本地 PLC），设置单元地址 00（CPU 单元），E1（内置板，仅 CS 系列），或单元号 +10H（串口单元）。

执行 CMND(490) 发送一个 FINS 命令到 CPU 单元（在 V1 以前的 CS1 CPU 单元除外）

CPU单元执行CMND(490)能发送一个FINS命令到它本身（没有后缀-V□的CS系列的 CS1 CPU 单元除外）。例如，发送文件内存命令（命令代码 22 □□十六进制）能格式化文件内存，删除文件，复制文件和执行另外的操作。详情参考*第 12 节 CS/CJ 系列 CPU 单元操作手册的文件内存函数*。

当任何 FINS 命令被发送到本地 CPU 单元时，文件内存操作标志 (A34313) 变为 ON（即使 FINS 命令与文件内存无关）。总是用常闭输入条件的 A34313 使 CMND(490) 确保 CPU 单元在同时只有一个 FINS 命令被执行。



标志

名称	标记	操作
错误标志	ER	如果 C+2 中指定的串口号不在 00 ～ 04 的范围内时置 ON。 如果 C+4 中指定的通信端口号通信端口允许标志是 OFF 时置 ON。 如果文件内存标志 ON（A34312），FINS 命令发送到本地 CPU 单元时置 ON。 其它情况时 OFF。

下表表示辅助区中的相关位和标志。

名称	地址	操作
通信端口允许标志	A20200 ～ A20207	这些标志变 ON 说明可以在相应端口 (00 ～ 07) 执行包括 PMCR(260) 的网络指令。 当网络指令被执行时，相应端口的允许标志变 OFF，当指令完成时，它又变 ON。
通信端口错误标志	A21900 ～ A21907	这些标志如果变 ON，说明在网络指令执行过程中相应的端口发生了错误。 这些标志的状态将保持到下一条网络指令执行时为止。当下一个指令执行时，即使以前发生了错误，这些标志仍将变 OFF。

名称	地址	操作
通信端口完成代码	A203 ～ A210	这些字包含有相应端口 (00 ～ 07) 网络指令执行的完成代码。 当网络指令被执行时，相应的字包含有 0000，当指令结束时将写入完成代码。这些字在程序开始执行时被清除。
文件内存操作标志	A34313	当任何 FINS 命令被发送到本地 CPU 单元 （即使 FINS 命令与文件内存无关）或当执行下列用于文件内存的指令或操作时 ON。 FREAD(700) 或 FWRIT(701) 程序用内存中的控制位 覆盖简化备份操作

注意

如果 C+4 中指定端口号的通信端口允许标志为 OFF，指令将等同于 NOP (000) 且不被执行。这种情况下错误标志将变 ON。

当数据将要传输到本地网络之外时，用户必须在每个网络的 PLC（CPU 单元）中注册路由表。（路由表指明到连接目标节点的其它网络的路径）。

关于网络通信的完成代码详细情况参阅在*CS/CJ串行通信命令参考手册(W342)*中的 FINS 命令响应代码。

一个通信端口一次只能执行一个网络指令。为了确保在端口忙时 CMND(490) 不被执行，编程中将端口的通信允许标志 (A20200 ～ A20207) 作为一个常开条件。

通信端口的号码00～07被网络指令和PMCR(260)所共享，因此如果CMND(490)和 PMCR(260) 指令正在使用相同的端口号时，它们不能被同时执行。

当发送一个 FINS 命令到本地 CPU 单元时，总是用在 NO 输入条件的通信端口允许标志 (A20200～A20207) 和在常闭输入条件的文件内存操作标志 (A34313) 中的一个。

噪音和其它的因素能导致传输或响应破坏和丢失，所以我们建议把重试次数设成非 0 值，这将使得当在响应时间内没收到应答时再次执行 CMND(490)。

例

下面程序显示了一个发送 FINS 命令到另一个 CPU 单元的例子。

当CIO 000000和A20207（端口07的通信端口允许标志）为ON时，CMND(490)将 FINS 命令 0101（内存区读）传输到节点号 3。应答存储到 D00200 ～ D00211 中。

内存区读命令从 D00010 ～ D00019 中读取 10 个字。应答包含有 2 字节的命令代码 (0101)，2 字节完成代码，然后是 10 字的数据，总共 12 字或 24 个字节。

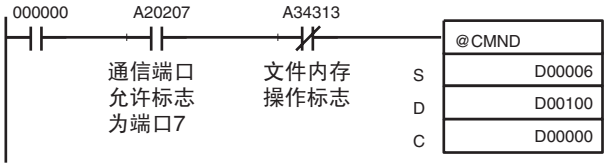
10 秒内未接收到应答，数据将最多可重复传输 3 次。

	15 87 0	
S:D00100	0 1 0 1	命令代码：0101 hex（内存区读）
S+1:D00101	8 2 0 0	数据区=82 hex，地址=000A00
S+2:D00102	0 A 0 0	
S+3:D00103	0 0 0 A	要读的字数=0A hex（10 十进制）

	15 87 0	
C:D00300	0 0 0 8	命令数据的字节：0008（8 十进制）
C+1:D00301	0 0 1 8	应答数据的字节：0018（24）
C+2:D00302	0 0 0 0	传输到本地网和设备本身
C+3:D00303	0 3 0 0	节点号3，单元地址00（CPU单元）
C+4:D00304	0 7 0 3	请求应答，端口号7，重试3次
C+5:D00305	0 0 6 4	应答监视时间：0064 十六进制（10秒）

下面程序显示了一个发送 FINS 命令到本地 CPU 单元的例子。

当 CIO 000000 和 A20207（端口 07 的通信端口允许标志）为 ON 并且 A34313 为 OFF 时， CMND(490) 将 FINS 命令 2215 （创建 / 删除目录）传输到本地 CPU 单元。应答存储到 D00100 ～ D00101 中。这里，FINS 命令将在 OMRON 目录下创建一个叫 CS/CJ 的目录。命令代码 （2 字节）和结束代码 （2 字节）将被返回并作为应答存储。



		15 8 7 0	
S:	D00006	2 2 1 5	命令代码：2215 十六进制（创建/删除目录）
S+1:	D00007	8 0 0 0	区号：8000 十六进制（内存卡）
S+2:	D00008	0 0 0 0	参数：0000 十六进制（创建目录）
S+3:	D00009	4 3 5 3	子目录名：CS1□□□□□.□□□（□=空间）
S+4:	D00010	3 1 2 0	
S+5:	D00011	2 0 2 0	
S+6:	D00012	2 0 2 0	
S+7:	D00013	2 E 2 0	
S+8:	D00014	2 0 2 0	
S+9:	D00015	0 0 0 6	目录名长度：0006（6个字母）
S+10:	D00016	5 C 4 F	绝对目录路径：/OMRON
S+11:	D00017	4 D 5 2	
S+12:	D00018	4 F 4 E	

		15 8 7 0	
S:	D00000	0 0 1 A	命令数据的字节：001A（26 十进制）
S+1:	D00001	0 0 0 4	应答数据的字节：0004(4)
S+2:	D00002	0 0 0 0	目标网络地址：00十六进制（本地网）
S+3:	D00003	0 0 0 0	目标单元地址：00十六进制目标节点号：00十六进制（本地节点的CPU单元）
S+4:	D00004	0 7 0 0	请求应答，端口号7，重试0次
S+5:	D00005	0 0 0 0	应答监视时间：0000 十六进制（6553.5秒）

3-26 文件存储指令

本节描述用于文件内存的指令 （EM 区或存储卡）。

注 文件内存可以通过执行 CMND(490) 发送一个 FINS 命令到本地 CPU 单元来处理。详情参阅 *CS/CJ 系列 PLC 操作手册*。

指令	助记符	功能代码	页
读数据文件	FREAD	700	896
写数据文件	FWRIT	701	906

3-26-1 使用存储卡时的注意事项

在使用存储卡时需确认下面几项。

格式化

出售前存储卡已作格式化处理，购买后无需格式化。但一旦用过后就要格式化，在 CPU 单元使用 CX-Programmer 或手持编程器时总要格式化。

如果存储卡直接在笔记本电脑或其它计算机上格式化，CPU 单元可能不识别这个存储卡。当这种情况发生时，即使重新在 CPU 单元格式化，你将不能再用这个存储卡。

在根目录下的文件数

放在存储卡中根目录下的文件数是有限制的 （仅对硬盘有限制）。尽管这个限制取决于存储卡的类型和格式，它介于 128 ～ 512 之间。当应用是在指定时刻写记录文件或其它文件时，这些文件写在子目录中而不是在根目录。

子目录能在计算机中或通过 CMND(490) 指令创建。参阅 *3-25-4 传送命令：CMND(490)* 指令 （用 CMND(490) 的一个特例）。

写操作次数

总的说来，对闪存写操作次数没有限制。然而，为确保起见，对存储卡限制为 100,000 写操作。例如，如果存储卡每 10 分钟写一次，两年可写 100,000 多次。

最小文件大小

如果有许多象 DM 区数据只有几个字的文件这样的小文件存储在存储卡，就不可能充分利用存储卡的容量。例如，如果用一个分配单元大小为 4,096 字节的存储卡，每一个文件都占据 4,096 字节，不管这个文件有多小。如果你存 DM 区数据的 10 个字到存储卡，即使文件的实际大小只有 68 字节它也要占据 4,096 字节。用这样小的文件将大大降低存储卡的利用率。减小分配单元大小可以提高利用率，但又会减小存取速度。

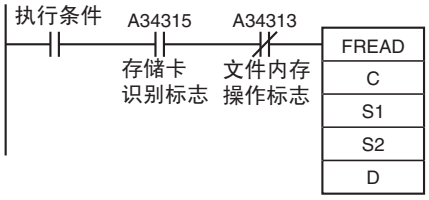
存储卡的分配单元大小可在 DOS 路径下用 CHKDSK 检查。这里不介绍具体步骤。更多分配单元大小的信息参阅通用计算机文献。

存储卡访问注意

当 PLC 访问存储卡时， CPU 单元上的 BUSY 指示器是亮的。注意下面事项。

1,2,3...

1. 当 BUSY 指示器亮着时，千万不要断开 CPU 单元的电源。如果发生这种情况，存储卡可能出现异常。
2. 当 BUSY 指示器亮着时，绝对不要从 CPU 单元拆卸存储卡。可按存储卡电源 OFF 按扭，等 BUSY 指示器熄灭了，方可拆除存储卡。如果不这样操作，存储卡可能出现异常。
3. 插入存储卡时标签面朝着右侧。不要试图以任何其它方向插入它。否则存储卡或 CPU 单元可能会损坏。
4. 存储卡插入后， CPU 单元需要几秒钟才能识别它。当电源变 ON 或插入存储卡立即访问存储卡时，对存储卡识别标志 (A34515) 的常闭下图用了常开条件条件编程作为一个输入条件，如下所示。

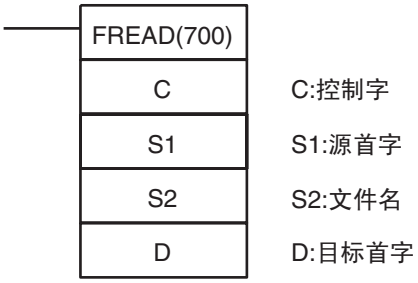


3-26-2 读数据文件：FREAD(700)

用途

从文件内存中指定的数据文件读取指定的数据或指定的数据量，并存放到 CPU 单元中指定的数据区。

梯形图符号



变化

变化	ON 条件时每次循环执行	FREAD(700)
	上升沿微分时执行一次	@FREAD(700)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

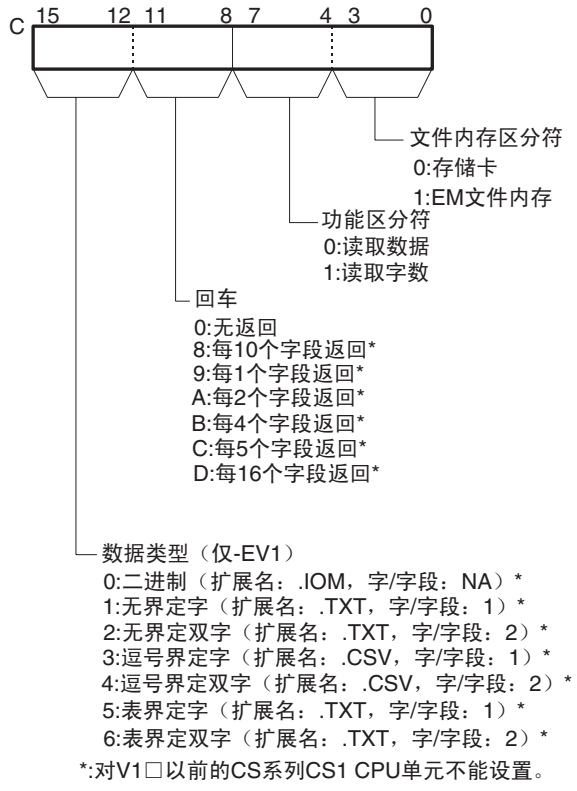
适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

C: 控制字

下图所示，控制字的第一个数字指明源文件是在存储卡还是在EM文件内存中，第二个数字指明是读取实际数据还是读取数据的字数，第三个数字指明回车是否存在，第四个数字指明数据类型。



注 1. 每个字段含有字数据类型的 I/O 内存的一个字和双字数据类型的 I/O 内存的二个字。

2. 当读取带回车的数据， C 中的位 00 ～ 11 必须设置在 8 ～ D 十六进制。

3. 用双字时，数据的第一个字储存在内存地址的最高位，例如，12345678 中的 1234 储存在 D00001， 5678 储存在 D00000。

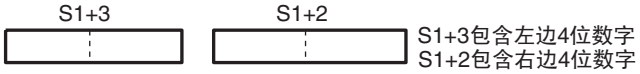
S1 和 S1+1：读取项的数

S1 和 S1+1 中的 8 位十六进制数字指定了从文件内存中读取多少字或字段。如果指定的字或字段数超过了数据文件的字数，文件中的数据仍可正常传递，而不发生错误。

数据类型	C 的 12 ～ 15 位	S1 和 S1+1 的内容
二进制	0 十六进制 （二进制）	从文件内存读取的字数。 00000000 ～ 3FFFFFFF Hex
字	1 十六进制 （无界定） 3 十六进制 （逗号界定） 或 5 十六进制 （表界定）	从文件内存读取的字段数，即，从文件内存读取的字数。 00000000 ～ 1FFFFFFF Hex
双字	2 十六进制 （无界定） 4 十六进制 （逗号界定） 或 6 十六进制 （表界定）	从文件内存读取的字段数，即，从文件内存读取的一半的字数。 00000000 ～ 0FFFFFFF Hex

S1+2 和 S1+3：源首字

S1+2 和 S1+3 中的 8 位十六进制数字指定了从文件起始部分读取的起始字。



数据类型	C 的 12 ～ 15 位	S1+2 和 S1+3 的内容
二进制	0 十六进制 （二进制）	从文件内存起始部分开始读的字。 00000000 ～ 3FFFFFFF Hex
字	1 十六进制 （无界定） 3 十六进制 （逗号界定） 或 5 十六进制 （表界定）	从文件内存起始部分开始读的字段，即，从开始起的字数。 00000000 ～ 1FFFFFFF Hex
双字	2 十六进制 （无界定） 4 十六进制 （逗号界定） 或 6 十六进制 （表界定）	从文件内存起始部分开始读的字段，即，从开始起的的一半的字数。 00000000 ～ 0FFFFFFF Hex

注 1. S1+2 和 S1+3 仅用在无回车的文本和 CVS 数据（即，C 的位 08 ～ 11 设为 0 十六进制）或二进制数据。当回车一起读取数据时，总是设置 S1+2 和 S1+3 为 0 十六进制 （即， C 的为 08 ～ 11 设为 8 和 D 十六进制）。

2. S1 ～ S1+3 必须在同一数据区。

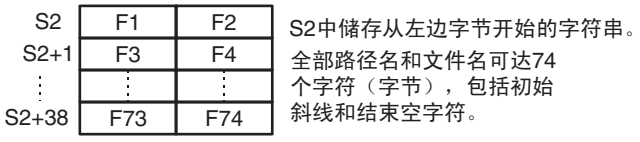
3. S1 ～ S1+3 仅用在读数据。

4. 如果指定起始字超出数据文件的字数，文件读取错误标志将变为 ON，文件将不能被读。

S2：文件名

S2 是包含以 ASCII 表示的绝对路径和文件名的字的起始地址，使用 ASCII 的 a ～ z， A ～ Z，和 0 ～ 9。

包含数据文件的完整目录路径名可达 65 个字符，包括起始斜线 (ASCII5C)。文件名可达 8 个字符，但不允许为空字符 (ASCII 00)，因为空字符用来标记字符串的结尾。不包括文件扩展名将自动添加 .IOM 扩展名。



注 1. 保证路径名和文件名的字符串不超出数据区的末尾。

2. 如果指定的文件或目录不存在，文件丢失标志 (A34311) 变 ON，且文件数据不被读取。

在S2中从左边开始以ASCII形式写入路径名和文件名。例如下例: \ABC\XYZ.IOM（.IOM 扩展名自动添加）。

S2	"\"	"A"	S2	5C	41
S2+1	"B"	"C"	→ S2+1	42	43
S2+2	"\"	"X"	S2+2	5C	58
S2+3	"Y"	"Z"	S2+3	59	5A
S2+4	NUL		S2+4	00	

D：目标首字

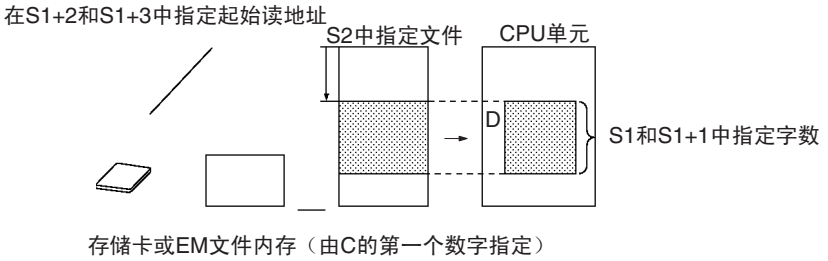
当读取数据时， D 指定从文件内存读取的数据所储存的起始地址。

当读取字数时，所读取的数据的字数以 8 位十六进制的形式写入 D 和 D+1 中（00000000 ～ FFFFFFFF）。 D 包含最右边 4 位， D+1 包含最左边 4 位。

描述

读数据 （C 的第三个数为 0）

FREAD(700) 从 S2 中指定的文件 （文件扩展名为 .IOM）中读取 S1 和 S1+1 中指定的字数，S1+2 和 S1+3 中指定的起始地址。然后将数据写入以 D 指定的开始的 RAM 中。

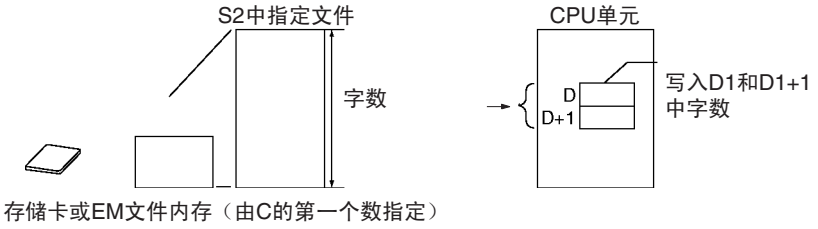


注 数据根据绝对内部 I/O 内存地址存储，因此如果数据超出 D 中所指定的数据区的容量，输出数据会覆盖下面的数据区。详细情况见注意事项。

当执行 FREAD(700) 时，S1 和 S1+1 中指定的字数（或字段数）写入 A346 和 A347 （传输的数据数），每个字或字段传输时，这个值减 1。字的内容能被检查，以校验是否已传输所期望的字数。

读数据字数 （C 的第三个数为 1）

FREAD(700) 寻找 S2 中指定的文件 （文件扩展名为 .IOM）的字数，并把 8 个十六进制值写入 D 和 D+1 中。



操作数规定

区域	C	S1	S2	D
CIO 区	CIO 0000 ～ CIO6143	CIO 0000 ～ CIO 6140	CIO 0000 ～ CIO 6143	
工作区	W000～W511	W000～W508	W000 ～ W511	
保持位区	H000 ～ H511	H000 ～ 508	H000 ～ W511	
辅助位区	A000 ～ A959	A000 ～ A444 A448 ～ A956	A000 ～ A447 A448 ～ A959	A448 ～ A959
定时器区	T0000 ～ T4095	T0000 ～ T4092	T0000 ～ T4095	
计数器区	C0000 ～ C4095	C0000 ～ C4092	C0000 ～ C4095	
DM 区	D00000 ～ D32767	D00000 ～ D32764	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32767	E00000 ～ E32764	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	En_00000 ～ En_32764 (n = 0 ～ C)	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 地址	–	@D00000 ～ @D32767 @E00000 ～ @E32767 @En_00000 ～ @En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	–	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	仅指定值	–		
数据寄存器	–			
索引寄存器	–			
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15			

标志

名称	标记	操作
错误标志	ER	如果 C 中指定的文件内存不存在时置 ON。 如果 C 中设置不在指定范围内时置 ON。 如果 S2 中指定的文件名不满足所要求的条件时置 ON。 如果文件内存指令标志为 ON 时置 ON。 如果没有给 C 指定一个常数时置 ON （仅对 V1 □以前的 CS 系列 CS1 CPU 单元）。 如果给 S1 指定的数据超出范围 （除了 V1 □以前的 CS 系列 CS1 CPU 单元外的所有 CPU 单元）时置 ON。 如果指定给 D 一个非法区域置 ON。 对 CS1D CPU 单元：如果主 CPU 和备用 CPU 不能同步时置 ON。 其它情况时 OFF。

下表表示辅助区中的相关位和标志。

名称	地址	操作
内存卡类型	A34300 ～ A34302	它包含了一个二进制数指明了所安装的内存卡类型。 （0：无， 4：闪存 ROM）
内存卡格式错误标志	A34307	当内存卡未格式化或格式化错误时置 ON。
文件读错误标志	A34310	当文件数据破坏而不能读取或它包含错误类型时置 ON。
文件丢失错误标志	A34311	当读取不存在的文件的数据时置 ON。
文件内存操作标志	A34313	下列任何情况都是 ON： CPU 使用 CMND(490) 发送一个 FINS 命令给它本身。 执行 FREAD(700) 或 FWRIT(701)。 用在内存中使用一个控制位覆盖这个程序时。 执行简单的备份操作。
访问文件标志	A34314	当数据备被访问时置 ON。 为了防止多个文件内存指令同时执行，将此标志作为执行条件。
内存卡检测标志	A34315	当检测内存卡时 ON。
EM 文件格式起始区域	A344	包含已格式化用于 EM 文件内存的 EM 起始区号。如果 EM 区没有格式化， A344 中为 FFFF。 要想把 EM 用于文件内存，在 PLC 设置的 EM 文件内存设置必须为 1，并且必须设置 EM 文件内存起始区号 (0 ～ C)。从起始区号到最后区号的所有 EM 都将格式化用于文件内存。
EM 文件内存格式错误标志	A34306	当 EM 文件内存的起始区有格式化错误时置 ON。
传输的数据数	A346 ～ A347	这些字的内容指明书记文件传输的状态。 当执行 FREAD(700) 或 FWRIT(701) 指令时，要传输的字数或字段数写入这些字，每个字被传输时该值减 1。 A346 包含 32 位二进制值的最右 16 位， A347 包含最左 16 位。

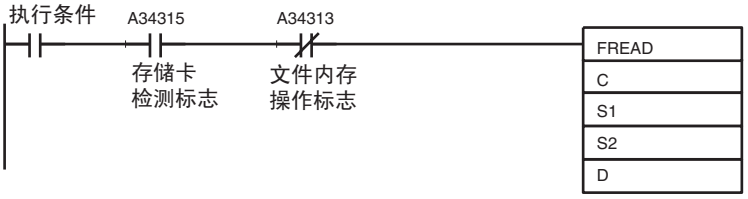
注意 在指令正常执行期间， FREAD(700) 只用于启动读文件内存。本手册最后给出的指令执行时间是指启动读所需要的时间，而不是完成读的时间。实际的读（传输）是由外围服务中的文件访问完成的。因此，一旦 FREAD(700) 被执行，读取是连续执行的，即使下一次循环时执行条件变 OFF。当传输完成时，文件内存指令标志 (A34313) 变 OFF。该标志可用于文件内存指令的专用控制。

FREAD(700) 完成数据传输所需的时间依赖于传输数据的数量、分配给文件访问处理的服务时间和其它的条件。大致上，一个根目录中的文件，在扫描周期为 10ms，服务时间设置为缺省的情况下，其传输时间为每 1024 字 0.92s，每 9999 字需 4.64s。

当 FREAD(700) 执行时文件内存操作标志 (A34313) 将变 ON。如果指令开始执行时 A34313 已经为 ON，将发生错误，指令不被执行。

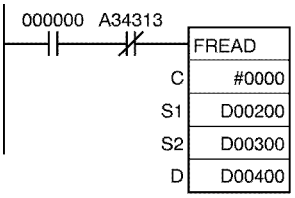
如果指定的文件不是一个数据文件或文件数据被破坏，文件读错误标志 (A34310)将变ON，指令不被执行。对于文本或CSV文件，字符代码必须是十六进制数据，并且界定值必须是对字数据每 4 个数字，对双字每 8 个数字。数据将停止在检测到非法字符处。

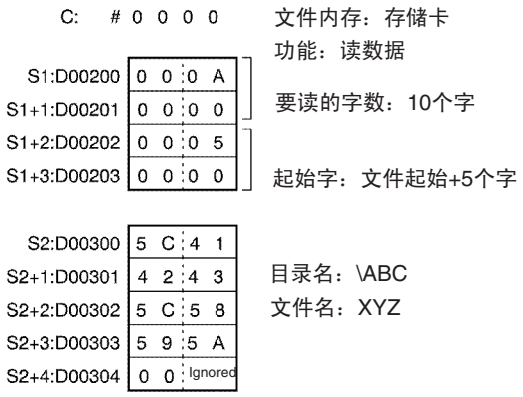
存储卡插入后， CPU 单元需要几秒钟才能识别它。当电源变 ON 或插入存储卡，存储卡立即被访问，对存储卡检测标志 (A34315) 的 NO 输入条件如下所示以，确保存储卡被检测到。

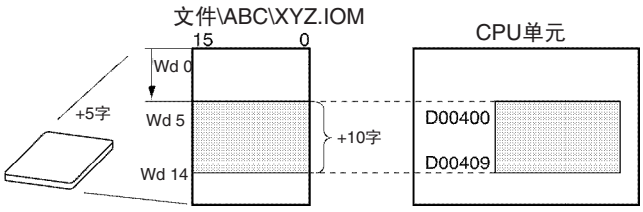


例

下例中当 CIO 00000 变 ON，FREAD(700) 从文件 \ABC\XYZ.IOM 中读取以文件起始+5个字开始的10个字的数据，并将这10个字输出到D00400～D00409中。





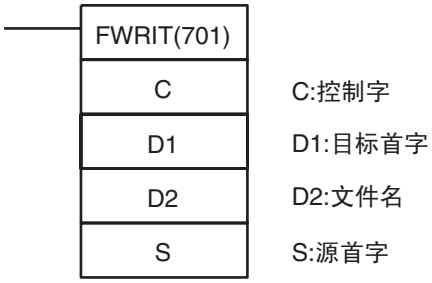


3-26-3 写数据文件：FWRIT(701)

用途

把 CPU 单元数据区中指定的数据添加或重新写入文件内存中指定的数据文件中。如果指定文件不存在，将产生该文件名的新文件。数据可以是二进制、文本或 CSV 格式数据。

梯形图符号



变化

变化	ON 条件时每次循环执行	FWRIT(701)
	上升沿微分时执行一次	@FWRIT(701)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

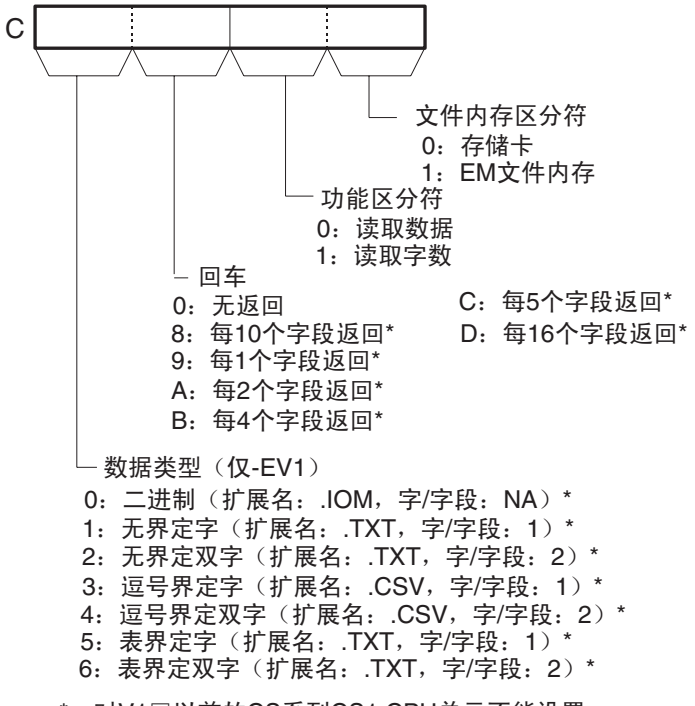
适用 程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

C: 控制字

下图所示，控制字的第三个数字指明是添加或重新写入数据文件的数据，第四个数字指明目标文件是在存储卡还是在 EM 文件内存中。



*：对V1□以前的CS系列CS1 CPU单元不能设置。

注 1. 每个字段含有字数据类型的 I/O 内存的一个字和双字数据类型的 I/O 内存的二个字。

2. 用双字时，数据的第一个字储存在内存地址的最高位，例如，12345678 中的 1234 储存在 D00001， 5678 储存在 D00000。
3. 如果指定界定，对字数据类型在每个字后添加界定值，对双字数据类型在每两个字后添加界定值。（对逗号界定添加逗号代码，对表界定添加表代码）。
4. 如果指定非界定字或双字，所有字段的数据无需界定地连续写入。
5. 如果指定回车，一个回车在每一组指定字数后添加。如果不指定回车，数据无需回车地连续写入。

D1 和 D1+1：写入项的数

D1 和 D1+1 中的 8 位十六进制数字指定了有多少字或字段可写入文件内存。

数据类型	C 的 12 ～ 15 位	D1 和 D1+1 的内容
二进制	0 十六进制 （二进制）	从文件内存写的字数。 00000000 ～ 3FFFFFFF Hex
字	1 十六进制 （无界定） 3 十六进制 （逗号界定） 或 5 十六进制 （表界定）	从文件内存写的字段数，即，从文件内存写的字数。 00000000 ～ 1FFFFFFF Hex
双字	2 十六进制 （无界定） 4 十六进制 （逗号界定） 或 6 十六进制 （表界定）	从文件内存写的字段数，即，从文件内存写的一半的字数。 00000000 ～ 0FFFFFFF Hex

D1+2 和 D1+3：源首字

D1+2 和 D1+3 中的 8 位十六进制数字指定了从文件起始部分写的起始字。

数据类型	C 的 12 ～ 15 位	D1+2 和 D1+3 的内容
二进制	0 十六进制 （二进制）	从文件内存起始部分开始写的字。 00000000 ～ 3FFFFFFF Hex
字	1 十六进制 （无界定） 3 十六进制 （逗号界定） 或 5 十六进制 （表界定）	从文件内存起始部分开始写的字段，即，从开始起的字数。 00000000 ～ 1FFFFFFF Hex
双字	2 十六进制 （无界定） 4 十六进制 （逗号界定） 或 6 十六进制 （表界定）	从文件内存起始部分开始写的字段，即，从开始起的的一半的字数。 00000000 ～ 0FFFFFFF Hex

注 1. D1+2 和 D1+3 仅用在重写数据时， 1）用于无字回车的文本和 CSV 数据（即，C 的位 08 ～ 11 设为 0 十六进制）或 2）用于二进制数据。当与回车一起写数据时，总是设置 D1+2 和 D1+3 为 00000000 十六进制 （即， C 的为 08 ～ 11 设为 8 ～ D 十六进制）。

2. D1 ～ D1+3 必须在同一数据区。
3. 如果指定起始字超出数据文件的字数，文件写错误标志 (A34308) 将变为 ON，数据将不能写入。

D2: 文件名
D2 是包含以 ASCII 表示的绝对路径和文件名的字的起始地址，使用 ASCII 的 a ～ z， A ～ Z，和 0 ～ 9。

包含数据文件的完整目录路径名可达 65 个字符，包括起始斜线 (ASCII5C)。文件名可达 8 个字符，但不允许为空字符 (ASCII00)，因为空字符用来标记字符串的结尾。不包括文件扩展名的将自动添加 .IOM 扩展名。

D2	F1	F2
D2+1	F3	F4
⋮	⋮	⋮
D2+38	F73	F74

D2中储存从左边字节开始的字符串。全部路径名和文件名可达74个字符（字节），包括初始斜线和结束空字符。

注 1. 保证包含路径名和文件名的字符串不超出数据区的末尾。
2. 如果指定的目录不存在，文件丢失标志(A34311)变ON，且文件数据不被写。

在 D2 中从最左位开始以 ASCII 形式写入路径名和文件名。例如下例：\ABC\XYZ.IOM （.IOM 扩展名自动添加）。

D2	"\"	"A"	D2	5C	41
D2+1	"B"	"C"	→ D2+1	42	43
D2+2	"\"	"X"	D2+2	5C	58
D2+3	"Y"	"Z"	D2+3	59	5A
D2+4	NUL		→ D2+4	00	

S: 源首字
S 指定要写入数据的文件内存的起始地址。数据由绝对 PLC 内存地址读取，所以，如果被读字数超过 S 中指定数据区的末尾， FWRIT(701) 从下一数据区连续读取源数据。

描述

在指令正常执行期间， FWRIT(701) 只用于开始写文件内存。本手册最后给出的指令执行时间是指开始写所需要的时间，而不是完成写的时间。实际的写（传输）是由外围服务中的文件访问处理完成的。因此，一旦 FWRIT(701) 被执行，写是连续执行的，即使下一次循环时执行条件变 OFF。当传输完成时，文件内存指令标志 (A34313) 变 OFF。该标志可用于文件内存指令的专用控制。

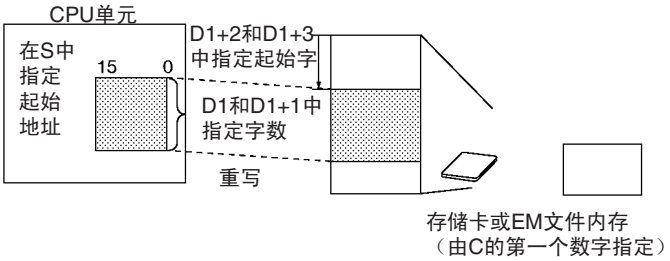
FWRIT(701) 完成数据传输所需的时间依赖于传输数据的数量、分配给文件访问处理的服务时间和其它的条件。大致上，一个根目录中的文件，在扫描周期为 10ms，服务时间设置为缺省的情况下，其传输时间为每 1024 字 1.97s（新文件）或 1.33s （已存在文件），每 9999 字需 6.64s （新文件）或 6.12s （已存在文件）。

源数据从 RAM 的绝对内部 I/O 内存地址中读取，如果数据占用 2 个或更多的数据区时，整个数据块都将被读取。例如，如果第一个目标地址在工作区，而数据的数量超出该区的容量， FWRIT(701) 将在下一个区的起始读取数据 （在这种情况下为定时区）。RAM 中数据区的位置参照 *CS/CJ 系列可编程控制器操作手册 (W339) 附录 D* 内存图。

当执行 FWRIT(701) 时，D1 和 D1+1 中指定的字数（或字段数）写入 A346 和 A347 （传输的数据数），每个字或字段传输时，这个值减 1。字的内容能被检查，以校验是否已传输所期望的字数。

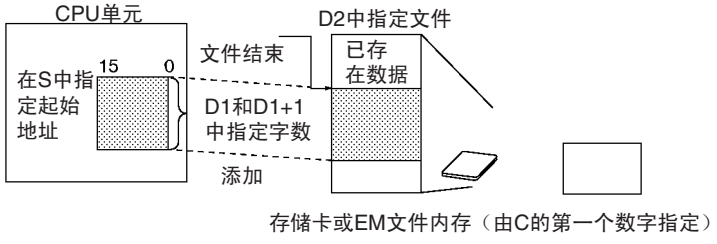
在已存文件中重写数据 （C 的第三个数为 1）

FWRIT(701) 用在 S 中指定起始字的数据区数据以指定的文件类型重写文件内存数据。重写字数或字段数在 D1 和 D1+1 中指定，D2 中为指定文件（文件扩展名为 .IOM， .TXT， .CSV），起始地址在 D1+2 和 D1+3 中指定。



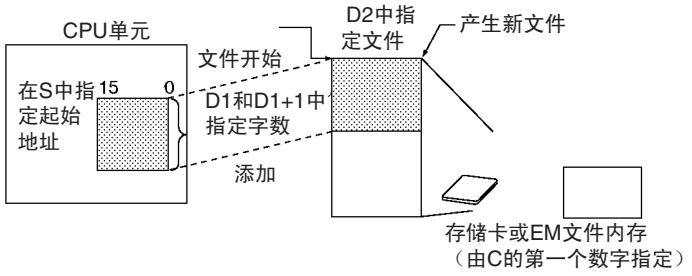
在已存文件中添加数据字数 （C 的第三个数为 0）

FWRIT(701) 用在 S 中指定起始字的数据区数据以指定的文件类型添加文件内存数据。添加字数或字段数在 D1 和 D1+1 中指定（文件扩展名为 .IOM，.TXT，.CSV），起始地址在 D1+2 和 D1+3 中指定。



用源数据创建一个新文件

如果 D2 中指定的文件不存在， FWRIT(701) 产生一个具有该文件名和件扩展（.IOM，.TXT，.CSV）的新文件，并将指定的源数据以指定的文件类型从文件起始写入。这种情况下，是否指定添加重写数据是无关系的。



操作数规定

区域	C	D1	D2	S
CIO 区	CIO 0000 ～ CIO 6143	CIO 0000 ～ CIO 6140	CIO 0000 ～ CIO 6143	
工作区	W000～W511	W000～W508	W000 ～ W511	
保持位区	H000 ～ H511	H000 ～ 508	H000 ～ H511	
辅助位区	A000 ～ A959	A000 ～ A444 A448 ～ A956	A000 ～ A447 A448 ～ A959	
定时器区	T0000 ～ T4095	T0000 ～ T4092	T0000 ～ T4095	
计数器区	C0000 ～ C4095	C0000 ～ C4092	C0000 ～ C4095	
DM 区	D00000 ～ D32767	D00000 ～ D32764	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32767	E00000 ～ E32764	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	En_00000 ～ En_32764 (n = 0 ～ C)	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 地址	–	@D00000 ～ @D32767 @E00000 ～ @E32767 @En_00000 ～ @En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	–	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	仅指定值	–		
数据寄存器	–			
索引寄存器	–			
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15			

标志

名称	标记	操作
错误标志	ER	如果 C 中指定的文件内存类型不存在时置 ON。 如果 C 中设置不在指定范围内时置 ON。 如果 D2 中指定的文件名不满足所要求的条件时置 ON。 如果文件内存指令标志为 ON 时置 ON。 如果没有给 C 指定一个常数时置 ON （仅对 V1 □以前的 CS 系列 CS1 CPU 单元）。 如果给 D1 指定的数据超出范围 （除了 V1 □以前的 CS 系列 CS1 CPU 单元外的所有 CPU 单元）时置 ON。 如果指定给 D 一个非法区域置 ON。 对 CS1D CPU 单元：如果主 CPU 和备用 CPU 不能同步时置 ON。 其它情况时 OFF。

下表表示辅助区中的相关标志。

名称	地址	操作
内存卡类型	A34300 ～ A34302	它包含了一个二进制数指明了所安装的内存卡类型。 （0：无， 4：闪存 ROM）
内存卡格式错误标志	A34307	当内存卡未格式化或格式化错误时置 ON。
文件写错误标志	A34308	当文件数据破坏而不能写或它包含错误类型时置 ON。
文件不能写标志	A34309	当文件是写保护或内存空间不够而不能写时置 ON。
无文件标志	A34311	当写入文件的指定路径不存在时置 ON。
文件内存操作标志	A34313	下列任何情况都是 ON： CPU 使用 CMND(490) 发送一个 FINS 命令给它本身。 执行 FREAD(700) 或 FWRIT(701)。 使用内存中一个控制位覆盖程序时。 执行简单的备份操作。
访问文件标志	A34314	当数据备被访问时置 ON。 为了防止多个文件内存指令同时执行，将此标志作为执行条件。
内存卡检测标志	A34315	当检测内存卡时 ON。
EM 文件格式起始区域	A344	包含已格式化用于 EM 文件内存的 EM 起始区号。如果 EM 区没有格式化， A344 中为 FFFF。 要想把 EM 用于文件内存，在 PLC 设置的 EM 文件内存设置必须为 1，并且必须设置 EM 文件内存起始区号 (0 ～ C)。从起始区号到最后区号的所有 EM 都将格式化用于文件内存。
EM 文件内存格式错误标志	A34306	当 EM 文件内存的起始区有格式化错误时置 ON。
传输的数据数	A346 ～ A347	这些字的内容指明书记文件传输的状态。 当执行 FREAD(700) 或 FWRIT(701) 指令时，要传输的字数或字段数写入这些字，每个字被传输时该值减 1。 A346 包含 32 位二进制值的最右 16 位， A347 包含最左 16 位。

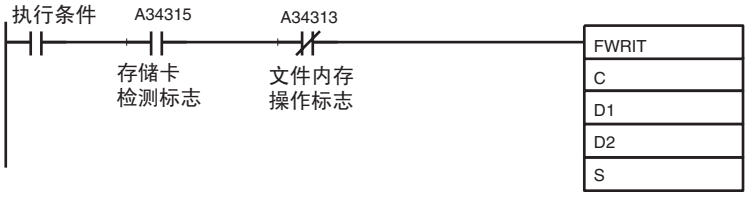
注意

当 FWRIT(701) 执行时文件内存操作标志 (A34313) 变 ON。如果指令开始执行时 A34313 已经为 ON，将发生错误，指令不被执行。

如果数据由于文件写保护或没有足够的内存空间而不能写时，文件不能写标志 (A34309) 变 ON，并且指令将不执行。

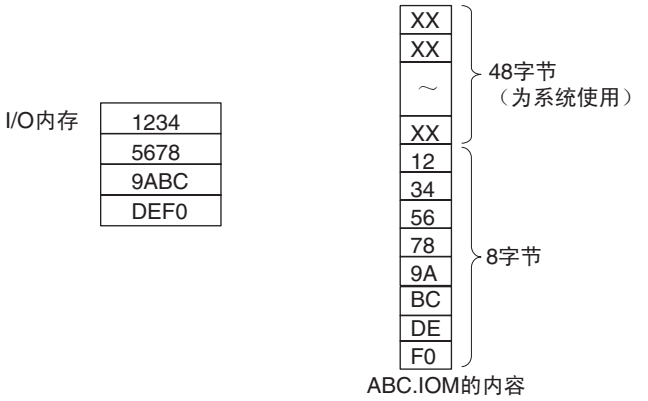
如果指定的文件不是一个数据文件或文件数据被破坏，文件写错误标志 (A34308) 将变 ON，指令不被执行。

存储卡插入后， CPU 单元需要几秒钟才能识别它。当电源变 ON 或插入存储卡，存储卡立即被访问，对存储卡检测标志 (A34315) 的常开输入条件如下所示以确保存储卡被检测到。

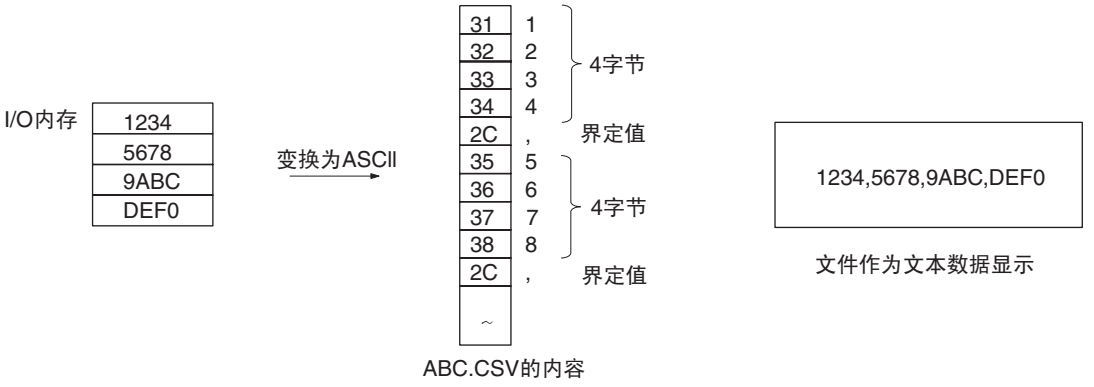


注 数据文件结构如下所示。

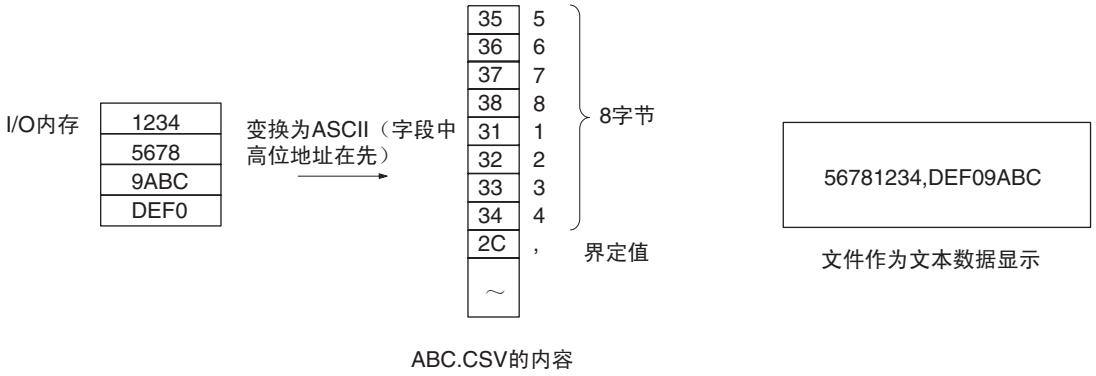
对二进制格式 (.IOM)，当 1234 十六进制，5678 十六进制，9ABC 十六进制和 DEF0 十六进制存储到文件 ABC.IOM 中时，数据结构如下 （尽管用户不会涉及到数据结构）：



对字 CSV 格式 (.CSV)：当 1234 十六进制， 5678 十六进制， 9ABC 十六进制和 DEF0 十六进制存储到文件 ABC.CSV 中时，数据结构如下 （基本结构与文本数据相同 (.TXT)）：

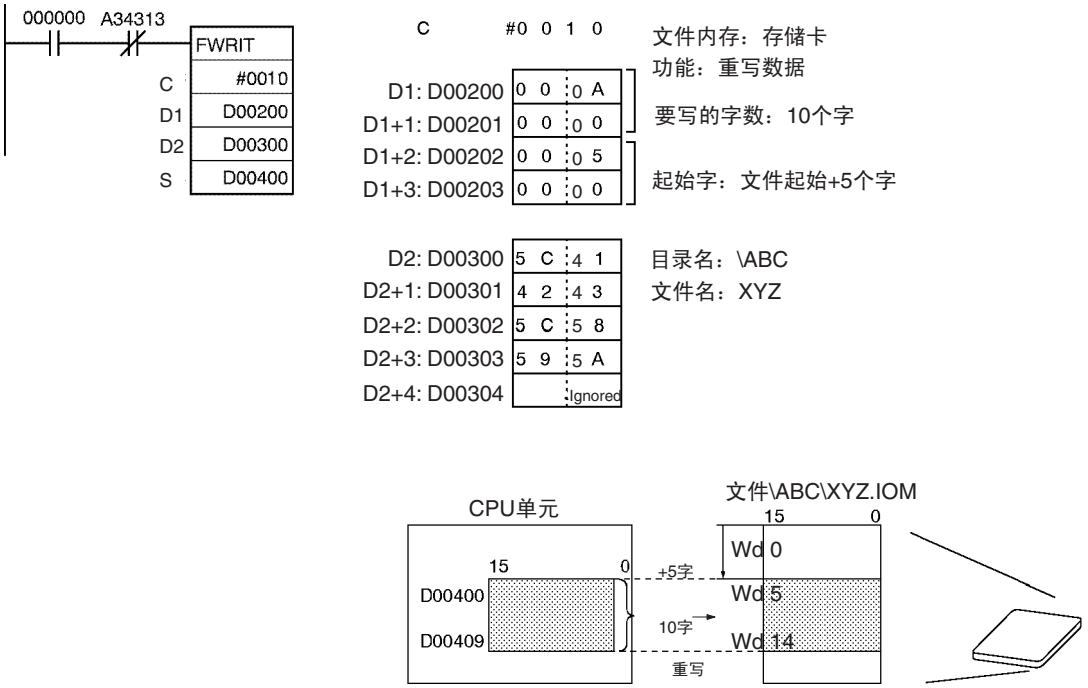


对长字 CSV 格式 (.CSV)：当 1234 十六进制， 5678 十六进制， 9ABC 十六进制和 DEF0 十六进制存储到文件 ABC.CSV 中时，数据结构如下 （基本结构与文本数据相同 (.TXT)）：



例

下例中当 CIO 000000 变 ON， FWRIT(701) 从 D00400 ～ D00409 中读 10 个字的数据，并将该数据重写到文件 \ABC\XYZ.IOM 中以文件起始 +5 个字开始的 10 个字中。

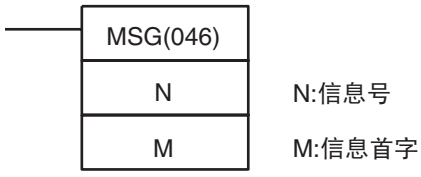


3-27 显示指令：显示信息：MSG(046)

用途

读取指定的十六个扩展ASCII字，并将信息显示在如编程器之类的外围设备上。

梯形图符号



变化

变化	ON 条件时每次循环执行	MSG(046)
	上升沿微分时执行一次	@MSG(046)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

N: 信息号
信息号必须是十六进制 0000 ～ 0007 （或十进制 0 ～ 7）。

M: 信息首字
显示信息时，M 指定包含 ASCII 信息的第一个字的地址。消除信息时，M 可以是任何的十六进制常数 （0000 ～ FFFF）。

操作数规定

区域	N	M
CIO 区	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W511	
保持位区	H000 ～ H511	
辅助位区	A000 ～ A959	
定时器区	T0000 ～ T4095	
计数器区	C0000 ～ C4095	
DM 区	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	#0000 ～ #0007 （二进制） 或 &0 ～ &7	#0000 ～ #FFFF （二进制）
数据寄存器	DR0 ～ DR15	---
索引寄存器	---	
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

描述

当执行条件为 ON 时，MSG(046) 对 N 指定的信息号，注册 M ～ M+15 中的 16 字的 ASCII 数据 （包括空字符在内最多可达 32 字符）。一旦一个信息已经注册，当连接编程器时，该信息将显示在产生的错误信息之后。

信息注册后，通过重写信息存储区的信息来改变信息显示。

要想清除已注册的信息，执行 MSG(046) 指令，把 S 设置成将要清除信息的信息号，将 N 设置成常数 (0000 ～ FFFF)。

即使程序停止执行，在程序执行时注册的信息，仍然一直保持，但当程序再次执行时，所有的信息都将被清除。

注　扩展 ASCII 表参考 *CS1 系列可编程控制器操作手册 (W341) 的附录 A*。

标志

名称	标记	操作
错误标志	ER	如果 S 中的内容不是十六进制 0000 ～ 0007 时置 ON。 其它情况时 OFF。

注意

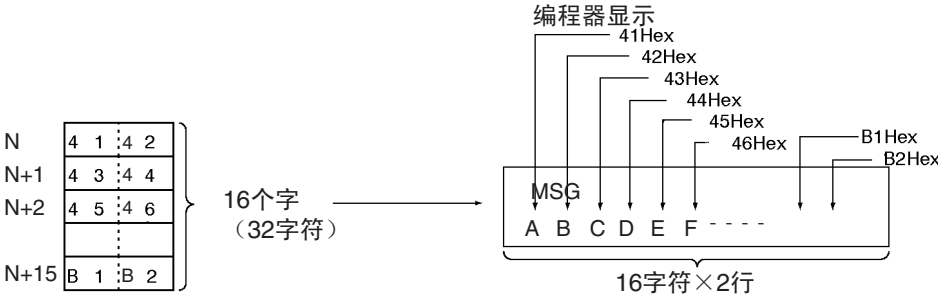
每次 MSG(046) 执行时都将刷新注册信息。

空字符 (00) 之后的所有信息字符被转化成编程器显示的空格。

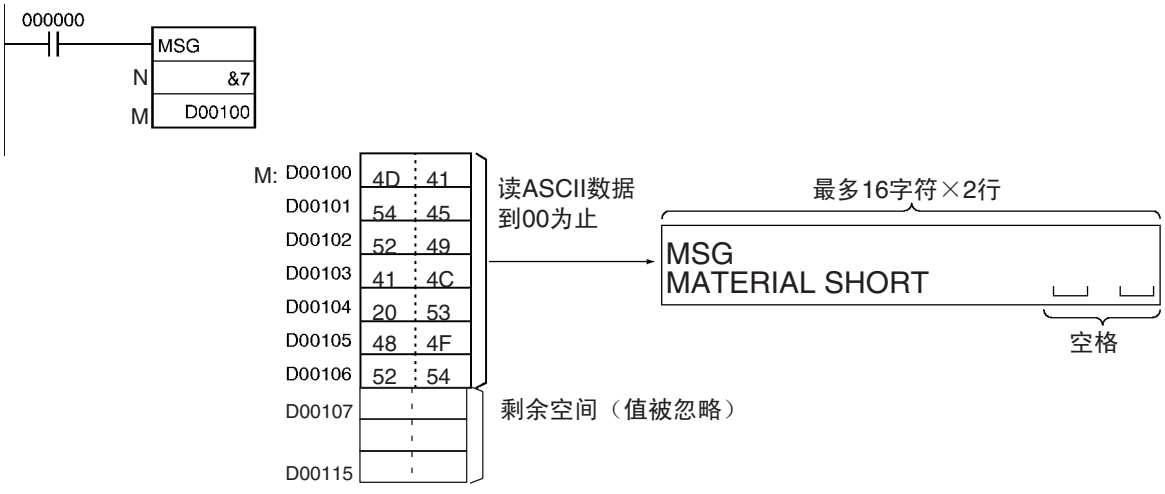
先显示最左字节中的字符，再显示右边字节中的字符。

例

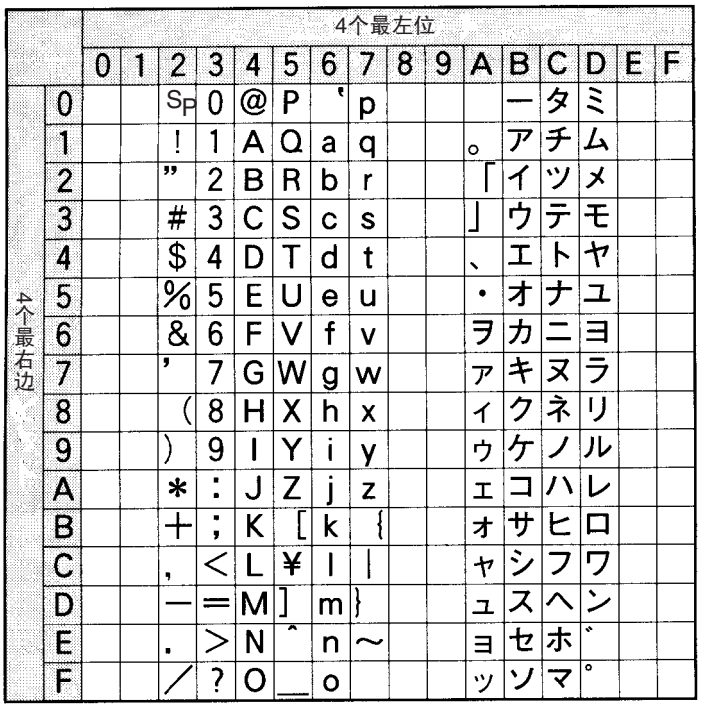
下例显示如何将 16 个字的十六进制数据转换成信息显示到编程器上。



下例中当 CIO 000000 变 ON，D00100 ～ D00115 中的 16 个字读作 32 个 ASCII 字符，并作为信息号 7 显示在外围设备上。



ASCII



4个最右边 \ 4个最左位	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0			SP	0	@	P	‘	p				ー	タ	ミ		
1			!	1	A	Q	a	q			。	ア	チ	ム		
2			”	2	B	R	b	r			「	イ	ツ	メ		
3			#	3	C	S	c	s			」	ウ	テ	モ		
4			$	4	D	T	d	t			、	エ	ト	ヤ		
5			%	5	E	U	e	u			・	オ	ナ	ユ		
6			&	6	F	V	f	v			ヲ	カ	ニ	ヨ		
7			’	7	G	W	g	w			ァ	キ	ヌ	ラ		
8			(	8	H	X	h	x			ィ	ク	ネ	リ		
9			)	9	I	Y	i	y			ゥ	ケ	ノ	ル		
A			*	:	J	Z	j	z			ェ	コ	ハ	レ		
B			+	;	K	[	k	{			ォ	サ	ヒ	ロ		
C			,	<	L	¥	l	|			ャ	シ	フ	ワ		
D			−	=	M	]	m	}			ュ	ス	ヘ	ン		
E			.	>	N	^	n	~			ョ	セ	ホ	゛		
F			/	?	O	_	o				ッ	ソ	マ	°		

