SYSMAC CS/CJ 系列
CS1G/H-CPU@@-EV1
CS1G/H-CPU@@H
CS1D-CPU@@H
CJ1G-CPU@@
CJ1G/H-CPU@@H
CJ1M-CPU@@

可编程序控制器

编程手册

2003 年 4 月出版

注意：

OMRON 制造的产品由具有一定资格的人员按适当步骤使用，并只能按本手册描述的功能使用。

下列约定用于指明本手册中几类注意事项，必须按照这些要求做，否则可能引起严重的人身伤害或产品损坏。

! **危险** 指示如果不按这些信息做，会引起严重的人身伤害。

! **警告** 指示如果不按这些信息做，可能引起严重的人身伤害。

! **注意** 指示如果不按这些信息做，可能造成一些伤害或财产损失。

OMRON 产品附注

在本手册中所有 OMRON 产品都以大写字母表示。当字 “单元”表示 OMRON 产品时，它也以大写字母表示，不管它是否以产品的正式名称出现。

缩写 “Ch”，它出现在某些显示中和某些 OMRON 产品上，往往表示 “字”，在这个意义上文件中缩写 “Wd”，也同样是 “字”的意义。

缩写 “PLC”表示可编程序控制器。但是，在有些编程设备的显示中用 “PC”来表示可编程序控制器。

直观标题

列在本手册左侧的下列标题是帮助读者确定各种不同类型的信息。

注 指出对有效而方便地运用产品特别重要的信息。

1,2,3... 1. 指出一种或另一种的列举说明，如步骤，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 单元编程和下页所述章节。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	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安装或使用 CS/CJ 系列 CPU 单元之前，请仔细阅读本手册及下页表中所列的相关手册，并确保理解所有信息。

本手册包含下列章节。

第 1 章 描述 CPU 单元的基本结构和操作。

第 2 章 描述写、检查和输入程序的所需基本知识。

第 3 章 用于编写用户程序的指令概要。

第 4 章 描述任务的操作。

第 5 章 描述用于操作文件存储器的功能。

第 6 章 提供高级功能的有关说明：循环时间 / 高速处理，变址寄存器，串行通信，启动设定和维护，诊断和调试，编程装置，以及 CJ 系列基本 I/O 单元的输入响应时间设定。

第 7 章 描述用于将程序传送到 CPU 单元的过程及测试和调试程序的功能。

附录 给出了 CS/CJ 系列可编程序控制器的比较，使用 C200H 特殊 I/O 单元的限制，以及在上位链接 /Host Link 系统中所作的变更。

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,CJ1G-CPU@@, CJ1G/H-CPU@@H 可编程序控制器编程手册	W394	本手册描述了 CS/CJ 系列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 系列可编程序控制器概述及其设计、安装维护及其基本操作。
SYSMAC CJ 系列 CJ1G-CPU@@, CJ1G/H-CPU@@H 可编程序控制器操作手册	W393	提供 CJ 系列可编程序控制器概述及其设计、安装维护及其基本操作。
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。
CS 系列 CS1D-CPU@@H CPU 单元 ,CS1D-DPL01 双机单元 , CS1D-PA207R 电源单元 双机系统操作	W405	基于 CS1D CPU 单元的概述和描述，如：设计、安装、维护和其它基本操作。
SYSMAC CS/CJ 系列 CS1G/H-CPU@@-EV1, CS1G/H-CPU@@H, CS1D-CPU@@H,CJ1G-CPU@@, CJ1G/H-CPU@@H 可编程序控制器指令参考手册	W340	介绍 CS/CJ 系列可编程序控制器支持的梯形图编程指令。
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 系列可编程序控制器。
SYSMAC CS/CJ 系列 CS1G/H-CPU@@-EV1, CS1G/H-CPU@@H, 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 C 系列 （上位链接）和 FINS 通信命令。
SYSMAC WS02-CXP@@-E CX-Programmer 用户手册	W361	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-CXP@@-E CX-Server 用户手册	W362	
SYSMAC CS/CJ 系列 CS1W-SCB21/41, CS1W-SCU21, CJ1W-SCU41 串行通信卡 / 单元操作手册	W336	说明与外部设备进行串行通信的串行通信单元和通信卡的使用，以及欧姆龙产品的标准系统协议用法。
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 以太网操作单元手册	W343	说明 CJ1W-ETN01， CJ1W-ENT11 和 CJ1W-ETN11 以太网单元的安装和操作。

注意事项

本章给出使用 CS/CJ 系列可编程序控制器 （PLC）和有关设备的一般注意事项。

本章含有的信息对可编程序控制器的安全和可靠应用是很重要的。在着手安装或操作 PLC 系统前必须阅读本章并理解所含有的信息。

1 面向的读者

本手册是为下列人员编写的，他们还必须是具有电气系统知识的人员（电气工程师或相当的）。

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

2 一般注意事项

用户必须按照操作手册中叙述的性能规格来应用产品。

在将本产品用于本手册未述及的条件下或将产品应用于核控制系统、铁路系统、航空系统、车辆、内燃机系统、医疗装置、娱乐、机械、安全装置和如果使用不当时对生命和财产可能有严重影响的其它系统、机械和装置前，请向欧姆龙代理商咨询。

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，并务必给系统、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，在着手使用本单元前务必阅读本手册，并将手册备在身边以供操作时参阅。

! 警告 PLC 和所有 PLC 单元用于规定用途和规定条件下，特别是用于能直接或间接影响人的生命的应用中是极重要的。在将 PLC 系统应用于上面提到的应用前必须向欧姆龙代理商咨询。

3 安全注意事项

! 警告 即使在程序没有运行的情况下（也就是在编程模式下），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 在供电时不要试图拆卸任何单元，否则可能导致电击。

! 警告 在供电时不要触及任一端子或端子板，否则可能导致电击。

! 警告 不要试图拆卸、修理或修改任一单元，任何这样做的企图都可能导致误动作、火灾或电击。

! 警告 不要在供电时或在断电后立即触及电源，否则可能导致电击。

! 警告 为了在因PLC误动作或其它影响PLC操作的外部因素引起不正常时保证系统安全，在外部电路中（即不在可编程序控制器中）要设有安全措施，包括下列项目，不这样做可能导致严重事故。

- 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，不管 CPU 单元的操作模式如何与输出单元连接的设备可能误动作，。

! 注意 用户必须采取容错措施，以保证在发生非正常情况，由信号线断路而引起的信号丢失或反常信号、电源的瞬间中断，或其它原因的安全。非正常操作会导致严重的事故。

! 注意 用户必须在外部电路中（即不在可编程序控制器内）设有联锁电路、限位电路以及类似的安全措施。非正常操作会导致严重的事故。

! 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快闪存储器内。然而， I/O 存储器（包括 DM、EM 和 HR 区）数据不写入快闪存储器。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 只有在确认延长循环时间不会引起有害的作用后才执行在线编辑。否则，输入信号可能读不到。

! 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。

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）或含盐处；
- 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，不合适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）。手持式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。然而手持式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。
  - 为分配位和DM区规约，在和CPU单元的数据传递中有限制（源和目标区域和地址的规约）。
  - 用现场总线网（CompoBus/D）主单元（CIO 0050 到 CIO 0099）的现场总线网（CompoBus/D）输出区与 I/O 位区（CIO 0000 到 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，然后将 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 单元可能都显示出错误操作。
  - 用于PLC连接单元（CIO 0247到CIO 0250）的特殊位和标志与I/O位区域 （CIO 0000 到 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，并将 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。如不这样，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100Ω接地电阻的接地。
- 在着手做下列任一项工作前，总是将加在 PLC 上的电源关断 （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快闪存储器中。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。
- CJ系列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。当在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，即使在生成 AUTOEXEC.IOM 文件时设置了另外地址。此外，如果数据量超出 DM 区 （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）设置连锁电路、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 并且仅有内部输出状态维持）。

- 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200～240 V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 ～ 240 V 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失灵或燃烧。
- 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失灵。
- 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。
- 切务将高于额定输入电压的电压施加输入单元，过电压可能导致燃烧。
- 切务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，不要连接功能地端子，否则可能引起燃烧。
- 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，可能导致失灵。
- 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，可能引起失灵。
- 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。否则可能引起失灵。
- 接线时，保留粘在单元上的标签。撕去标签后如有异物落入单元会引起失灵。
- 为确保合适的散热效果，在完成全部接线后撕去标签。保留标签会有散热问题而失灵。
- 接线时使用压接端子，不要把多股线直接连到端子上，这样的连接可能引起燃烧。
- 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其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，否则将导致失灵。
- 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置置位 / 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。
- 千万不要将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。否则可能导致失灵或燃烧。
- 在接触单元前，为使人体所聚积的静电放电，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PLC 最右的单元上。没有安装端盖的 PLC 将不能正常运行。
- 如果不合适的数据链接表和参数被设置，可能导致不可预料的运行。即使已经设置了合适的数据链接表和参数，在启动或停止数据链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 规定

EMC 规程

OMRON 公司的所有装置都符合 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，其随装有 OMRON 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）

CS 系列：EN61131-2 和 EN61000-6-2

CJ 系列： EN61000-6-2

EMI （电磁干扰）：

EN50081-2

（辐射发射：10 m 规定）

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，符合 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. 直流电源与直流电源单元和I/O单元的连接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（10 m 规定）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 降低继电器输出噪声的方法

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噪声滤波器或在 PLC 外部采用合适的预防措施。

为满足标准要采取的防范措施，随负载侧设备，按线和机器配置等变化。下面为降低噪声所采取措施的例子。

预防措施

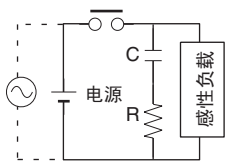
（详情参见 EN5001-2）

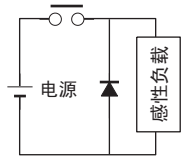
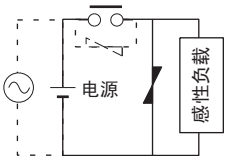
如果包括 PLC 在内的整个系统，其负载开关切换频率小于每分钟 5 次，则不需要采取预防措施。

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

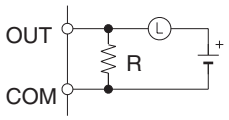
当切换感性负载时，请将浪涌保护器、二极管等与负载或触点并联，连接如下所示。

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回有一时间迟滞。 如果电源电压是 24 V 或 48 V，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。 电容器介质强度必须为 200 ～ 300 V。如果是交流回路，使用无极性的电容器。

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。 由这种方法引起延迟，在电路断开瞬间和负载重新接入瞬间之间，这个时间的延迟比由 RC 方法更长。	二极管的确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耐压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接。	---

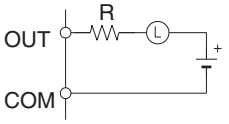
当切换一个具有浪涌电流负载如白炽灯时，抑制浪涌电流方法如下示。

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



提供限流电阻

第 1 章
CPU 单元操作

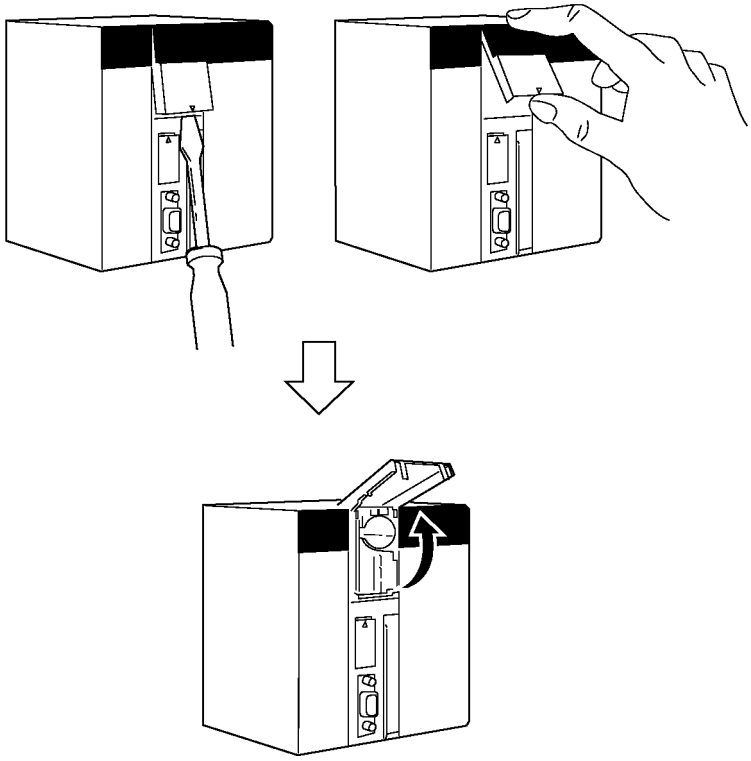
本章描述 CPU 单元的基本结构和操作。

1-1 初始化设置 （仅限于 CS1 CPU 单元）

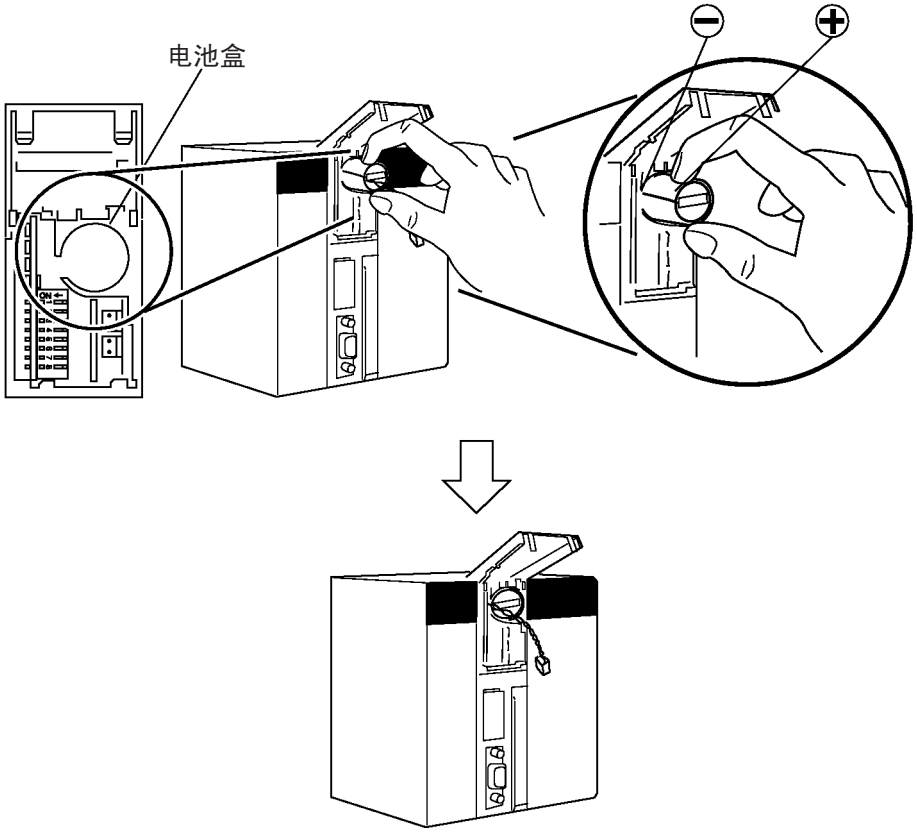
电池安装

在使用 CS1 CPU 单元前，请按下列步骤把电池安装到 CPU 单元。

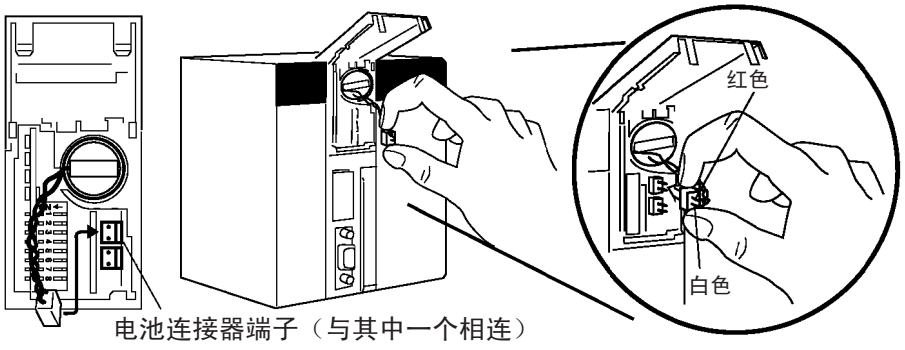
1,2,3... 1. 将平头螺丝刀插入电池盒底部的窄分缝，把盒盖向上翻打开它。



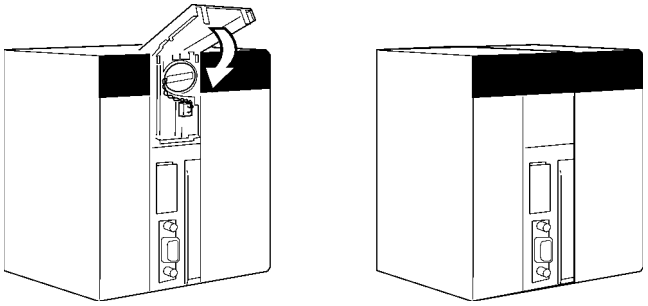
2. 拿住电池使连接电缆朝外，将电池插入电池盒。



3. 把电池连接器和电池连接器端子连接。红电线接顶部端子，白电线接底部端子。电池连接器端子有两个，将电池与两端子中任一相连，与上面或下面端子连接都可以。



4. 卷入电线并盖上盖子。



清存储器

装入电池后，采用存储器清除操作，初始化 CPU 单元内的 RAM 清除存储器内容。

手持编程器

手持编程器完成下列步骤。

注 当使用手持编程器清内存时，不能定义超过一个循环任务。用户可以定义一个循环任务和一个中断任务，或仅一个循环任务而无中断任务。若需进一步了解内存清楚操作，请查阅操作手册。*参考第 1 章 CPU 单元操作*和*第 4 章任务*有关任务的更多内容。

CX-Programmer

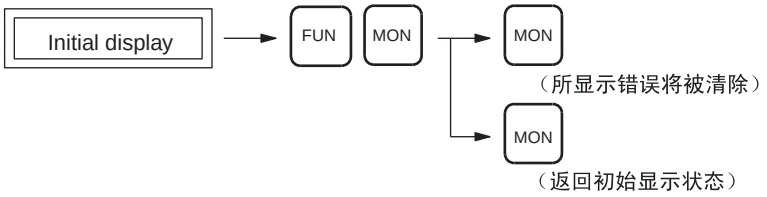
清内存也可用CX-Programmer。实际操作步骤参见CX-Programmer操作手册。

清除错误

清内存后，清除包括电池电压下降错误及 CPU 单元的所有错误。

手持编程器

用手持编程器完成下列步骤。



CX-Programmer

用 CX-Programmer 也可清除错误。实际操作步骤参见 *CX-Programmer 操作手册*。

注 当安装了内装板，即使已用 CX-Programmer 删除错误，一个内装板程序安排表错误可以继续存在。如果这种情况发生，将电源复位或重新启动内装板，然后再删除错误。（串行通信板错误时， A42407 为 ON）

1-2 内部时钟的使用 （仅限于 CS1 CPU 单元）

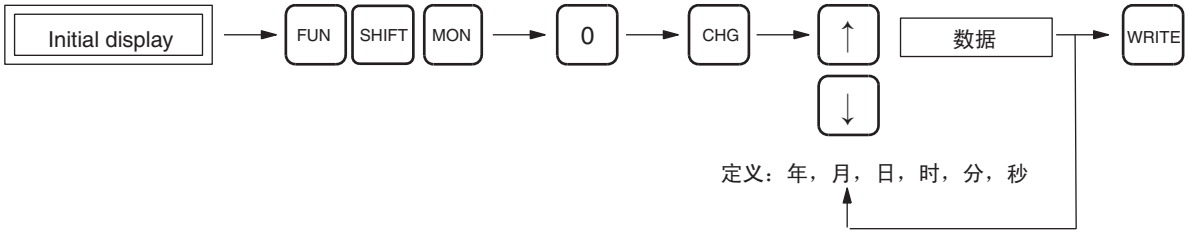
当电池装入 CS 系列 CPU 单元，CPU 单元的内部时钟设置为“00 年，01 月，01 日（00-01-01），00 点钟，00 分钟，00 秒（00：00：00），星期天（SUN）”

当装好电池，使用内部时钟时，接通电源并完成下列操作：

1）用编程装置 （手持编程器或 CX-Programmer）设定时钟时间； 2）执行 CLOCK ADJUSTMENT （DATE）指令或者 3）传送一条 FINS 命令 使内部时钟从当前正确的时间和日期开始启动。

采用手持编程器设置时钟如下操作所示。

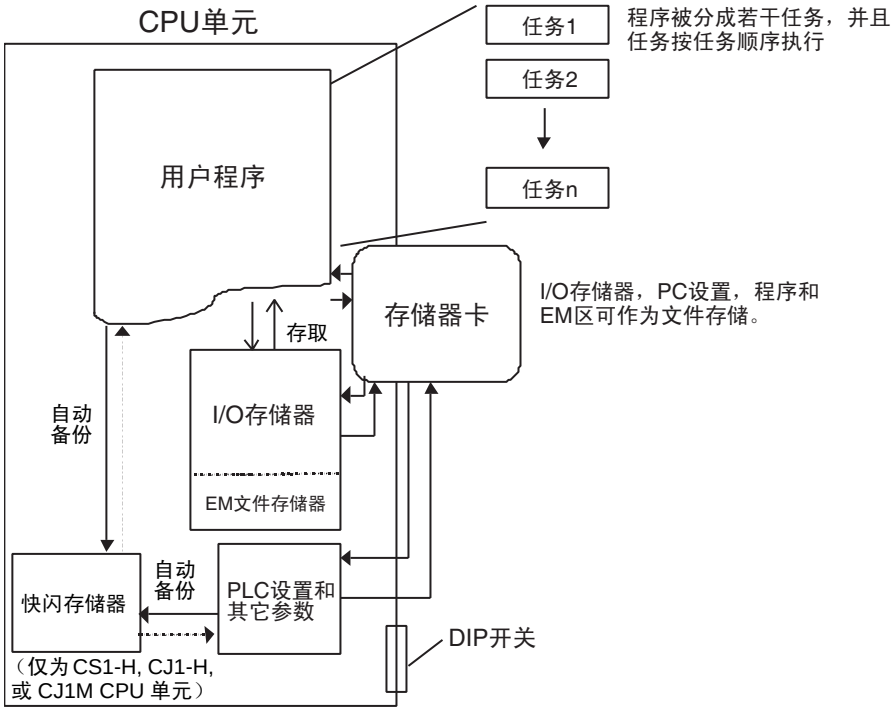
按键顺序



1-3 CPU 单元的内部结构

1-3-1 概述

下图展示了 CPU 单元的内部结构。



用户程序

用户程序可以生成包括中断任务在内的 288 个程序任务。由 CX-Programmer 编程软件将任务传送到 CPU 单元。

任务有两种。第一种是每个周期执行一次的循环任务（最多 32 个），另一种是仅当中断条件出现时才执行的中断任务（最多 256 个）。循环任务按次序执行。

注 1. 用一个 CS1-H、CJ1-H、CJ1M 或 CS1D CPU 单元，中断任务可以和循环任务一样的形式循环执行。这些我们称其为 “附加循环任务”。可以按循环方式执行的任务总数则不超过 288 个。

2. 当使用 CX-Programmer，CS1-H 或 CJ1-H CPU 单元用 2.1 或更高版本以及 CJM1 或 CS1D CPU 单元用 3.0 或更高版本。

对 I/O 存储器程序指令的读写以及从程序顶端按顺序执行 . 在所有循环任务执行后 , 所有单元的 I/O 被刷新 , 并从最低编号的循环任务开始周期重复。

若需了解 I/O 刷新详细情况 , 参考 *CS/CJ 系列操作手册* CPU 单元操作章节。

I/O 存储器

I/O 存储器是用于对用户程序读和写的 RAM 区 . 它是一个当电源按通或断开时内容被清除的存储区组成 , 而其它区域的数据将保持。

I/O 存储器也被划分成与所有单元交换数据和仅仅用于内部使用的一个区域 . 每个程序执行周期和所有单元交换一次数据 , 当特殊指令执行时也交换数据。

PLC 设置	PLC 设置是通过软件开关设置各种初始值或其它设定。
DIP 开关	DIP 开关是通过硬件开关设置各种初始值或其它设定。
存储器卡	存储器卡用于存储诸如程序、I/O 存储器数据、PLC 设置和由编程装置创建的 I/O 注释，当电源接通时，程序和各种系统设定可以由存储器卡自动写入（在启动时自动传送）。
快闪存储器（仅限于 CS1-H、CJI-H、CJ1M 或 CS1D CPU 单元）	任何时候，当用户写数据到 CPU 单元，若使用 CS1-H、CJ1-H、CJ1M 或 CS1D CPU 单元，用户程序、参数区数据如 PLC 的设置会自动地在内置式快闪存储器中备份。在没有用存储器卡时，则恢复无电池操作。I/O 存储器，包括大部分 DM 区，在没有电池时是不备份的。

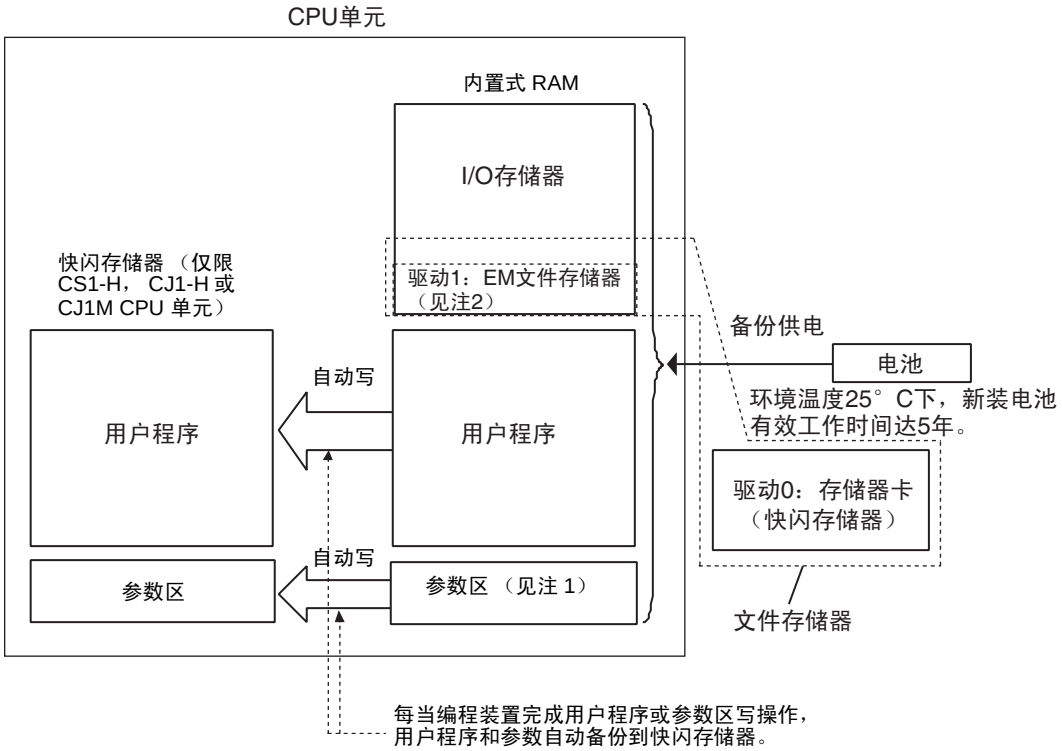
1-3-2 CPU 单元存储器的框图

CPU 单元存储器（RAM）由下列 CS/CJ 系列模块组成：

- 参数区（PLC 设置、 I/O 登记表、路由表和 CPU 总线单元设定）
- I/O 存储区
- 用户程序

在参数区和 I/O 存储器数据由电池支持（CS 系列：CS1W-BAT01， CJ1-H：CPM2A-BAT01），如果电池电源电压低，参数会丢失。

然而，CS1-H，CJ1-H，CJ1M 或 CS1D CPU 单元提供一内置式快闪存储器作数据备份。每当用户用编程装置（例如 CX-Programmer 或手持编程器）将数据写到 CPU 单元，包括下列操作：数据传送、在线编辑、从存储器卡传送等，用户程序和参数区数据会自动地在内置式快闪内存中作备份。这意味如果电池电压降低，用户程序和参数区数据也不会丢失。



注 1. 将 CPU 单元前面的 DIP 开关第一脚打到 ON 位置，参数区和用户程序（例如用户存储器）可为写保护状态。

2. 在 PLC 设置时 EM 文件存储器是转换成文件存储器 EM 区的一部分。所有 EM 存储单元，从规定的存储器单元到 EM 区的最后单元，只能作为文件存储器用于存储数据和程序文件。

3. 初次使用 CS1 CPU 单元前，请确认所提供的电池（CS1W-BAT01）已装入。装入电池后，用编程装置清 PLC 的 RAM（参数区、I/O 存储区和用户程序区）内容。

4. 当 CS1-H、CJ1、CJ1-H、CJ1M 或 CS1D CPU 单元出厂时，电池已经装入无需清内存和设置时间。

5. 当数据正写入快闪存储器时，CPU 单元前面的 BKUP 指示器会点亮。在备份操作尚未完成（例如 BKUP 指示器还亮着），不要断开电源。详情参考*第 6-6-10 快闪存储器*内容。

1-4 操作模式

1-4-1 操作模式的描述

下列操作模式适用于 CPU 单元，这些模式可控制整个用户程序并对所有任务有效。

编程模式

在编程模式下程序不执行，RUN 指示器不点亮。这种模式用于程序编辑或用于如下所列执行程序的准备工作：

- 登记 I/O 表。
- 改变 PLC 设置或其它设定。
- 传送和检查程序。
- 强制置位和复位检查接线及位分配。

在这种模式下，所有循环和中断任务不执行 （INI），即处于停止状态。有关任务的详细内容参考*第 1-6 任务的描述*。

在 PROGRAM 模式下运行 I/O 刷新。有关 I/O 刷新内容，参见操作手册。

！ 警告 即使程序未运行（例如在PROGRAM模式下）CPU单元刷新I/O，在对I/O单元，特殊 I/O 单元或 CPU 总线单元任何部分存储器分配的状态改变前，先确定绝对安全。任何单元数据分配的任何变化，可能在与负载连接的单元导致不可预料的运行。下列中任何操作可能导致存储器状态改变：

- 从编程装置向 CPU 单元传送 I/O 存储器数据。
- 由编程装置改变存储器中的当前值。
- 由编程装置强制置位 / 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。
- 从主计算机或网络中的其它 PLC 传送 I/O 存储器。

MONITOR 模式

当程序在 MONITOR 模式下执行时，下列操作该由编程工具完成。RUN 指示器点亮，此模式被用于测试运行和其它调整。

- 在线编辑。
- 强制置位和复位。
- 改变 I/O 存储器中数值。

在这模式下，用于启动执行的循环任务被逐一登记 （见注），且它们由 TKON（820）使其成为可执行任务。当程序执行到它们的任务编号时，就执行这些任务。如果中断条件出现，中断任务将执行。

注 在启动执行的任务由 CX-Programmer 在程序特性中登记。

RUN 模式

这种模式用于一般程序执行。RUN 指示器点亮。在这种模式下，一些编程工具对于类似在线编辑、强制置位 / 复位、I/O 存储器数值改变操作是不允许的，但另外一些编程工具对类似于程序执行状态监视 （监视程序和 I/O 存储器）是允许的。

一般系统运行用此模式。任务执行与在 MONITOR 模式下一样。对于各模式下允许情况详情，请参考*操作手册 10-2 CPU 单元操作模式*章节。

1-4-2 I/O 存储器的初始化

下表给出了当操作模式从 PROGRAM 模式转换到 RUN/MONITOR 模式或相反模式方式相反转换时 , 哪些数据区将被清除内容。

模式转换	非保持区域 （注 1）	保持区域 （注 2）
RUN/MONITOR → PROGRAM	清除 （注 3）	保持
PROGRAM → RUN/MONITOR	清除 （注 3）	保持
RUN ↔ MONITOR	保持	保持

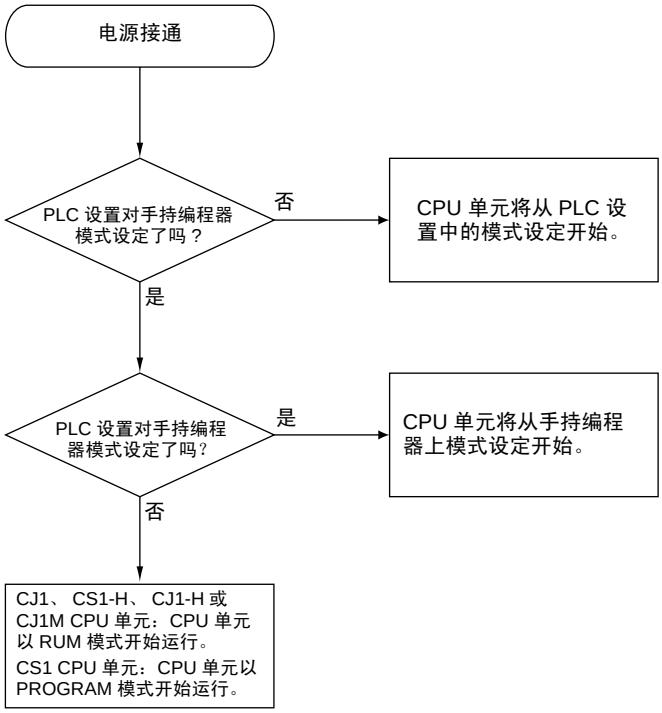
注
1. 非保持区域 :CIO 区、工作区、定时器当前值、计时器计时到标志、变址寄存器、数据寄存器、任务标志和条件标志。（在辅助区中的一些状态地址有些保持不变，有些被清除）。
2. 保持区域 : 保持区、数据区、 EM 区、计数器当前值、计数器计数到标志。
3. 当 IOM 保持位（A50012）为 ON，I/O 存储器中的数据被保持。当 IOM 保持位（A50012）为 ON，并且由于致命错误（包括 FALS（007）），而停止操作， I/O 存储器的内容将保持，而输出单元上的输出将被关断。

1-4-3 启动模式

有关设置 CPU 单元起动模式详细资料参见操作手册。

注 如手持编程器未连接，CJ1、CS1-H、CJ1-H、CJ1M 或 CS1D 的 CPU 单元将以 RUN 模式开始运行。但与 CS1 CPU 单元缺省操作不同，如果手持编程器未连接， CS1 CPU 单元将以 PROGRAM 模式运行。

条件	CS1 CPU 单元	CJ1, CS1-H, 或 CJ1-H， CJ1M，CS1D CPU 单元
PLC 设置是根据手持式编程器的运行模式设定而定，除非是手持编程器未连接	PROGRAM 模式	RUN 模式



1-5 程序和任务

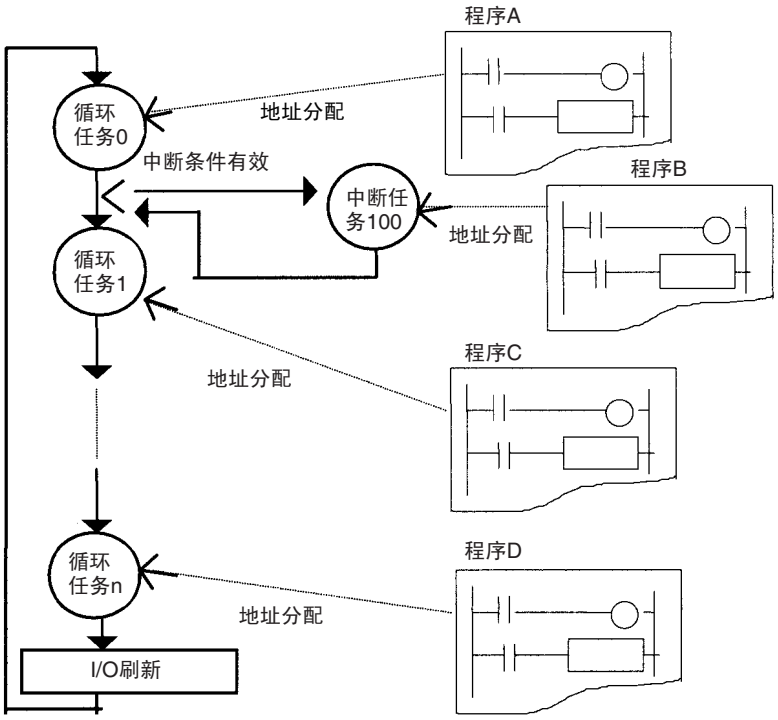
任务规定执行独立程序的中断条件和顺序．它们粗略地划分为下列类型：

1,2,3...