3-28 时钟指令

这节介绍使用系统时钟的指令。

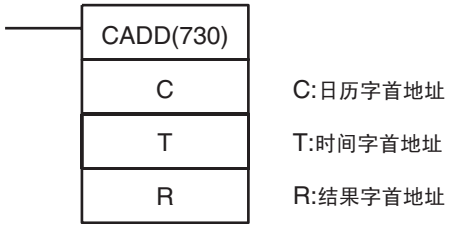
指令	助记符	功能代码	页
日历加	CADD	730	911
日历减	CSUB	731	915
小时到秒	SEC	065	923
秒到小时	HMS	066	925
时钟调节	DATE	735	928

3-28-1 日历加：CADD(730)

用途

在指定字的日历数据上增加时间。

梯形图符号



变化

变化	ON 条件时每次循环执行	CADD(730)
	上升沿微分时执行一次	@CADD(730)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

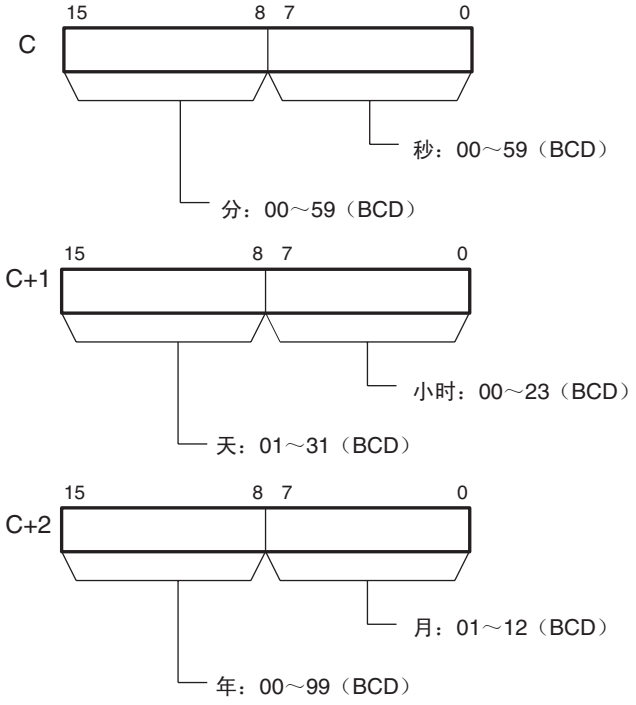
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

C ～ C+2：日历数据

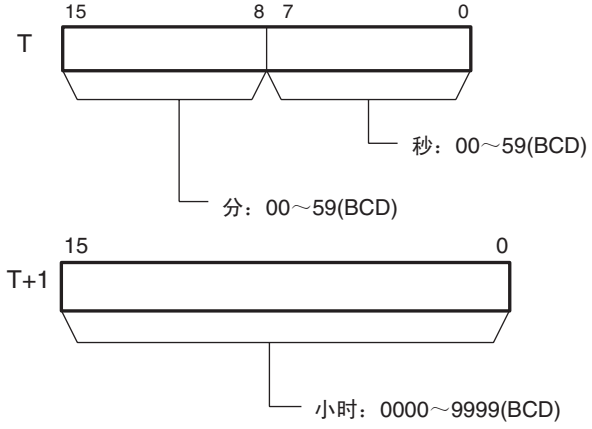
如下图所示，在 C ～ C+2 中设置日历数据。

C ～ C+2 必须在相同的数据区。

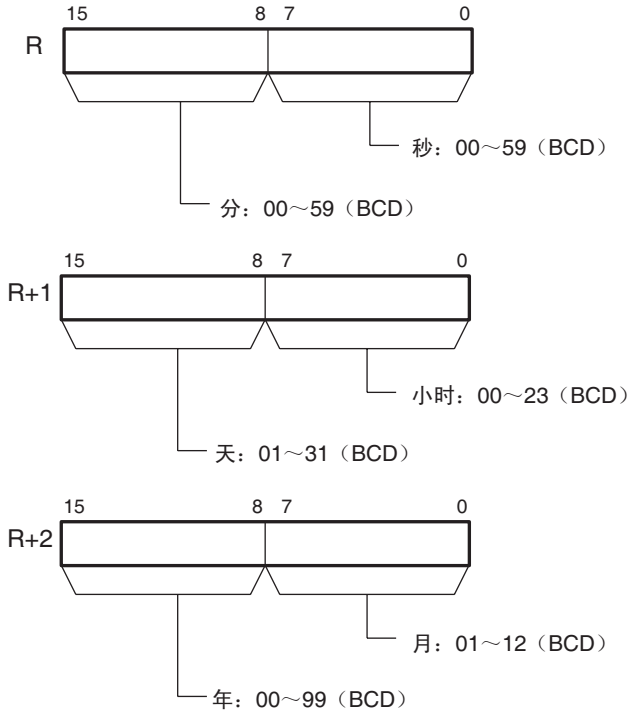


T ～ T+1：时间数据

按下图所示在 T ～ T+1 中设置时间。 T ～ T+1 必须在相同的数据区。



R ～ R+2：结果数据
R ～ R+2 包含相加的结果。R ～ R+2 必须在相同的数据区。



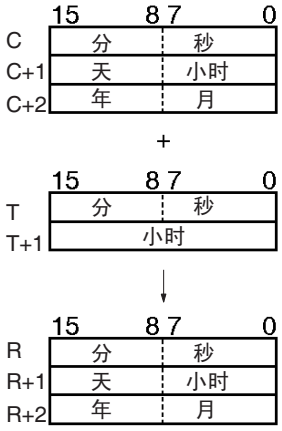
操作数规定

区域	C	T	R
CIO 区	CIO 0000 ～ CIO 6141	CIO 0000 ～ CIO 6142	CIO 0000 ～ CIO 6141
工作区	W000 ～ W509	W000 ～ W510	W000 ～ W509
保持位区	H000 ～ H509	H000 ～ H510	H000 ～ H509
辅助位区	A000 ～ A957	A000 ～ A958	A448 ～ A957
定时器区	T0000 ～ T4093	T0000 ～ T4094	T0000 ～ T4093
计数器区	C0000 ～ C4093	C0000 ～ C4094	C0000 ～ C4093
DM 区	D00000 ～ D32765	D00000 ～ D32766	D00000 ～ D32765
无区号 EM 区	E00000 ～ E32765	E00000 ～ E32766	E00000 ～ E32765
有区号 EM 区	En_00000 ～ En_32765 (n = 0 ～ C)	En_00000 ～ En_32766 (n = 0 ～ C)	En_00000 ～ 3En_2765 (n = 0 ～ C)
二进制间接 DM/EM 地址	@D00000 ～ @D32767 @E00000 ～ @E32767 @En_00000 ～ @En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ En_32767 (n = 0 ～ C)		
常数	---	仅指定值	---
数据寄存器	---		

区域	C	T	R
索引寄存器	–		
使用索引寄存器 间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR005+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

CADD(730) 将日历数据 （字 C ～ C+2）和时间数据 （字 T ～ T+1）相加，并将结果日历数据存放在 R ～ R+2。

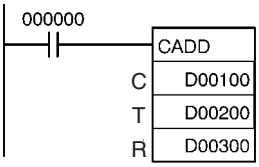


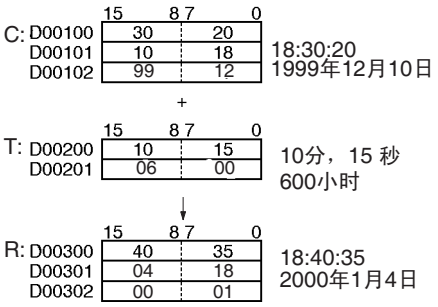
标志

名称	标记	操作
错误标志	ER	如果 C ～ C+2 中的日历数据不在指定的范围内时时置 ON。 如果 T ～ T+1 中的时间数据不在指定的范围内时时置 ON。 其它情况时 OFF。

例

下例中当 CIO 000000 变 ON，D00100 ～ D00102（年、月、天、小时、分、秒）中的日历数据与 D00200 ～ D00201（小时、分、秒）中的时间数据相加，结果输出到 D00300 ～ D00302 中。



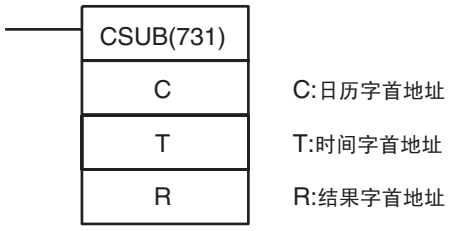


3-28-2 日历减：CSUB(731)

用途

在指定字的日历数据上减少时间。

梯形图符号



变化

变化	ON 条件时每次循环执行	CSUB(731)
	上升沿微分时执行一次	@CSUB(731)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

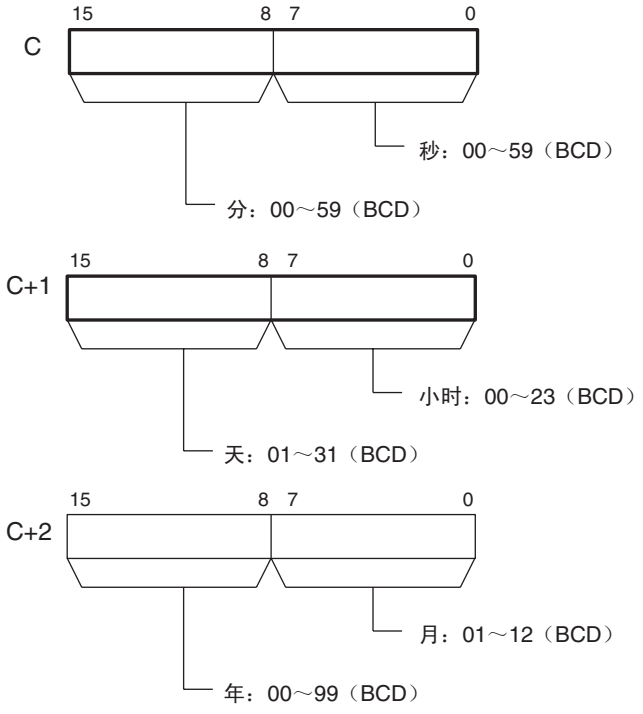
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

C ～ C+2：日历数据

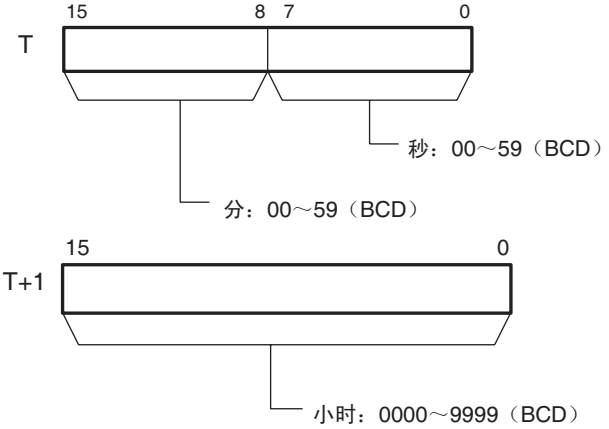
如下图所示，在 C ～ C+2 中设置日历数据。

C ～ C+2 必须在相同的数据区。



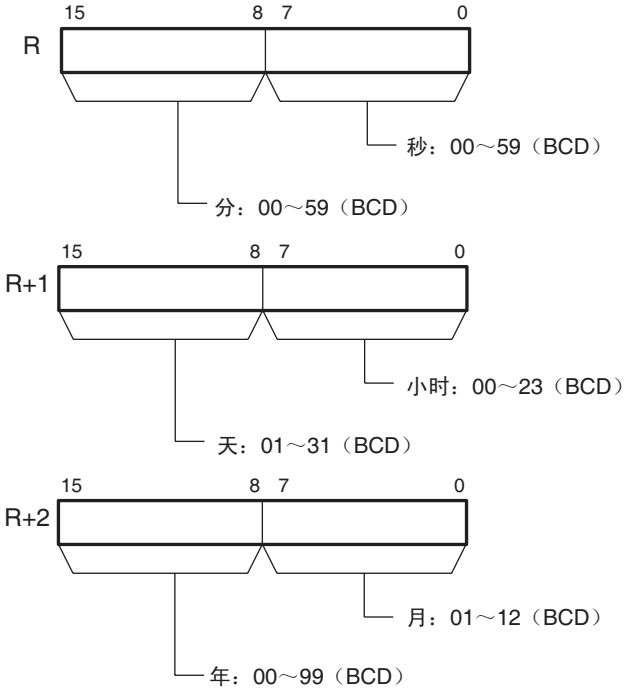
T ～ T+1：时间数据

按下图所示在 T ～ T+1 中设置时间。 T ～ T+1 必须在相同的数据区。



R ～ R+2：结果数据

R ～ R+2 包含相加的结果。R ～ R+2 必须在相同的数据区。



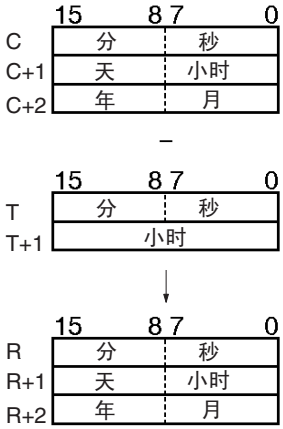
操作数规定

区域	C	T	R
CIO 区	CIO 0000 ～ CIO 6141	CIO 0000 ～ CIO 6142	CIO 0000 ～ CIO 6141
工作区	W000 ～ W509	W000 ～ W510	W000 ～ W509
保持位区	H000 ～ H509	H000 ～ H510	H000 ～ H509
辅助位区	A000 ～ A957	A000 ～ A958	A448 ～ A957
定时器区	T0000 ～ T4093	T0000 ～ T4094	T0000 ～ T4093
计数器区	C0000 ～ C4093	C0000 ～ C4094	C0000 ～ C4093
DM 区	D00000 ～ D32765	D00000 ～ D32766	D00000 ～ D32765

区域	C	T	R
无区号 EM 区	E00000 ～ E32765	E00000 ～ E32766	E00000 ～ E32765
有区号 EM 区	En_00000 ～ En_32765 (n = 0 ～ C)	En_00000 ～ En_32766 (n = 0 ～ C)	En_00000 ～ 3En_2765 (n = 0 ～ C)
二进制间接 DM/EM 地址	@D00000 ～ @D32767 @E00000 ～ @E32767 @En_00000 ～ @En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---	仅指定值	---
数据寄存器	---		
索引寄存器	---		
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR005+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

说明

CSUB(731) 从日历数据 （字 C ～ C+2）中减掉时间数据 （字 T ～ T+1），并将结果日历数据存放在 R ～ R+2。

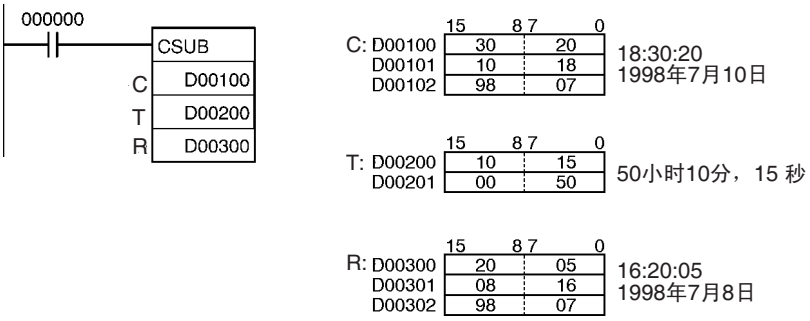


标志

名称	标记	操作
错误标志	ER	如果 C ～ C+2 中的日历数据不在指定的范围内时时置 ON。 如果 T ～ T+1 中的时间数据不在指定的范围内时时置 ON。 其它情况时 OFF。

例

下例中当 CIO 000000 变 ON，D00100 ～ D00102（年、月、天、小时、分、秒）中的日历数据减掉 D00200 ～ D00201（小时、分、秒）中的时间数据，结果输出到 D00300 ～ D00302 中。

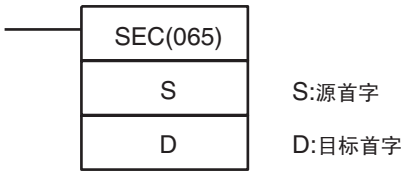


3-28-3 小时转化成秒：SEC(065)

用途

把小时 / 分 / 秒格式的时间数据转化为相等的以秒表示的时间。

梯形图符号



变化

变化	ON 条件时每次循环执行	SEC(065)
	上升沿微分时执行一次	@SEC(065)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

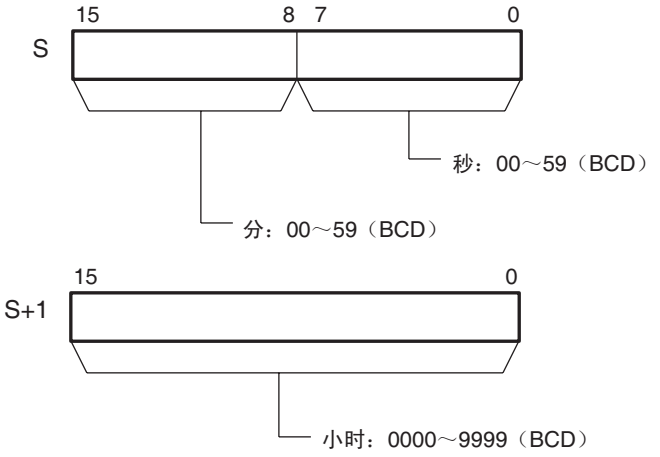
适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

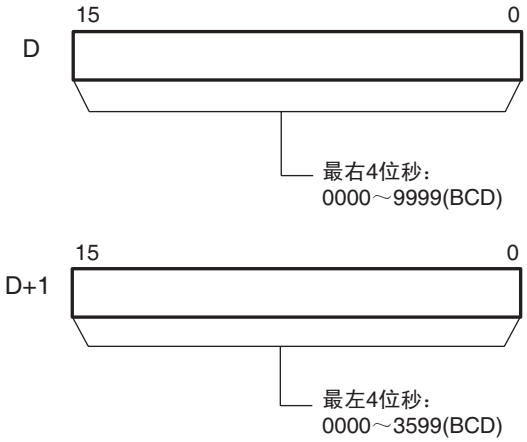
S ～ S+1：源数据

如下图所示，在 S ～ S+1 中设置小时 / 分 / 秒源数据。S ～ S+1 必须在相同的数据区。



D ～ D+1：结果数据

D ～ D+1 包含只用秒表示的结果数据。 D ～ D+1 必须在相同的数据区。

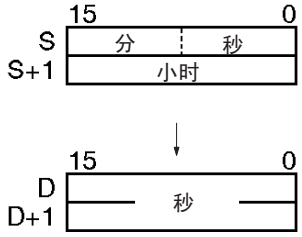


操作数规定

区域	S	D
CIO 区	CIO 0000 ～ CIO 6142	
工作区	W000 ～ W510	
保持位区	H000 ～ H510	
辅助位区	A000 ～ A958	A448 ～ A958
定时器区	T0000 ～ T4094	
计数器区	C0000 ～ C4094	
DM 区	D00000 ～ D32766	
无区号 EM 区	E00000 ～ E32766	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	仅指定值	---
数据寄存器	---	
索引寄存器	---	
使用索引寄存器 间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

说明

SEC(065) 把 S ～ S+1 中的 8 位 BCD 小时 / 分 / 秒数据转换成 8 位 BCD 以秒表示的数据，并将结果输出到 D ～ D+1 中。



标志

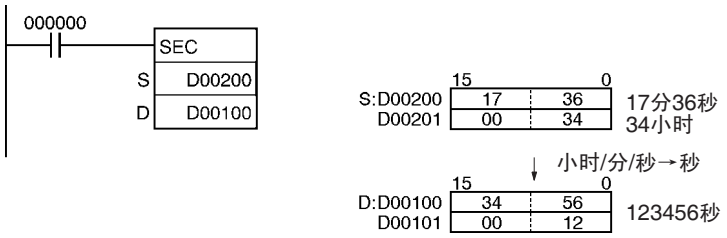
名称	标记	操作
错误标志	ER	如果 S （位 08 ～ 15）中的分钟数据不是 BCD 或不在 00 ～ 59 的范围内时置 ON。 如果 S （位 00 ～ 07）中的秒数据不是 BCD 或不在 00 ～ 59 的范围内时置 ON。 其它情况时 OFF。
等于标志	=	指令执行后如果 D 的内容是 0000 时为 ON。 其它情况时 OFF。

注意

源数据最大值是 9999 小时， 59 分， 59 秒 （35,999,999 秒）。

例

下例中当 CIO 000000 变 ON， D00200 ～ D00202 中的小时 / 分 / 秒数据转化成用秒表示的数据，结果存放在 D00100 ～ D00102 中。

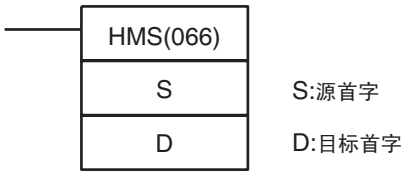


3-28-4 秒转化成小时：HMS(066)

用途

把秒数据转化为相等的以小时 / 分 / 秒格式。

梯形图符号



变化

变化	ON 条件时每次循环执行	HMS(066)
	上升沿微分时执行一次	@HMS(066)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

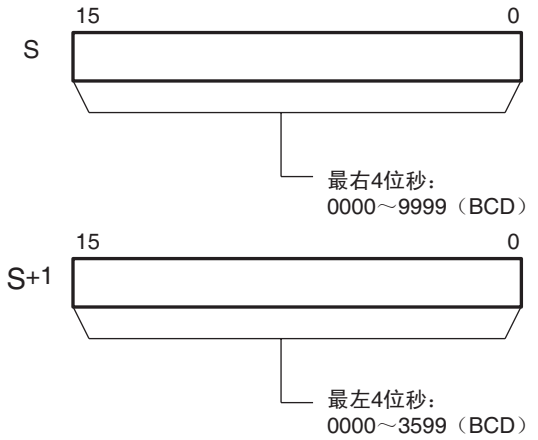
适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

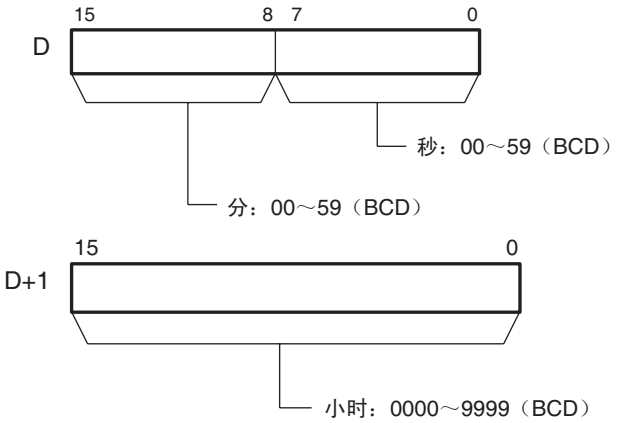
S ～ S+1：源数据

如下图所示，在 S ～ S+1 中设置以秒表示的源数据。S ～ S+1 必须在相同的数据区。



D ～ D+1：结果数据

D ～ D+1 包含小时 / 分 / 秒表示的结果数据。 D ～ D+1 必须在相同的数据区。

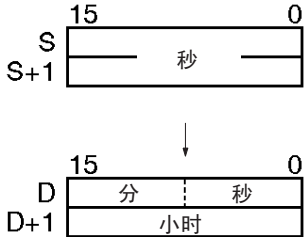


区域	S	D
CIO 区	CIO 0000 ～ CIO 6142	
工作区	W000 ～ W510	
保持位区	H000 ～ H510	
辅助位区	A000 ～ A958	A448 ～ A958
定时器区	T0000 ～ T4094	
计数器区	C0000 ～ C4094	
DM 区	D00000 ～ D32766	
无区号 EM 区	E00000 ～ E32766	
有区号 EM 区	En_00000 ～ En_32766 (n = 0 ～ C)	

区域	S	D
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	00000000 ～ 35999999 (BCD)	---
数据寄存器	---	
索引寄存器	---	
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 to IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

说明

HMS(066) 把 S ～ S+1 中的 8 位 BCD 秒数据转换成 8 位 BCD 以小时 / 分 / 秒表示的数据，并将结果输出到 D ～ D+1 中。



标志

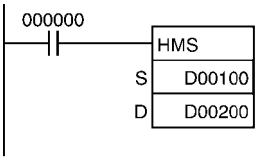
名称	标记	操作
错误标志	ER	如果 S 和 S+1 中的秒数据不是 BCD 或不在 0 ～ 35,999,999 的范围内时置 ON。 其它情况时 OFF。
等于标志	=	指令执行后如果 D 的内容是 0000 时为 ON。 其它情况时 OFF。

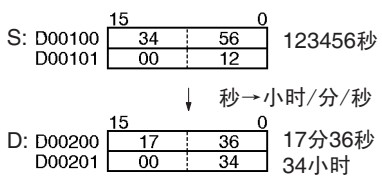
注意

源数据最大值是 35,999,999 秒 （9999 小时， 59 分， 59 秒）。

例

下例中当 CIO 000000 变 ON，D00100 ～ D00102 中的秒数据（123,456 秒）转化成用小时 / 分 / 秒表示的数据，结果存放在 D00200 ～ D00202 中。





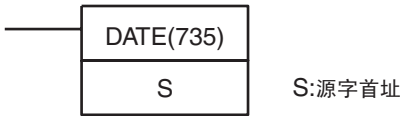
3-28-5 时钟调整：DATE(735)

用途

以指定源字中的设置或改变内部时钟的设置。

注 也可以通过外围设备或时钟写 FINS 命令 (0720) 来修改内部时钟设置。

梯形图符号



变化

变化	ON 条件时每次循环执行	DATE(735)
	上升沿微分时执行一次	@DATE(735)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

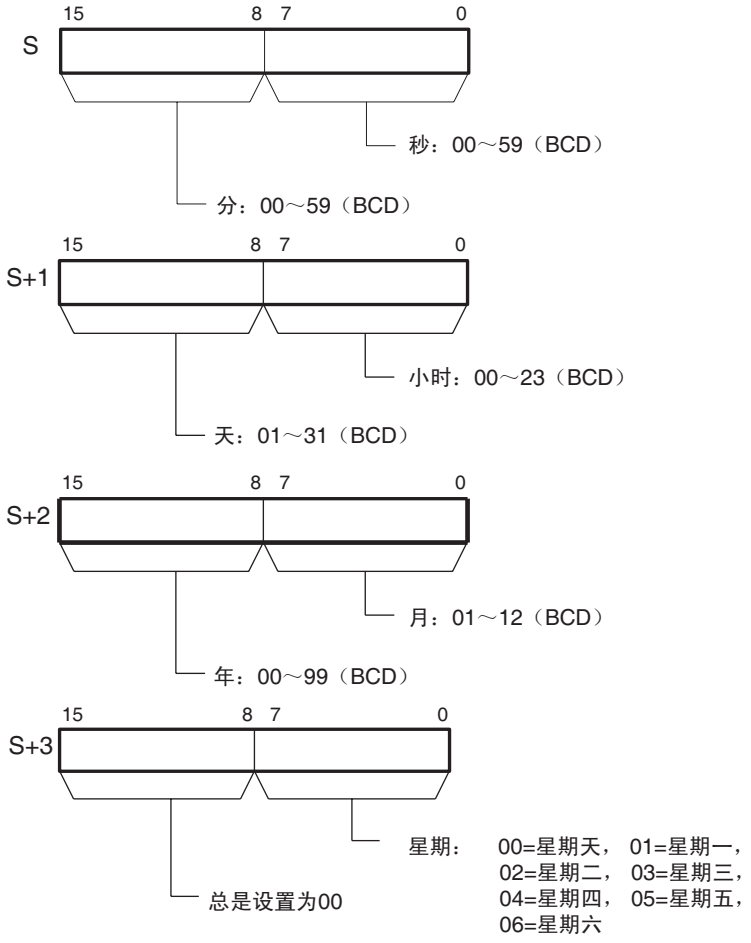
适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

S ～ S+3：新的时钟设置

如下图所示，在 S ～ S+3 中设置新的时钟设置。S ～ S+3 必须在相同的数据区。



下表显示日历 / 时钟区的结构。

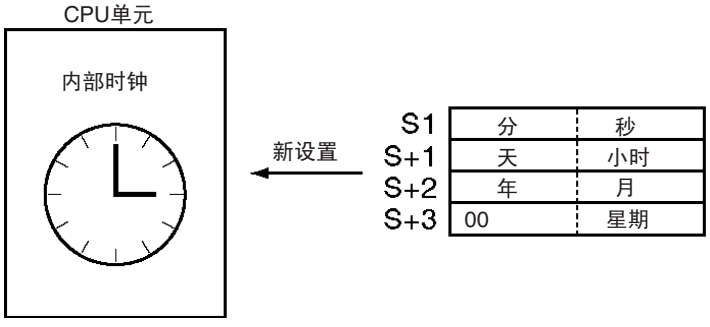
地址	内容
A35100 ～ A35107	秒 （00 ～ 59， BCD）
A35108 ～ A35115	分 （00 ～ 59， BCD）
A35200 ～ A35207	小时 （00 ～ 23， BCD）
A35208 ～ A35215	天 （00 ～ 31， BCD）
A35300 ～ A35307	月 （01 ～ 12， BCD）
A35308 ～ A35315	年 （00 ～ 99， BCD）
A35400 ～ A35407	星期 （00 ～ 06= 星期天～星期六，十六进制）
A35408 ～ A35415	总为 00

操作数规定

区域	S
CIO 区	CIO 0000 ～ CIO 6140
工作区	W000 ～ W508
保持位区	H000 ～ H508
辅助位区	A000 ～ A956
定时器区	T0000 ～ T4092
计数器区	C0000 ～ C4092
DM 区	D00000 ～ D32764
无区号 EM 区	E00000 ～ E32764
有区号 EM 区	En_00000 ～ En_32764 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767
常数	---
数据寄存器	---
索引寄存器	---
使用索引寄存器间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

说明

DATE(735) 按照 4 个源字中的时钟数据改变内部时钟设置。新的内部时钟立即在日历 / 时钟区 (A351 ～ A354) 中反映出来。

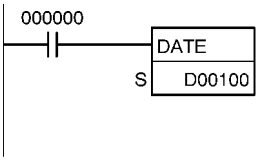


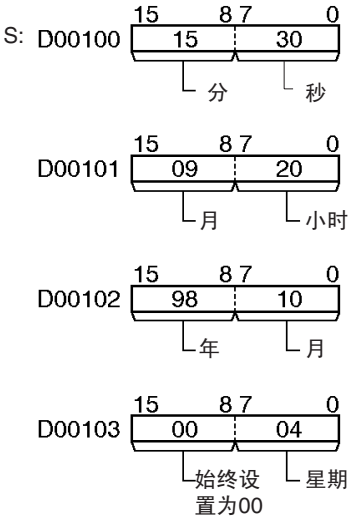
标志

名称	标记	操作
错误标志	ER	如果 S 和 S+3 中的新的时钟设置不在指定范围时置 ON。 其它情况时 OFF。

注意 即使内部时钟被设置成一个不存在的日期（例如 11 月 31 日），也不会产生错误。

例 下例中当 CIO 000000 变 ON，内部时钟被设置成 1998 年 10 月 9 日，星期四，20:15:30 秒。





3-29 调试指令

3-29-1 跟踪存储采样：TRSM(045)

用途 当执行 TRSM(045) 时，预先选定的位或字的状态被采样并存储到跟踪存储区。TRSM(045) 可以在程序任何地方使用任何次数。

梯形图符号

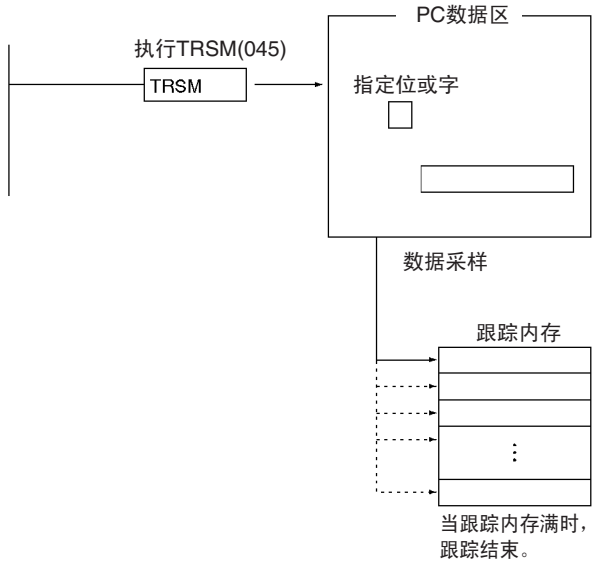
变化

变化	ON 条件时每次循环执行	TRSM(045)
	上升沿微分时执行一次	不支持
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

说明 在执行 TRSM(045) 之前，要跟踪的位或字必须先用外围设备来指定。每次执行 TRSM(045)，指定的位或字的当前值被采样并按序记录到跟踪存储器中。当跟踪存储器记录满时结束跟踪。必要时跟踪存储器的内容可以从外围设备上监视。

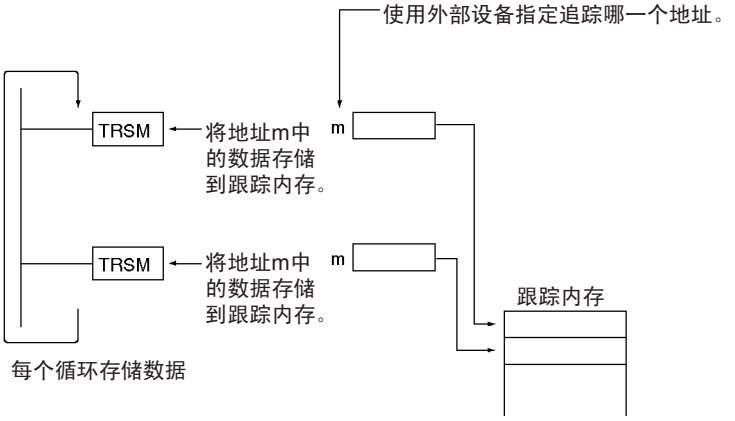


这个指令仅仅指明指定的数据何时采样，所有其它的设置和数据追踪操作都是使用外围设备来设置的。另外两种控制数据采样的方法是在每次循环的结尾处采样和以指定的时间间隔采样 （与循环时间无关）。

TRSM(045) 不需要执行条件，总是假定它的执行条件为 ON。把 TRSM(045) 直接与左边母线相连。

使用TRSM(045)在指令的执行条件为ON时对程序中该点指定的字或位的值采样。如果指令的执行条件每个循环都是 ON，每次循环指定的字或位的值都存储在跟踪内存中。

一个程序中允许 2 个或更多的 TRSM(045) 合并使用。在这种情况下，相同的指定字或位的值在每次执行 TRSM(045) 指令时都被存储在跟踪内存中。



注 数据跟踪的细节请参照外部设备手册。

使用外部设备完成的数据跟踪操作归纳如下。

1,2,3... 1. 使用外部设备设置以下参数。

a) 设置要跟踪的字或位的地址。
b) 设置触发条件。下列三种条件之一能够控制数据何时存储到跟踪内存中是有效的。
i) 跟踪启动位从 OFF 变 ON。
ii) 一个指定的位从 OFF 变 ON。
iii) 指定字的值与设置的值匹配。
c) 在程序执行 TRSM(045) 时设置采样时间间隔为 “TRSM”。
d) 设置延迟。

2. 当用外部设备把采样启动位的状态从 OFF 变 ON 时，每次执行 TRSM(045) 都将采样指定的数据，并把采样数据存储到跟踪内存。同时跟踪忙标志 (A50813) 将变 ON。
3. 当触发条件满足时 （跟踪启动位 ON，指定位 ON 或指定字的值与设置值相匹配），从下一次采样加上或减去由延迟设置的采样数开始采样的数据将有效。同时跟踪触发监视标志 (50811) 将变 ON。
4. 当 TRSM(045) 已执行多次，使得跟踪内存满时，跟踪将结束。当跟踪结束，跟踪结束标志 (A50812) 将变 ON，跟踪忙标志 (A50813) 将变 OFF。
5. 用外部设备读取跟踪内存的内容。

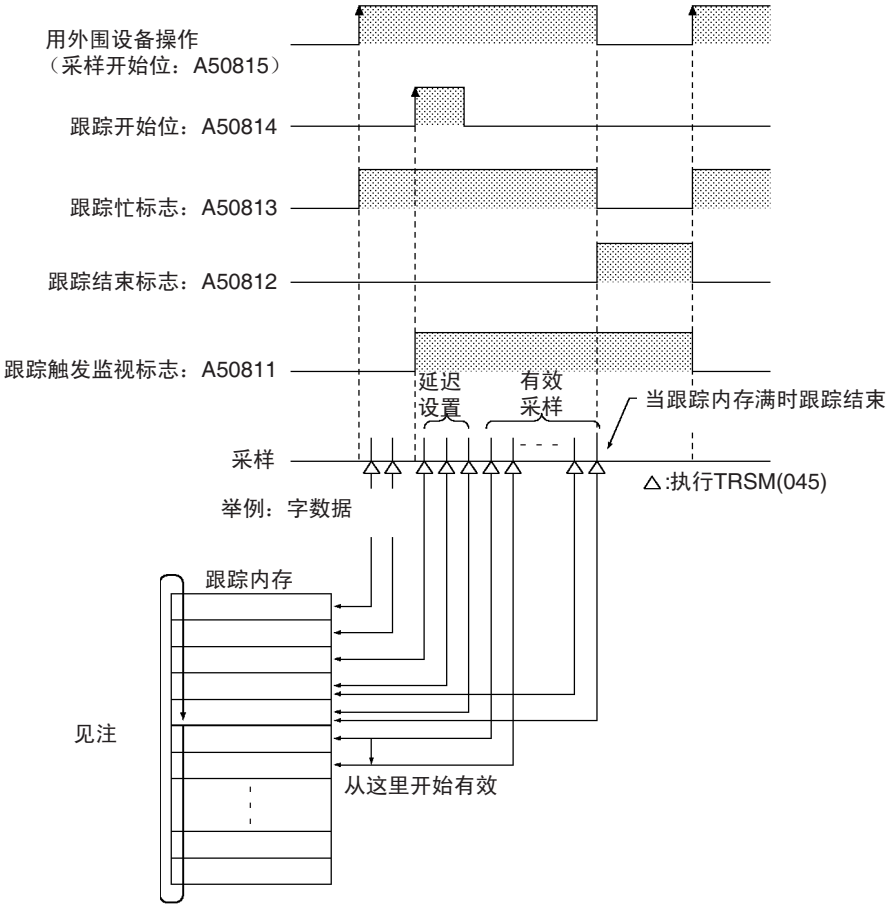
下表显示了辅助区的相关位和标志。只有 A50814 和 A50815 可以由用户控制，A00815 不能由程序置 ON，即它必须由外部设备置 ON。

名称	地址	操作
跟踪触发监视标志	A50811	当使用跟踪启动位建立触发条件时，该标志变 ON。当采样启动下一次跟踪时，该位变 OFF。（由采样启动位）
跟踪结束标志	A50812	当跟踪采样已填满跟踪内存区时该标志变 ON。当下次采样启动位从 OFF 变 ON 时该位变 OFF。
跟踪忙标志	A50813	当采样启动位从 OFF 变 ON 时该位变 ON。当跟踪结束时该位变 OFF。
跟踪启动位	A50814	当该位从 OFF 变 ON 时跟踪触发条件被建立。在指定的延迟之后 （正延时）采样被记录或指定的已存在的采用数据数将有效 （负延时）。
采样启动位	A50815	当该位由外围设备从 OFF 置 ON 时，将进行采样。用下列三种方法之一开始将数据存储到跟踪内存： 1）周期采样 （10 ～ 2550ms 间隔） 2）当执行 TRSM(045) 时采样 3）每次循环结束时采样 该位必须由外围设备置 ON 或置 OFF。

注意 当未执行数据跟踪，或未用外部设备在参数中设置采样间隔而使用执行 TRSM(045) 指令来采样时， TRSM(045) 等同于 NOP(000)。

不要在程序中将采样启动位 (A50815) 置 ON 或 OFF。该位必须由外部设备来置 ON 或置 OFF。

例 下例显示整个数据跟踪操作。



注 跟踪内存具有环行结构。数据被存储到跟踪内存的结尾，然后环绕到区域的开始，在第一个有效的数据采样前结束。

3-30 错误诊断指令

本节描述用于定义和处理错误的指令。

指令	助记符	功能代码	页
错误报警	FAL	006	934
严重错误报警	FALS	007	942
错误点检测	FPD	269	950

3-30-1 错误报警：FAL(006)

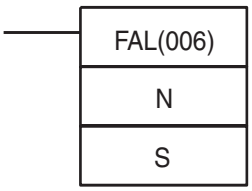
用途

产生或清除用户定义的非致命错误。非致命错误不停止 PLC 运行。

对 CS1-H，CJ1-H 和 CJ1M CPU 单元，FAL（006）也能用来产生非致命系统错误。

梯形图符号

- 产生或清除用户定义的非致命错误

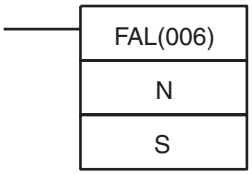


N：FAL 号

M：信息首字或

常数 （0000 ～ FFFF）

- 产生非致命系统错误 （仅 CS1-H， CJ1-H， CJ1M 或 CS1D）



N：FAL 号 （在 A529）

M：包含错误码和

错误详情首字

变化

变化	ON 条件时每次循环执行	FAL(006)
	上升沿微分时执行一次	@FAL(006)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

当 FAL(006(用于产生或清除用户定义的错误操作数功能与用 FAL(006) 产生系统错误 （仅 CS1-H， CJ1-H， CJ1M CPU 单元）的操作数功能有稍许不同。

产生或清除用户定义的非致命错误

下表显示了操作数的功能。

注　操作数 N 的值必须与 A529（系统产生的 FAL/FALS 号）的内容不同。

N	S	功能
0	#0001 ～ #01FF	清除带对应 FAL 号的非致命错误
	#FFFF	清除所有非致命错误
	其它 *	清除最严重非致命错误
1 ～ 511（这些 FAL 号与 FALS 共享）。	#0000 ～ #FFFF	产生一个带对应 FAL 号的非致命错误 （无信息）。
	字地址	产生一个带对应 FAL 号的非致命错误。 S ～ S+7 中包含的 16 字符的 ASCII 信息将显示在编程设备上。

注　其它设置是常数 #0200 ～ #FFFE 或一个字地址。

产生非致命系统错误 （仅 CS1-H， CJ1-H， CJ1M 或 CS1D）

下表显示了操作数的功能。

注　操作数 N 的值必须与 A529 的内容相同。（产生 FAL/FALS 号的系统）

操作数	功能
N	1 ～ 511 （这些 FAL 号与 FALS 共享）
S	产生的错误代码。（见下面的*说明*）
S+1	产生的错误详细代码。（见下面的*说明*）

操作数规定

区域	N	S
CIO 区	---	CIO 0000 ～ CIO 6143
工作区	---	W000 ～ W511
保持位区	---	H000 ～ H511
辅助位区	---	A000 ～ A959
定时器区	---	T0000 ～ T4095
计数器区	---	C0000 ～ C4095
DM 区	---	D00000 ～ D32767
无区号 EM 区	---	E00000 ～ E32767
有区号 EM 区	---	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	---	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 地址	---	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	0 ～ 511	#0000 ～ #FFFF （二进制）
数据寄存器	---	

区域	N	S
索引寄存器	---	
使用索引寄存器 间接寻址	---	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

说明

FAL(006) 的操作与 N 的值有关。将 N 设置为 0000 以清除错误，将 N 设置为 0001 ～ 01FF 以产生一个错误。如果 N 的值与 A529 的内容相同（仅 CS1-H，CJ1-H， CJ1M CPU 单元），将产生一个系统错误。

产生用户定义的非致命错误

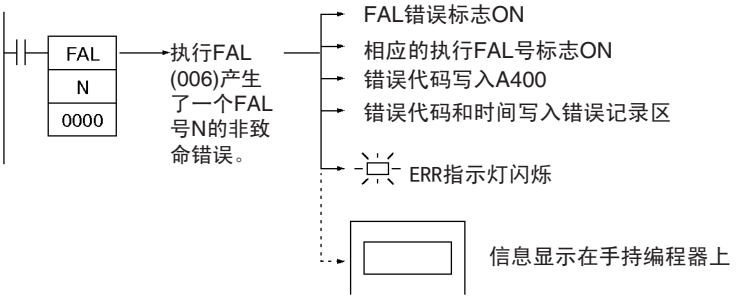
当执行 FAL(006) 且 N 设置了一个与 A529 （产生 FAL/FALS 系统号）内容不同的 FAL 号（&1 ～ &511），将产生该 FAL 号的非致命错误，并将执行下列过程:

1,2,3...

1. FAL 错误标志 (A40215) 将变 ON （PLC 将继续运行）。
2. 对应 FAL 号的执行 FAL 号标志将变 ON：标志 A36001 ～ A39115 对应于 FAL 号 0001 ～ 01FF(1 ～ 511)。
3. 错误代码将写入 A400。对应于 FAL 号 0001 ～ 01FF(1 ～ 511) 错误代码是 4101 ～ 42FF。

   注 在 FAL(006) 指令执行时，同时发生了一个致命错误或一个更严重的非致命错误，最严重的错误代码将写入 A400。
4. 错误发生的时间和错误代码将写入错误记录区 (A100 ～ A199)。

   注 对CS1-H，CJ1-H，CJ1M CPU单元，如果设置PLC设置以使FAL(006)产生的错误不被记录，错误记录不会写入错误记录区，即，如果手持编程器地址 129 位 15 设为 1。
5. CPU 单元上的 ERR 指示灯将闪烁。
6. 如果在 S 中指定了一个字地址，以 S 开始的信息将被记录（显示在编程设备上）。



下表显示错误代码和 FAL(006) 的 FAL 错误标志。

FAL 号	FAL 错误代码	执行 FAL 号标志
1 ～ 511（十进制）	4101 ～ 42FF	A36001 ～ A39115

显示用户定义的非致命错误

如果 S 是一个字地址并且其中已存储了 ASCII 信息，当 FAL(006) 执行时，该信息将显示在编程设备上。（如果不需要信息，在 S 中设置一常数）

当执行 FAL(006) 时，以 S 开始的信息将被登录。一旦信息登录，当手持编程器连接时它将显示出来。

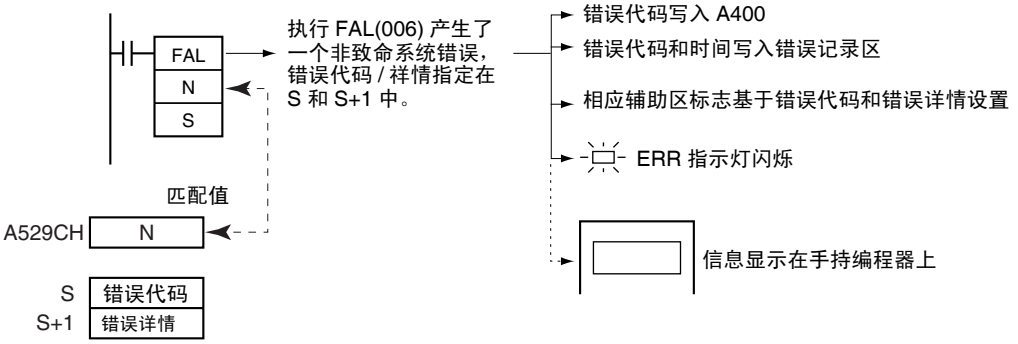
在 S ～ S+7 中可以存储最多 16 个字符长的 ASCII 信息。每个字中最左边的字节将首先显示。

信息的结束代码是空字符（十六进制 00）。如果省略了空字符，S ～ S+7 中 16 个字符都将显示出来。

如果执行 FAL(006) 之后包含信息的字的内容发生了改变，信息将相应地改变。

产生非致命系统错误 （仅 CS1-H， CJ1-H， CJ1M 或 CS1D）

当执行 FAL(006) 且 N 设置了一个与 A529 （产生 FAL/FALS 系统号）内容相同的 FAL 号 (&1 ～ &511)，将产生一个非致命错误，错误代码和错误相信详细代码指定在 S 和 S+1 中。同时执行下列过程：



1,2,3...

1. 指定的错误代码将写入 A400。
2. 错误发生的时间和错误代码将写入错误记录区 (A100 ～ A199)。
3. 专用的辅助区标志基于错误代码和错误详情设置。
4. CPU 单元上的 ERR 指示灯将闪烁， PLC 操作继续。
5. 指定系统错误的非致命错误信息将显示在手持编程器上。

注

1. 当调试程序时，FAL(006) 能产生系统的非致命错误。例如，可故意产生一个系统错误以检查是否错误信息能适当地显示在接口上，例如可编程终端 (PT)。
2. A529（产生的 FAL/FALS 系统号）中的值是系统故意产生一个非致命错误时用的 FAL 的虚号 （FAL、 FALS 和 FPD 共享）。这个号是一个 FAL 虚号，所以它不改变执行 FAL 号标志或错误代码的状态。当需要产生两个或更多系统错误 （致命和 / 或非致命错误）时，用 A529 和 N 中的相同值，但 S 和 S+1 中的不同值，多于一次地执行 FAL/FALS/FPD 指令能产生不同的错误。

3. 在 FAL(006) 指令执行时，同时发生了一个更严重的错误 （系统产生致命错误或 FALS(007) 错误），更严重的错误代码将写入 A400。
4. 为了清除一个 FAL(006) 产生的系统错误，置 PLC 为 OFF，然后再置 ON。PLC 将保持 ON，但仍需要相同的过程以清除错误好象指定错误实际发生似的。

下表显示了如何在 S 和 S+1 中指定错误代码和错误详情。

错误名字	S	S+1
中断任务错误	008B 十六进制	• 位 15 OFF: 中断任务错误 位 00 ～ 14: 当发生错误出现时的中断任务号 • 位 15 ON: 执行与特定 I/O 刷新冲突的中断任务 位 00 ～ 14: 有刷新冲突的特定 I/O 单元的单元号
基本 I/O 错误	009A 十六进制	错误产生单元的机架位置 • 位 08 ～ 15: 影响的匹配单元所在机架的机架号（二进制） • 位 00 ～ 07: 影响的匹配单元所在插槽的插槽号（二进制）
PLC 设置错误	009B 十六进制	PLC 设置错误位置
I/O 表验证错误	00E7 十六进制	… （不定）
内部板非致命错误	02F0 十六进制	内板错误信息 • 位 00 ～ 03：无效 • 位 04 ～ 15：由内部板定义的错误
CS1 CPU 总线单元错误	0200 十六进制	CS1 CPU 总线单元的单元号： 0000 ～ 000F 十六进制
特定 I/O 单元错误	0300 十六进制	特定 I/O 单元的单元号：0000 ～ 005F 十六进制 或 00FF 十六进制 （单元号不确定）
SYSMAC 总线错误	00A0 十六进制	SYSMAC 总线主单元的单元号： 0000 ～ 0001 十六进制
电池错误	00F7 十六进制	… （不定）
CS1 CPU 总线单元设置错误	0400 十六进制	CS1 CPU 总线单元的单元号： 0000 ～ 000F 十六进制
特定 I/O 单元设置错误	0500 十六进制	特定 I/O 单元的单元号：0000 ～ 005F 十六进制

禁止用户定义错误的错误记录登录 （仅 CS1-H， CJ1-H， CJ1M 或 CS1D）

一般当 FAL(006) 产生一个用户定义的错误时，错误产生的错误代码和时间 被写入到错误记录区 (A100 ～ A199)。可设置 PLC, 使 FAL(006) 产生的用户定义错误不记录到错误记录中。

即使错误不记录到错误记录中，FAL 错误标志 (40215) 将变 ON，在执行的 FAL 号标志 (A36001 ～ A39115) 的相应标志将变 ON，错误代码将写到 A400。

当你仅想记录系统产生的错误时，禁止用户定义的FAL(006) 错误的错误记录登录。例如，在调试时，如果 FAL(006) 指令用于几个场合，并且错误记录充满了用户定义的 FAL(006) 错误，这个功能是有用的。下表显示了 PLC 设置的设置:

项目	设置	
手持编程器设置地址	字	129
	位	15
名称	FAL 错误记录注册	
设置	0: 在错误记录中记录 FAL 错误 1: 不在错误记录中记录 FAL 错误	
缺省设置	0: 在错误记录中记录 FAL 错误	
读 PLC 设置的时间	每次循环 （当一个 FAL 错误产生时）	

只要 PLC 设置字 129 位 15 设为 1（不在错误记录中记录 FAL 错误），将记录下面的错误：

- 由 FAL(007) 产生的致命错误
- 系统产生的非致命错误
- 系统产生的致命错误
- 由 FAL(006) 或 FPD(269) 故意产生的系统非致命错误
- 由 FAL(007) 故意产生的非致命系统错误

无需编程器清除非致命错误

1. 清除用户定义的非致命错误
   当将 N 设为 0 执行 FAL(006) 时，能够清除非致命错误。S 的值决定了这个过程，如下表所示。

S	过程
&1 ～ &511（0001 ～ 01FF 十六进制）	清除指定号的 FAL 错误
FFFF 十六进制	清除所有的非致命错误 （包括系统错误）
0200～FFFE十六进制或指定字	产生的最严重的非致命错误 （即使它是一个非致命系统错误）。 当多于一个的 FAL 错误产生时，清除最小号的 FAL 错误。

2. 清除非致命系统错误 （仅 CS1-H， CJ1-H， CJ1M 和 CS1D CPU 单元）
   有两种清除由 FAL(006) 产生的非致命系统错误的方法。
   - 变 PLC 为 OFF，再变为 ON。
   - 当保持 PLC 为 ON 时，必须清除系统错误就象指定错误实际发生了似的。

标志

名称	标记	操作
错误标志	ER	如果 N 不在指定的范围 0 ～ 511 （十进制）时为 ON。 如果一个非致命系统错误产生，而指定错误代码或错误详情不正确时为 ON。 其它情况时 OFF。

下面这些表显示了在辅助区的相关位和标志。

- 仅用于户定义错误的辅助区字 / 标志

名称	地址	操作
FAL 错误标志	A40215	当由 FAL(006) 产生错误时为 ON。
执行的 FAL 号标志	A36001 ～ A39115	当一个错误由 FAL(006) 产生时，相应标志变为 ON。 对于 FAL 号 0001 ～ 01FF 相应的标志为 A36001 ～ A39115。

- 仅用于系统错误的辅助区字/标志（仅CS1-H，CJ1-H，CJ1M和CS1D CPU单元）

名称	地址	操作
产生的系统 FAL/FALS 号	A529	当一个系统错误由 FAL(006) 产生时，用 FAL/FALS 的虚号。在字 （0001 ～ 01FF 十六进制， 1 ～ 511 十进制）中设这个 FAL/FALS 的虚号。

- 用户定义和系统错误的辅助区字 / 标志

名称	地址	操作
错误记录区	A100 ～ A199	错误记录区包含最近 20 个错误的错误代码和时间 / 日期，包括由 FAL(006) 产生的错误。
错误代码	A400	当一个错误产生时，相应标志变为它的错误代码存储在 A400 中。 FAL 号 0001 ～ 01FF 的错误代码分别是 4101 ～ 42FF。 如果同时产生两个或两个以上的错误，最严重错误的错误代码将存储在 A400。

注意 N 必须在 0001 ～ 01FF 之间，如果 N 在这个范围之外，将发生错误，错误标志将变 ON。

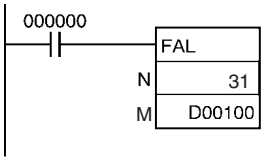
例 **产生一个非致命错误**

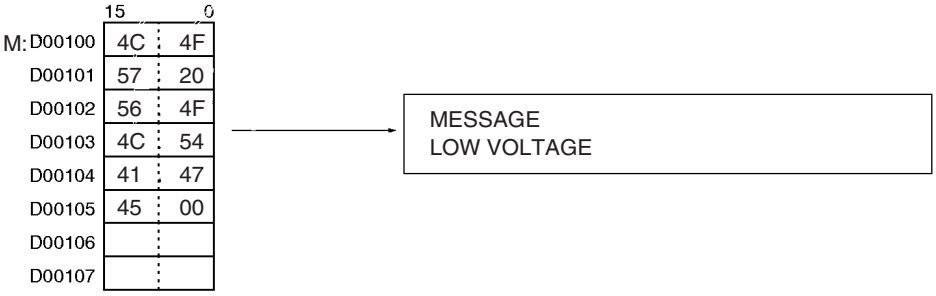
下例中当 CIO 000000 是 ON 时，FAL(006) 将产生一个 FAL 号为 31 的非致命错误，并将执行下列过程：

1,2,3...

1. FAL 错误标志 (A40215) 将变 ON。
2. 对应的执行 FAL 号标志 (A36114) 将变 ON。
3. 错误代码 (411F) 将写入 A400。

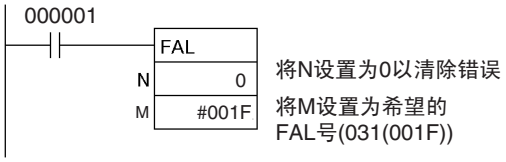
   注 在 FAL(006) 指令执行时，同时发生了一个致命错误或一个更严重的非致命错误，最严重的错误代码 （最高错误代码）将写入 A400。
4. 错误发生的时间 / 代码和错误代码将写入错误记录区 （A100 ～ A199）。
5. CPU 单元上的 ERR 指示灯将闪烁。
6. D00100 ～ D00107 中的 ASCII 信息将显示在外围设备上。（如果不需要显示信息，在 S 中设置一个常数）。





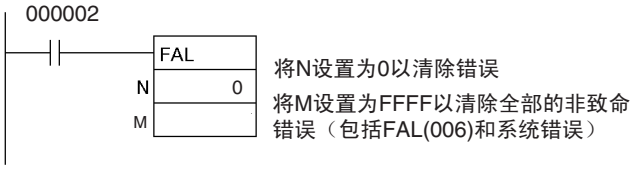
清除一个特定的非致命错误

下例中当 CIO 000001 是 ON 时，FAL(006) 将清除一个 FAL 号为 31 的非致命错误，使相应的已执行 FAL 标志 (A36114) 变 OFF，并使 FAL 错误标志 (A40215) 变 OFF。



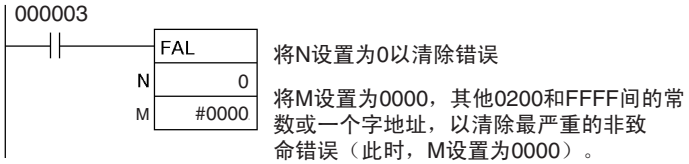
清除全部的非致命错误

下例中当 CIO 000002 是 ON 时，将清除全部的非致命错误，使已执行 FAL 标志 (A36001 ～ A36115) 变 OFF，并使 FAL 错误标志 (A40215) 变 OFF。



清除最严重的非致命错误

下例中当 CIO 000003 是 ON 时，FAL(006) 将清除已发生的最严重的非致命错误，并重新设置 A400 中的错误代码。如果清除的错误是 FAL(006) 产生的，对应的已执行 FAL 号标志和 FAL 错误标志 (A40215) 变 OFF。



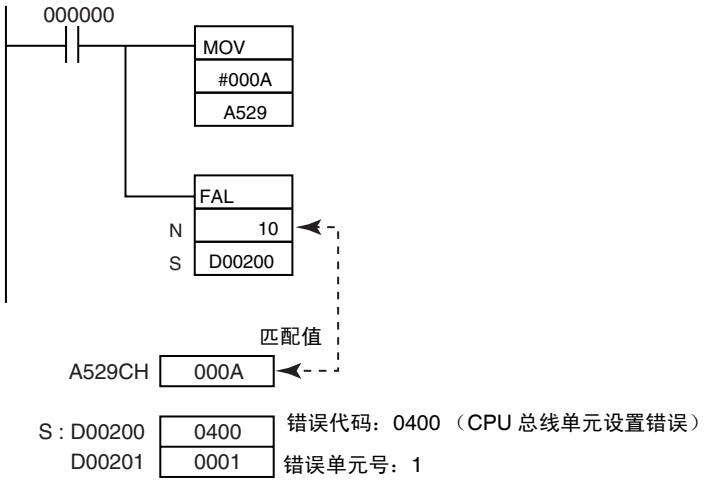
产生非致命系统错误 （仅 CS1-H， CJ1-H， CJ1M 或 CS1D）

下例中当 CIO 000000 是 ON 时，FAL(006) 将在单元号 1 产生一个 CPU 总线单元设置错误。本例中 FAL 虚号是 10，相应的值 (000A Hex) 存储在 A529。

1,2,3... 1. 如果这个错误是最严重错误，则指定错误代码 (0400) 将写入 A400。

2. 错误代码和错误发生的时间 / 日期将写入错误记录区 (A100 ～ A199)。

3. CPU 总线单元设置错误标志 (A40203) 和单元号 1(A42701) 的 CPU 总线单元设置错误标志将变为 ON。
4. CPU 单元上的 ERR 指示灯将闪烁。
5. 一个信息 (CPU BU ST ERR 01) 将显示在手持编程器上以指示 CPU 总线单元 1 发生错误。



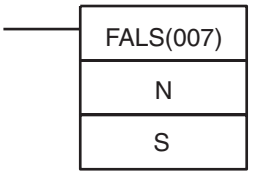
3-30-2 严重错误报警：FALS(007)

用途

产生用户定义的致命错误。致命错误停止 PLC 运行。
对CS1-H，CJ1-H和CJ1M CPU单元，FALS(007)也能用来产生致命系统错误。

梯形图符号

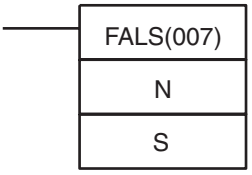
- 产生或清除用户定义的致命错误



N:FAL 号

S: 信息首字或常数 （0000 ～ FFFF）

- 产生致命系统错误 （仅 CS1-H， CJ1-H， CJ1M 或 CS1D）



N:FAL 号 （在 A529）

S: 包含错误和错误详情首字

变化

变化	ON 条件时每次循环执行	FALS(007)
	上升沿微分时执行一次	不支持
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

产生用户定义的致命错误

下表显示了操作数的功能。

注　操作数 N 的值必须与 A529（系统产生的 FAL/FALS 号）的内容不同。

操作数	功能
N	1 ～ 511 （这些 FALS 号与 FAL 共享）。
S	指定 8 个字中的首字，这 8 个字包含显示在编程设备上的 ASCII 信息。如果信息不需要，指定一个常数 （0000~FFFF）。

产生致命系统错误 （仅 CS1-H， CJ1-H， CJ1M 或 CS1D）

下表显示了操作数的功能。

注　操作数 N 的值必须与 A529 的内容相同。（系统产生的 FAL/FALS 号）

操作数	功能
N	1 ～ 511 （这些 FALS 号与 FAL 共享）。
S	产生的错误代码。（见下面的*描述*）
S+1	产生的错误详细代码。（见下面的*描述*）

操作数规定

区域	N	S
CIO 区	---	CIO 0000 ～ CIO 6143
工作区	---	W000 ～ W511
保持位区	---	H000 ～ H511
辅助位区	---	A000 ～ A959
定时器区	---	T0000 ～ T4095
计数器区	---	C0000 ～ C4095
DM 区	---	D00000 ～ D32767
无区号 EM 区	---	E00000 ～ E32767
有区号 EM 区	---	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	---	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 地址	---	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	仅指定值	#0000 ～ #FFFF （二进制）
数据寄存器	---	
索引寄存器	---	
使用索引寄存器间接寻址	---	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR+(++)0 ～ ,IR15+(++) ,–(– –)IR0 ～ ,–(– –)IR15

描述

FALS(007) 一个致命错误。在 CS1-H，CJ1-H，CJ1M CPU 单元，FALS(007) 也能用于产生致命系统错误及用户定义致命错误。（如果 N 的值与 A529 内容相同，将产生一个系统错误）。

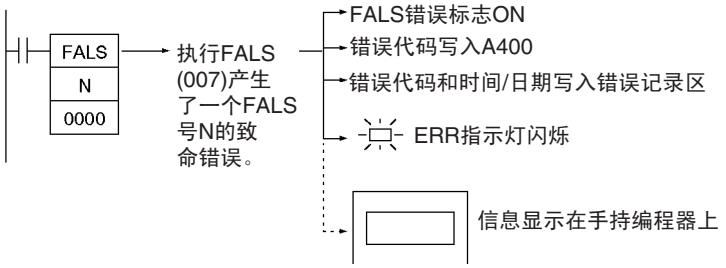
产生用户定义的致命错误

当执行 FALS(007) 且 N 设置了一个 FALS 号 (1~511) 与 A529 （系统产生的 FAL/FALS 号）内容不同，将产生该 FALS 号的致命错误，并将执行下列过程:

1,2,3...

1. FALS 错误标志 (A40106) 将变 ON （PLC 将停止运行）。
2. 错误代码将写入 A400。对应于 FAL 号 0001 ～ 01FF(1 ～ 511) 错误代码是 C101 ～ C2FF。

   注 在 FALS(007) 指令执行时，同时发生了一个致命错误或一个更严重的致命错误，最严重的错误代码将写入 A400。
3. 错误代码和错误发生的时间 / 日期将写入错误记录区 (A100 ～ A199)。
4. CPU 单元上的 ERR 指示灯将闪烁。
5. 如果在 S 中指定了一个字地址，以 S 开始的 ASCII 信息将被记录（显示在外部设备上）。



下表显示 FALS(007) 的错误代码。

FALS 号	FALS 错误代码
1 ～ 511	C101 ～ C2FF

注 对于 CX-Programmer 和手持编程器，FALS 号 N 的输入方法不同。CX-Programmer 输入 #1 ～ #511，手持编程器 001 ～ 511。

显示用户定义的致命错误

如果 S 是一个字地址并且其中已存储了 ASCII 信息，当 FALS(007) 执行时，该信息将显示在编程设备上。（如果不需要信息，在 S 中设置一常数）。

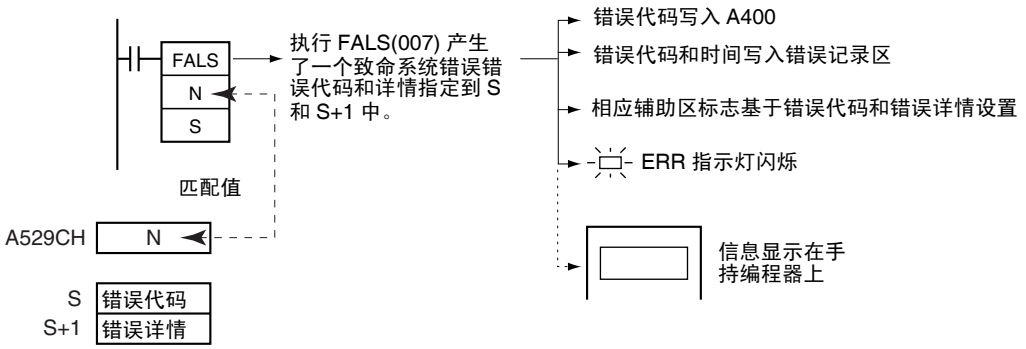
当执行 FALS(007) 时，以 S 开始的信息将被登录。一旦信息登录，当手持编程器连接时它将显示出来。

在 S ～ S+7 中最多可以存储 16 个字符长的 ASCII 信息。每个字中最左边的字节分首先显示。

信息的结束代码是空字符 （十六进制 00）。如果省略了空字符， S ～ S+7 中 16 个字符都将显示出来。

如果执行 FALS(007) 之后包含信息的字的内容发生了改变，信息将相应地改变。

产生致命系统错误 （仅 CS1-H， CJ1-H， CJ1M 或 CS1D）



当执行 FALS(007) 且 N 设置了一个与 A529（产生 FAL/FALS 系统号）内容相同的 FAL 号 (1 ～ 511)，将产生一个致命错误，错误代码和错误相信详细代码指定在 S 和 S+1。同时执行下列过程

1,2,3...

1. 指定错误代码将写入 A400。
2. 错误代码和错误发生的时间将写入错误记录区 (A100 ～ A199)。
3. 专用的辅助区标志基于错误代码和错误详情设置。
4. CPU 单元上的 ERR 指示灯将闪烁， PLC 操作将停止。
5. 指定系统错误的致命错误信息将在手持编程器上显示。

注

1. A529（产生的 FAL/FALS 系统号）中的值是系统有意产生一个非致命错误时用的 FAL 的虚号 （FAL。 FALS 和 FPD 共享）。这个号是一个 FAL 虚号，所以它不改变执行 FAL 号标志或错误代码的状态。
   当需要产生两个或更多系统错误（致命和 / 或致命错误）时，用 A529 和 N 中的相同值，但 S 和 S+1 中的不同值，多于一次地执行 FAL/FALS/FPD 指令能产生不同的错误。
2. 在 FALS(007) 指令执行时，同时发生了一个更严重的错误（系统产生致命错误或另一个 FALS(007) 错误），更严重的错误代码将写入 A400。
3. 为了清除一个 FALS(007) 产生的系统错误，置 PLC 为 OFF，然后再置 ON。PLC 将保持 ON，但仍需要相同的过程以清除错误好象指定错误实际发生似的。详情参考 CS 系列或 CJ 系列操作手册中故障信息。
4. 下表显示了 FALS(007) 产生了一个致命系统错误后，IOM 保持位是如何影响 I/O 内存状态和输出单元的输出状态的。

IOM 保持位（A50012）	I/O 内存状态	输出单元的输出状态
ON	保持	OFF
OFF	清除	OFF

注 与用户定义致命错误不同，如果 IOM 保持位是 OFF， FALS(007) 产生系统错误将清除 I/O 内存。下面区域将被清除：CIO 区，工作区，定时器区和 PV，索引寄存器和数据寄存器。

下表显示了如何在 S 和 S+1 中指定错误代码和错误详情。

错误名称	S	S+1
	错误代码	错误详情
内存错误	80F1 Hex	• 位 00 ～ 09：内存错误位置 位 00：用户程序 位 04：PLC 设置 位 05：I/O 注册表 位 07：路由表 位 08：CPU 总线单元设置 位 09：内存卡传输错误 • 位 10 ～ 15：无效
I/O 总线错误	80C0 Hex	• 位 00 ～ 07：产生 I/O 总线错误的插槽号 插槽 0 ～ 9 ：00 ～ 09 十六进制 未知插槽：0F 十六进制 • 位 08 ～ 15：产生 I/O 总线错误的机架号 插槽 0 ～ 7：00 ～ 07 十六进制 未知插槽：0F 十六进制
单元号重叠错误	80E9 Hex	CPU 总线单元的重叠单元号 0000 ～ 000F Hex
		特定 I/O 单元的重叠单元号 8000 ～ 805F Hex
机架号重叠错误	80EA Hex	重叠机架号 （重叠字分配） 0000 ～ 0006 Hex
内部板致命错误	82F0 Hex	错误原因 位 00 ～ 03：由内板定义的错误 位 04 ～ 15：无效

错误名称	S	S+1
	错误代码	错误详情
过多 I/O 点错误	80E1 Hex	位 13 ～ 15：错误原因 位 00 ～ 12：详情 • I/O 点的总数太大 位 13 ～ 15: 000 位 00 ～ 12：I/O 点数 （二进制） • 中断输入数太大 位 13 ～ 15: 001 位 00 ～ 12：中断输入数 （二进制） 位 00 ～ 12：全为 0 • 从属单元的单元号重叠或一个 C500 从属单元有多于 320 个 I/O 点。 位 13 ～ 15: 010 位 00 ～ 12：从属单元的单元号 （二进制） • 一个 I/O 接口的单元号重叠 位 13 ～ 15: 011 位 00 ～ 12：单元号 （二进制） • 主单元的单元号重叠或在允许的设置范围外 位 13 ～ 15: 100 位 00 ～ 12：主单元的单元号 （二进制） • 扩展机架数太大 位 13 ～ 15: 101 位 00 ～ 12：扩展机架数 （二进制） • C200H 指定 I/O 或远程 I/O 不被识别 位 13 ～ 15: 110
I/O 表设置错误	80E0 Hex	--- （不定）
编程错误	80F0 Hex	• 位 08 ～ 15：错误原因 位 15:UM 溢出错误 位 14: 非法指令错误 位 13: 微分溢出错误 位 12: 任务错误 位 11: 无结束错误 位 10: 非法访问错误 位 09: 间接 DM/EM BCD 错误 位 08: 指令错误 • 位 00 ～ 07: 无效
循环时间超时错误	809F Hex	--- （不定）

清除 FALS(007) 致命系统错误（仅 CS1-H，CJ1-H，CJ1M 和 CS1D CPU 单元）

有两种清除由 FALS(007) 产生的致命系统错误的方法。

1. 变 PLC 为 OFF，再变为 ON。
2. 当保持 PLC 为 ON 时，必须清除系统错误好象指定错误实际发生了似的。

清除 FALS(007) 用户定义的致命错误

为了清除 FALS(007) 产生的错误，首先消除错误的原因，然后通过编程设备清除错误或变 PLC 为 OFF，再变为 ON。

标志

名称	标记	操作
错误标志	ER	如果 N 不在指定的范围 0001 ～ 01FF （0 ～ 511 十进制）时为 ON。 如果一个致命系统错误产生 （仅 CS1/CJ1-H/CJ1M/CS1D），而指定错误代码或错误详情不正确时为 ON。 其它情况时 OFF。

下面这些表显示了在辅助区的相关字和标志。

- 仅用户定义的错误辅助区字 / 标志

名称	地址	操作
FALS 错误标志	A40106	当由 FALS(007) 产生错误时为 ON。

- 仅系统错误的辅助区字 / 标志（仅 CS1-H，CJ1-H，CJ1M 和 CS1D CPU 单元）

名称	地址	操作
系统产生的 FAL/FALS 号	A529	当一个系统错误由 FALS(007) 产生时，用 FAL/FALS 的虚号。在字 （0001 ～ 01FF 十六进制， 1 ～ 511 十进制）中设这个 FAL/FALS 的虚号。

- 用户定义和系统错误的辅助区字 / 标志

名称	地址	操作
错误记录区	A100 ～ A199	错误记录区包含最近 20 个错误的错误代码和时间 / 日期，包括由 FALS(007) 产生的错误。
错误代码	A400	当一个错误产生时，相应标志变为它的错误代码存储在 A400 中。 FAL 号 0001 ～ 01FF 的错误代码分别是 4101 ～ 42FF。 如果同时产生两个或两个以上的错误，最严重错误的错误代码将存储在 A400。

注意

信息的结束代码是空字符 （00 十六进制）。如果空字符被省略，将显示在字 S ～ S+7 中的 16 个字符。

N 必须在 0001 ～ 01FF 之间，如果 N 在这个范围之外，将发生错误，错误标志将变 ON。

例

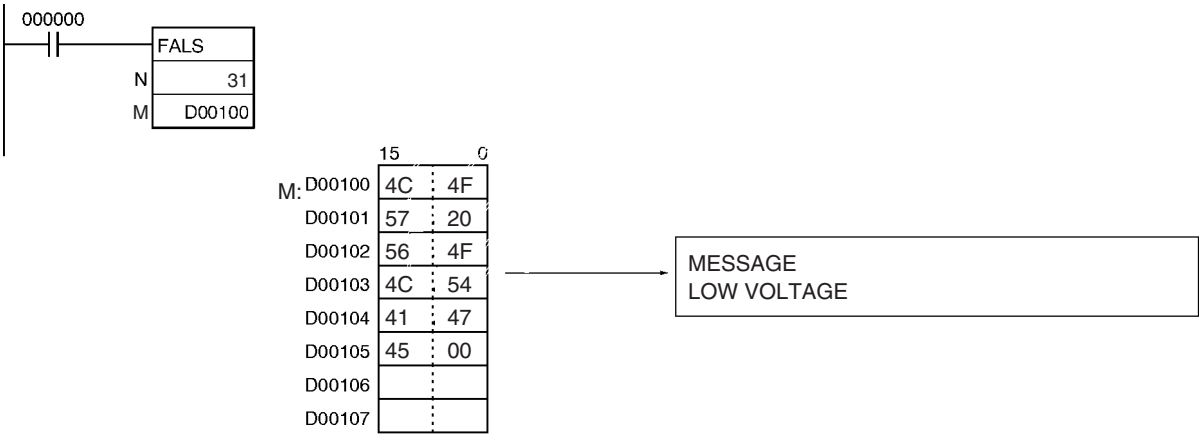
产生一个用户定义的错误

下例中当 CIO 000000 是 ON 时， FALS(007) 将产生一个 FAL 号为 31 的致命错误， 并将执行下列过程。

1,2,3...

1. FAL 错误标志 (A40106) 将变 ON。
2. 对应的错误代码 (C11F) 将写入 A400 。

   注 A400 中将包含发生的所有错误中最严重的错误的错误代码，包括非致命的和致命 的系统错误，以及 FAL(006) 和 FALS(007) 所产生的错误。
3. 错误发生的时间 / 代码和错误代码将写入错误记录区 (A100 ～ A199)。
4. CPU 单元上的 ERR 指示灯将闪烁。
5. D00100 ～ D00107 中的 ASCII 信息将显示在外围设备上。（如果不需要显示信息，在 S 中设置一个常数）。

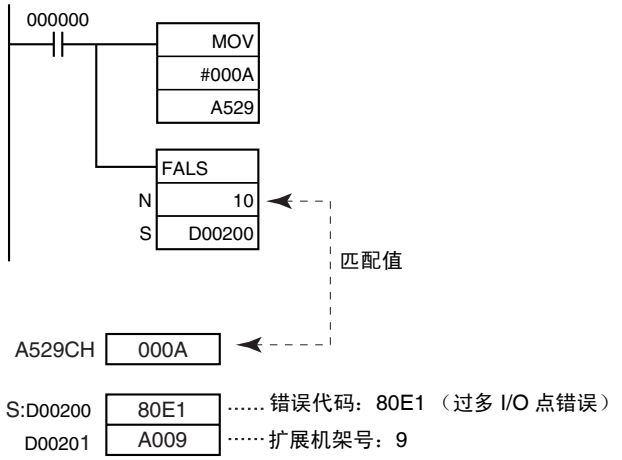


产生一个非致命系统错误 （仅 CS1-H， CJ1-H， CJ1M 和 CS1D CPU 单元）

下例中当 CIO 000000 是 ON 时， FALS (007) 将产生一个过多 I/O 点错误（连接过多扩展基架，本例中是 9 个）。本例中，FAL 虚号是 10，相应的值（000A 十六进制）存储在 A529 中。

1,2,3...

1. 如果这个错误是最严重错误，则指定错误代码 (80E1) 将写入 A400。
2. 错误代码和错误发生的时间 / 日期将写入错误记录区 (A100 ～ A199)。
3. 过多 I/O 点标志 (A40111) 变为 ON。
4. CPU 单元上的 ERR 指示灯将闪烁， PLC 操作停止 。
5. 一个信息 (TOO MANY I/O PNT) 将显示在手持编程器上以指示过多 I/O 点错误已发生。

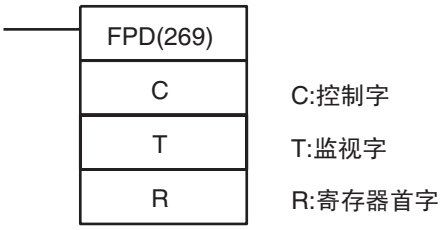


3-30-3 故障点检测：FPD(269)

用途

通过监视执行 FPD(269) 和执行诊断输出之间来诊断指令块中的故障，并查找哪一个输入阻止输出变 ON。

梯形图符号



变化

变化	ON 条件时每次循环执行	FPD(269)
	上升沿微分时执行一次	不支持
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

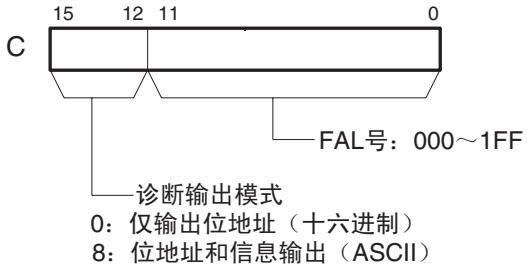
块程序区	步程序区	子程序	中断任务
不允许	OK	OK	不允许

操作数

C: 控制字

C 必须是 0000 ～ 01FF 之间或 8000 ～ 81FF 之间的常数。

下图显示控制字中各位的功能。



T: 监视时间

T 必须在 0000 ～ 270F（十进制 0 ～ 9999）之间。T 值为 0，禁止时间监视；1 ～ 270F 对应监视时间 0.1 ～ 999.9 秒。

R: 寄存器首字

寄存器字的功能见 953 页。

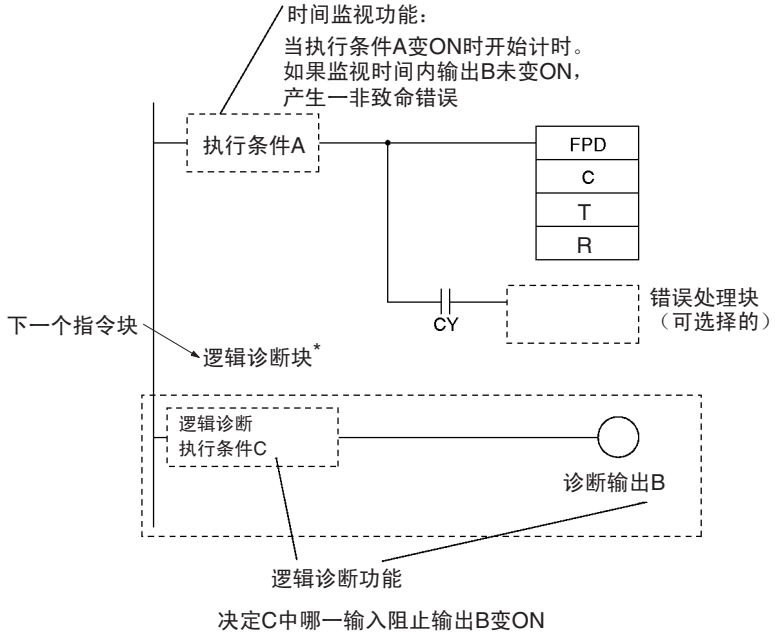
操作数规定

区域	C	T	R
CIO 区	---	CIO 0000 ～ CIO 6143	
工作区	---	W000 ～ W511	
保持位区	---	H000 ～ H511	
辅助位区	---	A000 ～ A447 A448 ～ A959	A448 ～ A959
定时器区	---	T0000 ～ T4095	
计数器区	---	C0000 ～ C4095	
DM 区	---	D00000 ～ D32767	
无区号 EM 区	---	E00000 ～ E32767	

区域	C	T	R
有区号 EM 区	---	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 地址	---	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	---	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	仅指定值	#0000 ～ #270F （二进制）	---
数据寄存器	---		
索引寄存器	---		
使用索引寄存器 间接寻址	---	,IR0 ～ ,IR15 –2048～+2047 ,IR0～–2048～+2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15	

描述

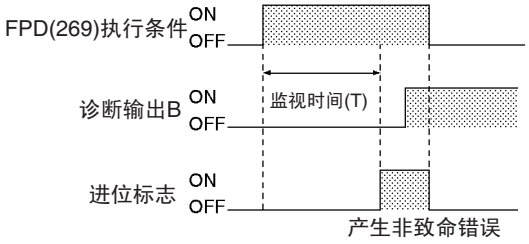
FPD(269) 进行时间监视和逻辑诊断。如果在指定的监视时间内诊断数据不变 ON，时间监视功能产生指定 FAL 号的错误，逻辑诊断功能指出哪一个输入阻止输出变 ON。



注 * 逻辑诊断块从 FPD 之后的第一个 LD （不是 LD TR）或 LD NOT 指令开始，在第一个 OUT （非 OUT TR）或其它右侧指令处结束。

时间监视功能

FPD(269) 从开始执行即记时 （当执行条件 A 变 ON）；如果在指定的监视时间内诊断输出没有变 ON，将产生非致命错误，并使进位标志变 ON。



注 在监视时间内诊断输出必须变 ON。可以使用学习功能自动设置监视时间。

当进位标志变 ON 时，将执行以下过程。（如果在 C 中的 FAL 号设置为 000，不执行该过程）。

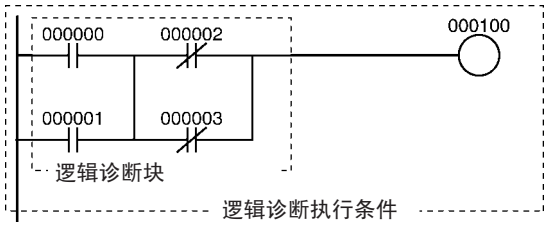
1,2,3...

1. FAL 错误标志 (A40215) 将变为 ON （PLC 继续运行）。
2. 指定 FAL 号的执行 FAL 号标志变 ON（标志 A36001 ～ A39115 对应 FAL 号 001 ～ 1FF）。
3. 相应的错误代码将写入 A400。错误代码 4101 ～ 42FF 对应 FAL 号 001 ～ 1FF。
   （如果同时发生了一个更严重的错误（具有更高的错误代码），更严重错误代码 A400 中）。
4. 错误代码和错误发生的时间 / 日期将写入错误记录区 (A100 ～ A199)。
5. CPU 单元上的 ERR 指示灯将闪烁。
6. 如果输出模式设置成位地址和信息输出（C 中最左位数设置成 8）。R+2 ～ R+9 中存储的 ASCII 信息将作为非致命错误信息显示。

逻辑诊断功能

每次 FPD(269) 的执行条件为 ON 时，FPD(269) 决定哪一个输入位致使诊断输出为 OFF，并将位地址写入到以 R 开始的寄存器区。

下例中如果输入位 CIO 000000 ～ CIO 000003 全为 ON，FPD(269) 将决定常闭条件 CIO 000002 导致输出 CIO 000100 保持位 OFF，FPD(269) 将使位地址找到标志 （R 的位 15）变 ON，并将位地址写入寄存器字 R+2 ～ R+4 中。



只要 FPD(269) 的执行条件为 ON，逻辑诊断功能每个循环都执行。逻辑诊断功能的运行独立于时间监视功能。

当有两个或更多的输入位阻止诊断输出变ON时，执行条件中第一个输入位（最上面的指令行和最接近左边母线）的地址将输出到 R+2 ～ R+4 中。

逻辑诊断功能检查 LD，LD NOT，AND，AND NOT，OR 和 OR NOT 指令的输入位 （包括微分和立即刷新变化），而不检查其它指令的输入位和通过索引寄存器间接寻址的操作数。

逻辑诊断块以 FPD(269) 之后的第一个 LD（非 LD TR）或 LD NOT 指令开始，在第一个 OUT （非 OUT TR）或其它右边指令处结束。

用 C 的最左位数字可以设置两个诊断输出模式。

1,2,3...

1. 位地址输出模式 （C 的最左位数字 =0）

   当查找到输入位地址后，R 的 15 位（位地址找到标志）变 ON，R 的位 14 指明输入是常开还是常闭。
   输入位的 8 位十六进制内部 I/O 内存地址输出到 R+3 和 R+2 中。

2. 位地址和信息输出模式 （C 的最左位数字 =8）

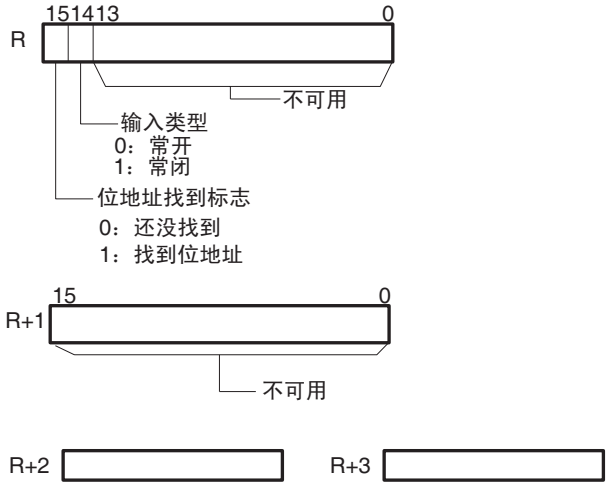
   当查找到输入位地址后，R 的 15 位（位地址找到标志）变 ON，R 的位 14 指明输入是常开还是常闭。
   输入位的地址将以 6 个 ASCII 字输出到 R+2 ～ R+4 中。

寄存器字功能

寄存器字包含诊断功能的结果和由时间监视功能产生错误时显示的 ASCII 错误信息。寄存器字功能依赖于在 C 的最左位数字中所设置的诊断输出模式。

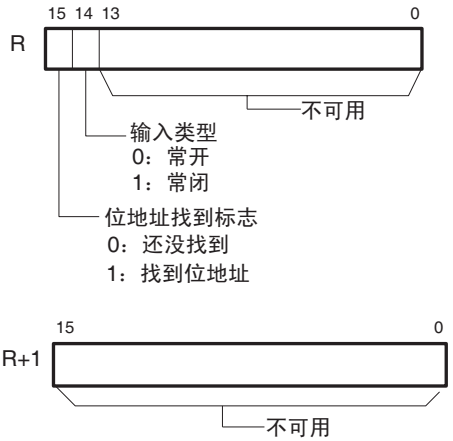
位地址输出 (C=0@@@)

当 C 的最左位数字设置为 0 时，输入位的 8 位十六进制内部 I/O 内存地址输出到 R+3 和 R+2 中。 R 包含 2 个标志，用来指明输出是否已找到和它是用常开条件还是常闭条件。



位地址和信息输出 (C=8@@@)

当 C 的最左位数字设置为 8 时，输入位的 ASCII 码地址输出到 R+2~R+4 中。R 包含 2 个标志，用来指名输出是否已找到和它是用常开还是常闭输入条件。



寄存器字 R+2 ～ R+4 指明阻止诊断输出变 ON 的输入地址。位地址以 ASCII 形式输出到这些字中。下表显示每区的 ASCII 表示。

区域	ASCII 文本	注释
辅助区	A00000 ～ A95915	---
保持区	H00000 ～ H51115	---
工作区	W00000 ～ W51115	---
CIO 区	000000 ～ 665515	---
任务标志	TK0000 ～ TK0031	---
定时器区	_T0000 ～ _T4095	“_”代表一个 ASCII 空格 （字符代码 20）
计数器区	_C0000 ～ _C4095	

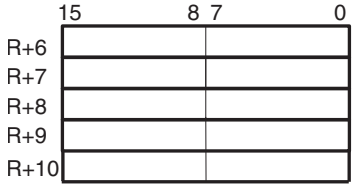
	15	
R+2	W	5
R+3	1	1
R+4	1	5

ASCII形式的位地址

对于 W51115，寄存器字 R+2 ～ R+5 中有以下值：

字	位 8 ～ 15	位 0 ～ 7
R+2	W	5
R+3	1	1
R+4	1	5
R+5	2D （十六进制）	输入类型 （十六进制） 30: 常开 31: 常闭

用户能够把 ASCII 信息存储到寄存器字 R+6 ～ R+9 中。如果由时间监视功能产生了一个非致命错误，该信息可显示在编程设备上。用空字符 （十六进制 00）标记信息的结束。



禁止用户定义错误的错误记录登录 （仅 CS1-H， CJ1-H， CJ1M 或 CS1D）

一般当 FPD(269) 产生一个用户定义的错误时，错误产生的错误代码和时间 被写入到错误记录区 (A100 ～ A199)。对于 CS1-H，CJ1-H，CJ1M 和 CS1D CPU 单元，可以设置 PLC 设置 , 从而可使 FPD(269) 产生的用户定义错误不记录到错误记录中。

即使错误不记录到错误记录中，FAL 错误标志 (40215) 将变 ON，在执行的 FAL 号标志 (A36001 ～ A39115) 的相应标志将变 ON，错误代码将写到 A400。

当你仅想记录系统产生的错误时，禁止用户定义的 FPD(269) 错误的错误记录登录。例如，在调试时，如果 FPD(269) 和 FAL(006) 指令用于几个场合，并且错误记录充满了这些错误，此刻，这个功能是很有用的。下表显示了 PLC 设定的设置：

项目	设置	
手持编程器设置地址	字	129
	位	15
名称	FAL 错误记录注册	
设置	0: 在错误记录中记录 FAL 错误 1: 不在错误记录中记录 FAL 错误	
缺省设置	0: 在错误记录中记录 FAL 错误	
读 PLC 设置的时间	每次循环 （当一个 FAL 错误产生时）	

只要 PLC 设置字 129 位 15 设为 1（不在错误记录中记录 FAL 错误），将记录下面的错误

- 由 FAL(007) 产生的致命错误
- 系统产生的非致命错误
- 系统产生的致命错误
- 由 FAL(006) 或 FPD(269) 有意产生的系统非致命错误
- 由 FAL(007) 有意产生的致命系统错误

用学习功能设置监视时间

如果 T 中指定了一个字地址，监视时间能够用学习功能自动设置。如果 T 中已设置了一个字地址，使用以下步骤。

1,2,3...

1. FPD 学习位 (A59800) 变 ON。
2. FPD(269) 将测量从 FPD(269) 的执行条件变 ON 到诊断输出变 ON 的时间。
3. 如果测量时间超出监视时间设置， T 中将存入测量时间的 1.5 倍。

标志

名称	标记	操作
错误标志	ER	如果 C 不在指定的范围 0000 ～ 01FF 或 8000 ～ 81FF 内时为 ON。 如果 T 不在指定的范围 0000 ～ 270F 内时为 ON。 其它情况时 OFF。
进位标志	CY	如果监视时间过去而诊断输出仍为 OFF 时为 ON。 其它情况时 OFF。

下表显示了在辅助区的相关位和标志。

名称	地址	操作
FAL 错误标志	A40215	在时间监视中登记非致命 (FAL) 错误时为 ON。
执行的 FAL 号标志	A36001 ～ A39115	当非致命 (FAL) 错误在时间监视中登记时，相关标志变 ON。标志 A36001 ～ A39115 对应 FAL 号 0001 ～ 01FF。
错误记录区	A100 ～ A199	错误记录区包含最近 20 个错误的错误代码和时间 / 日期，包括由 FPD(269) 产生的错误。
错误代码	A400	当错误产生时，相应标志变为它的错误代码存储在 A400 中。FAL 号 0001 ～ 01FF 的错误代码分别对应错误代码 4101 ～ 42FF。 如果同时产生两个或两个以上的错误，最严重错误的错误代码将存储在 A400。
FPD 学习位	A59800	如果在 FPD(269) 执行时，想要监视时间学习功能，使该位变 ON。

注意

当使用时间监视功能时， FPD(269) 的执行条件必须在 T 中所设置的整个监视时间内都为 ON。

FPD(269) 的执行条件必须由常开和常闭输入所组成。

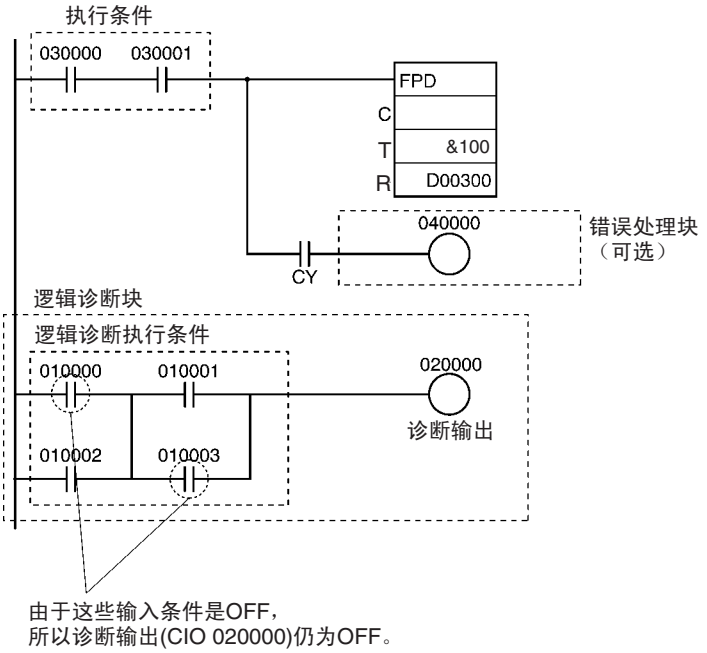
错误处理块可选。如果选择了错误处理块，必须使用输出或其它右侧指令。该点不能使用 LD 和 LD NOT。

FPD(269) 可以在程序中使用多次，但每一个指令必须有一个唯一的寄存器 (R) 设置。

仅当 FPD(269) 执行时，才刷新监视时间。如果循环时间长于 100ms，一般监视时间将不刷新， FPD(269) 将不会正确工作，因为监视时间是以 100ms 为单位刷新的。

例

下例用于演示时间监视功能和逻辑诊断功能的运行。该例中诊断输出 (CIO 020000) 不变 ON，因为在逻辑诊断执行条件中 CIO 010000 和 CIO 010003 保持为 OFF。



时间监视功能

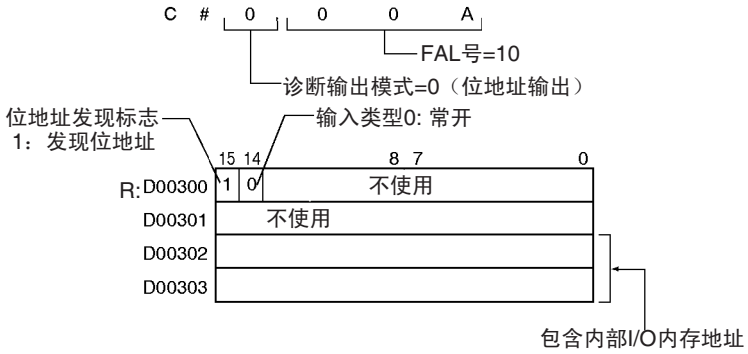
如果CIO 030000和CIO 030001都变ON之后的10秒内诊断输出(CIO 020000)不变 ON，将产生一个非致命错误，并执行以下过程。

1,2,3...

1. 进位标志标志变 ON。
2. 当 C 的最右 3 个数字指定 FAL 号十六进制 000A(10)，则相应的已执行的 FAL 号标志 (A36010) 变 ON，相应的错误代码 (410A) 写入 A400，且 FAL 错误标志 (A40215) 变 ON。

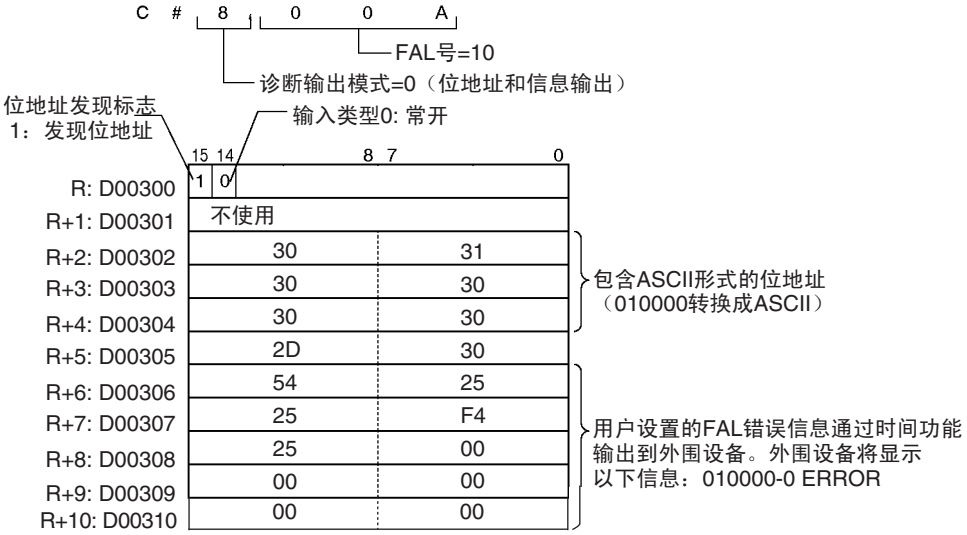
逻辑诊断功能 (C=000A)

由于 C 的最左数字为 0（位地址输出模式），CIO 010000 的内部 I/O 内存地址输出到 D00303 和 D00302 中 （CIO 010000 的指令行比 CIO 010003 的指令行高）。



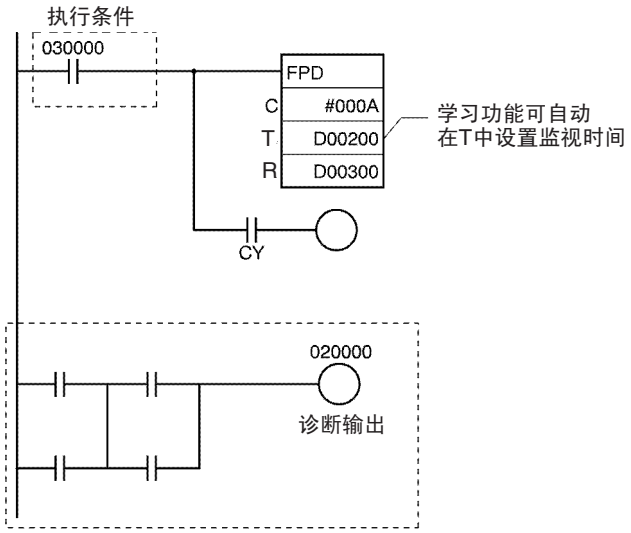
逻辑诊断功能 (C=800A)

由于 C 的最左位数字为 8 （位地址和信息输出模式）， CIO 010000 的地址以 ASCII 形式输出到 D00302 和 D00304 中。

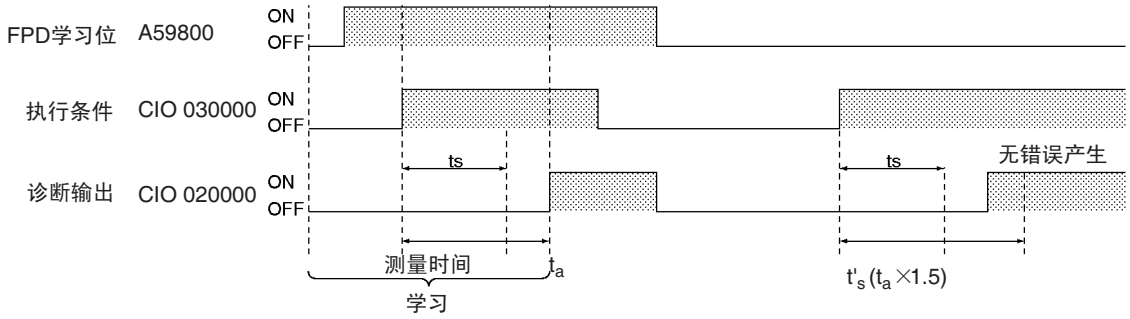


用学习功能设置监视时间

当 T 中指定了一个字地址时，能够使用学习功能自动设置监视时间。



启动学习功能时，将 A59800 （FPD 学习位）置 ON。 A59800 为 ON 后，FPD(269)测量执行条件(CIO 030000)变ON后多长时间诊断输出(CIO 020000)变 ON。如果测量时间超出 T 中的监视时间，测量时间乘以 1.5 后存储到 T 中，作为新的监视时间。



t_s:T 中初始设置
t_a: 测量时间
t'_s: 学习后 T 中新的设置
（当 $t_a > t_s$, $t'_s = t_a \times 1.5$）

3-31 其它指令

本节描述操作进位标志、选择 EM 区和扩大最大扫描时间的指令。

指令	助记符	功能代码	页
设置进位	STC	040	959
清除进位	CLC	041	960
选择 EM 区	EMBC	281	961
扩展最大扫描时间	WDT	094	963
存入条件标志	CCS	282	966
载入条件标志	CCL	283	967
从 CV 转变地址	FRMCV	284	968
转变地址到 CV	TOCV	285	973
外设服务禁止	IOSP	287	977
外围服务允许	IORS	288	978

3-31-1 设置进位：STC(040)

设置进位标志 (CY)。

梯形图符号

变化

变化	ON 条件时每次循环执行	STC(040)
	上升沿微分时执行一次	@STC(040)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

描述

当执行条件为 ON，STC(040) 使进位标志 (CY) 变 ON。但该标志仍能由接下来的能影响该标志的其它指令所改变。

标志

名称	标记	操作
错误标志	ER	OFF 或不变 （见注）
等于标志	=	OFF 或不变 （见注）
进位标志	CY	ON
负标志	N	OFF 或不变 （见注）

注 在 CS1 和 CJ1 CPU 单元 , 这些标志都置 OFF。

在 CS1-H,CJ1-H 和 CJ1M CPU 单元 , 这些标志保持不变。

注意 ROL(027)、ROLL(572)、ROR(028)和RORL(5573)在旋转和位操作中使用进位标志。使用这些指令时，先用 STC(040) 和 CLC(041) 设置和清除进位标志。

3-31-2 清除进位：CLC(041)

用途 把进位标志 (CY) 置 OFF。

梯形图符号

变化

变化	ON 条件时每个循环执行	CLC(041)
	上升沿微分时执行一次	@CLC(041)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

描述 当执行条件 ON 时，CLC(040) 使进位标志 (CY) 变 OFF，但接下来执行的能影响该标志的指令仍可使该标志变 ON/OFF。

标志

名称	标记	操作
错误标志	ER	OFF 或不变 （见注）
等于标志	=	OFF 或不变 （见注）
进位标志	CY	OFF
负标志	N	OFF 或不变 （见注）

注 在 CS1 和 CJ1 CPU 单元这些标志都置 OFF。

在 CS1-H， CJ1-H 和 CJ1M CPU 单元这些标志保持不变。

注意 +C(402)、 +CL(403)、 +BC(406) 和 +BCL(407) 在加操作中使用进位标志。为防止其它先前的指令影响到这指令，使用这些指令前要先使用 CLC(041)。
-C(412)、 -CL(413)、 -BC(416) 和 -BCL(417) 在减操作中使用进位标志。为防止其它先前的指令影响到这指令，使用这些指令前要先使用 CLC(041)。
ROL(027)、ROLL(572)、ROR(028) 和 RORL(5573) 在旋转和位操作中使用进位标志。使用这些指令时，先用 STC(040) 和 CLC(041) 设置和清除进位标志。

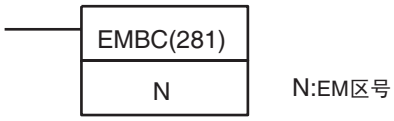
注 +(400)、+L(401)、+B(404)、+BL(405)、-(410)、-L(411)、-B(414) 和 -BL(415) 在加和减操作中不包括进位标志。执行加和减时按通常情况使用这些指令。

3-31-3 选择 EM 区：EMBC(281)

用途

改变当前的 EM 区。

梯形图符号



变化

变化	ON 条件时每个循环执行	EMBC(281)
	上升沿微分时执行一次	@EMBC(281)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

N：EM 区号
以 16 进制 （0000 ～ 000C）指定新的 EM 区号。

操作数定义

区域	N
CIO 区	CIO 0000 ～ CIO 6143
工作区	W000 ～ W511
保持位区	H000 ～ H511
辅助位区	A000 ～ A959
定时器区	T0000 ～ T4095
计数器区	C0000 ～ C4095
DM 区	D00000 ～ D32767
无区号 EM 区	E00000 ～ E32767
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767
常数	#0000 ～ #000C （二进制）
数据寄存器	DR0 ～ DR15
索引寄存器	---
使用索引寄存器的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

EMBC(281) 以指定的 EM 区号 (N) 去改变当前的 EM 区（扩展数据内存）。同时新的 EM 区号输出到 A301。

EM 区可用到 13 个区 (0 ～ C)，每个区有 32768 字 （E0000 ～ E32767）。

以下两种方式可以确定 EM 地址。如果使用第一种，必须使用 EMBC(281) 改变当前 EM 区。

1,2,3...

1. 指定的 EM 区可以无区号，即 E00000 ～ E32767，指示当前 EM 区的地址。
2. 指定的 EM 区可以带区号，即 E_00000 ～ E_32767（n=0 ～ C），指示特定的 EM 区地址。

标志

名称	标记	操作
错误标志	ER	如果 N 不在 0000 ～ 000C 范围内时置 ON。 如果 N 指定一个不存在的 EM 区号时置 ON。 （如果指定的 EM 区已经在 PC 设置中注册为文件内存时也产生此错误）。 其它情况置 OFF。

下表显示辅助区的相关标志

名称	地址	操作
当前 EM 区	A301	包含 16 进制的当前 EM 区号 （0000 ～ 000C）。

注意

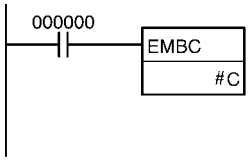
当操作在任务之间转换时，当前 EM 区号仍保持。例如，如果任务 1 中使用 EMBC(281) 把当前 EM 区从 B 区改为 C 区， C 区将保持为当前 EM 区，即使运行切换到任务 2 中。

一个中断任务中当前 EM 区号的改变仅在它被改变的中断的执行过程中是有效的。 先前 EM 区号将被复原到已完成中断任务的那一次执行。

如果指定的 EM 区已经在 PC 设置中被注册为文件内存，将出现错误。

例

下例中，当 CIO 000000 变 ON，当前 EM 区号改变为 C 区，新的区号（16 进制 000C）输出到 A301。



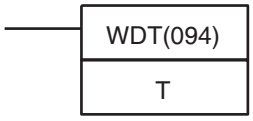
A301 | 0 0 0 C |

3-31-4 扩展最大循环时间：WDT(094)

用途

扩展最大循环时间，但只扩展执行该指令的循环。当特殊处理暂时需要一个较长的扫描周期时， WDT(094) 能防止长扫描周期错误。

梯形图符号



T:定时器设置

变化

变化	ON 条件时每个循环执行	WDT(094)
	上升沿微分时执行一次	@WDT(094)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

T：定时器设置
指定16进制0000和0F9F范围或十进制&0000和&3999范围内的时钟狗定时。

操作数定义

区域	T
CIO 区	---
工作区	---
保持位区	---
辅助位区	---
定时器区	---
计数器区	---
DM 区	---
无区号 EM 区	---
有区号 EM 区	---
二进制间接 DM/EM 地址	---
BCD 间接 DM/EM 地址	---
常数	0000 ～ 0F9F （二进制）
数据寄存器	---
索引寄存器	---
使用索引寄存器的间接寻址	---

描述

WDT(094) 扩展该指令执行周期的最大循环时间。PC 设置中设定的时钟定时器的间隔扩展 T × 10ms （0 ～ 39990ms）。

下表显示 PC 设置中的时钟定时器。最大循环时间的缺省值是 1000ms，它可以从 1 ～ 40000ms 范围内以 10 ms 为单位任意设置。

名称	功能	设置
监视循环时间	如果循环时间超过最大设置，将记录循环时间太长错误 （致命错误）。	0：缺省设置 （1000ms） 1：用户时间设置
	设置最大循环时间。（只有第一个设置已设置为 1 时，该设置才有效）	0001 ～ 0FA0 （1 ～ 40000ms， 10-ms 单位）

标志

名称	标记	操作
错误标志	ER	如果时钟狗定时器设置超过 40 秒置 ON。 其它情况置 OFF。

下表显示辅助区的相关标志和字。

名称	地址	操作
循环时间太长	A40108	当前循环时间超过 PLC 设置中设置的最大循环时间 （监视循环时间）时置 ON。这是致命错误，将导致程序停止执行。
最大循环时间	A262 和 A263	这些字包含以 32 位二进制表示的最大循环时间。每次循环这些值都会刷新。
当前循环时间	A264 和 A265	这些字包含以 32 位二进制表示的当前循环时间。每次循环这些值都会刷新。

注意

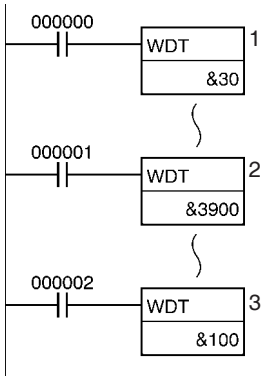
在一个循环中可以多次使用 WDT(094)。这种情况下扩展扫描时间相加起来，但总和不能超过 40000ms。如果循环时间已扩大到 40000ms，WDT(094) 不能再被执行。

例

该例中使用缺省最大扫描时间 (1000ms)。

1,2,3...