1. 以顺序执行的任务称之为循环任务。
2. 按中断条件执行的任务称之为中断任务。

注 采用 CS1-H、CJ1-H、CJ1M 或 CS1D CPU 单元，中断任务可以和循环任务一样循环执行。这样的中断任务称之为 “附加循环任务”。

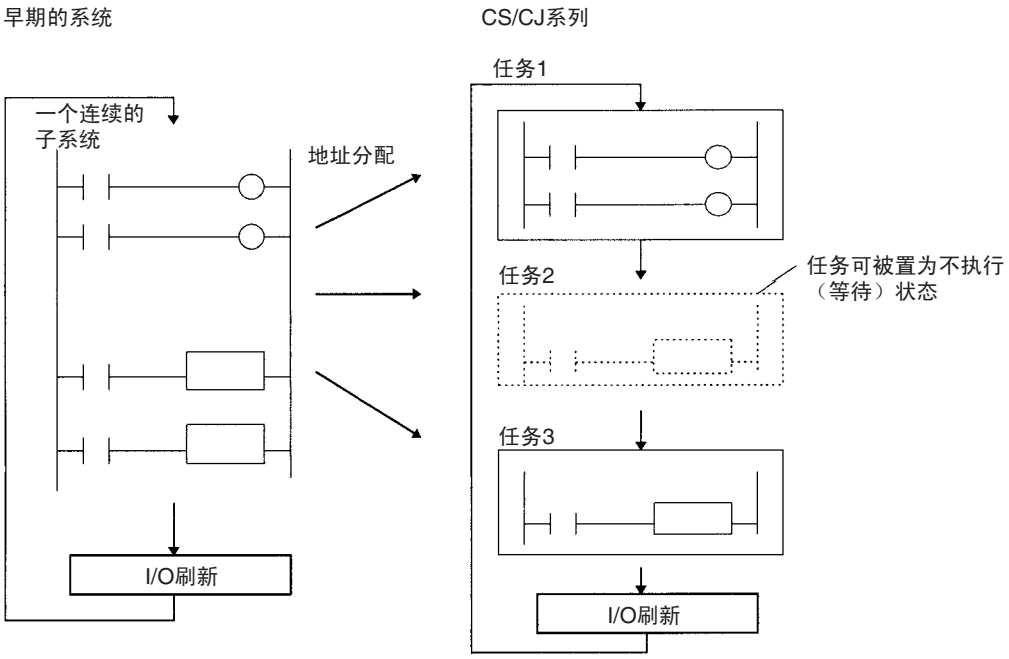
规定为循环任务的程序将按任务编号顺序执行，在所有任务（更切确地说任务处于允许执行状态）执行后，I/O 在每个周期刷新一次。如果在循环任务处理过程中，一个中断条件有效，循环任务将被中断执行，并且程序跳到将要执行的中断程序处。有关I/O刷新内容参考*CS/CJ系列操作手册*的CPU单元操作章节。



在上例中，程序将按下列顺序执行：A 开始 → B → 回到 A → C → D。这个执行顺序是在执行程序 A 时中断任务 100 的中断条件产生。当程序 B 执行完毕，程序 A 从程序执行被中断处继续执行余下未执行的程序 A 部分。

对早期的 OMRON PLC，一个连续的程序由几个连续部分形成。就如同早期的 PLC，现在的程序把单个用 END 指令结束的程序划分为每一个任务。

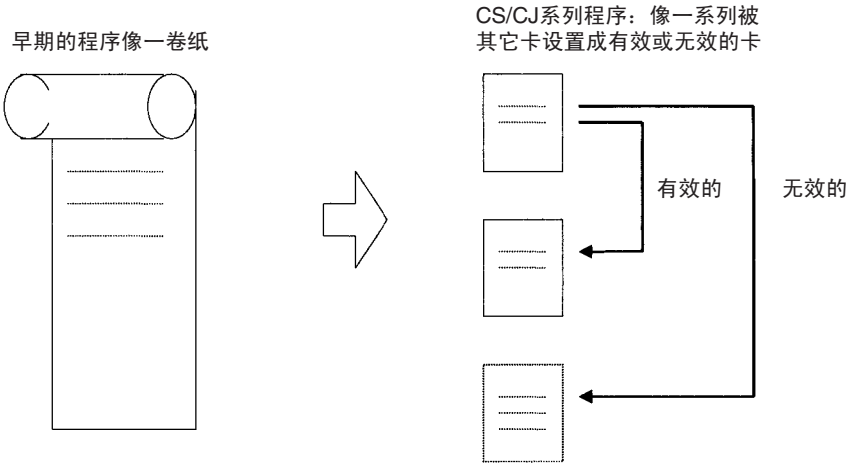
循环任务的一个特性是它们可以由任务控制指令控制允许 （执行状态）和不允许 （等待状态）执行。这就意味几个程序块可组成一个任务，这时它们可根据当前生产模式或正完成的步骤（程序步切换），规定为可执行的程序（任务）。所以操作 （扫描周期）可因为程序仅在需要时再执行而大大改进。



一个已生效可执行的任务将在连续的循环中执行，而一个处于等待状态的任务在连续循环中仍保持等待，除非其它任务使它重新生效。

注 任务可比喻成从一系列各自独立的卡片上读，不象早期的程序，它可比喻成从一卷纸上读。

- 根据预设定的顺序，从编号最小的卡开始读所有的卡。
- 所有卡被指定为有效的或无效的，无效的卡将被跳过不读 （由任务控制指令确定卡是否有效）。
- 一张有效的卡将保持有效，并将在连续的顺序中被读取。一张无效的卡将保持无效，在其它卡将其变为有效前，无效的卡将被跳过不读取。



1-6 任务描述

任务被粗略地划分为下列类型：

1,2,3... 1. 循环任务 （最多 32 个）。

如果可执行，每个循环周期执行一次。如果需要，循环任务也可被改为不允许执行。

2. 中断任务

不管循环任务是否正在执行，当中断信号出现，中断任务就被执行。中断任务 （见注 1 和注 2）被划分为下列队种 （5 种包括 CS1-H、 CJ1-H、CJ1M 或 CS1D CPU 单元的附加循环任务）：

a) 电源中断任务：当电源中断时执行 （最多一个）；

b) 定时中断任务：在规定的时间间隙执行 （最多 2 个）

c) I/O 中断任务 （注）： 当一个中断输入单元输入变为 ON （最多 32 个）时执行；

d) 外部中断任务（注）： 当一个特殊 I/O 单元、CPU 总线单元或内部板子 （仅 CS 系列）提出中断申请时执行。

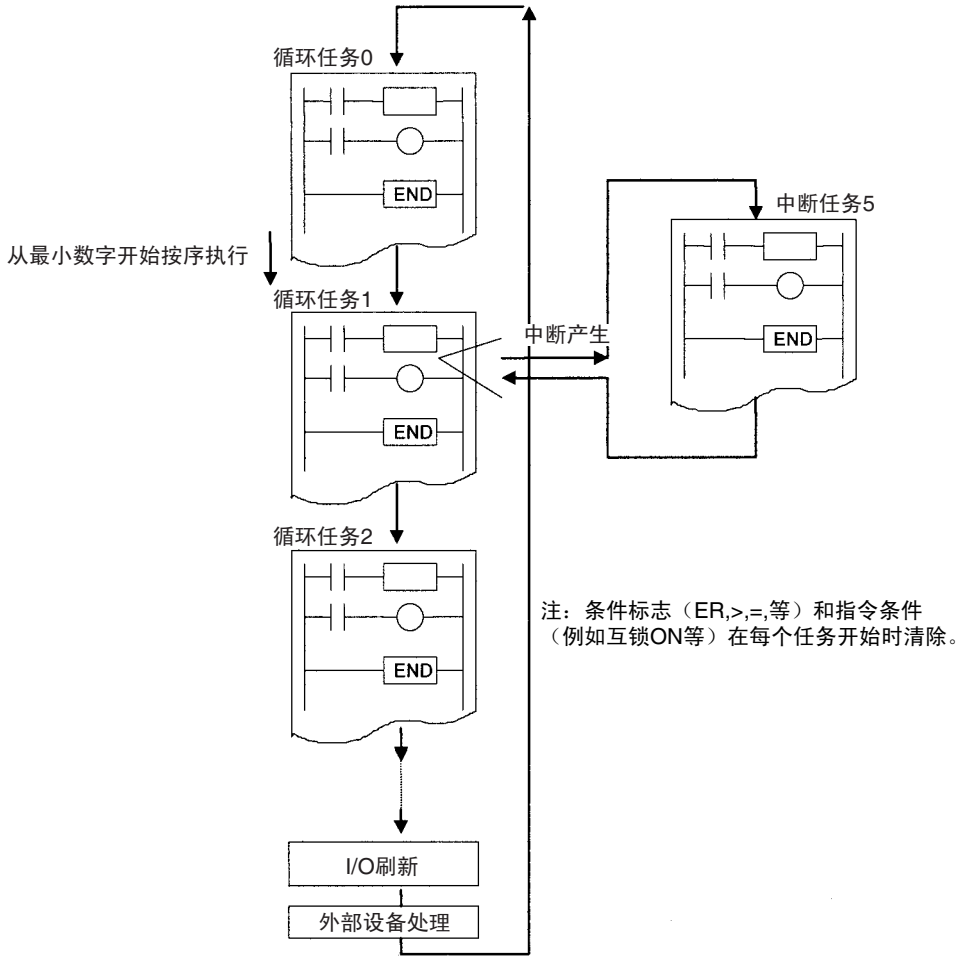
e) 附加循环任务（CS1-H，CJ1-H，CJ1M和CS1D CPU单元支持 ）：
附加循环任务被看成是循环任务功能的中断任务，只要它们的执行条件满足，每个扫描周期被执行一次。

用 CX-Programmer 可以产生和控制总额为 288 个任务程序。这些任务包括 32 个循环任务和 256 个中断任务。

注 1. 目前 CJ1 CPU 单元不支持 I/O 中断任务和外部中断任务，这样， CJ1 CPU 单元最多任务数 35 个 （32 个循环任务和 3 个中断任务）。产生和管理的总的程序数也就是 35 个。

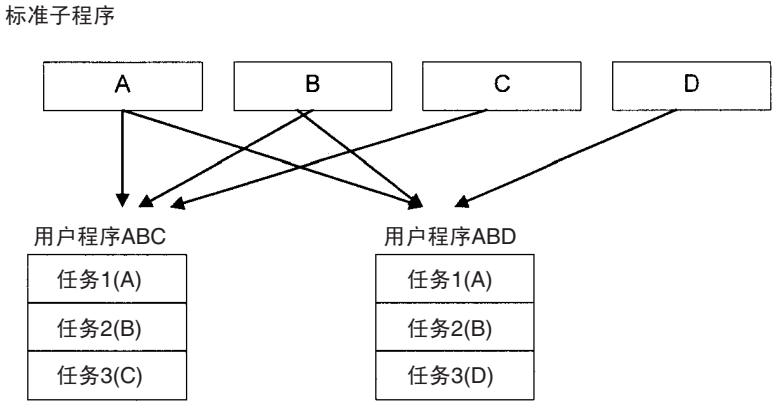
2.CS1D CPU 单元不支持任何中断任务。然而，CS1D CPU 单元中断任务可用作附加循环任务。

用 CX-Programmer 对各程序特性设置值设定并对每个程序 1：1 分配给一个任务。



程序结构

生成标准的子程序，并根据需要把子程序并划归为任务。这意味程序可由模块生成（标准元件），并且那些任务（模块）可单独调试。



当生成一个模块化标准程序，地址可用符号来规定，以简化标准。

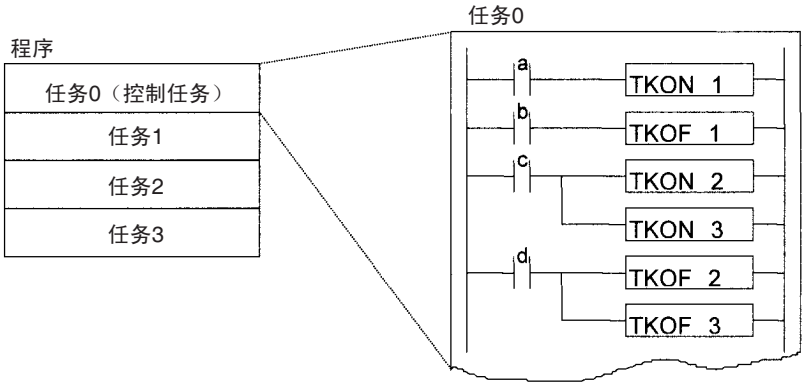
可执行和等待状态

任务可执行和等待指令 (TKON(820) 和 TKOF(821)) 可在一个任务中执行 , 用于设定另一个任务可执行或等待状态。处于等待状态任务中的指令将不执行，但它们的 I/O 状态将保持不变。当一个任务返回到可执行状态，那 I/O 状态保持的指令将被执行。

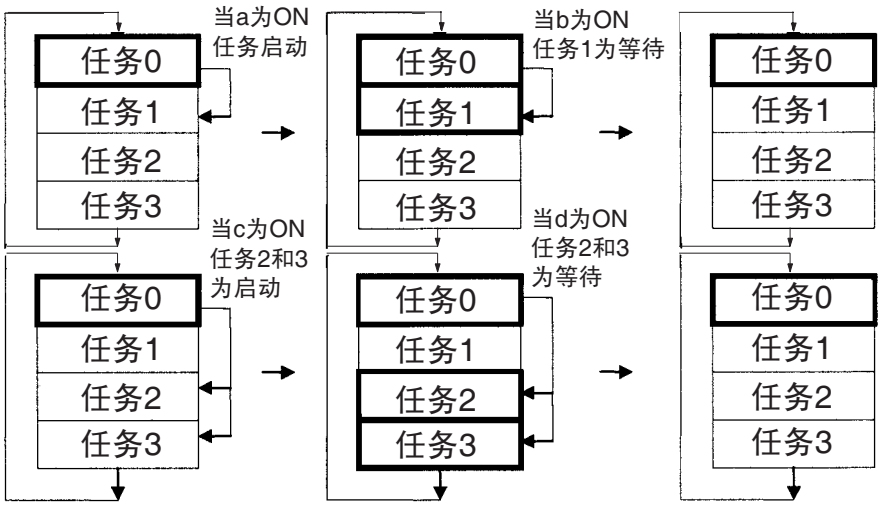
例 : 带控制任务的程序

在例子中任务 0 是在运行开始时首先执行的控制任务。其它任务可由 CX-Programmer（但不是手持编程器）设定，在开始运行时启动或不启动。

一旦程序执行开始，任务可由 TKON （820）和 TKOF （821）控制。

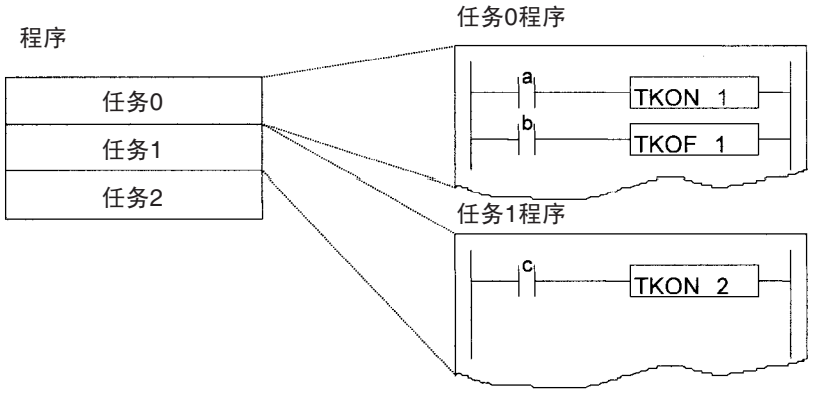


例： 任务O在运行开始时设定为已执行
（由CX-Programmer在程序属性中设定）。
当a为ON,任务1为可执行。
当b为ON,任务1为等待。
当c为ON,任务2和3为可执行。
当d为ON,任务2和3为等待。

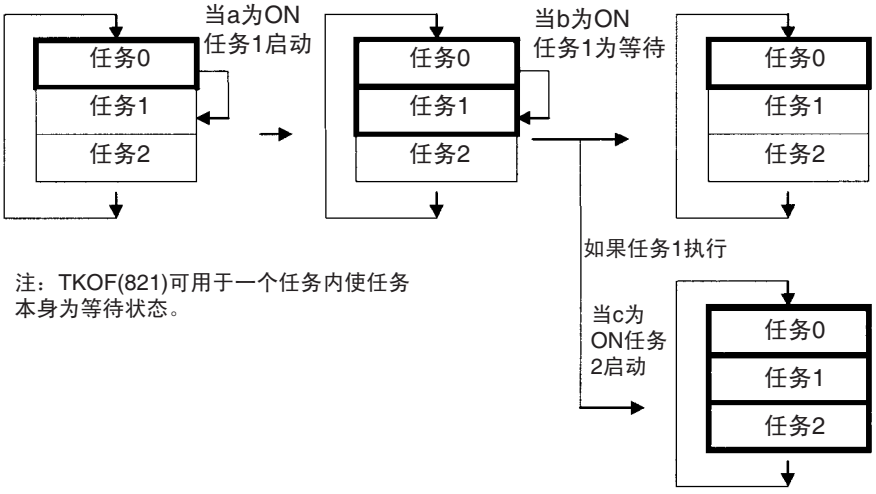


例 : 每一任务由另一任务控制

在这个例子中，每一任务都由另外一个任务控制。



例：任务0设定在运行开始是已无条件地完成。
当a为ON，任务1为允许执行。
当b为ON，使任务1为等待。
当c为ON，并且任务1已执行，任务2为允许执行。



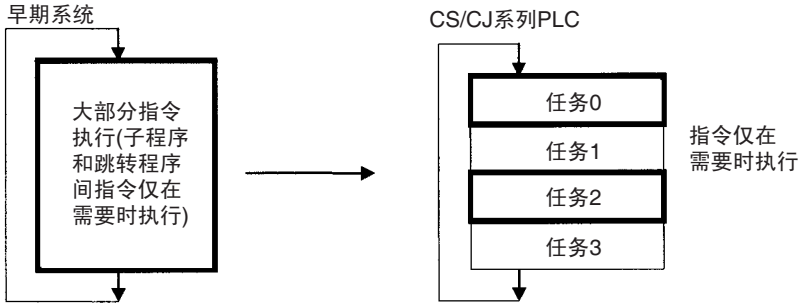
注：TKOF(821)可用于一个任务内使任务本身为等待状态。

任务执行时间

当一个任务为等待状态，在这任务中指令不执行，所以，它们为 OFF 的指令执行时间将不加入周期时间。

注 从这等待点，一个等待任务中指令就如同在跳转程序部分（JMP-JME）中的指令。

因为处于不执行任务指令执行时间不加入周期时间中，将系统分成一个总的控制任务和仅当需要时才执行的数个独立任务，这样整个系统性能可得到明显提高。



第 2 章
编程

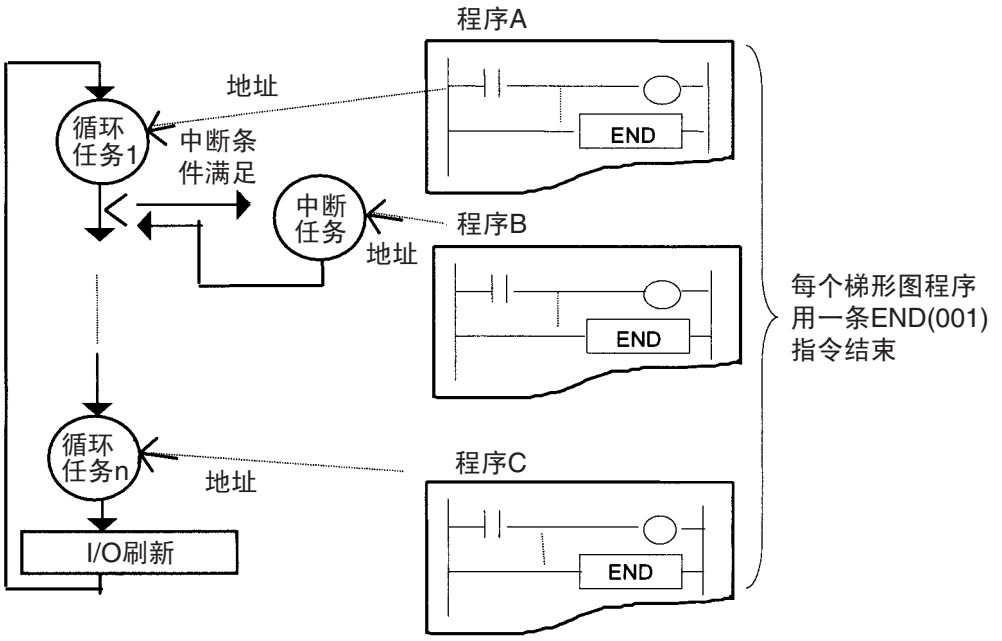
本章为写、检查和输入程序所需要的基本知识。

2-1 基本概念

2-1-1 程序和任务

CS/CJ 系列 PLC 执行在任务中的梯形图程序，如同常规的 PLC 一样。在每个任务中的梯形图程序用一个 E ND （001）指令结束。

任务被用于确定执行梯形图程序的顺序，以及执行中断的条件。



本节描述了写 CS/CJ 系列程序的基本概念，为获得更多有关任务以及它们和梯形图程序间关系的知识，参见*第 4 章任务内容*。

注 **任务和编程装置**

任务管理如下面编程装置所论述。*参考第 4-4 章关于任务的编程装置操作*。为获得更多信息，请参考 *CS/CJ 系列手持编程器操作手册（W341）和 CX-Programmer 操作手册*。

CX-Programmer

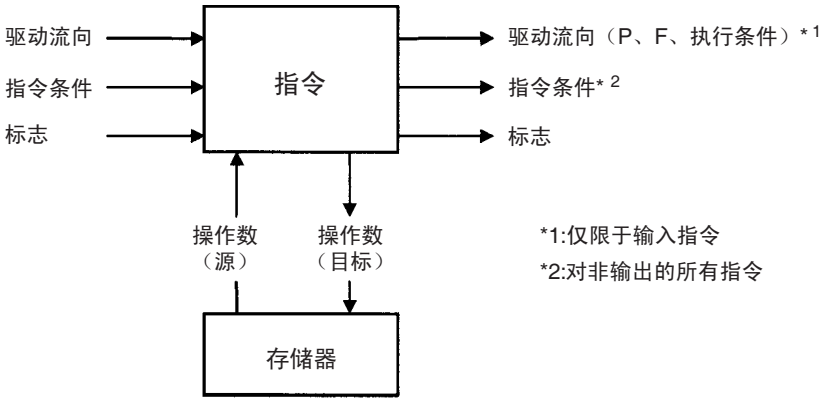
CX-Programmer 用于指定各独立程序的属性，即：任务类型和任务编号。

手持式编程器

在手持式编程器上可对定义好的 CT00 到 CT31 循环任务，IT00 到 IT255 中断任务读 / 写和编辑程序。当用手持式编程器完成存储器清除操作，仅循环任务 0（CT00）可写入新程序。使用 CX- Programmer 创建循环任务 1 到 31（CT01 到 CT31）。

2-1-2 指令的基本知识

程序由指令组成。一条指令从输入到输出的基本结构如下图所示。

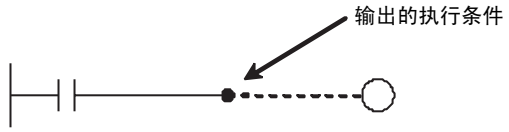


驱动流向

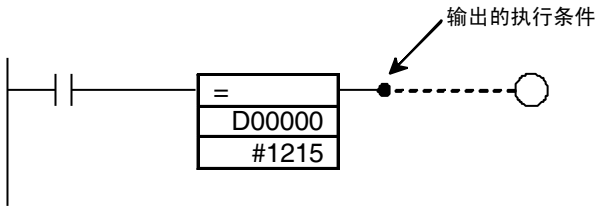
通常当程序执行时，驱动流向是用于控制执行和指令的执行条件。在梯形图中，驱动流向表示执行的状态。

输入指令

- 取指令表示一个逻辑开始和输出的执行条件。

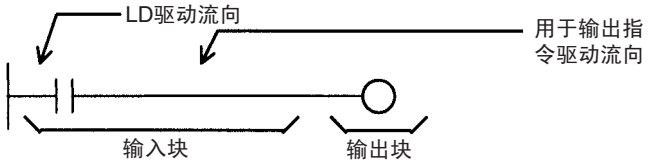


- 中间的指令输入驱动流向作为执行条件和对中间和输出指令的输出驱动流向。



输出指令

用驱动流向作为执行条件，输出指令执行所有功能。



指令条件

指令条件是一些特殊条件，通过下列指令的输出影响各处指令的执行。当指令条件处于决定是否执行一条指令时，它比驱动流向具有更高的优先权。根据指令执行条件，一条指令可能变成不执行或可能以不同方式执行。

在任务开始时指令条件复位（取消），例如任务改变时它们被复位。

下列指令成对使用，用于设定和取消指令条件。这些成对指令必须在同一任务中使用。

指令条件	描述	设定指令	取消指令
内部锁存	内部锁存可锁定一部分程序：锁定时，输出位变 OFF、计时器复位、计数器保持。	IL(002)	ILC(003)
BREAK(514) 执行	执行过程中以 FOR （512）一 NEXT （513）结束 （直到 NEXT （513）指令前的所有指令都不执行）。	BREAK(514)	NEXT(513)
	执行一条 JMP0 （515）到 JME0 （516）跳转指令。	JMP0(515)	JME0(516)
块程序执行	从 BPRG （096）到 BEND （801）执行程序块。	BPRG(096)	BEND(801)

标志

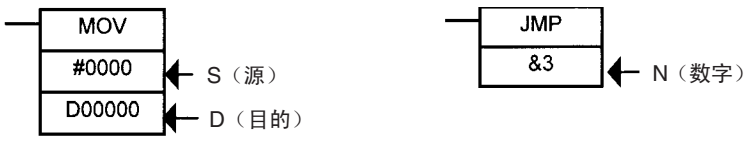
在这里，标志是在指令间作为连接的一个位。

输入标志	输出标志
• 微分标志 微分结果标志。这些标志的状态是自动地输入用于所有上升沿 / 下降沿微分输出指令，以及 DIFU （13） /DIFD （14）指令。 • 进位标志 进位标志在数据移位指令和加法 / 减法指令中用作非定义的操作数。 • 特殊指令标志 特殊指令标志包括用于 FPD （269）指令示教位标志和网络通讯使能标志。	• 微分标志 微分结果标志。这些标志的状态是从所有上升沿 / 下降沿微分输出指令和 UP （521） /DOWN （522）指令自动地输出。 • 条件标志 条件标志包括 ON/OFF 标志，以及那些由指令执行结果更新的标志。在用户程序中，这些标志可由符号表示，例如 ER， CY，>， =， A1， A0 等而不用地址表示。 • 用于特殊指令的标志 这些指令包括存储器卡指令标志和 MSG （046）执行完成标志。

操作数

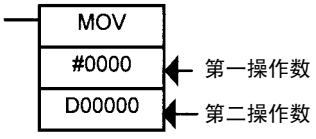
操作数定义预置指令参数 （梯形图中的框），它用于定义 I/O 存储器内容或常数。操作数地址或常数输入后，指令就能被执行。操作数分成源操作数，目的操作数或数字操作数。

例：



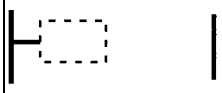
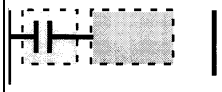
操作数类型		操作数符号	说明	
源	定义要读取数据的地址或常数	S	源操作数	源操作数不同于控制数据 （C）
		C	控制数据	在源操作数中的复合数据，它取决于位状态含义各不同。
目的 （结果）	定义要写入数据处的地址	D (R)	---	
数字	定义用于指令中的特殊数字，例如一条跳转指令的跳转号码或子程序的程序号。	N	---	

注 从指令的最上面开始，操作数也称作为第一操作数，第二操作数，依此类推。



2-1-3 指令位置和执行条件

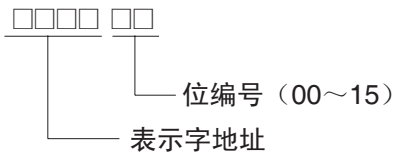
下表给出了指令允许的位置。指令分成需要执行条件的指令和不需要执行条件的指令。各指令的详细情况参见*第 3 章指令功能*。

指令类型		允许位置	执行条件	梯形图	例
输入指令	逻辑开始（输入取指令）	直接与左面母线或指令块的开始连接	不需要		LD，LD TST（350），LD >（及其它符号比较指令）
	中间指令	在逻辑开始和输出指令间	需要		AND，OR，AND TEST（350），AND >（及其它 ADD 符号比较指令） UP(521),DOWN(522), NOT(520)等。
输出指令		直接与右面母线连接	需要		大多数指令包括 OUT 和 MOV（021）指令。
			不需要		END（001），JME（005），FOR（512），ICL（003）等。

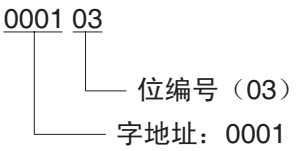
1. 另外一类指令的执行，基于单个输入上的一组助记符的指令，这类指令称作块程序指令。细节参阅 *CS/CJ 系列 CPU 块程序指令参考手册*。
2. 如果一条指令需要执行条件，而直接连接在左母线上，没有逻辑开始指令，当在编程装置（CX-Programmer 或手持编程器）上检查程序时，程序出错信号会出现。

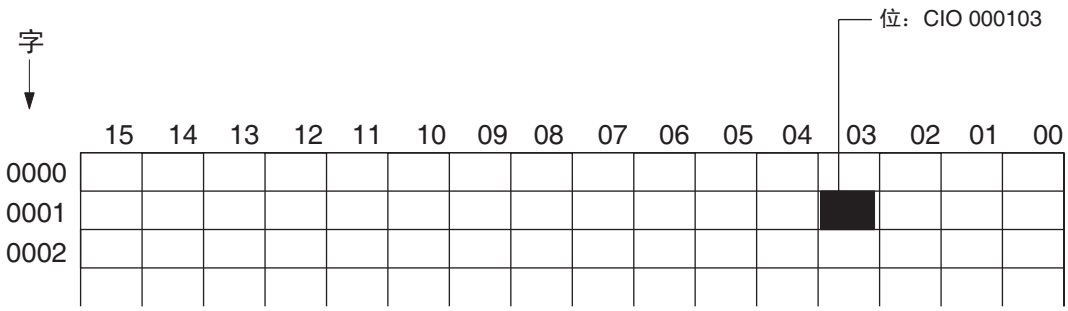
2-1-4 I/O 存储区寻址

位地址

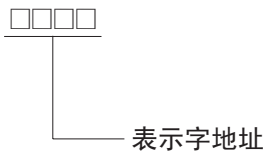


例：在 CIO 区字地址 0001，位地址 03 的地址如下所示，在本手册中，这个地址表示为 “CIO 000103”。

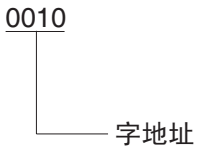




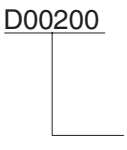
字地址



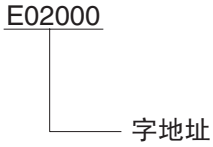
例：在 CIO 区字 0010 中位 00 到 15 的地址如下所示，在本手册中这个地址表示为 CIO 0010。



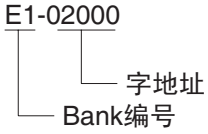
DM 和 EM 区地址在地址前加前缀 "D" 或 "E"，如下地址 D00200 所示。



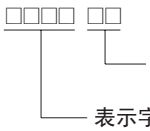
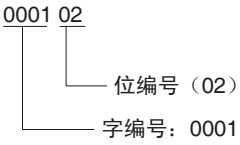
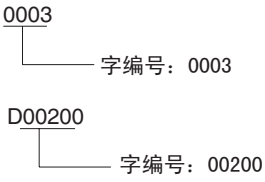
例：在扩展数据存储器的当前 Bank 中字 2000 的地址可表达为如下：

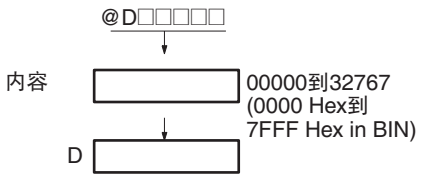
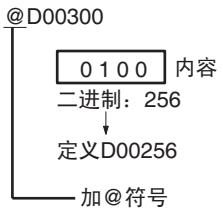
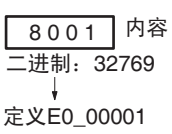
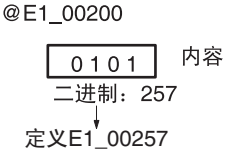
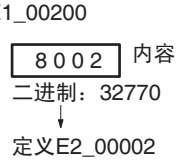


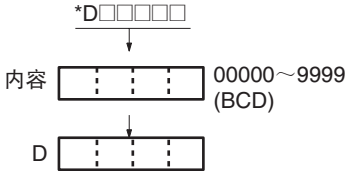
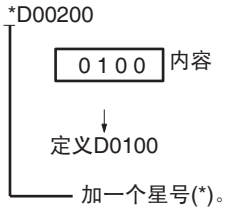
在扩展数据存储器的 1 号 Bank 中字 2000 的地址可表达为：



2-1-5 定义操作数

操作数	描述	注释	应用举例
定义位地址	用字和位编号直接定义一位地址  注 访问计时/计数器完成标志和当前值采用相同地址，任务标志也只有一个地址。		
定义字地址	字的编号直接用于定义16位的字地址。 		MOV 0003 D00200

<table>
<tr><th>操作数</th><th>描述</th><th>注释</th><th>应用举例</th></tr>
<tr><td rowspan="5">以二进制形式定义 DM/EM 间接地址</td><td>从数据区开始定义一个相对地址。在DM或EM前加一个@前缀以表示用二进制定义一个间接地址。地址的内容将作为二进制数据（00000到32767）在数据存储器（DM）中或扩展数据存储器（EM）中定义一个字地址。
 
</td><td></td><td></td></tr>
<tr><td>1) 如果 @D （□□□□□）的内容在 0000 Hex ~7FFF Hex （000000 ~ 32767）之间，定义了 D00000 到 D32767 之间相应的字。</td><td> 
</td><td rowspan="2">MOV #0001
@D00300</td></tr>
<tr><td>2) 如果 @D （□□□□□）的内容在 8000 Hex ~7FFF Hex （32768 ~65535）之间，定义了扩充数据存储器 （EM）单元 E0_00000 ~E0_32767 之间的相应字。</td><td> 
</td></tr>
<tr><td>3) 如果@E□_□□□□□的内容在0000 Hex ~7FFF Hex （000000 ~ 32767）之间，定义了 E □ _00000~ E □ _32767 之间相应的字。</td><td> 
</td><td rowspan="2">MOV #0001
@E1_00200</td></tr>
<tr><td>4) 如果@E□_□□□□□的内容在8000 Hex ~7FFF Hex （32768 ~65535）之间，定义了 E （□ +1） _00000~E （□ +1） _32767 （在下一个 EM 单元）之间相应的字。</td><td> 
</td></tr>
<tr><td colspan="3">注 当用二进制方式定义一个间接地址时，数据存储区 DM 和扩展数据存储区 （EM）（Bank 0 ~C）当作连续存储地址处理。如果带 @ 符号地址的内容大于 32767，那么这地址就会定义在 Bank 号为 0 从 00000 开始连续的扩展数据存储区 （EM）内。
例：如果数据存储区 （DM）字的内容为 32768，扩展存储器 （EM） Bank 0 中的 E1-00000 字被指定。
注 如果扩展存储器 （EM） Bank 号定义为 “n”，字的内容大于 32767，那么地址就会定义在 Bank 号为 N+1 从 00000 开始连续的扩展数据存储区 （EM）内。
例：如果扩展数据存储器 Bank 号为 2, 内容为 32768，扩展数据存储器 Bank 号为 3 中的 E3-00000 被定义。</td></tr>
</table>

操作数	描述	注释	应用举例
BCD 码定义 DM/EM 间接地址	从数据区开始定义一个在DM或EM前加一星号（*）以数据存储器BCD码定义一个间接地址。地址的内容将作为BCD码数据（0000～9999）在数据存储器（DM）中或扩展数据存储器（EM）中定义一个字地址。 		MOV #0001 *D00200

操作数	描述	注释	应用举例	操作数
直接定义一个寄存器	由定义 IR □ （□：0~15）或 DR □ （□：0~15）直接定义一个变址寄存器 （IR）或数据寄存器 （DR）		IR0 IR1	MOVR 000102 IR0 把 CI0 0001 的位 02 内部 I/O 内存地址储存到IR0 中。 MOVR 0010 IR1 把 CI0 0010 的内部 I/O 内存地址储存到 IR1 中。
采用寄存器定义一个间接地址	间接地址（无偏移）	PLC 内存位或字地址由 IR □内容定义。在指针寄存器前放逗号 （,IR □）定义指令操作数。	,IR0 ,IR1	LD，IR0 取位地址在IR0中，位的状态。 MOV #0001,IR1 把 #0001 传送到 PLC 内存字中，字地址在 IR1 中。
	常数偏移	将 IR □中内容加或减一常数作为 PLC 内存位或字地址。 用 +/- 常数， IR □来定义。偏移常数范围从 -2048~+2047 （十进制）。当指令执行时，偏移值转换为二进制。	+5,IR0 +31,IR1	LD +5,IR0 取 PLC 内存位状态，位地址为 IR0 内容加5。 MOV #0001 +31,IR1 把 #0001 传送到 PLC 内存字中，字地址为 IR1 内容加 31。
	DR 偏移	将 DR □内容加上 IR □内容作为 PLC 内存位或字地址。 用 DR □， IR □定义， DR 数据寄存器的内容作带符号二进制处理，如果带符号二进制值为负数， IR □的内容作负偏移。	DR0 ,IR0 DR0 ,IR1	LD DR0,IR0 取 PLC 内存位状态，位地址为 IR0 内容加DR0内容。 MOV #0001 DR0,IR1 把 #0001 传送到 PLC 内存字中，字地址为 IR1 内容加 DR0 内容
	自动递增	将 IR □中内容自动加 1 或加 2 作为 PLC 内存一个地址。 +1：定义为，IR □ + +2：定义为， IR □ ++	,IR0 ++ ,IR1 +	LD,IR0++ 取 PLC 内存位状态，位地址为 IR0 内容加递增 2。 MOV #0001 ,IR1+ 把 #0001 传送到 PLC 内存字中，字地址为 IR1 内容递增 1。
	自动递减	将 IR □中内容自动减 1 或减 2 作为 PLC 内存一个地址。 -1：定义为，-IR □ -2：定义为， - -IR □	,– –IR0 ,–IR1	LD,- -IR0 取 PLC 内存位状态，位地址为 IR0 内容加递减 2。 MOV #0001 ,-IR1 把 #0001 传送到 PLC 内存字中，字地址为 IR1 内容递减 1。

<table>
<tr><th>数据</th><th>操作数</th><th>数据格式</th><th>符号</th><th>范围</th><th>应用举例</th></tr>
<tr><td rowspan="4">16 位常数</td><td rowspan="3">所有二进制数据或一定范围内的二进制数据</td><td>不带符号二进制</td><td>#</td><td>#0000 ～ #FFFF</td><td>---</td></tr>
<tr><td>带符号十进制</td><td>±</td><td>–32768～+32767</td><td>---</td></tr>
<tr><td>不带符号十进制</td><td>& （见注）</td><td>&0 ～ &65535</td><td>---</td></tr>
<tr><td>所有 BCD 码数据或一定范围内的 BCD 码数据</td><td>BCD</td><td>#</td><td>#0000 ～ #9999</td><td>---</td></tr>
<tr><td rowspan="4">32 位常数</td><td rowspan="3">所有二进制数据或一定范围内的二进制数据</td><td>不带符号二进制</td><td>#</td><td>#00000000 ～ #FFFFFFFF</td><td>---</td></tr>
<tr><td>带符号十进制</td><td>+</td><td>–2147483648 ～ +2147483647</td><td>---</td></tr>
<tr><td>不带符号十进制</td><td>& （见注）</td><td>&0 ～ &429467295</td><td>---</td></tr>
<tr><td>所有 BCD 码数据或一定范围内的 BCD 码数据</td><td>BCD</td><td>#</td><td>#00000000 ～ #99999999</td><td>---</td></tr>
<tr><td rowspan="2">文本字符串</td><td colspan="2">描述</td><td>符号</td><td>例</td><td>---</td></tr>
<tr><td colspan="2">文本字符串数据以 ASCII 码方式存放（除特殊字符外，一个字符占一个字节）顺序从最左位到最右位和从最右位（最低位）到最左位。
如果字符数为奇数个，那么就将 00 Hex （零码）存放到最后一个字最低位上。 如果字符数为偶数个，那么就将 00 Hex （2 个零码）最后一个字后面空着的最低和最高位上。</td><td>---</td><td>'ABCDE'
<table><tr><td>'A'</td><td>'B'</td></tr><tr><td>'C'</td><td>'D'</td></tr><tr><td>'E'</td><td>NUL</td></tr></table>=
<table><tr><td>41</td><td>42</td></tr><tr><td>43</td><td>44</td></tr><tr><td>45</td><td>00</td></tr></table>
'ABCD'
<table><tr><td>'A'</td><td>'B'</td></tr><tr><td>'C'</td><td>'D'</td></tr><tr><td>NUL</td><td>NUL</td></tr></table>=
<table><tr><td>41</td><td>42</td></tr><tr><td>43</td><td>44</td></tr><tr><td>00</td><td>00</td></tr></table></td><td>MOV$ D00100 D00200
<table><tr><td>D00100</td><td>41</td><td>42</td></tr><tr><td>D00101</td><td>43</td><td>44</td></tr><tr><td>D00102</td><td>45</td><td>00</td></tr></table>↓
<table><tr><td>D00200</td><td>41</td><td>42</td></tr><tr><td>D00201</td><td>43</td><td>44</td></tr><tr><td>D00202</td><td>45</td><td>00</td></tr></table></td></tr>
<tr><td colspan="5">可用于字符串的 ASCII 字符包括字母数字字符，日文片假名和符号、（除特殊字符外），这些字符如下表所示。</td></tr>
</table>

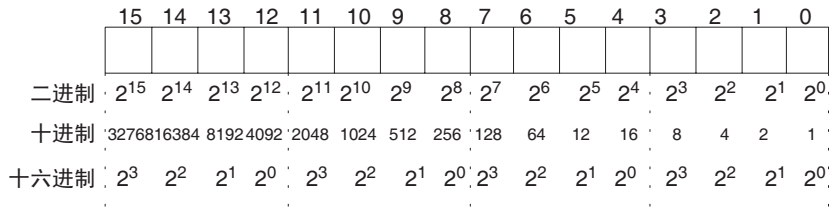
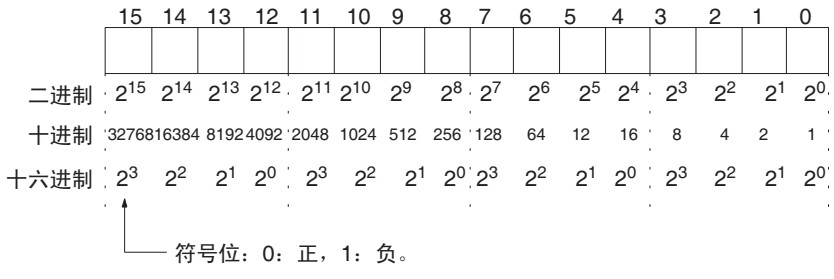
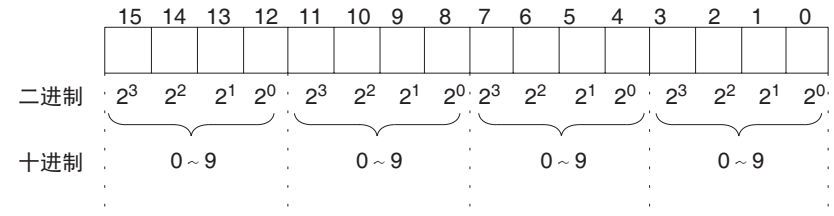
注 不带符号的十进制数标识符仅限于 CX-Programmer 使用。