1. 当 CIO 000000 变 ON，第一个 WDT(094) 指令把最大循环时间扩大 300ms （30 × 10ms），于是这点的最大循环时间是 1300ms。
2. 当 CIO 000001 变 ON，第二个 WDT(094) 指令试图把最大循环时间再增大 39000ms。新的最大循环时间 （40300ms）超过最大极限 （40000ms），300ms 被忽略。所以第二个 WDT(094) 指令实际把最大循环时间增大 38700ms。
3. 当 CIO 000002 变 ON，第三个 WDT(094) 指令试图把最大循环时间再增大 1000ms，由于最大扫描时间早已到达极限 （40000ms），所以第三个 WDT(094) 指令不被执行。

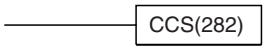


3-31-5 保存条件标志：CCS(282)

在 CPU 单元一个独立区域中保存条件标志的当前状态。标志当前状态被保存，所以它能在程序中不同位置、不同任务、甚至在随后循环中用 CCL(283) 来读 （恢复）。

此指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每个循环执行	CCS(282)
	上升沿微分时执行一次	@CCS(282)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

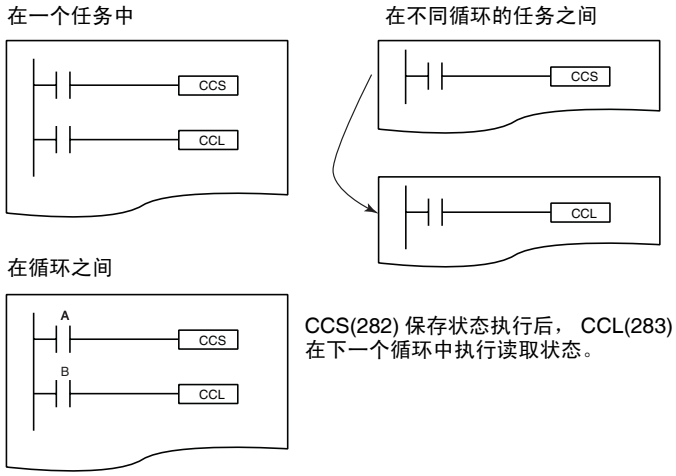
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

描述

当执行条件 ON 时，CCS(282) 在 CPU 单元一个独立区域中存储条件标志（除常 ON 和常 OFF 标志之外）当前状态。下列条件标志的状态将被保留：ER，CY， >， =， <， N， OF， UF， >=， <> 和 <= 。

已保留的条件标志的状态仅能用 CCL(283) 即加载条件标志指令随后读 （恢复）。这些状态可在下列任意情况下读：

- 在一个任务中
- 在不同循环的任务之间
- 在循环之间

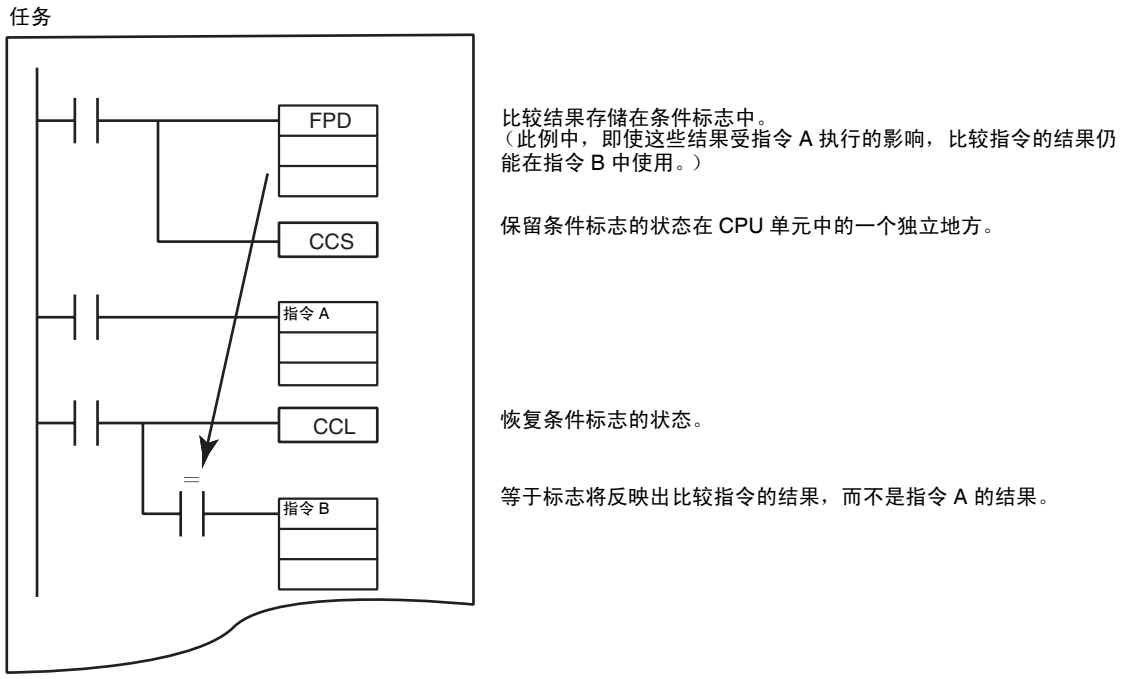


注 1. 条件标志的状态在一个循环任务和中断任务之间不能被保存 / 加载。

2. 当 CCS(282) 执行时，它重写已经保存的先前条件标志信息。

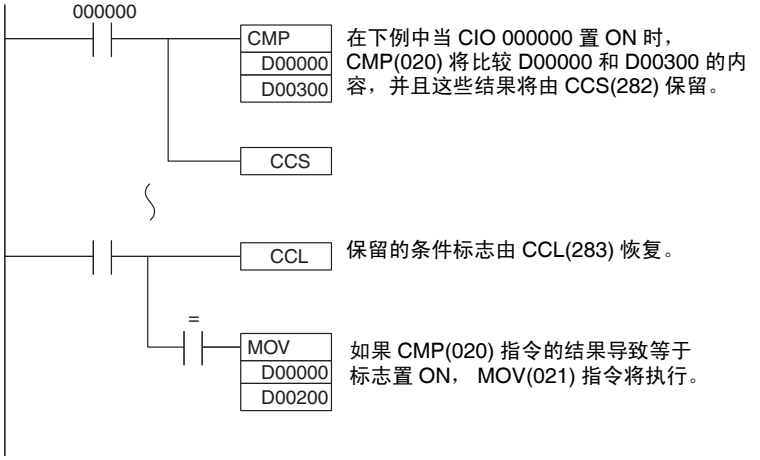
当操作从一个任务切换到另一个任务时，所有的条件标志都被清除。使用 CCS(282) 和 CCL(283) 指令在任务或循环之间保存和加载条件标志状态。

例如，CCS(282) 和 CCL(283) 指令可使从执行 FPD(296) 得到的 CY 标志状态（时间监视诊断错误）用于程序中随后的地方，而不是在此指令执行后立即使用。



标志 没有受这些指令影响的标志。

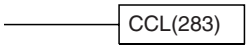
例 下例中，CCS(282)保留一个比较的结果，所以这个结果能在程序中作为一个随后的执行条件使用。



3-31-6 加载条件标志：CCL(283)

恢复在 CPU 单元一个独立区域中由 CCS(282) 保存的条件标志的当前状态。
也可单独使用 CCL(283) 清条件标志。
此指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持。

梯形图符号



变化

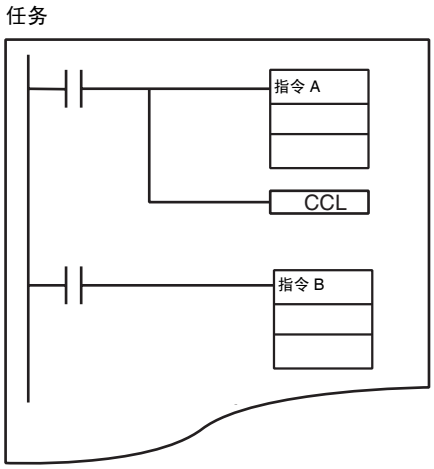
变化	ON 条件时每个循环执行	CCL(283)
	上升沿微分时执行一次	@CCL(283)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

描述

当执行条件 ON 时， CCL(283) 恢复（读）条件标志的状态（除常 ON 常 OFF 标志之外）当前状态。下列条件标志的状态将被恢复（读）：ER，CY，>，=，<， N， OF， UF， >=， <> 和 <= 。
条件标志由所有指令共同分享，所以在 PLC 循环中随着每条指令的执行这些标志的状态可以改变多次。以前，在控制指令后需要立即使用条件标志，这样条件标志的状态不会受插入指令的影响。CCS(282) 和 CCL(283) 指令允许控制指令从依赖于结果的执行条件中分离出来。
例如，比较指令执行后 CCS(282) 能存储等于标志的状态并且结果能随后恢复。指令执行后结果不需要立即使用。



指令 A 执行后单独使用 CCL(283) 清除条件标志，所以这些结果不会影响指令 B 和随后的指令

更多关于如何使用 CCS(282) 和 CCL(283) 的例子请*参阅 3-31-5 保存条件标志 CCS(282)*。

标志

没有标志受这些指令影响。

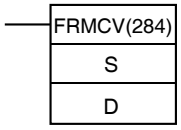
3-31-7 从 CV 转换地址：FRMCV(284)

用途

把一个 CV 系列 PLC 内存地址转换到它相应的 CS/CJ 系列 PLC 内存地址。当转换使用 PLC 内存地址的 CV 系列程序时，可使用 FRMCV(284)，所以它们与 CS/CJ 系列 PLC 兼容。

此指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持。

梯形图符号



S：包含 CV 系列 PLC 内存地址的字
D：目的索引寄存器

变化

变化	ON 条件时每个循环执行	FRMCV(284)
	上升沿微分时执行一次	@FRMCV(284)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

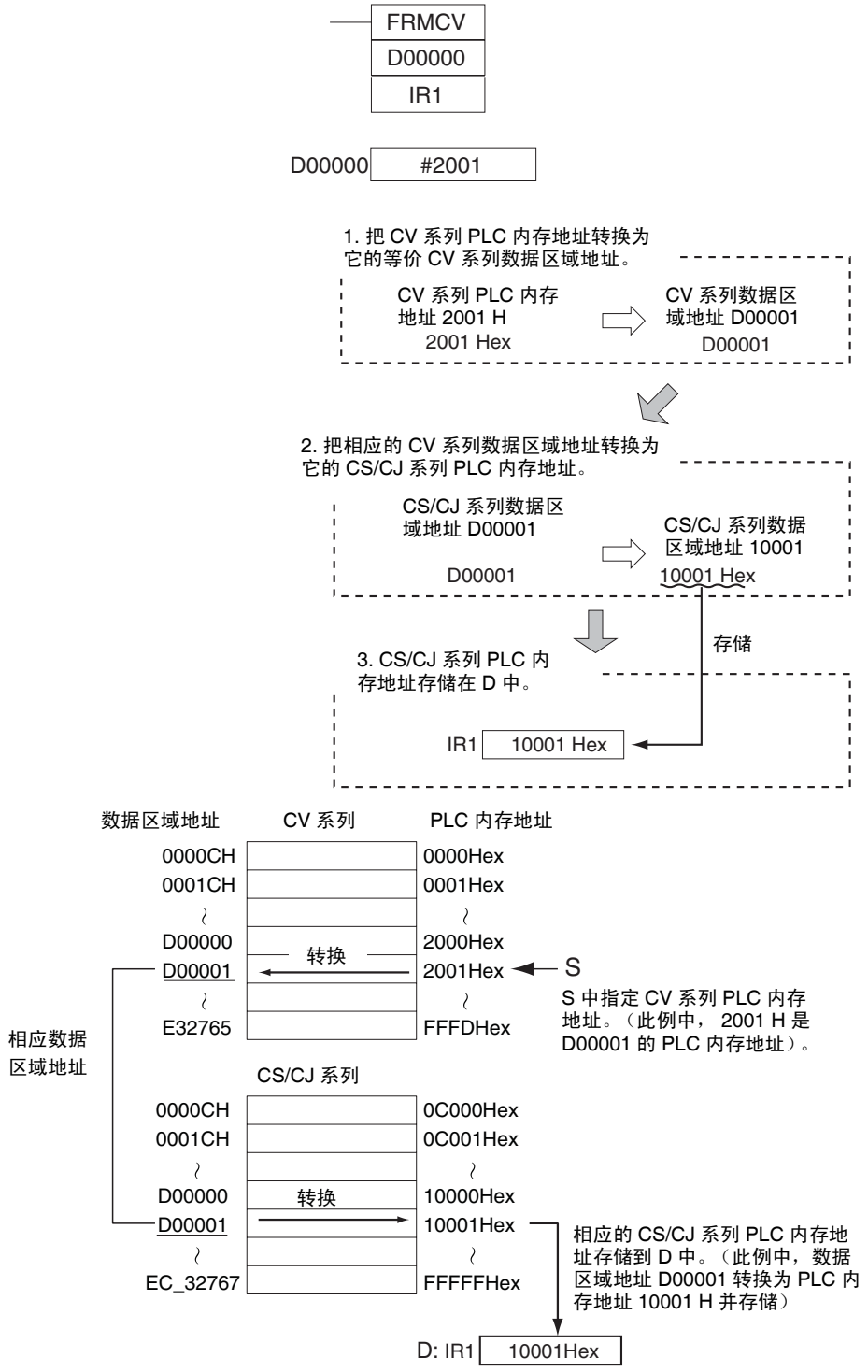
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

描述

当执行条件为 ON 时， FRMCV(284) 执行下列操作。

1. S 中指定的 CV 系列 PLC 内存地址转换为它的等价 CV 系列数据区域地址。
2. FRMCV(284) 决定了对应相同 CV 系列数据区域地址的 CS/CJ 系列 PLC 内存地址。
3. CS/CJ 系列 PLC 内存地址是对 D （一个索引寄存器 （IR0 ～ IR15）必须指定给 D）的输出。

下例显示了 FRMCV(284) 指令用于对 D00001 中 CV 系列 PLC 内存地址转换。



注 如果没有 CS/CJ 系列等价到指定的 CV 系列 PLC 内存地址，将出现错误，错误标志置 ON，并且地址不被转换。

当一个索引寄存器作为一个带一个 “IR” 前缀的操作数使用时，指令将操作由索引寄存器中 PLC 内存地址指示的字，而不是索引寄存器本身。一旦期望的 PLC 内存地址已经存储在一个索引寄存器中，索引寄存器本身可作为对一个指令的操作数来使用。

FRMCV(284) 指令可用来转换带有下列两种在一个 CS/CJ 系列 PLC 中使用设计的一个 CV 系列程序。*见这部分随后的例子。*

1. 当使用间接二进制模式 DM 寻址 (*DM)（当间接的指定 DM 中带有一个 PLC 内存地址的一个数据区域。
2. 当直接使用 CV 系列 PLC 内存地址作为数值 （当把 PLC 内存地址存储在使用一个指令如 MOV(021) 直接寻址的索引寄存器中）。

操作数规定

区域	S	D
CIO 区	CIO 0000 ～ CIO 6143	---
工作区	W000 ～ W511	---
保持位区	H000 ～ H511	---
辅助位区	A448 ～ A959	---
定时器区	T0000 ～ T4095	---
计数器区	C0000 ～ C4095	---
DM 区	D00000 ～ D32767	---
无区号 EM 区	E00000 ～ E32767	---
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	---
二进制间接 DM/EM 地址	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	---
BCD 码间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	---
常数	除 09FF H 外任意常数，0A00 ～ 0AFF Hex, 或 0D00 ～ 0E3F Hex	---
数据寄存器	DR0 ～ DR15	---
索引寄存器	---	IR0 ～ IR15
使用索引寄存器 的间接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15	---

标志

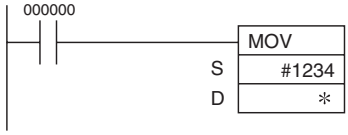
名称	标记	操作
错误标志	ER	如果 S 指定一个下列 CS/CJ 系列中不存在的 PLC 内存地址时置 ON： 暂存继点器 (TR) 区域 (09FF H) CPU 总线链接 (G) 区域 (0A00 ～ 0AFF H) SFC 区域 (0D00 ～ 0E3F H) 其它情况置 OFF。

例

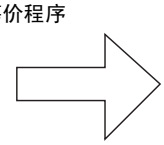
例 1：用 *DM 间接二进制模式 DM 寻址转换一个 CV 系列程序

在这个 FRMCV(284) 例子中，在 S 中指定一个 DM 字， PLC 内存地址存储在一个索引寄存器中，并且索引寄存器用作间接寻址。

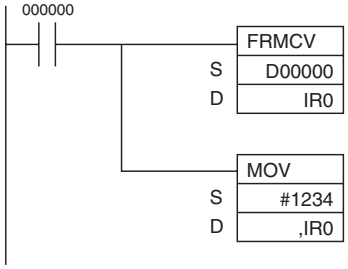
- CV 系列程序
  （程序使用间接DM二进制模式寻址）



等价程序

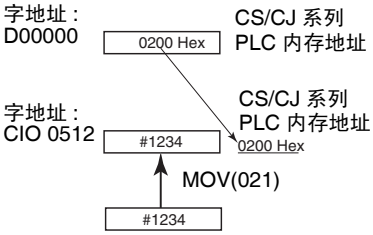


- CS/CJ 系列程序

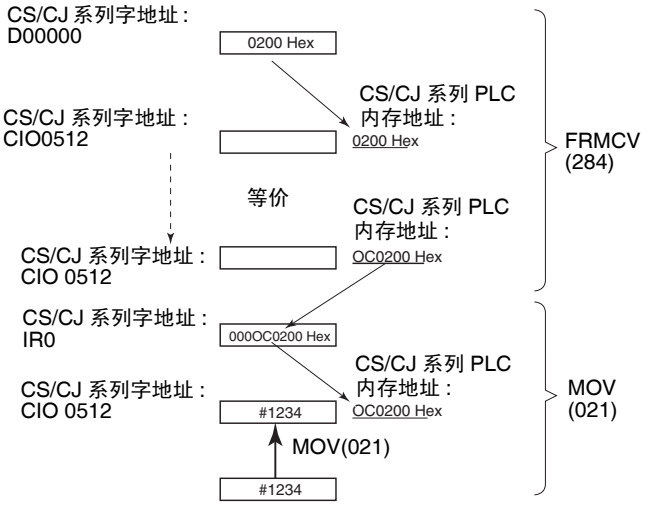


PLC 设置
间接 DM 数据：
当间接 DM 地址是二进制时， DM 字的内容看作是一个 PLC 内存地址并且指定在 I/O 内存中的相应地址。

此例中， D00000 的数值是 0200 H。
相应的数据区域地址是 CIO 0512，
所以 #1234 转换为 CIO 0512。



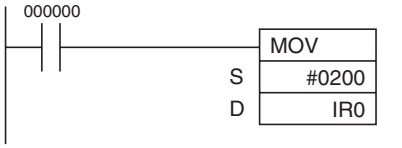
此例中， D00000 的数值是 0200 H。相应的 CV 系列数据区域地址是 CIO 0512。 CIO 0512 的 CS/CJ 系列 PLC 内存地址是 000C200 H，所以这个数值存储在 IR0 中。
MOV(021) 中的目标操作数间接寻址 IR0 的内容，所以 #1234 转换为 CIO 0512。



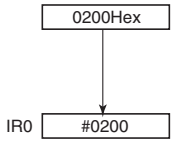
例 2：用直接存储在索引寄存器中的 PLC 内存地址转换一个 CV 系列程序

在这个 FRMCV(284) 例子中，在 S 中直接指定 CV 系列 PLC 内存地址。

- CV 系列程序
  （程序使用 PLC 内存地址直接存储在 IR 中）

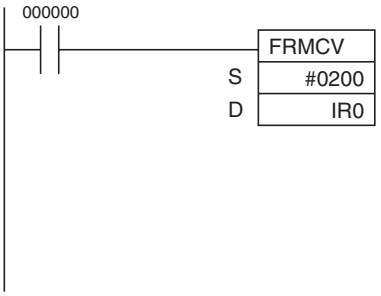


此例中， PLC 内存地址 0200 H 存储在索引寄存器 IR0 中。

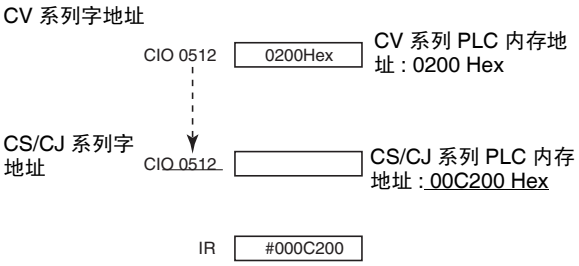


等价程序

- CS/CJ 系列程序



此例中， CV 系列 PLC 内存地址 0200 H。对应 CIO 0512。 CIO 0512 的 CS/CJ 系列 PLC 内存地址是 0000C200 H，所以这个数值存储在 IR0 中。



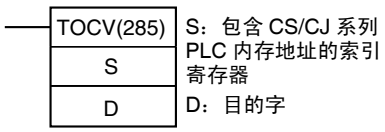
3-31-8 转换地址到 CV：TOCV(285)

用途

把一个 CS/CJ 系列 PLC 内存地址转换到它相应的 CV 系列 PLC 内存地址。当转换使用 PLC 内存地址的 CS/CJ 系列程序时，可使用 TOCV(285)，所以它们与 CV 系列 PLC 兼容。

此指令仅由 CS1-H， CJ1-H， CJ1M，和 CS1D CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每个循环执行	TOCV(285)
	上升沿微分时执行一次	@TOCV(285)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

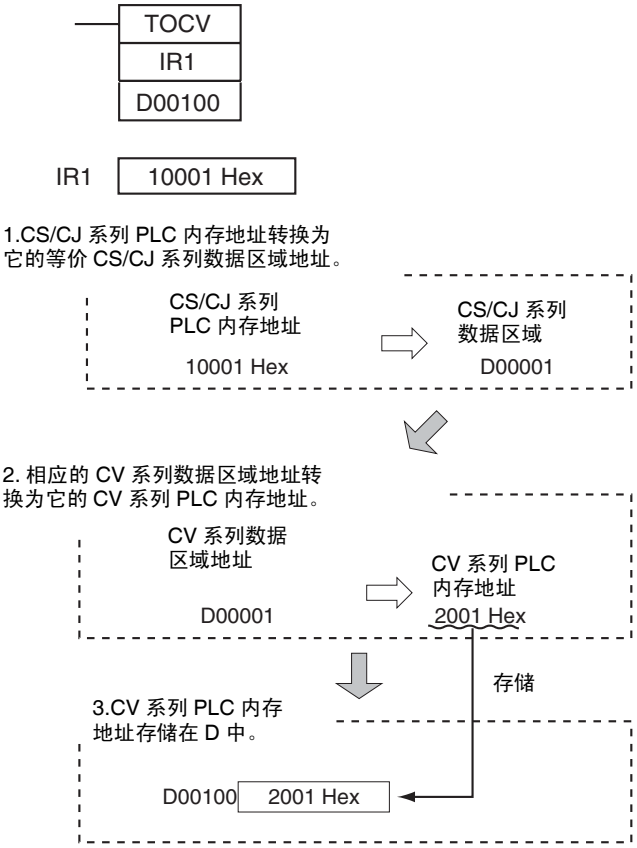
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

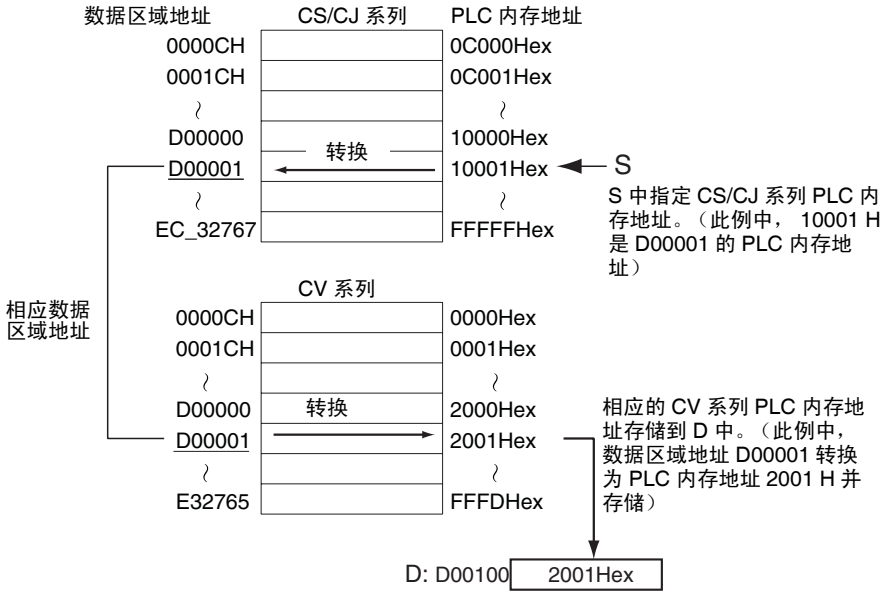
描述

当执行条件为 ON 时， TOCV(285) 执行下列操作。

1. S 中指定的 CS/CJ 系列 PLC 内存地址转换为它的等价 CS/CJ 系列数据区域地址。（一个索引寄存器 (IR0 ～ IR15) 必须指定给 S）
2. TOCV(285) 决定了对应相同 CS/CJ 系列数据区域地址的 CV 系列 PLC 内存地址。
3. CV 系列 PLC 内存地址输出到 D。

下例显示了TOCV(285)指令用于对D00001中CS/CJ系列PLC内存地址转换。





注 1. 如果没有 CV 系列等价到指定的 CS/CJ 系列 PLC 内存地址，将出现错误，错误标志置 ON，并且地址不被转换。

2. 由 TOCV(285) 存储的 CV 系列 PLC 内存地址数据可转换为一个使用 CX 编程器的 CV 系列 PLC。

3. 在 CS/CJ 系列程序中使用的相同数据区域地址通过使用间接索引寄存器寻址（“，IR”前缀）或间接二进制模式DM寻址(*DM)指定在CV系列程序中。

操作数规定

区域	S	D
CIO 区	---	CIO 0000 ～ CIO 6143
工作区	---	W000 ～ W511
保持位区	---	H000 ～ H511
辅助位区	---	A448 ～ A959
定时器区	---	T0000 ～ T4095
计数器区	---	C0000 ～ C4095
DM 区	---	D00000 ～ D32767
无区号 EM 区	---	E00000 ～ E32767
有区号 EM 区	---	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 地址	---	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 码间接 DM/EM 地址	---	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	见注 1	---
数据寄存器	---	DR0 ～ DR15

区域	S	D
索引寄存器	IR0 ～ IR15	---
使用索引寄存器 的间接寻址	---	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15

注 1. 如果S指定下列CV系列中不存在的PLC内存地址，则出现一个错误，错误标志置ON：

区域或地址	PLC 内存地址
任务标志区域	0000 B800 ～ 0000 B801 Hex
A512 ～ A959	0000 BA40 ～ 0000 BBFF Hex
CIO 2556 ～ CIO 6143	0000 C9FC ～ 0000 D7FF Hex
T1024 ～ T4095	0000 BE40 ～ 0000 BEFF Hex 和 0000 E400 ～ 0000 EFFF Hex
C1024 ～ C4095	0000 BF40 ～ 0000 BFFF Hex 和 0000 F400 ～ 0000 FFFF Hex
HR 区域	0000 D800 ～ 0000 D9FF Hex
WR 区域	0000 DE00 ～ 0000 DFFF Hex
D24576 ～ D32767	0001 6000 ～ 0001 7FFF Hex
EM 区规定	0001 8000 ～ 000F 7FFF Hex
E32766 ～ D32767	000F FFFE ～ 000F FFFF Hex

2. 如果 S 指定一个是索引寄存器区域的区域，则出现一个错误，错误标志置ON。

标志

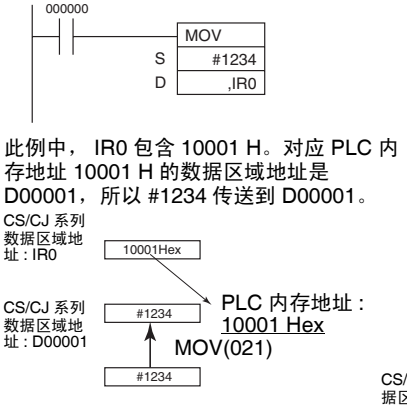
名称	标记	操作
错误标志	ER	如果 S 指定一个 CV 系列 PLC 中不存在的 PLC 内存地址时置 ON： 如果 S 不是一个常数或索引寄存器时置 ON。 其它情况置 OFF。

例

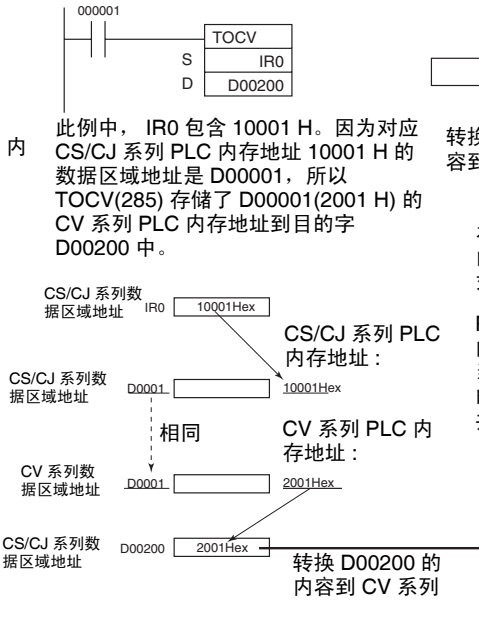
用间接索引寄存器寻址转换一个 CS/CJ 系列程序

1. 在这个 TOCV(285) 例子中，在 S 中指定一个索引寄存器。在这个索引寄存器中的 CS/CJ 系列 PLC 内存地址转换为它的等价 CV 系列。
2. CV 系列 PLC 内存地址被转换到指定的数据区域地址。
3. 使用 CV 系列程序中的 CV 系列 PLC 内存地址。

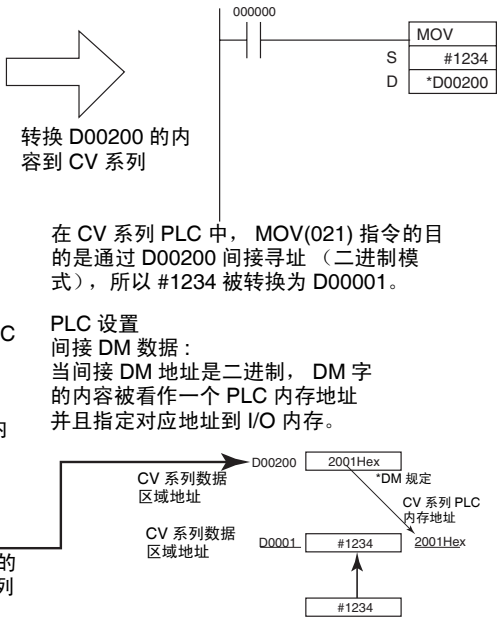
- CS/CJ 系列程序（程序使用间接索引寄存器寻址）



- CS/CJ 系列程序



- CV 系列程序



3-31-9 禁止外围设备服务：IOSP(287)（仅对 CS1-H/CJ1-H/CJ1M）

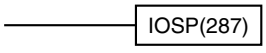
用途

在并行处理模式或外围设备优先模式的程序执行中禁止外围设备服务。

关于并行处理模式或外围设备优先模式的详细情况参阅*第6节CS/CJ PLC程序设计手册中的高级功能*。

注 此指令仅由 CS1-H，CJ1-H，和 CJ1M CPU 单元支持。不能用于 CS1，CJ1，或 CS1D CPU 单元。

梯形图符号



变化

变化	ON 条件时每个循环执行	IOSP(287)
	上升沿微分时执行一次	@IOSP(287)
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

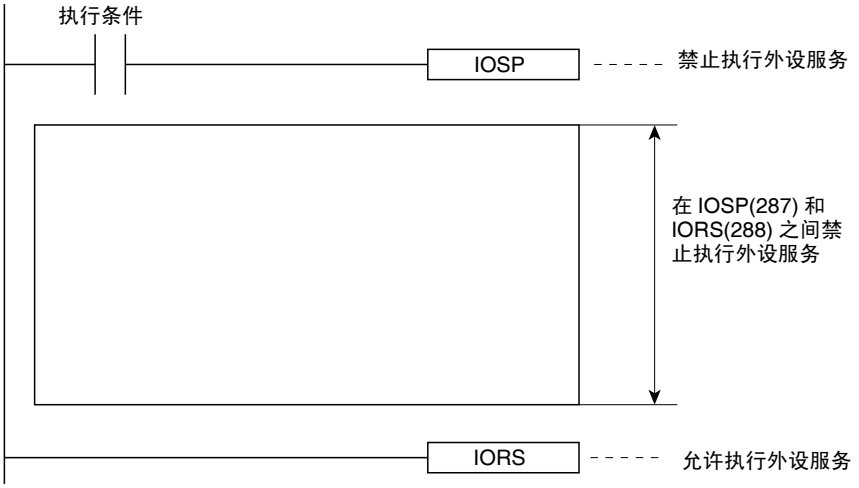
块程序区	步程序区	子程序	中断任务
OK	OK	OK	不允许

描述

在并行处理模式（同步或异步内存访问）的一个循环任务中使用 IOSP(287) 来禁止下列各种外围设备服务。当 IORS(288) 即启动外围设备服务指令被执行时，才允许外围设备服务。

- 用特殊 I/O 单元进行事件服务
- 用 CPU 总线单元进行事件服务
- 外设端口服务

- RS-232C 端口事件
- 用内部底板 （仅对 CS 系列）进行事件服务
- 用一个通讯端口号，即一个内部逻辑端口进行事件服务 （包括后台指令处理）



当 IOSP(287) 禁止外围设备服务时，它将保持禁止直到执行 IORS(288)，执行 END(001)，或停止 PLC 操作。

标志

名称	标记	操作
错误标志	ER	如果在一个中断任务中执行 IOSP(287) 则置 ON。 其它情况置 OFF。

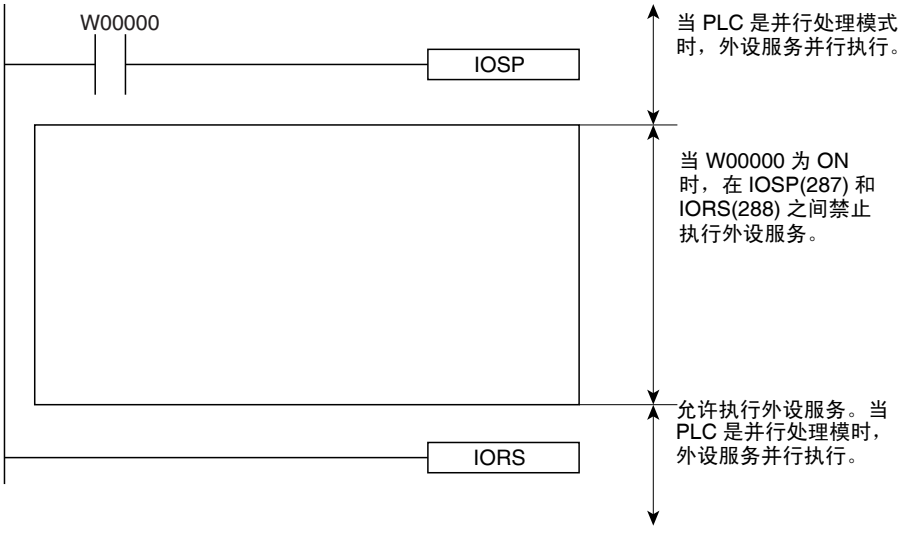
注意

IOSP(287) 在一个中断任务中不能执行。如果在一个中断任务中执行 IOSP(287)，则将出现一个错误，并且错误标志将置 ON。

IOSP(287) 在多个任务中不能禁止外设服务。如果要在多个任务中需要禁止外设服务，则在每个任务中分别用 IOSP(287) 指令编程。

例

下例显示 IOSP(287) 和 IORS(288) 在一个程序段中用来禁止外设服务。



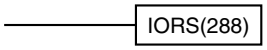
3-31-10 允许外设服务：IORS(288) （仅对 CS1-H/CJ1-H/CJ1M）

用途

在被 IOSP(287) 即禁止外设服务指令禁止的并行处理模式程序执行中，允许外设服务。

此指令仅由 CS1-H， CJ1-H，和 CJ1M CPU 单元支持。

梯形图符号



变化

变化	ON 条件时每个循环执行	IORS(288)
	上升沿微分时执行一次	不支持
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	不允许

描述

在一个循环任务中使用 IORS(288) 来解除由 IOSP(287) 即禁止外设服务指令对外设服务的禁止。

对 IORS(288) 编程不需要使用一个执行条件。

IORS(288) 不能在一个中断任务中执行。如果 IORS(288) 在一个中断任务中执行，则出现一个错误并且错误标志将置 ON。

标志

名称	标记	操作
错误标志	ER	如果在一个中断任务中执行 IORS(288) 则置 ON。 其它情况置 OFF。

3-32 块程序指令

本节描述块程序和块编程指令。

指令	助记符	功能代码	页
块程序开始	BPRG	096	984
块程序结束	BEND	801	984
块程序暂停	BPPS	811	985
块程序重新启动	BPRS	812	985
条件块退出 （非）	EXIT (NOT)	806	991
如果 （非）	IF (NOT)	802	989
否则	ELSE	803	989
如果结束	IEND	804	989
一个循环和等待 （非）	WAIT (NOT)	805	994
定时器等待	TIMW (BCD)	813	998
	TIMWX （二进制）	816	
计数器等待	CNTW (BCD)	814	1001
	CNTWX （二进制）	818	

指令	助记符	功能代码	页
高速计数器等待	TMHW (BCD)	817	1004
	TMHWX （二进制）	815	
循环	LOOP	809	1007
循环结束 （非）	LEND (NOT)	810	1007

3-32-1 介绍

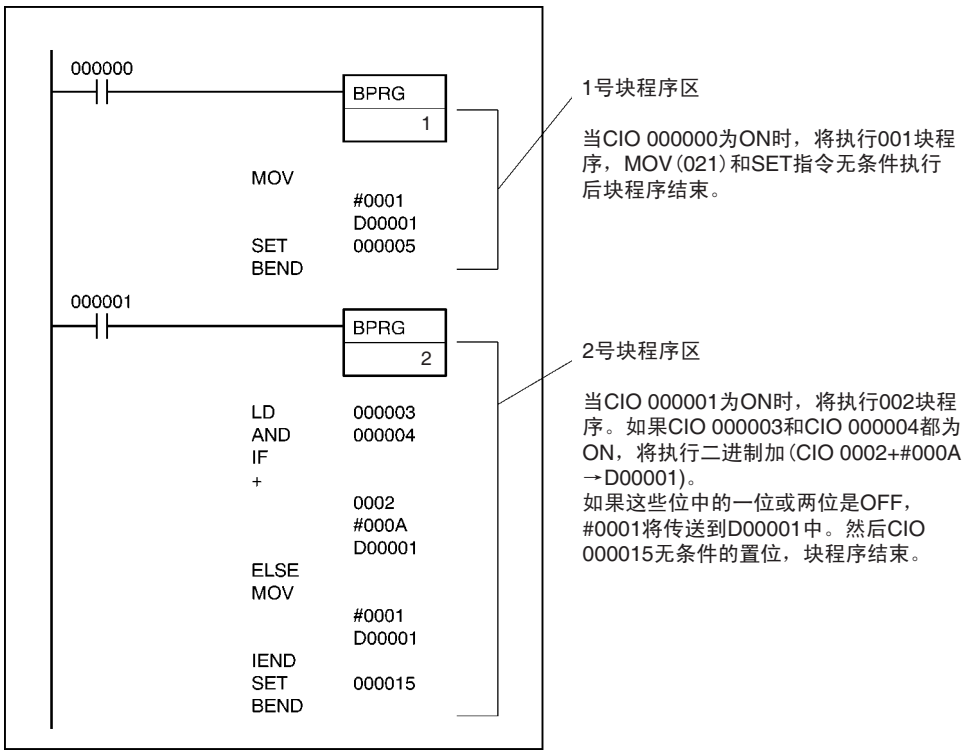
块程序

CS1 的用户程序中（所有任务）最多可使用 128 个块程序。每一个块程序都是由一个执行条件所控制。当 BPRG(096) 的执行条件变 ON 时， BPRG(096) 和 BEND(801) 之间的所有指令都将无条件执行。除了 BPRG(096) 以外，所有的块程序指令都不受执行条件影响。这将允许把所有由一个执行条件控制的程序组织到一个块程序中。

每一个块都是由梯形图的一个执行条件所启动，块中的所有指令都是由助记符形 式表示。块程序是梯形图和助记符指令的组合。

使用梯形图编程比较困难时，可以使用块程序来编程，例如条件分支和步过程。

下面显示两个块程序。



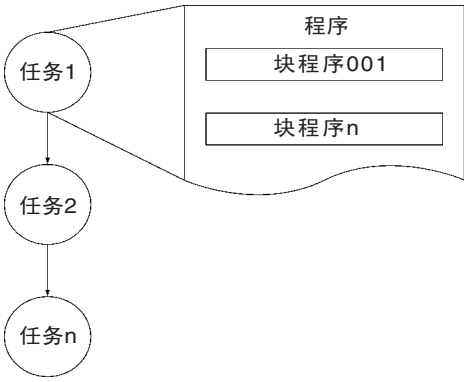
任务和块程序

块程序可位于任务中。当用任务来分隔大的程序单元时，块程序可用于进一步把程序划分为更小的单元，该单元是用一个单一的梯形图程序执行条件来控制的。

同任务一样，不被执行的块程序（即当执行条件为 OFF 时）不占用执行时间，因而能够缩减循环时间 （有点象跳转）。同样与任务一样，也可以在块程序中暂停或重新启动其它的块程序。

但是任务和块程序仍有不同。差别之一是块程序不使用输入条件，除非特意使用 IF(802)、WAIT(805)、EXIT(806)、IEND(810) 或其它指令来编写程序。某些指令不能在块程序中使用，例如那些检测上升和下降微分的指令。

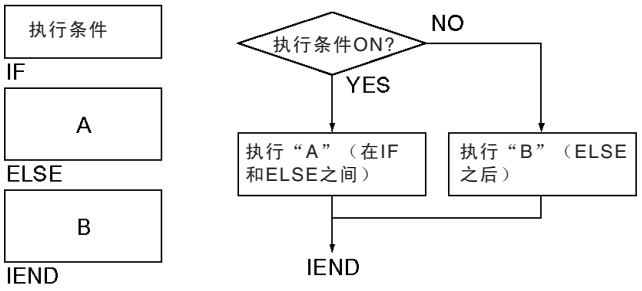
块程序既可以在循环任务中使用，也可以在中断任务中使用。从 0 ～ 127 号中的每块程序只能使用一次，不能再次使用，即使在不同任务中使用也不例外。



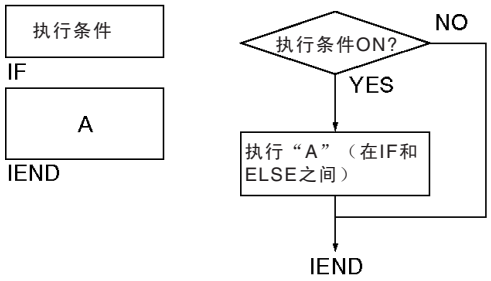
使用块程序指令

一般来说，IF(802)、ELSE(803) 和 IEND(810) 在块程序内部使用执行条件（以及位）。

如果 “A” 或 “B” 被执行，那么使用 IF A ELSE B IEND，如下所示。



如果 “A” 被执行或什么都不执行，使用 IF A IEND，如下所示。



如果执行要等待执行条件或位变 ON （例如步过程），那么使用 WAIT(805)。

如果执行要等待一个的时间 （例如定时的步过程），那么使用 TIMW(813)、TIMX(816)、 TMHW(815) 或 TMHWX(817)。

如果执行要等待到达指定的计数（例如计数器步过程），那么使用 CNTW(814)/CNTWX(818)。

如果块程序的一部分重复执行直到满足条件为止，那么使用 LOOP(809) 和 LEND(810)。

如果在块程序中间根据执行条件结束块程序的执行，使用 EXIT(806)。

如果在块程序中暂停或重启动另一个正在执行的程序，使用 BPPS(811) 和 BPRS(812)。

块程序中取执行条件的指令

下列指令能够在块程序中取执行条件。

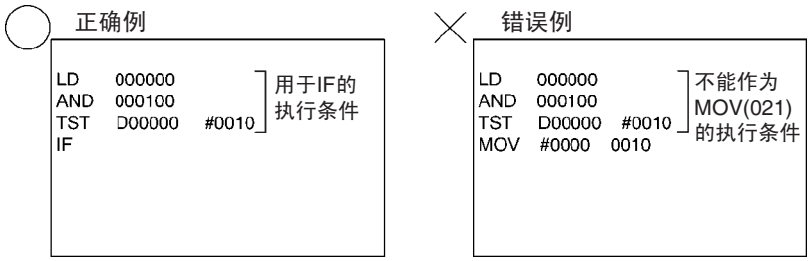
指令类型	指令名	助记符
块程序指令	如果 （非）	IF(802) (NOT)
	一次循环并等待（非）	WAIT(805) (NOT)
	退出	EXIT(806) NOT
	循环结束	LEND(810) NOT
梯形图指令	条件跳转	CJP(510)
	条件跳转非	CJPN(511)

块程序中具有使用限制的指令

下表所列指令只能用于为 IF(802)、 WAIT(805)、 EXIT(806)、 LEND(810)、CJP(510) 或 CIPN(511) 创建执行条件，而不能单独使用。如果单独使用或与其它指令组合使用，执行结果将不可预测。

助记符	名字
LD/LD NOT	加载 / 加载非
AND/AND NOT	与 / 与非
OR/OR NOT	或 / 或非
UP/DOWN	条件 ON/ 条件 OFF
>, <,=, >=, <=, <> (S) (L)	符号比较指令 （非右侧指令）
LD TST/TST NOT	加载位测试指令

助记符	名字
AND TST/TST NOT	与位测试指令
OR TST/TST NOT	或位测试指令
>$, <$,=$, >=$, <=$, <>$	文本字符串比较指令



块程序中不可用的指令

下表所列指令不能用在块程序中。

指令组	助记符	名称	替换
顺序输出指令	OUT	输出	使用 SET 和 RSET
	OUT NOT	输出非	
	DIFU(013)	上升微分	无
	DIFD(014)	下降微分	无
	KEEP(011)	保持	无
顺序控制指令	FOR(512) 和 NEXT(513)	FOR-NEXT 循环	使用 LOOP(809) 和 LEND(810) （非）
	BREAK(514)	退出循环	
	IL(002) 和 ILC(003)	互锁和互锁清除	把块程序分为小块
	JMP(004)0 和 JME(005) 0	多路跳转和多路跳转结束	使用 JMP(004) 和 JME(005) （但跳转应该无条件）
	END(001)	结束	使用 BEND(801)
定时器和计数器指令	TIM	定时器	使用 TIMW(813)、TIMWX(816)、TMHW(815)、TMHWX(817)、CNTW(814) 和 CNTWX(818)。块程序中其它指令直到定时器时间到或计数器计数值到才执行。
	TIMH(015)	高速定时器	
	TMHH(540)	1ms 定时器	
	TTIM(087)	累加定时器	
	TIML(542)	长定时器	
	MTIM(543)	多输出定时器	
	CNT	计数器	
	CNTR(012)	可逆计数器	
子程序指令	SBN(092) 和 RET(093)	子程序入口和子程序返回	无
移位指令	SFT(010)	移位寄存器	使用其它移位指令
步指令	STEP(008) 和 SNXT(009)	步和下一步	使用 WAIT(805)
数据控制指令	PID(190)	PID 控制	无
诊断指令	FPD(269)	故障点检测	无
上升和下降微分指令	助记符带 @	上升微分指令	无
	助记符带 %	下降微分指令	无

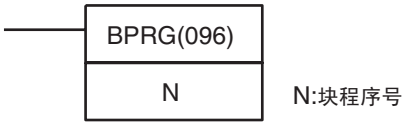
3-32-2 块程序开始 / 结束：BPRG(069)/BEND(801)

用途

定义块程序区。每一条 BPRG(069) 必须对应一条 BEND(801) 指令。

梯形图符号

块程序开始



块程序结束

BEND(801)

变化

BPRG(096)

变化	ON 条件时每个循环执行	BPRG(096)
	上升沿微分时执行一次	不支持
	下降沿微分时执行一次	不支持
立即刷新功能		不支持

BEND(801)

变化	块程序中总是执行

适用程序区

块程序区	步程序区	子程序	中断任务
（见注）	OK	OK	OK

注 BPRG(069) 只在每一个块程序的起始允许执行一次。

操作数

N：块程序号
块程序号必须在 0 和 127 十进制之间。

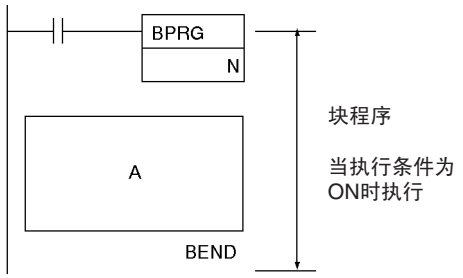
操作数规格 (BPRG(096))

区域	N
CIO 区	---
工作区	---
保持位区	---
辅助位区	---
定时器区	---
计数器区	---
DM 区	---
无区号 EM 区	---
有区号 EM 区	---

区域	N
二进制间接 DM/EM 地址	---
BCD 码间接 DM/EM 地址	---
常数	0 ～ 127 （十进制）
数据寄存器	---
索引寄存器	---
使用索引寄存器的间接寻址	---

描述

BPRG(096) 执行块号为 N 的块程序，即在该指令后面开始到 BEND(801) 为止的块程序。执行条件为 ON 时， BPRG(096) 和 BEND(801) 之间的所有指令都被执行 （即无条件的）。



当 BPRG(096) 的执行条件为 OFF，块程序不被执行，块程序中的指令不需要执行时间。

可以从另一个块程序中使用BPPS(811)来停止执行一个块程序，即使BPRG(096)的执行条件为 ON 也不例外。

标志

BPRG(096)

名称	标记	操作
错误标志	ER	如果 BPRG(096) 已经被执行时为 ON。 如果 N 不在 0 和 127 之间时为 ON。 如果多次使用相同的块程序是为 ON。 其它情况为 OFF。

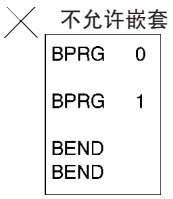
BEND(801)

名称	标记	操作
错误标志	ER	如果块程序未执行时为 ON。 其他情况下为 OFF。

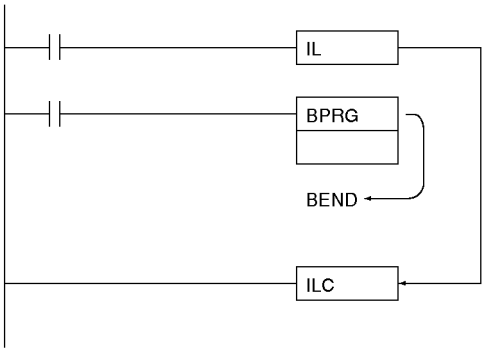
注意

在整个用户程序中每一个块程序号只能使用一次。

块程序不能嵌套。



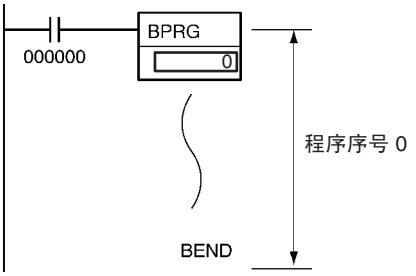
如果块程序在一个互锁程序段中，并且 IL(002) 的执行条件是 OFF，块程序不被执行。



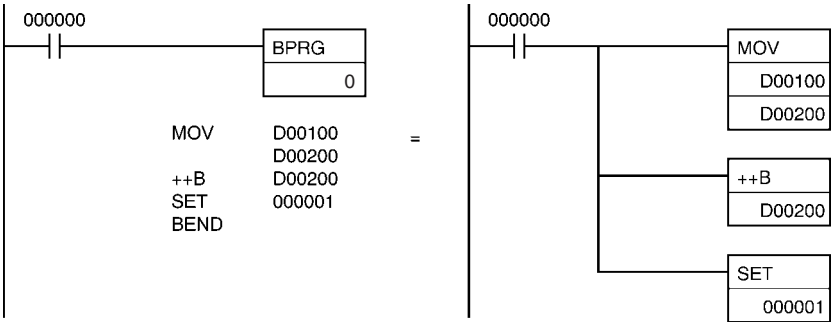
BPRG(096) 和对应的 BEND(801) 不在块程序中， N 不在＃ 0000 和＃ 007F（二进制）之间、无块程序或多次使用同一个块程序号时，这些情况都会产生错误，同时错误标志变 ON。

例

下例中当 CIO 000000 变 ON，将执行块程序 00。当 CIO 000000 为 OFF 将不执行块程序。



以下所示的两段程序在相同的执行条件下 （即当 CIO 000000 变 ON）都执行 MOV (021)、＋＋ B(594) 和 SET。

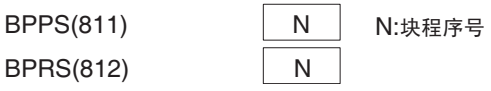


3-32-3 块程序暂停 / 重启动：BPPS(811)/BPRS(812)

用途

从另一个块程序暂停和重启动指定的块程序。

梯形图符号



变化

变化	在块程序中一直执行

使用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

注 即使在子程序和中断任务中，BPRG(096)和BPRS(812)也必须在块程序范围中。

操作数

N：块程序号
块程序号必须在十六进制 0000 和 007F Hex 之间 （十进制 0 和 127）。

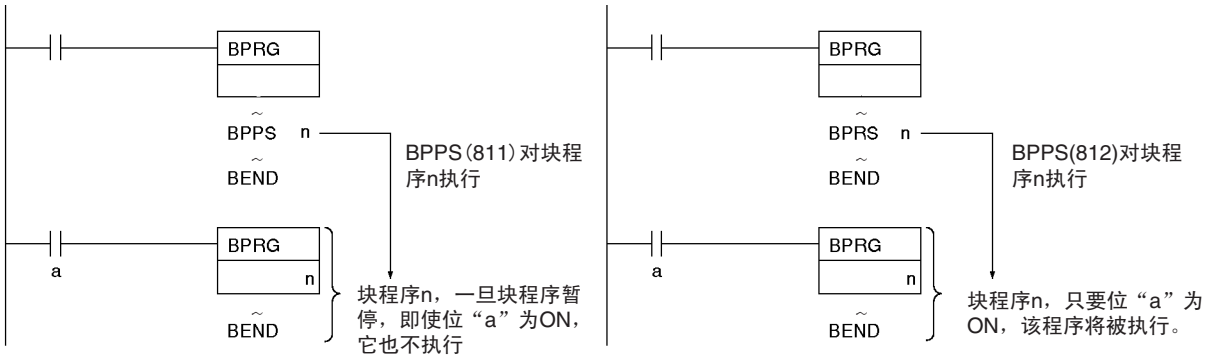
操作数规定

区域	N
CIO 区	---
工作区	---
保持位区	---
辅助位区	---
定时器区	---
计数器区	---
DM 区	---
无区号 EM 区	---
有区号 EM 区	---
二进制间接 DM/EM 区地址	---
BCD 间接 DM/EM 地址	---
常数	0 ～ 127 （十进制）
数据寄存器	---
索引寄存器	---
使用索引寄存器的间接寻址	---