ASCII 字符

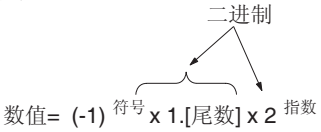
位 0 ～ 3		位 4 ～ 7															
二进制		0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
	十六进制	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0			Space	0	@	P	`	p				–	タ	ミ		
0001	1			!	1	A	Q	a	q			。	ア	チ	ム		
0010	2			”	2	B	R	b	r			「	イ	ツ	メ		
0011	3			#	3	C	S	c	s			」	ウ	テ	モ		
0100	4			$	4	D	T	d	t			\	エ	ト	ヤ		
0101	5			%	5	E	U	e	u			.	オ	ナ	ユ		
0110	6			&	6	F	V	f	v			ヲ	カ	ニ	ヨ		
0111	7			’	7	G	W	g	w			ァ	キ	ヌ	ラ		
1000	8			(	8	H	X	h	x			ィ	ク	ネ	リ		
1001	9			)	9	I	Y	i	y			ゥ	ケ	ノ	ル		
1010	A			*	:	J	Z	j	z			ェ	コ	ハ	レ		
1011	B			+	;	K	[	k	{			ォ	サ	ヒ	ロ		
1100	C			,	<	L	\	l	|			ャ	シ	フ	ワ		
1101	D			-	=	M	]	m	}			ュ	ス	ヘ	ン		
1110	E			.	>	N	^	n	~			ョ	セ	ホ	``		
1111	F			/	?	O	_	o				ッ	ソ	マ	。		

2-1-6 数据格式

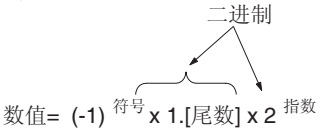
下表给出了 CS/CJ 系列 PLC 中可使用的数据格式 。

数据类型	数据格式	十进制	4 位十六进制
不带符号的二进制数据		0 ～ 65535	0000 ～ FFFF
带符号的二进制数据		–32768 ～ +32767	8000 ～ 7FFF
BCD（二一十进制数据）		0 ～ 9999	0000 ～ 9999

<table>
<tr><th>数据类型</th><th>数据格式</th><th>十进制</th><th>4 位十六进制</th></tr>
<tr><td>单精度十进制浮点数</td><td> 



符号(31位) 1: 负或 0: 正
尾数 从00位到22位的23位内容为尾数，例如：小数点后面的部分1.□□□.....,以二进制形式表示。
指数 从23位到30位的8位内容为指数。二进制形式的指数e用127+n表示（$2^n = 2^{e-127}$ ）。
注 这是符合标准IEEE754单精度浮点数格式，并且它仅用于浮点数转换和运算指令中。使用这种数据格式时，可用CX-Programmer（不能用手持式编程器）进行设定和监控I/O内存数据。为此，用户不必了解这种数据格式的细节，只要知道每个这种格式数占用2个字就可以了。</td><td>---</td><td>---</td></tr>
<tr><td>双精度十进制浮点数</td><td> 



符号（63位） 1: 负或 0: 正
尾数 从00位到51位的52位内容为尾数，例如：小数点后面的部分1.□□□.....,以二进制形式表示。
指数 从52位到62位的11位内容为指数。二进制形式的指数e用1023+n表示2（$2^n = 2^{e-1023}$ ）。
注 这是符合标准IEEE754双精度浮点数格式，并且它仅用于浮点数转换和运算指令中。使用这种数据格式时，可用CX-Programmer（不能用手持式编程器）进行设定和监控I/O内存数据。为此，用户不必了解这种数据格式的细节，只要知道每个这种格式数占用4个字就可以了。</td><td>---</td><td>---</td></tr>
</table>

带符号二进制数

在带符号二进制数据中，最高位表示 16 位二进制数的符号。数值用 4 位 16 进制表示。

正数：如果一个数的最高位是 0（OFF），这个值是正数或零。在 4 位 16 进制数中表示为 0000~7FFF Hex。

负数：如果一个数的最高位是 1（ON），这个值是负数。在 4 位 16 进制数中表示为 8000~FFFF Hex。负数（十进制）绝对值由 2 进制的补码表示。

例如：将十进制 -19 处理成带符号 2 进制，将 FFFF Hex 减 0013 Hex（19 的绝对值）再加 0001 Hex，得到 FFED Hex。

	F	F	F	F
	1111	1111	1111	1111
原码数	0	0	1	3
	0000	0000	0001	0011
–)				
	F	F	E	C
	1111	1111	1110	1100
	0	0	0	1
	0000	0000	0000	0001
+)				
二进制补码	F	F	E	D
	1111	1111	1110	1101

补码

一般以 x 为基数的补码用 X-1 减去一个给定数字的所有位，然后再在最低位加 1 求得（例如：7556 的十进制补码是：9999-7556 +1= 2444。）一个补码通常用于减法变为其它运算如加法运算等。

例如：根据 8954-7556 = 1398，8954 +（7556 的十进制补码）=8954 + 2444 =11398。如果不计最高位，我们得到减法运算的结果 1398。

2 的补码

一个二进制补码是一个以 2 为基数的补码，这里，我们可用 1（2-1=1）减所有位再加 1 求得。

例如：二进制数 1101 的二进制补码是 1111（F Hex）-1101（D hex）+1（1 Hex） =0011 （3 Hex）。下面是以 4 位十六进制 （Hex）表达。

一个 a Hex 的二进制补码 b Hex 是：FFFF Hex - a Hex + 0001 Hex = b Hex 确定 "a Hex" 的二进制补码 b Hex 的方法为：b Hex = 10000 Hex - a Hex。

例如：求 3039 Hex 的二进制补码，采用：10000 Hex - 3039 Hex =CFC7 Hex。同样地方用a Hex= 10000 Hex - b Hex方法从二进制补码b Hex求得a Hex值。

例如：由二进制补码 CFC7 Hex 求它的真值，采用 10000 Hex - CFC7 Hex =3039 Hex。

CS/CJ 系列有两条指令：NEG （160）二进制求补，和 NEGL （161）（双字二进制求补）这二条指令可以用于由一个真值求二进制补码或者由二进制补码求其真值。

带符号 BCD 码数据

带符号 BCD 码数据是一种专门用于表示 BCD 码负数的数据格式。尽管我们可在应用中可以看到这种数据格式，但它并非是严格地定义，而是随特殊的应用而定。 CS/CJ 系列支持下列指令转换这种数据格式：带符号 BCD 码转换到二进制：BINS （470），双精度带符号 BCD 码转换成二进制：BISL （472），带

符号二进制转换到 BCD 码：BCDS（471）和双精度带符号二进制转换到 BCD 码：BDSL （473）。详细内容请参阅 *CS/CJ 系列可编程序控制器编程手册*。

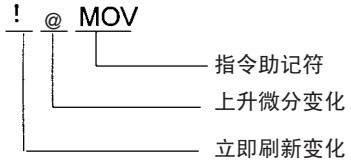
进制	十六进制	二进制	BCD	
0	0	0000		0000
1	1	0001		0001
2	2	0010		0010
3	3	0011		0011
4	4	0100		0100
5	5	0101		0101
6	6	0110		0110
7	7	0111		0111
8	8	1000		1000
9	9	1001		1001
10	A	1010	0001	0000
11	B	1011	0001	0001
12	C	1100	0001	0010
13	D	1101	0001	0011
14	E	1110	0001	0100
15	F	1111	0001	0101
16	10	10000	0001	0110

十进制	不带符号二进制 （4 位十六进制）	带符号二进制 （4 位十六进制）
+65,535	FFFF	不能表达
+65534	FFFE	
. . .	. . .	
+32,769	8001	
+32,768	8000	
+32,767	7FFF	7FFF
+32,766	7FFE	7FFE
. . .	. . .	
+2	0002	0002
+1	0001	0001
0	0000	0000
–1	不能表达	FFFF
–2		FFFE
. . .		
–32,767		8001
–32,768		8000

2-1-7 指令变化

下列变化允许指令以条件微分执行或当指令执行时立即刷新数据。

变化		符号	描述
微分	ON	@	当执行条件变为 ON 时，指令执行微分操作。
	OFF	%	当执行条件变为 OFF 时，指令执行微分操作。
立即刷新		!	当指令执行时，由操作数或特殊 I/O 单元字定义刷新 I/O 区的数据。（CS1D CPU 单元不支持立即刷新功能）。



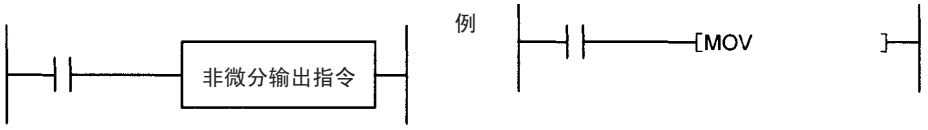
2-1-8 执行条件

CS/CJ 系列提供下列基本和特殊指令形式

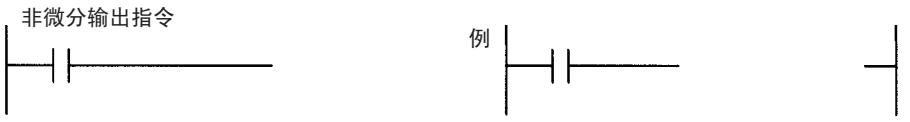
- 非微分指令每个周期执行
- 微分指令仅执行一个周期

非微分指令

当执行条件有效时 (ON 或 OFF), 需要执行条件的输出指令每个周期执行一次。



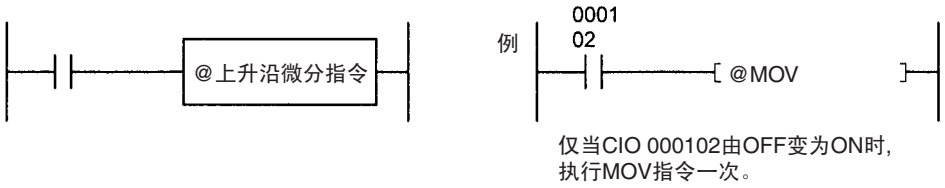
生成逻辑开始和中间指令读取位状态的输入指令在每个周期作比较，测位或完成其它的处理。如果其结果为 ON，驱动流向就输出（例如执行的条件变为 ON）。



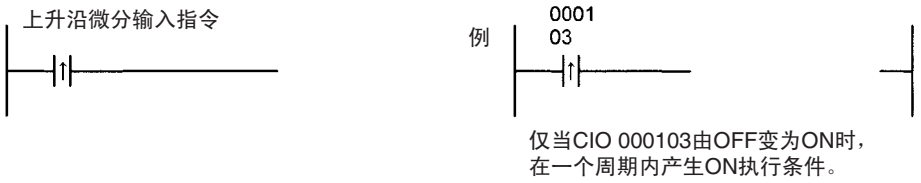
输入微分指令

上升微分指令 （指令前加 @ 符号）

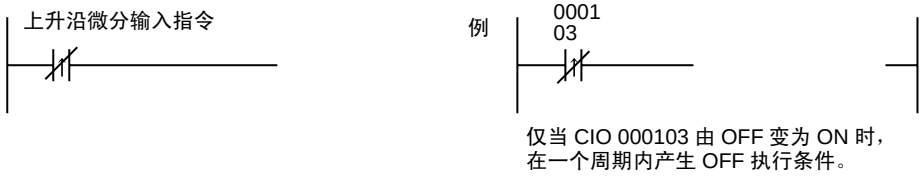
- **输出指令:** 这类指令仅当周期中执行条件变为 ON(OFF → ON) 时执行 , 为后面周期不再执行。



- **输入指令（逻辑开始和中间指令）:** 当使开关从 OFF 到 ON 时，指令在每个周期读位状态，作比较，测位或完成其它处理，并输出一个 ON 执行条件（驱动流向）。在下一个周期执行条件变为 OFF。

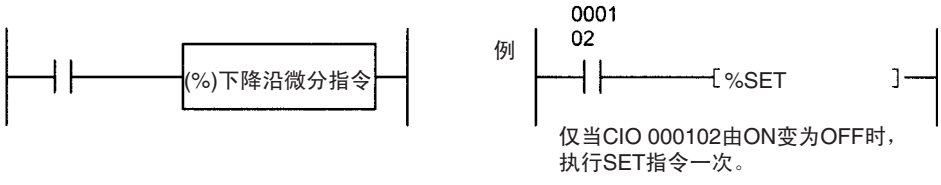


- **输入指令（逻辑开始和中间指令）:** 当使开关从 OFF 到 ON 时，指令在每个周期读位状态作比较，测位或完成其它处理，并输出一个 OFF 执行条件（功率流向停止）。在下一个周期中，执行条件变为 ON。

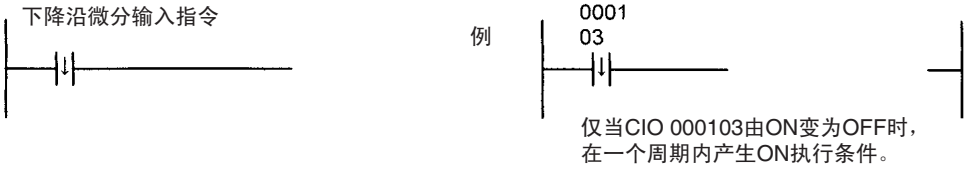


下降沿微分指令 （指令前加%符号）

- **输出指令:** 这类指令仅当周期中执行条件变为 OFF（NO → OFF）时执行，在后面周期不再执行。



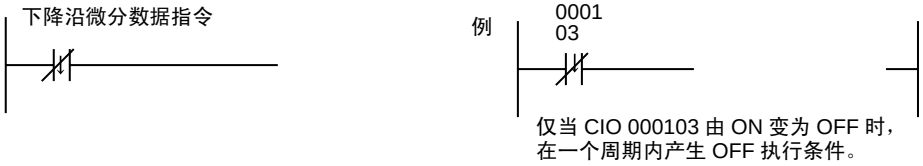
- **输入指令（逻辑开始和中间指令）**：当开关从 ON 到 OFF 时，指令在每个周期读位状态作比较，测位或完成其它处理，并输出一个 ON 执行条件（驱动流向）。在下一个周期执行条件变为 OFF。



注 a) 与上升沿微分指令不同，下降沿微分变化（%）只能用于 LD、AND、 SET 和 RSET 指令。其它指令要执行下降沿微分功能，可与 DIFD 和 DOWN 指令结合使用。

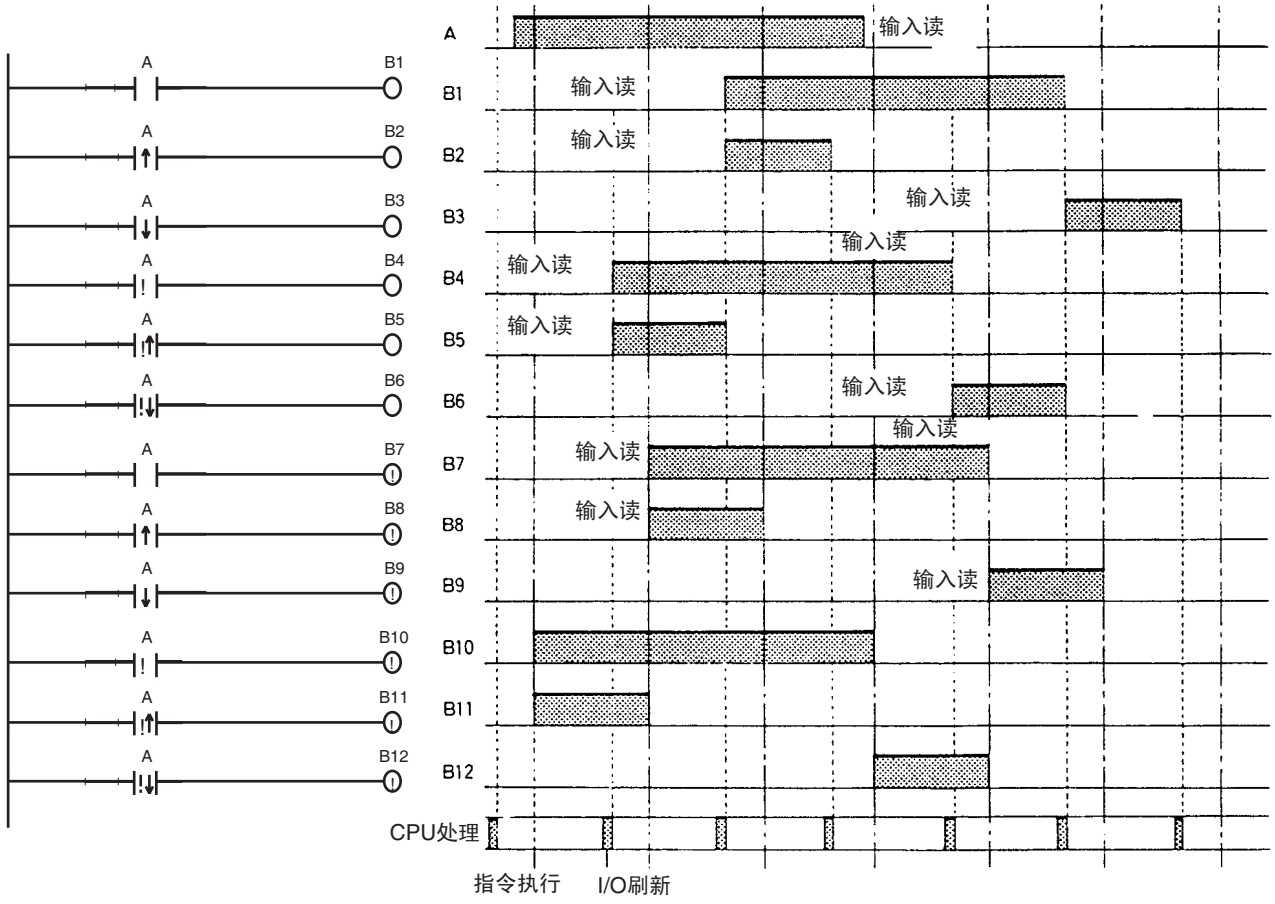
b) 上升沿和下降沿微分指令可由 DIFU 和 DIFD 指令，驱动流向上升沿 / 下降沿指令以及上升沿 / 下降沿微分 Load 指令（@LD/%LD）的结合使用来替代。

- **输入指令（逻辑开始和中间指令）**：当开关从 ON 到 OFF 时，指令在每个周期读位状态作比较，测位或完成其它处理，并输出一个 OFF 执行条件（驱动流向停止）。在下一个周期执行条件变为 ON。



2-1-9 I/O 指令时序

下列时间特性给出了只用 LD 和 OUT 指令组成的程序中各个指令的不同运行时间。

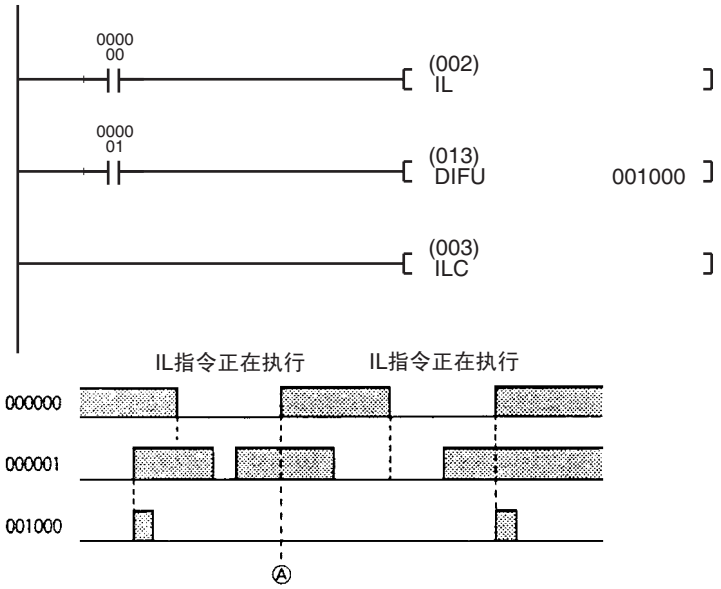


微分指令

- 微分指令具有区分以前ON还是OFF的内部标志，在运行开始，上升沿微分指令（DIFU 和 @ 前缀指令）的前状态标志设置为 ON，而下降沿微分指

令（DIFU 和 % 前綴指令）的前状态设置为 OFF。这防止微分输出在运行开始时出现意外的输出。

- 当执行条件为 ON，且前状态标志为 OFF 时，上升沿微分指令（DIFU 或 @ 前綴指令）将输出 ON。
- **使用内部锁存（IL-ILC 指令）**
  在下面例子中，用于微分指令的前状态保持以前内部锁存状态，并在 A 点由于在内部锁存有效时刻而不刷新，不输出一个微分输出。



- **使用跳转（JMP-JME 指令）**：如同联锁一样，当指令跳转，用于微分指令的前状态标志不变化，即前状态标志保持不变。仅当由前状态标志指示的输入状态变化，上升、下降沿微分指令才输出执行条件。

  注 a) 请不要使用常 ON 标志或 A20011（初始标志）作为上升沿微分指令的输入位条件，否则这条指令将永不执行。

  b) 请不要使用常 OFF 标志作为下降微分指令的输入位条件，否则这条指令将永不执行。

2-1-10 刷新时间

以下的方法用于刷新外部 I/O。

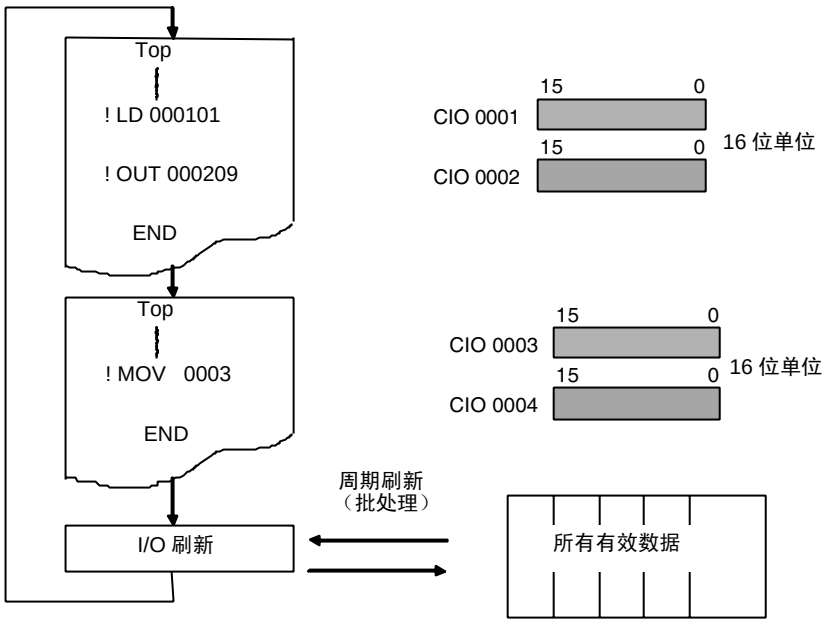
- 周期刷新
- 立即刷新 （用 ! 定义的指令和 IORF 指令）

详细内容请参阅 *CS/CJ 系列操作手册*中 CPU 单元操作章节。

周期刷新

每一分配到就绪循环任务或中断条件已满足的中断任务程序将从程序开始地址启动并运行到 END(001) 结果指令为止。在所有就绪循环程序和满足中断条件的中断程序执行完后，周期刷新在此刻刷新所有的 I/O。

注 程序可在多任务中执行，I/O 刷新是在内存中最后（在所有现有的循环任务中）的最终 END （01）指令后刷新，而在程序中其它循环任务后的 END （001）指令后不刷新。



如果在其它任务中需要刷新 I/O，在 END （001）指令前对所有需要的字执行一条 IORF 指令。

立即刷新

指令带立即刷新符号 （!） 如果一个有效 I/O 位被定义为一个操作数，下面情况 I/O 将刷新。

单元	刷新数据
C200H 基本单元 （仅适用于 CS 系列）	对包含 16 位的位， I/O 将被刷新。
CJ 基本 I/O 单元	

- 当一条指令字操作数被定义，对于所定义的这 16 位， I/O 将被刷新。
- 在一条指令被执行前，对输入或源操作数，输入将被刷新。
- 在一条指令被执行前，对输出或目的操作数，输出将被刷新。

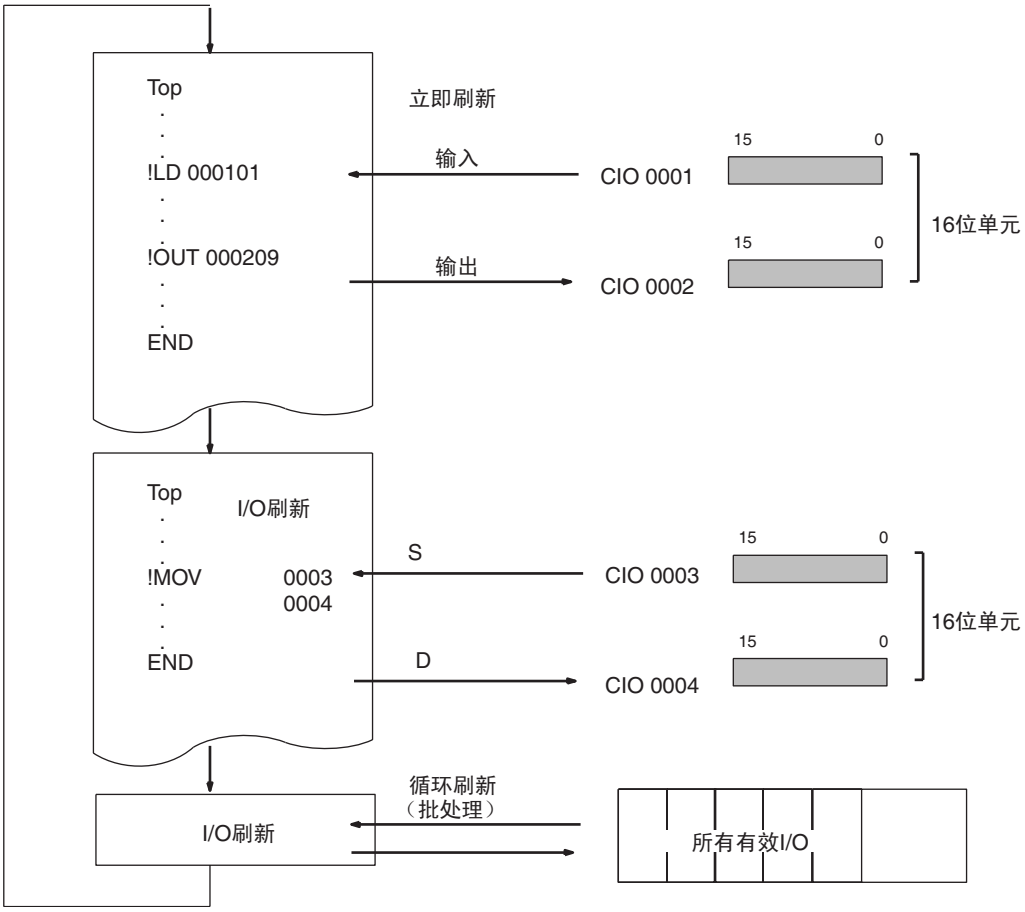
在指令前加一个感叹号 (!)（选择立即刷新）。

注　CS1D CPU 单元不支持立即刷新，但是可用 IORF（097）和 DLNK（226）指令刷新。

用 I/O 刷新指令，刷新单元。

地址	CPU 或扩展机架 （从机架上单元不允许）上单元		
单元	基本 I/O 单元	CS/CJ 系列基本 I/O 单元	被刷新
		C200H 基本 I/O 单元 （见注）	被刷新
		C200H Group-2 高密度 I/O 单元 （见注）	不被刷新
	特殊 I/O 单元		不被刷新

注　C200H I/O 单元不能装在 CJ 系列 PLC 上。



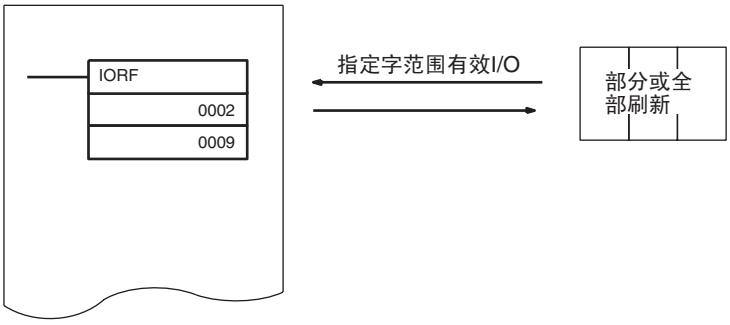
用 IORF(097) 或 DLNK(226) 刷新单元

作为一条特殊指令，I/O 刷新（IORF(097)）指令可用于在定义字范围内刷新有效 I/O 数据。用这条指令可在一个周期内刷新所有或定义范围内的有效 I/O 数据。 IORF 指令也可用于刷新位于特殊 I/O 单元中的字。

另外一条指令， CPU 总线单元刷新 （DLNK(226)）指令可用于刷新位于 CPU 总线单元 CIO 和 DM 区内的字以及完成单元的特殊刷新。例如刷新数据链接。仅 CS1-H、 CJ1-H、 CJ1M 或 CS1D CPU 单元支持 DLNK(226) 指令。

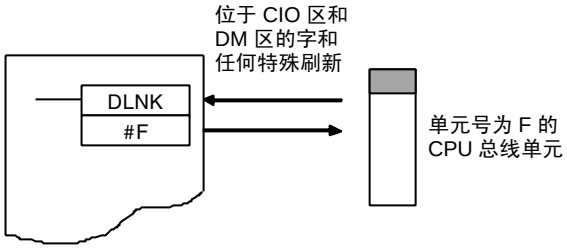
用 IORF(097) 指令刷新单元

地址	CPU 或扩展 I/O 机架 （SYSMAC 总线从机架无效）		
单元	基本 I/O 单元	CS/CJ 系列基本 I/O 单元	可刷新
		C200H 基本 I/O 单元	可刷新
		C200H Group-2 高密度 I/O 单元	可刷新
	特殊 I/O 单元		可刷新
	CPU 总线单元		不可刷新



用 DLNK(226) 指令刷新单元

地址	CPU 或扩展 I/O 机架 （SYSMAC 总线从机架无效）	
单元	基本 I/O 单元	不可刷新
	特殊 I/O 单元	不可刷新
	CPU 总线单元 在CIO区或单元中的字 在DM区域单元中的字 对 （控制）链接单元数据链接和 SYSMAC 链接单元或用于 DeviceNet 网络单元远距 I/O 单元的特殊刷新。	可刷新



2-1-11 程序容量

可存储所有用户程序的 CS/CJ 系列 CPU 单元最大程序容量由下表给出。所有容量是以最大程序步数形式表示。程序容量不得超过该值，如果想写入的程序超过这个最大程序容量，则不允许写入超出部分程序。

每条指令的步从一到七步不等。对每条指令详细步数，请参阅操作手册*第 10-5 章指令执行时间和步数*章节 （如果使用双倍长度操作数，每条指令将增加一步）。

系列	CPU 单元	最大程序容量	I/O 点数
CS 系列	CS1H-CPU67H/CPU67-E	250K steps	5,120
	CS1H-CPU66H/CPU66-E	120K steps	
	CS1H-CPU65H/CPU65-E	60K steps	
	CS1H-CPU64H/CPU64-E	30K steps	
	CS1D-CPU65H	60K steps	
	CS1H-CPU63H/CPU63-E	20K steps	
	CS1G-CPU45H/CPU45-E	60K steps	
	CS1G-CPU44H/CPU44-E	30K steps	1,280
	CS1G-CPU43H/CPU43-E	20K steps	960
	CS1G-CPU42H/CPU42-E	10K steps	
CJ 系列	CJ1H-CPU66H	120K steps	2,560
	CJ1H-CPU65H	60K steps	
	CJ1G-CPU45H/CPU45	60K steps	1280
	CJ1G-CPU44H/CPU44	30K steps	
	CJ1G-CPU43H	20K steps	960
	CJ1G-CPU42H	10K steps	
	CJ1M-CPU23/CPU13	20K steps	640
	CJ1M-CPU22/CPU12	10K steps	320

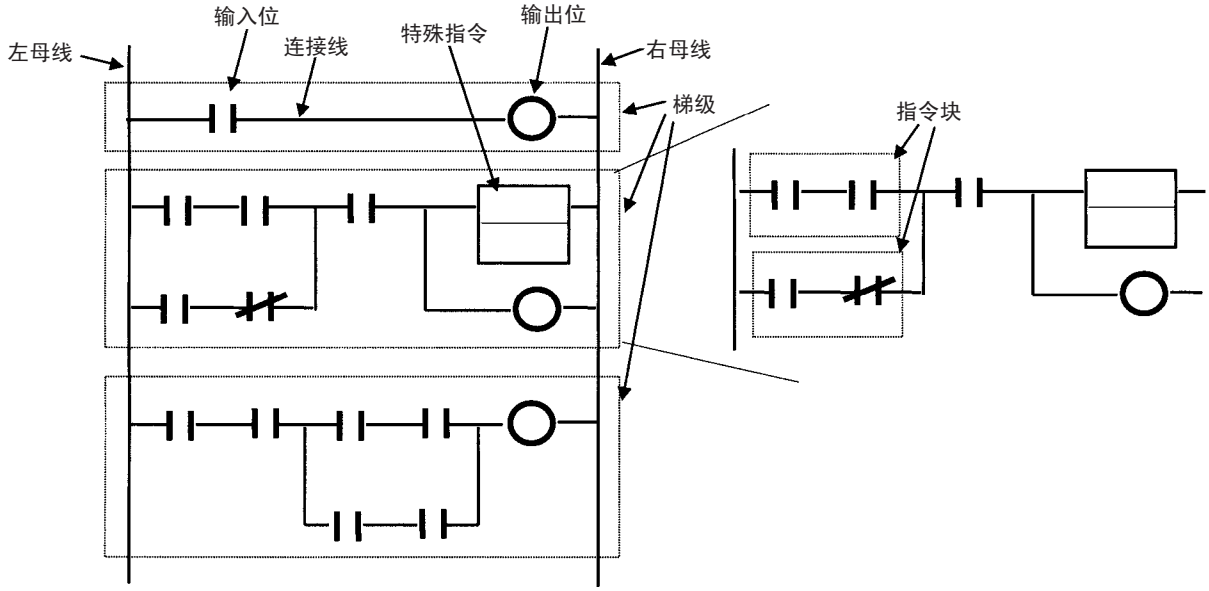
注 CS/CJ 系列 PLC 的内存容量用步数表示，尽管对以前的 OMRON PLC 的容量（如 C200HX/HG/HE 和 CV 系列 PLC）是用字来表示。从原来 OMRON PLC 产品的程序容量转换到现在产品的准则，请参阅本操作手册*第 10-5 章结尾部分的指令执行时间和指令步数*内容。

2-1-12 梯形图编程基本概念

指令的执行顺序是按在存储器中指令次序 （助记符的顺序）进行。编程的基本概念以及执行顺序必须正确。

梯形图的基本结构

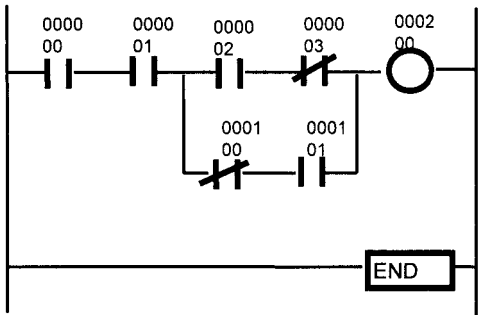
梯形图由左和右母线，连接线、输入位、输出位和特殊指令组成，一段程序由一个或多个梯级组成。当母线平行分开时，一个程序梯级是一个可分割的单元。对助记符而言，一个梯级是包含从 LD/LD NOT 指令开始到下一个 LD/LD NOT 前输出指令间的所有指令，一个程序梯级由表示逻辑加载的 LD/LD NOT 指令开始的指令块组成。



助记符

助记符程序是以助记符形式给出的一串梯形图指令。它有程序地址，一个程序地址对应一条。程序地址是从 000000 开始的六位组成。

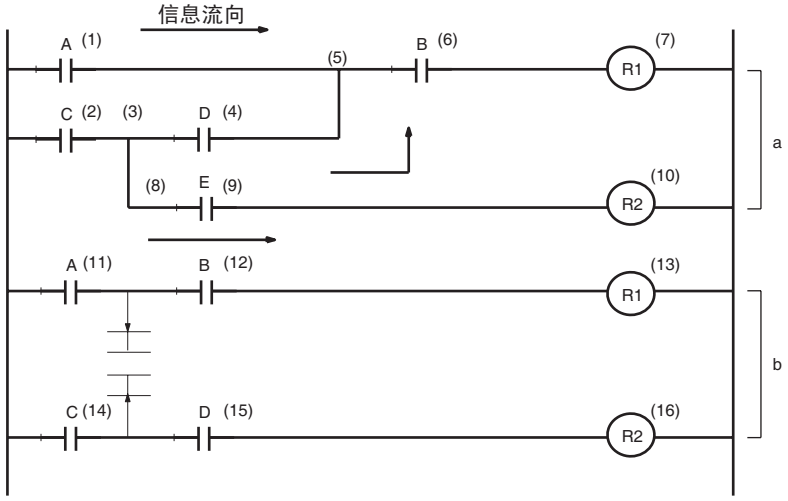
例



程序地址	指令 （助记符）	操作符
000000	LD	000000
000001	AND	000001
000002	LD	000002
000003	AND NOT	000003
000004	LD NOT	000100
000005	AND	000101
000006	OR LD	
000007	AND LD	
000008	OUT	000200
000009	END	

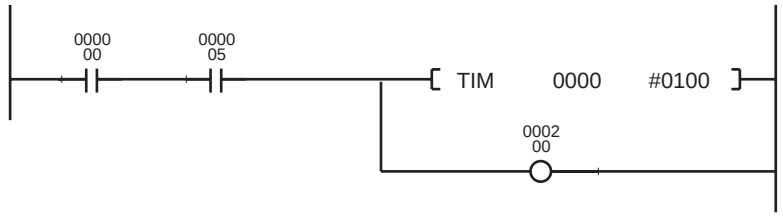
梯形图基本概念

1,2,3... 1. 程序中的驱动流向是由左向右，在梯级 "a" 和 "b" 驱动流向如同流过插入的二极管。梯级必须改变为像没有二极管普通电路的生产控制。梯型图中指令的执行是从左母线至右母线，并从上到下按序进行，其执行顺序与助记符的列序一致。

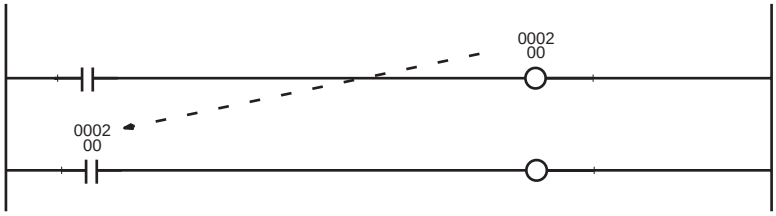


执行顺序	助记符	执行顺序	助记符
(1)	LD A	(9)	AND E
(2)	LD C	(10)	OUT R2
(3)	OUT TR0	(11)	LDA
(4)	AND D	(12)	AND B
(5)	OR LD	(13)	OUT R1
(6)	AND B	(14)	LD C
(7)	OUT R1	(15)	AND D
(8)	LD TR0	(16)	OUT R2