描述

BPPS(811) 用于在一个块程序内部暂停另一个块程序号指定为 N 的块程序的执行。一旦块程序暂停，即使作用于该块程序的 BPRG(096) 的执行条件为 ON，该程序也不执行，当 BPRS(812) 被执行时，该块程序才能重启动。

BPRS(812) 重启动块程序号为 N 的块程序。一旦重启动，只要 BPRG(096) 的执行条件为 ON，块程序就会被执行。



标志

名称	标记	操作
错误标志	ER	如果 BPPS(811) 或 BPRS(812) 不在块程序中时为 ON。 N 不在＃ 0000 和＃ 007F （二进制）之间时为 ON。 其它情况下为 OFF。

注意

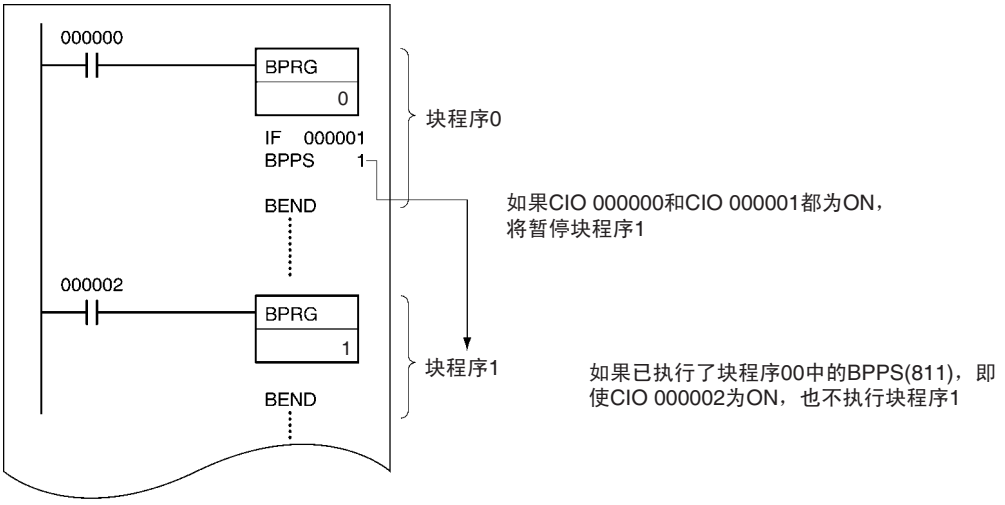
如果 BPPS(811) 或 BPRS(812) 不在块程序中或 N 不在＃ 0000 和＃ 007F（二进制）之间时会产生错误，错误标志变为 ON。

BPPS(811) 能够用于暂停包含的块程序。当该程序又被另一个块程序中的 BPRS(812) 重新启动时，暂停的块程序将从 BPPS(811) 之后的下一个指令开始重新启动。

如果一个暂停的块程序包含 TIMW(813)，TIMW(816)，TIMW(815) 或 TIMW(817)，即使块程序已被暂停，时间的当前值仍将继续下去。

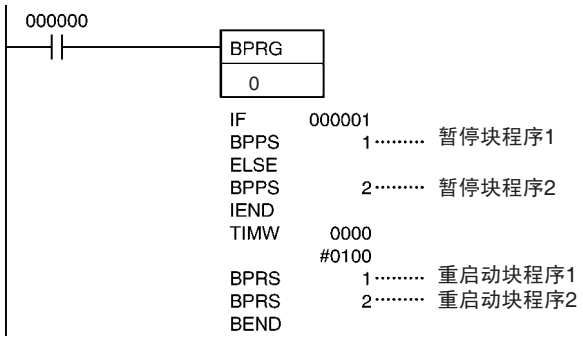
例

下例显示了一个暂停块程序的基本实例



注 如果正在暂停的块程序出现在BPPS(811)之后，将不被执行。如果出现在BPPS(811)之前，将在下一个循环开始时被暂停。

下面程序中如果 CIO 000000 为 ON，程序根据 000001 的状态，暂停块程序 1，或者暂停块程序 2。 10 秒之后重新启动暂停的块程序。



地址	指令	操作数
000000	LD	000000
000001	BPRG(096)	00
000002	IF(802)	000001
000003	BPPS(811)	01
000004	ELSE(803)	
000005	BPPS(811)	02
000006	IEND(804)	
000007	TIMW(803)	0000
		# 0100
000008	BPRS(812)	1
000009	BPRS(812)	2
000010	BEND(801)	

3-32-4 分支：IF(802)、 ELSE(803) 和 IEND(804)

用途

根据执行条件或操作数位的状态分支块程序。

梯形图符号

IF(802) B B:位操作数
IF(802)
IF(802) NOT B
ELSE(803)
IEND(804)

变化

变化	在块程序中一直执行

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

注 IF(802)、ELSE(803) 和 IEND(804) 必须在块程序区中使用，即使在子程序和中断任务中也不例外。

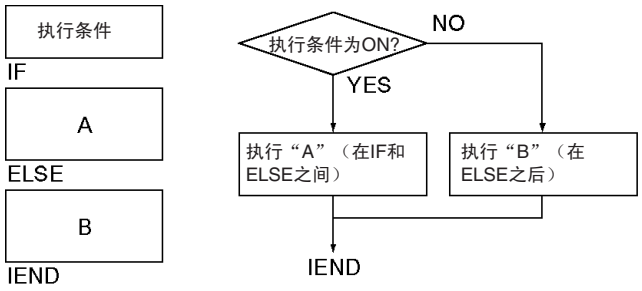
操作数规定

区域	B
CIO 区	CIO 000000 ～ CIO 614315
工作区	W00000 ～ W51115
保持位区	H00000 ～ H51115
辅助位区	A00000 ～ A44715 A44800 ～ A95915
定时器区	T0000 ～ T4095
计数器区	C0000 ～ C4095
任务标志	TK0000 ～ TK0031
条件标志	ER, CY, >, =, <, N, OF, UF, >=, <>, <=, ON, OFF, AER
时钟脉冲	0.02 s, 0.1 s, 0.2 s, 1 s, 1 min
DM 区	---
无区号 EM 区	---
有区号 EM 区	---
二进制间接 DM/EM 区地址	---
BCD 间接 DM/EM 地址	---
常数	---
数据寄存器	---
索引寄存器	---
使用索引寄存器接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

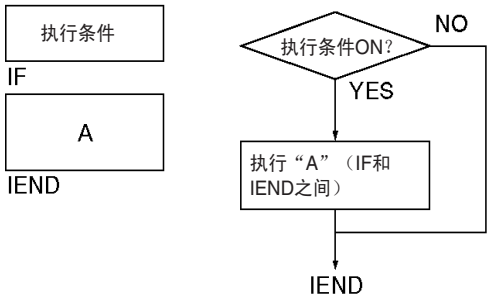
描述

IF(802) 无操作数时的操作

如果没指定操作数位，IF(802) 之前必须建立一个以 LD 开始的执行条件。如果执行条件为 ON。将执行 IF(802) 和 ELSE(803) 之间的指令。如果执行条件为 OFF，将执行 ELSE(803) 和 IEND(804) 之间的指令。

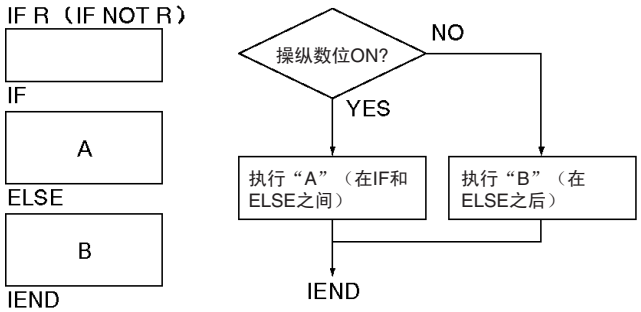


如果省略掉ELSE(803)指令，并且执行条件为ON，将执行IF(802)和IEND(804)之间的指令。如果执行条件为 OFF，只执行 IEND(804) 之后的指令。

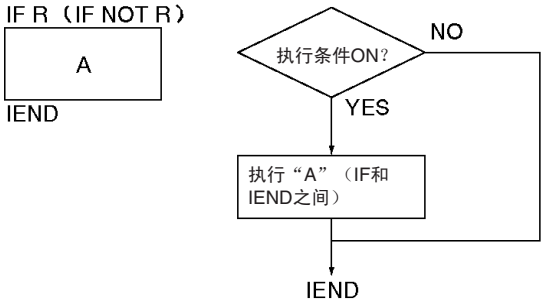


IF(802) 或 IF NOT(802) 有操作数时的操作

可以为 IF(802) 或 IF NOT(802) 指定一个操作数位 B。如果操作数位为 ON，将执行 IF(802) 和 ELSE(803) 之间的指令。对于 IF NOT(802)，操作数位为 OFF 时，执行 IF(802) 和 ELSE(803) 之间的指令，操作数为 ON 时，执行 ELSE(803) 和 IEND(804) 之间的指令。



如果省略掉 ELSE(803)，并且操作数位为 ON，将执行 IF(802) 和 IEND(804) 之间的指令。如果操作数位为 OFF，只执行 IEND(804) 之后的指令。如果使用 IF NOT(802)，操作数位将取相反状态。



标志

名称	标记	操作
错误标志	ER	如果分支指令不在块程序中时为 ON。 如果多于 254 个分支嵌套时为 ON。 其它情况下为 OFF。

注意

一般块程序中的指令是无条件执行的，但是可使用分支创建基于执行条件或操作数位的有条件的执行。

使用 IF A ELSE B IEND 来分支 A 和 B。

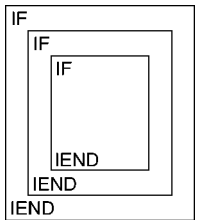
在 A 和不执行之间使用 IF A IEND 进行分支。

可最多嵌套 253 级分支。

如果分支指令不在块程序中或嵌套多余 254 个分支时，会产生错误，错误标志为 ON。

嵌套分支

在起始级分支中最多可嵌套 253 个分支。

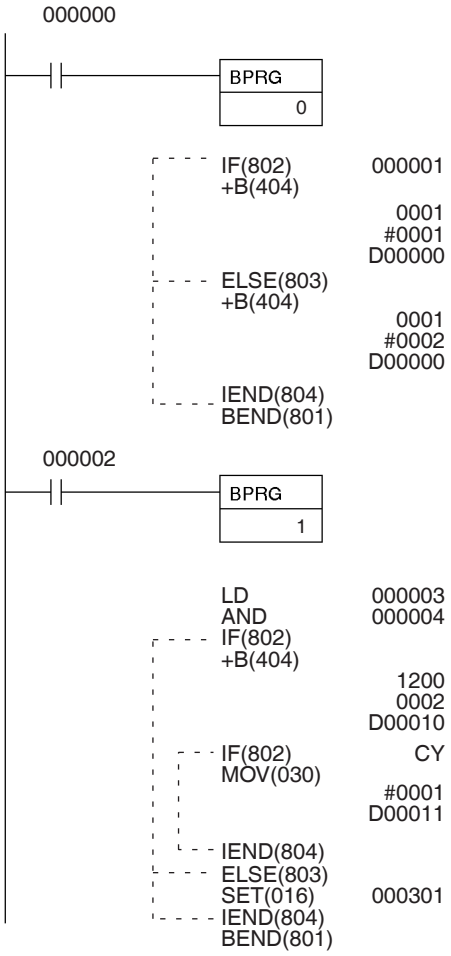


例

下例显示由 CIO 000000 和 CIO 000002 控制的两个不同的块程序。

第一块根据 CIO 000001 的状态执行 2 个加法中的一个。该块在 CIO 000000 为 ON 时执行。如果 CIO 000001 为 ON， 0001 加到 CIO 0001 的内容中。如果 CIO 000001 为 OFF，0002 加到 CIO 0001 的内容中。对任一种情况，结果放到 D00000。

第二块在 CIO 000002 为 ON 时执行，并显示了两层嵌套，如果 CIO 000003 和 CIO 000004 都为 ON，CIO 1200 和 CIO 0002 的内容相加，结果放到 D 00010，然后根据 CY 的状态将 0001 传送到 D 0001。如果 CIO 000003 或 CIO 000004 为 OFF，整个加法操作跳过，且 CIO 000301 变 ON。



地址	指令	操作数
000000	LD	000000
000001	BPRG(096)	00
000002	IF(802)	000001
000003	+B(404)	
		0001
		#0001
		D00000
000004	ELSE(803)	
000005	+B(404)	
		0001
		#0002
		D00000
000006	IEND(804)	
000007	BEND(801)	
000008	LD	000002
000009	BPRG(096)	1
000010	LD	000003
000011	AND	000004
000012	IF(802)	
000013	+B(404)	
		1200
		0002
		D00010
000014	IF(802)	A50004
000015	MOV(030)	
		#0001
		D00011
000016	IEND(804)	
000017	ELSE(803)	
000018	SET(016)	000301
000019	IEND(804)	
000020	BEND(801)	

3-32-5 条件块退出 （非）：EXIT(NOT)(806)

用途

根据操作位的状态或执行条件退出块程序 （即不执行块程序中到 BEND(801) 为止的所有指令）。当 EXIT(806) 无操作数位时，如果执行条件为 ON 时退出程序。当 EXIT(806) 有操作数位时，如果这位为 ON 时退出程序。 EXIT NOT(806) 必须有操作数位，并且当位为 OFF 时退出程序。

梯形图符号

EXIT(806)

EXIT(806) B

EXIT NOT(806) B

B:位操作数

变化

变化	在块程序中一直执行	EXIT(806) EXIT(806) B EXIT NOT(806) B

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

注 EXIT(806) 和 EXIT NOT(806) 必须在块程序区中使用，即使在子程序和中断任务中也不例外。

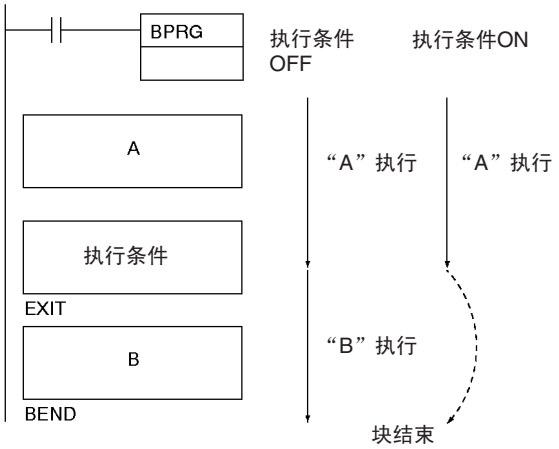
操作数规格

区域	B
CIO 区	CIO 000000 ～ CIO 614315
工作区	W00000 ～ W51115
保持位区	H00000 ～ H51115
辅助位区	A00000 ～ A44715 A44800 ～ A95915
定时器区	T0000 ～ T4095
计数器区	C0000 ～ C4095
任务标志	TK0000 ～ TK0031
条件标志	ER, CY, >, =, <, N, OF, UF, >=, <>, <=, ON, OFF, AER
时钟脉冲	0.02 s, 0.1 s, 0.2 s, 1 s, 1 min
DM 区	---
无区号 EM 区	---
有区号 EM 区	---
二进制间接 DM/EM 区地址	---
BCD 间接 DM/EM 地址	---
常数	---
数据寄存器	---
索引寄存器	---
使用索引寄存器接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

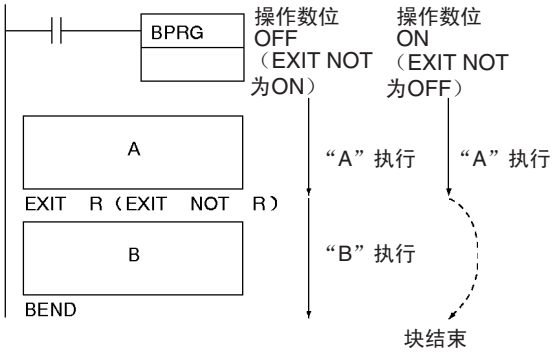
无操作数的操作

EXIT(806) 执行时可不带操作数。此时必须创建以 LD 起始的执行条件。如果执行条件为 OFF，块程序的剩余部分将正常执行。如果执行条件为 ON，将不执行块程序中到 BEND(801) 为止的剩余指令。



有操作数的操作

如果 EXIT(806) 的操作数位 B 为 OFF，块程序的剩余部分将正常执行。如果 EXIT(806) 的操作数位为 ON，将不执行块程序中到 BEND(801) 为止的剩余指令。对于 EXIT NOT(806)，当操作数位为 ON 时将执行块程序中的剩余部分，如果操作数位为 OFF，将跳过该部分。



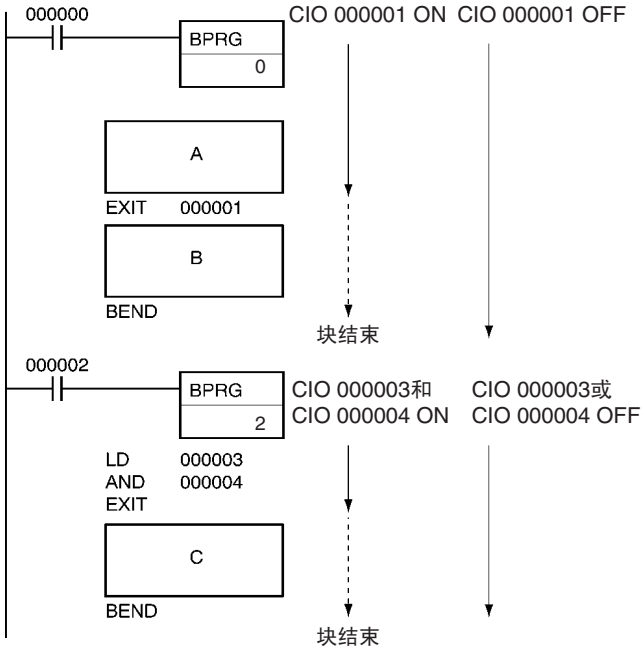
标志

名称	标记	操作
错误标志	ER	如果 EXIT(806) 或 EXIT NOT(806) 不在块程序中时为 ON。 其它情况下为 OFF。

注意 如果 EXIT(806) 或 EXIT NOT(806) 不在块程序中会产生错误，并且错误标志变 ON。

例 当 CIO 000000 为 ON 时，执行块程序。如果 CIO 000001 为 ON，执行 A，跳过 B，程序控制跳到 BEND(801)。在 CIO 000001 变为 OFF 之前，程序 B 一直被跳过。

尽管 EXIT(NOT)(806) 类似于 IF-END 编程，但通常 EXIT(NOT)(806) 的执行时间较短，因为从 EXIT(NOT)(806) 到块程序结尾之间的指令根本不执行。



3-32-6 一次循环并等待 （非）：WAIT(805)/WAIT(805) NOT

用途

停止块程序的剩余部分，直到执行条件变 ON 或 OFF 为止。

梯形图符号

WAIT(805)

WAIT(805) B　　B:位操作数

WAIT(805) NOT B

变化

变化	在块程序中一直执行

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

注　WAIT(805)/WAIT(805) NOT 可以用在块程序区，也可以是子程序和中断任务中。

操作数规格

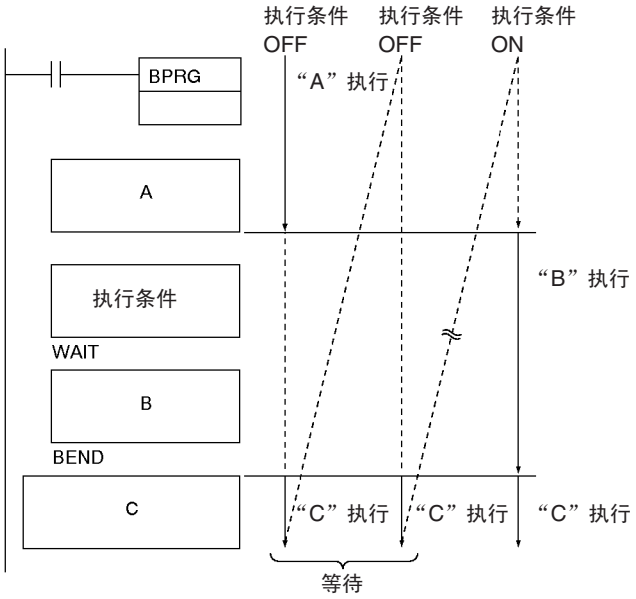
区域	B
CIO 区	CIO 000000 ～ CIO 614315
工作区	W00000 ～ W51115
保持位区	H00000 ～ H51115
辅助位区	A00000 ～ A44715 A44800 ～ A95915
定时器区	T0000 ～ T4095
计数器区	C0000 ～ C4095
任务标志	TK0000 ～ TK0031
条件标志	ER, CY, >, =, <, N, OF, UF, >=, <>, <=ON, OFF, AER
时钟脉冲	0.02 s, 0.1 s, 0.2 s, 1 s, 1 min
DM 区	---

区域	B
无区号 EM 区	---
有区号 EM 区	---
二进制间接 DM/EM 区地址	---
BCD 间接 DM/EM 地址	---
常数	---
数据寄存器	---
索引寄存器	---
使用索引寄存器接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

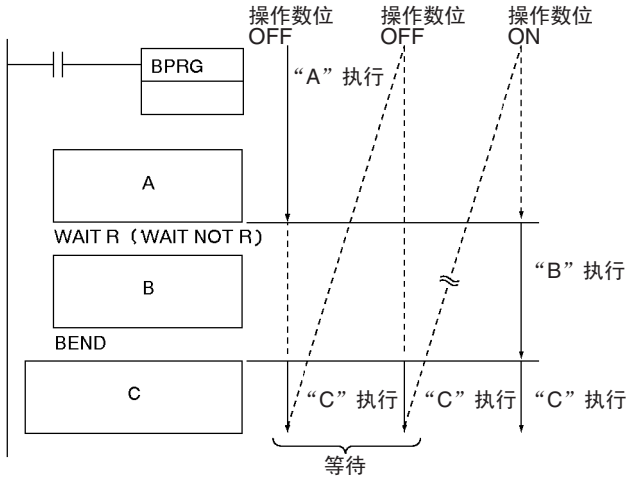
无操作数时的操作

如果没有指定操作数位，必须在 WAIT(805)/WAIT(805)NOT 之前创建以 LD 起始的执行条件。如果 WAIT(805) 的执行条件为 OFF，将跳过块程序中的剩余指令。在下一循环中，除了 WAIT(805) 的执行条件外，什么都不执行，当执行条件变为 ON 时，将执行从 WAIT(805) 开始到程序结束的指令。



有操作数的操作

可以为 WAIT(800) 或 WAIT NOT(805) 指定一个操作数位 B。如果操作数位为 OFF（对于 WAIT NOT(805) 为 ON），将跳过块程序的剩余指令。在下一个循环中，除了执行 WAIT(805) 或 WAIT(805)NOT 的执行条件外，不执行块程序的任何指令。当执行条件变 ON （对于 WAIT NOT(805) 为 OFF），将执行 WAIT(805) 或 WAIT(805)NOT 和程序结尾之间的指令。



标志

名称	标记	操作
错误标志	ER	如果 WAIT(805) 或 WAIT(805)NOT 不在块程序中时为 ON。 其它情况下为 OFF。

注意

WAIT(805) 和 WAIT(805)NOT 能够用于块程序中的步过程。

如果 WAIT(805) 或 WAIT(805)NOT 不在块程序中，会产生错误，且错误标志为 ON。

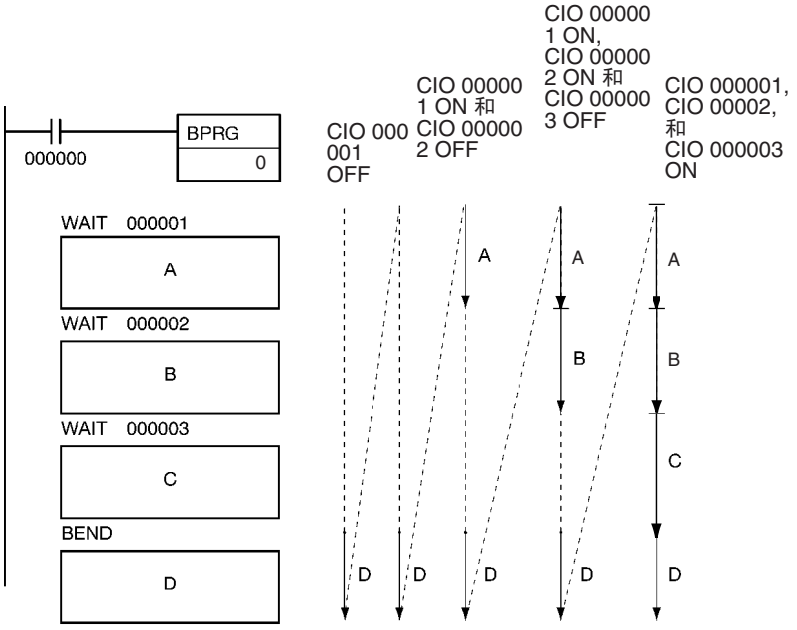
注 当指定操作数的 WAIT 指令的程序地址和无操作数的 WAIT 指令的执行条件的第一个指令的程序地址记录在内存中，这将使得基于执行条件 / 位操作数的执行过程得以继续。但是如果用外围设备执行了在线操作。WAIT 状态将被清除，并且块程序又从开头重新执行。

例

下例中当 CIO 000000 为 ON 时，块程序 00 将会执行。执行过程如下：

1,2,3...

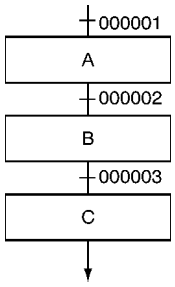
1. 如果 CIO 000001 为 OFF，不执行块程序，直到 CIO 000001 变 ON。当 CIO 000001 变 ON，将执行 “A”。
2. 如果执行 “A”之后 CIO 000002 为 OFF，将不执行块程序的剩余部分，直到 CIO 000002 变 ON。当 CIO 000002 变 ON，将执行 “B”。
3. 如果执行 “B”之后 CIO 000003 为 OFF，将不执行块程序的剩余部分，直到 CIO 000003 变 ON，当 CIO 000003 变 ON，将执行 C，并且重复以上的执行过程。



下表显示操作数位和块程序之间的关系。

操作数位			程序执行		
CIO 000001	CIO 000002	CIO 000003	第一个循环 CIO 000000 为 ON	下一个循环	以后的循环
OFF	任何状态	任何状态	都不执行	都不执行 等待 CIO 000001	当 CIO 000001 变 ON，执行 “A” 并检查 CIO 000002 的状态。
ON	OFF	任何状态	执行 “A”	等待 CIO 000002	当 CIO 000002 变 N，执行 “B” 并检查 CIO 000003 状态。
ON	ON	OFF	执行 “A” 和 “B”	等待 CIO 000003	当 CIO 000003 变 ON，执行 “C”。
ON	ON	ON	执行 “A”、“B” 和 “C”	执行 “A”、“B” 和 “C”	

如下例所示，WAIT(805) 和 WAIT(805)NOT 能用于在块程序中逐步执行各步。



3-32-7 定时器等待：TIMW(813)

用途

延迟执行块程序的剩余部分，直到指定的时间过去为止。当定时器时间到，将继续执行 TIMW(813) 之后的指令。

梯形图符号

当前值 (PV) 刷新模式 : BCD

TIMW(813)	N	N: 定时器号
	SV	SV:设置值

当前值 (PV) 刷新模式 : 二进制

TIMWX(816)	N	N: 定时器号
	SV	SV:设置值

变化

变化	在块程序区中一直执行

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	不允许

注 TIMW(813) 必须在块程序区中使用，即使在子程序中也不例外。

操作数

N：定时器号
BCD: 0000 ～ 4095 （十进制）
二进制 : 0000 ～ 4095 （十进制）
S: 设置值
BCD: #0000 ～ #9999(BCD)
二进制 : &0 ～ &65535 （十进制）
#0000 ～ FFFF （十六进制）

操作数规格

区域	N	SV
CIO 区	---	CIO 0000 ～ CIO 6143
工作区	---	W000 ～ W511
保持位区	---	H000 ～ H511
辅助位区	---	A000 ～ A447 A448 ～ A959
定时器区	0000 ～ 4095	T0000 ～ T4095
计数器区	---	C0000 ～ C4095
DM 区	---	D00000 ～ D32767
无区号 EM 区	---	E00000 ～ E32767
有区号 EM 区	---	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 区地址	---	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 地址	---	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)

区域	N	SV
常数	---	BCD: #0000 ～ 9999 (BCD) “&”不能使用 二进制： &0 ～ &65535 （十进制） #0000 ～ #FFFF （十六进制）
数据寄存器	---	DR0 ～ DR15
索引寄存器	---	---
使用索引寄存器接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

描述

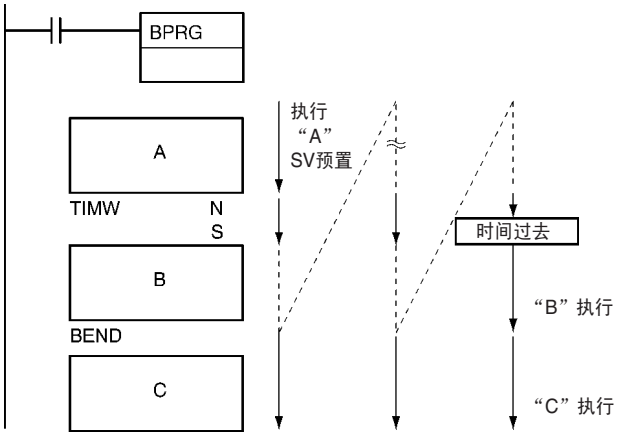
TIMW(813)/TIMWX(816) 在它之前和之后的块程序指令之间创建一个 ON 一延时倒计时定时器（SV 设置为 100ms 定时器）。TIMW(873) 能够在 0～999.99 之间定时，精确度为 0 ～ 0.01s。

注 CS1D CPU 单元的定时器精度为 10ms + 扫描周期

刚开始进入块程序时，从块程序的开始部分执行。当执行到 TIMW(813) 时，完成标志复位为 OFF，定时器预置成 SV，块程序的其余部分将等到 SV 时间到后再执行。

定时器倒计时过程中，只执行 TIMW(813) 刷新定时器。当定时器时间到时，完成标志为 ON，并执行块程序的剩余部分。整个块程序执行完成后，重复该过程。

TIMW(813) 可以看作是一个以定时器为执行条件的 WAIT 指令，因此它可用于定时的同步过程。



标志

名称	标记	操作
错误标志	ER	如果 TIMW(813) 不在块程序中时为 ON。 如果 N 以 BCD 使用间接 IR 指定而地址不是定时器当前值时为 ON。 如果 SV 不是 BCD 时为 ON。 其它情况下为 OFF。

注意

如果定时器的完成标志强制置位，将执行 TIMW(813) 之后的剩余块程序。

如果强制复位定时器完成标志，只执行块程序中的 TIMW(813)，直到清除强制复位状态为止。

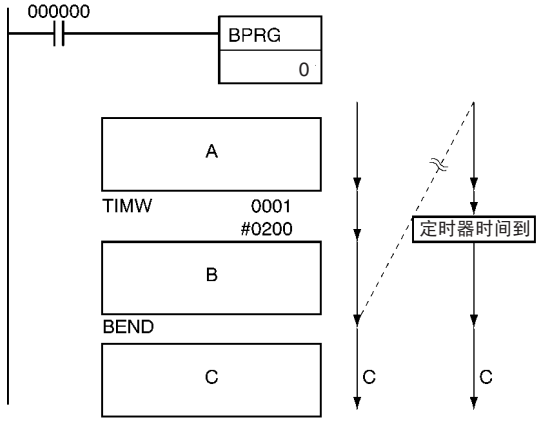
即使定时器处于待机状态， 0000 ～ 2047 号定时器的当前值也仍然刷新。2048~4095 号定时器在待机状态下保持不变。

其它的定时器指令也使用定时器号。如果多个定时器指令使用相同的定时器号，程序运行将不可预测。每个定时器只使用一次。只有同时只有一个定时器在运行时，才可使用同一个定时器号。如果相同定时器在多个定时器指令中使用，程序检查时会出现错误。

如果 N 使用间接 IR 指令，而地址不是用于一个定时器的当前值，或者如果 SV 不是 BCD 时，会产生错误，错误标志变 ON。

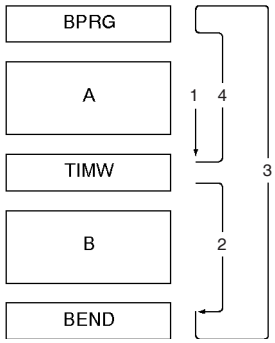
例

下例中，只要 CIO 000000 为 ON，“A”执行 20 秒后执行 “B”



地址	指令	操作数
000200	LD	000000
000201	BPRG	0
. .	A	. .
000210	TIMW	0001
		#0200
. .	B	. .
000220	BEND	---

如下图所示，程序执行流程为从 2 ～ 3 ～ 4，然后返回到 2，20s 后执行“B”。



3-32-8 计数器等待：CNTW(814) 和 CNTWX(818)

用途

延迟执行程序的其余部分，直到到达指定的计数值为止。当计数器值计到时，程序将从 CNTW(814)/CNTWX(818) 之后的下一条指令开始继续执行。

梯形图符号

当前值 (PV) 刷新模式 : BCD

CNTW(814)	N	N:计数器号
	SV	SV:设定值
	I	I:计数输入

当前值 (PV) 刷新模式 : 二进制

CNTWX(818)	N	N:计数器号
	SV	SV:设定值
	I	I:计数输入

变化

变化	在块程序中一直执行

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

注 CNTW (814)/CNTWX(818) 必须在块程序中使用，即使在子程序中和中断任务中也不例外。

操作数

N：计数器号
BCD:0 ～ 4095 （十进制）
二进制：0 ～ 4095 （十进制）
S：设定值
BCD：#0000 ～ #9999 （BCD）
二进制：&0 ～ 65535 （十进制）
#0000 ～ FFFF （十六进制）

操作数规格

区域	N	SV	I
CIO 区	---	CIO 0000 ～ CIO 6143	CIO 000000 ～ CIO 614315
工作区	---	W000 ～ W511	W00000 ～ W51115
保持位区	---	H000 ～ H511	H00000 ～ H51115
辅助位区	---	A000 ～ A447 A448 ～ A959	A00000 ～ A44715 A44800 ～ A95915
定时器区	---	T0000 ～ T4095	T0000 ～ T4095
计数器区	C0000 ～ C4095	C0000 ～ C4095	C0000 ～ C4095
任务标志	---		TK0000 ～ TK0031
条件标志	---		ER, CY, >, =, <, N, OF, UF, >=, <>, <=, ON,OFF, AER
时钟脉冲	---		0.02 s, 0.1 s, 0.2 s, 1 s, 1 min
DM 区	---	D00000 ～ D32767	---
无区号 EM 区	---	E00000 ～ E32767	---
有区号 EM 区	---	En_00000 ～ En_32767 (n = 0 ～ C)	---
二进制间接 DM/EM 区地址	---	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	---
BCD 间接 DM/EM 地址	---	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	---
常数	---	BCD: #0000 ～ 9999 (BCD) “&”不能使用 二进制 : &0 ～ &65535 （十进制） #0000 ～ #FFFF （十六进制）	---
数据寄存器	---	DR0 ～ DR15	---
索引寄存器	---		
使用索引寄存器接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

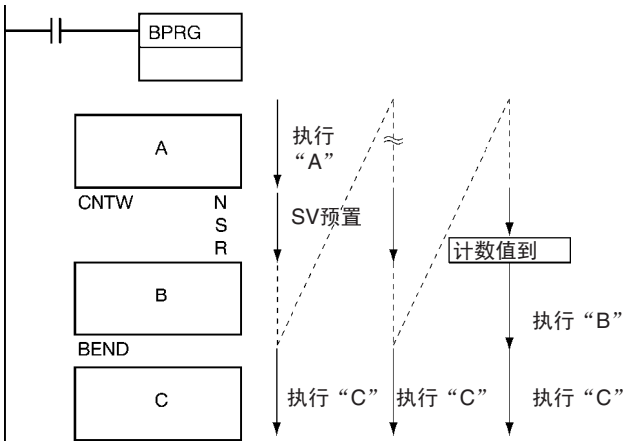
描述

CNTW(814)/CNTWX(818) 创建一个递减计数器，用来延迟执行该指令之后的块程序指令，直到计数值到为止。 CNTW(814) 设定值指定为 0000 和 9999 之间的 BCD 数。CNTWX(818) 设定值指定为 0000 和 FFFF hex 之间的二进制数。

首次进入块程序时执行块程序的开始部分，当执行到 CNTW(814))/CNTWX(818) 时，完成标志复位为 0，计数器预置为 SV，块程序的其余部分将等到计数器值到后再执行。计数器对 I （计数器输入）的脉冲 （上升微分计数）。

计数器往下计数时，仅执行 CNTW(814))/CNTWX(818) 来刷新计数器。计数器计数到时，完成标志变 ON，执行余下的块程序。一旦整个块程序执行后，过程将重复。

CNTW(814))/CNTWX(818) 可看作一个以计数器为执行条件的 WAIT 指令，因而它能够用于定时的步过程。



标志

名称	标记	操作
错误标志	ER	如果 CNTW(814) 不在块程序中时为 ON。 如果 N 使用间接 IR 指定，而该地址不适用于计数器当前值时为 ON。 如果设置为 BCD 模式，而 SV 不是 BCD 时为 ON。 其它情况下为 OFF。

注意

如果计数器的完成标志被强制置位，将执行 CNTW(814)/CNTWX(818) 之后的块程序的剩余部分。

如果计数器的完成标志强制复位，将只执行块程序中的 CNTW(814)/CNTWX(818)，直到清除强制复位状态为止。

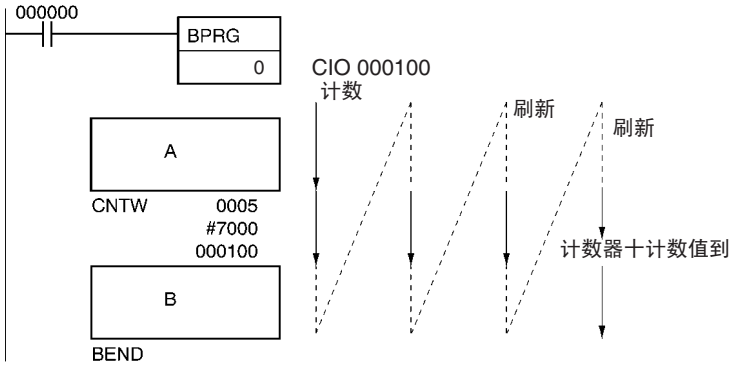
计数器号也能由其他的计数器指令所使用。如果多个计数器指令使用相同的计数器号，操作过程将变得不可预计。每一个计数器号只能使用一次。

只有当同一时间只有一个计数器在运行时，才可以使用同一个计数器号。如果多个计数器指令使用了相同的计数器号，程序检查将出现错误。

如果 N 使用了间接 IR 指令，而该地址不是计数器当前值，或者如果设置了 BCD 模式，而 SV 不是 BCD 时，会产生错误，错误标志变 ON。

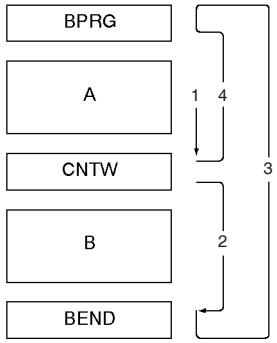
例

下例中当 CIO 000000 为 ON，将执行 “A”，等到 CIO 000100 计数到 7000 时执行块程序的剩余部分 “B”。



地址	指令	操作数
000200	LD	000000
000201	BPRG	0
. .	A	. .
000210	CNTW	0005
		#7000
		000100
. .	B	. .
000220	BEND	---

程序执行流程为2～3～4，再返回到2，7000计数到后执行“B”，如下图所示。



3-32-9 高速定时器等待：TMHW(815)/TMHWX(817)

用途

延迟剩余程序块的执行，直到指定的时间已经过去。当定时器时间到时，将从TMHW(815)/TMHWX(817) 之后的下一个指令继续执行。

梯形图符号

当前值 (PV) 刷新模式 : BCD

TMHW(815)	N	N: 定时器号
	SV	SV:设置值

当前值 (PV) 刷新模式 : 二进制

TMHWX(817)	N	N:定时器号
	SV	SV:设置值

变化

变化	在块程序中一直执行

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	不允许

注 TMHW(815)/TMHWX(817) 必须在块程序中使用，即使在子程序和中断任务中也不例外。

操作数

N：计时器号
BCD:0 ～ 4095 （十进制）
二进制：0 ～ 4095 （十进制）
S：设定值
BCD：#0000 ～ #9999(BCD)
二进制：&0 ～ 65535 （十进制）
#0000 ～ FFFF （十六进制）

操作数规格

区域	N	SV
CIO 区	---	CIO 0000 ～ CIO 6143
工作区	---	W000 ～ W511
保持位区	---	H000 ～ H511
辅助位区	---	A000 ～ A447 A448 ～ A959
定时器区	0000 to 4095	T0000 ～ T4095
计数器区	---	C0000 ～ C4095
DM 区	---	D00000 ～ D32767
无区号 EM 区	---	E00000 ～ E32767
有区号 EM 区	---	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 区地址	---	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 地址	---	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---	BCD： #0000 ～ 9999 (BCD) “&”不能使用 二进制： &0 ～ &65535 （十进制） #0000 ～ #FFFF （十六进制）
数据寄存器	---	DR0 ～ DR15
索引寄存器	---	---
使用索引寄存器接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

描述

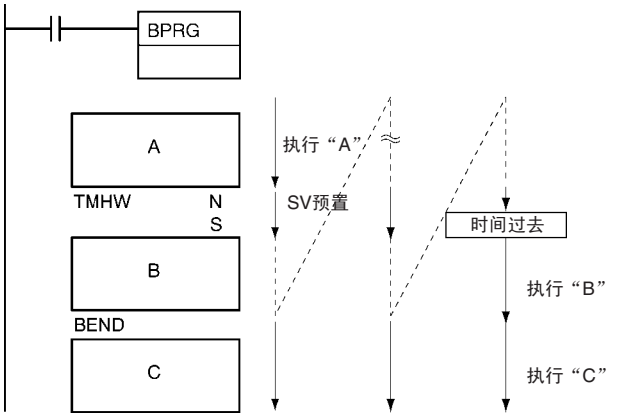
TMHW(815)/TMHWX(817)在它之前执行的块程序指令和之后的指令之间创建一个ON延迟定时器（SV中设置为10ms定时器）。THMW(815)能够在0～99.99s之间定时，精度为0～0.01s。TMHWX(817)能够在0～655.35s之间定时，精度为0～0.01s。

注 CS1D CPU单元的定时器精度为10 ms + 扫描周期

首次进入块程序时，执行块程序的开始部分。当执行到TMHW(815)/TMHWX(817)时，完成标志复位为OFF，定时器预置成SV，块程序的其余部分将等到SV时间到后再执行。

定时器倒计时时，只执行TMHW(815)/TMHWX(817)来刷新定时器，当定时器时间到时，完成标志变 ON，并执行块程序的其余部分。一旦整个块程序执行完成后，该过程重负执行。

TMHW(815)/TMHWX(817)可以看作是一个以定是器为执行条件的WAIT指令，因此它可用于定时的步过程。



标志

名称	标记	操作
错误标志	ER	如果CNTW(815)不在块程序中时为ON。 如果N使用间接IR指定，而该地址不适用于计数器当前值时为ON。 如果BCD模式下， SV不是BCD时为ON。 其它情况下为OFF。

注意

如果计数器的完成标志被强制置位，将执行CNTW(815)/TMHWX(817)之后的块程序的剩余部分。

如果计数器的完成标志强制复位，将只执行块程序中的CNTW(815)/TMHWX(817)，直到清除强制复位状态为止。

即使定时器处于待机状态， 0000～2047号定时器的当前值也会刷新。

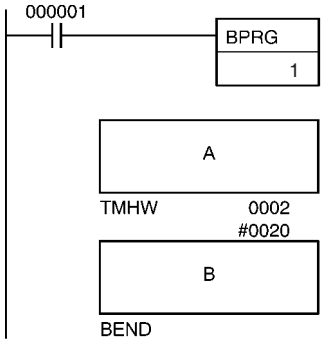
2048～4095号定时器的当前值在待机状态时保持不变。

其它的定时器指令也使用这些定时器号，如果多个定时器指令使用相同的定时器号，程序执行将不能预测，每个定时器号只使用一次。只是在同时只有一个定时器在执行时，才可能使用相同的定时器号。如果多个定时器指令使用相同的定时器号，程序检查将出现错误。

如果 N 使用了间接 IR 指令，而该地址不是计数器当前值，或者 BCD 模式下，SV 不是 BCD 时，会产生错误，错误标志变 ON。

例

下例中，只要 CIO 000000 为 ON，“A”执行 0.2 秒后执行 “B”。



地址	指令	操作
000221	LD	000001
000222	BPRG	1
. .	A	. .
000250	TMHW	0002
		#0020
. .	B	. .
000281	BEND	---

3-32-10 循环控制：LOOP(809)/LEND(810)/LEND(810)NOT

用途

创建一个循环，在位操作数变 ON 或 OFF 或者执行条件变 ON 之前重复执行该循环。

梯形图符号

LOOP(809)

LEND(810)

LEND(810) B B:位操作数

LEND(810) NOT B

变化

变化	在块程序中一直执行

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

注 LOOP(809)， LEND(810) 和 LEND(810)NOT 必须在块程序中使用，即使在子程序和中断任务中也不例外。

操作数规格

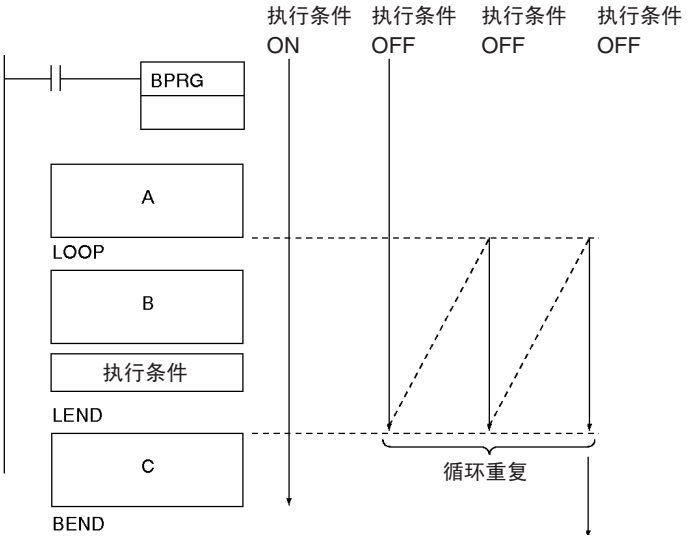
区域	B
CIO 区	CIO 000000 ～ CIO 614315
工作区	W00000 ～ W51115
保持位区	H00000 ～ H51115
辅助位区	A00000 ～ A44715 A44800 ～ A95915
定时器区	T0000 ～ T4095
计数器区	C0000 ～ C4095
任务标志	TK0000 ～ TK0031
条件标志	ER, CY, >, =, <, N, OF, UF, >=, <>, <=, ON,OFF, AER
时钟脉冲	0.02 s, 0.1 s, 0.2 s, 1 s, 1 min
DM 区	---
无区号 EM 区	---
有区号 EM 区	---
二进制间接 EM 区	---
BCD 间接 DM / EM 地址	---
常数	---
数据寄存器	---
索引寄存器	---
使用索引寄存器接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

LOOP(809) 指定循环程序的开始。LEND(810) 或 LEND(810)NOT 指定循环的结束。当到达LEND(810)NOT时，程序将返回前面的LOOP(809)，直到LEND(810)或 LEND(810)NOT 的操作数位分别变 ON 或变 OFF，或者直到 LEND(810) 的执行条件变 ON。

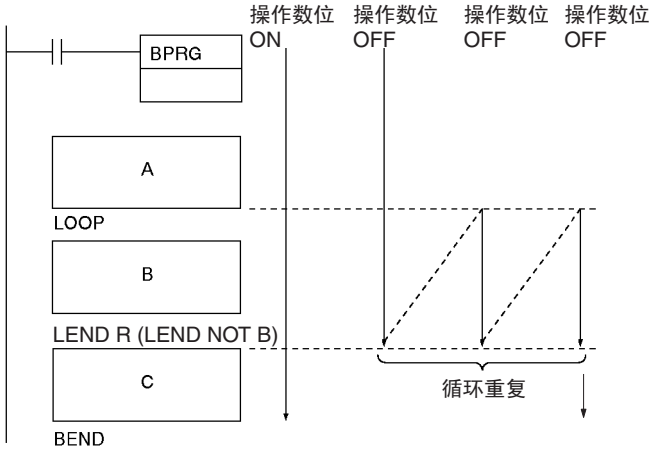
LEND(810) 使用执行条件

LEND(810) 可以带操作数位也可以不带操作数位。如果未指定操作数位，必须在 LEND(810) 之前以 LD 开始指定一个执行条件。如果执行条件为 OFF，则重复循环执行 LOOP(809) 之后的指令。如果执行条件为 ON，循环结束，并执行 LEND(810) 之后的指令。



LEND(810) 或 LEND(810)NOT 使用操作数位

LEND(810) 或 LEND(810)NOT 都可以带操作数位。如果 LEND(810) 的操作数位为 OFF （或 LEND(810)NOT 为 ON），程序将重复循环执行 LOOP(809) 之后的指令。如果 LEND(810) 的操作数位为 ON（或 LEND(810)NOT 为 OFF），循环结束，并继续执行 LEND(810) 或 LEND(810)NOT 之后的指令。



注 对LEND(810)NOT指令操纵位的状态将相反。

注 1. 执行循环内内部指令时不刷新 I/O 数据。如果必须在循环中刷新 I/O 数据，使用 IORF(184)。

2. 如果循环重复时间太长，可能超过最大扫描时间。设计程序以使得不超出最大 扫描时间。

标志

名称	标记	操作
错误标志	ER	如果循环控制指令不在块程序中时为 ON。 其它情况下为 OFF。

注意

循环内不能嵌套

不正确：

```
LOOP(809)
LOOP(809)
LEND(810)
LEND(810)
```

不要颠倒 LOOP 和 LEND 的次序

不正确：

```
LEND(810)
:
:
LOOP(809)
```

循环内部可使用有条件的块分支，但整个分支操作都必须在循环内。

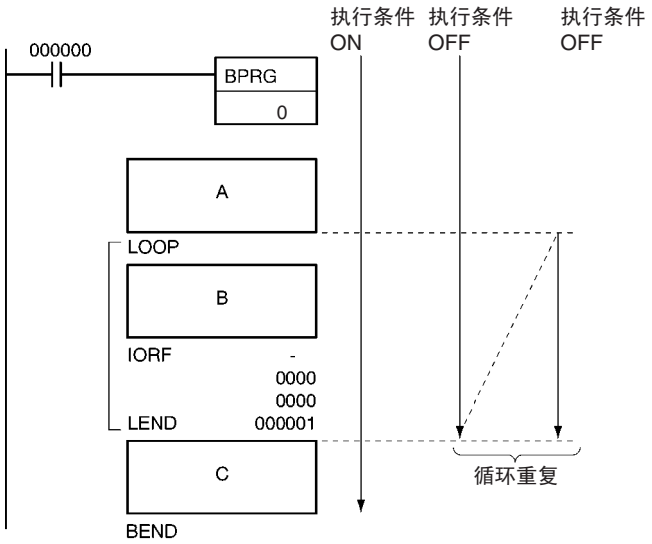
正确：	不正确：
LOOP(809)	LOOP(809)
IF(802)	IF(802)
IF(802)	IF(802)
IEND(804)	IEND(804)
IEND(804)	LEND(810)
LEND(810)	IEND(804)

如果 LOOP(809) 不被执行，将执行 NOP 过程。

如果循环控制指令不在块程序中，将产生一个错误并且错误标志将变 ON。

例

下例中当 CIO 000000 为 ON 时，块程序将被执行。“A”执行后，“B”和 IORF(184) 重复执行，直到 CIO 000001 变 ON 为止，这时“C”将被执行，然后块程序结束。



地址	指令	操作数
000220	LD	000000
000201	BPRG	0
. .	A	. .
000210	LOOP	---
. .	B	. .