2. 编程中对 I/O、工作位、计时器和其它可使用的输入位的使用次数是不受限制的。尽管有时为了使程序易懂和便于维护而增加了输入位，但应尽可能地使梯级保持简洁。
3. 梯级中对串联、串并联、并联支路中连接的输入位的个数不受限制。
4. 两个以上输出位可并联连接。



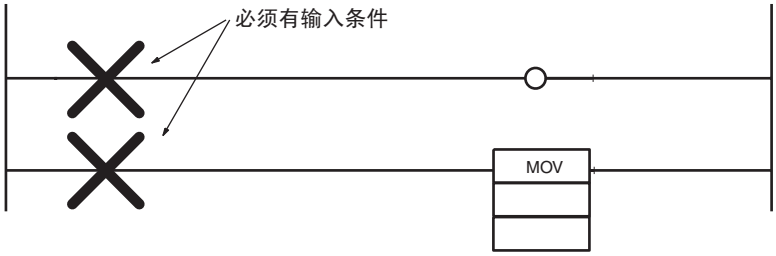
5. 输出位也可被用作编程输入位。



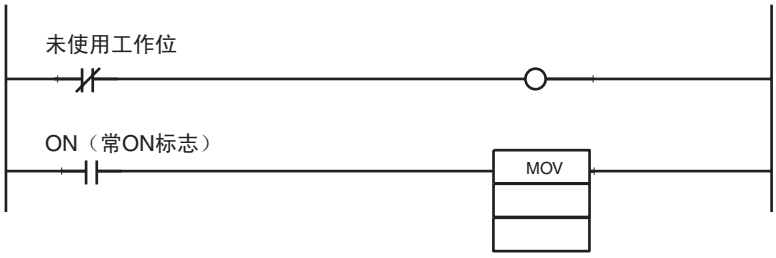
限制

1,2,3... 1. 梯形图必须封闭，这样信号 （驱动流向）就可以从左母线流向右母线。如果梯形图不封闭，一个梯级错误信息将会出现 （但程序仍可运行）。

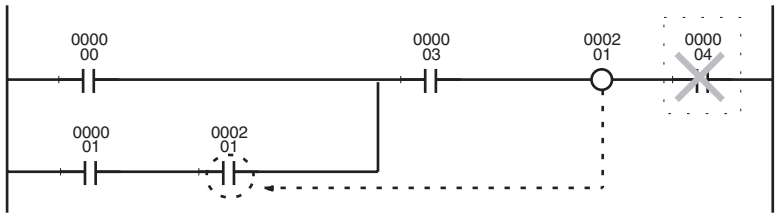
2. 输出位、计时器、计数器和其它输出指令不允许与左母线直接连接。如果有一个与左母线直接连接，编程装置在程序检查时将出现梯级出错指示（程序仍可执行，但这个 OUT 或 MOV(021) 指令不执行）。



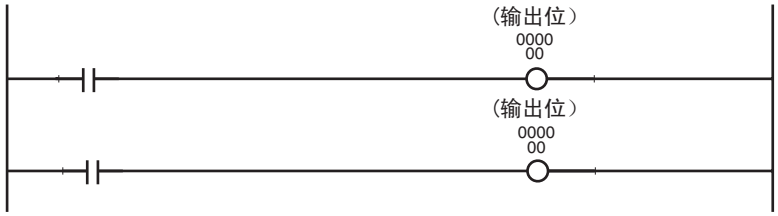
如果需要输入条件保持常 ON，可插入一个未使用输入常闭 （N.C）工作位或 ON 标志 （常 ON 标志）作为虚拟输入位。



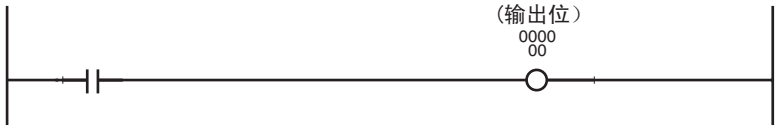
3. 输入位必须总是位于输出指令之前而不能插入到输出指令之后。如果输入指令插在输出指令之后，那么编程装置在程序检查时会给出位置出错提示。



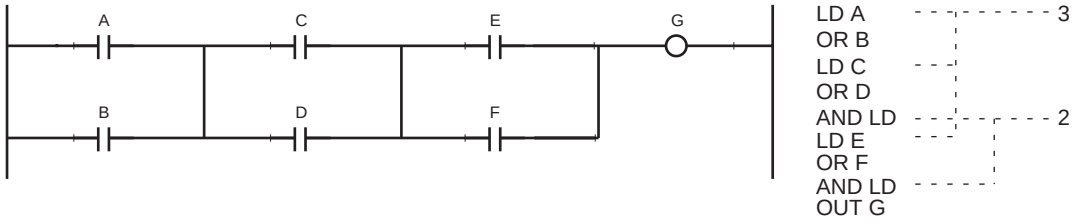
4. 同一输出位不能在输出指令编程时重复使用，否则，重复输出出错提示会出现，且第一次编程使用的那条输出指令无效，而第二梯级处的输出结果有效。



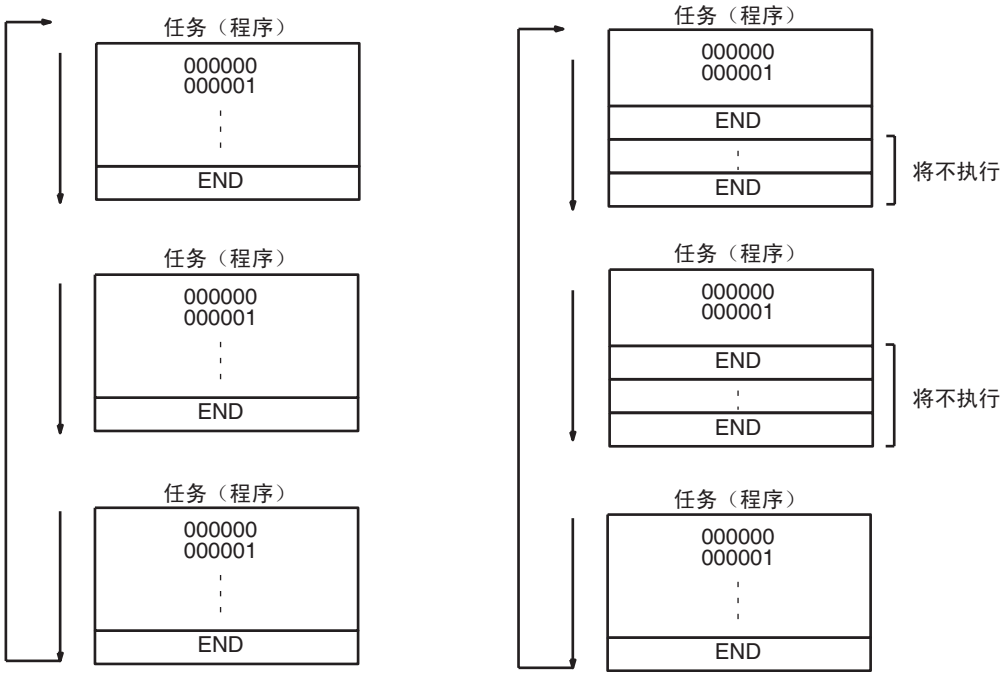
5. 输入位不能用于输出 （OUT）指令。



6. 表示逻辑启动的 LD/LD NOT 指令数减一必须和指令块连接指令 AND LD 和 OR LD 数相等，否则在编程装置程序检查时将出现梯级错误指示。



7. 每个任务程序结束处必须插入一条 END （001）指令。
   - 如果开始运行时程序没有 END（001）指令，程序无 END 指令错误将会指示。在 CPU 单元前面的 ERR/LAM LED 会点亮，程序将不能执行。
   - 如果程序中有多个 END （001）指令，那么第一个 END （001）指令将作为程序结束指令而不运行其它部分程序。
   - 如果在梯级层间的断点处输入 END （001）指令，并在程序检查后删除插在这些梯级中的 END （001）指令，这可使程序调试工作变得极为顺利。

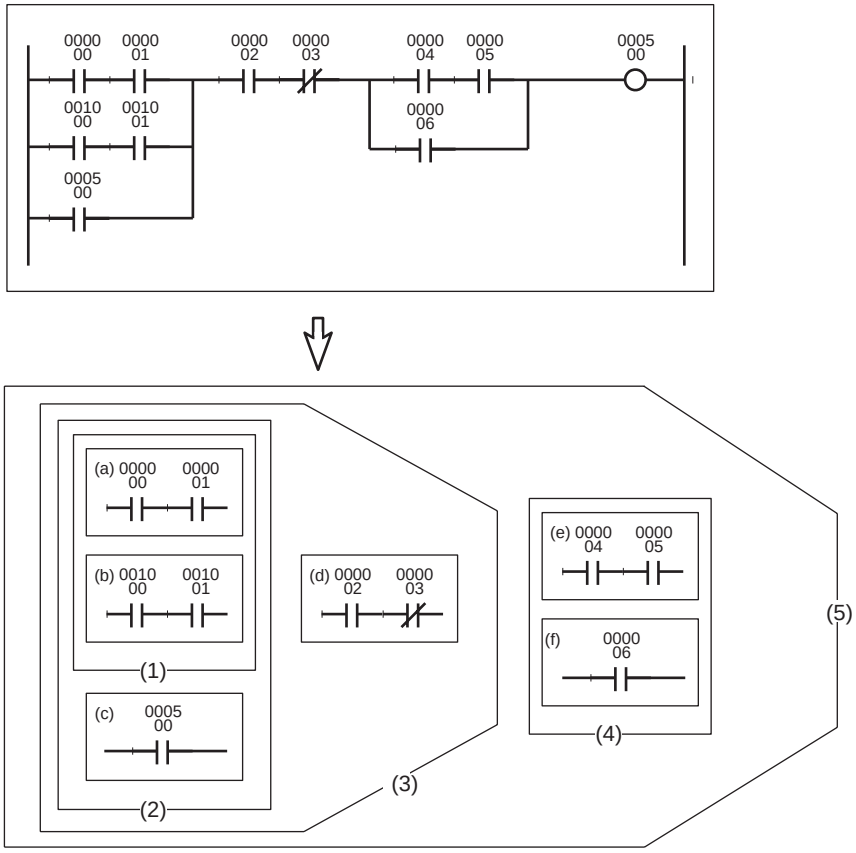


2-1-13 输入助记符

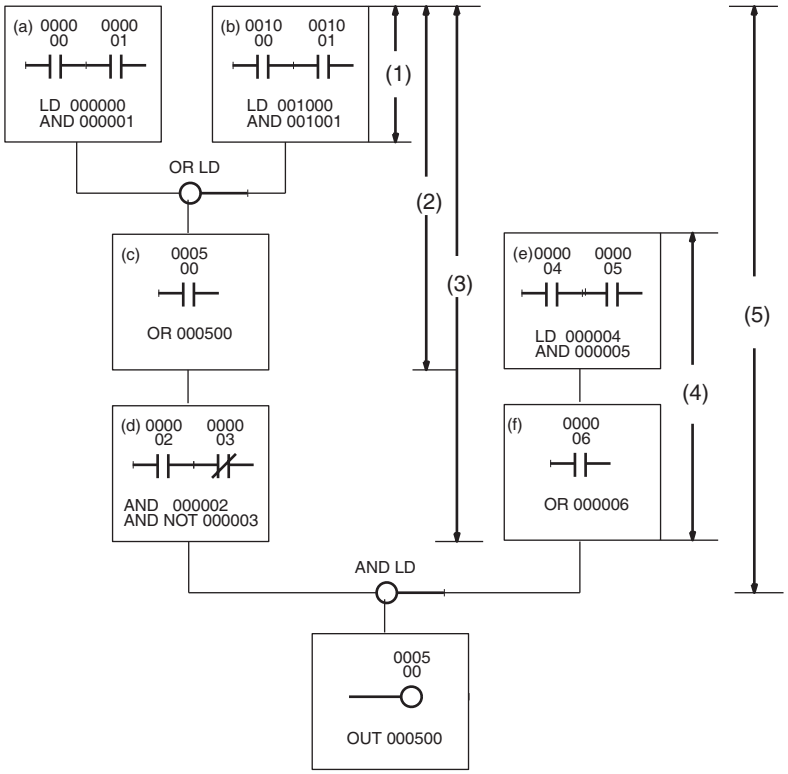
一个逻辑的开始由 LD/LD NOT 指令完成，我们可以把从逻辑开始直到下一个 LD/LD NOT 指令前一条指令可看作一个独立的指令块。

生成一个由两个指令块组成的独立梯级，可采用 AND LD 指令将指令块相“与”，或采用 OR LD 指令将指令块相 “或”。下面的例子表明了一个复杂梯级输入助记符的过程 （梯级概要和顺序）。

1,2,3... 1. 首先将梯级分解成小的指令块 (a)~(f)。



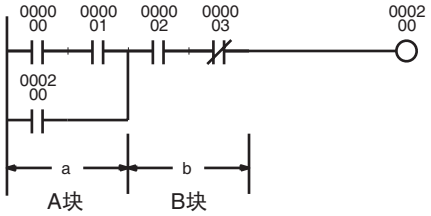
• 自上而下编制指令块，然后再从左到右编程。



	地址	指令	操作数
(a)	000200	LD	000000
	000201	AND	000001
(b)	000202	LD	001000
	000203	AND	001001
	000204	OR LD	---
(c)	000205	OR	000500
(d)	000206	AND	000002
	000207	AND NOT	000003
(e)	000208	LD	000004
	000209	AND	000005
(f)	000210	OR	000006
	000211	AND LD	---
	000212	OUT	000500

2-1-14 编程举例

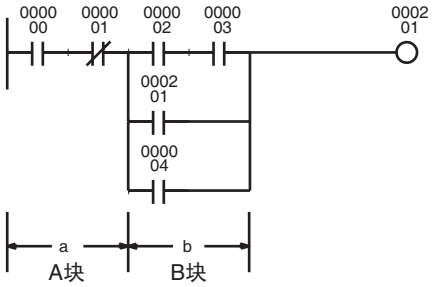
1,2,3... 1. 并 / 串梯级



指令	操作数	
LD	000000	a
AND	000001	
OR	000200	
AND	000002	b
AND NOT	000003	
OUT	000200	

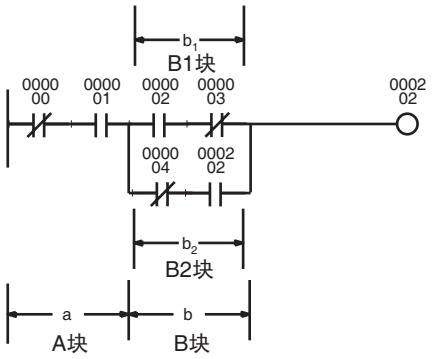
在A块中编并联指令，然后再编B块程序。

2. 并 / 串梯级



指令	操作数	
LD	000000	a
AND NOT	000001	
LD	000002	b
AND	000003	
OR	000201	
OR	000004	
AND LD	---	
OUT	000201	

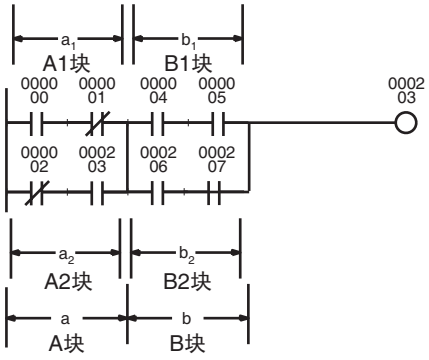
- 现将梯级分为A和B块，然后分别编程。
- 用AND LD指令连接A和B块。
- 编A块程序。



指令	操作数	
LD NOT	000000	a
AND	000001	
LD	000002	b_1
AND NOT	000003	
LD NOT	000004	b_2
AND	000202	
OR LD	---	$b_1 + b_2$
AND LD	---	a b
OUT	000202	

- 编B_1块程序,再编B_2块程序。
- 用OR LD指令连接B_1和B_2块程序，再用AND LD指令连接A和B块程序。

3. 串联梯级中的串联连接举例



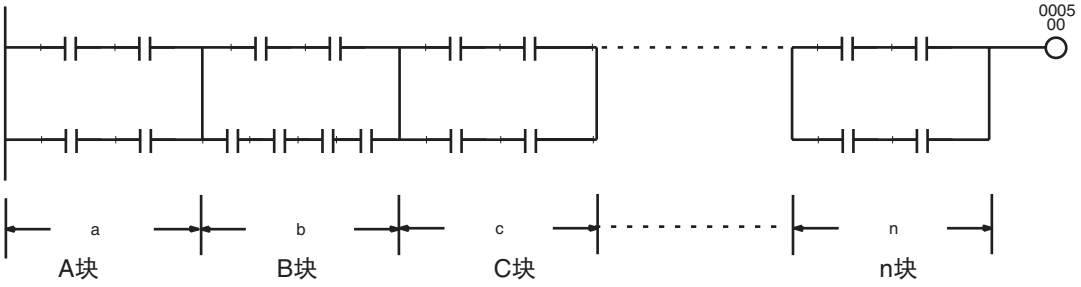
指令	操作数	
LD	000000	a_1
AND NOT	000001	
LD NOT	000002	a_2
AND	000003	
OR LD	---	$a_1 + a_2$
LD	000004	b_1
AND	000005	
LD	000006	b_2
AND	000007	
OR LD	---	$b_1 + b_2$
AND LD	---	a b
OUT	000203	

分别编A1和A2块程序，然后用OR LD指令连接A1和A2块。

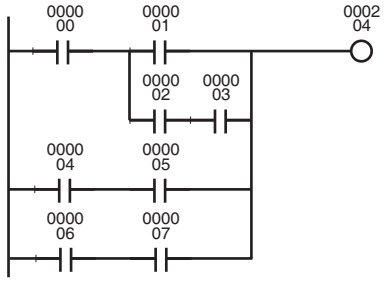
用相同方法编B1和B2块程序。

用AND LD指令连接A和B块程序。

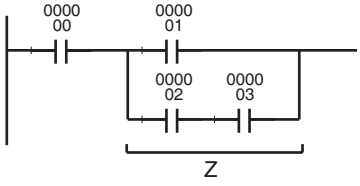
如下所示，重复A与B块程序连接编程



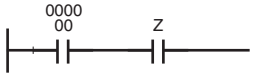
4. 复杂梯级



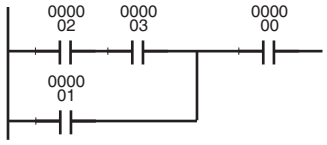
指令	操作数	
LD	000000	
LD	000001	Z
LD	000002	Z
AND	000003	Z
OR LD	---	Z
AND LD	---	
LD	000004	
AND	000005	
OR LD	---	
LD	000006	
AND	000007	
OR LD	---	
OUT	000204	

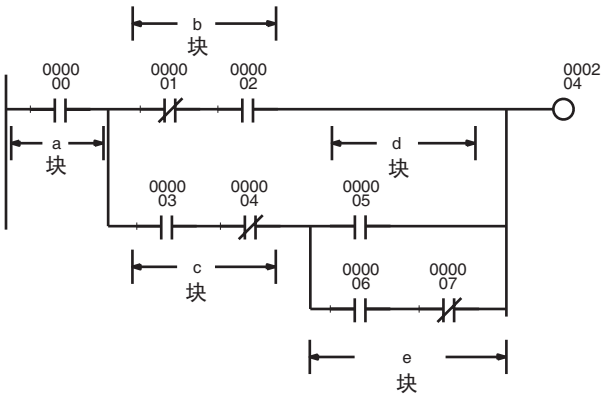


上面梯形图是基于下图



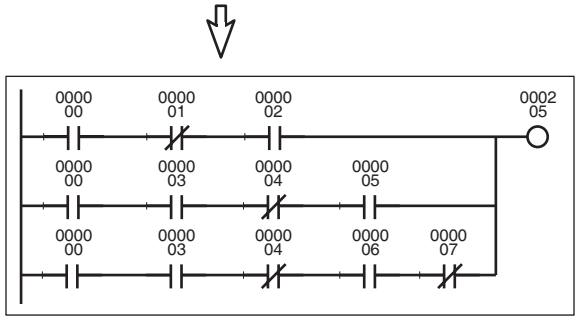
重画的简化梯形图如下所示

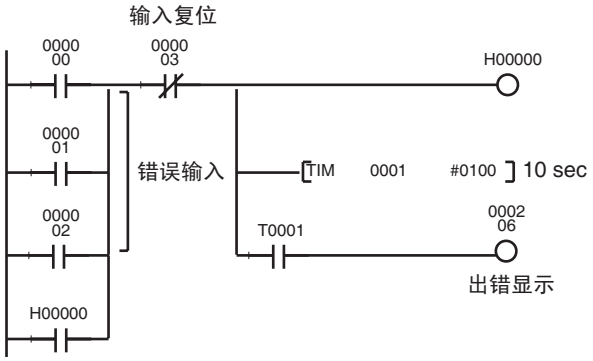




指令	操作数	
LD	000000	a
LD NOT	000001	b
AND	000002	b
LD	000003	c
AND NOT	000004	c
LD	000005	d
LD	000006	d
AND NOT	000007	e
OR LD	---	e
AND LD	---	d + e
OR LD	---	(d + e) c
AND LD	---	(d + e) c + b
OUT	000205	((d + e) c + b) a

上面梯级重画如下所示



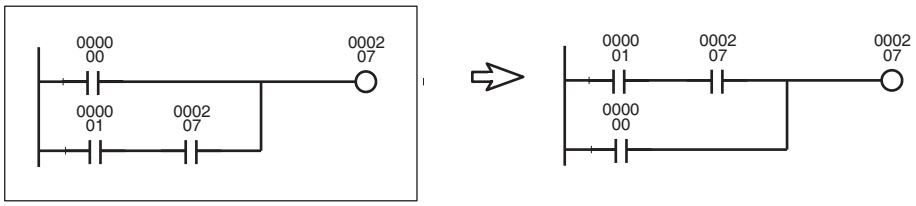


指令	操作数
LD	000000
OR	000001
OR	000002
OR	H00000
AND NOT	000003
OUT	H00000
TIM	0001
	0100
AND	T0001
OUT	000206

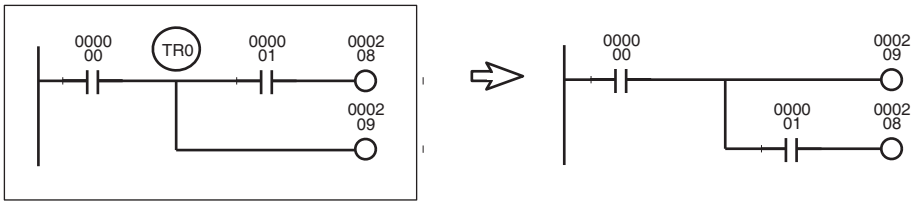
如果使用保持位，即使电源断开，内存中的ON/OFF状态字保持原状，当电源重新恢复接通，出错信号将仍然有效。

5. 梯级注意事项和重画

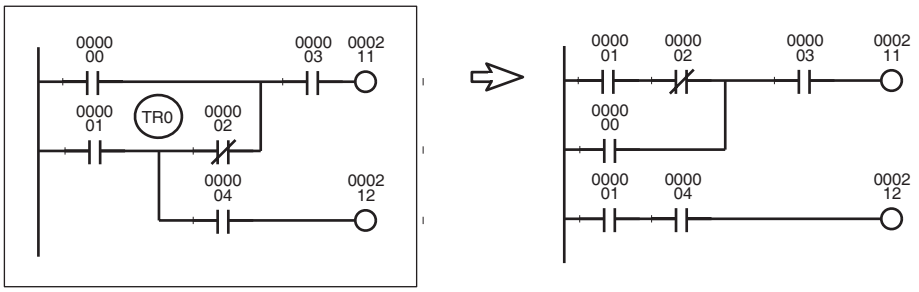
OR指令
对当前执行条件采用"或"逻辑，应采用一条OR/OR NOT指令，即梯级逻辑的结果取决于OR、OR NOT指令。
在左面例子中，如果所编程序的梯级未作改进，则需要一条OR LD指令。通过重复画梯级如下所示，可省去些程序步。



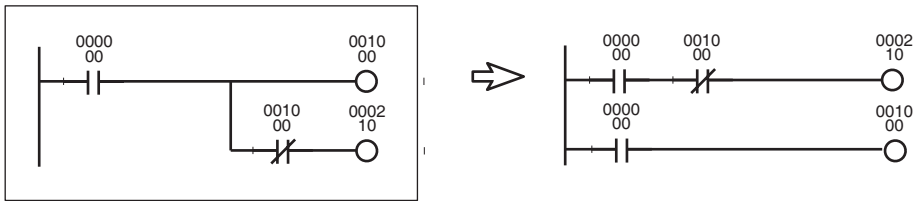
输出指令分支
如果AND/AND NOT指令前有分支，一个TR位将需要使用。如果分支点与第一输出指令直接相连，TR位就不必使用。在第一条输出指令后，无需改进，用AND/AND NOT指令和第二条输出指令连接。
在左面例子中，如果梯级不改进，在分支点需要一个暂存位TRO输出指令和LD指令。通过重画梯级,可省去一些程序步。有关TR位详细内容参阅下面说明。



在下例子中，用TRO在分支点存储执行条件或重画梯级以改进。

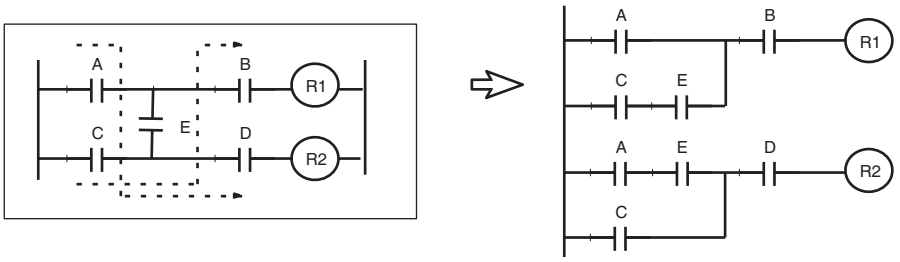


助记符执行顺序
由于PLC以助记符顺序执行，下面给出的CIO 000210将永远不会变为ON。重画梯级改进后，CIO 000210可在一个周期中变为ON。



左面重画的梯级不能执行

当梯级由控制继电器组成时，箭头表明了信号（驱动流向）流向。



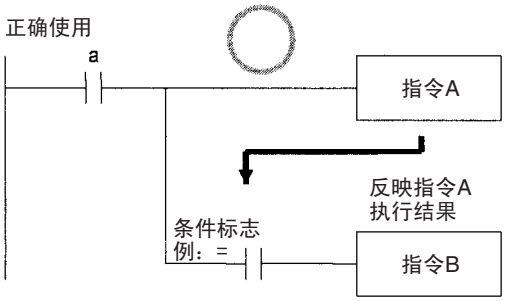
2-2 注意事项

2-2-1 条件标志

条件指令的使用

条件标志由所有指令共享。因为它们的状态取决于各指令的执行结果，而在一个周期中经常变化。所以在反映指令执行结果后立即在同一执行条件分支处使用条件标志，绝对不要将条件标志与母线直接相连，因为这样做会导致反映其它指令的执行结果。

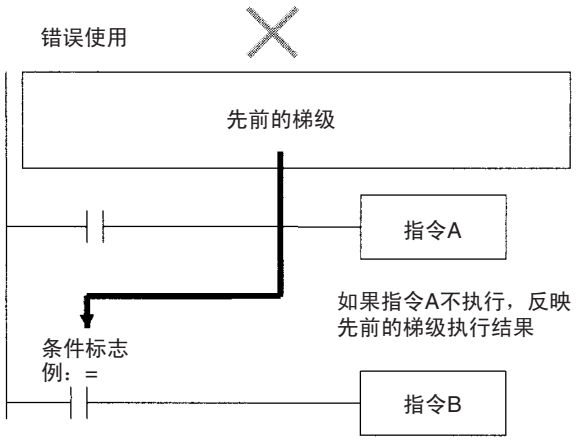
例：使用指令 A 执行结果



助记符

指令	操作数
LD	a
指令	A
AND	=
指令	B

相同的执行条件 (a) 用于指令 A 和 B，指令 B 执行条件还取决于指令 A 的执行结果。本例中，仅当指令A执行后，指令B执行取决于指令A执行后的条件标志。

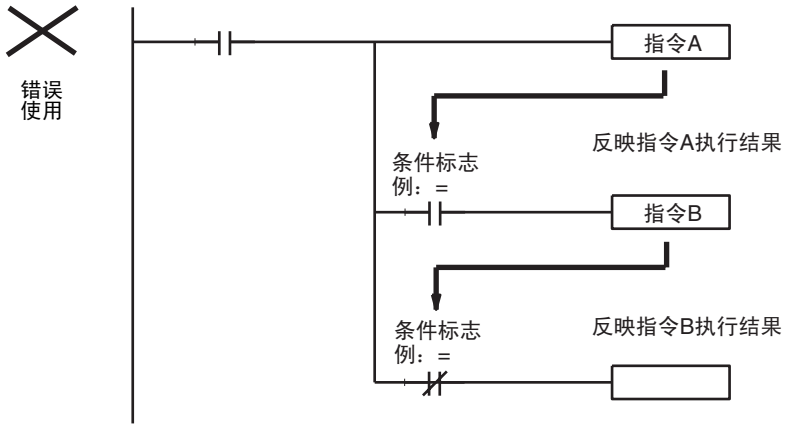


如果条件标志直接与左母线相连，当指令 A 不执行时，指令 B 的执行将取决于前面梯级的执行结果。

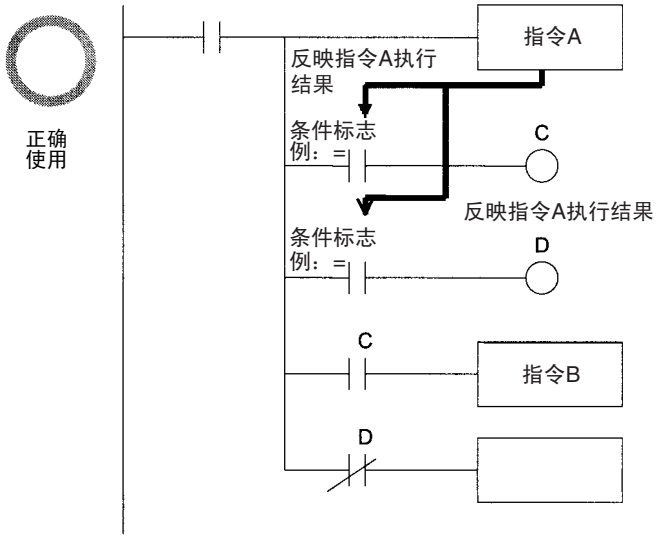
注 条件标志可用于单个程序 （任务）中的所有指令，但当任务切换时，条件标志状态 会被清除。所以，前面任务的执行结果将不会反映到后面的任务中。因为条件标志由所有指令共享，请确认在一个单独的梯形图中它们不相互冲突。

使用执行结果的常开 （N.O.）和常闭 （N.C.）输入

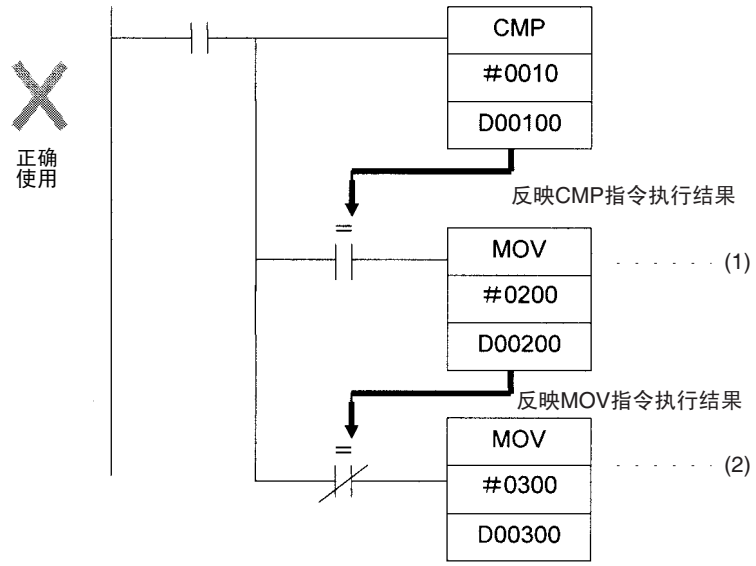
如下所示，尽管输入位的常开和常闭从相同输出分支引出执行，但条件标志将取决于指令 B 的执行结果。



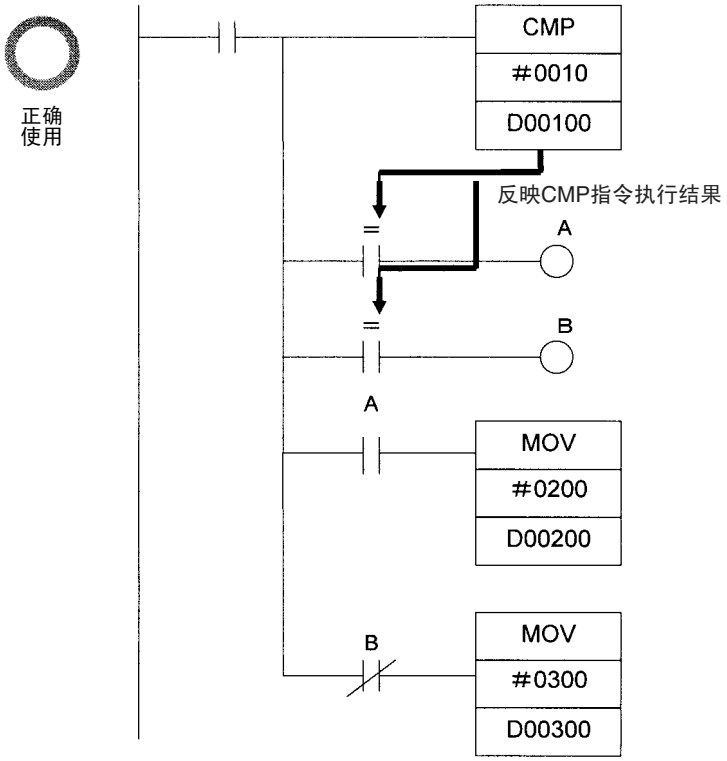
确认每一个结果从输出指令一次获取，保证指令B的指令结果不影响条件标志。



例：下列中如果 D00100 的内容为＃ 0010，则将＃ 0200 传送到 D00200；如果 D00100 内容不为＃ 0010，则将 #0300 传送到 D00300。



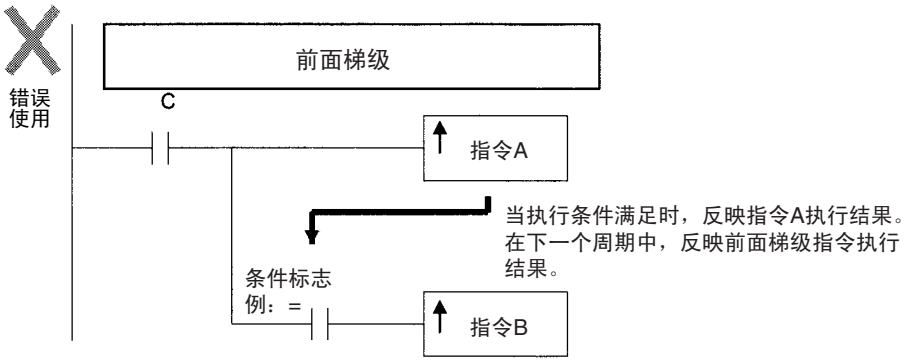
对指令 (1)，如果在上面梯级中 D00100 的内容为＃ 00100，相等标志将变为 ON，＃ 0200 将传送到 D00200 中去。但由于源数据＃ 0200 不是 0000Hex, 则使得等于标志变为 OFF。在 (2) 中的 MOV 指令将执行把＃ 0300 传送到 D0300 中去的指令。所以，如下所示，应插入一梯级以防止第一条 MOV 指令的执行结果刷新条件标志



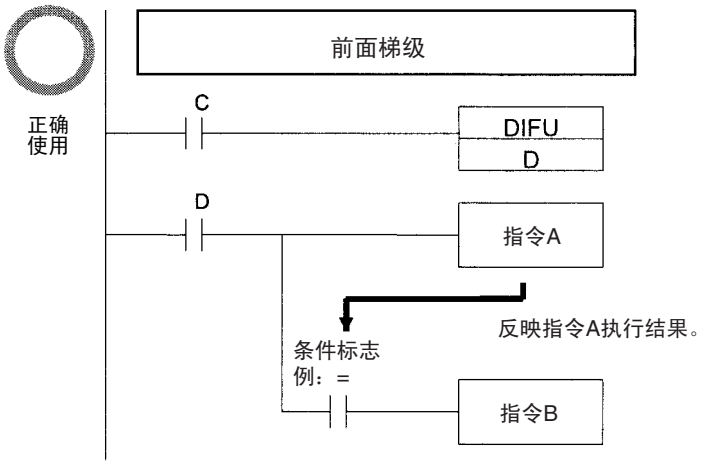
使用微分指令执行结果

对于微分指令，仅当执行条件满足，指令的执行结果才会反映在条件标志上，前面梯级指令执行结果（而不是这微分指令的执行结果）将在下个周期中反映在条件标志上。所以，你必须知道，如果使用了微分指令执行结果后，下一个周期条件标志会如何变化。

在下列中，仅当执行条件 C 满足，指令 A 和 B 才会执行，但下面的问题将会出现：指令 B 执行时，会取指令 A 的执行结果条件标志状态。如果在指令 A 执行后，执行条件 C 在下一个周期中仍保持 ON 状态，并且由于前面梯级反映的执行结果由 OFF 变为 ON 时，那么指令 B 就会发生不应有的再次执行情况。



然而在这种情况下，如下所示指令 A 和 B 不是微分指令，用 DIFD 指令来取代。指令 A 和 B 两者都成为上升沿（或下降沿）微分指令并只执行一个周期。



注 CS-H、CJ1-H 或 CJM CPU 单元支持 CCS(282) 和 CCL(283) 保存和获取条件标志状态指令，这两条指令可在一个任务或另一任务中的其它处访问条件标志的状态。

主要条件变 ON 的条件标志

出错标志

在一些特殊条件下， ER 标志将变为 ON，例如用于指令的操作数出错。当 ER 标志变为 ON，指令将不执行。

当 ER 标志为 ON，其它一些条件标志状态如 <、>、OF、UF 等不再发生变化，而 = 和 N 标志状态将随指令而变化。

参阅 CS/CJ 系列可编程序控制器编程手册 （W340）对各指令引起 ER 标志变为 ON 的条件说明。要小心，有些指令会不管条件而将 ER 标志复位 （变为 OFF）。

注 PLC 设置可设定当指令出错 ER 为 ON 时，运行是否要停止。在默认设定情况下，当 ER 标志变为 ON，运行将继续。如果定义停止运行，当 ER 标志变为 ON，运行停止 （当程序出错对待），运行停止点的程序地址将存入 A298 到 A299 内。同时， A29508 变为 ON。

相等标志

除了当比较结果相等（=），相等标志对所有指令都是暂时的。它由系统自动设定并会改变。在前一条指令使相等标志变为 ON （OFF）后，相等标志可由下一条指令变为 OFF （ON）。例如，当 MOV 或其它传送指令将 0000Hex 作为源数据传送，相等标志就会变 ON，而其它所有时候都为 OFF。即使一条指令将相等标志变为 ON，传送指令一执行，相等标志马上会根据传送的源数据是否为 0000Hex 而使状态变为 ON 或 OFF。