地址	指令	操作数
000220	IORF	. .
		0000
		0000
000221	LEND	000001
. .	C	. .
000220	BEND	---

3-33 文本字符串处理指令

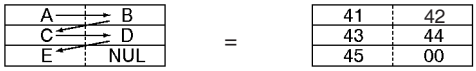
这部分描述用于操作文本字符串的指令

指令	助记符	函数代码	页
传送字符串	MOV$	664	1013
串接字符串	+$	656	1015
获取左边字符串	LEFT$	652	1018
获取右边字符串	RGHT$	653	1020
获取中间字符串	MID$	654	1022
寻找字符串	FIND$	660	1024
字符串长度	LEN$	650	1026
替换字符串	RPLC$	661	1028
删除字符串	DEL$	658	1031
交换字符串	XCHG$	665	1033
清除字符串	CLR$	666	1035
插入字符串	INS$	657	1037
字符串比较指令	=$, <>$, <$, <=$, >$, >=$	670 ～ 675	1040

3-33-1 文本字符串处理概述

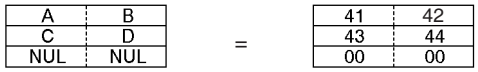
从起始到 NUL 代码 (00H) 之间的数据被作为用 ASCII 表示的字符串数据来处理（除了 1 个字节、特殊字符外）。它由左至右存储字节，由右至左存储字。当字符数为奇数时，十六进制 00（NUL 代码）被存储在最后一个字的右边字节处。

例：文本字符串



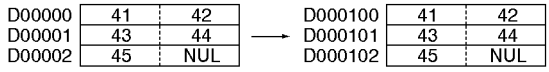
当字符数为偶数时，十六进制 0000 （两个 NUL 代码）存贮在最后一个字的左边和右边字节处。

例：文本字符串

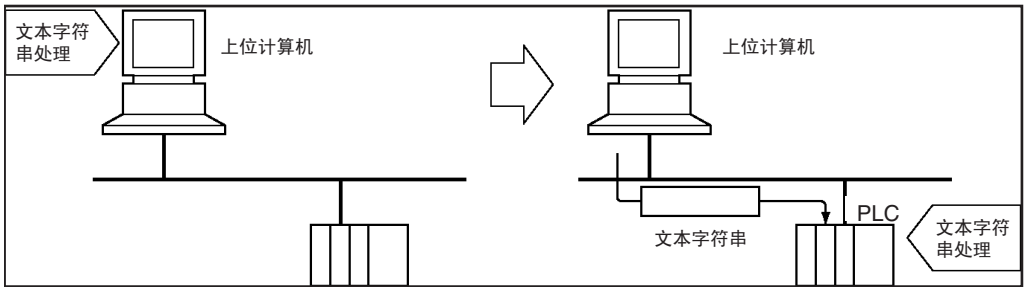


如下图所示，当指定一个文本字符串时，可以只简单的指定该字符串的第一个字。从开始到下一个 NUL 代码 （十六进制 00）之间的文本字符串数据作为一个 ASCII 数据块处理。

: MOV$ D00000 D00100



PLC 中的文本字符串处理指令能够用于执行原来上位计算机中的各种文本字符串的处理 （产品数据等）。



例如，生产计划数据中的产品能够从上位计算机传送到 PLC。各种操作能够在PLC中完成，如插入和重排文本字符串，从而减少上位计算机的数据处理负担。

ASCII 字符

文本字符串处理指令能够处理的 ASCII 字符如下表所示。

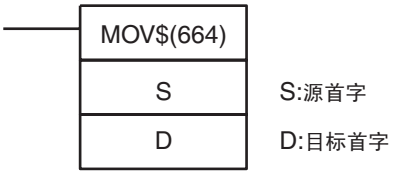
右四位 \ 左四位	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0			SP	0	@	P	‘	p				ー	タ	ミ		
1			!	1	A	Q	a	q			。	ア	チ	ム		
2			”	2	B	R	b	r			「	イ	ツ	メ		
3			#	3	C	S	c	s			」	ウ	テ	モ		
4			$	4	D	T	d	t			、	エ	ト	ヤ		
5			%	5	E	U	e	u			・	オ	ナ	ユ		
6			&	6	F	V	f	v			ヲ	カ	ニ	ヨ		
7			’	7	G	W	g	w			ァ	キ	ヌ	ラ		
8			(	8	H	X	h	x			ィ	ク	ネ	リ		
9			)	9	I	Y	i	y			ゥ	ケ	ノ	ル		
A			*	:	J	Z	j	z			ェ	コ	ハ	レ		
B			+	;	K	[	k	{			ォ	サ	ヒ	ロ		
C			,	<	L	¥	l	|			ャ	シ	フ	ワ		
D			ー	=	M	]	m	}			ュ	ス	ヘ	ン		
E			.	>	N	^	n	~			ョ	セ	ホ	゛		
F			/	?	O	_	o				ッ	ソ	マ	°		

3-33-2 传送字符串：MOV$(664)

用途

传送文本字符串

梯形图符号



变化

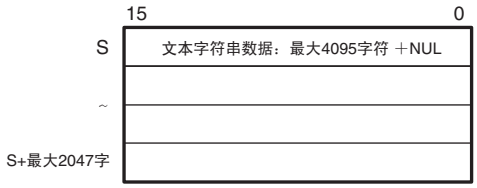
变化	ON 条件时每个循环执行	MOV$(664)
	上升沿微分执行一次	@MOV$(664)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

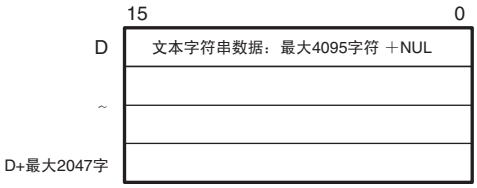
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

S：源首字



D：目标首字



注 1. S ～ S+ 最大 2047 字之间的数据以及 D ～ D+ 最大 2047 字之间的数据必须在相同的数据区。

2. S～S+最大2047字之间的数据和D～D+最大2047字之间的数据可以重叠。

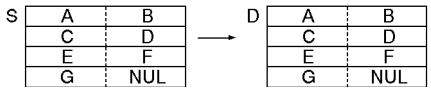
操作数规格

区域	S	D
CIO 区	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W511	
保持位区	H000 ～ H511	
辅助位区	A000 ～ A447 A448 ～ A959	A448 ～ A959
定时器区	T0000 ～ T4095	
计数器区	C0000 ～ C4095	
DM 区	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 区	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	
数据寄存器	---	

区域	S	D
索引寄存器	---	
使用索引寄存器接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

描述

MOV$(664) 按文本字符串数据 （包括最后的 NUL）的原样把 S 中指定的文本字符串传送到 D 中。 S 中能够指定的最大字符数是 4095 （十六进制 0FFF）。



标志

名称	标记	操作
错误标志	ER	如果 S 中指定的字符超出 4095 个时为 ON。 其它情况下为 OFF。
等于标志	=	如果 0000 （十六进制）传送到 D 时为 ON。 其它情况下为 OFF。

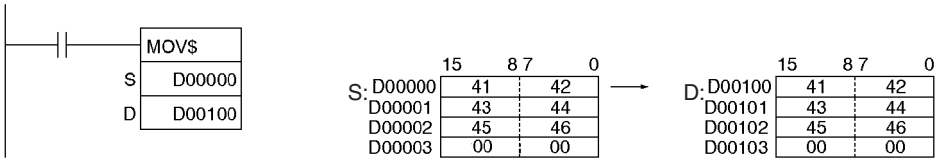
注意

如果 S 中指定的字符超出 4095 个，将产生错误，错误标志变 ON。

如果 0000 （十六进制）被传送到 D，等于标志变 ON。

例

该例中， MOV$(664) 用于传送文本字符串 ABCDEF。

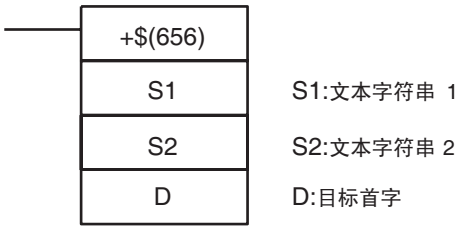


3-33-3 串接字符串：+$(656)

用途

链接一个文本字符串到另一个文本字符串。

梯形图符号



变化

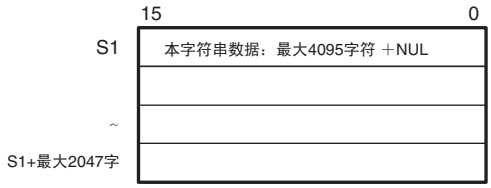
变化		
	ON 条件时每个循环执行	+$(656)
	上升沿微分执行一次	@+$(656)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

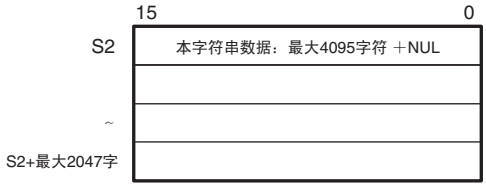
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

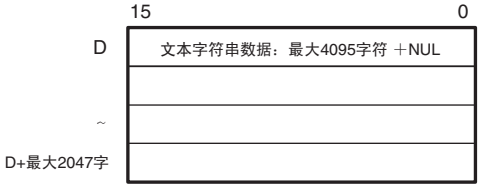
S1：文本字符串



S2：文本字符串 2



D+：目标首字



注

1. S1 ～ S1+ 最大 2047 字之间的数据、S2 ～ S2 ＋最大 2047 字之间的数据 D ～ D+ 最大 2047 字之间的数据必须在相同的数据区。
2. S2 ～ S2+ 最大 2047 字之间的数据和 D ～ D+ 最大 2047 字之间的数据可以重叠。

操作数规格

区域	S1	S2	D
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A447 A448 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ 32767 (n = 0 ～ C)		
二进制间接 DM/EM 区	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		

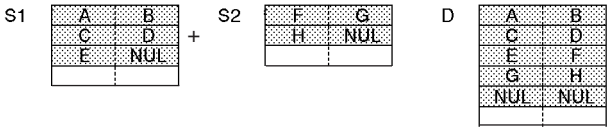
区域	S1	S2	D
BCD 间接 DM / EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---		
数据寄存器	---		
索引寄存器	---		
使用索引寄存器接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0V ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

描述

+$(664) 把 S1 指定的文本字符串数据链接到 S2 指定的文本字符串数据上，并把结果作为文本字符串数据 （包括最后的 NUL）输出到 D 中。

S1 和 S2 所能指定的最大字符数是 4095 （十六进制 0FFF）。如果第 4096 字符之前无 NUL，将产生错误，错误标志变 ON。此外，串接结果不能超出 4095 字符（十六进制 0FFF），如果超出此限，只有 4095 字符＋ NUL 被输出到 D 中。

如果 S1 和 S2 都有 NUL，两个 NUL 字符 （十六进制 0000）将输出到 D。



标志

名称	标记	操作
错误标志	ER	如果 S1 和 S2 中指定的字符超出 4095 个时为 ON。 其它情况下为 OFF。
等于标志	=	如果 0000 （十六进制）传送到 D 时为 ON。 其它情况下为 OFF。

注意

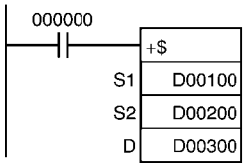
如果 S1 和 S2 中指定的字符超出 4095 个，将产生错误，错误标志变 ON。

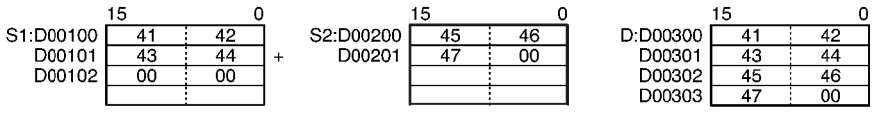
如果 0000 （十六进制）被传送到 D，等于标志变 ON。

不要使 D 中指定的起始字与 S2 的字符数据区重叠，否则指令不能正确执行。

例

该例中，使用 +$(656) 连接文本字符串 ABCD 和 EFG，并输出到结果 D 中。



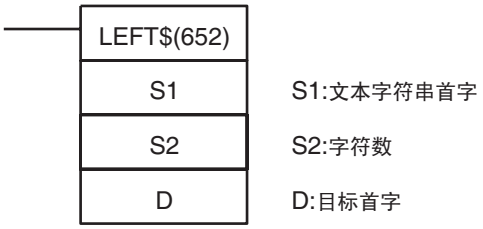


3-33-4 获取左边字符串：LEFT$(652)

用途

从文本字符串的左边 （开始）获取指定数目的字符。

梯形图符号



变化

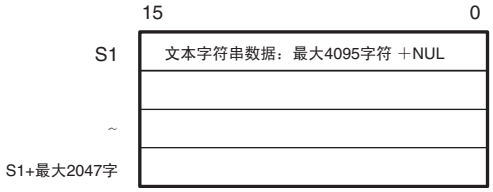
变化	ON 条件时每个循环执行	LEFT$(652)
	上升沿微分执行一次	@LEFT$(652)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

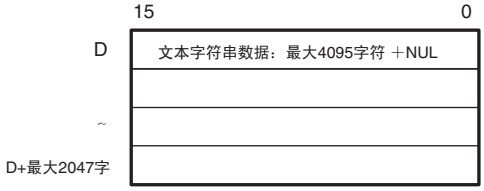
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

S1：文本字符串



S2：字符数 （十六进制 0000 ～ 0FFF 或 &0 ～ &4095）



注
1. S1 ～ S1+ 最大 2047 字之间的数据和 D ～ D+ 最大 2047 字之间的数据必须在相同的数据区。
2. S1 ～ S1+ 最大 2047 字之间的数据和 D ～ D+ 最大 2047 字之间的数据可以重叠。

操作数规格

区域	S1	S2	D
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A447 A448 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		

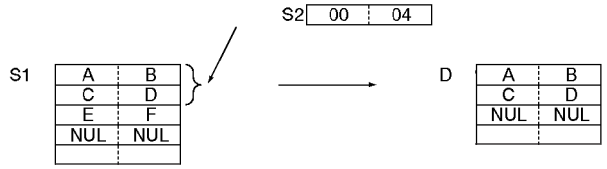
区域	S1	S2	D
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 区	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---	#0000 ～ #0FFF（二进制）或 &0 ～ &4095	---
数据寄存器	---	DR0 ～ DR15	---
索引寄存器	---		
使用索引寄存器接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

描述

LEFT$(652) 读取从 S1 指定的文本字符串第一个字的左边（开始处），到 NUL 代码 （十六进制 00）之间的字符，读取的字数由 S2 指定，并将结果输出到 D（结束处加 NUL）。

如果读取的字符数超过 S1 指定的字符数，S1 全部的文字字符串输出到 D 中。

如果读取的字符数为 0 （0000 十六进制），两个 NUL 字符 （十六进制 0000）将输出到 D。



标志

名称	标记	操作
错误标志	ER	如果 S1 和 S2 中指定的字符超出 4095 个时为 ON。 其它情况下为 OFF。
等于标志	=	如果 0000 （十六进制）传送到 D 时为 ON。 其它情况下为 OFF。

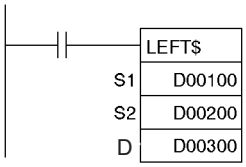
注意

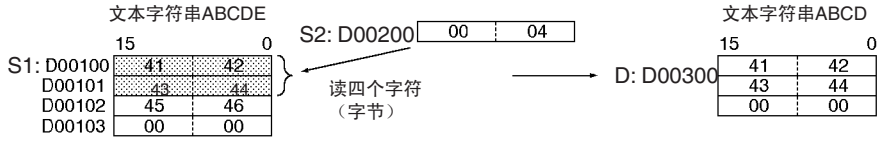
S2 能够指定读取的最大字符数是 4095 （十六进制 0FFF）个，超出将产生错误，错误标志变 ON。

如果 0000 （十六进制）被传送到 D，等于标志变 ON。

例

下例中， LEFT$(652) 用于读取 4 个字符。



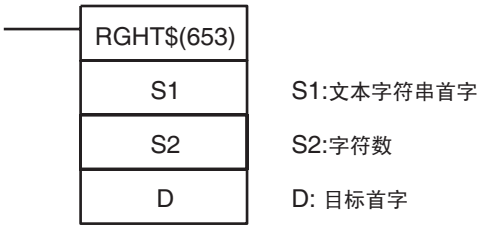


3-33-5 获取右边字符串：RGHT$(653)

用途

从文本字符串右边 （结束）位置开始读取指定的字符数。

梯形图符号



变化

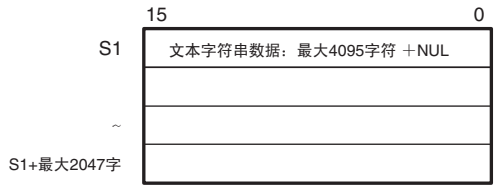
变化	ON 条件时每个循环执行	RGHT$(653)
	上升沿微分执行一次	@RGHT$(653)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

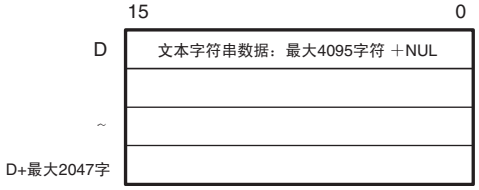
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

S1：文本字符串



S2：字符数 （十六进制数 0000 ～ 0FFF 或 0 ～ &4095）



注

1. S1 ～ S1+ 最大 2047 字之间的数据和 D ～ D+ 最大 2047 字之间的数据必须在相同的数据区。
2. S1 ～ S1+ 最大 2047 字之间的数据和 D ～ D+ 最大 2047 字之间的数据可以重叠。

操作数规格

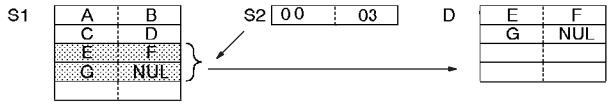
区域	S1	S2	D
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A447 A448 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 区	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---	#0000 ～ #0FFF（二进制）或 &0 ～ &4095	---
数据寄存器	---	DR0 ～ DR15	---
索引寄存器	---		
使用索引寄存器接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

描述

RGHT$(653) 从 S1 指定的文本字符串中从最右边（结尾处）向前读取 S2 指定数目的字符，并将结果输出到 D （NUL 代码添加到末尾）。

如果读取的字符数超过 S1 指定的字符数，整个 S1 文本字符串都将输出。

如果读取的字符数指定为 0（十六进制 0000），两个 NUL 字符（十六进制 0000）将输出到 D。



标志

名称	标记	操作
错误标志	ER	如果 S1 和 S2 中指定的字符超出 4095 个时为 ON。 其它情况下为 OFF。
等于标志	=	如果 0000 （十六进制）传送到 D 时为 ON。 其它情况下为 OFF。

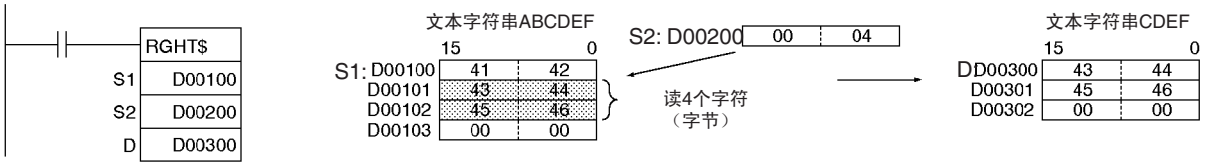
注意

S2 能够指定的最大字符数是 4095 个（十六进制 0FFF），如果指定超出此值，将产生错误，错误标志变 ON。

如果 0000 （十六进制）输出到 D，等于标志将变 ON。

例

下例中， RGHT$(653) 用于读取 4 个字符。

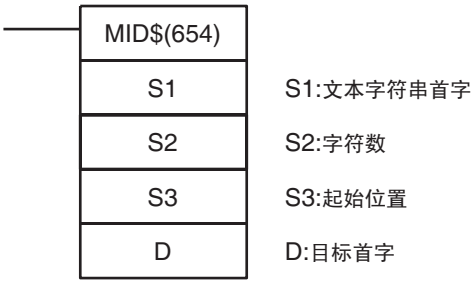


3-33-6 获取中间字符串：MID$(654)

用途

从文本字符串中间的任何位置读取指定数目的字符。

梯形图符号



变化

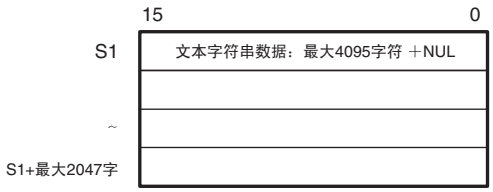
变化	ON 条件时每个循环执行	MID$(654)
	上升沿微分执行一次	@MID$(654)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

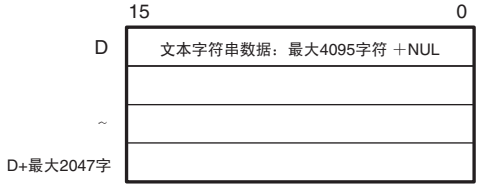
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

S1：文本字符串



S2：字符数 （十六进制数 0000 ～ 0FFF 或 0 ～ &4095）
S3：起始位置 （十六进制 0001 ～ 0FFF 或 &0 ～ &4095）



注 1. S1 ～ S1+ 最大 2047 字之间的数据和 D ～ D+ 最大 2047 字之间的数据必须在相同的数据区。

2. S1 ～ S1+ 最大 2047 字之间的数据和 D ～ D+ 最大 2047 字之间的数据可以重叠。

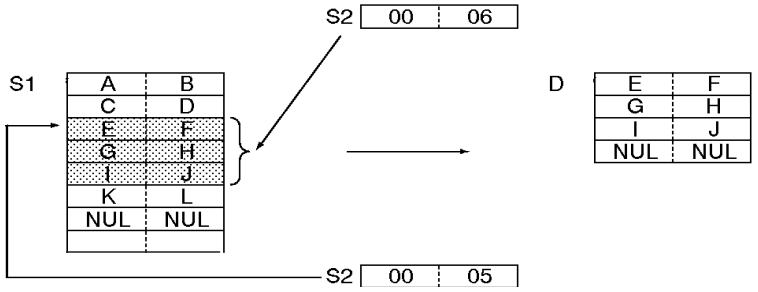
操作数规格

区域	S1	S2	S3	D
CIO 区	CIO 0000 ～ CIO 6143			
工作区	W000 ～ W511			
保持位区	H000 ～ H511			
辅助位区	A000 ～ A447 A448 ～ A959			A448 ～ A959
定时器区	T0000 ～ T4095			
计数器区	C0000 ～ C4095			
DM 区	D00000 ～ 32767			
无区号 EM 区	E00000 ～ E32767			
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)			
二进制间接 DM/EM 区	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)			
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)			
常数	---	#0000 ～ #0FFF （二进制）或 &0 ～ &4095	#0001 ～ #0FFF （二进制）或 &1 ～ &4095	---
数据寄存器	---	DR0 ～ DR15		
索引寄存器	---			
使用索引寄存器接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15			

描述

文本字符串的范围在S1指定的首字到NUL代码（十六进制00）之间，MID$(654)读取的字符数由 S2 指定，读取的起始处由 S3 指定，输出文本字符串结果到 D 中 （NUL 加在的末尾）。

如果要读取的字符的数目超过 S1 指定的文本字符串的结尾，输出的字符串将到末尾为止。



标志

名称	标记	操作
错误标志	ER	如果 S1 和 S2 中指定的字符超出 4095 个时为 ON。 S2 指定的字符数超过 4095 个 （0FFF hex）时为 ON。 如果 S3 数据在 1 ～ 4095 （十六进制 0001 ～ 0FFF）的范围外时为 ON。 其它情况下为 OFF。
等于标志	=	如果 0000 （十六进制）传送到 D 时为 ON。 其它情况下为 OFF。

注意

S3 指定的起始位置的范围是第 1 个字符到第 4095 个字符 （十六进制 0001 ～ 0FFF），如果设置超过此范围，将产生错误，错误标志变 ON。

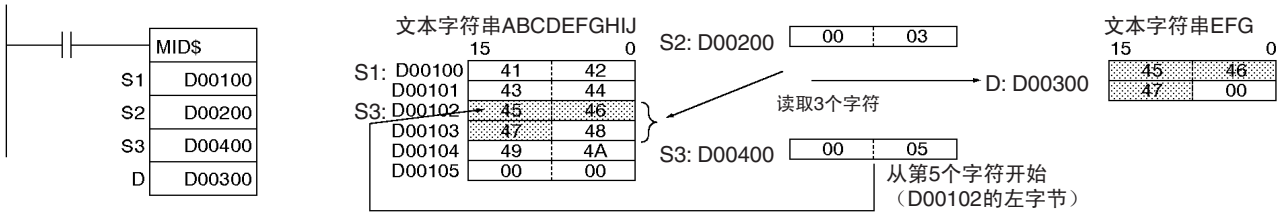
S2 所能指定的最大字符数是 4095 （十六进制 0FFF）。如果超过此限，将产生错误，错误标志变 ON。

如果读取的字符数指定为 0（十六进制 0000），2 个 NUL 字符（十六进制 0000）将输出到 D 中。

如果 0000 （十六进制）输出到 D，等于标志变 ON。

例

该例中， MID$(654) 用于读取 3 个字符。

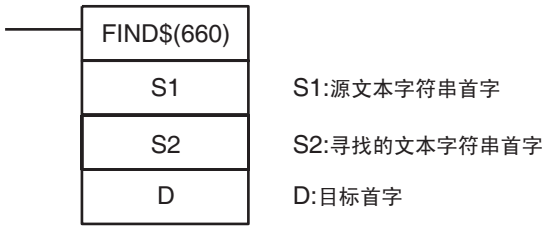


3-33-7 寻找字符串：FIND$(660)

用途

在文本字符串中寻找一个指定的文本字符串。

梯形图符号



变化

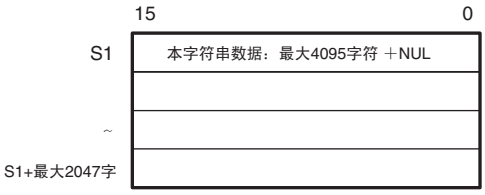
变化	ON 条件时每个循环执行	FIND$(660)
	上升沿微分执行一次	@FIND$(660)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

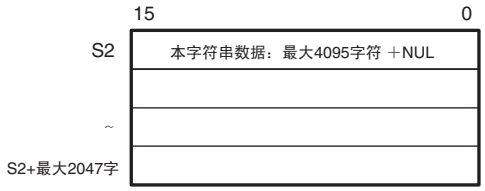
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

S1：源文本字符串



S2：寻找文本字符串



注 S1 ～ S1+ 最大 2047 字之间的数据和 S2 ～ S2+ 最大 2047 字之间的数据必须在相同的区域。

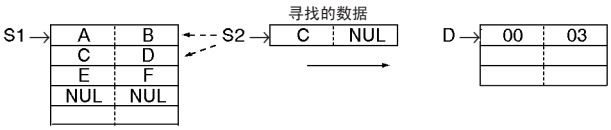
操作数规格

区域	S1	S2	D
CIO 区	CIO 0000 ～ CIO 6143		
工作区	W000 ～ W511		
保持位区	H000 ～ H511		
辅助位区	A000 ～ A447 A448 ～ A959		A448 ～ A959
定时器区	T0000 ～ T4095		
计数器区	C0000 ～ C4095		
DM 区	D00000 ～ D32767		
无区号 EM 区	E00000 ～ E32767		
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)		
二进制间接 DM/EM 区	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)		
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)		
常数	---		
数据寄存器	---		

区域	S1	S2	D
索引寄存器	---		
使用索引寄存器接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15		

描述

FIND$(660) 在 S1 指定的文本字符串中寻找 S2 指定的文本字符串，并把结果（从 S1 起始处算起的字符数）以二进制形式输出到 D。如果没有匹配的文本字符串，十六进制 0000 输出到 D 中。



标志

名称	标记	操作
错误标志	ER	如果 S1 和 S2 中指定的字符超出 4095 个时为 ON。 其它情况下为 OFF。
等于标志	=	如果 0000 （十六进制）传送到 D 时为 ON。 其它情况下为 OFF。

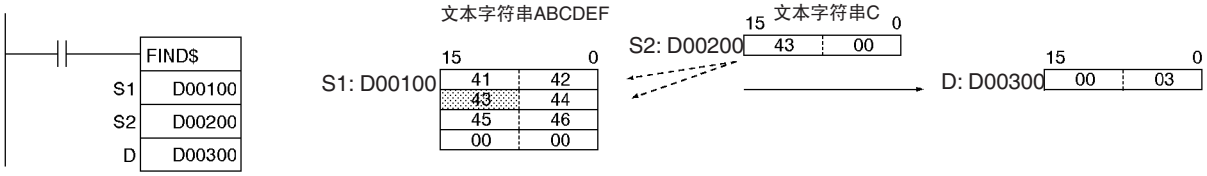
注意

S1 和 S2 所能指定的最大字符数为 4095（十六进制 0FFF）。如果超出此范围，将产生错误，错误标志变 ON。

如果 0000 （十六进制）输出到 D，等于标志变 ON。

例

该例中， FIND$(660) 用于从文本字符串中寻找一个字符。

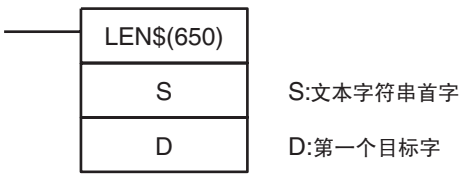


3-33-8 字符串长度 ： LEN$(650)

用途

计算文本字符串的长度。

梯形图符号



变化

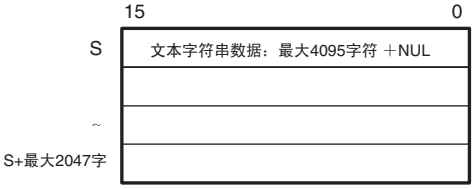
变化		
	ON 条件时每个循环执行	LEN$(650)
	上升沿微分执行一次	@LEN$(650)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

S：源文本字符串



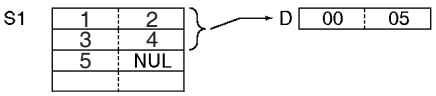
注 S ～ S+ 最大 2047 字之间的数据必须在相同的区域。

操作数规格

区域	S	D
CIO 区	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W511	
保持位区	H000 ～ H511	
辅助位区	A000 ～ A447 A448 ～ A959	A448 ～ A959
定时器区	T0000 ～ T4095	
计数器区	C0000 ～ C4095	
DM 区	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 区	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	
数据寄存器	---	DR0 ～ DR15
索引寄存器	---	
使用索引寄存器接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

描述

LENS$(650) 计算从 S 指定的文本字符串首字到 NUL 代码 （十六进制 00）的字符数 （包括 NUL 代码），并将结果以二进制数据形式输出到 D 中。如果文本字符串的起始为 NUL，计算结果为十六进制 0000。



标志

名称	标记	操作
错误标志	ER	如果计算结果超出 4095 个字符时为 ON。 其它情况下为 OFF。
等于标志	=	如果 0000 （十六进制）传送到 D 时为 ON。 其它情况下为 OFF。

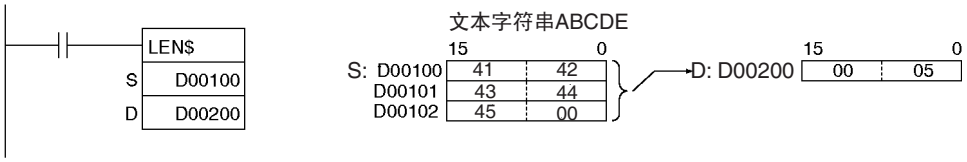
注意

最大字符数是 4095 （十六进制 0FFF）。如果超出此范围 （即在第 4096 字符前无 NUL），将产生错误，错误标志变 ON。

如果 0000 （十六进制）输出到 D，等于标志将变 ON。

例

该例中， LENS$(650) 计算字符数并输出结果。



3-33-9 替换字符串：RPLC$(661)

用途

在指定位置用指定文本字符串替换一个文本字符串。

梯形图符号

变化

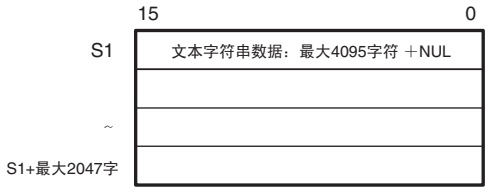
变化	ON 条件时每个循环执行	RPLC$(661)
	上升沿微分执行一次	@RPLC$(661)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

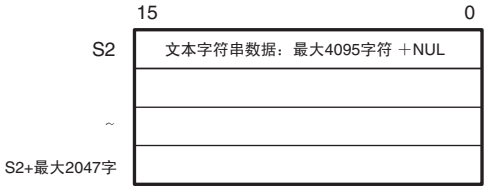
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

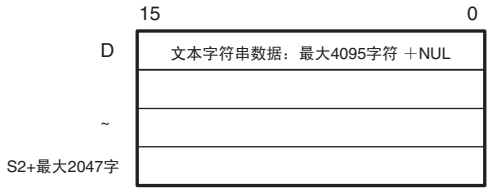
S1：文本字符串



S2：替换文本字符串



S3：字符数 （十六进制 0000 ～ 0FFF 或 &0 ～ &4095）
S4：起始位置 （十六进制 0001 ～ 0FFF&0 ～ &4095）



注 1. S1 ～ S1+ 最大 2047 字、S2 ～ S2+ 最大 2047 字和 D ～ D+ 最大 2047 字的数据必须在相同的数据区。

2. D ～ D+ 最大 2047 字和 S1 ～ S1+ 最大 2047 字或 S2 ～ S2+ 最大 2047 字的数据可以重叠。

操作数规格

区域	S1	S2	S3	S4	D
CIO 区	CIO 0000 ～ CIO 6143				
工作区	W000 ～ W511				
保持位区	H000 ～ H511				
辅助位区	A000 ～ A447 A448 ～ A959				A448 ～ A959
定时器区	T0000 ～ T4095				
计数器区	C0000 ～ C4095				
DM 区	D00000 ～ D32767				
无区号 EM 区	E00000 ～ E32767				
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)				
二进制间接 DM/EM 区	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)				
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)				

区域	S1	S2	S3	S4	D
常数	---		#0000 ～ #0FFF（二进制）或 &0 ～ &4095	#0001 ～ #0FFF（二进制）或 &1 ～ &4095	---
数据寄存器	---	DR0 ～ DR15			---
索引寄存器	---				
使用索引寄存器接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15				

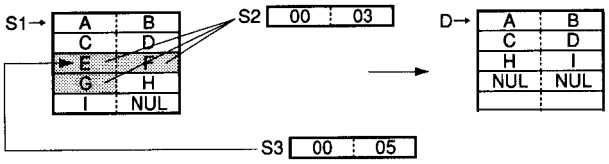
描述

RPLC$(661) 从 S4 指定的起始位置处替换 S1 指定的部分文本字符串，替换的文本字符串由 S2 指定，输出文本字符串数据结果到 D 中。（NULl 添加到末尾）。替换的字符数由 S3 指定。

结果中的最大字符数是 4095（十六进制 0FFF）。如果超出此范围，仅输出 4095 个字符 （NUL 添加到第 4095 个字符处）。

可以替换 0 ～ 4095 个字符 （十六进制 0000 ～ 0FFF）。如果替换数为 0，则 S1 指定的文本字符串原样输出到 D 中。如果 S2 文本字符串是 NUL，则操作等同于删除 S1 文本中的指定区域。

如果 S1 整个文本字符串都被 NUL 代替，则两个 NUL 字符 （十六进制 0000）将输出到 D。



标志

名称	标记	操作
错误标志	ER	如果 S1 和 S2 中指定的字符数超出 4095 个时为 ON。 如果 S3 中指定的字符数超出 4095 个时为 ON。 如果 S4 中的数据在范围 1~4095 之外时为 ON。 其它情况下为 OFF。
等于标志	=	如果 0000 （十六进制）传送到 D 时为 ON。 其它情况下为 OFF。

注意

S1 或 S2 指定的最大字符数是 4095（十六进制 0FFF），如果超出此范围（即第 4095 个字符之前没有 NUL），将会产生错误，错误标志变 ON。

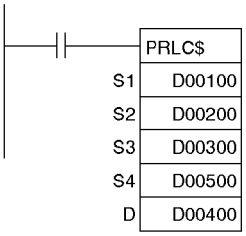
S4 指定的起始位置范围是 1 ～ 4095 个字符，（十六进制 0001 ～ 0FFF）。如果超出此范围，将产生错误，错误标志变 ON。

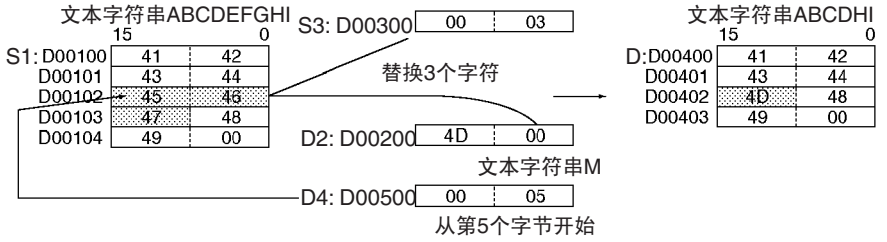
如果 S4 指定的起始位置超出 S1 指定的文本字符串范围，将产生错误，错误标志变 ON。

如果 0000 （十六进制）输出到 D，等于标志将变 ON。
目标首字D不要与替换文本字符串首字S2重叠。否则RPLC$将不能正常工作。

例

该例中， RPLC$(654) 用于替换 3 个字符。





3-33-10 删除字符串 ： DEL$(658)

用途

从文本字符串的中间删除指定的字符串。

梯形图符号

变化

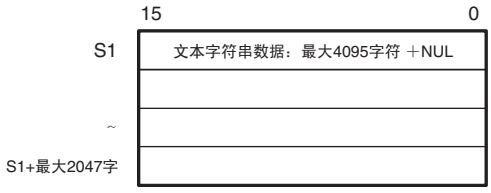
变化	ON 条件时每个循环执行	DEL$(658)
	上升沿微分执行一次	@DEL$(658)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

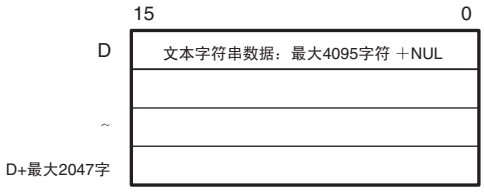
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

S1：文本字符串



S2：字符数 （十六进制 0000 ～ 0FFF 或 &0 ～ &4095）
S3：起始位置 （十六进制 0001 ～ 0FFF&0 ～ &4095）



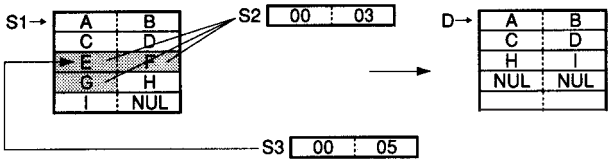
注 1. S1 ～ S1+ 最大 2047 字、S2 ～ S2+ 最大 2047 字和 D ～ D+ 最大 2047 字的数据必须在相同的数据区。
2. S1 ～ S1+ 最大 2047 字和 D ～ D+ 最大 2047 字的数据可以重叠。

操作数规格

区域	S1	S2	S3	D
CIO 区	CIO 0000 ～ CIO 6143			
工作区	W000 ～ W511			
保持位区	H000 ～ H511			
辅助位区	A000 ～ A447 A448 ～ A959			A448 ～ A959
定时器区	T0000 ～ T4095			
计数器区	C0000 ～ C4095			
DM 区	D00000 ～ D32767			
无区号 EM 区	E00000 ～ E32767			
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)			
二进制间接 DM/EM 区	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)			
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 t ～ *En_32767 (n = 0 ～ C)			
常数	---	#0000 ～ #0FFF （二进制）或 &0 ～ &4095	#0001 ～ #0FFF （二进制）或 &1 ～ &4095	---
数据寄存器	---	DR0 ～ DR15		---
索引寄存器	---			
使用索引寄存器接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15			

描述

在 S1 指定的文本字符串范围之内，DEL$(658) 删除的字符数由 S2 指定，起始位置由 S3 指定，结果作为文本字符串输出到 D 中 （NUL 加在末尾）。



标志

名称	标记	操作
错误标志	ER	如果 S1 和 S2 中指定的字符数超出 4095 个时为 ON。 如果 S2 中指定的字符数超出 4095 个时为 ON。 如果 S3 中的数据在范围 1~4095 之外时为 ON 如果 S3 大于 S1 时为 ON。 其它情况下为 OFF。
等于标志	=	如果 0000 （十六进制）传送到 D 时为 ON。 其它情况下为 OFF。

注意

S1 的最大字符数是 4095（十六进制 0FFF）。如果超过此范围（即第 4096 字符之前没有 NUL），将会产生错误，错误标志变 ON。

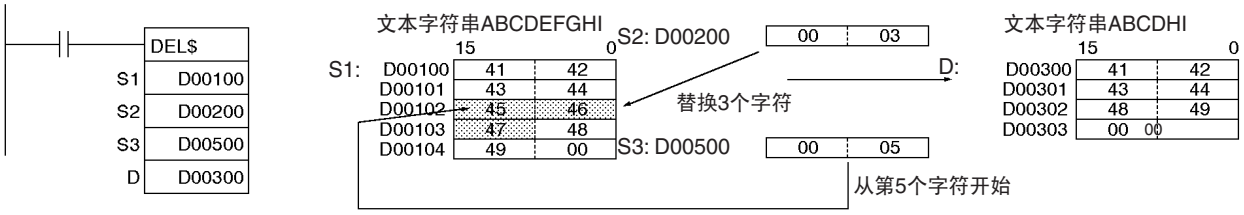
S3 指定的起始位置范围是第 1 ～第 4095 字符（十六进制 0001 ～ 0FFF）。如果超出此范围，将产生错误，错误标志变 ON。

如果 S2 指定的字数超过文本字符串的长度，错误标志变 ON。

如果删除的字符数超出 S1 文本字符串的末尾，一直到结尾的所有字符都将被删除。如果从 S1 开始到结尾的所有字符都被删除，十六进制 0000 将输出到 D 中。

例

该例中， DEL$(658) 用于删除 3 个字符。

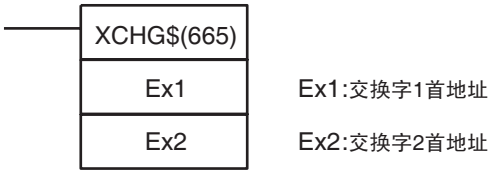


3-33-11 交换字符串：XCHG$(665)

用途

用一个指定的文本字符串替换指定的文本字符串。

梯形图符号



变化

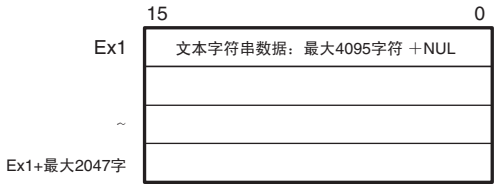
变化	ON 条件时每个循环执行	XCHG$(665)
	上升沿微分执行一次	@XCHG$(665)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

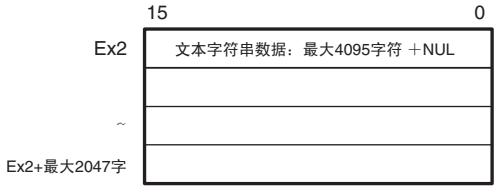
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

Ex1：交换字 1 首地址



Ex2：交换字 2 首地址



注 1. Ex1 ～ Ex1+ 最大 2047 字和 Ex2 ～ Ex2+ 最大 2047 字的数据必须在相同的数据区。

2. Ex1 ～ Ex1+ 最大 2047 字和 Ex2 ～ Ex2+ 最大 2047 字的数据不可重叠。

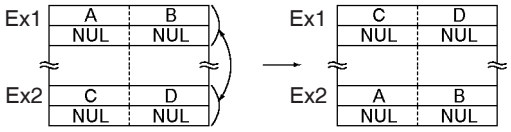
操作数规格

区域	Ex1	Ex2
CIO 区	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W511	
保持位区	H000 ～ H511	
辅助位区	A448 ～ A959	
定时器区	T0000 ～ T4095	
计数器区	C0000 ～ C4095	
DM 区	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 区	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	

区域	Ex1	Ex2
数据寄存器	---	
索引寄存器	---	
使用索引寄存器接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

描述

XCHG$(665) 交换 Ex1 指定的文本字符串和用 Ex2 指定的文本字符串。如果 Ex1 或 EX2 是 NUL，两个 NUL （十六进制 0000）将输出到另一方。



标志

名称	标记	操作
错误标志	ER	如果 Ex1 或 EX2 指定的字符超过 4,095 时为 ON。 Ex1 和 Ex2 重叠时为 ON。 其他情况下为 OFF。

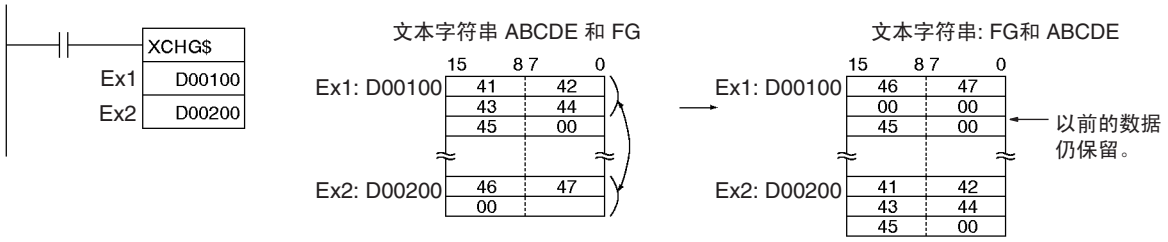
注意

Ex1 或 Ex2 指定的最大字符数是 4095 （十六进制 0FFF hex）。如果超出此范围，将产生错误，错误标志变 ON。

如果 Ex1 和 Ex2 指定的文本字符重叠，将产生错误，错误标志变 ON。

例

该例中， XCHG$(665) 交换两个文本字符串。

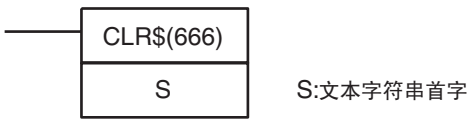


3-33-12 清除字符串：CLR$(666)

用途

用 NUL （十六进制 00）清除整个文本字符串。

梯形图符号



变化

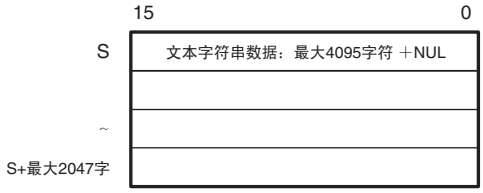
变化	ON 条件时每个循环执行	CLR$(666)
	上升沿微分执行一次	@CLR$(666)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

S：文本字符串首字



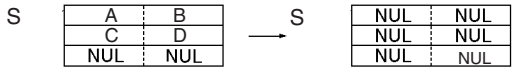
注 S ～ S+ 最大 2047 字的数据必须在相同的数据区。

操作数规格

区域	S
CIO 区	CIO 0000 ～ CIO 6143
工作区	W000 ～ W511
保持位区	H000 ～ H511
辅助位区	A448 ～ A959
定时器区	T0000 ～ T4095
计数器区	C0000 ～ C4095
DM 区	D00000 ～ D32767
无区号 EM 区	E00000 ～ E32767
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)
二进制间接 DM/EM 区	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)
常数	---
数据寄存器	---
索引寄存器	---
使用索引寄存器接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15

描述

CLR$(666) 使用 NUL （十六进制 00）清除 S 指定的首字到 NUL 代码 （十六进制00）之间的全部的文本字符串。所能清除的最大字符数是4096，如果4096字符之前无 NUL，只清除 4096 个字符。

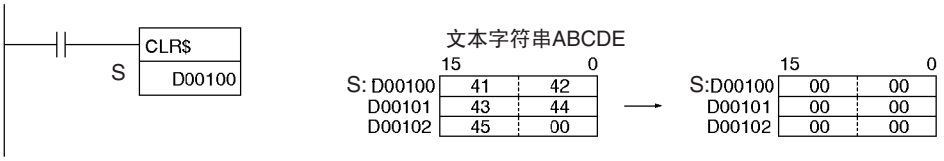


标志

名称	标记	操作
错误标志	ER	OFF

注意 当指令执行时，错误标志变 OFF。

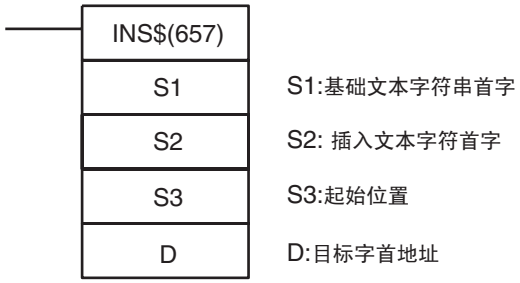
例 该例中， CLR$(666) 清除文本字符串 ABCDE。



3-33-13 插入字符串：INS$(657)

用途 从一个文本字符串的中间插入一指定的文本字符串。

梯形图符号



变化

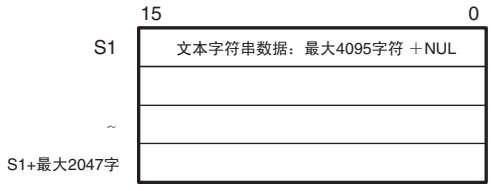
变化	ON 条件时每个循环执行	INS$(657)
	上升沿微分执行一次	@INS$(657)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

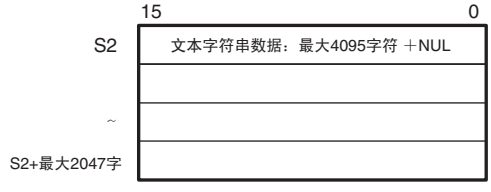
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

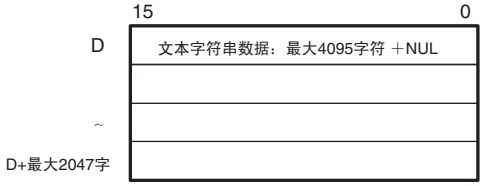
S1：基本文本字符串



S2：插入的文本字符串



S3：起始位置 （十六进制 0001 ～ 0FFF&0 ～ &4095）



注 1. S1 ～ S1+ 最大 2047 字、S2 ～ S2+ 最大 2047 字和 D ～ D+ 最大 2047 字的数据必须在相同的数据区。

2. S1 ～ S1+ 最大 2047 字和 D ～ D+ 最大 2047 字的数据不可重叠。 S1 ～ S1+ 最大 2047 字和 D ～ D+ 最大 2047 字的数据可以重叠，S1 ～ S1+ 最大 2047 字和 S2 ～ S2+ 最大 2047 字的数据也可以重叠。

操作数规格

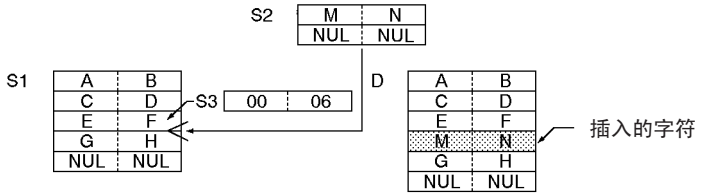
区域	S1	S2	S3	D
CIO 区	CIO 0000 ～ CIO 6143			
工作区	W000 ～ W511			
保持位区	H000 ～ H511			
辅助位区	A000 ～ A447 A448 ～ A959			A448 ～ A959
定时器区	T0000 ～ T4095			
计数器区	C0000 ～ C4095			
DM 区	D00000 ～ D32767			
无区号 EM 区	E00000 ～ E32767			
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)			
二进制间接 DM/EM 区	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)			
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)			
常数	---		#0000 ～ #0FFF （二进制）或 &0 ～ &4095	---
数据寄存器	---		DR0 ～ DR15	---
索引寄存器	---			
使用索引寄存器接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15			

描述

INS$(657) 在 S1 指定的文本字符串范围之内，在 S3 指定的起始字后面插入由 S2 指定的文本字符串，结果以文本字符串形式输出到 D 中（NUL 加在末尾）。

能插入的最大字符数是 4095（十六进制 0FFF），如果超出此范围，只有 4095 个字符输出到 D 中 （NUL 作为第 4096 个字符添加到末尾）。

如果 S1 或 S2 中某一个为 NUL，则另一个的指定的文本字符串原样输出到 D。如果 S1 和 S2 都是 NUL，则两个 NUL 字符输出到 D 中。



标志

名称	标记	操作
错误标志	ER	如果 S1 和 S2 中指定的字符数超出 4095 个时为 ON。 如果 S3 中指定的字符数超出 4095 个时为 ON。 其它情况下为 OFF。
等于标志	=	如果 0000 （十六进制）传送到 D 时为 ON。 其它情况下为 OFF。

注意

S1 和 S2 的最大字符数是 4095 （十六进制 0FFF）。如果超过此范围 （即第 4096 字符之前没有 NUL），将会产生错误，错误标志变 ON。

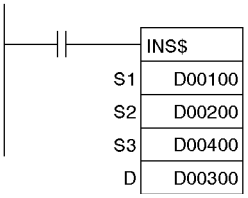
S3 指定的起始位置范围是 0 ～ 4095。如果超出此范围，将产生错误，错误标志变 ON。

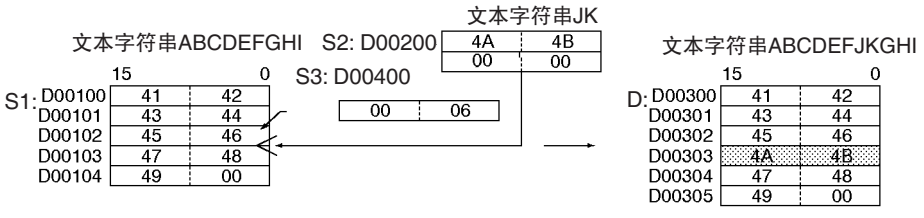
如果 0000 输出到 D 中，等于标志变 ON。

不要把 D 中指定的目标字与 S2 指定的文本字符串数据重叠，否则操作无法正确执行。

例

该例中， INS$(657) 插入两个字符。





3-33-14 字符串比较指令 (670-675)

用途

字符串比较指令（=$、<>$、<$、<=$、>、>=$）根据文本字符串的 ASCII 代码值，从字符的起始处对两个文本字符串进行比较。如果比较的结果为真，将为 LOAD、 AND 或 OR 产生一个 ON 执行条件。

梯形图符号

LD（加载）

AND （串联）

OR（并联）

变化

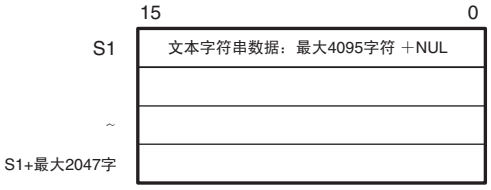
变化	每个循环比较为真时产生 ON	字符串比较指令
立即刷新功能		不支持

适用程序区

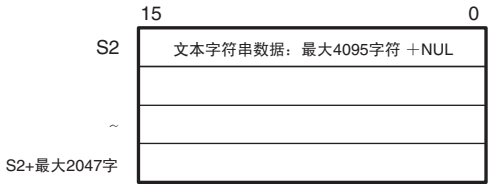
块程序区	步程序区	子程序	中断任务
OK	OK	OK	OK

操作数

S1：文本字符串 1



S2：文本字符串 2



注 1. S1 ～ S1+ 最大 2047 字、 S2 ～ S2+ 最大 2047 字必须在相同的数据区。
2. S1 ～ S1+ 最大 2047 字和 S2 ～ S2+ 最大 2047 字的数据不能重叠。

操作数规格

区域	S1	S2
CIO 区	CIO 0000 ～ CIO 6143	
工作区	W000 ～ W511	
保持位区	H000 ～ H511	
辅助位区	A000 ～ A447 A448 ～ A959	
定时器区	T0000 ～ T4095	
计数器区	C0000 ～ C4095	
DM 区	D00000 ～ D32767	
无区号 EM 区	E00000 ～ E32767	
有区号 EM 区	En_00000 ～ En_32767 (n = 0 ～ C)	
二进制间接 DM/EM 区	@ D00000 ～ @ D32767 @ E00000 ～ @ E32767 @ En_00000 ～ @ En_32767 (n = 0 ～ C)	
BCD 间接 DM/EM 地址	*D00000 ～ *D32767 *E00000 ～ *E32767 *En_00000 ～ *En_32767 (n = 0 ～ C)	
常数	---	
数据寄存器	---	
索引寄存器	---	
使用索引寄存器接寻址	,IR0 ～ ,IR15 –2048 ～ +2047 ,IR0 ～ –2048 ～ +2047 ,IR15 DR0 ～ DR15, IR0 ～ IR15 ,IR0+(++) ～ ,IR15+(++) ,–(– –)IR0 ～ , –(– –)IR15	

描述

字符串比较指令比较 S1 和 S2 中指定的文本字符串。如果比较结果为真，梯形图中产生一个ON执行条件。S1或S2最大的字符数是4095（十六进制0FFF）。

字符串比较指令用 18 个不同的助记符表示，如下所示（LD、AND 和 OR 不出现在梯形图中）。

LD=$,AND=$,OR=$

LD<>$,AND<>$,OR<>$

LD<$,AND<$,OR<$

LD<=$,AND<=$,OR<=$

LD>$,AND>$,OR>$

LD>=$,AND>=$,OR>=$

下表提供了这些指令的详细介绍。

助记符	名称	功能
LD=$(670)	加载字符串相等	当 S1 文本字符串与 S2 文本字符串相等时为真
AND=$(670)	与字符串相等	
OR=$(670)	或字符串相等	
LD<>$(671)	加载字符串不相等	当 S1 文本字符串与 S2 文本字符串不相等时为真
AND<>$(671)	与字符串不相等	
OR<>$(671)	或字符串不相等	
LD<$(672)	加载字符串小于	当 S1 文本字符串小于 S2 文本字符串时为真
AND<$(672)	与字符串小于	
OR<$(672)	或字符串小于	
LD<=$(673)	加载字符串小于等于	当 S1 文本字符串小于或等于 S2 文本字符串时为真
AND<=$(673)	和字符串小于等于	
OR<=$(673)	或字符串小于等于	
LD>$(674)	加载字符串大于	当 S1 文本字符串大于 S2 文本字符串时为真
AND>$(674)	和字符串大于	
OR>$(674)	或字符串大于	
LD>=$(675)	加载字符串大于等于	当 S1 文本字符串大于或等于 S2 文本字符串时为真
AND>=$(675)	和字符串大于等于	
OR>=$(675)	或字符串大于等于	

比较方法

比较方法如下：

每个文本字符串的第一个字符（字节）的 ASCII 代码与另一个字符串的相对应的字符相比较，如果两个字符的 ASCII 代码不相同，则由此确立两个文本字符串的大小关系。如果两个 ASCII 相同，则比较下一个字符。如果这两个 ASCII 码不相同，则由此确立两个文本字符串的大小关系。

以这种方式，两个文本字符串逐个字符进行比较。如果所有的字符包括 NUL 都相同，两个文本字符串具有相等关系。

如果两个文本字符串具有不同的长度，NUL（十六进制 00）将被添加到较短的字符串所余下的长度中，并在此基础上进行比较。

比较举例

AD （十六进制 414400）和 BC （十六进制 424300）：

AD<BC，因在文本字符串的开始， 41 （十六进制）小于 42 （十六进制）。

ADC （十六进制 41444300）和 B （十六进制 4200）：

ADC<B，因在文本字符串的起始处， 41 （十六进制）小于 42 （十六进制）

ABC （41424300）和 ABD （41424400）

ABC<ABD，因为文本字符串的起始处 41S 和 42S 相同，所以结果是 43 小于 44 所决定的。

ABC （十六进制 41424300）和 AB （十六进制 414200）

ABC>AB，因为从在文本字符的开始 41S 和 42S 是相同的，所以结果由 43 大于 00 所决定。

AB （十六进制 414200）和 AB （十六进制 414200）

AB ＝ AB，因为 41S、 42S 和 00S 都相同。

用相同的方法处理 LD、AND 和 OR 指令后继续编程，LD 和 OR 指令可以直接连接在母线上，但 AND 指令不能连接在母线上。

标志

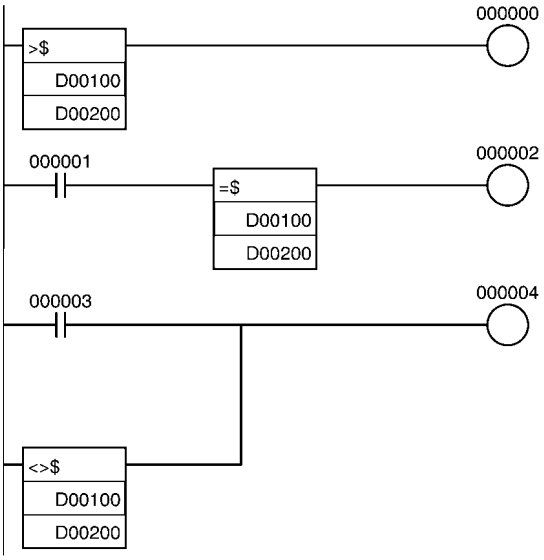
名称	标记	操作
错误标志	ER	如果 S1 或 S2 指定的字符数超过 4095 个时为 ON。 其他情况下为 OFF。
大于标志	>	如果比较结果 S1 大于 S2 时为 ON。 其它情况下为 OFF。
大于等于标志	>=	如果比较结果 S1 大于或等于 S2 时为 ON。 其它情况下为 OFF。
等于标志	=	如果比较结果 S1 等于 S2 时为 ON。 其它情况下为 OFF。
不等于标志	<>	如果比较结果 S1 不等于 S2 时为 ON。 其它情况下为 OFF。
小于标志	<	如果比较结果 S1 小于 S2 时为 ON。 其它情况下为 OFF。
小于等于标志	<=	如果比较结果 S1 小于或等于 S2 时为 ON。 其它情况下为 OFF。