进位标志

进位标志用于移位指令、带进位的加法和减法指令、加法和减法指令的进位和借位以及特殊 I/O 单元指令、 PID 指令和 FPD 指令。请注意下面的注意事项。

注
1. 对某些指令因为指令的执行结果， CY 标志可保持 ON(OFF) 状态，然后这标志可用于其它指令 （如带进位的加法、减法指令和移位指令） 。请在需要的时候确认进位标志位的清零。
2. 进位标志可由某些指令的执行结果置 ON(OFF)，也可由另一些指令的执行结果置 OFF(ON)，请在使用进位标志时确认其反映的结果是需要的。

小于和大于标志

< 和 > 标志是用于比较指令以及 LMT、 BAND、 ZONE、 PID 和其它指令中。即使小于和大于标志由某些指令执行的结果置 ON(OFF)，但它也能由其它指令置 OFF(ON)。

负标志

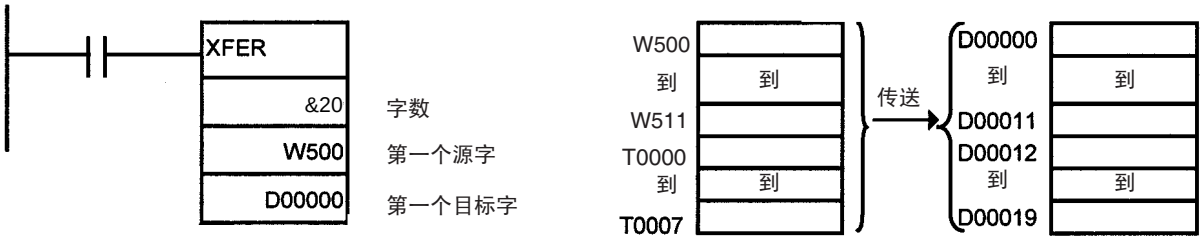
某些指令，当指令执行结果的最高位 "1" 时， N 标志变为 ON，而对其它指令，N 标志变为 OFF 是无条件的。

多字的特殊操作数

对于 CS/CJ 系列 PLC，即使一个操作数定义为多字的，指令将按所定义的执行，以至于操作数的所有字不在同一个数据区内。这种情况下，字将按 PLC 内存地址依次存放。出错标志不会变为 ON。

如例子所示，如果 20 个字传送到 W500 开始内存，工作区用 XFER（070）执行一个块传送功能。这里工作区的结束地址为 W511，传送将会超出范围，然而出错标志不会为 ON，指令照样执行。在 PLC 内存中，计时器当前值存放在工作区后的内存里，这样对下面的指令传送到 W500~W511 的内容将会转到 D00000 到 D00011， T0000 到 T0007 的当前值传送到 D00012 到 D00019。

注　对 PLC 将特殊内存地址，参阅*附录 D PLC 内存地址*分布。



2-2-2　特殊程序段

CS/CJ 系列程序具有控制指令条件的特殊程序段。下面的特殊程序段是可用的。

程序段	指令	指令条件	状态
子程序	SBS, SBN 和 RET 指令	子程序执行	执行在 SBN 和 RET 间程序段的指令
IL - ILC 段	IL 和 ILC 指令	程序段内部锁存	输出位变为 OFF，计时器复位。其它指令不执行，且以前状态保持。
步进梯级段	STEP S 指令和 STEP 指令		
FOR-NEXT 循环	FOR 指令和 NEXT 指令	进程断开	循环
JMP0 - JME0 段	JMP0 指令和 JME0 指令		跳转
块程序段	BPRG 指令和 BEND 指令	块程序执行	执行在 BPRG 和 BEND 指令间所列助记符块程序

指令组合

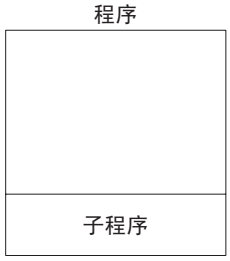
下表中给出那些特殊指令可用在其它程序段中。

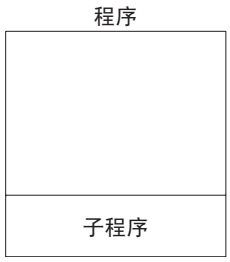
	子程序	IL-ILC 段	步进梯级段	FOR - NEXT 循环	JMP0 - JME0 段	块程序段
子程序	不可以	不可以	不可以	不可以	不可以	不可以
IL-ILC	可以	不可以	不可以	可以	可以	不可以
步进梯级段	不可以	可以	不可以	不可以	可以	不可以
FOR - NEXT 循环	可以	可以	不可以	可以	可以	不可以
JMP0-JME0	可以	可以	不可以	不可以	不可以	不可以
块程序段	可以	可以	可以	不可以	可以	不可以

注 规定程序区域的指令不能用于其它任务的程序中。详情参考*第 4-2-2 章任务指令限制*。

子程序

把所有的子程序都放在所有程序的结束 END(001) 指令前，在程序的后面，只能放子程序（而且，任一子程序不能放在步进指令、块程序、FOR ～ NEXT 或 JMP0 ～ JME0 段中）。如果将不是子程序的程序放在子程序 （SBN ～ RET）之后，这段程序将不被执行。





子程序中不允许使用的指令

以下指令不允许用于子程序中

功能	助记符	指令
步进控制处理	STEP(008)	定义步进梯形图
	SNXT(009)	通过步进梯形图

注 **块程序段**

子程序可包含块程序段。当执行从子程序返回主程序时，如果块程序处于等待状态，在下次调用时，块程序段仍保持等待状态。

步进梯形图程序段不能使用指令

功能	助记符	指令
顺序控制	FOR(512), NEXT(513) 和 BREAK(514)	FOR,NEXT 和中断循环
	END(001)	END 结束
	IL(002 和 ILC(003)	内部锁存和内部锁存清除
	JMP(004) 和 JME(005)	跳转和跳转结束
	CJP(510) 和 CJPN(511)	条件跳转和条件跳转非
	JMP0(515) 和 JME0(516)	多路跳转和多路跳转结束
子程序	SBN(092) 和 RET(093)	子程序进入和子程序返回
块程序	IF(802) (NOT), ELSE(803), 和 IEND(804)	分支指令
	BPRG(096) 和 BEND(801)	块程序开始 / 结束
	EXIT(806) (NOT)	块条件退出 （非）
	LOOP(809 和 LEND(810) (NOT)	循环控制
	WAIT(805) (NOT)	一个周期等待 （非）
	TIMW(813)	定时器等待
	TMHW(815)	高速定时器等待
	CNTW(814)	计数器等待
	BPPS(811) 和 BPRS(812)	块程序暂停和重新启动

注 1. 步进梯形图程序可用在内部锁存段（在 IL 和 ILC 之间）内。当内部锁存为 ON 时，步进梯形图将被完全复位。

2. 步进梯形图程序可用在多路跳转 （JMP0）和多路跳转结束 （JME0）间。

在块程序段中不允许使用的指令

下列指令不能用于块程序段中。

按功能分类	助记符	指令
顺序控制	FOR(512), NEXT(513) 和 BREAK(514)	FOR, NEXT 和中断循环
	END(001)	END 结束
	IL(002) 和 ILC(003)	内部锁存和内部锁存清除
	JMP0(515) 和 JME0(516)	多路跳转和多路跳转结束
顺序输入	UP(521)	条件 ON
	DOWN(522)	条件 OFF
顺序输出	DIFU	上升沿微分
	DIFD	下降沿微分
	KEEP	保持
	OUT	输出
	OUT NOT	输出非
定时 / 计数器	TIM	定时器
	TIMH	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)	多路输出定时器
	CNT	计数器
	CNTR	可逆计数器
子程序	SBN(092) 和 RET(093)	子程序进入和返回
数据移位	SFT	移位
步进控制	STEP(008) 和 SNXT(009)	步定义和步开始
数据控制	PID	PID 控制
块程序	BPRG(096)	块程序开始
故障诊断	FPD(269)	故障点检测

注 1. 块程序可用于步进程序段中。

2. 一个块程序可用于内部锁存段 （IL 和 ILC 间）中。当内部锁存为 ON 时，块程序将不执行。

3. 一个块程序可用于多路跳转 （JMP0）和多路跳转结束 （JME0）指令间。

4. 跳转指令 （JMP）和条件跳转指令 （CJP/CJPN）可用于块程序段中。但跳转 （JMP）和跳转结束 （END）指令及条件跳转 （CJP/CJPN）和条件跳转结束指令 (JME) 在块程序中必须成对使用，不允许单独使用。

2-3 检查程序

CS/CJ 系列程序可在下列时刻检查。

- 在手持式编程器输入操作时作输入检查
- 用 CX-Programmer 检查
- 在程序执行时作指令检查
- 在程序执行时作致命错误检查 （程序出错）

2-3-1 编程工具输入时出错

手持式编程器

在输入操作时，下列错误将在手持式编程器上显示。

错误显示	原因
CHK MEM	CPU 单元上 DIP 开关脚 1 置 ON （写保护）
IO No. ERR	一个非法的 I/O 输入

CX-Programmer

在下列时刻， CX-Programmer 会自动检查程序

时刻	检查内容
输入梯形图时	指令输入，操作数输入，编程格式。
存储文件时	所有操作数，指令和编程格式。
下载文件时	CS/CJ 系列 PLC 支持的模型和所有指令的操作数
在线编辑时	容量等

检查的结果会在输出窗口表格中给出。同样以梯形图显示的程序会对非法程序部分的左母线用红色显示。

2-3-2 用 CX-Programmer 检查程序

由 CX-Programmer 检查程序所发现的错误以下列表格形式列出。

CX-Programmer 不对指令中间接寻址的操作数作范围出错检查，间接寻址错误在程序执行检查时检测，如果出错 ER 标志置 ON，相关内容在下章描述。详情参阅 *CS/CJ 系列可编程序控制器编程手册 (W340)*。

当用 CX-Programmer 检查程序，操作可定义程序检查等级 A、 B 或 C （出错严重程度），以及用户检查级。

区域	检查
非法数据：梯形图编程	指令位置
	I/O 连线
	连接
	指令和操作完整性
PLC 支持的指令	PLC 支持的指令和操作数
	指令变化 (NOT, ! , @ , 和 %)
	目标码完整性

区域	检查
操作数范围	操作数区域范围
	操作数据类型
	只读字的存取检查
	检查下列操作数范围： • 常数 (#, &, +, –) • 控制码 • 用于多字操作数区域范围检查 • 多字操作数大小关系检查 • 操作数范围重叠 • 多字分配 • 双字长操作数 • 偏移量区域检查
PLC 程序容量	步数
	总容量
	任务数
语法	要求成对指令的检查 • IL–ILC • JMP–JME, CJP/CJPN–JME • SBS–SBN–RET, MCRO–SBN–RET • STEP–SNXT • BPRG–BEND • IF–IEND • LOOP–LEND
	对 BPRG-BEND 编程位置限制
	对 SBN-RET 编程位置限制
	对 STEP-SNXT 编程位置限制
	对 FOR-NEXT 编程位置限制
	对中断任务编程位置限制
	对 BPRG-BEND 编程位置的要求
	对 FOR-NEXT 编程位置的要求
	非法嵌套
	END(001) 指令
	数字一致性
梯形图结构	堆栈溢出
重复输出	重复输出检查 • 测位 • 测字 • 定时器 / 计数器指令 • 长字 （2 个字和 4 个字） • 多重分配的字 • 开始结束范围 • FAL 数 • 多重输出操作数指令
任务	任务开始运行时检查任务设定
	任务程序分配

注　重复输出仅在单独的任务中检查，而在任务间不检查。

多字节操作数

程序检查中多字操作数内存区域边界检查如下表所示。

CX-Programmer	手持编程器
CX-Programmer 提供下列对多字操作数超出内存区域的功能。 • 该程序不能传送到 CPU 单元。 • 该程序也不能从 CPU 单元读出。 • 程序检查产生编辑出错。 • 离线编程时在屏幕上显示警告。 • 在线编程或监控方式下显示警告。	当程序输入时检查，即超出内存区域的操作数不能写入。

2-3-3 程序执行检查

操作数和指令位置检查在编程工具（包括手持编程器）作输入时和在编程工具（不包括手持编程器）作程序检查时完成。然而这些检查不是最终检查。

下列检查是在指令执行时完成。

错误类型	出错变为 ON 的标志	停止 / 继续运行
1. 指令处理出错	ER 标志 当错误出现，指定停止运行，指令处理出错标志 (A29508) 也将变为 ON。	在 PLC 设置中有一项设定可对指令处理出错发生时是否停止或继续运行作确定。默认设定作继续运行操作。 仅当确定为停止操作，程序出错标志会产生并停止运行。
2. 存 / 取出错	AER 标志 当存取错误出现，如果指定停止运行，存 / 取出错标志 （A29510）将会变为 ON。	在 PLC 设置中有一项设定可对指令处理出错发生时是否停止或继续运行作确定。默认设定作继续运行操作。 仅当确定为停止操作，程序出错标志会产生并停止运行。
3. 非法指令错误	非法指令错误标志 （A29514）	致命 （程序错误）
4.UM （用户存储器）溢出错误	UM 溢出错误标志 (A29515)	致命 （程序错误）

指令处理出错

当执行一条指令，如果提供的数据不正确或试图执行一条任务之外的指令，指令处理出错就会出现。这里，在指令处理开始时所需的数据被检查并作为检查结果，该指令不执行。ER 标志 (出错标志) 将变为 ON， EQ 和 N 标志可能保持不变或变为 OFF，这取决于这条指令。

如果这条指令 (包括输入指令) 结束，通常 ER 标志将变为 OFF。使 ER 标志变为 ON 的条件随各指令而不同。详情参阅 *CS/CJ 系列可编程序控制器编程手册 (W340)* 有关各指令的描述。

假如指令处理错误出现， ER 标志变为 ON。在 PLC 设置时设定指令错误停止运行，这时运行将停止 (致命错误), 并且指令处理出错标志 (A29508) 变为 ON。

非法存取错误

当定义指令操作数地址被存取时，非法存取错误表示以下方式中的一种存取错误区域。

a) 对参数区进行读或写操作。
b) 对未安装的内存区域进行写操作 （见注）。
c) 对定义为 EM 文件内存的 EM 区进行写操作。
d) 对只读区进行写操作。
e) 在间接 DM/EM 寻址中定义为 BCD 码，但 DM/EM 中内容不是 BCD 码（如：*D000001 值为 #A000）。

如果存取错误出现，指令处理仍将继续，并且出错标志 (ER 标志) 不变为 ON，但存取错误标志 （AER 标志）将会变为 ON。

注 一个存取错误会在下列情况出现：

- 对当前 Bank 指定 EM 地址超过 32767 （例如 E32768）。
- 对间接 EM 用二进制寻址，指定字内容为 8000 ～ FFFF Hex （例如：@EC-00001 内容为＃ 8000），指定末级 Bank （例如：C）。
- 对间接 EM 用二进制寻址，指定字内容为 8000 ～ FFFF Hex （例如：@EC-00001 内容为＃ 8000），指定当前 Bank （例如：C）。
- 含有一个位地址的一个 IR 寄存器用作一个字地址或含有一个内存字地址的一个 IR 寄存器用作位地址。

如果 PLC 设置设定指令出错停止运行方式，当一个非法存取错误出现则 AER 标志变为 ON, 运行停止 （致命错误），且 “非法存取错误标志” (A29510) 会变为 ON。

注 存取错误标志（AER 标志）在一个任务执行后不会被清除。如果指令出错设置为继续运行状态，这个标志可以在 END(001) 指令执行前得以监视，以知道是否有一个非法存取在一个任务程序中出现 （如果这个 AER 标志用手持编程器监视，最终的 AER 标志的状态是所有用户程序执行后可检测到的状态）。

其它错误

非法指令错误

非法指令错误表示想要执行的指令数据不在系统定义范围内，只要程序是用 CS/CJ 系列编程装置 （包括手持编程器）生成，这样的错误一般不会出现。

在实际中，即使这种错误出现，也把这种错误当作程序错误对待，运行将会停止 （致命错误），并且非法指令标志 (A29514) 将会变为 ON。

UM （用户内存）溢出错误

UM 溢出错误表示试图在定义为程序存储区的用户内存最后的地址后存放执行指令数据，只要程序是用 CS/CJ 系列编程装置（包括手持编程器）生成，这样的错误一般不会出现。

在实际中，即使这种错误出现，也把这种错误当作程序出错对待，运行将会停止 （致命错误），且 UM 溢出标志 (A29515) 将会变为 ON。

2-3-4 检查致命错误

下列错误是致命程序错误，如果其中任一点发生，CPU 单元将停止运行。当由程序错误导致运行停止，程序中止的任务编号将存储在 A294 中，程序中止地址将存储在 A298/A299 中。引起程序错误的原因可由这些信息确定。

地址	描述	存储数据
A294	如果运行中止是由于一个程序错误，这里将存储运行中止点处的任务类型和任务编号。 如果循环中无循环任务有效，将存储 FFFF Hex，即没有循环任务执行。	循环任务：0000 ～ 001F Hex （环任务 0 ～ 31） 中断任务：8000 ～ 80 FFHex （中断任务 0 ～ 255）
A298/A299	如果运行中止是由于一个程序错误，这里将以二进制数形式存储运行中止点处的程序地址。 如果 END(001) 指令忘记使用 （A29511 将为 ON），那里需要 END(001) 指令处的地址将被存储。 如果有一个任务执行错误（A29512 将为 ON），FFFFFFFF Hex 将存入 A298/A299。	A298: 程序地址低位部分 A299: 程序地址高位部分

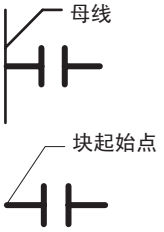
注 如果出错标志或存取错误标志变为 ON，它被看作程序错误，并可用它使 CPU 从运行变为停止。对程序错误可在 PLC 设置时定义操作方式。

程序错误	描述	对应标志
没有 END 指令	在程序中未出现一条 END 指令	无 END 标志 (A29511) 变为 ON。
任务执行过程中出错	无任务在循环中就绪。 没有程序分配给一个任务。 尽管中断任务指令执行条件满足，对应的中断任务标号未出现。	任务出错标志 (A29512) 变为 ON。
指令处理出错 （ER 标志为 ON）和在 PLC 设置中对指令错误设定为停止运行。	当试图去执行一条指令，在操作数中提供了错误的数据值。	如果在 PLC 设置中对指令错误设定为停止运行， ER 标志和指令处理错误标志 (A29508) 都变为 ON。
非法存取错误 （AER 标志为 ON）和在 PLC 设置中对指令错误设定为停止运行。	对参数区进行读或写操作。 对未安装的内存区域 (见注) 进行写操作。 对定义为 EM 文件内存的 EM 区域进行写操作。 对只读区进行写操作。 在间接 DM/EM 寻址中定义为 BCD 码值，但 DM/EM 中内容不是 BCD 码。	如果在 PLC 设置中对指令错误设定为停止运行， AER 标志和非法存取出错标志 (A29510) 变为 ON。
间接 DM/EM BCD 码错误和在 PLC 设置时对指令错误设定为停止运行。	在间接 DM/EM 寻址中定义为 BCD 码，但定义的值不是 BCD 码。	如果在 PLC 设置中对指令错误设定为停止运行， AER 标志和 DM/EM 间接 BCD 码出错标志 (A29509) 变为 ON。
微分地址溢出错误	在在线编辑过程中，插入或删除了超过 131， 071 条微分指令。	微分溢出错误标志 (A29513) 变为 ON。
UM （用户内存）溢出错误	试图在定义为程序存储区的用户内存最后的地址后存放指令数据。	UM(用户内存) 溢出标志 (A29516) 变为 ON。
非法指令错误	试图使一条不能执行的指令执行。	非法指令标志 (A29514) 变为 ON。

第 3 章
指令功能

本章介绍可用于用户程序编程的指令概要。

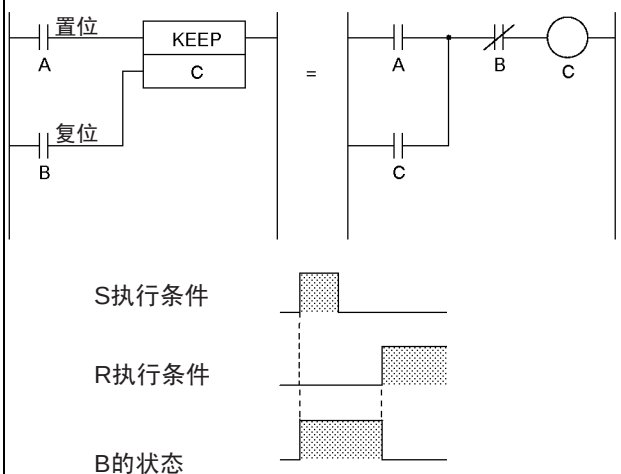
3-1 顺序输入指令

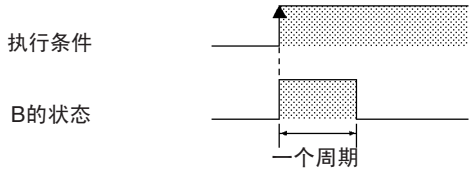
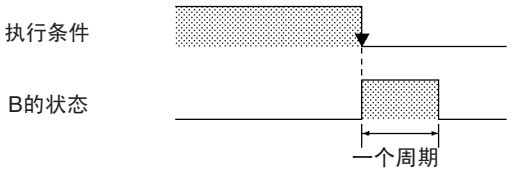
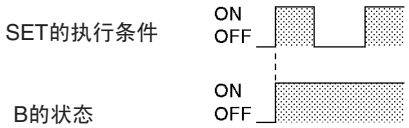
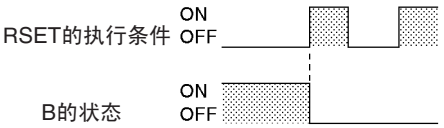
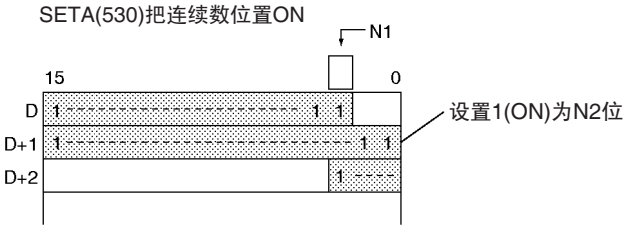
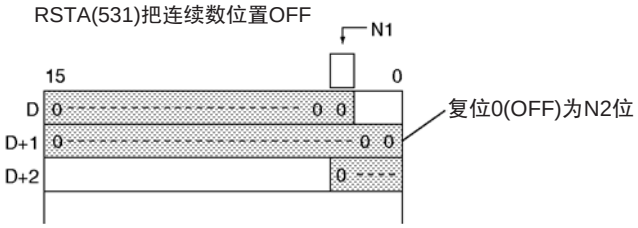
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
装载 LD @LD %LD !LD !@LD !%LD		指定一个逻辑开始，根据指定操作位的 ON/OFF 状态建立一个 ON/OFF 执行条件。	不需要
装载非 LD NOT @LD NOT %LD NOT !LD NOT !@LD NOT !%LD NOT CS1-H, CJ1-H 或 CJ1M CPU 单元仅：@LD NOT %LD NOT !@LD NOT !%LD NOT		指定一个逻辑开始，根据指定操作位的 ON/OFF 状态取非建立一个 ON/OFF 执行条件。	不需要
与 AND @AND %AND !AND !@AND !%AND		把指定的操作位状态和当前执行条件进行逻辑与操作。	需要
与非 AND NOT @AND NOT %AND NOT !AND NOT !@AND NOT !%AND NOT CS1-H, CJ1-H, or CJ1M CPU Units only: @AND NOT %AND NOT !@AND NOT !%AND NOT		把指定的操作位状态取非和当前执行条件进行逻辑与操作。	需要
或 OR @OR %OR !OR !@OR !%OR		把指定的操作位 ON/OFF 状态和当前执行条件进行逻辑或操作。	需要
或非 OR NOT @OR NOT %OR NOT !OR NOT !@OR NOT !%OR NOT CS1-H, CJ1-H, or CJ1M CPU Units only: @OR NOT %OR NOT !@OR NOT !%OR NOT		把指定的操作位状态取非和当前执行条件进行逻辑或操作。	需要

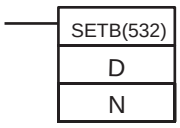
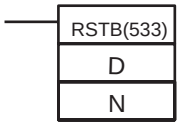
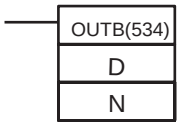
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
逻辑块与 AND LD	逻辑块 逻辑块	在逻辑块之间进行逻辑与 LD ~ 逻辑块A LD ~ 逻辑块B AND LD ····· 把逻辑块A和逻辑块B串联起来	需要
逻辑块或 OR LD	逻辑块 逻辑块	在逻辑块之间进行逻辑或 LD ~ 逻辑块A LD ~ 逻辑块B OR LD ····· 把逻辑块A和逻辑块B并联起来	需要
非 NOT 520	NOT(520)	执行条件取非。	需要
条件 ON UP 521	UP(521)	当执行条件从 OFF → ON,UP(521) 把执行条件在一个周期内变 ON。	需要
条件 OFF DOWN 522	DOWN(522)	当执行条件从 ON → OFF,DOWN(522) 把执行条件在一个周期内变 ON。	需要
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) 指令在程序中的用途类似于 LD, AND 和 OR 指令，当指定字中的指定位为 ON 时，执行条件为 ON，反之执行条件为 OFF。	不需要
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) 指令在程序中的用途类似于 LD NOT,ANT NOT 和 OR NOT 指令，当指定字中的指定位为 ON 时，执行条件为 OFF，反之执行条件为 ON。	不需要
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) 指令在程序中的用途似于 LD, AND 和 OR 指令，当指定字中的指定位为 ON 时，执行条件为 ON，反之执行条件为 OFF。	需要

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
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) 指令在程序中的用途类似于 LD NOT,ANT NOT 和 OR NOT 指令，当指定字中的指定位为 ON 时，执行条件为 OFF，反之执行条件为 ON。	需要
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) 指令在程序中的用途类似于 LD, AND 和 OR 指令，当指定字中的指定位为 ON 时，执行条件为 ON，反之执行条件为 OFF。	需要
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) 指令在程序中的用途类似于 LD NOT,ANT NOT 和 OR NOT 指令，当指定字中的指定位为 ON 时，执行条件为 OFF，反之执行条件为 ON。	需要

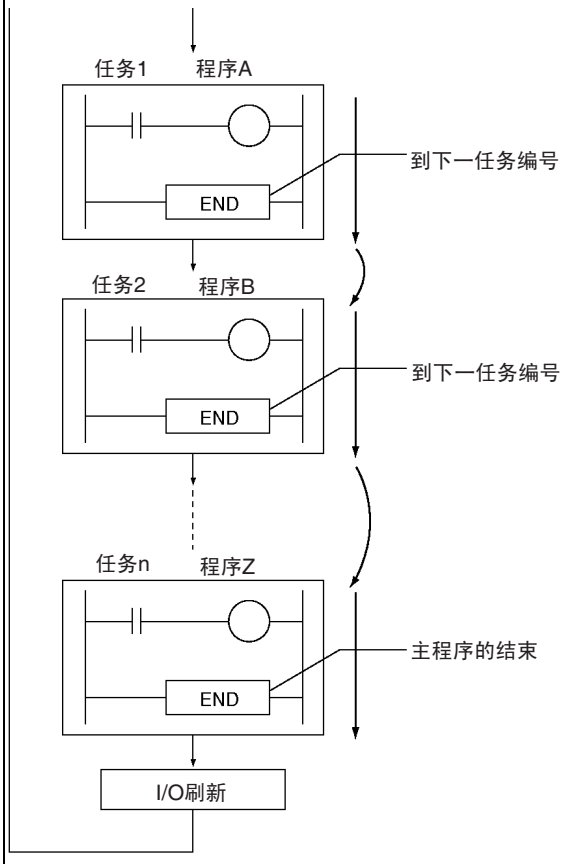
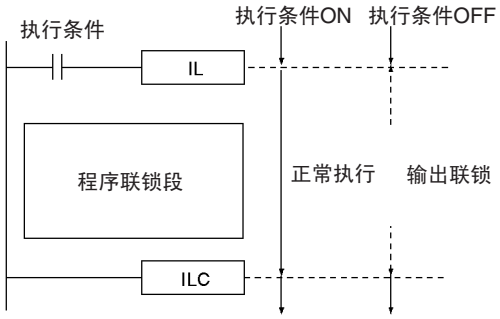
3-2 顺序输出指令

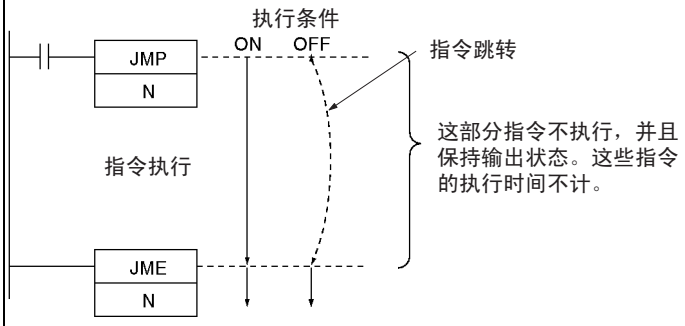
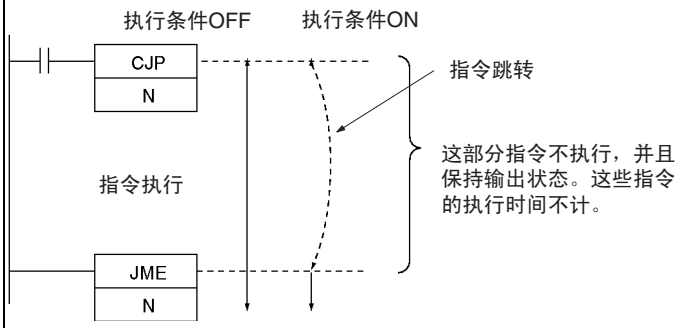
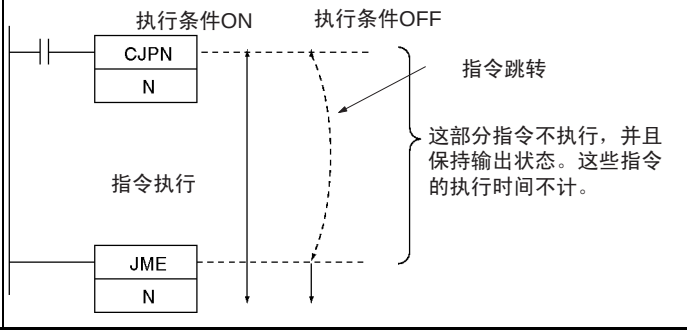
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
输出 OUT !OUT		把逻辑运算的结果 （执行条件）输出到指定位。	输出需要
输出非 OUT NOT !OUT NOT		把逻辑运算的结果 （执行条件）取非，再输出到指定位。	输出需要
保持 KEEP !KEEP 011	S (置位) KEEP(011) B R (复位) B:位	作为一个锁存继电器操作。 	输出需要

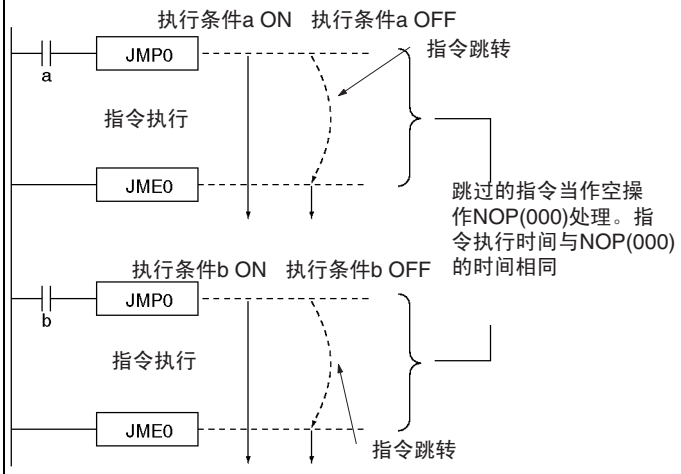
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
上升沿微分 DIFU !DIFU 013	DIFU(013) B B:位	当执行条件从OFF→ON（上升沿），DIFU(013)指令将所指定的位在一个周期内变为ON。 	输出需要
下降沿微分 DIFD !DIFD 014	DIFD(014) B B:位	当执行条件从ON→OFF（下降沿），DIFD(014)指令将所指定的位在一个周期内变为ON。 	输出需要
置位 SET @SET %SET !SET !@SET !%SET	SET B B:位	SET在执行条件为ON时，把操作位变为ON 	输出需要
复位 RSET @RSET %RSET !RSET !@RSET !%RSET	RSET B B:位	RSET在执行条件为ON时，把操作位变为OFF 	输出需要
多位置位 SETA @SETA 530	SETA(530) D N1 N2 D:开始字 N1:开始位 N2:位的个数	SETA(530)把连续数位置ON 	输出需要
多位复位 RSTA @RSTA 531	RSTA(531) D N1 N2 D:开始字 N1:开始位 N2:位的个数	RSTA(531)把连续数位置OFF 	输出需要

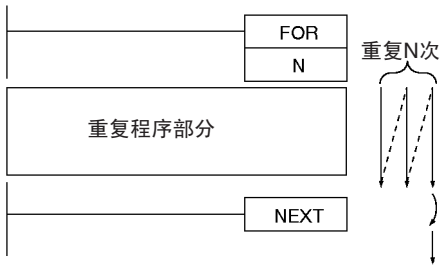
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
单个位置位 （仅限于 CS1-H，CJ1-H， CJM 或 CS1D） SETB @SETB !SETB 532	 D:字地址 N:位号	SETB(532) 在执行条件为 ON 时，把指定字中的指定位置 ON。 与 SET 指令不同， SETB （532）指令可用于对 DM 和 EM 字中的位置 ON。	输出需要
单个位复位 （仅限于 CS1 － H， CJ1 － H，CJM 或 CS1D） RSTB @RSTB !RSTB 533	 D:字地址 N:位号	RSTB(533) 在执行条件为 ON 时，把指定字中的指定位置 OFF。 与 RSET 指令不同， RSTB （533）指令可用于对 DM 和 EM 字中的位置 OFF。	输出需要
单个位输出 （仅适用于 CS1 － H， CJ1 － H，CJM 或 CS1D） OUTB @OUTB !OUTB 534	 D:字地址 N:位号	OUTB(534) 把逻辑运算的结果 （执行条件）输出到指定位。 与 OUT 指令不同， OUTB （534）指令可用于控制 DM 和 EM 字中的位。	输出需要

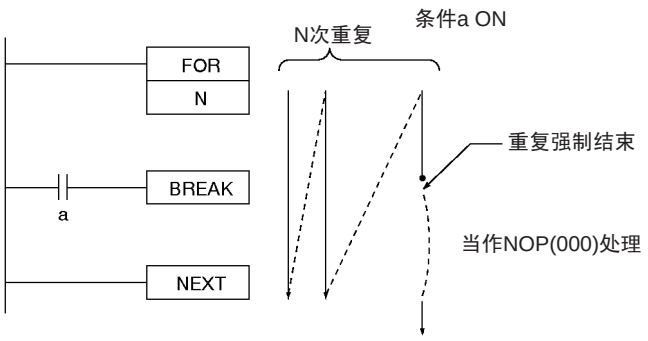
3-3 顺序控制指令

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
结束 END 001	END(001)	表示程序的结束。 END(001)作为一个循环内的程序执行完成。 在END(001)后的任何指令不执行。接着执行进行到下一编号任务。 当执行到程序中最高任务编号，END(001)意味着整个 主程序结束。  	输出不需要
空操作 NOP 000		这条指令无功能 （NOP(000) 不完成任何处理）。	输出不需要
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
联锁清除 ILC 003	ILC(003)	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。	输出不需要

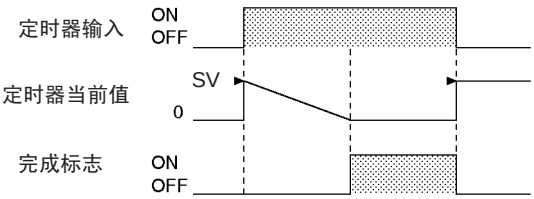
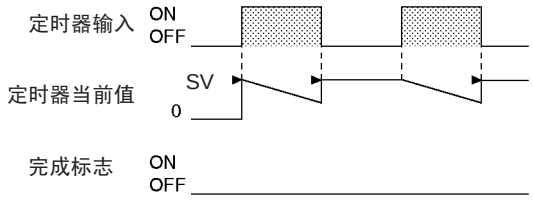
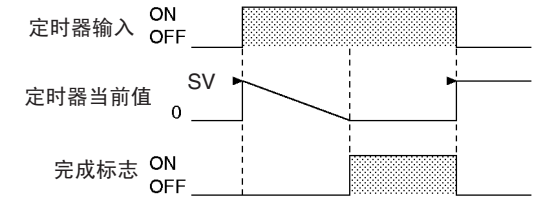
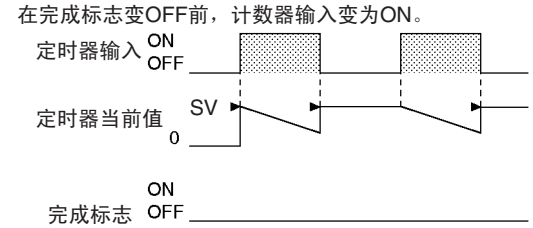
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
跳转 JMP 004	JMP(004) N N:跳转号	执行条件JMP（004）为OFF时，程序执行直接跳转至与JMP（004）指令相同编号的第一个JME（005）。JMP（004）和JME（005）总是成对使用。 	输出需要
跳转结束 JME 005	JME(005) N N:跳转号	表示从 JMP(004) 或 CJP(510) 开始的跳转结束。	输出不需要
条件跳转 CJP 510	CJP(510) N N:跳转号	CJP(510)的用法与JMP(004)相反 当CJP(510)的执行条件为ON时，程序执行直接跳转至与CJP(510)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
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CJPN(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(510)和JME(005)总是成对使用。 	输出不需要

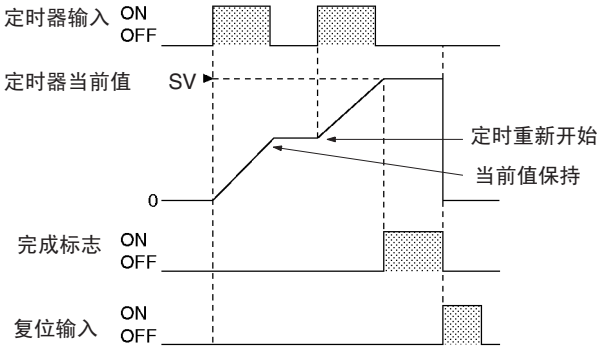
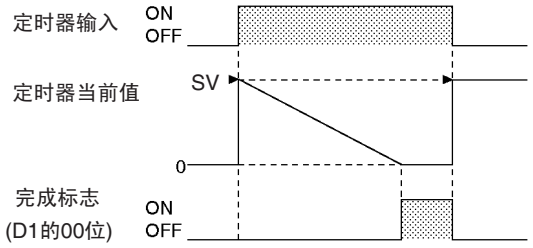
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
多路跳转 JMP0 515	JMP0(515)	当JMP0(515)的执行条件为OFF，从JMP0(515)至下一JME0(516)的所有指令都被当作空操作NOP(000)。JMP0(515)和JME0(516)成对使用，程序中能使用对数无限制。 	输出需要
多路跳转结束 JME0 516	JME0(516)	当 JMP0(515) 的执行条件为 OFF，从 JMP0(515) 至下一 JME0(516) 的所有指令都被当作空操作 NOP （000）。 JMP0(515) 和 JME0(516) 成对使用，程序中能使用对数无限制。	输出不需要

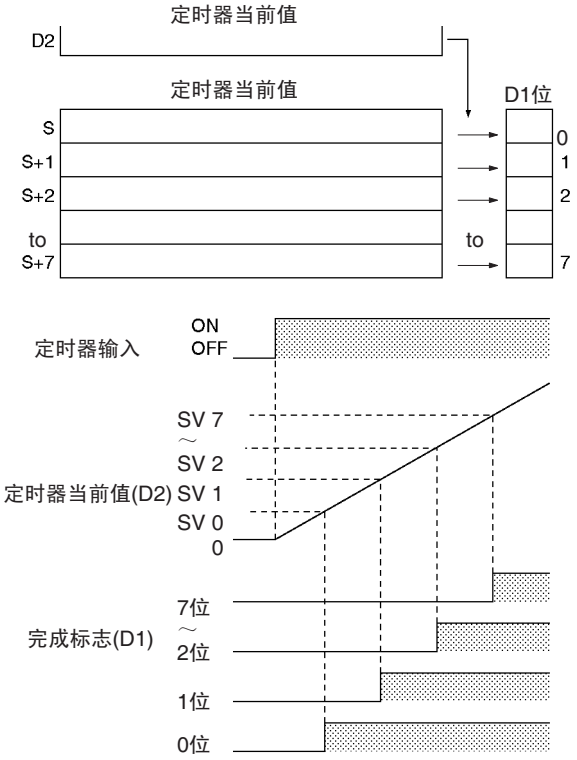
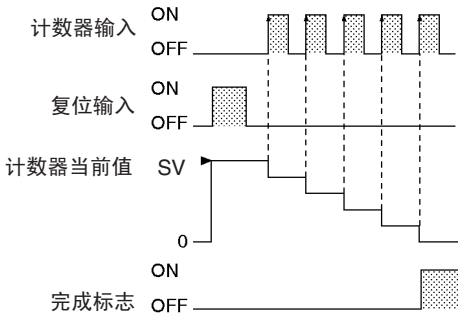
FOR-NEXT 循环
FOR
512
	FOR(512)
	N
	N:循环次数
		FOR(512)和NEXT(513)之间的指令按指定次数重复执行。
		FOR(512)和NEXT(513)指令成对使用。
		 
			输出不需要

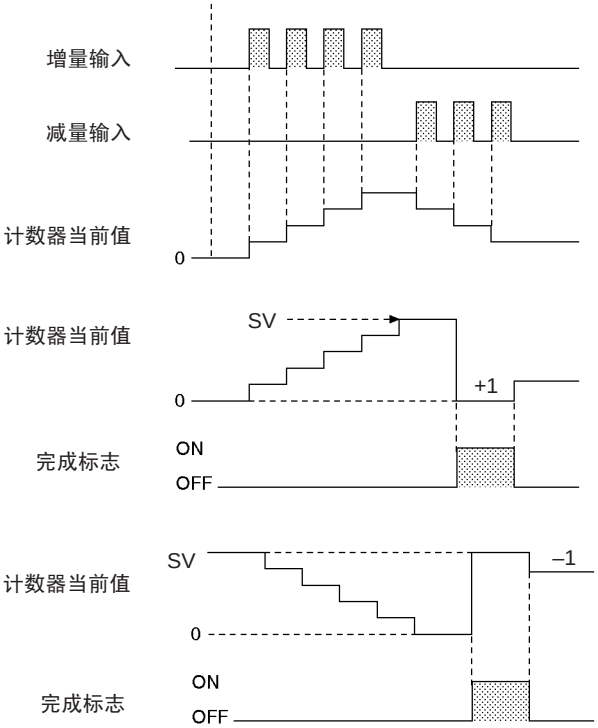
退出循环 BREAK 514	BREAK(514)	在FOR-NEXT循环中编程，对所给的执行条件取消循环指令。循环中余下的指令作NOP(000)处理。 	输出需要
FOR-NEXT 循环 NEXT 513	NEXT(513)	FOR(512) 和 NEXT(513) 之间的指令按指定次数重复执行。 FOR(512) 和 NEXT(513) 指令成对使用。	输出不需要

3-4 定时器和计数器指令

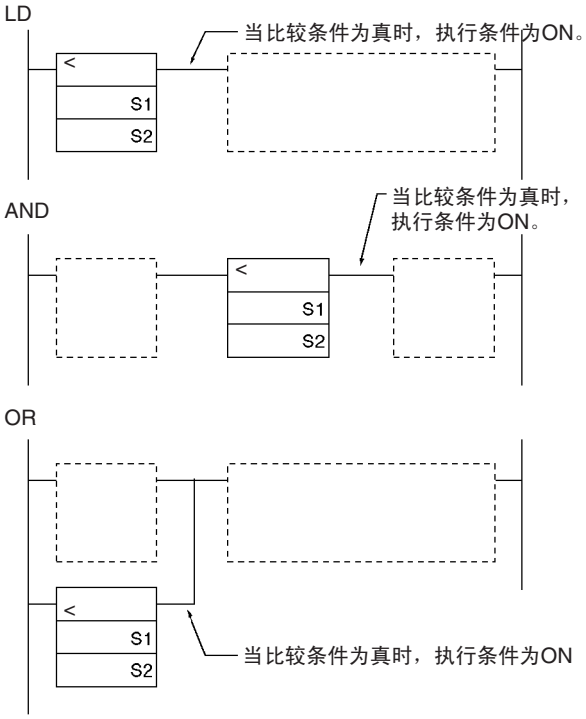
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
定时器 TIM(BCD)	TIM N S N:定时器编号 S:设定值	定时器以0.1s为单位作一减量定时。此设定值(SV)设定范围为0～999.9 s。 	输出需要
TIMX （二进制） （仅适用于 CS1-H，CJ1-H，CJ1M 或 CS1D）	TIMX(550) N S N:定时器编号 S:设定值	在完成标志变OFF前，计数器输入变为ON。 	
高速定时器 TIMH 015 (BCD)	TIMH(015) N S N:定时器编号 S:设定值	TIMH(015)以10ms为单位作一减量定时。此设定值(SV)设定范围为0～99.99 s。 	输出需要
TIMHX 551 （二进制） （仅适用于 CS1-H，CJ1-H，CJ1M 或 CS1D）	TIMHX(551) N S N:定时器编号 S:设定值	在完成标志变OFF前，计数器输入变为ON。 	
1MS 定时器 TMHH 540 (BCD)	TMHH(540) N S N:定时器编号 S:设定值	TIMH(540) 以 1ms 为单位作一减量定时。此设定值 （SV）设定范围为 0 ～ 9.999S。 TIMH(540) 的定时特性和上面所给的 TIMH(015) 特性相同。	输出需要
TMHHX 552 (BCD) （仅适用于 CS1-H，CJ1-H，CJ1M 或 CS1D）	TMHHX(552) N S N:定时器编号 S:设定值		

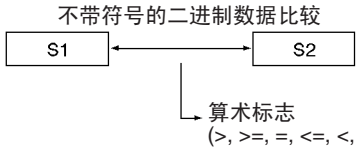
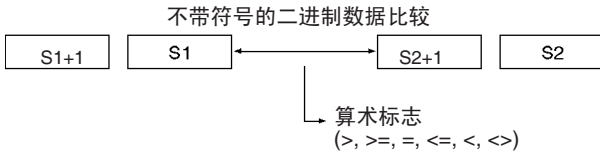
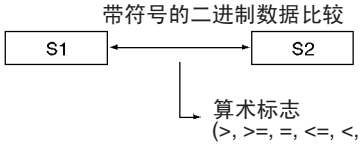
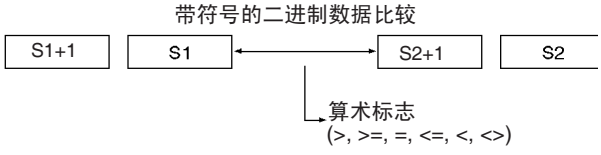
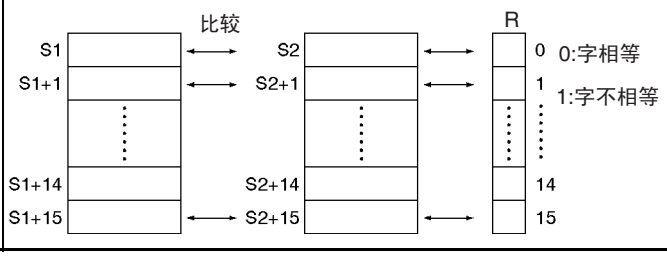
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
累加定时器 TTIM 087 (BCD)	定时器输入 TTIM(087) N S 复位输入 N:定时器编号 S:设定值	TTIM(087)以0.1s为单位作一减量定时。此设定值(SV)设定范围为0～999.9 s 。 	输出需要
TTIMX 555 （二进制） （仅适用于 CS1-H，CJ1-H，CJ1M 或 CS1D）	定时器输入 TTIMX(555) N S 复位输入 N:定时器编号 S:设定值		
长定时器 TIML 542 (BCD)	TIML(542) D1 D2 S D1:完成标志 D2:当前值字 S:设定值字	TIML(542)以0.1s为单位作一减量定时。定时最长可达9999999.9 S（约115天） 	输出需要
TIMLX 553 （二进制） （仅适用于 CS1-H，CJ1-H，CJ1M 或 CS1D）	TIMLX(553) D1 D2 S D1:完成标志 D2:当前值字 S:设定值字		

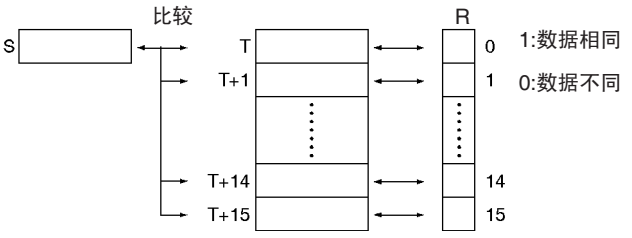
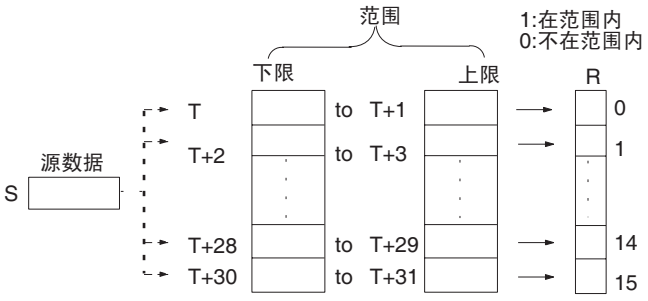
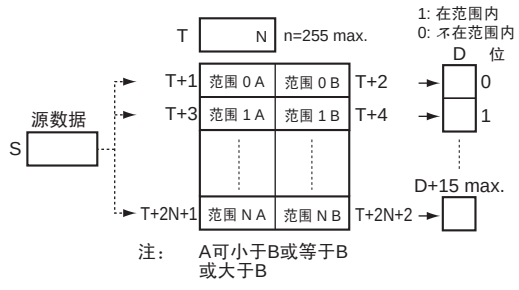
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
多路输出定时器 MTIM 543 (BCD)	MTIM(543) D1 D2 S D1:完成标志 D2:当前值字 S:第一设定值	MTIM(543)以0.1s为单位作一减量定时，此定时器有八个独立的(SV)和完成标志。此设定值(SV)设定范围为0～999.9 s 	输出需要
MTIMX 554 （二进制） （仅适用于 CS1-H，CJ1-H，CJ1M 或 CS1D）	MTIMX(554) D1 D2 S D1:完成标志 D2:当前值字 S:第一设定值		
计数器 CNT (BCD)	计数器输入 CNT N S 复位输入 N:计数器编号 S:设定值	CNT以减量计数方式运行，设定值的范围为0～9,999 	输出需要
CNTX 546 （二进制） （仅适用于 CS1-H，CJ1-H，CJ1M 或 CS1D）	计数器输入 CNTX(546) N S 复位输入 N:计数器编号 S:设定值		

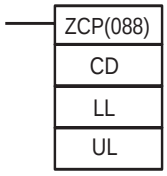
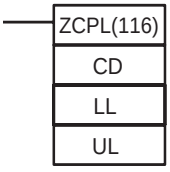
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
可逆计数器 CNTR 012 (BCD)	增量输入 减量输入 复位输入 CNTR(012) N S N:计数器编号 S:设定值	CNTR(012)以可逆计数器方式操作。 	输出需要
CNTRX 548 （二进制） （仅适用于 CS1-H，CJ1-H，CJ1M 或 CS1D）	增量输入 减量输入 复位输入 CNTRX(548) N S N:计数器编号 S:设定值		
复位定时器 / 计数器 CNR @CNR 545 (BCD)	CNR(545) N1 N2 N_1:范围中第一个编号 N_2:范围中最后一个编号	在一指定定时器或计数器编号范围内复位定时器和计数器。把设定值设置最大为 9999。	输出需要
CNRX @CNRX 547 （二进制） （仅适用于 CS1-H，CJ1-H，CJ1M 或 CS1D）	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		

3-5 比较指令

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
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两个数值（常数和/或指定字的内容）进行比较，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
符号比较 （双字，不带符号） LD, AND, OR + =, <>, <, <=, >, >= + L 301 (=) 306 (<>) 311 (<) 316 (<=) 321 (>) 326 (>=)	S1: 比较数据 1 S2: 比较数据 2	符号比较指令 （双字长，不带符号）以不带符号 32 位二进制数据对两个数值（常数和 / 或指定字双字长数据的内容）进行比较，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
符号比较 （带符号） LD, AND, OR + =, <>, <, <=, >, >= +S 302 (=) 307 (<>) 312 (<) 317 (<=) 322 (>) 327 (>=)	S1: 比较数据 1 S2: 比较数据 2	符号比较指令 （带符号）以带符号 16 位二进制 （4 位 16 进制数）对两个数值（常数和 / 或指定字的内容）进行比较，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
符号比较 （带符号双字长） LD, AND, OR + =, <>, <, <=, >, >= +SL 303 (=) 308 (<>) 313 (<) 318 (<=) 323 (>) 328 (>=)	S1: 比较数据 1 S2: 比较数据 2	符号比较指令 （双字长，带符号）以带符号 32 位二进制 （8 位 16 进制数）对两个数值 （常数和 / 或指定双字长数据的内容）进行比较，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

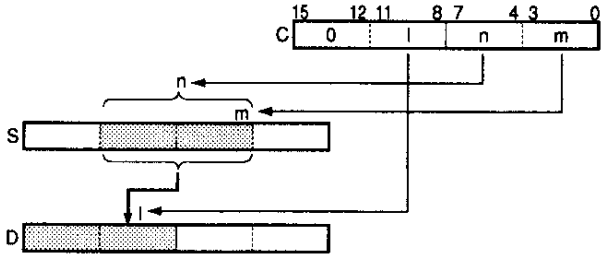
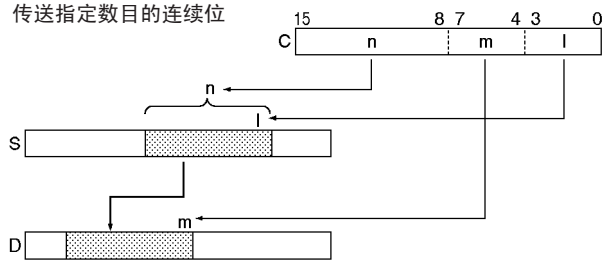
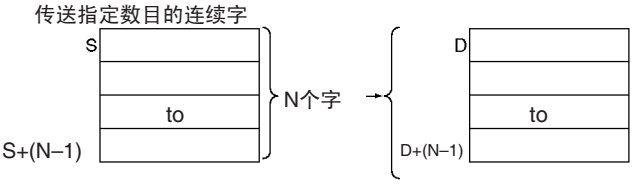
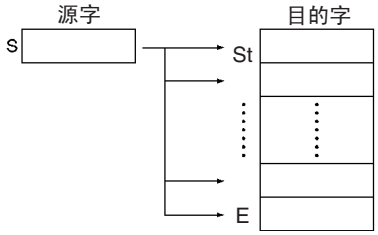
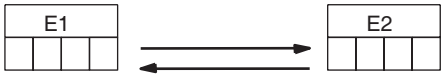
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
不带符号比较 CMP !CMP 020	CMP(020) S_1 S_2 S_1:比较数据1 S_2:比较数据2	比较两个不带符号的二进制数值（常数和/或指定字的内容），并把结果输出到辅助区的算术标志中。 	输出需要
双字长不带符号比较 CMPL 060	CMPL(060) S_1 S_2 S_1:比较数据1 S_2:比较数据2	比较两个双字长不带符号的二进制数值（常数和/或指定字的内容），并把结果输出到辅助区的算术标志中。 	输出需要
带符号二进制数比较 CPS !CPS 114	CPS(114) S_1 S_2 S_1:比较数据1 S_2:比较数据2	比较两个带符号的二进制数值（常数和/或指定字的内容），并把结果输出到辅助区的算术标志中。 	输出需要
带符号双字长二进制数比较 CPSL 115	CPSL(115) S_1 S_2 S_1:比较数据1 S_2:比较数据2	比较两个双字长带符号的二进制数值（常数和/或指定字的内容），并把结果输出到辅助区的算术标志中。 	输出需要
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16个字进行比较，并当比较内容不相等时使结果字中的相应位为ON。 	输出需要

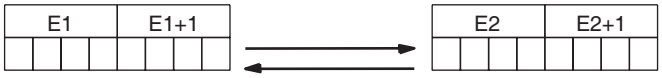
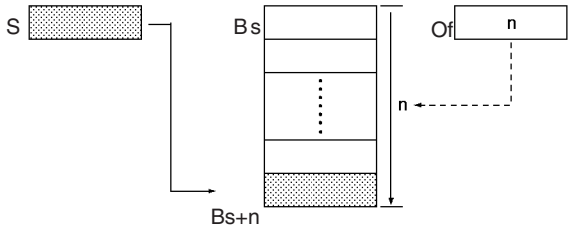
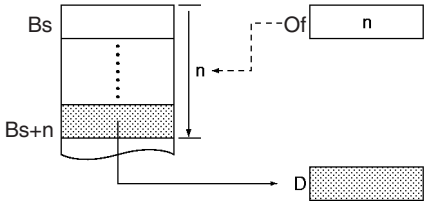
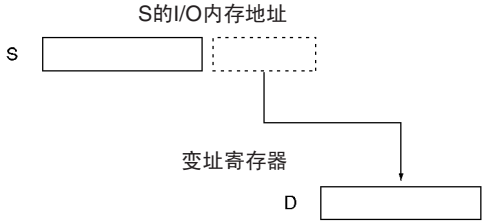
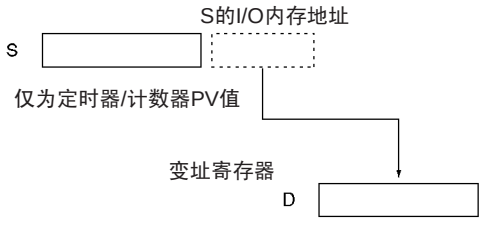
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
表比较 TCMP @TCMP 085	TCMP(085) S T R S:源数据 T:表格首字 R:结果字	将源数据和16个连续字的内容比较，与字内容相同时，使结果字中相应位为ON 	输出需要
不带符号的块比较 BCMP @BCMP 068	BCMP(068) S T R S:源数据 T:表格首字 R:结果字	将源数据在16个范围（以16个下限和16个上限来定义）内比较，源数据在范围内时，使结果字中相应位为ON。 	输出需要
扩展块比较 BCMP2 @BCMP2 502 （仅适用于CJ1M）	BCMP2(502) S T R S:源数据 T:表格首字 R:结果字	将源数据在 256 个范围（由上、下限定义）内比较，源数据在范围内时，使结果字中相应位变 ON。 	输出需要

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
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，将输出结果送辅助区算术标志。	输出需要
双字长区域范围比较 （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，将输出结果送辅助区算术标志。	输出需要

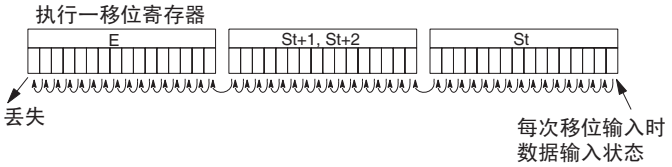
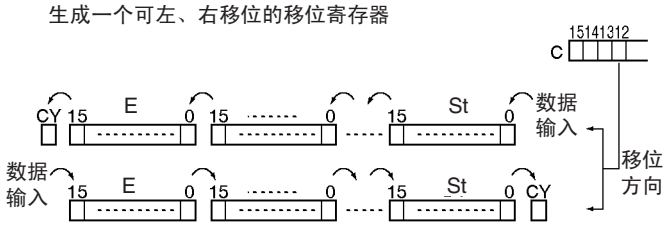
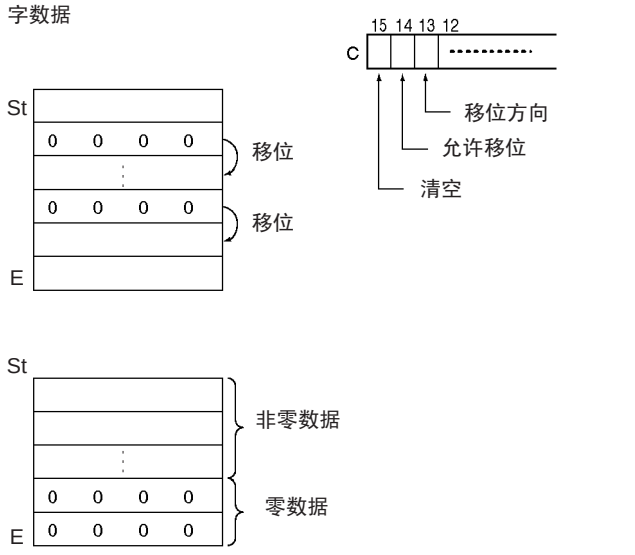
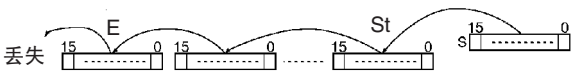
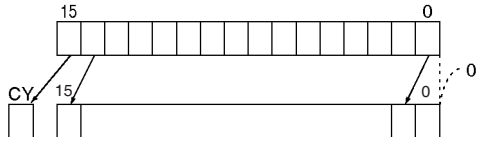
3-6 数据传送指令

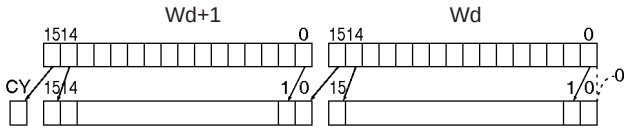
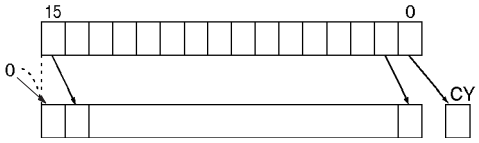
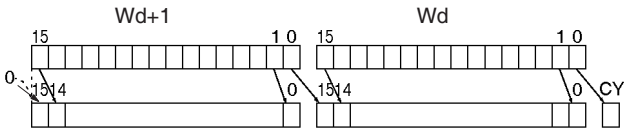
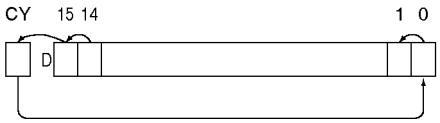
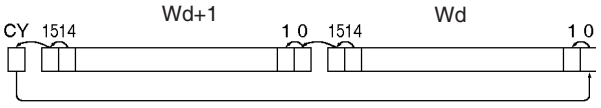
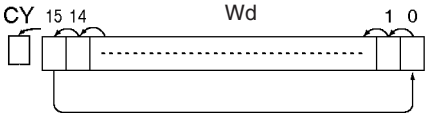
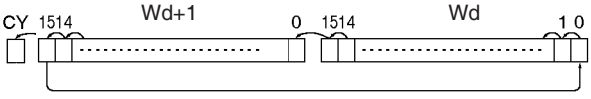
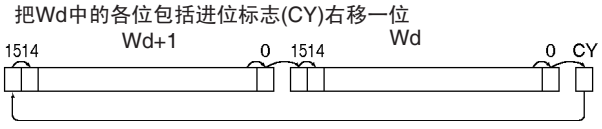
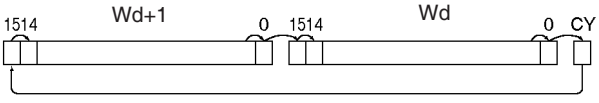
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
传送 MOV @MOV !MOV !@MOV 021	MOV(021) S D S:源 D:目的	把一个字的数据传送到指定的字中。 源字 不改变位的状态 目的字	输出需要
双字长传送 MOVL @MOVL 498	MOVL(498) S D S:源起始字 D:目的起始字	把两个字的数据传送到指定的字中。 S S+1 不改变位的状态 D D+1	输出需要
取反传送 MVN @MVN 022	MVN(022) S D S:源 D:目的	把一个字的数据的反码传送到指定的字中。 源字 位状态取反 目的字	输出需要
双字长取反传送 MVNL @MVNL 499	MVNL(499) S D S:源起始字 D:目的起始字	把两个字的数据的反码传送到指定的字中。 S S+1 位状态取反 D D+1	输出需要
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 C m n n S m D	输出需要

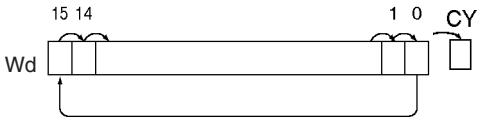
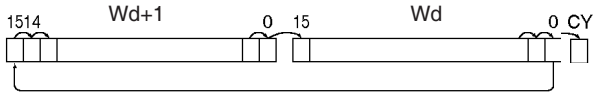
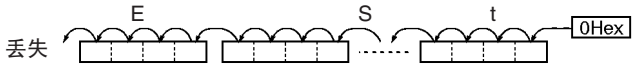
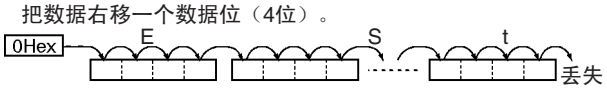
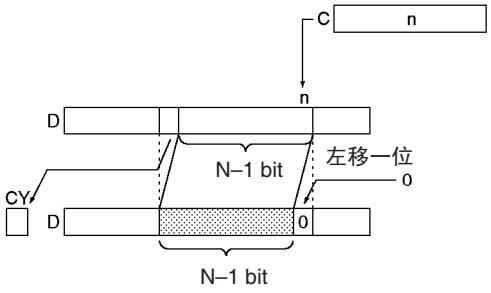
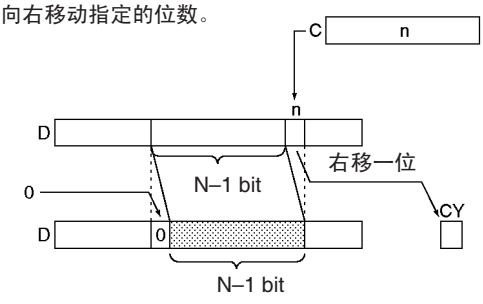
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
数字传送 MOVD @MOVD 083	MOVD(083) S C D S:源字或数据 C:控制字 D:目的字	传送指定的数字或多个数字（每个数字由4位组成） 	输出需要
多位传送 XFRB @XFRB 062	XFRB(062) C S D C:控制字 S:源起始字 D:目的起始字	传送指定数目的连续位 	输出需要
块传送 XFER @XFER 070	XFER(070) N S D N:字的数目 S:源起始字 D:目的起始字	传送指定数目的连续字 	输出需要
块设定 BSET @BSET 071	BSET(071) S St E S:源字 St:开始字 E:结束字	将一个字的内容复制到几个连续的字中。 	输出需要
数据交换 XCHG @XCHG 073	XCHG(073) E1 E2 E1:第一交换字 E2:第二交换字	把两个字的内容进行交换。 	输出需要

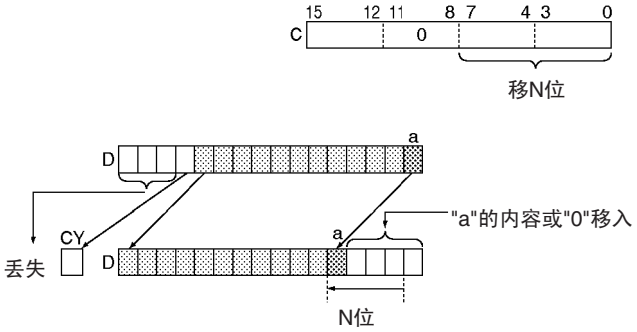
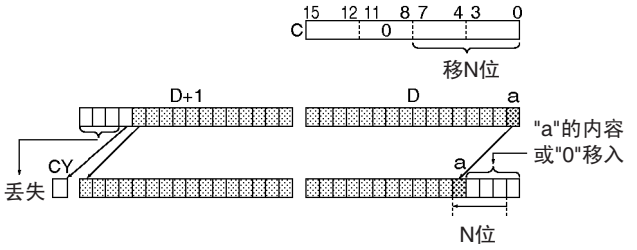
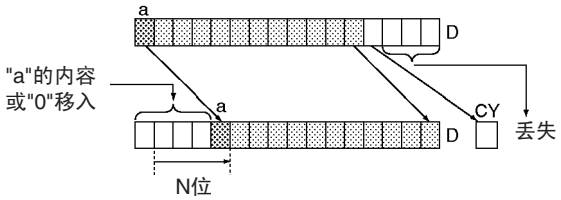
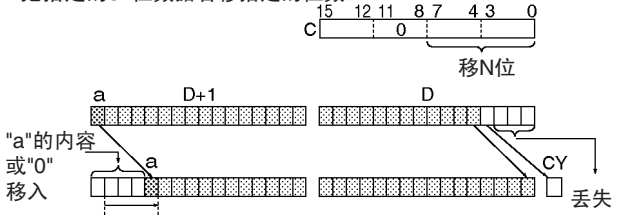
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
双字长数据交换 XCGL @XCGL 562	XCGL(562) E1 E2 E1:第一交换字 E2:第二交换字	把两个连续字的内容与另两个连续字的内容进行交换。 	输出需要
单字分配 DIST @DIST 080	DIST(080) S Bs Of S:源字 Bs:目的的基地址 Of:偏移	把源字传送到目的字中，目的字的地址由基地地址加上一个偏移值。 	输出需要
数据采集 COLL @COLL 081	COLL(081) Bs Of D Bs:源基地址 Of :偏移 D: 目的字	把源字（在源基地地址加上一个偏移值）传送到目的字。 	输出需要
传送到寄存器 MOVR @MOVR 560	MOVR(560) S D S:源 （所需的字或位） D:目的 （变址寄存器）	把指定的字、位或定时器/计数器完成标志的PC内存地址存放到指定的变址寄存器中（使用MOVRW（561）指令把一个定时器/计数器PV内存地址存放到一个变址寄存器中） 	输出需要
传送定时器 / 计数器 PV 值地址至寄存器 MOVRW @MOVRW 561	MOVRW(561) S D S:源 (所需TC编号） D:目的 （索引寄存器）	把指定定时器/计数器PV值的PC内存地址存放到指定的变址寄存器中（使用MOVR(560)指令把一个字、位或定时器/计数器的完成标志的PC内存地址存放到一个变址寄存器中） 	输出需要

3-7 数据移位指令

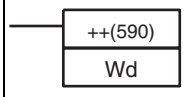
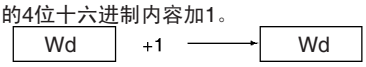
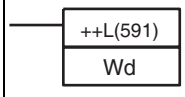
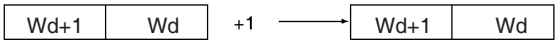
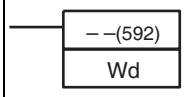
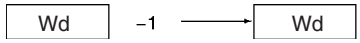
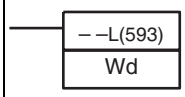
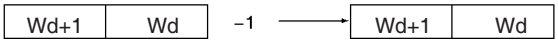
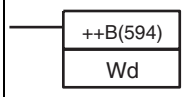
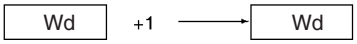
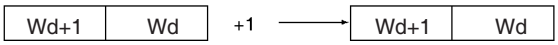
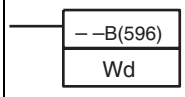
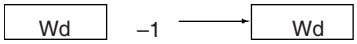
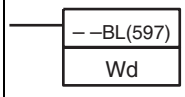
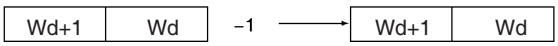
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
移位寄存器 SFT 010	数据输入 移位输入 复位输入 SFT(010) St E St:开始字 E:结束字	执行一移位寄存器 	输出需要
可逆移位寄存器 SFTR @SFTR 084	SFTR(084) C St E C:控制字 St:开始字 E:结束字	生成一个可左、右移位的移位寄存器 	输出需要
异步移位寄存器 ASFT @ASFT 017	ASFT(017) C St E C:控制字 St:开始字 E:结束字	在指定字范围内朝St或E方向移动非零字数据，以取代0000Hex字数据 	输出需要
字移位 WSFT @WSFT 016	WSFT(016) S St E S:源字 St:开始字 E:结束字	把St和E之间的数据按字移位 	输出需要
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

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
双字长左移 ASLL @ASLL 570	ASLL(570) Wd Wd:字	把Wd和Wd +1的内容左移一位 	输出需要
算术右移 ASR @ASR 026	ASR(026) Wd Wd:字	把Wd的内容右移一位 	输出需要
双字长右移 ASRL @ASRL 571	ASRL(571) Wd Wd:字	把Wd和Wd +1的内容右移一位 	输出需要
循环左移 ROL @ROL 027	ROL(027) Wd Wd:字	把Wd中的各位包括进位标志(CY)左移一位 	输出需要
双字长循环左移 ROLL @ROLL 572	ROLL(572) Wd Wd:字	把Wd和Wd+1中的各位包括进位标志(CY)左移一位 	输出需要
无进位循环左移 RLNC @RLNC 574	RLNC(574) Wd Wd:字	把Wd中的各位不包括进位标志(CY)左移一位 	输出需要
双字长无进位循环左移 RLNL @RLNL 576	RLNL(576) Wd Wd:字	把Wd和Wd+1中的各位不包括进位标志(CY)左移一位 	输出需要
循环右移 ROR @ROR 028	ROR(028) Wd Wd:字	把Wd中的各位包括进位标志(CY)右移一位 	输出需要
双字长循环右移 RORL @RORL 573	RORL(573) Wd Wd:字	把Wd和Wd+1中的各位包括进位标志(CY)右移一位 	输出需要

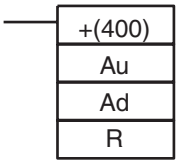
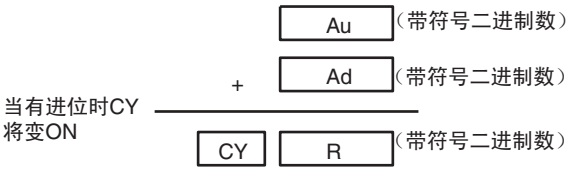
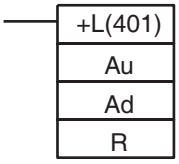
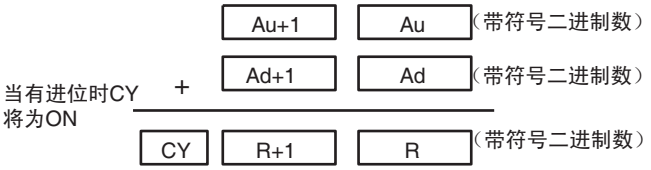
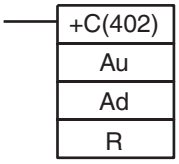
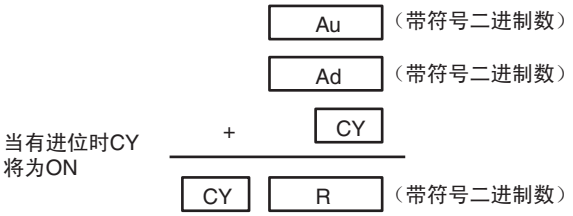
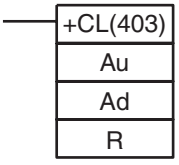
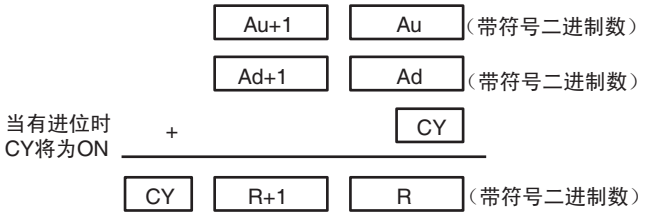
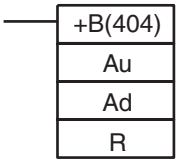
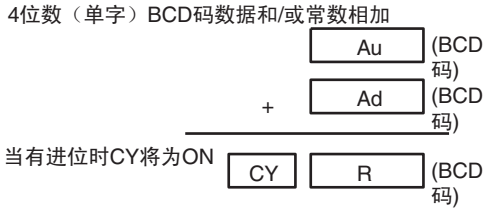
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
无进位循环右移 RRNC @RRNC 575	RRNC(575) Wd Wd:字	把Wd中各位不包括进位标志(CY)右移一位。Wd的最右位的内容移至最左位和进位标志(CY)。 	输出需要
双字长无进位循环右移 RRNL @RRNL 577	RRNL(577) Wd Wd:字	把Wd和Wd+1中各位不包括进位标志(CY)右移一位。Wd的最右位的内容移至Wd+1最左位和进位标志(CY)。 	输出需要
一个数字左移 SLD @SLD 074	SLD(074) St E St:开始字 E:结束字	把数据左移一个数据位（4位）。 	输出需要
一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

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
左移 N 位 NASL @NASL 580	NASL(580) D C D:移位字 C:控制字	把指定的16位数据左移指定的位数。 	输出需要
双字长左移 N 位 NSLL @NSLL 582	NSLL(582) D C D:移位字 C:控制字	把指定的32位数据左移指定的位数。 	输出需要
右移 N 位 NASR @NASR 581	NASR(581) D C D:移位字 C:控制字	把指定的16位数据右移指定的位数。 	输出需要
双字长右移 N 位 NSRL @NSRL 583	NSRL(583) D C D:移位字 C:控制字	把指定的32位数据右移指定的位数。 	输出需要

3-8 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
二进制递增 ++ @++ 590	 Wd:字	把指定字的4位十六进制内容加1。 	输出需要
双字长二进制递增 ++L @++L 591	 Wd:字	把指定字的8位十六进制内容加1。 	输出需要
二进制递减 – – @– – 592	 Wd:字	把指定字的4位十六进制内容减1。 	输出需要
双字长二进制递减 – –L @– –L 593	 Wd:首字	把指定字的8位十六进制内容减1。 	输出需要
BCD 码递增 ++B @++B 594	 Wd:字	把指定字的4位BCD码内容加1。 	输出需要
双字长 BCD 码递增 ++BL @++BL 595	 Wd:首字	把指定字的8位BCD码内容加1。 	输出需要
BCD 码递减 – –B @– –B 596	 Wd:字	把指定字的4位BCD码内容减1。 	输出需要
双字长 BCD 码递减 – –BL @– –BL 597	 Wd:首字	把指定字的8位BCD码内容减1。 	输出需要

3-9 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
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
无进位带符号双字长二进制加法 +L @+L 401	 Au :起始被加字 Ad:起始加字 R: 起始结果字	8位数（双字）十六进制数据和/或常数相加 	输出需要
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）十六进制数据和/或常数带进位标志(CY)相加 	输出需要
有进位带符号双字长二进制加法 +CL @+CL 403	 Au:被加首字 Ad:加首字 R:结果首字	8位数（双字）十六进制数据和/或常数带进位标志(CY)相加 	输出需要
无进位 BCD 码加法 +B @+B 404	 Au:被加字 Ad:加字 R:结果字	4位数（单字）BCD码数据和/或常数相加 	输出需要