注 字符串比较指令用于把文本字符串按 ASCII 次序进行排序。例如字母 A 至 Z 的 ASCII 码的次序是由低到高，所以文本字符串可以以字母顺序排序。

注意 这些指令之后请使用右侧指令。字符串比较指令不能出现在梯形图的右边。这些指令不能用在逻辑块的最后一条。

能够比较的最大字符数是 4095。如果超出此范围，将产生错误，错误标志变 ON。如果发生此情况， OFF 执行条件将输出到下一条指令中。

例 该例中，字符串比较指令用来比较数据。



地址	助记符	操作
000000	LD > $	--- D00100 D00200
000001	OUT	000000
000002	LD	000001
000003	AND=$	--- D00100 D00200
000004	OUT	000002
000005	LD	000003
000006	OR <> $	--- D00100 D00200
000007	OUT	000004

> $	= $	<>$
D00100	D00100	D00100
D00200	D00200	D00200
ON	OFF	ON

文本字符串ABCD

D00100	41	42
D00101	44	43
D00102	00	00

文本字符串ABC

D00200	41	42
D00201	43	00

文本字符串ABC

D00100	41	42
D00101	43	00

文本字符串ABC

D00200	41	42
D00201	43	00

OFF	ON	OFF

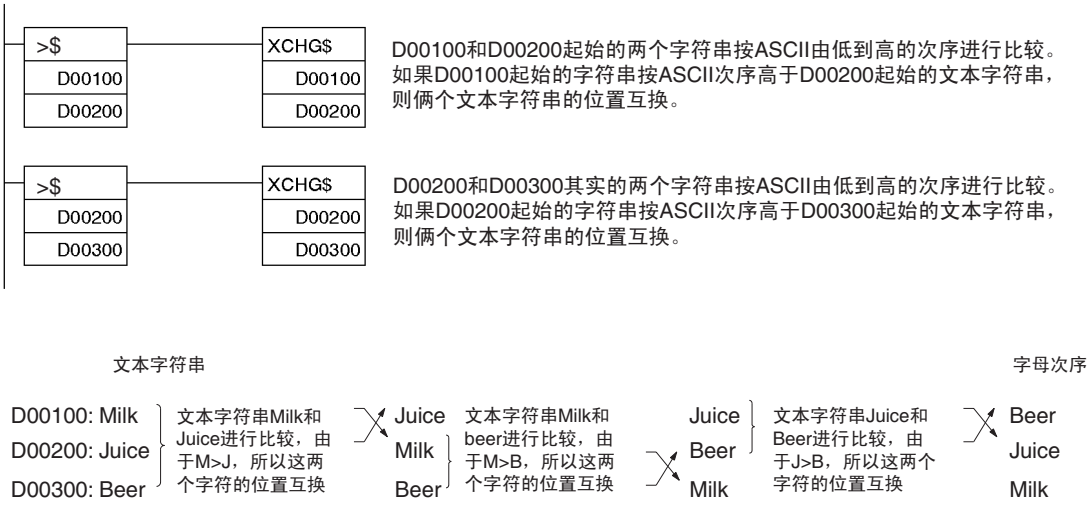
该例中，三个文本字符以字母次序重新排列。原来的次序如下：

D00100：Milk

D00200：Juice

D00300 ； Beer

重排后的次序如下：Bear,Juice,Milk



3-34 任务控制指令

本节描述用于控制的指令。

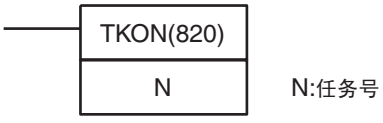
指令	助记符	函数代码	页
TASK ON	TKON	820	1045
TASK OFF	TKOF	821	1049

3-34-1 任务 ON：TKON(820)

用途

使得指定的任务可执行

梯形图符号



变化

变化	ON 条件时每个循环执行	TKON(820)
	上升沿微分执行一次	@TKON(820)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	不允许

操作数

N：任务号

N 的允许范围根据任务的类型指定。

- 循环任务：
  N 必须是十进制 00 和 31（十进制）之间的一个常数。（数值 0 ～ 31 定义任务 0 ～ 31）。

- 扩充循环任务 （仅 CS1-H, CJ1-H 和 CJ1M CPU 单元支持）：
  N必须是8000和8255之间的十进制数。（值8000～8255定义0～255扩充循环任务）。

操作数规格

区域	N
CIO 区	---
工作区	---
保持位区	---
辅助位区	---
定时器区	---
计数器区	---
DM 区	---
无区号 EM 区	---
有区号 EM 区	---
二进制间接 DM/EM 区	---
BCD 间接 DM/EM 地址	---
常数	00 ～ 31 或 8000 ～ 8255 （十进制）
数据寄存器	---
索引寄存器	---
使用索引寄存器接寻址	---

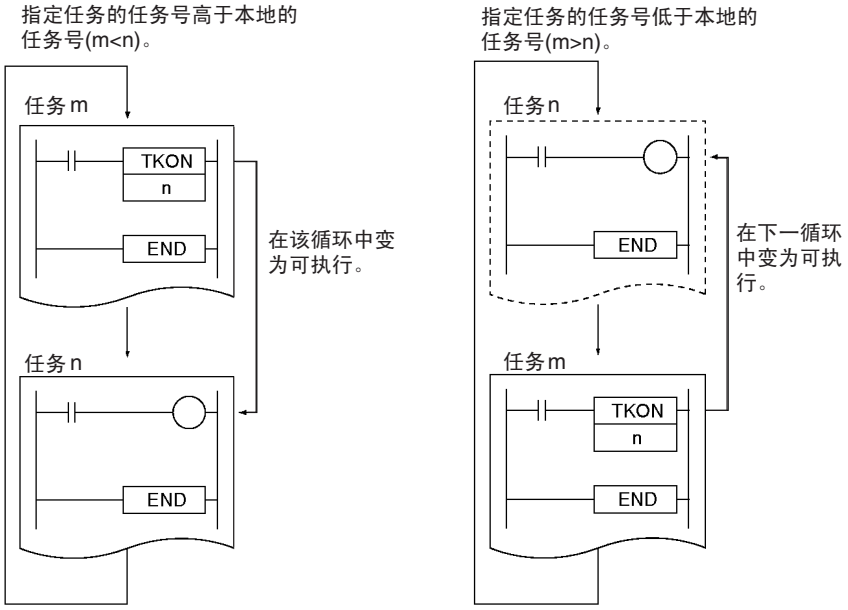
描述

TKON(820) 使指定的任务（常规任务）处于可执行状态，并使相应的任务标志 (TK00 ～ TK31) 置 ON。

该指令只能位于常规 （循环执行）任务中。不能位于扩充循环任务和中断任务中。

由 TKON(820) 指定的循环任务或扩充循环任务只要没有被 TKOF(821) 置为待机状态，在以后的扫描中仍是可执行的。

一个任务能够从任何循环任务中被置于可执行状态。如果该任务的任务号小于本地任务的任务号，指定的任务将从下一个循环开始执行。如果该任务的任务大于本地任务的任务号，该任务在当前的循环执行。



如果指定的任务已经可执行或指定了一个本地任务时，TKON(820)等同于NOP(000)。TKOF(821)、CX-Programmer 或 FINS 命令能够使任务由可执行状态进入待机状态。可执行和正在执行的这两个术语不能互相替换。可执行任务在循环程序执行中以任务号的次序执行。如果程序执行到某一任务号之前该任务已处于待机状态，它将不能执行。

注 1. CX-Programmer 的常规特性表对每一任务有一个设定*（运行开始逻辑框）*，它定义该循环任务在启动时是否可执行。当运行开始逻辑框已被检查，对应的循环任务在 PLC 开始运行时自动地进入可执行状态。其他所有循环任务将进入不可执行状态。 （然而，如果用手持编程器执行内存全清操作，任务 0 将自动地置为可执行状态）。

2. 如果一个任务处于不可执行状态，可执行 TKON(820) 使该任务进入可执行状态。同样，一个处于可执行状态的任务也可用 TKOF(821) 使其进入不可执行状态。

3. 按任务号次序的循环中，已置成可执行的循环任务或扩充循环任务将进入可执行状态。因此，程序执行到某一任务号之前，该任务已处于等待状态，它将不能执行。

标志

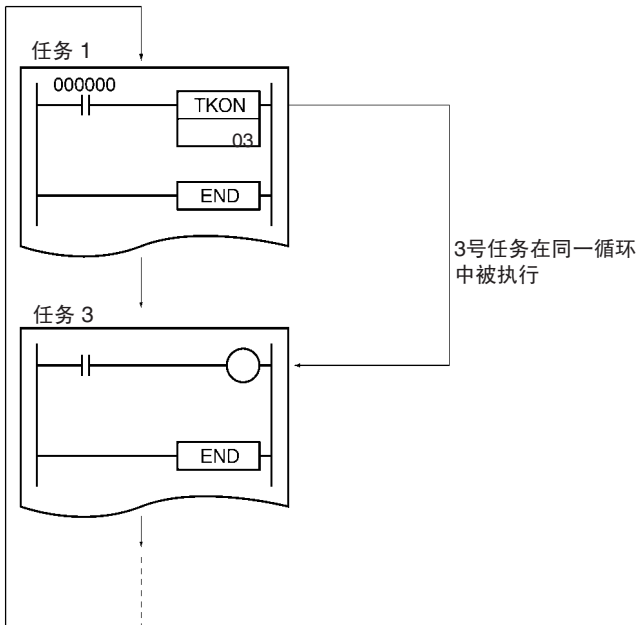
名称	标记	操作
错误标志	ER	如果 N 不是 00 和 31 之间的常数时为 ON。 如果 N 指定的任务不存在时为 ON。 如果在中断任务中执行 TKON （820）时为 ON。 其它情况下为 OFF。

名称	地址	操作
任务标志	TK00 ～ TK31	当相应的任务可执行时这些标志变 ON，当相应的任务不可执行时或处于待机状态时这些标志变 OFF。 TK00 ～ TK31 对应 00 ～ 31 号任务。

例

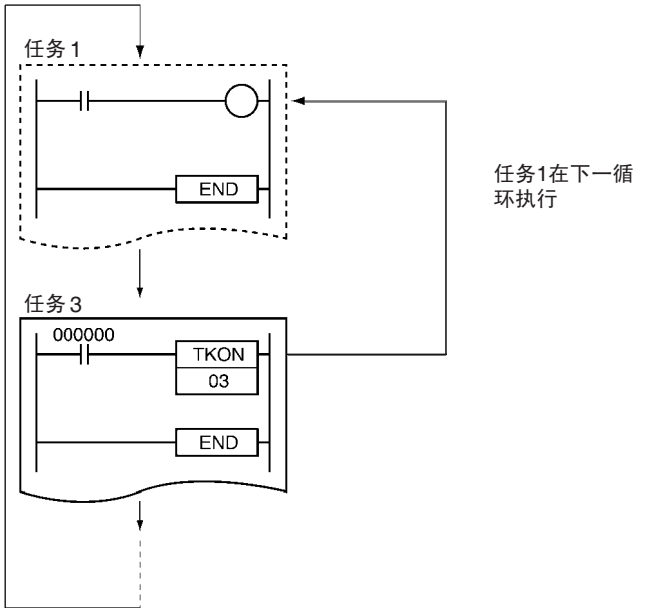
指定一个后面的任务

该例中当 CIO 000000 为 ON 时，在 1 号任务中使得 3 号任务可执行。当程序执行到 3 号任务时， 3 号任务将在同一循环中被执行。



指定一个前面的任务

该例中当 CIO 000000 为 ON 时，在 3 号任务中使得 1 号任务可执行。当程序执行到 1 号任务时，将在下一个循环中执行 1 号任务。

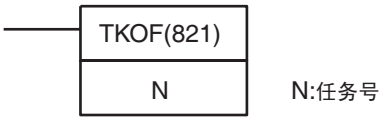


3-34-2 任务 OFF ： TKOF(821)

用途

把指定的任务置为待机状态，即禁止任务的执行。

梯形图符号



变化

变化	ON 条件时每个循环执行	TKOF(821)
	上升沿微分执行一次	@TKOF(821)
	下降沿微分执行一次	不支持
立即刷新功能		不支持

适用程序区

块程序区	步程序区	子程序	中断任务
OK	OK	OK	不允许

操作数

N：任务号

N 的允许范围根据任务的类型指定。

- 循环任务：
  N 必须是十进制 00 和 31（十进制）之间的一个常数。（数值 0 ～ 31 定义任务 0 ～ 31）。
- 扩充循环任务（仅适用于 CS1-H，CJ1-H，和 CJ1M CPU 单元）：N 必须是一个在 8000 和 8255 （十进制）间的常数。（值 8000 ～ 8255 定义扩 充循环任务 0 ～ 255）。

操作数规格

区域	N
CIO 区	---
工作区	---
保持位区	---
辅助位区	---
定时器区	---
计数器区	---
DM 区	---
无区号 EM 区	---
有区号 EM 区	---
二进制间接 DM/EM 区	---
BCD 间接 DM/EM 地址	---
常数	00 ～ 31 或 8000 ～ 8255 （十进制）
数据寄存器	---
索引积存器	---
使用索引寄存器接寻址	---

描述

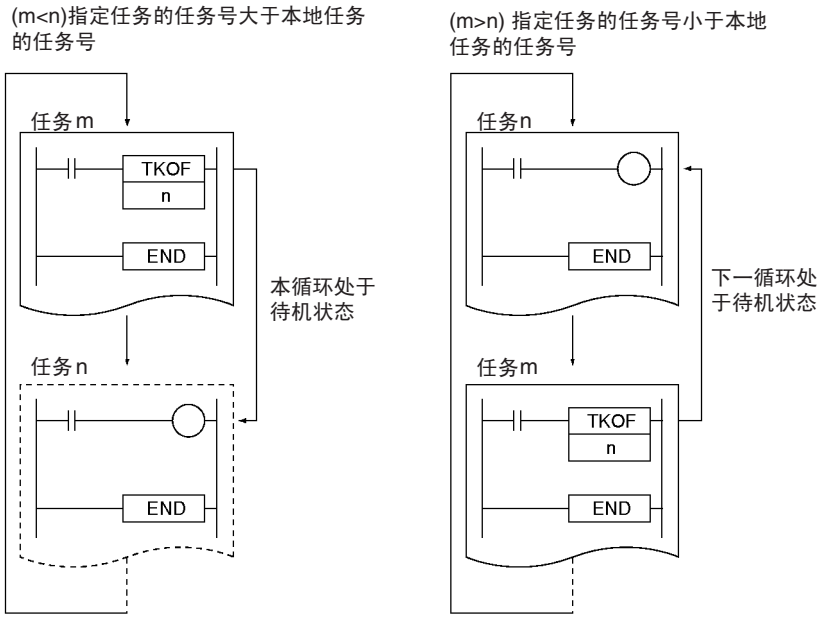
TKOF(821)把指定的循环任务置为待机状态，并把相应的任务标志(TK00～TK31)置 OFF。

TKOF(821) 指定的任务在以后的循环中也处于待机状态，只要 TKON（820），运行 CX-Programmer 的外围设备或 FINS 命令不将它置为可执行状态，如果该任务的任务号小于本任务的任务号，指定的任务将在下一个循环中被置为待机状态。如果该任务的任务号大于本任务的任务号，该任务将在同一循环中被置为待机状态。

如果 TKOF(821) 指定了一个本任务，该任务将立即进入到待机状态，任务中接下来的指令将不执行。

注

1. CX-Programmer 的常规特性表对每一任务有一个设定 *（运行开始逻辑框）*，它定义该循环任务在启动时是否可执行。当运行开始逻辑框已被检查，对应的循环任务在 PLC 开始运行时自动地进入可执行状态。其他所有循环任务将进入不可执行状态。（然而，如果用手持编程器执行内存全清操作，任务 0 将自动地置为可执行状态）。
2. 如果一个任务处于不可执行状态，可执行 TKON(820) 使该任务进入可执行状态。同样，一个处于可执行状态的任务也可用 TKOF(821) 使其进入不可执行状态。
3. 已处于可执行状态的循环任务或扩充循环任务可由 TKOF(821) 指令使其处于待机状态。



设置成启动时执行的常规任务在 PC 开始运行时就自动处于可执行状态，而其它的常规任务都处于不可执行状态。

TKOF(821)、运行 CX-Programmer 的外部设备或 FINS 命令能够把可执行状态的任务设置为待机状态。

可执行和正在执行这两个术语是不能互相混淆。可执行任务在循环程序执行中以任务号的次序执行。如果程序执行到某一任务号之前该任务被至于待机状态，它将不被执行。

与 TKON(820) 不同，该指令可位于中断任务和循环任务中。

标志

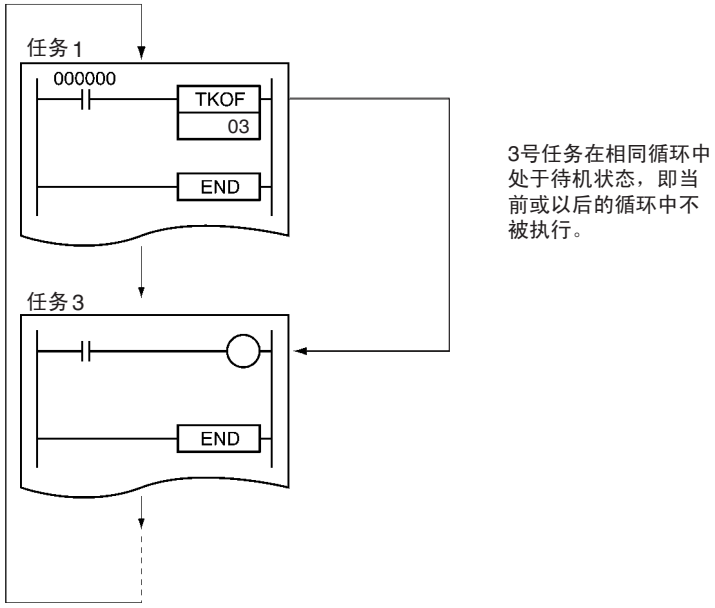
名称	标记	操作
错误标志	ER	如果 N 不是 00 和 31 的常数或不是 8000 和 8255 之间的常数（仅 CS1-H,CJ1-H,CJ1M CPU 单元）时为 ON。如果 N 指定的任务不存在时为 ON。 其它情况下为 OFF。

名称	地址	操作
任务标志	TK00 ～ TK31	当相应的任务可执行时这些标志变 ON，当相应的任务不可执行时或处于待机状态时这些标志变 OFF。 TK00 ～ TK31 对应 00 ～ 31 号任务。

例

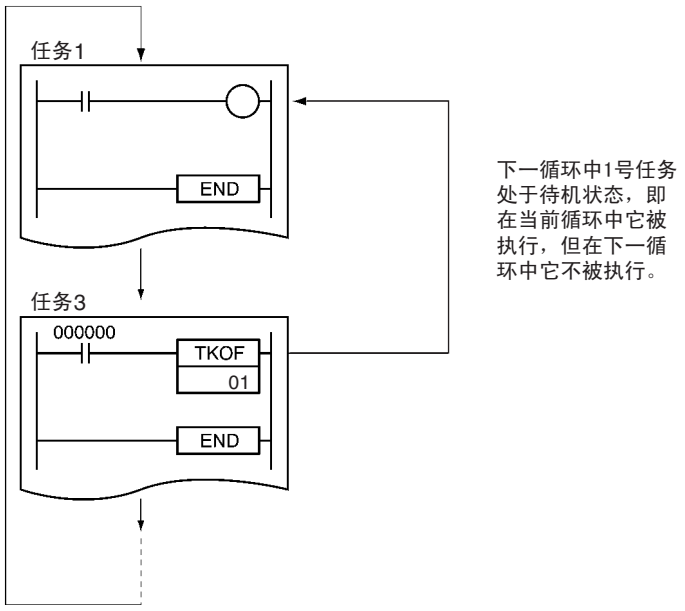
指定一个后面的任务

该例中当 CIO 000000 为 ON，在 1 号任务中把 3 号任务置为待机状态。本循环中程序执行到 3 号任务时， 3 号任务不被执行。



指定一个前面的任务

该例中当 CIO 000000 为 ON 时，在 3 号任务中将 1 号任务置为待机状态。当下一循环中程序执行到 1 号任务时， 1 号任务将不被执行。



第 4 章
指令执行时间和步数

本章提供了 CS/CJ 系列每条指令执行时间和步数。

4-1 CS 系列指令执行时间和步数

下表列举了 CS 系列 PLC 所有可用指令的执行时间。

整个用户程序中指令总的执行时间是计算循环时间时所指的程序执行处理时间（见注）。

注 用户程序是分配给循环任务中可执行任务和满足中断条件的中断任务。

根据使用的 CPU 单元（CS1 □ -CPU6 □ H， CS1 □ -CPU6 □， CS1 □ -CPU4 □ H， CS1 □ -CPU4 □和 CS1D-CPU6 □）不同以及指令执行的条件不同，大多数指令执行时间是不同的。下表中每一条指令上面的一行显示最小时间以及必要的执行条件。下面一行显示最大时间以及必要的执行条件。

当执行条件是 OFF 时，执行时间也会变化。

下表长度 *（步长）*列中显示出各指令的长度。 CS 系列每一条指令所需的步数从 1 ～ 7 步不同，根据指令和它使用的操作数来决定。程序中的步数不等同于指令数。

注 1. CS 系列 PLC 程序容量以步衡量。以前的 OMRON PLC，例如 C 系列和 CV 系列 PLC 里以字来决定程序容量的。一般地， 1 步相当于 1 字。但是对于某些 CS 系列指令来说，每一指令所需的内存容量是不同的。如果将另一 PLC 的用户程序转换为 CS 系列 PLC 时，假定 1 步是 1 字，则可能发生误差。*参考 4-1CS 系列指令执行时间和步数*最后有关转换以前 OMRON PLC 程序容量规则。

大多数指令支持微分形式（用↑ ↓ @ 和 % 表示）。指令指定微分形式将增加下列指令执行时间。

符号	CS1-H CPU 单元		CS1 CPU 单元		CS1D CPU 单元
	CPU6@H	CPU4@H	CPU6@	CPU4@	CPU6@H
↑或↓	+0.24	+0.32	+0.41	+0.45	+0.24
@ 或 %	+0.24	+0.32	+0.29	+0.33	+0.24

2. 当指令不执行时，取下列时间作为指标。

CS1-H CPU 单元		CS1 CPU 单元		CS1D CPU 单元
CPU6@H	CPU4@H	CPU6@	CPU4@	CPU6@H
约 0.1	约 0.2	约 0.1 ～ 0.3	约 0.2 ～ 0.4	约 0.1

4-1-1 顺序输入指令

指令	助记符	代码	长度（步）	ON 执行时间 (µs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
加载	LD	---	1	0.02	0.04	0.04	0.08	---
	!LD（见注 2）	---	2	+21.14	+21.16	+21.16	+21.16	CS 系例增加
				+45.1	+45.1	+45.1	+45.1	C200H 系例增加（见注 3）
加载非	LD NOT	---	1	0.02	0.04	0.04	008	---
	!LD NOT（见注 2）	---	2	+21.14	+21.16	+21.16	+21.16	CS 系例增加
				+45.1	+45.1	+45.1	+45.1	C200H 系例增加（见注 3）
与	AND	---	1	0.02	0.04	0.04	0.08	---
	!AND（见注 2）	---	2	+21.14	+21.16	+21.16	+21.16	CS 系例增加
				+45.1	+45.1	+45.1	+45.1	C200H 系例增加（见注 3）
与非	AND NOT	---	1	0.02	0.04	0.04	0.08	---
	!AND NOT（见注 2）	---	2	+21.14	+21.16	+21.16	+21.16	CS 系例增加
				+45.1	+45.1	+45.1	+45.1	C200H 系例增加（见注 3）
或	OR	---	1	0.02	0.04	0.04	0.08	---
	!OR（见注 2）	---	2	+21.14	+21.16	+21.16	+21.16	CS 系例增加
				+45.1	+45.1	+45.1	+45.1	C200H 系例增加（见注 3）
或非	OR NOT	---	1	0.02	0.04	0.04	0.08	---
	!OR NOT（见注 2）	---	2	+21.14	+21.16	+21.16	+21.16	CS 系例增加
				+45.1	+45.1	+45.1	+45.1	C200H 系例增加（见注 3）
与加载	AND LD	---	1	0.02	0.04	0.04	0.08	---
或加载	OR LD	---	1	0.02	0.04	0.04	0.08	---
非	NOT	520	1	0.02	0.04	0.04	0.08	---
条件 ON	UP	521	3	0.3	0.42	0.46	0.54	---
条件 OFF	DOWN	522	4	0.3	0.42	0.46	0.54	---
加载位测试	LD TST	350	4	0.14	0.24	0.25	0.37	---
加载位测试非	LD TSTN	351	4	0.14	0.24	0.25	0.37	---
与位测试非	AND TSTN	351	4	0.14	0.24	0.25	0.37	---
或位测试	OR TST	350	4	0.14	0.24	0.25	0.37	---
或位测试非	OR TSTN	351	4	0.14	0.24	0.25	0.37	---

注 1. 当使用双字长度的操作数时，将下表长度列的值加 1。
2. CS1D CPU 单元不支持立即刷新功能。
3. CS1D CPU 单元不支持。

4-1-2 顺序输出指令

指令	助记符	代码	长度 （步）（见注）	ON 执行时间 (µs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
输出	OUT	---	1	0.02	0.04	0.17	0.21	---
	!OUT（见注 2）	---	2	+21.37	+21.37	+21.37	+21.37	CS 系例增加
				+49.3	+49.3	+49.3	+49.3	C200H 系例增加（见注 3）
输出非	OUT NOT	---	1	0.02	0.04	0.17	0.21	---
	!OUT NOT（见注 2）	---	2	+21.37	+21.37	+21.37	+21.37	CS 系例增加
				+49.3	+49.3	+49.3	+49.3	C200H 系例增加（见注 3）
保持	KEEP	011	1	0.06	0.08	0.25	0.29	---
上升沿微分	DIFU	013	2	0.24	0.40	0.46	0.54	---
下降沿微分	DIFD	014	2	0.24	0.40	0.46	0.54	---
置位	SET	---	1	0.02	0.06	0.17	0.21	---
	!SET（见注 2）	---	2	+21.37	+21.37	+21.37	+21.37	CS 系例增加
				+49.3	+49.3	+49.3	+49.3	C200H 系例增加（见注 3）
复位	RSET	---	1	0.02	0.06	0.17	0.21	特殊字
	!RSET（见注 2）	---	2	+21.37	+21.37	+21.37	+21.37	CS 系例增加
				+49.3	+49.3	+49.3	+49.3	C200H 系例增加（见注 3）
多位置位	SETA	530	4	5.8	6.1	7.8	7.8	置 1 位
				25.7	27.2	38.8	38.8	置 1,000 位
多位复位	RSTA	531	4	5.7	6.1	7.8	7.8	复 1 位
				25.8	27.1	38.8	38.8	复 1,000 位
单位置位	SETB	532	2	0.24	0.34	---	---	---
	!SETB（见注 2）		3	+21.44	+21.54	---	---	---
单位复位	RSTB	534	2	0.24	0.34	---	---	---
	!RSTB（见注 2）		3	+21.44	+21.54	---	---	---
单位输出	OUTB	534	2	0.22	0.32	---	---	---
	!OUTB（见注 2）		3	+21.42	+21.52	---	---	---

注 1. 当使用双字长度的操作数时，将下表长度列的值加 1。
2. CS1D CPU 单元不支持立即刷新功能。
3. CS1D CPU 单元不支持。

4-1-3 顺序控制指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
结束	END	001	1	5.5	6.0	4.0	4.0	---
空操作	NOP	000	1	0.02	0.04	0.08	0.12	---
联锁	IL	002	1	0.06	0.06	0.12	0.12	---
联锁解除	ILC	003	1	0.06	0.06	0.12	0.12	---
跳转	JMP	004	2	0.38	0.48	8.1	8.1	---
跳转结束	JME	005	2	---	---	---	---	---
条件跳转	CJP	510	2	0.38	0.48	7.4	7.4	当 JMP 条件满足
条件跳转非	CJPN	511	2	0.38	0.48	8.5	8.5	当 JMP 条件满足
多路跳转	JMP0	515	1	0.06	0.06	0.12	0.12	---
多路跳转结束	JME0	516	1	0.06	0.06	0.12	0.12	---
FOR 循环	FOR	512	2	0.52	0.54	0.12	0.21	指定一个常数
中断循环	BREAK	514	1	0.06	0.06	0.12	0.12	---
NEXT 循环	NEXT	513	1	0.18	0.16	0.17	0.17	当循环继续
				0.22	0.40	0.12	0.12	当循环结束

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-1-4 定时器和计数器指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
定时器	TIM	---	3	0.56	0.88	0.37	0.42	---
	TIMX	550	3	0.56	0.88	---	---	---
计数器	CNT	---	3	0.56	0.88	0.37	0.42	---
	CNTX	546	3	0.56	0.88	---	---	---
高速定时器	TIMH	015	3	0.88	1.14	0.37	0.42	---
	TIMHX	551	3	0.88	1.14	---	---	---
1ms 定时器	TMHH	540	3	0.86	1.12	0.37	0.42	---
	TMHHX	552	3	0.86	1.12	---	---	---

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
累加定时器	TTIM	087	3	16.1	17.0	21.4	21.4	---
				10.9	11.4	14.8	14.8	当复位时
				8.5	8.7	10.7	10.7	当互锁时
	TTIMX	555	3	16.1	17.0	---	---	---
				10.9	11.4	---	---	当复位时
				8.5	8.7	---	---	当互锁时
长时间定时器	TIML	542	4	7.6	10.0	12.8	12.8	---
				6.2	6.5	7.8	7.8	当互锁时
	TIMLX	553	4	7.6	10.0	---	---	---
				6.2	6.5	---	---	当互锁时
多路输出定时器	MTIM	543	4	20.9	23.3	26.0	26.0	---
				5.6	5.8	7.8	7.8	当复位时
	MTIMX	554	4	20.9	23.3	---	---	---
				5.6	5.8	---	---	当复位时
可逆计数器	CNTR	012	3	16.9	19.0	20.9	20.9	---
	CNTRX	548	3	16.9	19.0	---	---	---
复位定时器 / 计数器	CNR	545	3	9.9	10.6	13.9	13.9	当复位 1 个字时
				4.16 ms	4.16 ms	5.42 ms	5.42 ms	当复位 1,000 个字时
	CNRX	547	3	9.9	10.6	---	---	当复位 1 个字时
				4.16 ms	4.16 ms	---	---	当复位 1,000 个字时

注　当使用双字长度的操作数时，将下表长度列的值加 1。

4-1-5　比较指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
输入比较指令（无符号）	LD, AND, OR +=	300	4	0.10	0.16	0.21	0.37	---
	LD, AND, OR + <>	305						
	LD, AND, OR + <	310						
	LD, AND, OR +<=	315						
	LD, AND, OR +>	320						
	LD, AND, OR +>=	325						

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (µs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
输入比较指令（无符号双字）	LD, AND, OR +=+L	301	4	0.10	0.16	0.29	0.54	---
	LD, AND, OR +<>+L	306						
	LD, AND, OR +<+L	311						
	LD, AND, OR +<=+L	316						
	LD, AND, OR +>+L	321						
	LD, AND, OR +>=+L	326						
输入比较指令（带符号）	LD, AND, OR +=+S	302	4	0.10	0.16	6.50	6.50	---
	LD, AND, OR +<>+S	307						
	LD, AND, OR +<+S	312						
	LD, AND, OR +<=	317						
	LD, AND, OR +>+S	322						
	LD, AND, OR +>=+S	327						
输入比较指令（带符号双字）	LD, AND, OR +=+SL	303	4	0.10	0.16	6.50	6.50	---
	LD, AND, OR +<>+SL	308						
	LD, AND, OR +<+SL	313						
	LD, AND, OR +<=+SL	318						
	LD, AND, OR +>+SL	323						
	LD, AND, OR +>=+SL	328						
比较	CMP	020	3	0.04	0.04	0.17	0.29	---
	!CMP（见注 2）	020	7	+42.1	+42.1	+42.4	+42.4	CS 系例增加
				+90.4	+90.4	+90.5	+90.5	C200H 系例增加（见注 3）
双字比较	CMPL	060	3	0.08	0.08	0.25	0.46	---
带符号二进制比较	CPS	114	3	0.08	0.08	6.50	6.50	---
	!CPS（见注 2）	114	7	+35.9	+35.9	+42.4	+42.4	CS 系例增加
				+84.1	+84.1	+90.5	+90.5	C200H 系例增加
带符号双字二进制比较	CPSL	115	3	0.08	0.08	6.50	6.50	---

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
表格比较	TCMP	085	4	14.0	15.2	21.9	21.9	---
多路比较	MCMP	019	4	20.5	22.8	31.2	31.2	---
无符号块比较	BCMP	068	4	21.5	23.7	32.6	32.6	---
区域比较	ZCP	088	3	5.3	5.4	---	---	---
双字区域比较	ZCPL	116	3	5.5	6.7	---	---	---

注 1. 当使用双字长度的操作数时，将下表长度列的值加 1。

2. CS1D CPU 单元不支持立即刷新功能。

3. CS1D CPU 单元不支持。

4-1-6 数据传送指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
传送	MOV	021	3	0.18	0.20	0.25	0.29	---
	!MOV（见注 2）	021	7	+21.38	+21.40	+42.36	+42.36	CS 系例增加
				+90.52	+90.52	+90.52	+90.52	C200H 系例增加（见注 3）
双字传送	MOVL	498	3	0.32	0.34	0.42	0.50	---
传送非	MVN	022	3	0.18	0.20	0.25	0.29	---
双字传送非	MVNL	499	3	0.32	0.34	0.42	0.50	---
位传送	MOVB	082	4	0.24	0.34	7.5	7.5	---
数字传送	MOVD	083	4	0.24	0.34	7.3	7.3	---
多位传送	XFRB	062	4	10.1	10.8	13.6	13.6	传送 1 位
				186.4	189.8	269.2	269.2	传送 255 位
块传送	XFER	070	4	0.36	0.44	11.2	11.2	传送 1 个字
				300.1	380.1	633.5	633.5	传送 1,000 个字
块设置	BSET	071	4	0.26	0.28	8.5	8.5	设置 1 个字
				200.1	220.1	278.3	278.3	设置 1,000 个字
数据交换	XCHG	073	3	0.40	0.56	0.5	0.7	---
双字数据交换	XCGL	562	3	0.76	1.04	0.9	1.3	---

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
单字分配	DIST	080	4	5.1	5.4	7.0	7.0	---
数据收集	COLL	081	4	5.1	5.3	7.1	7.1	---
传送到寄存器	MOVR	560	3	0.08	0.08	0.42	0.50	---
把定时器 / 计数器的 PV 值传送到寄存器	MOVRW	561	3	0.42	0.50	0.42	0.50	---

注 1. 当使用双字长度的操作数时，将下表长度列的值加 1。
2. CS1D CPU 单元不支持立即刷新功能。
3. CS1D CPU 单元不支持。

4-1-7 数据移位指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
移位寄存器	SFT	010	3	7.4	10.4	10.4	10.4	移位 1 个字
				433.2	488.0	763.1	763.1	移位 1,000 个字
可逆移位寄存器	SFTR	084	4	6.9	7.2	9.6	9.6	移位 1 个字
				615.3	680.2	859.6	859.6	移位 1,000 个字
异步移寄存器	ASFT	017	4	6.2	6.4	7.7	7.7	移位 1 个字
				1.22 ms	1.22 ms	2.01 ms	2.01 ms	移位 1,000 个字
字移位	WSFT	016	4	4.5	4.7	7.8	7.8	移位 1 个字
				171.5	171.7	781.7	781.7	移位 1,000 个字
算术左移	ASL	025	2	0.22	0.32	0.29	0.37	---
双字左移	ASLL	570	2	0.40	0.56	0.50	0.67	---
算术右移	ASR	026	2	0.22	0.32	0.29	0.37	---
双字右移	ASRL	571	2	0.40	0.56	0.50	0.67	---
循环左移	ROL	027	2	0.22	0.32	0.29	0.37	---
双字循环左移	ROLL	572	2	0.40	0.56	0.50	0.67	---
无进位循环左移	RLNC	574	2	0.22	0.32	0.29	0.37	---
无进位双字循环左移	RLNL	576	2	0.40	0.56	0.50	0.67	---
循环右移	ROR	028	2	0.22	0.32	0.29	0.37	---

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
双字循环右移	RORL	573	2	0.40	0.56	0.50	0.67	---
无进位循环右移动	RRNC	575	2	0.22	0.32	0.29	0.37	---
无进位双字循环右移	RRNL	577	2	0.40	0.56	0.50	0.67	---
一个数字左移	SLD	074	3	5.9	6.1	8.2	8.2	移位 1 个字
				561.1	626.3	760.7	760.7	移位 1,000 个字
一个数字右移	SRD	075	3	6.9	7.1	8.7	8.7	移位 1 个字
				760.5	895.5	1.07 ms	1.07 ms	移位 1,000 个字
左移 N 位数据	NSFL	578	4	7.5	8.3	10.5	10.5	移位 1 位
				40.3	45.4	55.5	55.5	移位 1,000 位
右移 N 位数据	NSFR	579	4	7.5	8.3	10.5	10.5	移位 1 位
				50.5	55.3	69.3	69.3	移位 1,000 位
左移 N 位	NASL	580	3	0.22	0.32	0.29	0.37	---
双字左移 N 位	NSLL	582	3	0.40	0.56	0.50	0.67	---
右移 N 位	NASR	581	3	0.22	0.32	0.29	0.37	---
双字右移 N 位	NSRL	583	3	0.40	0.56	0.50	0.67	---

注　当使用双字长度的操作数时，将下表长度列的值加 1。

4-1-8　递增 / 递减指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
二进制递增	++	590	2	0.22	0.32	0.29	0.37	---
双字二进制递增	++L	591	2	0.40	0.56	0.50	0.67	---
二进制递减	– –	592	2	0.22	0.32	0.29	0.37	---
双字二进制递减增	– –L	593	2	0.40	0.56	0.50	0.67	---
BCD 递增	++B	594	2	6.4	4.5	7.4	7.4	---
双字 BCD 递增	++BL	595	2	5.6	4.9	6.1	6.1	---

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
BCD 递减	– –B	596	2	6.3	4.6	7.2	7.2	---
双字 BCD 递减	– –BL	597	2	5.3	4.7	7.1	7.1	---

注　当使用双字长度的操作数时，将下表长度列的值加 1。

4-1-9　四则运算指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
无进位带符号二进制加法	+	400	4	0.18	0.20	0.25	0.37	---
无进位带符号双字二进制加法	+L	401	4	0.32	0.34	0.42	0.54	---
有进位带符号二进制加法	+C	402	4	0.18	0.20	0.25	0.37	---
有进位带符号双字二进制加法	+CL	403	4	0.32	0.34	0.42	0.54	---
无进位 BCD 加法	+B	404	4	8.2	8.4	14.0	14.0	---
无进位双字 BCD 加法	+BL	405	4	13.3	14.5	19.0	19.0	---
有进位 BCD 加法	+BC	406	4	8.9	9.1	14.5	14.5	---
有进位双字 BCD 加法	+BCL	407	4	13.8	15.0	19.6	19.6	---
无进位带符号二进制减法	–	410	4	0.18	0.20	0.25	0.37	---
无进位双字带符号二进制减法	–L	411	4	0.32	0.34	0.42	0.54	---

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
有进位带符号二进制减法	–C	412	4	0.18	0.20	0.25	0.37	---
有进位带符号双字二进制减法	–CL	413	4	0.32	0.34	0.42	0.54	---
无进位 BCD 减法	–B	414	4	8.0	8.2	13.1	13.1	---
无进位双字 BCD 减法	–BL	415	4	12.8	14.0	18.2	18.2	---
有进位 BCD 减法	–BC	416	4	8.5	8.6	13.8	13.8	---
有进位双字 BCD 减法	–BCL	417	4	13.4	14.7	18.8	18.8	---
带符号二进制乘法	*	420	4	0.38	0.40	0.50	0.58	---
带符号双字二进制乘法	*L	421	4	7.23	8.45	11.19	11.19	---
不带符号二进制乘法	*U	422	4	0.38	0.40	0.50	0.58	---
不带符号双字二进制乘法	*UL	423	4	7.1	8.3	10.63	10.63	---
BCD 乘法	*B	424	4	9.0	9.2	12.8	12.8	---
双字 BCD 乘法	*BL	425	4	23.0	24.2	35.2	35.2	---
带符号二进制除法	/	430	4	0.40	0.42	0.75	0.83	---
带符号双字二进制除法	/L	431	4	7.2	8.4	9.8	9.8	---
不带符号二进制除法	/U	432	4	0.40	0.42	0.75	0.83	---

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
不带符号双字二进制除法	/UL	433	4	6.9	8.1	9.1	9.1	---
BCD 除法	/B	434	4	8.6	8.8	15.9	15.9	---
双字 BCD 除法	/BL	435	4	17.7	18.9	26.2	26.2	---

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-1-10 转换指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
BCD →二进制	BIN	023	3	0.22	0.24	0.25	0.29	---
双字 BCD →双字二进制	BINL	058	3	6.5	6.8	9.1	9.1	---
二进制→BCD	BCD	024	3	0.24	0.26	8.3	8.3	---
双字二进制→双字 BCD	BCDL	059	3	6.7	7.0	9.2	9.2	---
二进制求补	NEG	160	3	0.18	0.20	0.25	0.29	---
双字二进制求补	NEGL	161	3	0.32	0.34	0.42	0.5	---
带符号二进制 16 位→32 位	SIGN	600	3	0.32	0.34	0.42	0.50	---
数据译码	MLPX	076	4	0.32	0.42	8.8	8.8	1 位数字译码（4 ～ 6）
				0.98	1.20	12.8	12.8	4 位数字译码（4 ～ 6）
				3.30	4.00	20.3	20.3	1 位数字译码（8 ～ 256）
				6.50	7.90	33.4	33.4	2 位数字译码（8 ～ 256）
数据编码	DMPX	077	4	7.5	7.9	10.4	10.4	1 位数字编码（16 ～ 4）
				49.6	50.2	59.1	59.1	4 位数字编码（16 ～ 4）
				18.2	18.6	23.6	23.6	1 位数字编码（256 ～ 8）
				55.1	57.4	92.5	92.5	2 位数字编码（256 ～ 8）

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
ASCII 转换	ASC	086	4	6.8	7.1	9.7	9.7	1 位数字转换到 ASCII
				11.2	11.7	15.1	15.1	4 位数字转换到 ASCII
ASCII 码→十六进制	HEX	162	4	7.1	7.4	10.1	10.1	转换到 1 位数字
列→行	LINE	063	4	19.0	23.1	29.1	29.1	---
行→列	COLM	064	4	23.2	27.5	37.3	37.3	---
带符号 BCD →二进制	BINS	470	4	8.0	8.3	12.1	12.1	数据格式设置 0
				8.0	8.3	12.1	12.1	数据格式设置 1
				8.3	8.6	12.7	12.7	数据格式设置 2
				8.5	8.8	13.0	13.0	数据格式设置 3
带符号双字 BCD →二进制	BISL	472	4	9.2	9.6	13.6	13.6	数据格式设置 0
				9.2	9.6	13.7	13.7	数据格式设置 1
				9.5	9.9	14.2	14.2	数据格式设置 2
				9.6	10.0	14.4	14.4	数据格式设置 3
带符号二进制→ BCD	BCDS	471	4	6.6	6.9	10.6	10.6	数据格式设置 0
				6.7	7.0	10.8	10.8	数据格式设置 1
				6.8	7.1	10.9	10.9	数据格式设置 2
				7.2	7.5	11.5	11.5	数据格式设置 3
带符号双字二进制→ BCD	BDSL	473	4	8.1	8.4	11.6	11.6	数据格式设置 0
				8.2	8.6	11.8	11.8	数据格式设置 1
				8.3	8.7	12.0	12.0	数据格式设置 2
				8.8	9.2	12.5	12.5	数据格式设置 3

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-1-11 逻辑指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
逻辑与	ANDW	034	4	0.18	0.20	0.25	0.37	---
双字逻辑与	ANDL	610	4	0.32	0.34	0.42	0.54	---
逻辑或	ORW	035	4	0.22	0.32	0.25	0.37	---
双字逻辑或	ORWL	611	4	0.32	0.34	0.42	0.54	---
异或	XORW	036	4	0.22	0.32	0.25	0.37	---
双字异或	XORL	612	4	0.32	0.34	0.42	0.54	---
异或非	XNRW	037	4	0.22	0.32	0.25	0.37	---
双字异或非	XNRL	613	4	0.32	0.34	0.42	0.54	---
求反	COM	029	2	0.22	0.32	0.29	0.37	---
双字求反	COML	614	2	0.40	0.56	0.50	0.67	---

注　当使用双字长度的操作数时，将下表长度列的值加 1。

4-1-12 特殊数学指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
二进制平方根	ROTB	620	3	49.6	50.0	530.7	530.7	---
BCD 平方根	ROOT	072	3	13.7	13.9	514.5	514.5	---
算术处理	APR	069	4	6.7	6.9	32.3	32.3	指定 SIN 和 COS
				17.2	18.4	78.3	78.3	指定线段逼近
浮点除法	FDIV	079	4	116.6	176.6	176.6	176.6	---
位计数器	BCNT	067	4	0.3	0.38	22.1	22.1	计数 1 个字

注　当使用双字长度的操作数时，将下表长度列的值加 1。

4-1-13 浮点数学指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (µs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
浮点→ 16 位	FIX	450	3	10.6	10.8	14.5	14.5	---
浮点→ 32 位	FIXL	451	3	10.8	11.0	14.6	14.6	---
16 位→浮点	FLT	452	3	8.3	8.5	11.1	11.1	---
32 位→浮点	FLTL	453	3	8.3	8.5	10.8	10.8	---
浮点加法	+F	454	4	8.0	9.2	10.2	10.2	---
浮点减法	–F	455	4	8.0	9.2	10.3	10.3	---
浮点乘法	/F	457	4	8.7	9.9	12.0	12.0	---
浮点除法	*F	456	4	8.0	9.2	10.5	10.5	---
度→弧度	RAD	458	3	10.1	10.2	14.9	14.9	---
弧度→度	DEG	459	3	9.9	10.1	14.8	14.8	---
正弦	SIN	460	3	42.0	42.2	61.1	61.1	---
余弦	COS	461	3	31.5	31.8	44.1	44.1	---
正切	TAN	462	3	16.3	16.6	22.6	22.6	---
反正弦	ASIN	463	3	17.6	17.9	24.1	24.1	---
反余弦	ACOS	464	3	20.4	20.7	28.0	28.0	---
反正切	ATAN	465	3	16.1	16.4	16.4	16.4	---
平方根	SQRT	466	3	19.0	19.3	28.1	28.1	---
指数	EXP	467	3	65.9	66.2	96.7	96.7	---
对数	LOG	468	3	12.8	13.1	17.4	17.4	---
指数幂	PWR	840	4	125.4	126.0	181.7	181.7	---
单精度浮点比较指令	LD, AND, OR +=F	329	3	6.6	8.3	---	---	---
	LD, AND, OR +<>F	330						
	LD, AND, OR +<F	331						
	LD, AND, OR +<=F	332						
	LD, AND, OR +>F	333						
	LD, AND, OR +>=F	334						

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
浮点→ ASCII	FSTR	448	4	48.5	48.9	---	---	---
ASCII →浮点	FVAL	449	3	21.1	21.3	---	---	---

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-1-14 双精度浮点指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
双精度浮点比较指令	LD, AND, OR +=D	335	3	8.5	10.3	---	---	---
	LD, AND, OR +<>D	336						
	LD, AND, OR +<D	337						
	LD, AND, OR +<=D	338						
	LD, AND, OR +>D	339						
	LD, AND, OR +>=D	340						
双字浮点→ 16 位	FIXD	841	3	11.7	12.1	---	---	---
双字浮点→ 32 位	FIXLD	842	3	11.6	12.1	---	---	---
16 位→双字浮点	DBL	843	3	9.9	10.0	---	---	---
32 位→双字浮点	DBLL	844	3	9.8	10.0	---	---	---
双字浮点加法	+D	845	4	11.2	11.9	---	---	---
双字浮点减法	–D	846	4	11.2	11.9	---	---	---
双字浮点乘法	*D	847	4	12.0	12.7	---	---	---
双字浮点除法	/D	848	4	23.5	24.2	---	---	---

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
双字度→弧度	RADD	849	3	27.4	27.8	---	---	---
双字弧度→度	DEGD	850	3	11.2	11.9	---	---	---
双字正弦	SIND	851	3	45.4	45.8	---	---	---
双字余弦	COSD	852	3	43.0	43.4	---	---	---
双字正切	TAND	853	3	20.1	20.5	---	---	---
双字反正弦	ASIND	854	3	21.5	21.9	---	---	---
双字反余弦	ACOSD	855	3	24.7	25.1	---	---	---
双字反正切	ATAND	856	3	19.3	19.7	---	---	---
双字平方根	SQRTD	857	3	47.4	47.9	---	---	---
双字指数	EXPD	858	3	121.0	121.4	---	---	---
双字对数	LOGD	859	3	16.0	16.4	---	---	---
双字指数幂	PWRD	860	4	223.9	224.2	---	---	---

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-1-15 表格数据处理指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
设置堆栈	SSET	630	3	8.0	8.3	8.5	8.5	堆栈区指定 5 个字
				231.6	251.8	276.8	276.8	堆栈区指定 1,000 个字
推入堆栈	PUSH	632	3	6.5	8.6	9.1	9.1	---
先进先出	FIFO	633	3	6.9	8.9	10.6	10.6	堆栈区指定 5 个字
				352.6	434.3	1.13 ms	1.13 ms	堆栈区指定 1,000 个字
后进先出	LIFO	634	3	7.0	9.0	9.9	9.9	---
定维记录表	DIM	631	5	15.2	21.6	142.1	142.1	---

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
设置记录位置	SETR	635	4	5.4	5.9	7.0	7.0	---
获取记录号	GETR	636	4	7.8	8.4	11.0	11.0	---
数据搜索	SRCH	181	4	15.5	19.5	19.5	19.5	搜索 1 个字
				2.42 ms	3.34 ms	3.34 ms	3.34 ms	搜索 1,000 个字
字节交换	SWAP	637	3	12.2	13.6	13.6	13.6	交换 1 个字
				1.94 ms	2.82 ms	2.82 ms	2.82 ms	交换 1,000 个字
寻找最大值	MAX	182	4	19.2	24.9	24.9	24.9	搜索 1 个字
				2.39 ms	3.36 ms	3.36 ms	3.36 ms	搜索 1,000 个字
寻找最小值	MIN	183	4	19.2	25.3	25.3	25.3	搜索 1 个字
				2.39 ms	3.33 ms	3.33 ms	3.33 ms	搜索 1,000 个字
求和	SUM	184	4	28.2	38.5	38.5	38.3	1 字相加
				1.42 ms	1.95 ms	1.95 ms	1.95 ms	1,000 字相加
帧校验和	FCS	180	4	20.0	28.3	28.3	28.3	表长 1 字
				1.65 ms	2.48 ms	2.48 ms	2.48 ms	表长 1,000 字
读堆栈大小	SNUM	638	3	6.0	6.3	---	---	---
读堆栈数据	SREAD	639	4	8.0	8.4	---	---	---
写堆栈数据	SWRIT	640	4	7.2	7.6	---	---	---
插入堆栈数据	SINS	641	4	7.8	9.9	---	---	---
				354.0	434.8	---	---	1,000 字表
删除堆栈数据	SDEL	642	4	8.6	10.6	---	---	---
				354.0	436.0	---	---	1,000 字表

注　当使用双字长度的操作数时，将下表长度列的值加 1。

4-1-16 数据控制指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
PID 控制	PID	190	4	436.2（见注 2）	678.2	678.2	678.2	初始化执行
				332.3（见注 3）	474.9	474.9	474.9	采样
				97.3（见注 4）	141.3	141.3	141.3	不采样
限位控制	LMT	680	4	16.1	22.1	22.1	22.1	---
静带控制	BAND	681	4	17.0	22.5	22.5	22.5	---
静域控制	ZONE	682	4	15.4	20.5	20.5	20.5	---
标度	SCL	194	4	37.1	53.0	56.8	56.8	---
标度 2	SCL2	486	4	28.5	40.2	50.7	50.7	---
标度 3	SCL3	487	4	33.4	47.0	57.7	57.7	---
求平均值	AVG	195	4	36.3	52.6	53.1	53.1	1 个操作的平均值
				291.0	419.9	419.9	419.9	64 个操作的平均值
自动 PID 控制	PIDAT	191	4	446.3	712.5	---	---	初始化执行
				339.4	533.9	---	---	采样
				100.7	147.1	---	---	不采样
				189.2	281.6	---	---	自动的初始化执行
				535.2	709.8	---	---	采样时自动

注 1. 当使用双字长度的操作数时，将下表长度列的值加 1。

2. CS1D CPU 单元：单机模式下 436.2μs，双机模式下 676.2μs。

3. CS1D CPU 单元：单机模式下 332.3μs，双机模式下 572.3μs。

4. CS1D CPU 单元：单机模式下 97.3μs，双机模式下 337.3μs。

4-1-17 子程序指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
子程序调用	SBS	091	2	1.26	1.96	17.0	17.0	---
子程序入口	SBN	092	2	---	---	---	---	---
子程序返回	RET	093	1	0.86	1.60	20.60	20.60	---
宏指令	MCRO	099	4	23.3	23.3	23.3	23.3	---
进入全局子程序	GSBN	751	2	---	---	---	---	---

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
全局子程序返回	GRET	752	1	1.26	1.96	---	---	---
调用全局子程序	GSBS	750	2	0.86	1.60	---	---	---

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-1-18 中断控制指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
设置中断屏蔽	MSKS（见注 2）	690	3	25.6	38.4	39.5	39.5	---
读中断屏蔽	MSKR（见注 2）	692	3	11.9	11.9	11.9	11.9	---
清除中断	CLI（见注 2）	691	3	27.4	41.3	41.3	41.3	---
禁止中断	DI	693	1	15.0	16.8	16.8	16.8	---
允许中断	EI	694	1	19.5	21.8	21.8	21.8	---

注 1. 当使用双字长度的操作数时，将下表长度列的值加 1。

2. CS1D CPU 单元不支持这些指令。

4-1-19 步指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
步定义	STEP	008	2	17.4	20.7	27.1	27.1	步控制位 ON
				11.8	13.7	24.4	24.4	步控制位 OFF
步启动	SNXT	009	2	6.6	7.3	10.0	10.0	---

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-1-20 基本 I/O 单元指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
I/O 刷新	IORF	097	3	58.5	63.2	81.7	81.7	C200H 基本 I/O 单元刷新 1 字（输入）
				62.6	67.0	86.7	86.7	C200H 基本 I/O 单元刷新 1 字（输出）
				15.5（见注 2）	16.4	23.5	23.5	CS 系列基本 I/O 单元刷新 1 字（输入）
				17.20（见注 3）	18.40	25.6	25.6	CS 系列基本 I/O 单元刷新 1 字（输出）
				303.3	343.9	357.1	357.1	C200H 基本 I/O 单元刷新 10 字（输入）
				348.2	376.6	407.5	407.5	C200H 基本 I/O 单元刷新 10 字（输出）
				319.9（见注 4）	320.7	377.5	377.6	CS 系列基本 I/O 单元刷新 60 字（输入）
				358.00（见注 5）	354.40	460.1	460.1	CS 系列基本 I/O 单元刷新 60 字（输出）
7 段译码	SDEC	078	4	6.5	6.9	14.1	14.1	---
智能 I/O 读	IORD	222	4	读 / 写时间取决于正在执行该指令的特殊 I/O 单元				---
智能 I/O 写	IOWR	223	4					---
CPU 总线单元 I/O 刷新	DLNK	226	4	287.8	315.5	---	---	分配 1 个字