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
无进位双字长 BCD 码加法 +BL @+BL 405	+BL(405) Au Ad R Au:起始被加字 Ad:起始加字 R:起始结果字	8位数（双字）BCD码数据和/或常数相加 Au +1 Au (BCD码) 当有进位时CY将为ON + Ad+1 Ad (BCD码) CY R+1 R (BCD码)	输出需要
有进位 BCD 码加法 +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 Au (BCD码) Ad (BCD码) 当有进位时CY将变ON + CY CY R (BCD码)	输出需要
有进位双字长 BCD 码加法 +BCL @+BCL 407	+BCL(407) Au Ad R Au:起始被加字 Ad:起始加字 R:起始结果字	8位数（双字）BCD码数据和/或常数及进位标志(CY)相加 Au +1 Au (BCD码) Ad+1 Ad (BCD码) 当有进位时CY将变ON + CY CY R+1 R (BCD码)	输出需要
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 Mi （带符号二进制数） – Su （带符号二进制数） 当有借位时CY变ON CY R （带符号二进制数）	输出需要
无进位带符号双字长二进制减法 –L @–L 411	–L(411) Mi Su R Mi:被减字 Su:减字 R:结果字	8位数（双字）十六进制数据和/或常数相减 Mi+1 Mi （带符号二进制数） – Su+1 Su （带符号二进制数） 当有借位时CY 变ON CY R+1 R （带符号二进制数）	输出需要
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 （带符号二进制数） Su （带符号二进制数） – CY 当有借位时CY变ON CY R （带符号二进制数）	输出需要

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
有进位带符号双字长二进制减法 –CL @–CL 413	–CL(413) Mi Su R Mi:被减字 Su:减字 R:结果字	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 （带符号二进制数） Su+1 Su （带符号二进制数） 当有借位时 CY 变ON – CY CY R+1 R （带符号二进制数）	输出需要
无进位 BCD 码减法 –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码数据和/或常数相减 Mi (BCD码) – Su (BCD码) 当有借位时CY变ON CY R (BCD码)	输出需要
无进位双字长 BCD 码减法 –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码数据和/或常数相减 Mi +1 Mi (BCD码) Su+1 Su (BCD码) 当有借位时 CY变ON – CY R+1 R (BCD码)	输出需要
有进位 BCD 码减法 –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码数据和/或常数及进位标志（CY）相减 Mi (BCD码) Su (BCD码) – CY 当有借位时 CY变ON CY R (BCD码)	输出需要
有进位双字长 BCD 码减法 –BCL @–BCL 417	–BCL(417) Mi Su R Mi:被减首字 Su：减首字 R:结果首字	8位数（双字）BCD码数据和/或常数及进位标志（CY）相减 Mi +1 Mi (BCD码) Su+1 Su (BCD码) – CY 当有借位时 CY变ON CY R+1 R (BCD码)	输出需要

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
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（带符号二进制数） × Mr（带符号二进制数） R +1 | R（带符号二进制数）	输出需要
带符号双字长二进制乘法 *L @*L 421	*L(421) Md Mr R Md:被乘首字 Mr: 乘首字 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（带符号二进制数） × Mr + 1 | Mr（带符号二进制数） R + 3 | R + 2 | R + 1 | R（带符号二进制数）	输出需要
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（不带符号二进制数） × Mr（不带符号二进制数） R +1 | R（不带符号二进制数）	输出需要
不带符号双字长二进制乘法 *UL @*UL 423	*UL(423) Md Mr R Md:被乘首字 Mr: 乘首字 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（不带符号二进制数） × Mr + 1 | Mr（不带符号二进制数） R + 3 | R + 2 | R + 1 | R（不带符号二进制数）	输出需要
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码数据和/或常数相乘 Md (BCD码) × Mr (BCD码) R +1 | R (BCD码)	输出需要

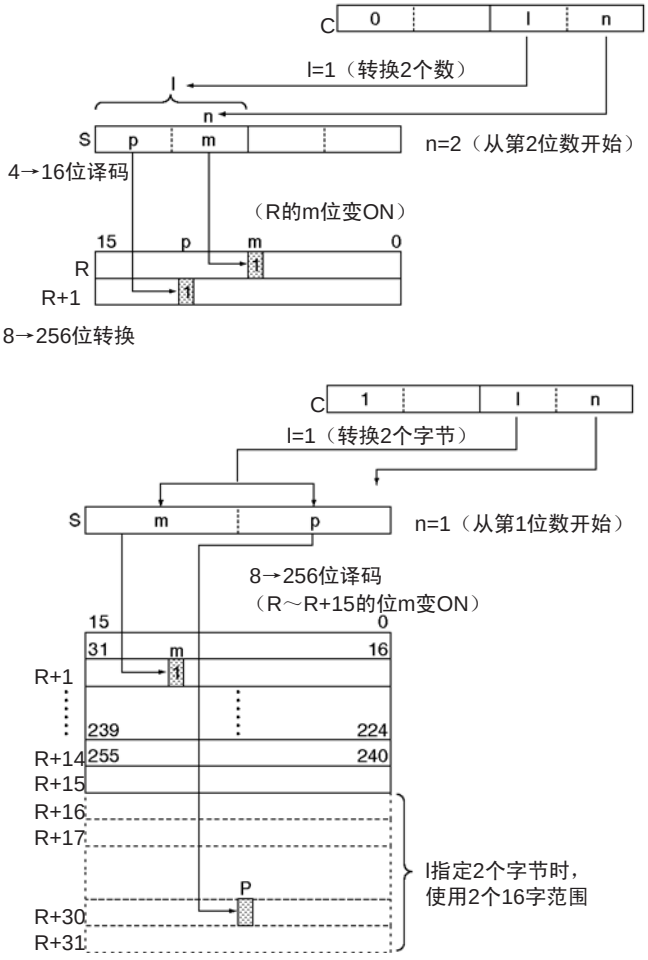
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
双字长 BCD 码乘法 *BL @*BL 425	*BL(425) Md Mr R Md:被乘首字 Mr: 乘首字 R:结果首字	8位数（双字）BCD码数据和/或常数相乘 Md + 1 | Md （BCD码） × Mr + 1 | Mr （BCD码） R + 3 | R + 2 | R + 1 | R （BCD码）	输出需要
带符号二进制除法 / @/ 430	/(430) Dd Dr R Dd:被除字 Dr:除字 R:结果字	4位数（单字）带符号十六进制数据和/或常数相除 Dd （带符号二进制数） ÷ Dr （带符号二进制数） R +1 | R （带符号二进制数） 余数 商	输出需要
带符号双字长二进制除法 /L @/L 431	/L(431) Dd Dr R Dd:被除首字 Dr:除首字 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 （带符号二进制数） ÷ Dr + 1 | Dr （带符号二进制数） R + 3 | R + 2 | R + 1 | R （带符号二进制数） 余数 商	输出需要
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 （不带符号二进制数） ÷ Dr （不带符号二进制数） R +1 | R （不带符号二进制数） 余数 商	输出需要
不带符号双字长二进制除法 /UL @/UL 433	/UL(433) Dd Dr R Dd:被除首字 Dr:除首字 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 （不带符号二进制数） ÷ Dr + 1 | Dr （不带符号二进制数） R + 3 | R + 2 | R + 1 | R （不带符号二进制数） 余数 商	输出需要

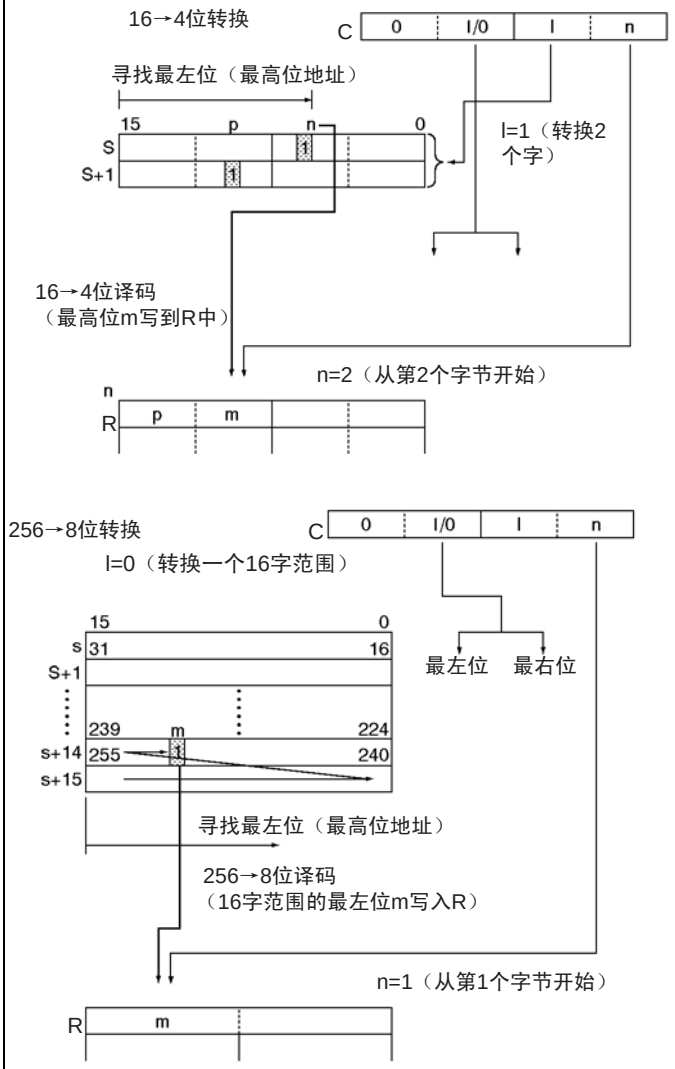
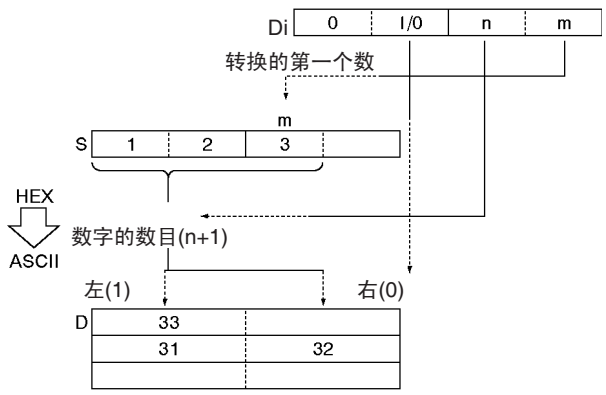
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
BCD 码除法 /B @/B 434	/B(434) Dd Dr R Dd:被除字 Dr: 除字 R:结果字	4位数（单字）BCD码数据和/或常数相除 Dd (BCD码) ÷ Dr (BCD码) R +1 R (BCD码) 余数 商	输出需要
双字长 BCD 码除法 /BL @/BL 435	/BL(435) Dd Dr R Dd:被除首字 Dr: 除首字 R:结果首字	8位数（双字）BCD码数据和/或常数相除 Dd + 1 Dd (BCD码) ÷ Dr + 1 Dr (BCD码) R + 3 R + 2 R + 1 R (BCD码) 余数 商	输出需要

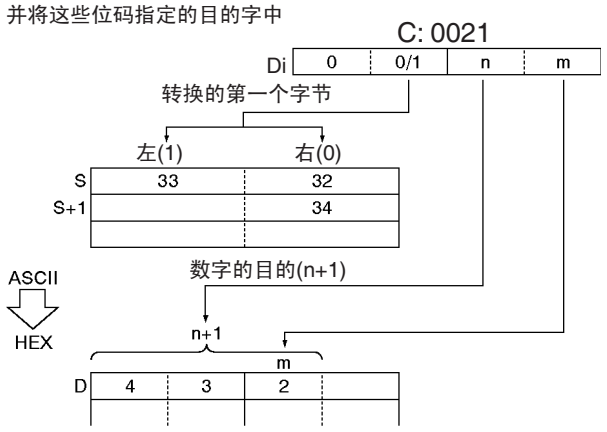
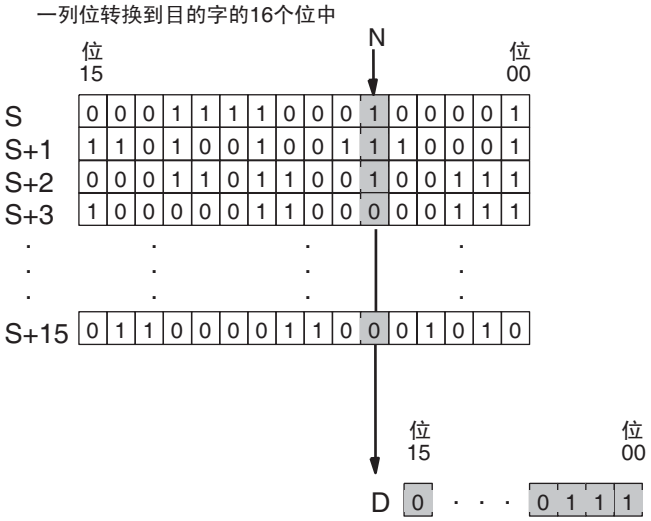
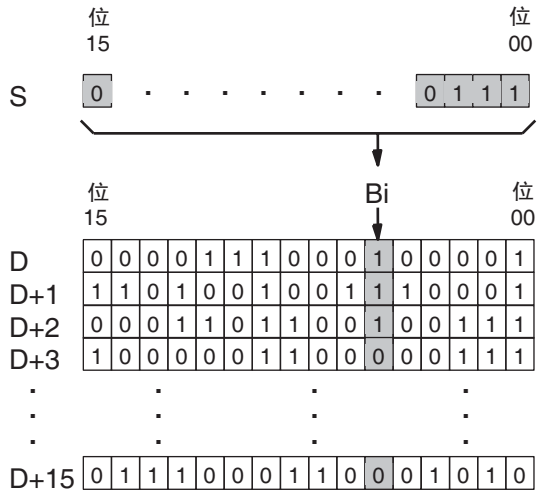
3-10 转换指令

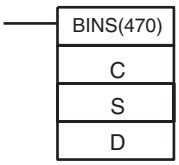
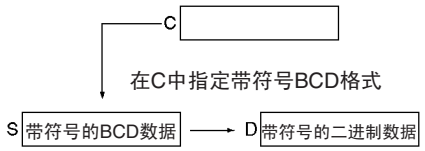
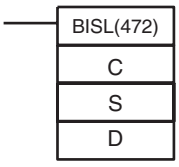
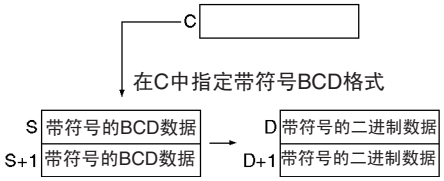
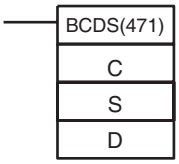
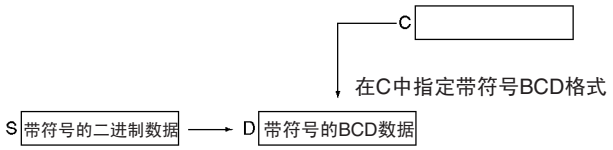
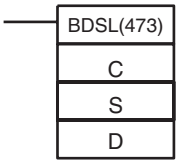
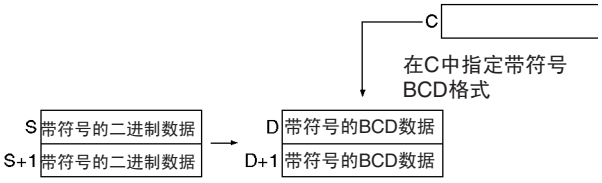
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
BCD 码→二进制 BIN @BIN 023	BIN(023) S R S: 源字 R: 结果字	把BCD码数据转换成二进制数据 S (BCD) → R (BIN)	输出需要
双字长 BCD 码→双字长二进制 BINL @BINL 058	BINL(058) S R S:源首字 R:结果首字	8位数BCD码数据转换成8位十六进制（32位二进制）数据 S (BCD) → R (BIN) S+1 (BCD) R+1 (BIN)	输出需要
二进制→ BCD 码 BCD @BCD 024	BCD(024) S R S:源字 R:结果字	把一个字的二进制数据转换成一个字的BCD码数据 S (BIN) → R (BCD)	输出需要
双字长二进制→双字长 BCD 码 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）转换成8位数BCD码数据 S (BIN) → R (BCD) S+1 (BIN) R+1 (BCD)	输出需要

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
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。 2的补码 (求反+1) $\overline{(S)}$ → (R)	输出需要
双字长二进制求补 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 2的补码 (求反加1) $\overline{(S+1, S)}$ → (R+1, R)	输出需要
带符号 16 位→ 32 位二进制 SIGN @SIGN 600	SIGN(600) S R S:源字 R:结果首字	把一个带符号的16位二进制值扩展为等值的32位。 MSB S MSB = 1: FFFF Hex MSB = 0: 0000 Hex D+1 D D = S的内容	输出需要

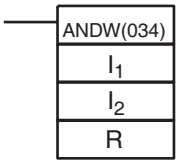
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
数据译码 MLPX @MLPX 076	MLPX(076) S C R S:源字 C:控制字 R:结果首字	读源字中指定数（或字节）的数字值，把结果字（或16字范围）中的相应位变并且把结果字（或16字范围）中的所有其它位变OFF。4→16位转换 	输出需要

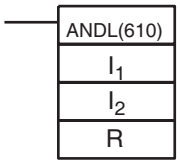
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
数据编码 DMPX @DMPX 077	DMPX(077) S R C S:源首字 R:结果字 C:控制字	在源字（或16字范围）中寻找第一个或最后个ON位的位置，并将该值结果字中指定的数字（或字节） 	输出需要
ASCII 转换 ASC @ASC 086	ASC(086) S Di D S:源字 Di:数字指定器 D:目的首字	把源字中的4位十六进制数转换成相应的8位ASCII码 	输出需要

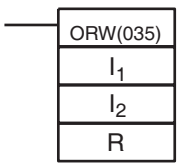
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
ASCII 码→十六进制 HEX @HEX 162	HEX(162) S Di D S: 源首字 Di:数字指定器 D:目标字	将源字中最多4字节的ASCII数据转换成相应的十六进制数， 并将这些位码指定的目的字中 	输出需要
列→行 LINE @LINE 063	LINE(063) S N D S:源首字 N:位号 D:目的字	把16个字范围（16个连续字中的相同位号）中 一列位转换到目的字的16个位中 	输出需要
行→列 COLM @COLM 064	COLM(064) S D N S:源字 D:目的首字 N:位号	把源字中的16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

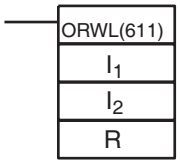
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
带符号的 BCD 码→二进制 BINS @BINS 470	 C:控制字 S:源字 D:目的字	把一个字的带符号BCD码数据转换成一个字的带符号的二进制数据 	输出需要
带符号的双字长 BCD 码→二进制 BISL @BISL 472	 C:控制字 S:源首字 D:目的首字	把双字带符号BCD码数据转换成双字带符号的二进制数据 	输出需要
带符号的二进制→ BCD 码 BCDS @BCDS 471	 C:控制字 S:源字 D:目的字	把一个字的带符号的二进制数据转换成一个字的带符号BCD码数据 	输出需要
带符号的双字长二进制→ BCD 码 BDSL @BDSL 473	 C:控制字 S:源首字 D:目的首字	把双字带符号的二进制数据转换成双字带符号BCD码数据 	输出需要

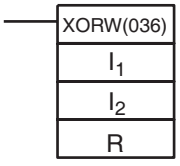
3-11 逻辑指令

<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></tr>
<tr><td>逻辑与
ANDW
@ANDW
034</td><td> 

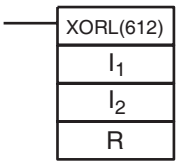
I1: 输入1
I2: 输入2
R: 结果字</td><td>把一个单字数据和单字/或常数的相应位作逻辑与运算
I1. I2 R
<table>
<tr><th>I1</th><th>I2</th><th>R</th></tr>
<tr><td>1</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
</table></td><td>输出需要</td></tr>
<tr><td>双字长逻辑与
ANDL
@ANDL
610</td><td> 

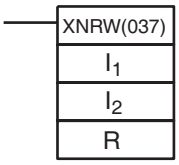
I1:输入1
I2:输入2
R:结果字</td><td>把一个双字数据和双字/或常数的相应位作逻辑与运算
(I1,I1+1). (I2,I2+1) (R, R+1)
<table>
<tr><th>I1,I1+1</th><th>I2,I2+1</th><th>R, R+1</th></tr>
<tr><td>1</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
</table></td><td>输出需要</td></tr>
<tr><td>逻辑或
ORW
@ORW
035</td><td> 

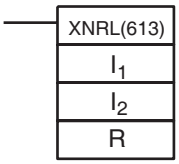
I1:输入1
I2:输入2
R:结果字</td><td>把一个单字数据和单字/或常数的相应位作逻辑或运算
I1 + I2 R
<table>
<tr><th>I1</th><th>I2</th><th>R</th></tr>
<tr><td>1</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>1</td></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
</table></td><td>输出需要</td></tr>
<tr><td>双字长逻辑或
ORWL
@ORWL
611</td><td> 

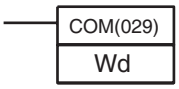
I1: 输入1
I2: 输入2
R: 结果字</td><td>把一个双字数据和双字/或常数的相应位作逻辑或运算
(I1,I1+1) + (I2,I2+1)→ (R, R+1)
<table>
<tr><th>I1,I1+1</th><th>I2,I2+1</th><th>R, R+1</th></tr>
<tr><td>1</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>1</td></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
</table></td><td>输出需要</td></tr>
<tr><td>异或
XORW
@XORW
036</td><td> 

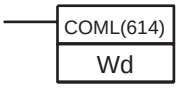
I1: 输入1
I2: 输入2
R: 结果字</td><td>把一个单字数据和单字/或常数的相应位作逻辑异或运算
$I_1.\overline{I_2} + \overline{I_1}.I_2$ R
<table>
<tr><th>I1</th><th>I2</th><th>R</th></tr>
<tr><td>1</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>1</td></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
</table></td><td>输出需要</td></tr>
</table>

<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></tr>
<tr><td>双字长异或
XORL
@XORL
612</td><td> 

I_1: 输入1
I_2: 输入2
R: 结果字</td><td>把一个双字数据和双字/或常数的相应位作逻辑异或运算
$(I_1,I_1+1).\ \overline{(I_2,I_2+1)} + \overline{(I_1,I_1+1)}.\ (I_2,I_2+1)$ (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></tr>
<tr><td>异或非
XNRW
@XNRW
037</td><td> 

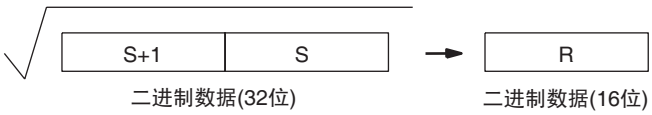
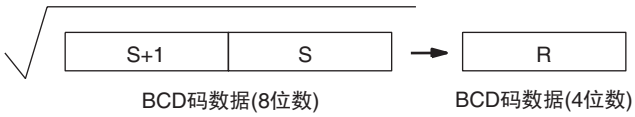
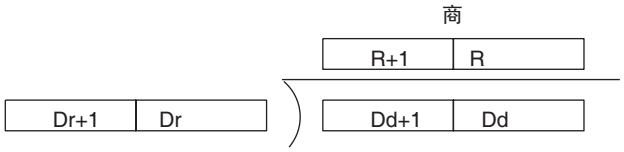
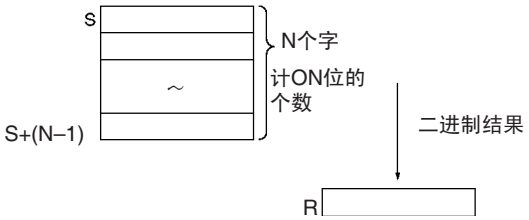
I_1:输入1
I_2:输入2
R:结果字</td><td>把一个单字数据和单字/或常数的相应位作逻辑异或非运算
$I_1.I_2 + \overline{I_1}.\overline{I_2}$ R
<table>
<tr><th>I_1</th><th>I_2</th><th>R</th></tr>
<tr><td>1</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>1</td></tr>
</table></td><td>输出需要</td></tr>
<tr><td>双字长异或非
XNRL
@XNRL
613</td><td> 

I_1:输入1
I_2:输入2
R:结果字</td><td>把一个双字数据和双字/或常数的相应位作逻辑异或非运算
$(I_1,I_1+1).\ (I_2,I_2+1) + \overline{(I_1,I_1+1)}.\ \overline{(I_2,I_2+1)}$ (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></tr>
<tr><td>求反
COM
@COM
029</td><td> 

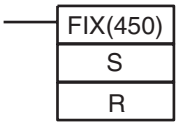
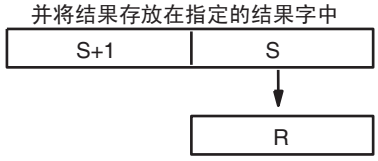
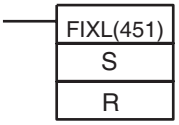
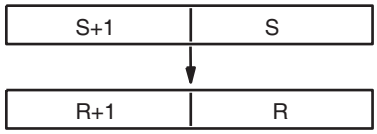
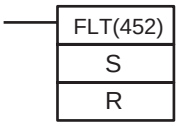
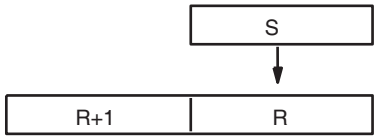
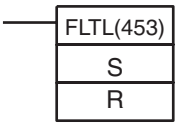
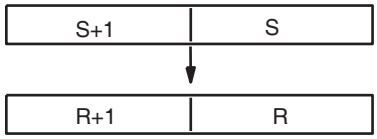
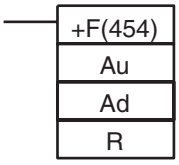
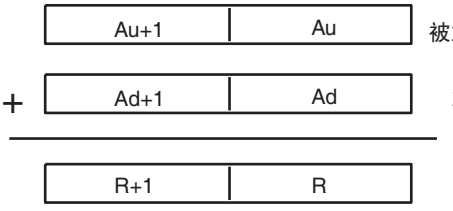
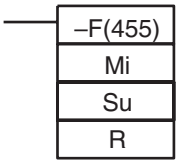
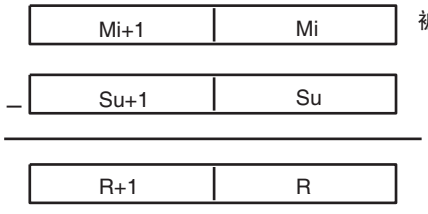
Wd: 字</td><td>把字中所有ON位变OFF ,而把所有OFF位变ON
$\overline{Wd} \rightarrow Wd$: 1 → 0 and 0 → 1</td><td>输出需要</td></tr>
<tr><td>双字长求反
COML
@COML
614</td><td> 

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></tr>
</table>

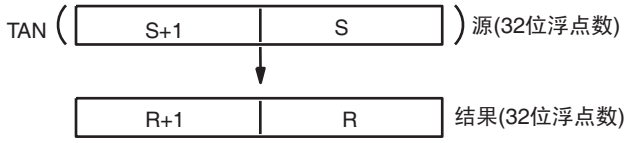
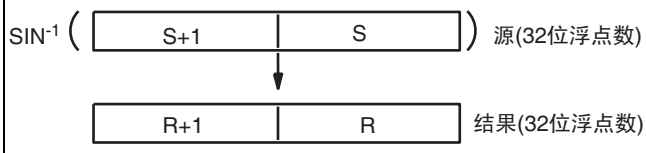
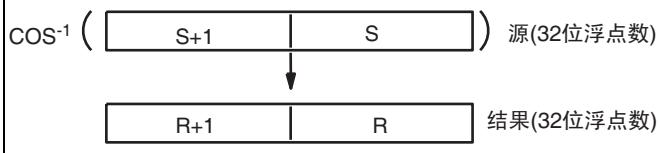
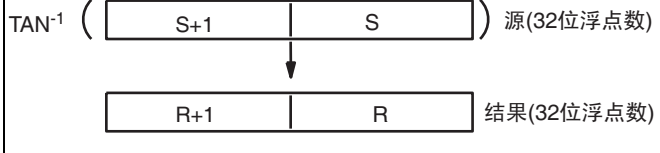
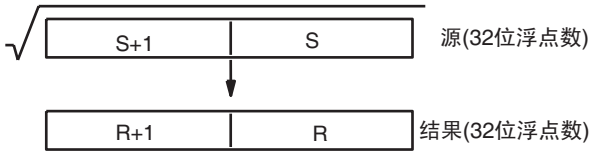
3-12 特殊算术指令

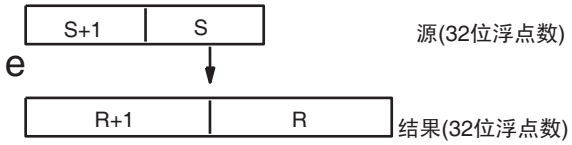
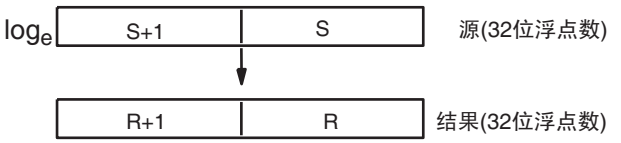
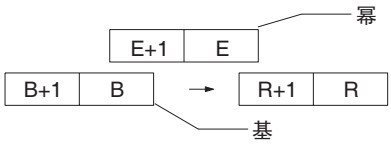
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
二进制平方根 ROTB @ROTB 620	ROTB(620) S R S:源首字 R:结果字	算出指定字的32位二进制数的平方根, 并且把结果的整数部分输出到指定结果字中 	输出需要
BCD 码平方根 ROOT @ROOT 072	ROOT(072) S R S:源首字 R:结果字	算出8位数BCD码的平方根,并把结果的整数部分输出到指定的结果字中 	输出需要
数学处理 APR @APR 069	APR(069) C S R C:控制字 S:源数据 R:结果字	计算源数据的正弦、余弦或线性逼近。 线性逼近功能允许 X 和 Y 间的任何关系用线段来模拟。	输出需要
浮点数除法 FDIV @FDIV 079	FDIV(079) Dd Dr R Dd:被除数首字 Dr: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
位计数器 BCNT @BCNT 067	BCNT(067) N S R N:字数 S:源首字 R:结果字	在指定的字中计所有ON位个数 	输出需要

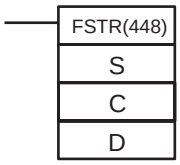
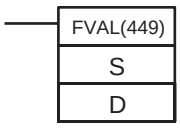
3-13 浮点数运算指令

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
浮点数→ 16 位 FIX @FIX 450	 S:源首字 R:结果字	把一个32位浮点数转换成带符号的16位二进制数据， 并将结果存放在指定的结果字中  浮点数据(32位) 带符号二进制数据(16位)	输出需要
浮点数→ 32 位 FIXL @FIXL 451	 S:源首字 R:结果首字	把一个32位浮点数转换成带符号的32位二进制数据， 并将结果存放在指定的结果字中  浮点数数据(32位) 带符号二进制数据(32位)	输出需要
16 位→浮点数 FLT @FLT 452	 S:源字 R:结果首字	把一个带符号的16位二进制数据转换成32位浮点数据， 并将结果存放在指定的结果字中  带符号二进制数据(16位) 浮点数数据(32位)	输出需要
32 位→浮点数 FLTL @FLTL 453	 S:源首字 R:结果首字	把一个带符号的32位二进制数转换成32位浮点数据， 并将结果存放在指定的结果字中  带符号二进制数据(32位) 浮点数数据(32位)	输出需要
浮点数加法 +F @+F 454	 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，并将结果存放在指定的结果字中  被加数(浮点数，32位) 加数(浮点数，32位) 结果(浮点数，32位)	输出需要
浮点数减法 –F @–F 455	 Mi :被减数首字 Su:减数首字 R:结果首字	把一个32位浮点数从另一个32位浮点数中减去， 并将结果存放在指定的结果字中  被减数(浮点数，32位) 减数(浮点数，32位) 结果(浮点数，32位)	输出需要

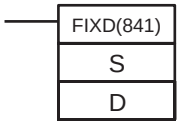
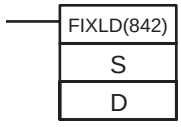
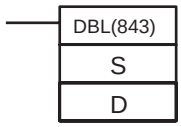
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
浮点数乘法 *F @*F 456	*F(456) Md Mr R Md：被乘首字 Mr：乘首字 R：结果首字	把两个32位浮点数相乘，并将结果存放在指定的结果字中 Md+1 | Md 被乘数(浮点数，32位) × Mr+1 | Mr 乘数(浮点数，32位) R+1 | R 结果(浮点数，32位)	输出需要
浮点数除法 /F @/F 457	/F(457) Dd Dr R Dd:被除首字 Dr: 除首字 R:结果首字	把一个32位浮点数去除另一个32位浮点数， 并将结果存放在指定的结果字中 Dd+1 | Dd 被除数(浮点数，32位) ÷ Dr+1 | Dr 除数(浮点数，32位) R+1 | R 结果(浮点数，32位)	输出需要
度→弧度 RAD @RAD 458	RAD(458) S R S:源首字 R:结果首字	把一个32位浮点数从度转换成弧度，并将结果存放在指定的结果字中 S+1 | S 源(度，32位浮点数) ↓ R+1 | R 结果(弧度，32位浮点数)	输出需要
弧度→度 DEG @DEG 459	DEG(459) S R S:源首字 R:结果首字	把一个32位浮点数从弧度转换成度，并将结果存放在指定的结果字中 S+1 | S 源(弧度，32位浮点数) ↓ R+1 | R 结果(度，32位浮点数)	输出需要
正弦 SIN @SIN 460	SIN(460) 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)的正弦， 并将结果存放在指定的结果字中 SIN (S+1 | S) 源(32位浮点数) ↓ R+1 | R 结果(32位浮点数)	输出需要
余弦 COS @COS 461	COS(461) 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)的余弦， 并将结果存放在指定的结果字中 COS (S+1 | S) 源(32位浮点数) ↓ R+1 | R 结果(32位浮点数)	输出需要

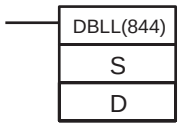
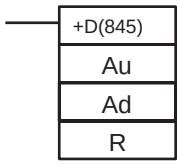
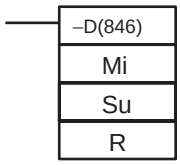
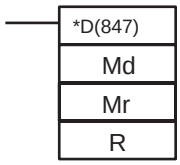
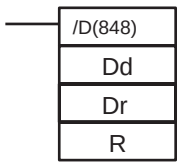
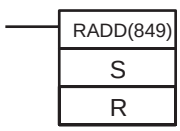
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
正切 TAN @TAN 462	TAN(462) S R S:源首字 R:结果首字	求一个32位浮点数(用弧度表示)的正切， 并将结果存放在指定的结果字中 	输出需要
反正弦 ASIN @ASIN 463	ASIN(463) S R S:源首字 R:结果首字	求一个32位浮点数的反正弦，并将结果存放在指定的结果字中。 （反正弦函数与正弦函数正好相反，它复原一个角度值，该角度的正弦值在－1和1之间） 	输出需要
反余弦 ACOS @ACOS 464	ACOS(464) S R S:源首字 R:结果首字	求一个32位浮点数的反余弦，并将结果存放在指定的结果字中。 （反余弦函数与余弦函数正好相反，它复原一个角度值，该角度的余弦值在－1和1之间） 	输出需要
反正切 ATAN @ATAN 465	ATAN(465) S R S:源首字 R:结果首字	求一个32位浮点数的反正切，并将结果存放在指定的结果字中。 （反正切函数与正切函数正好相反，它复原一个角度值，该角度产生一个正切值） 	输出需要
平方根 SQRT @SQRT 466	SQRT(466) S R S:源首字 R:结果首字	求一个32位浮点数的平方根，并将结果存放在指定的结果字中。 	输出需要

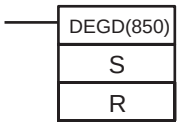
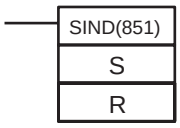
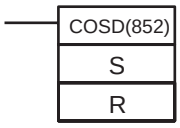
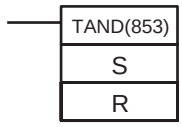
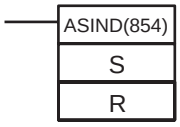
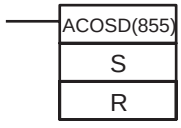
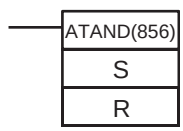
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
指数 EXP @EXP 467	EXP(467) S R S:源首字 R:结果首字	求一个32位浮点数的自然指数（底为e）， 并将结果存放在指定的结果字中。 	输出需要
对数 LOG @LOG 468	LOG(468) S R S:源首字 R:结果首字	求一个32位浮点数的自然对数（底为e）， 并将结果存放在指定的结果字中。 	输出需要
指数幂 PWR @PWR 840	PWR(840) B E R B:基首字 E:指数首字 R:结果首字	自乘一个32位浮点数到另一个32位浮点数的幂 	输出需要
浮点符号比较 （仅适用于 CS1-H, CJ1-H, CJ1M 或 CS1D） LD， AND 或 OR + =F (329), <>F (330), <F (331), <=F (332), >F (333), or >=F (334)	使用LD: 符号，选择 S1 S2 使用AND: 符号，选择 S1 S2 使用OR: 符号，选择 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，如果比较结果为真，生成一个 ON 执行条件。 浮点符号比较指令可用三种符号：LD (Load)， AND 和 OR。	LD：不需要 AND 或 OR：需要

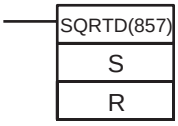
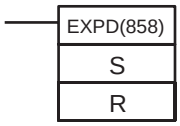
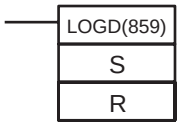
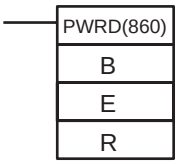
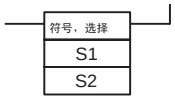
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
浮点数→ ASCII （仅适用于 CS1-H, CJ1-H, CJ1M 或 CS1D） FSTR @FSTR 448	 S:源首字 C:控制字 D:目的字	把指定的单精度浮点数数据 （32 位小数点或指数格式）转换成字符串数据 (ASCII)，并把结果存入目的字中。	输出需要
ASCII →浮点数 （仅适用于 CS1-H, CJ1-H, CJ1M 或 CS1D） FVAL @FVAL 449	 S:源字 D:目的首字	把指定的单精度浮点数数据字符串 (ASCII) 表达成 （小数点或指数格式）转换成单精度 32 位浮点数数据，并把结果存入目的字中。	输出需要