注 1. 当使用双字长度的操作数时，将下表长度列的值加 1。

2. CS1D CPU 单元：单机模式下 15.5μs，双机模式下 255.5μs。

3. CS1D CPU 单元：单机模式下 17.2μs，双机模式下 257.2μs。

4. CS1D CPU 单元：单机模式下 319.9μs，双机模式下 559.9μs。

5. CS1D CPU 单元：单机模式下 358.0μs，双机模式下 598.0μs。

4-1-21 串行通信命令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
协议宏	PMCR	260	5	100.1	142.1	276.8	276.8	发送 0 字，接收 0 字
				134.2	189.6	305.9	305.9	发送 249 字，接收 249 字
传送	TXD	236	4	68.5	98.8	98.8	98.8	发送 1 字节
				734.3	1.10 ms	1.10 ms	1.10 ms	发送 256 字节
接收	RXD	235	4	89.6（见注 2）	131.1	131.1	131.1	存储 1 字节
				724.2（见注 3）	1.11 ms	1.11 ms	1.11 ms	存储 256 字节
修改串行口设置	STUP	237	3	341.2	400.0	440.4	440.4	---

注 1. 当使用双字长度的操作数时，将下表长度列的值加 1。

2. CS1D CPU 单元：单机模式下 89.6μs，双机模式下 329.6μs。

3. CS1D CPU 单元：单机模式下 724.2μs，双机模式下 964.2 μs。

4-1-22 网络指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
网络发送	SEND	090	4	84.4	123.9	123.9	123.9	---
网络接收	RECV	098	4	85.4	124.7	124.7	124.7	---
发布命令	CMND	490	4	106.8	136.8	136.8	136.8	---

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-1-23 文件存储指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
读数据文件	FREAD	700	5	391.4（见注 2）	632.4	684.1	684.1	2 字符目录 + 文件名二进制
				836.1（见注 3）	1.33 ms	1.35 ms	1.35 ms	73 字符目录 + 文件名二进制

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
写数据文件	FWRIT	701	5	387.8（见注 4）	627.0	684.7	684.7	2 字符目录 + 文件名二进制
				833.3（见注 5）	1.32 ms	1.36 ms	1.36 ms	73 字符目录 + 文件名二进制

注 1. 当使用双字长度的操作数时，将下表长度列的值加 1。
2. CS1D CPU 单元：单机模式下 391.4μs，双机模式下 631.4μs。
3. CS1D CPU 单元：单机模式下 836.1μs，双机模式下 1,076.1μs。
4. CS1D CPU 单元：单机模式下 382.8μs，双机模式下 627.8μs。
5. CS1D CPU 单元：单机模式下 833.5μs，双机模式下 1,073.3μs。

4-1-24 显示指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
显示信息	MSG	046	3	10.1	14.2	14.3	14.3	显示信息
				8.4	11.3	11.3	11.3	删除显示的信息

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-1-25 时钟指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
日历加法	CADD	730	4	38.3	201.9	209.5	209.5	---
日历减法	CSUB	731	4	38.6	170.4	184.1	184.1	---
小时→秒	SEC	065	3	21.4	29.3	35.8	35.8	---
秒→小时	HMS	066	3	22.2	30.9	42.1	42.1	---
时钟调整	DATE	735	2	60.5	87.4	95.9	95.9	---

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-1-26 调试指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
跟踪存储采样	TRSM	045	1	80.4	120.0	120.0	120.0	采样 1 位和 0 字
				848.1	1.06 ms	1.06 ms	1.06 ms	采样 31 位和 6 字

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-1-27 故障诊断指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
故障报警	FAL	006	3	15.4	16.7	16.7	16.7	记录错误
				179.8	244.8	244.8	244.8	按优先次序删除错误
				432.4	657.1	657.1	657.1	删除全部错误
				161.5	219.4	219.4	219.4	删除单个错误
严重故障报警	FALS	007	3	---	---	---	---	---
故障点检测	FPD	269	4	140.9	202.3	202.3	202.3	当执行时
				163.4	217.6	217.6	217.6	第一次
				185.2	268.9	268.9	268.9	当执行时
				207.5	283.6	283.6	283.6	第一次

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-1-28 其它命令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
设置进位	STC	040	1	0.06	0.06	0.12	0.12	---
清除进位	CLC	041	1	0.06	0.06	0.12	0.12	---
设置 EM 块	EMBC	281	2	14.0	15.1	15.1	15.1	---
延长最大循环时间	WDT	094	2	15.0	19.7	19.7	19.7	---
存入条件标志	CCS	282	1	8.6	12.5	---	---	---
载入条件标志	CCL	283	1	9.8	13.9	---	---	---
从 CV 转变地址	FRMCV	284	3	13.6	19.9	---	---	---
转变地址到 CV	TOCV	285	3	11.9	17.2	---	---	---

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
外设服务禁止	IOSP（见注 2）	287	---	13.9	19.8	---	---	---
外围服务允许	IORS（见注 2）	288	---	63.6	92.3	---	---	---

注 1. 当使用双字长度的操作数时，将下表长度列的值加 1。

2. CS1， CJ1 或 CS1D CPU 单元不支持这些指令。

4-1-29 块程序指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
块程序开始	BPRG	096	2	12.1	13.0	13.0	13.0	---
块程序结束	BEND	801	1	9.6	12.3	13.1	13.1	---
块程序暂停	BPPS	811	2	10.6	12.3	14.9	14.9	---
块程序重新启动	BPRS	812	2	5.1	5.6	8.3	8.3	---
条件块退出	（执行条件）EXIT	806	1	10.0	11.3	12.9	12.9	退出条件满足
				4.0	4.9	7.3	7.3	退出条件不满足
条件块退出	EXIT（位地址）	806	2	6.8	13.5	16.3	16.3	退出条件满足
				4.7	7.2	10.7	10.7	退出条件不满足
条件块退出（非）	EXIT NOT（位地址）	806	2	12.4	14.0	16.8	16.8	退出条件满足
				7.1	7.6	11.2	11.2	退出条件不满足
分支	IF（执行条件）	802	1	4.6	4.8	7.2	7.2	如果真
				6.7	7.3	10.9	10.9	如果假
分支	IF（继电器号）	802	2	6.8	7.2	10.4	10.4	如果真
				9.0	9.6	14.2	14.2	如果假
分支 （非）	IF NOT（继电器号）	802	2	7.1	7.6	10.9	10.9	如果真
				9.2	10.1	14.7	14.7	如果假
分支	ELSE	803	1	6.2	6.7	9.9	9.9	如果真
				6.8	7.7	11.2	11.2	如果假
分支	IEND	804	1	6.9	7.7	11.0	11.0	如果真
				4.4	4.6	7.0	7.0	如果假

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
一个循环与等待	WAIT（执行条件）	805	1	12.6	13.7	16.7	16.7	等待条件满足
				3.9	4.1	6.3	6.3	等待条件不满足
一个循环与等待	WAIT（执行条件）	805	2	12.0	13.4	16.5	16.5	等待条件满足
				6.1	6.5	9.6	9.6	等待条件不满足
一个循环与等待（非）	WAIT NOT（继电器号）	805	2	12.2	13.8	17.0	17.0	等待条件满足
				6.4	6.9	10.1	10.1	等待条件不满足
计数器等待	CNTW	814	4	17.9	22.6	27.4	27.4	缺省设置
				19.1	23.9	28.7	28.7	正常执行
	CNTWX	818	4	17.9	22.6	---	---	缺省设置
				19.1	23.9	---	---	正常执行
高速定时器等待	TMHW	815	3	25.8	27.9	34.1	34.1	缺省设置
				20.6	22.7	28.9	28.9	正常执行
	TMHWX	817	3	25.8	27.9	---	---	缺省设置
				20.6	22.7	---	---	正常执行
				9.3	10.8	---	---	LEND条件不满足
循环控制	LOOP	809	1	7.9	9.1	12.3	12.3	---
循环控制	LEND（执行条件）	810	1	7.7	8.4	10.9	10.9	LEND 条件满足
				6.8	8.0	9.8	9.8	LEND条件不满足
循环控制	LEND（继电器号）	810	2	9.9	10.7	14.4	14.4	LEND 条件满足
				8.9	10.3	13.0	13.0	LEND条件不满足
循环控制	LEND NOT（继电器号）	810	2	10.2	11.2	14.8	14.8	LEND 条件满足
				9.3	10.8	13.5	13.5	LEND条件不满足
定时器等待	TIMW	813	3	22.3	25.2	33.1	33.1	缺省设置
				24.9	27.8	35.7	35.7	正常执行
	TIMWX	816	3	22.3	25.2	---	---	缺省设置
				24.9	27.8	---	---	正常执行

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-1-30 文本串处理指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (µs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
串传送	MOV$	664	3	45.6	66.0	84.3	84.3	传送 1 个字符
串链接	+$	656	4	86.5	126.0	167.8	167.8	1 个字符 +1 个字符
取左串	LEFT$	652	4	53.0	77.4	94.3	94.3	从 2 个字符中获取 1 个字符
取右串	RGHT$	653	4	52.2	76.3	94.2	94.2	从 2 个字符中获取 1 个字符
取中间串	MID$	654	5	56.5	84.6	230.2	230.2	从 3 个字符中获取 1 个字符
寻找串	FIND$	660	4	51.4	77.5	94.1	94.1	从 2 个字符中搜索 1 个字符
串长度	LEN$	650	3	19.8	28.9	33.4	33.4	检测 1 个字符
替换串	RPLC$	661	6	175.1	258.7	479.5	479.5	用一个字符替换 2 个字符中的第一个字符
删除串	DEL$	658	5	63.4	94.2	244.6	244.6	删除 2 个字符中前面的字符
交换串	XCHG$	665	3	60.6	87.2	99.0	99.0	1 对 1 交换字符
清除串	CLR$	666	2	23.8	36.0	37.8	37.8	清除 1 个字符
插入串	INS$	657	5	136.5	200.6	428.9	428.9	在 2 个字符首字符之后插入 1 个字符
串比较指令	LD, AND, OR +=$	670	4	48.5	69.8	86.2	86.2	1 个字符与另一个字符比较
	LD, AND, OR +<>$	671						
	LD, AND, OR +<$	672						
	LD, AND, OR +>$	674						
	LD, AND, OR +>=$	675						

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-1-31 任务控制指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU6@	CPU4@	
任务 ON	TKON	820	2	19.5	26.3	26.3	26.3	---
任务 OFF	TKOF	821	2	13.3	19.0	26.3	26.3	---

从以前的 OMRON PLC 转换程序容量的规则

下表提供了从以前的 OMRON PLC （SYSMAC C200HX/HG/HE，CVM1 或 CV 系列 PLC）的程序容量（单位：字）转换到 CS 系列 PLC 的程序容量（单位：步）的规则。

为了获得 CS 系列 PLC 的程序容量 （单位：步），在以前 PLC 每条指令的程序容量 （单位：字）上加上值 (n)。

CS1 步 = 以前 PC 的 “a”（字） +n			
指令	变化	C200HX/HG/HE 转换到 CS 系列时的 n 值	CV 系列 PLC 或 CVM1 转换到 CS 系列时的 n 值
基本指令	无	OUT, SET, RSET, 或 KEEP(011): –1 其它指令：0	0
	上升微分	无	+1
	立即刷新	无	0
	上升微分和立即刷新	无	+2
特殊指令	无	0	–1
	上升微分	+1	0
	立即刷新	无	+3
	上升微分和立即刷新	无	+4

例如，如果使用了一个地址为 CIO 000000 ～ CIO 25515 的 OUT，以前的 PC 的程序容量每个指令为 2 个字，而 CS 系列 PC 每指令是 1(2-1) 步。

例如，如果使用了 !MOV 指令（立即刷新的 MOVE 指令）， CV 系列的程序容量每个指令为 4 个字，而 CS 系列 PC 是 7(4+3) 步。

4-2 CJ 系列指令执行时间和步数

下表列举了 CJ 系列 PLC 所有可用指令的执行时间。

整个用户程序中指令总的执行时间是计算循环时间时所指的程序执行处理时间（见注）。

注 用户程序是分配给循环任务中可执行任务和满足中断条件的中断任务。

根据使用的 CPU 单元（CJ1H-CPU6 □ H，CJ1H-CPU4 □ H，CJ1M-CPU □ □和 CJ1G-CPU4 □ ）不同以及指令执行的条件不同，大多数指令执行时间是不同的。下表中每一条指令上面的一行显示最小时间以及必要的执行条件。下面一行显示最大时间以及必要的执行条件。

当执行条件是 OFF 时，执行时间也会变化。

下表长度 *（步长）*列中显示出各指令的长度。 CJ 系列每一条指令所需的步数从 1 ～ 7 步不同，根据指令和它使用的操作数来决定。程序中的步数不等同于指令数。

注 1. CJ 系列 PLC 程序容量以步衡量。以前的 OMRON PLC，例如 C 系列和 CV 系列 PLC 里以字来决定程序容量的。一般地， 1 步相当于 1 字。但是对于某些 CJ 系列指令来说，每一指令所需的内存容量是不同的。如果将另一 PLC 的用户程序转换为 CJ 系列 PLC 时，假定 1 步是 1 字，则可能发生误差。参考 *4-1CS 系列指令执行时间和步数*最后有关转换以前 OMRON PLC 程序容量规则。

2. 大多数指令支持微分形式（用 ↑ ↓ @ 和 % 表示）。指令指定微分形式将增加下列指令执行时间。

符号	CJ1-H		CJ1M	CJ1
	CPU6@H	CPU4@H	CPU@@	CPU4@
↑或↓	+0.24 μs	+0.32 μs	+0.5 μs	+0.45 μs
@ 或 %	+0.24 μs	+0.32 μs	+0.5 μs	+0.33 μs

3. 当指令不执行时，取下列时间作为规则。

CJ1-H		CJ1M	CJ1
CPU6@H	CPU4@H	CPU@@	CPU4@
约 0.1 μs	约 0.2 μs	约 0.2 ～ 0.5 μs	约 0.2 ～ 0.4 μs

4-2-1 顺序输入指令

指令	助记符	代码	长度（步）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
加载	LD	---	1	0.02	0.04	0.08	0.10	---
	!LD	---	2	+21.14	+21.16	+21.16	+24.10	增加立即刷新
加载非	LD NOT	---	1	0.02	0.04	008	0.10	---
	!LD NOT	---	2	+21.14	+21.16	+21.16	+24.10	增加立即刷新
与	AND	---	1	0.02	0.04	0.08	0.10	---
	!AND	---	2	+21.14	+21.16	+21.16	+24.10	增加立即刷新
与非	AND NOT	---	1	0.02	0.04	0.08	0.10	---
	!AND NOT	---	2	+21.14	+21.16	+21.16	+24.10	增加立即刷新
或	OR	---	1	0.02	0.04	0.08	0.10	---
	!OR	---	2	+21.14	+21.16	+21.16	+24.10	增加立即刷新
或非	OR NOT	---	1	0.02	0.04	0.08	0.10	---
	!OR NOT	---	2	+21.14	+21.16	+21.16	+24.10	增加立即刷新
与加载	AND LD	---	1	0.02	0.04	0.08	0.05	---
或加载	OR LD	---	1	0.02	0.04	0.08	0.05	---
非	NOT	520	1	0.02	0.04	0.08	0.05	---
条件 ON	UP	521	3	0.3	0.42	0.54	0.50	---
条件 OFF	DOWN	522	4	0.3	0.42	0.54	0.50	---
加载位测试	LD TST	350	4	0.14	0.24	0.37	0.35	---
加载位测试非	LD TSTN	351	4	0.14	0.24	0.37	0.35	---
与位测试非	AND TSTN	351	4	0.14	0.24	0.37	0.35	---
或位测试	OR TST	350	4	0.14	0.24	0.37	0.35	---
或位测试非	OR TSTN	351	4	0.14	0.24	0.37	0.35	---

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-2 顺序输出指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
输出	OUT	---	1	0.02	0.04	0.21	0.35	---
	!OUT	---	2	+21.37	+21.37	+21.37	+23.07	增加立即刷新
输出非	OUT NOT	---	1	0.02	0.04	0.21	0.35	---
	!OUT NOT	---	2	+21.37	+21.37	+21.37	+23.07	增加立即刷新
保持	KEEP	11	1	0.06	0.08	0.29	0.40	---
上升沿微分	DIFU	13	2	0.24	0.40	0.54	0.50	---
下降沿微分	DIFD	14	2	0.24	0.40	0.54	0.50	---
置位	SET	---	1	0.02	0.06	0.21	0.30	---
	!SET	---	2	+21.37	+21.37	+21.37	+23.17	增加立即刷新
复位	RSET	---	1	0.02	0.06	0.21	0.30	特殊字
	!RSET	---	2	+21.37	+21.37	+21.37	+23.17	增加立即刷新
多位置位	SETA	530	4	5.8	6.1	7.8	11.8	置 1 位
				25.7	27.2	38.8	64.1	置 1,000 位
多位复位	RSTA	531	4	5.7	6.1	7.8	11.8	复 1 位
				25.8	27.1	38.8	64.0	复 1,000 位
单位置位	SETB	532	2	0.24	0.34	---	0.5	---
	!SETB		3	+21.44	+21.54	---	+23.31	---
单位复位	RSTB	533	2	0.24	0.34	---	0.5	---
	!RSTB		3	+21.44	+21.54	---	+23.31	---
单位输出	OUTB	534	2	0.22	0.32	---	0.45	---
	!OUTB		3	+21.42	+21.52	---	+23.22	---

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-3 顺序控制指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
结束	END	1	1	5.5	6.0	4.0	7.9	---
空操作	NOP	0	1	0.02	0.04	0.12	0.05	---
联锁	IL	2	1	0.06	0.06	0.12	0.15	---
联锁解除	ILC	3	1	0.06	0.06	0.12	0.15	---
跳转	JMP	4	2	0.38	0.48	8.1	0.95	---
跳转结束	JME	5	2	---	---	---	---	---
条件跳转	CJP	510	2	0.38	0.48	7.4	0.95	当 JMP 条件满足
条件跳转非	CJPN	511	2	0.38	0.48	8.5	0.95	当 JMP 条件满足

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
多路跳转	JMP0	515	1	0.06	0.06	0.12	0.15	---
多路跳转结束	JME0	516	1	0.06	0.06	0.12	0.15	---
FOR 循环	FOR	512	2	0.21	0.21	0.21	1.00	指定一个常数
中断循环	BREAK	514	1	0.12	0.12	0.12	0.15	---
NEXT 循环	NEXT	513	1	0.17	0.17	0.17	0.45	当循环继续
				0.12	0.12	0.12	0.55	当循环结束

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-4 定时器和计数器指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
定时器	TIM	---	3	0.56	0.88	0.42	1.30	---
	TIMX	550				---		
计数器	CNT	---	3	0.56	0.88	0.42	1.30	---
	CNTX	546				---		
高速定时器	TIMH	15	3	0.88	1.14	0.42	1.80	---
	TIMHX	551				---		
1 秒定时器	TMHH	540	3	0.86	1.12	0.42	1.75	---
	TMHHX	552				---		
累加定时器	TTIM	87	3	16.1	17.0	21.4	27.4	---
				10.9	11.4	14.8	19.0	当复位时
				8.5	8.7	10.7	15.0	当互锁时
	TTIMX	555		16.1	17.0	---	27.4	---
				10.9	11.4	---	19.0	当复位时
				8.5	8.7	---	15.0	当互锁时
长时间定时器	TIML	542	4	7.6	10.0	12.8	16.3	---
				6.2	6.5	7.8	13.8	当互锁时
	TIMLX	553		7.6	10.0	---	16.3	---
				6.2	6.5	---	13.8	当互锁时
多路输出定时器	MTIM	543	4	20.9	23.3	26.0	38.55	---
				5.6	5.8	7.8	12.9	当复位时
	MTIMX	554		20.9	23.3	---	38.55	---
				5.6	5.8	---	12.9	当复位时
可逆计数器	CNTR	12	3	16.9	19.0	20.9	31.8	---
	CNTRX	548				---		
复位定时器 / 计数器	CNR	545	3	9.9	10.6	13.9	14.7	当复位 1 个字时
				4.16 ms	4.16 ms	5.42 ms	6.21 ms	当复位 1,000 个字时
	CNRX	547		9.9	10.6	---	14.7	当复位 1 个字时
				4.16 ms	4.16 ms	---	6.21 ms	当复位 1,000 个字时

注　当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-5　比较指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
输入比较指令（无符号双字）	LD, AND, OR +=	300	4	0.10	0.16	0.37	0.35	---
	LD, AND, OR + <>	305						
	LD, AND, OR + <	310						
	LD, AND, OR +<=	315						
	LD, AND, OR +>	320						
	LD, AND, OR +>=	325						
输入比较指令（带符号）	LD, AND, OR +=+L	301	4	0.10	0.16	0.54	0.35	---
	LD, AND, OR +<>+L	306						
	LD, AND, OR +<+L	311						
	LD, AND, OR +<=+L	316						
	LD, AND, OR +>+L	321						
	LD, AND, OR +>=+L	326						
输入比较指令（带符号双字）	LD, AND, OR +=+S	302	4	0.10	0.16	6.50	0.35	---
	LD, AND, OR +<>+S	307						
	LD, AND, OR +<+S	312						
	LD, AND, OR +<=	317						
	LD, AND, OR +>+S	322						
	LD, AND, OR +>=+S	327						

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
输入比较指令（带符号双字）	LD, AND, OR +=+SL	303	4	0.10	0.16	6.50	0.35	---
	LD, AND, OR +<>+SL	308						
	LD, AND, OR +<+SL	313						
	LD, AND, OR +<=+SL	318						
	LD, AND, OR +>+SL	323						
	LD, AND, OR +>=+SL	328						
比较	CMP	20	3	0.04	0.04	0.29	0.10	---
	!CMP	20	7	42.1	42.1	42.4	+45.2	增加立即刷新
双字比较	CMPL	60	3	0.08	0.08	0.46	0.50	---
带符号二进制比较	CPS	114	3	0.08	0.08	6.50	0.30	---
	!CPS	114	7	35.9	35.9	42.4	+45.2	增加立即刷新
带符号双字二进制比较	CPSL	115	3	0.08	0.08	6.50	0.50	---
表格比较	TCMP	85	4	14.0	15.2	21.9	29.77	---
多路比较	MCMP	19	4	20.5	22.8	31.2	45.80	---
无符号块比较	BCMP	68	4	21.5	23.7	32.6	47.93	---
扩展块比较	BCMP2	502	4	---	---	---	13.20	数据字数：1
				---	---	---	650.0	数据字数：255
区域比较	ZCP	88	3	5.3	5.4	---	11.53	---
双字区域比较	ZCPL	116	3	5.5	6.7	---	11.28	---

注　当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-6 数据传送指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
传送	MOV	21	3	0.18	0.20	0.29	0.30	---
	!MOV	21	7	21.38	21.40	42.36	+35.1	增加立即刷新
双字传送	MOVL	498	3	0.32	0.34	0.50	0.60	---
传送非	MVN	22	3	0.18	0.20	0.29	0.35	---
双字传送非	MVNL	499	3	0.32	0.34	0.50	0.60	---
位传送	MOVB	82	4	0.24	0.34	7.5	0.50	---
数字传送	MOVD	83	4	0.24	0.34	7.3	0.50	---
多位传送	XFRB	62	4	10.1	10.8	13.6	20.9	传送 1 位
				186.4	189.8	269.2	253.3	传送 255 位
块传送	XFER	70	4	0.36	0.44	11.2	0.8	传送 1 个字
				300.1	380.1	633.5	650.2	传送 1,000 个字
块设置	BSET	71	4	0.26	0.28	8.5	0.55	设置 1 个字
				200.1	220.1	278.3	400.2	设置 1,000 个字
数据交换	XCHG	73	3	0.40	0.56	0.7	0.80	---
双字数据交换	XCGL	562	3	0.76	1.04	1.3	1.5	---
单字分配	DIST	80	4	5.1	5.4	7.0	6.6	---
数据收集	COLL	81	4	5.1	5.3	7.1	6.5	---
传送到寄存器	MOVR	560	3	0.08	0.08	0.50	0.60	---
把定时器 / 计数器的 PV 值传送到寄存器	MOVRW	561	3	0.42	0.50	0.50	0.60	---

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-7 数据移位指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
移位寄存器	SFT	10	3	7.4	10.4	10.4	11.9	移位 1 个字
				433.2	488.0	763.1	1.39 ms	移位 1,000 个字
可逆移位寄存器	SFTR	84	4	6.9	7.2	9.6	11.4	移位 1 个字
				615.3	680.2	859.6	1.43 ms	移位 1,000 个字
异步移寄存器	ASFT	17	4	6.2	6.4	7.7	13.4	移位 1 个字
				1.22 ms	1.22 ms	2.01 ms	2.75 ms	移位 1,000 个字
字移位	WSFT	16	4	4.5	4.7	7.8	9.6	移位 1 个字
				171.5	171.7	781.7	928.0	移位 1,000 个字
算术左移	ASL	25	2	0.22	0.32	0.37	0.45	---

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
双字左移	ASLL	570	2	0.40	0.56	0.67	0.80	---
算术右移	ASR	26	2	0.22	0.32	0.37	0.45	---
双字右移	ASRL	571	2	0.40	0.56	0.67	0.80	---
循环左移	ROL	27	2	0.22	0.32	0.37	0.45	---
双字循环左移	ROLL	572	2	0.40	0.56	0.67	0.80	---
无进位循环左移	RLNC	574	2	0.22	0.32	0.37	0.45	---
无进位双字循环左移	RLNL	576	2	0.40	0.56	0.67	0.80	---
循环右移	ROR	28	2	0.22	0.32	0.37	0.45	---
双字循环右移	RORL	573	2	0.40	0.56	0.67	0.80	---
无进位循环右移动	RRNC	575	2	0.22	0.32	0.37	0.45	---
无进位双字循环右移	RRNL	577	2	0.40	0.56	0.67	0.80	---
一个数字左移	SLD	74	3	5.9	6.1	8.2	7.6	移位 1 个字
				561.1	626.3	760.7	1.15 ms	移位 1,000 个字
一个数字右移	SRD	75	3	6.9	7.1	8.7	8.6	移位 1 个字
				760.5	895.5	1.07 ms	1.72 ms	移位 1,000 个字
左移 N 位数据	NSFL	578	4	7.5	8.3	10.5	14.8	移位 1 位
				40.3	45.4	55.5	86.7	移位 1,000 位
右移 N 位数据	NSFR	579	4	7.5	8.3	10.5	14.7	移位 1 位
				50.5	55.3	69.3	114.1	移位 1,000 位
左移 N 位	NASL	580	3	0.22	0.32	0.37	0.45	---
双字左移 N 位	NSLL	582	3	0.40	0.56	0.67	0.80	---
右移 N 位	NASR	581	3	0.22	0.32	0.37	0.45	---
双字右移 N 位	NSRL	583	3	0.40	0.56	0.67	0.80	---

注　当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-8 递增 / 递减指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
二进制递增	++	590	2	0.22	0.32	0.37	0.45	---
双字二进制递增	++L	591	2	0.40	0.56	0.67	0.80	---
二进制递减	– –	592	2	0.22	0.32	0.37	0.45	---
双字二进制递减	– –L	593	2	0.40	0.56	0.67	0.80	---
BCD 递增	++B	594	2	6.4	4.5	7.4	12.3	---
双字 BCD 递增	++BL	595	2	5.6	4.9	6.1	9.24	---
BCD 递减	– –B	596	2	6.3	4.6	7.2	11.9	---
双字 BCD 递减	– –BL	597	2	5.3	4.7	7.1	9.0	---

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-9 四则运算指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
无进位带符号二进制加法	+	400	4	0.18	0.20	0.37	0.30	---
无进位带符号双字二进制加法	+L	401	4	0.32	0.34	0.54	0.60	---
有进位带符号二进制加法	+C	402	4	0.18	0.20	0.37	0.40	---
有进位带符号双字二进制加法	+CL	403	4	0.32	0.34	0.54	0.60	---
无进位 BCD 加法	+B	404	4	8.2	8.4	14.0	18.9	---
无进位双字 BCD 加法	+BL	405	4	13.3	14.5	19.0	24.4	---
有进位 BCD 加法	+BC	406	4	8.9	9.1	14.5	19.7	---

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
有进位双字 BCD 加法	+BCL	407	4	13.8	15.0	19.6	25.2	---
无进位带符号二进制减法	–	410	4	0.18	0.20	0.37	0.3	---
无进位双字带符号二进制减法	–L	411	4	0.32	0.34	0.54	0.60	---
有进位带符号二进制减法	–C	412	4	0.18	0.20	0.37	0.40	---
有进位带符号双字二进制减法	–CL	413	4	0.32	0.34	0.54	0.60	---
无进位 BCD 减法	–B	414	4	8.0	8.2	13.1	18.1	---
无进位双字 BCD 减法	–BL	415	4	12.8	14.0	18.2	23.2	---
有进位 BCD 减法	–BC	416	4	8.5	8.6	13.8	19.1	---
有进位双字 BCD 减法	–BCL	417	4	13.4	14.7	18.8	24.3	---
带符号二进制乘法	*	420	4	0.38	0.40	0.58	0.65	---
带符号双字二进制乘法	*L	421	4	7.23	8.45	11.19	13.17	---
不带符号二进制乘法	*U	422	4	0.38	0.40	0.58	0.75	---
不带符号双字二进制乘法	*UL	423	4	7.1	8.3	10.63	13.30	---
BCD 乘法	*B	424	4	9.0	9.2	12.8	17.5	---
双字 BCD 乘法	*BL	425	4	23.0	24.2	35.2	36.3	---
带符号二进制除法	/	430	4	0.40	0.42	0.83	0.70	---
带符号双字二进制除法	/L	431	4	7.2	8.4	9.8	13.7	---

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
不带符号二进制除法	/U	432	4	0.40	0.42	0.83	0.8	---
不带符号双字二进制除法	/UL	433	4	6.9	8.1	9.1	12.8	---
BCD 除法	/B	434	4	8.6	8.8	15.9	19.3	---
双字 BCD 除法	/BL	435	4	17.7	18.9	26.2	27.1	---

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-10 转换指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
BCD →二进制	BIN	023	3	0.22	0.24	0.29	0.40	---
双字 BCD →双字二进制	BINL	058	3	6.5	6.8	9.1	12.3	---
二进制→BCD	BCD	024	3	0.24	0.26	8.3	7.62	---
双字二进制→双字 BCD	BCDL	059	3	6.7	7.0	9.2	10.6	---
二进制求补	NEG	160	3	0.18	0.20	0.29	0.35	---
双字二进制求补	NEGL	161	3	0.32	0.34	0.5	0.60	---
带符号二进制 16 位→ 32 位	SIGN	600	3	0.32	0.34	0.50	0.60	---
数据译码	MLPX	076	4	0.32	0.42	8.8	0.85	1 位数字译码（4 ～ 6）
				0.98	1.20	12.8	1.60	4 位数字译码（4 ～ 6）
				3.30	4.00	20.3	4.70	1 位数字译码（8 ～ 256）
				6.50	7.90	33.4	8.70	2 位数字译码（8 ～ 256）
数据编码	DMPX	077	4	7.5	7.9	10.4	9.4	1 位数字编码（16 ～ 4）
				49.6	50.2	59.1	57.3	4 位数字编码（16 ～ 4）
				18.2	18.6	23.6	56.8	1 位数字编码（256 ～ 8）
				55.1	57.4	92.5	100.0	2位数字编码（256 ～ 8）

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
ASCII 转换	ASC	086	4	6.8	7.1	9.7	8.3	1 位数字转换→ ASCII
				11.2	11.7	15.1	19.1	4 位数字转换→ ASCII
ASCII 码→十六进制	HEX	162	4	7.1	7.4	10.1	12.1	转换到 1 位数字
列→行	LINE	063	4	19.0	23.1	29.1	37.0	---
行→列	COLM	064	4	23.2	27.5	37.3	45.7	---
带符号 BCD →二进制	BINS	470	4	8.0	8.3	12.1	16.2	数据格式设置 0
				8.0	8.3	12.1	16.2	数据格式设置 1
				8.3	8.6	12.7	16.5	数据格式设置 2
				8.5	8.8	13.0	16.5	数据格式设置 3
带符号双字 BCD →二进制	BISL	472	4	9.2	9.6	13.6	18.4	数据格式设置 0
				9.2	9.6	13.7	18.5	数据格式设置 1
				9.5	9.9	14.2	18.6	数据格式设置 2
				9.6	10.0	14.4	18.7	数据格式设置 3
带符号二进制→ BCD	BCDS	471	4	6.6	6.9	10.6	13.5	数据格式设置 0
				6.7	7.0	10.8	13.8	数据格式设置 1
				6.8	7.1	10.9	13.9	数据格式设置 2
				7.2	7.5	11.5	14.0	数据格式设置 3
带符号双字二进制→ BCD	BDSL	473	4	8.1	8.4	11.6	11.4	数据格式设置 0
				8.2	8.6	11.8	11.7	数据格式设置 1
				8.3	8.7	12.0	11.8	数据格式设置 2
				8.8	9.2	12.5	11.9	数据格式设置 3

注　当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-11 逻辑指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
逻辑与	ANDW	034	4	0.18	0.20	0.37	0.30	---
双字逻辑与	ANDL	610	4	0.32	0.34	0.54	0.60	---
逻辑或	ORW	035	4	0.22	0.32	0.37	0.45	---
双字逻辑或	ORWL	611	4	0.32	0.34	0.54	0.60	---
异或	XORW	036	4	0.22	0.32	0.37	0.45	---
双字异或	XORL	612	4	0.32	0.34	0.54	0.60	---
异或非	XNRW	037	4	0.22	0.32	0.37	0.45	---
双字异或非	XNRL	613	4	0.32	0.34	0.54	0.60	---
求反	COM	029	2	0.22	0.32	0.37	0.45	---
双字求反	COML	614	2	0.40	0.56	0.67	0.80	---

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-12 特殊数学指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
二进制平方根	ROTB	620	3	49.6	50.0	530.7	56.5	---
BCD 平方根	ROOT	072	3	13.7	13.9	514.5	59.3	---
算术处理	APR	069	4	6.7	6.9	32.3	14.0	指定 SIN 和 COS
				17.2	18.4	78.3	32.2	指定线段逼近
浮点除法	FDIV	079	4	116.6	176.6	176.6	246.0	---
位计数器	BCNT	067	4	0.3	0.38	22.1	0.65	计数 1 个字

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-13 浮点数学指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
浮点→ 16 位	FIX	450	3	10.6	10.8	14.5	16.2	---
浮点→ 32 位	FIXL	451	3	10.8	11.0	14.6	16.6	---
16 位→浮点	FLT	452	3	8.3	8.5	11.1	12.2	---
32 位→浮点	FLTL	453	3	8.3	8.5	10.8	14.0	---
浮点加法	+F	454	4	8.0	9.2	10.2	13.3	---
浮点减法	–F	455	4	8.0	9.2	10.3	13.3	---
浮点乘法	/F	457	4	8.7	9.9	12.0	14.0	---
浮点除法	*F	456	4	8.0	9.2	10.5	13.2	---
度→弧度	RAD	458	3	10.1	10.2	14.9	15.9	---
弧度→度	DEG	459	3	9.9	10.1	14.8	15.7	---
正弦	SIN	460	3	42.0	42.2	61.1	47.9	---
余弦	COS	461	3	31.5	31.8	44.1	41.8	---
正切	TAN	462	3	16.3	16.6	22.6	20.8	---
反正弦	ASIN	463	3	17.6	17.9	24.1	80.3	---
反余弦	ACOS	464	3	20.4	20.7	28.0	25.3	---
反正切	ATAN	465	3	16.1	16.4	16.4	45.9	---
平方根	SQRT	466	3	19.0	19.3	28.1	26.2	---
指数	EXP	467	3	65.9	66.2	96.7	68.8	---
对数	LOG	468	3	12.8	13.1	17.4	69.4	---
指数幂	PWR	840	4	125.4	126.0	181.7	134.0	---
单精度浮点比较指令	LD, AND, OR +=F	329	3	6.6	8.3	---	12.6	---
	LD, AND, OR +<>F	330						
	LD, AND, OR +<F	331						
	LD, AND, OR +<=F	332						
	LD, AND, OR +>F	333						
	LD, AND, OR +>=F	334						
浮点→ ASCII	FSTR	448	4	48.5	48.9	---	58.4	---
ASCII →浮点	FVAL	449	3	21.1	21.3	---	31.1	---

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-14 双精度浮点指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
双精度浮点比较指令	LD, AND, OR +=D	335	3	8.5	10.3	---	16.2	---
	LD, AND, OR +<>D	336						
	LD, AND, OR +<D	337						
	LD, AND, OR +<=D	338						
	LD, AND, OR +>D	339						
	LD, AND, OR +>=D	340						
双字浮点→ 16 位	FIXD	841	3	11.7	12.1	---	16.1	---
双字浮点→ 32 位	FIXLD	842	3	11.6	12.1	---	16.4	---
16 位→双字浮点	DBL	843	3	9.9	10.0	---	14.3	---
32 位→双字浮点	DBLL	844	3	9.8	10.0	---	16.0	---
双字浮点加法	+D	845	4	11.2	11.9	---	18.3	---
双字浮点减法	–D	846	4	11.2	11.9	---	18.3	---
双字浮点乘法	*D	847	4	12.0	12.7	---	19.0	---
双字浮点除法	/D	848	4	23.5	24.2	---	30.5	---
双字度→弧度	RADD	849	3	27.4	27.8	---	32.7	---
双字弧度→度	DEGD	850	3	11.2	11.9	---	33.5	---
双字正弦	SIND	851	3	45.4	45.8	---	67.9	---
双字余弦	COSD	852	3	43.0	43.4	---	70.9	---
双字正切	TAND	853	3	20.1	20.5	---	97.9	---

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
双字反正弦	ASIND	854	3	21.5	21.9	---	32.3	---
双字反余弦	ACOSD	855	3	24.7	25.1	---	29.9	---
双字反正切	ATAND	856	3	19.3	19.7	---	24.0	---
双字平方根	SQRTD	857	3	47.4	47.9	---	52.9	---
双字指数	EXPD	858	3	121.0	121.4	---	126.3	---
双字对数	LOGD	859	3	16.0	16.4	---	21.6	---
双字指数幂	PWRD	860	4	223.9	224.2	---	232.3	---

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-15 表格数据处理指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
设置堆栈	SSET	630	3	8.0	8.3	8.5	14.2	堆栈区指定 5 个字
				231.6	251.8	276.8	426.5	堆栈区指定 1,000 个字
推入堆栈	PUSH	632	3	6.5	8.6	9.1	15.7	---
先进先出	FIFO	633	3	6.9	8.9	10.6	15.8	堆栈区指定 5 个字
				352.6	434.3	1.13 ms	728.0	堆栈区指定 1,000 个字
后进先出	LIFO	634	3	7.0	9.0	9.9	16.6	---
定维记录表	DIM	631	5	15.2	21.6	142.1	27.8	---
设置记录位置	SETR	635	4	5.4	5.9	7.0	12.8	---
获取记录号	GETR	636	4	7.8	8.4	11.0	16.1	---
数据搜索	SRCH	181	4	15.5	19.5	19.5	29.1	搜索 1 个字
				2.42 ms	3.34 ms	3.34 ms	4.41 ms	搜索 1,000 个字
字节交换	SWAP	637	3	12.2	13.6	13.6	21.0	交换 1 个字
				1.94 ms	2.82 ms	2.82 ms	3.65 ms	交换 1,000 个字
寻找最大值	MAX	182	4	19.2	24.9	24.9	35.3	搜索 1 个字
				2.39 ms	3.36 ms	3.36 ms	4.39 ms	搜索 1,000 个字

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
寻找最小值	MIN	183	4	19.2	25.3	25.3	35.4	搜索 1 个字
				2.39 ms	3.33 ms	3.33 ms	4.39 ms	搜索 1,000 个字
求和	SUM	184	4	28.2	38.5	38.3	49.5	1 字相加
				1.42 ms	1.95 ms	1.95 ms	2.33 ms	1,000 字相加
帧校验和	FCS	180	4	20.0	28.3	28.3	34.8	表长 1 字
				1.65 ms	2.48 ms	2.48 ms	3.11 ms	表长 1,000 字
读堆栈大小	SNUM	638	3	6.0	6.3	---	12.1	---
读堆栈数据	SREAD	639	4	8.0	8.4	---	18.1	---
写堆栈数据	SWRIT	640	4	7.2	7.6	---	16.9	---
插入堆栈数据	SINS	641	4	7.8	9.9	---	18.2	---
				354.0	434.8	---	730.7	1,000 字表
删除堆栈数据	SDEL	642	4	8.6	10.6	---	19.3	---
				354.0	436.0	---	732.0	1,000 字表

注　当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-16 数据控制指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
PID 控制	PID	190	4	436.2	678.2	678.2	612.0	初始化执行
				332.3	474.9	474.9	609.3	采样
				97.3	141.3	141.3	175.3	不采样
限位控制	LMT	680	4	16.1	22.1	22.1	27.1	---
静带控制	BAND	681	4	17.0	22.5	22.5	27.4	---
静域控制	ZONE	682	4	15.4	20.5	20.5	28.0	---
标度	SCL	194	4	37.1	53.0	56.8	25.0	---
标度 2	SCL2	486	4	28.5	40.2	50.7	22.3	---
标度 3	SCL3	487	4	33.4	47.0	57.7	25.6	---
求平均值	AVG	195	4	36.3	52.6	53.1	62.9	1 个操作的平均值
				291.0	419.9	419.9	545.3	64 个操作的平均值
自动 PID 控制	PIDAT	191	4	446.3	712.5	---	765.3	初始化执行
				339.4	533.9	---	620.7	采样
				100.7	147.1	---	180.0	不采样
				189.2	281.6	---	233.7	自动的初始化执行
				535.2	709.8	---	575.3	采样时自动

注　当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-17 子程序指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
子程序调用	SBS	91	2	1.26	1.96	17.0	2.04	---
子程序入口	SBN	92	2	---	---	---	---	---
子程序返回	RET	93	1	0.86	1.60	20.60	1.80	---
宏	MCRO	99	4	23.3	23.3	23.3	47.9	---
进入全局子程序	GSBN	751	2	---	---	---	---	---
全局子程序返回	GRET	752	1	1.26	1.96	---	2.04	---
调用全局子程序	GSBS	750	2	0.86	1.60	---	1.80	---

注　当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-18 中断控制指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
设置中断屏蔽	MSKS	690	3	25.6	38.4	39.5	44.7	---
读中断屏蔽	MSKR	692	3	11.9	11.9	11.9	16.9	---
清除中断	CLI	691	3	27.4	41.3	41.3	42.7	---
禁止中断	DI	693	1	15.0	16.8	16.8	30.3	---
允许中断	EI	694	1	19.5	21.8	21.8	37.7	---

注　当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-19 高速计数器 / 脉冲输出指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
模式控制	INI	880	4	---	---	---	77.00	开始高速计数器比较
				---	---	---	43.00	停止高速计数器比较
				---	---	---	43.40	改变脉冲输出 PV 值
				---	---	---	51.80	改变高速计数器 PV 值
				---	---	---	31.83	中断模式中改变计数器的 PV 值
				---	---	---	45.33	停止脉冲输出
				---	---	---	36.73	停止 PWM(891) 输出
高速计数器 PV 读	PRV	881	4	---	---	---	42.40	读脉冲输出 PV 值
				---	---	---	53.40	读高速计数器 PV 值
				---	---	---	33.60	中断模式中读计数器 PV 值
				---	---	---	38.80	读脉冲输出状态
				---	---	---	39.30	读高速计数器状态
				---	---	---	38.30	读 PWM(891) 状态
				---	---	---	117.73	读高速计数器范围比较结果
				---	---	---	48.20	读高速计数器 0 的频率
寄存器比较表	CTBL	882	4	---	---	---	238.0	对 1 个目标值登记目标值表并开始比较
				---	---	---	14.42 ms	对 48 个目标值登记目标值表并开始比较
				---	---	---	289.0	登记范围表并开始比较
				---	---	---	198.0	只对 1 个目标值登记目标值表
				---	---	---	14.40 ms	只对 48 个目标值登记目标值表
				---	---	---	259.0	只登记范围表

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
速度输出	SPED	885	4	---	---	---	56.00	连续模式
				---	---	---	62.47	独立模式
设置脉冲	PULS	886	4	---	---	---	26.20	---
脉冲输出	PLS2	887	5	---	---	---	100.80	---
加速度控制	ACC	888	4	---	---	---	90.80	连续模式
				---	---	---	80.00	独立模式
寻找原点	ORG	889	3	---	---	---	106.13	原点搜索
				---	---	---	52.00	原点返回
可变占空比脉冲	PWM	891	4	---	---	---	25.80	---

4-2-20 步指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
步定义	STEP	008	2	17.4	20.7	27.1	35.9	步控制位 ON
				11.8	13.7	24.4	13.8	步控制位 OFF
步启动	SNXT	009	2	6.6	7.3	10.0	12.1	---

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-21 基本 I/O 单元指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
I/O 刷新	IORF	097	3	15.5	16.4	23.5	26.7	基本 I/O 单元刷新 1 字（输入）
				319.9	320.7	377.6	291.0	基本 I/O 单元刷新 60 字（输入）
				358.00	354.40	460.1	325.0	基本 I/O 单元刷新 60 字（输出）
7 段译码	SDEC	78	4	6.5	6.9	14.1	8.1	---
智能 I/O 读	IORD	222	4	读 / 写时间取决于正在执行该指令的特殊 I/O 单元				---
智能 I/O 写	IOWR	223	4					---
CPU 总线单元 I/O 刷新	DLNK	226	4	287.8	315.5	---	321.3	分配 1 个字

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-22 串行通信命令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
协议宏	PMCR	260	5	100.1	142.1	276.8	158.4	发送 0 字，接收 0 字
				134.2	189.6	305.9	210.0	发送 249 字，接收 249 字
传送	TXD	236	4	68.5	98.8	98.8	109.3	发送 1 字节
				734.3	1.10 ms	1.10 ms	1.23 ms	发送 256 字节
接收	RXD	235	4	89.6	131.1	131.1	144.0	存储 1 字节
				724.2	1.11 ms	1.11 ms	1.31 ms	存储 256 字节
修改串行口设置	STUP	237	3	341.2	400.0	440.4	504.7	---

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-23 网络指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
网络发送	SEND	090	4	84.4	123.9	123.9	141.6	---
网络接收	RECV	098	4	85.4	124.7	124.7	142.3	---
发布命令	CMND	490	4	106.8	136.8	136.8	167.7	---

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-24 文件存储指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
读数据文件	FREAD	700	5	391.4	632.4	684.1	657.3	2 字符目录 + 文件名二进制
				836.1	1.33 ms	1.35 ms	1.45 ms	73 字符目录 + 文件名二进制
写数据文件	FWRIT	701	5	387.8	627.0	684.7	650.7	2 字符目录 + 文件名二进制
				833.3	1.32 ms	1.36 ms	1.44 ms	73 字符目录 + 文件名二进制

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-25 显示指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
显示信息	MSG	046	3	10.1	14.2	14.3	16.8	显示信息
				8.4	11.3	11.3	14.7	删除显示的信息

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-26 时钟指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
日历加法	CADD	730	4	38.3	201.9	209.5	217.0	---
日历减法	CSUB	731	4	38.6	170.4	184.1	184.7	---
小时→秒	SEC	065	3	21.4	29.3	35.8	36.1	---
秒→小时	HMS	066	3	22.2	30.9	42.1	45.1	---
时钟调整	DATE	735	2	216.0	251.5	120.0	118.7	---

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-27 调试指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
跟踪存储采样	TRSM	045	1	80.4	120.0	120.0	207.0	采样 1 位和 0 字
				848.1	1.06 ms	1.06 ms	1.16 ms	采样 31 位和 6 字

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-28 故障诊断指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
故障报警	FAL	006	3	15.4	16.7	16.7	26.1	记录错误
				179.8	244.8	244.8	294.0	按优先次序删除错误
				432.4	657.1	657.1	853.3	删除全部错误
				161.5	219.4	219.4	265.7	删除单个错误
严重故障报警	FALS	007	3	---	---	---	---	---
故障点检测	FPD	269	4	140.9	202.3	202.3	220.7	当执行时
				163.4	217.6	217.6	250.3	第一次
				185.2	268.9	268.9	220.7	当执行时
				207.5	283.6	283.6	320.7	第一次

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-29 其它命令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
设置进位	STC	040	1	0.06	0.06	0.12	0.15	---
清除进位	CLC	041	1	0.06	0.06	0.12	0.15	---
设置 EM 块	EMBC	281	2	14.0	15.1	15.1	---	---
延长最大循环时间	WDT	094	2	15.0	19.7	19.7	23.6	---
存入条件标志	CCS	282	1	8.6	12.5	---	14.2	---
载入条件标志	CCL	283	1	9.8	13.9	---	16.3	---
从 CV 转变地址	FRMCV	284	3	13.6	19.9	---	23.1	---
转变地址到 CV	TOCV	285	3	11.9	17.2	---	22.5	---
外设服务禁止	IOSP	287	---	13.9	19.8	---	21.5	---
外围服务允许	IORS	288	---	63.6	92.3	---	22.2	---

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-30 块程序指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
块程序开始	BPRG	096	2	12.1	13.0	13.0	27.5	---
块程序结束	BEND	801	1	9.6	12.3	13.1	23.2	---
块程序暂停	BPPS	811	2	10.6	12.3	14.9	16.0	---
块程序重新启动	BPRS	812	2	5.1	5.6	8.3	9.0	---
条件块退出	（执行条件）EXIT	806	1	10.0	11.3	12.9	23.8	退出条件满足
				4.0	4.9	7.3	7.2	退出条件不满足
条件块退出	EXIT （位地址）	806	2	6.8	13.5	16.3	28.4	退出条件满足
				4.7	7.2	10.7	11.4	退出条件不满足
条件块退出 （非）	EXIT NOT（位地址）	806	2	12.4	14.0	16.8	28.4	退出条件满足
				7.1	7.6	11.2	11.8	退出条件不满足
分支	IF （执行条件）	802	1	4.6	4.8	7.2	6.8	如果真
				6.7	7.3	10.9	12.2	如果假
分支	IF （继电器号）	802	2	6.8	7.2	10.4	11.0	如果真
				9.0	9.6	14.2	16.5	如果假
分支 （非）	IF NOT（继电器号）	802	2	7.1	7.6	10.9	11.5	如果真
				9.2	10.1	14.7	16.8	如果假
分支	ELSE	803	1	6.2	6.7	9.9	11.4	如果真
				6.8	7.7	11.2	13.4	如果假
分支	IEND	804	1	6.9	7.7	11.0	13.5	如果真
				4.4	4.6	7.0	6.93	如果假
一个循环与等待	WAIT （执行条件）	805	1	12.6	13.7	16.7	28.6	等待条件满足
				3.9	4.1	6.3	5.6	等待条件不满足
一个循环与等待	WAIT （继电器号）	805	2	12.0	13.4	16.5	27.2	等待条件满足
				6.1	6.5	9.6	10.0	等待条件不满足
一个循环与等待（非）	WAIT NOT（继电器号）	805	2	12.2	13.8	17.0	27.8	等待条件满足
				6.4	6.9	10.1	10.5	等待条件不满足
计数器等待	CNTW	814	4	17.9	22.6	27.4	41.0	首先执行
				19.1	23.9	28.7	42.9	正常执行
	CNTWX	818	4	17.9	22.6	---	41.0	首先执行
				19.1	23.9	---	42.9	正常执行

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
高速定时器等待	TMHW	815	3	25.8	27.9	34.1	47.9	首先执行
				20.6	22.7	28.9	40.9	正常执行
	TMHWX	817	3	25.8	27.9	---	47.9	首先执行
				20.6	22.7	---	40.9	正常执行
循环控制	LOOP	809	1	7.9	9.1	12.3	15.6	---
循环控制	LEND（执行条件）	810	1	7.7	8.4	10.9	13.5	LEND 条件满足
				6.8	8.0	9.8	17.5	LEND条件不满足
循环控制	LEND（继电器号）	810	2	9.9	10.7	14.4	17.5	LEND 条件满足
				8.9	10.3	13.0	21.6	LEND条件不满足
循环控制	LEND NOT（继电器号）	810	2	10.2	11.2	14.8	21.9	LEND 条件满足
				9.3	10.8	13.5	17.8	LEND条件不满足
定时器等待	TIMW	813	3	22.3	25.2	33.1	47.4	缺省设置
				24.9	27.8	35.7	46.2	正常执行
	TIMWX	816	3	22.3	25.2	33.1	47.4	缺省设置
				24.9	27.8	35.7	46.2	正常执行

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-31 文本串处理指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
串传送	MOV$	664	3	45.6	66.0	84.3	79.3	传送 1 个字符
串链接	+$	656	4	86.5	126.0	167.8	152.0	1 个字符 +1 个字符
取左串	LEFT$	652	4	53.0	77.4	94.3	93.6	从 2 个字符中获取 1 个字符
取右串	RGHT$	653	4	52.2	76.3	94.2	92.1	从 2 个字符中获取 1 个字符
取中间串	MID$	654	5	56.5	84.6	230.2	93.7	从 3 个字符中获取 1 个字符
寻找串	FIND$	660	4	51.4	77.5	94.1	89.1	从 2 个字符中搜索 1 个字符
串长度	LEN$	650	3	19.8	28.9	33.4	33.8	检测 1 个字符
替换串	RPLC$	661	6	175.1	258.7	479.5	300.7	用一个字符替换 2 个字符中的第一个字符

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
删除串	DEL$	658	5	63.4	94.2	244.6	11.3	删除 2 个字符中前面的字符
交换串	XCHG$	665	3	60.6	87.2	99.0	105.2	1 对 1 交换字符
清除串	CLR$	666	2	23.8	36.0	37.8	42.0	清除 1 个字符
插入串	INS$	657	5	136.5	200.6	428.9	204.0	在 2 个字符首字符之后插入 1 个字符
串比较指令	LD, AND, OR +=$	670	4	48.5	69.8	86.2	79.9	1 个字符与另一个字符比较
	LD, AND, OR +<>$	671						
	LD, AND, OR +<$	672						
	LD, AND, OR +>$	674						
	LD, AND, OR +>=$	675						

注 当使用双字长度的操作数时，将下表长度列的值加 1。

4-2-32 任务控制指令

指令	助记符	代码	长度（步）（见注）	ON 执行时间 (μs)				条件
				CPU6@H	CPU4@H	CPU4@	CJ1M	
任务 ON	TKON	820	2	19.5	26.3	26.3	33.1	---
任务 OFF	TKOF	821	2	13.3	19.0	26.3	19.7	---

4-2-33 从以前的 OMRON PLC 转换程序容量的规则

下表提供了从以前的 OMRON PLC（SYSMAC C200HX/HG/HE，CVM1 或 CV 系列 PLC）的程序容量（单位：字）转换到 CJ 系列 PLC 的程序容量（单位：步）的规则。

为了获得 CJ 系列 PLC 的程序容量（单位：步），在以前 PLC 每条指令的程序容量（单位：字）上加上值 (n)。

CJ 系列步 = 以前 PC 的 “a”（字） +n			
指令	变化	C200HX/HG/HE 转换到 CS 系列时的 n 值	CV 系列 PLC 或 CVM1 转换到 CS 系列时的 n 值
基本指令	无	OUT, SET, RSET, 或 KEEP(011): –1 其它指令：0	0
	上升微分	无	+1
	立即刷新	无	0
	上升微分和立即刷新	无	+2
特殊指令	无	0	–1
	上升微分	+1	0
	立即刷新	无	+3
	上升微分和立即刷新	无	+4

例如，如果使用了一个地址为 CIO 000000 ～ CIO 25515 的 OUT，以前的 PC 的程序容量每个指令为 2 个字，而 CJ 系列 PC 每指令是 1(2-1) 步。

例如，如果使用了！ MOV 指令 （立即刷新的 MOVE 指令）， CV 系列的程序容量每个指令为 4 个字，而 CJ 系列 PC 是 7(4+3) 步。