3-14 双精度浮点数指令 （仅适用于 CS1-H, CJ1-H, CJ1M 或 CS1D 单元）

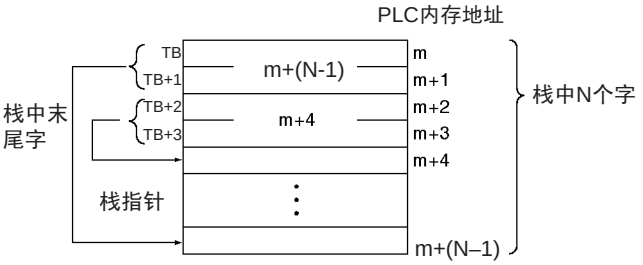
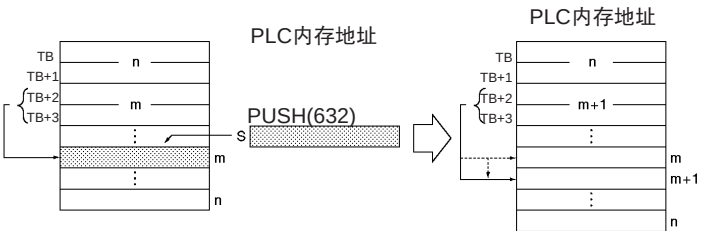
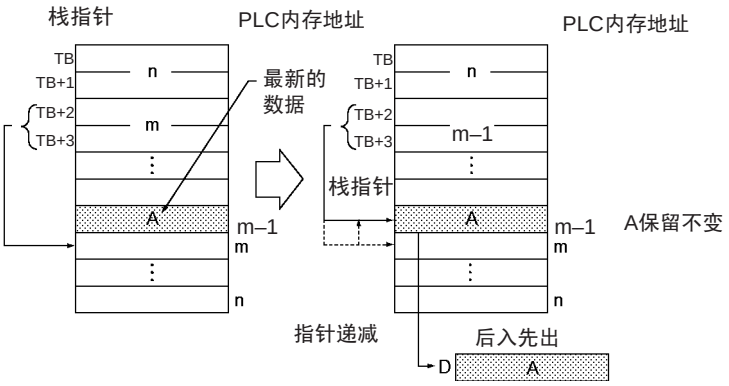
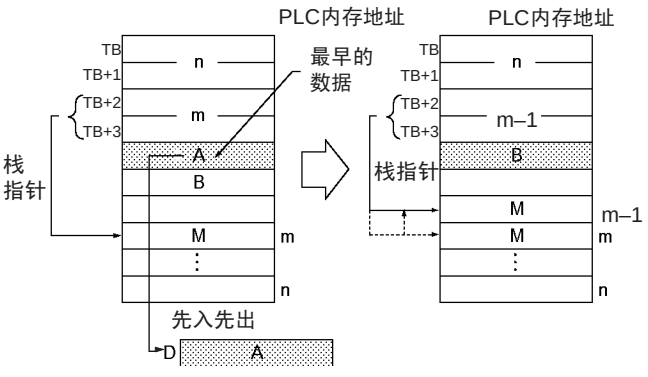
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
双字长浮点数→ 16 位二进制 FIXD @FIXD 841	 S:源首字 D:目的字	把指定的双精度浮点数数据 （64 位）转换成 16 位带符号二进制数据，并把结果存入到目的字中。	输出需要
双字长浮点数→ 32 位二进制 FIXLD @FIXLD 842	 S:源首字 D:目的首字	把指定的双精度浮点数数据 （64 位）转换成 32 位带符号二进制数据，并把结果存入到目的字中。	输出需要
16 位二进制→双字长浮点数 DBL @DBL 843	 S:源字 D:目的首字	把指定的 16 位带符号二进制数据转换成双精度浮点数数据 （64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

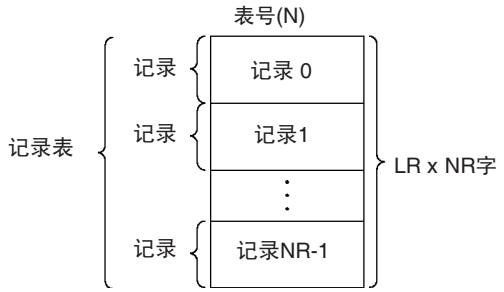
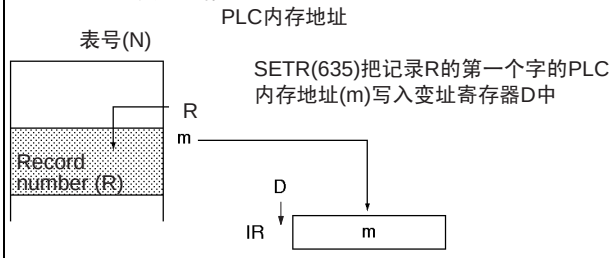
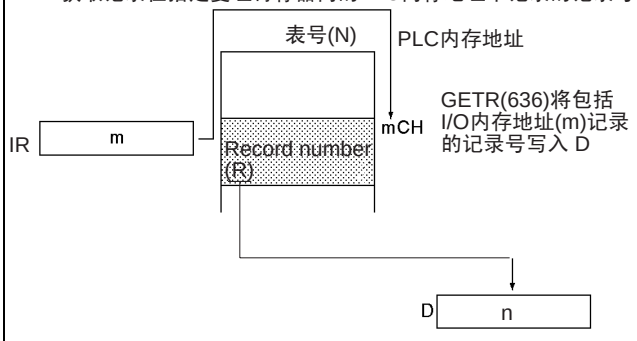
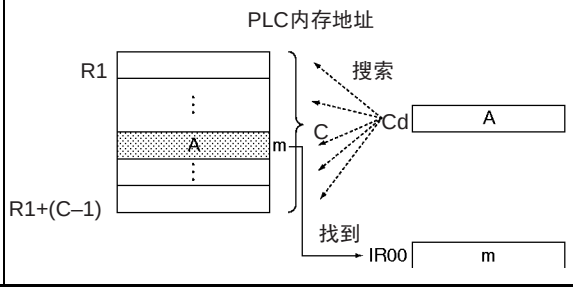
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
32 位二进制→双字长浮点数 DBLL @DBLL 844	 S:源首字 D:目的首字	把指定的 32 位带符号二进制数据转换成双精度浮点数数据 （64 位），并把结果存入到目的字中。	输出需要
双字长浮点数加法 +D @+D 845	 Au:被加数首字 Ad:加数首字 R:结果首字	把两个指定的双精度浮点数 （64 位）相加，并把结果存入到目的字中。	输出需要
双字长浮点数减法 –D @–D 846	 Mi:被减数首字 Su:减数首字 R: 结果首字	把两个指定的双精度浮点数 （64 位）相减，并把结果存入到目的字中。	输出需要
双字长浮点数乘法 *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 （64 位）相乘，并把结果存入到目的字中。	输出需要
双字长浮点数除法 /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 （64 位）相除，并把结果存入到目的字中。	输出需要
双字长度→弧度 RADD @RADD 849	 S: 源首字 R: 结果首字	把指定的双精度浮点数数据 （64 位）从度转换成弧度，并把结果存入到目的字中。	输出需要

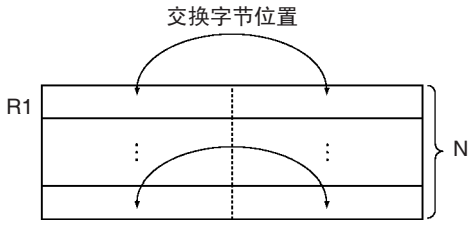
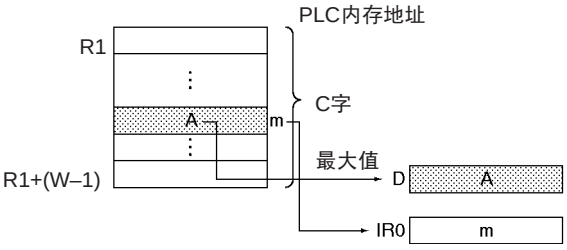
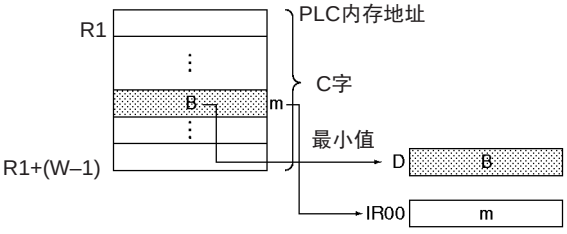
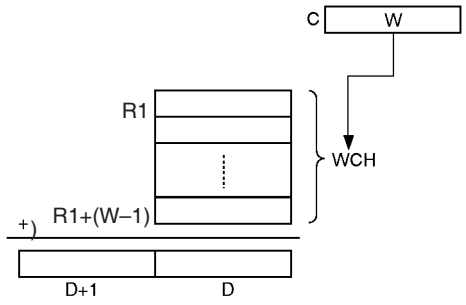
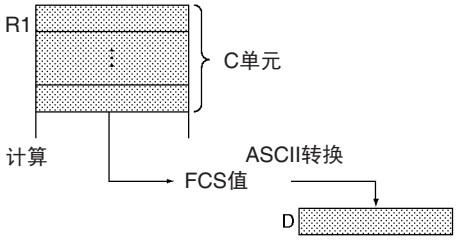
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
双字长弧度→度 DEGD @DEGD 850	 S:源首字 R:结果首字	把指定的双精度浮点数数据 （64 位）从弧度转换成度，并把结果存入到目的字中。	输出需要
双字长正弦 SIND @SIND 851	 S:源首字 R:结果首字	求出一个指定的双精度浮点数数据 （64 位用弧度表示）的正弦，并把结果输出到结果字中。	输出需要
双字长余弦 COSD @COSD 852	 S:源首字 R:结果首字	求出一个指定的双精度浮点数数据 （64 位用弧度表示）的余弦，并把结果输出到结果字中。	输出需要
双字长正切 TAND @TAND 853	 S:源首字 R:结果首字	求出一个指定的双精度浮点数数据 （64 位用弧度表示）的正切，并把结果输出到结果字中。	输出需要
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浮点数数据 （64 位用弧度表示）的反正弦，并把结果输出到结果字中。（反正弦函数和正弦函数正好相反，它复原一个角度值，该角度的正弦值在 –1 和 1 之间）	输出需要
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浮点数数据 （64 位用弧度表示）的反余弦，并把结果输出到结果字中。（反余弦函数和余弦函数正好相反，它复原一个角度值，该角度的余弦值在 –1 和 1 之间）	输出需要
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浮点数数据 （64 位用弧度表示）的反正切，并把结果输出到结果字中。（反正切函数和正切函数正好相反，它复原一个角度值，该角度产生一个正切值）	输出需要

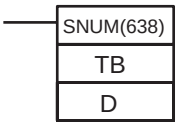
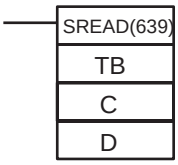
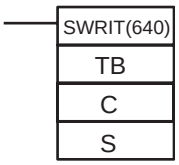
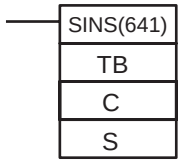
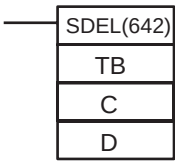
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
双字长平方根 SQRTD @SQRTD 857	 S:源首字 R:结果首字	求一个指定的双精度浮点数 （64 位）的平方根，并将结果输出到结果字中。	输出需要
双字长指数 EXPD @EXPD 858	 S:源首字 R:结果首字	求一个指定的双精度浮点数 （64 位）的自然指数 （底为 e），并将结果输出到结果字中。	输出需要
双字长对数 LOGD @LOGD 859	 S:源首字 R:结果首字	求一个指定的双精度浮点数 （64 位）的自然对数 （底为 e），并将结果输出到结果字中。	输出需要
双字长指数幂 PWRD @PWRD 860	 B:基首字 E: 指数首字 R: 结果首字	自乘一个双精度浮点数 （64 位）到另一双精度浮点数的幂，并将结果输出到结果字中。	输出需要
双字长符号比较 LD， AND 或 OR + =D (335), <>D (336), <D (337), <=D (338), >D (339), or >=D (340)	使用LD:  使用AND:  使用OR:  S1: 比较数据1 S2: 比较数据2	比较指定的双精度数据 （64 位），如果比较结果为真，生成一个 ON 执行条件。 浮点数比较指令可用三种符号：LD (Load)， AND 和 OR。	LD：不需要 AND 或 OR： 需要

3-15 表格数据处理指令

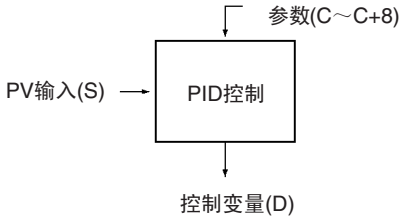
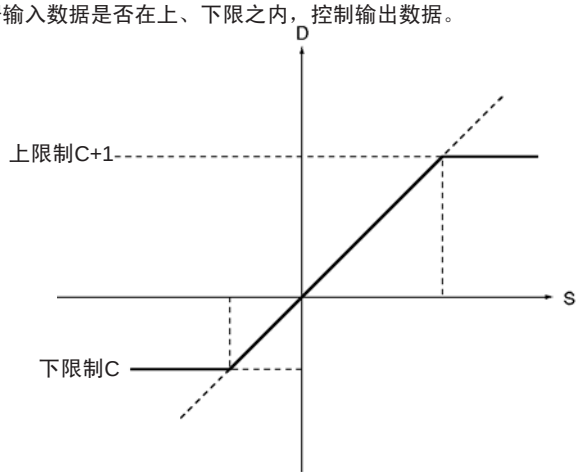
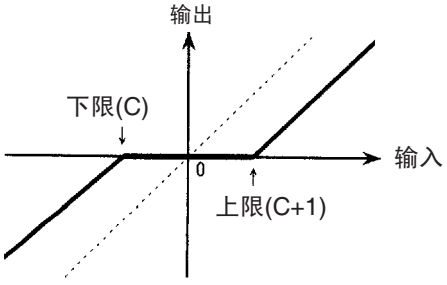
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
堆栈设置 SSET @SSET 630	SSET(630) TB N TB:栈首址 N:字数	在指定字开始定义一个指定长度的栈，并初始化此数据区的字全为零 	输出需要
压入栈 PUSH @PUSH 632	PUSH(632) TB S TB:栈首址 S:源字	把一个数据的字写入指定栈 	输出需要
后入先出 LIFO @LIFO 634	LIFO(634) TB D TB:栈首址 D: 目的字	读最后写入指定栈中的数据字（栈中最新的字） 	输出需要
先入先出 FIFO @FIFO 633	FIFO(633) TB D TB:栈首址 D: 目的字	读第一个写入指定栈中的数据字（栈中最早的数据） 	输出需要

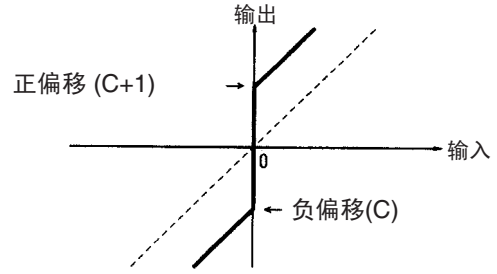
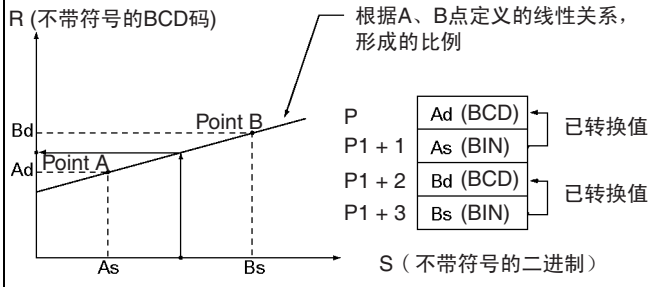
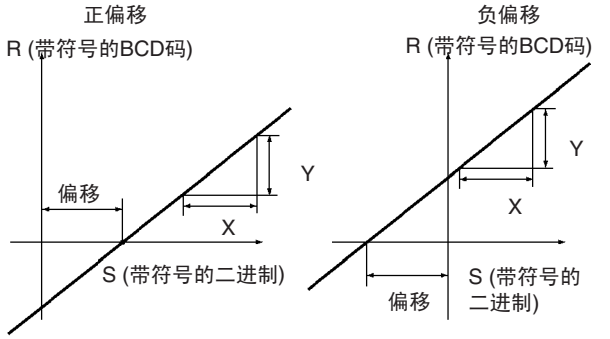
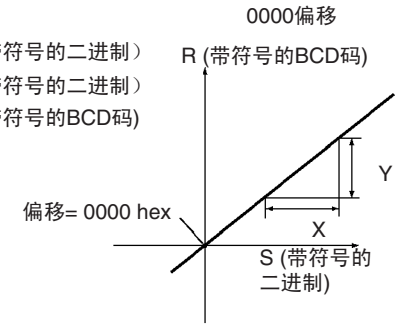
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
定义记录表 DIM @DIM 631	DIM(631) N LR NR TB N:表号 LR:每个记录长度 NR:记录数 TB:表首字	用说明每个记录长度及记录数来定义记录表，最多可定义16个记录表 	输出需要
设置记录位置 SETR @SETR 635	SETR(635) N R D N:表号 R:记录号 D:目的变址 寄存器	把指定记录的位置（开始记录的PLC内存地址） 写入指定的变址寄存器中 	输出需要
获得记录号 GETR @GETR 636	GETR(636) N IR D N:表号 IR:变址寄存器 D:目的字	获取记录在指定变址寄存器内的PLC内存地址中记录的记录号 	输出需要
数据搜索 SRCH @SRCH 181	SRCH(181) C R1 Cd C:控制首字 R1:范围内首字 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

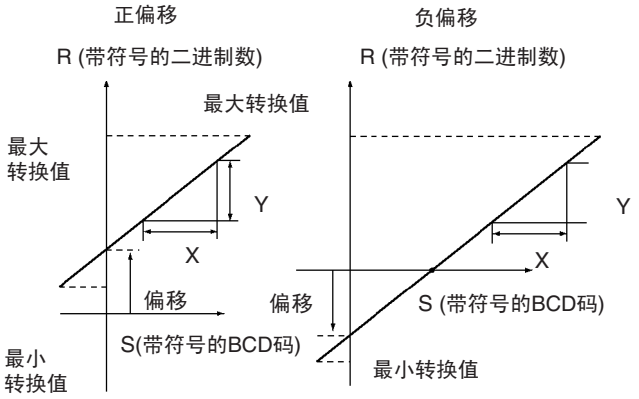
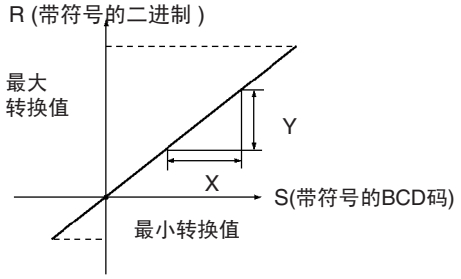
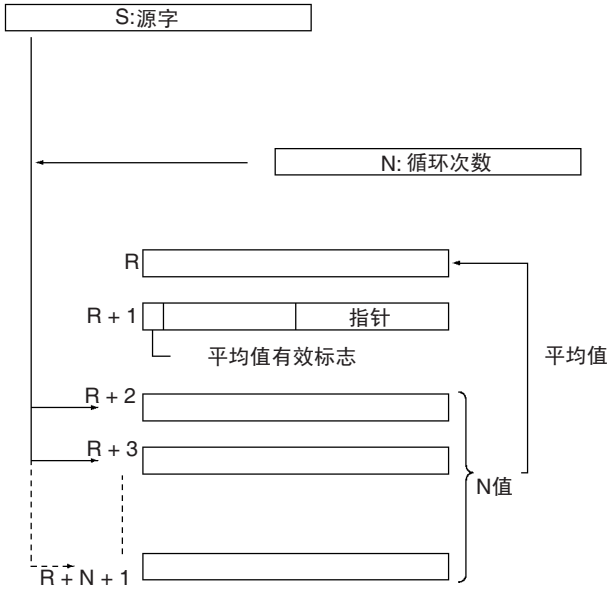
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
交换字节 SWAP @SWAP 637	SWAP(637) N R1 N:字数 R1:范围首字	将范围内的所有字的左字节和右字节交换 	输出需要
寻找最大值 MAX @MAX 182	MAX(182) C R1 D C:控制首字 R1:范围首字 D:目的字	在范围中找出最大值 	输出需要
寻找最小值 MIN @MIN 183	MIN(183) C R1 D C: 控制首字 R1:范围首字 D :目的字	在范围中找出最小值 	输出需要
求和 SUM @SUM 184	SUM(184) C R1 D C: 控制首字 R1:范围首字 D :目的字	将范围中的字节或字相加，并把结果输出到两个字中 	输出需要
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

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
读栈大小 （仅适用于 CS1-H，CJ1-H，CJ1M 或 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
读栈数据 （仅适用于 CS1-H，CJ1-H，CJ1M 或 CS1D） SREAD @SREAD 639	 TB:栈首址 C:偏移值 D:目的字	从栈中指定数据元素读数据，偏移值指明了所要数据元素的位置（在当前指针位置前多少个数据元素）。	输出需要
覆盖栈数据 （仅适用于 CS1-H，CJ1-H，CJ1M 或 CS1D） SWRIT @SWRIT 640	 TB:栈首址 C:偏移值 S:源数据	把源数据写入栈中指定数据元素中（覆盖原来数据）。偏移值指明了所要数据元素的位置（在当前指针位置前多少个数据元素）。	输出需要
插入栈数据 （仅适用于 CS1-H，CJ1-H，CJ1M 或 CS1D） SINS @SINS 641	 TB:栈首址 C:偏移值 S:源数据	把源数据插入栈中指定数据元素中，并将栈中其余的数据向下移。偏移值指明了插入点的位置（在当前指针位置前多少个数据元素）。	输出需要
删除栈数据 （仅适用于 CS1-H，CJ1-H，CJ1M 或 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，并将栈中其余的数据向上移。偏移值指明了删除点的位置（在当前指针位置前多少个数据元素）。	输出需要

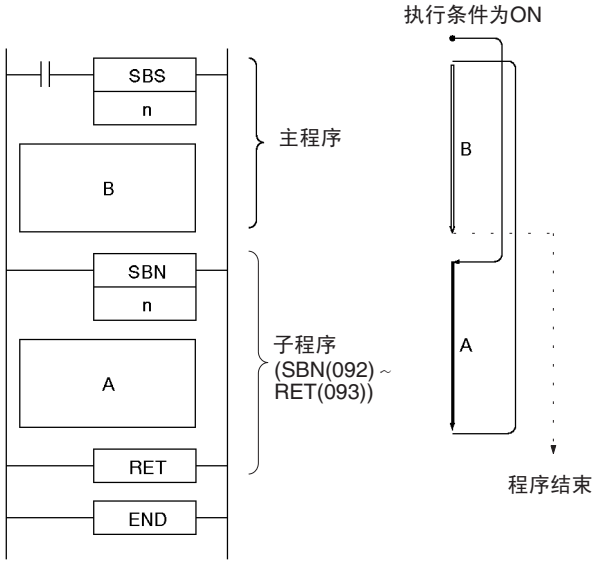
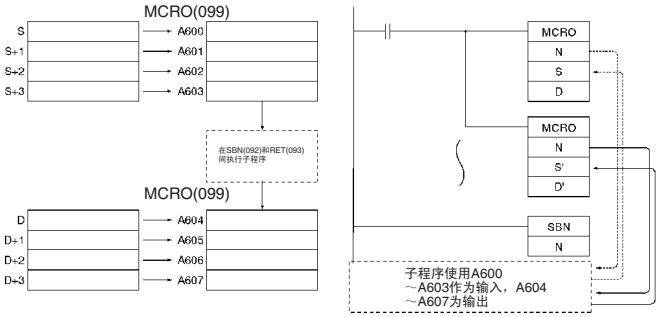
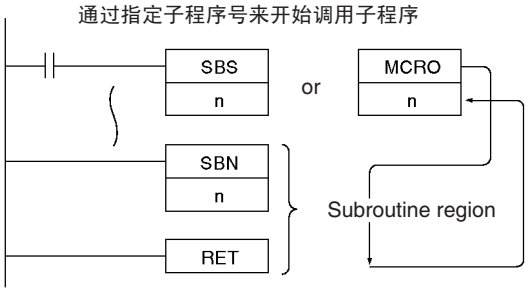
3-16 数据控制指令

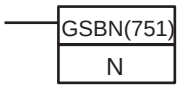
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
PID 控制 PID 190	PID(190) S C D S:输入字 C:参数首字 D:输出字	根据指定的参数进行PID控制 	输出需要
带自调整 PID 控制（仅适用于 CS1-H，CJ1-H，CJ1M 或 CS1D） PIDAT 191	PIDAT(191) S C D S:输入字 C:参数首字 D:输出字	根据指定参数进行 PID 控制。PID 常数可用 PIDAT(191) 指令自动调整。	输出需要
限制控制 LMT @LMT 680	LMT(680) S C D S:输入字 C:限制首字 D:输出字	根据输入数据是否在上、下限之内，控制输出数据。 	输出需要
静区控制 BAND @BAND 681	BAND(681) S C D S:输入字 C:限制首字 D:输出字	根据输入数据是否在静区范围内，控制输出数据 	输出需要

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
静域控制 ZONE @ZONE 682	ZONE(682) S C D S:输入字 C:限制首字 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
标度 SCL @SCL 194	SCL(194) S P1 R S:源字 P1:参数首字 R: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码数据 	输出需要
标度 2 SCL2 @SCL2 486	SCL2(486) S P1 R S:源字 P1:参数首字 R:结果字	根据指定的线性函数，把带符号的二进制数据转换成带符号的BCD码数据。偏移值可在定义线性函数时输入。  P1 偏移 (带符号的二进制） P1 + 1 Y (带符号的二进制） P1 + 2 X (带符号的BCD码) 	输出需要

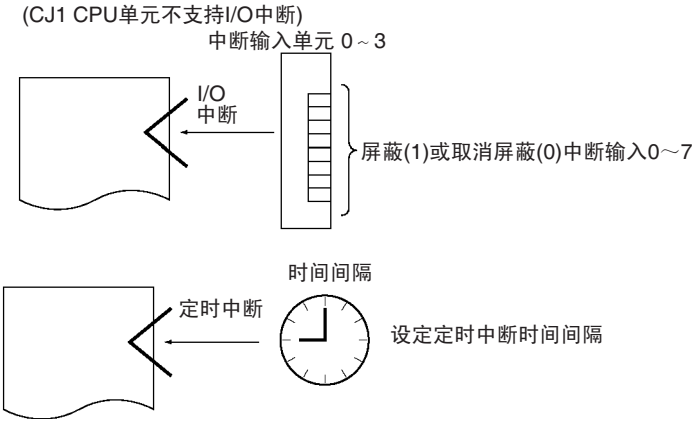
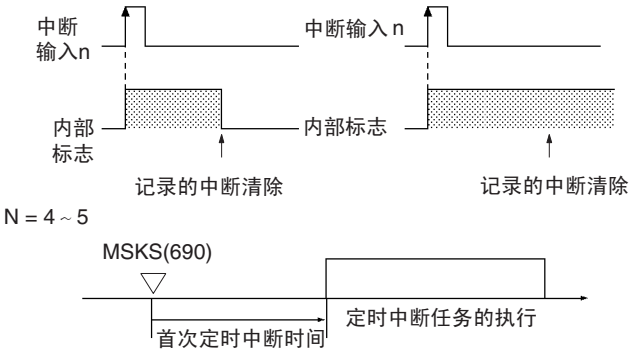
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
标度 3 SCL3 @SCL3 487	SCL3(487) S P1 R S:源字 P1:参数首字 R:结果字	根据指定的线性函数，把带符号的BCD码数据转换成带符号的二进制数据。偏移值可在定义线性函数时输入。  **0000**偏移 	输出需要
平均值 AVG 195	AVG(195) S N R S:源字 N:循环数 R:结果字	求出对一个输入字指定循环次数的平均值 	输出需要

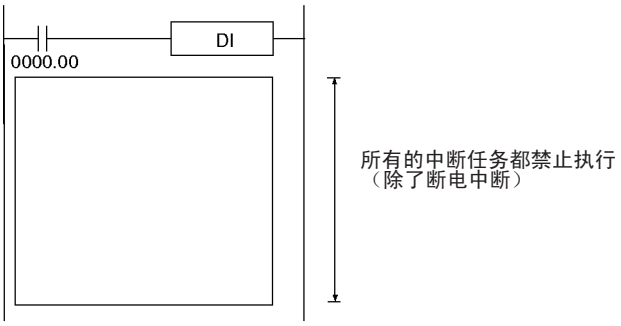
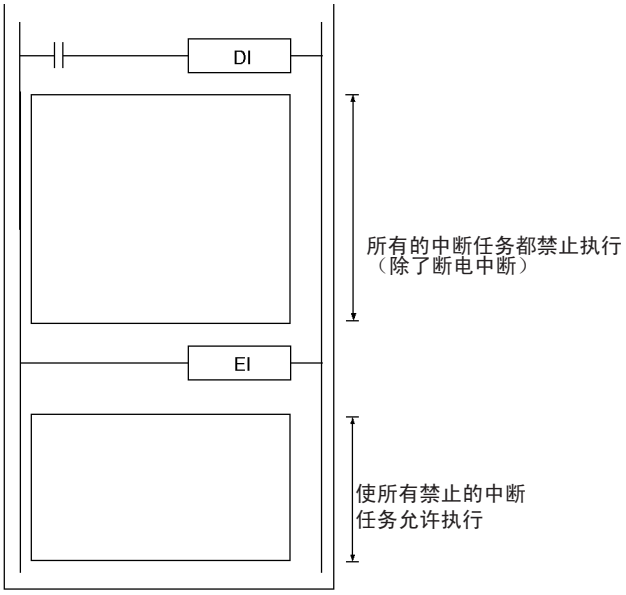
3-17 子程序指令

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
子程序调用 SBS @SBS 091	SBS(091) N N:子程序编号	调用指定子程序编号的子程序并执行它 	输出需要
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	调用指定子程序编号的子程序， 并用S～S+3中的输入参数执行子程序，并输出到D～D+3 	输出需要
子程序入口 SBN 092	SBN(092) N N:子程序编号	通过指定子程序号来开始调用子程序 	输出不需要
子程序返回 RET 093	RET(093)	表示子程序结束。	输出不需要

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

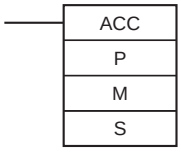
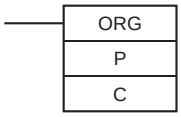
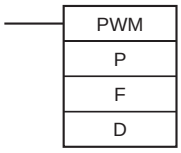
3-18 中断控制指令

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
设置中断屏蔽 （CS1D 不支持） MSKS @MSKS 690	MSKS(690) N S N:中断标识符 S:中断数据	对I/O中断或定时中断设定中断处理。当PC刚通电时，I/O中断和定时中断被屏蔽（禁止）。MSKS(690)可用于取消屏蔽或屏蔽中断以及设定定时中断的时间间隔。 (CJ1 CPU单元不支持I/O中断) 	输出需要
读中断屏蔽 （CS1D 不支持） MSKR @MSKR 692	MSKR(692) N D N: 中断标识符 D: 目的字	读出 MSKS(690) 设定的 当前中断处理设定。	输出需要
清除中断 （CS1D 不支持） CLI @CLI 691	CLI(691) N S N:中断标识符 S:中断数据	对I/O中断清除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时间。N = 0～3 (CJ1 CPU单元不支持I/O单元) 	输出需要

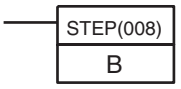
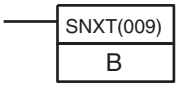
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
禁止中断 DI @DI 693	DI(693)	除了断电中断，其余所有中断任务都禁止执行 	输出不需要
允许中断 EI 694	EI(694)	允许执行所有由DI(693)禁止的中断任务 	输出不需要

3-19 高速计数器和脉冲输出指令 （仅适用于 CJ1M-CPU22/23）

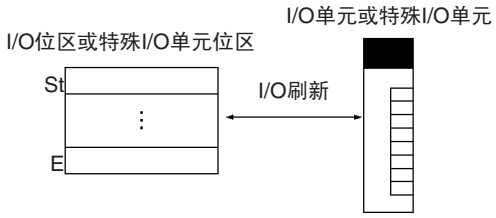
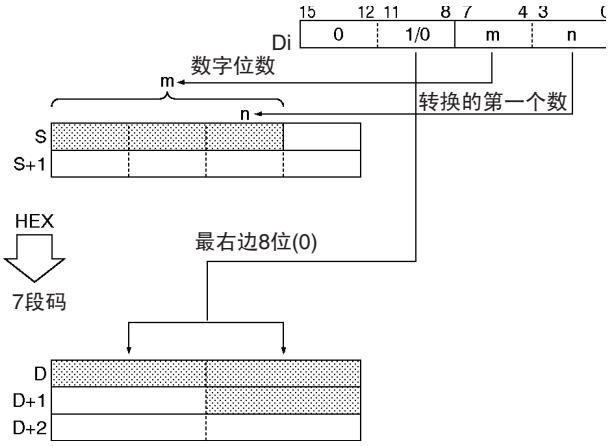
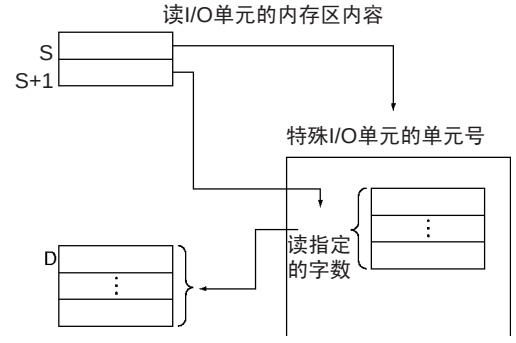
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
控制模式 INI @INI 880	INI P C NV P: 口定义 C: 控制字 NV: 最新 PV 值首字	INI(880) 指令用于启动和停止目标值比较，改变一个高速计数器的当前值 (PV)，改变中断输入 （计数器模式）的当前值 （PV），改变一个脉冲输出的当前值 （PV）或停止脉冲输出。	输出需要
读高速计数器 PV 值 PRV @PRV 881	PRV P C D P: 口定义 C: 控制字 D: 目的首字	PRV(881) 指令用于读取一个高速计数器的当前值 (PV) 和脉冲输出或中断输入 （计数器模式）。	输出需要
比较表写入 CTBL @CTBL 882	CTBL P C TB P: 口定义 C: 控制字 TB: 比较表首字	CTBL(882) 指令用于完成对一个高速计数器的当前值 (PV) 进行目标值或范围比较。	输出需要
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	SPED(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
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
脉冲输出 PLS2 @PLS2 887	PLS2 P M S F P: 口定义 M: 输出模式 S: 设定表的首字 F: 启动频率首字	PLS2(887)指令用于设定脉冲频率和加速度/减速度率，并完成带加速度/减速度（不同加 / 减速度率）的脉冲输出。但仅适用于定位控制。	输出需要

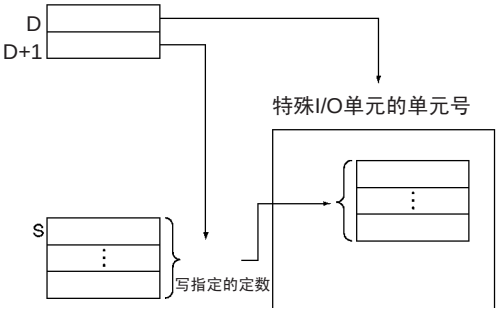
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
加速度控制 ACC @ACC 888	 P: 口定义 M: 输出模式 S: 设定表首字	ACC(888) 指令用于设定脉冲频率和加速度 / 减速度率，并完成带加速度 / 减速度（相同加 / 减速度率）的脉冲输出。可用于定位控制和速度控制。	输出需要
原点搜索 ORG @ORG 889	 P: 口定义 C: 控制数据	ORG(889) 用于完成原点搜索和返回。	输出需要
带占空比可变的脉冲 PWM @ 891	 P: 口定义 F: 频率 D: 占空比	PWM(891) 用于输出占空比可变的脉冲。	输出需要

3-20 步指令

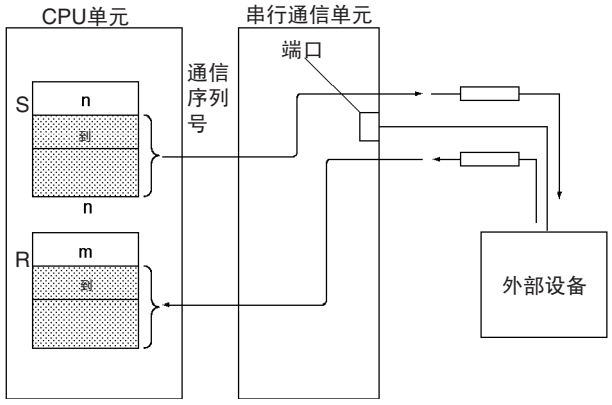
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
步定义 STEP 008	 B:位	STEP(008) 按下列 2 种方式作用，它取决于它的位置和控制位是否被指定。 (1) 启动一个指定的步。 (2) 结束该步的程序区 （例：步执行）	输出需要
步启动 SNXT 009	 B:位	SNXT(009) 用于下列三种方式： (1) 启动步程序执行。 (2) 继续到下一步的控制位。 (3) 结束步程序执行。	输出需要

3-21 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
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
七段译码 SDEC @SDEC 078	SDEC(078) S Di D S:源字 Di:数字指定器 D:目的首字	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，并把它存入指定目的字中的高或低8位。 	输出需要
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:传送目的和字数		输出需要

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
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:传送目的和字数	把CPU单元的 I/O内存区的内容输出到特殊I/O单元 	输出需要
CPU 总线单元 I/O 刷新（仅适用于 CS1-H，CJ1-H，CJ1M 或 CS1D） DLNK @DLNK 226	DLNK(226) N N:单元号	立即刷新指定单元号的 CPU 总线单元的 I/O。	输出需要

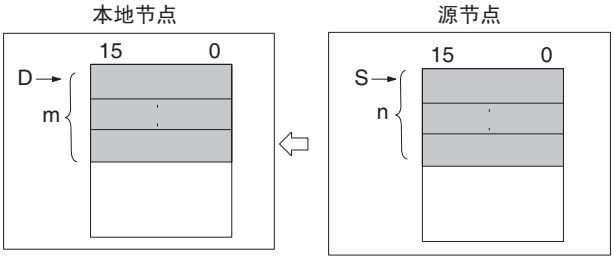
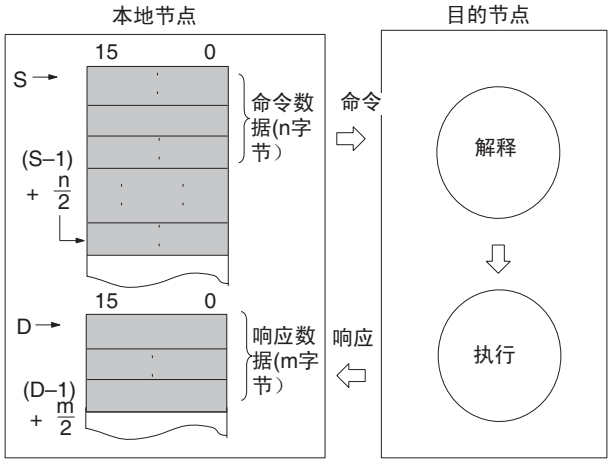
3-22 串行通信指令

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
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(CS系列) 或串行通信单元中登记过的通信序列 	输出需要
发送 TXD @TXD 236	TXD(236) S C N S:源首址 C:控制字 N:字节数 0000～0100 十六进制数 (0～256十进制字数)	从 CPU 单元内置的 RS-232C 端号，发送指定字节数的数据。	输出需要

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
接收 RXD @RXD 235	RXD(235) D C N D:目的首字 C:控制字 N:存储的字节数 0000～0100 十六进制 (0～256十进制)	从 CPU 单元内置的 RS-232C 端口读取指定字节数的数据。	输出需要
修改串行口设置 STUP @STUP 237	STUP(237) C S C:控制字（口） S:源	修改 CPU 单元，串行通信单元 （CPU 总线单元），或串行通信板 （仅为 CS 系列）上的串行通信口的通信参数。在 PLC 运行时， STUP(237) 能使协议方式改变。	输出需要

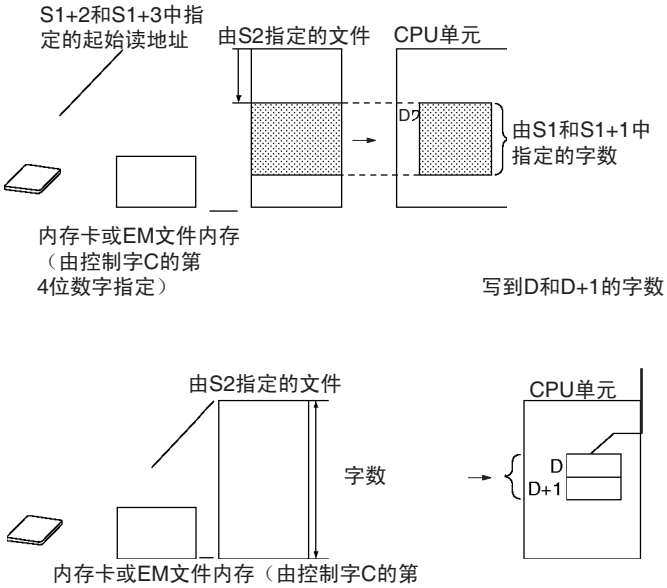
3-23 网络指令

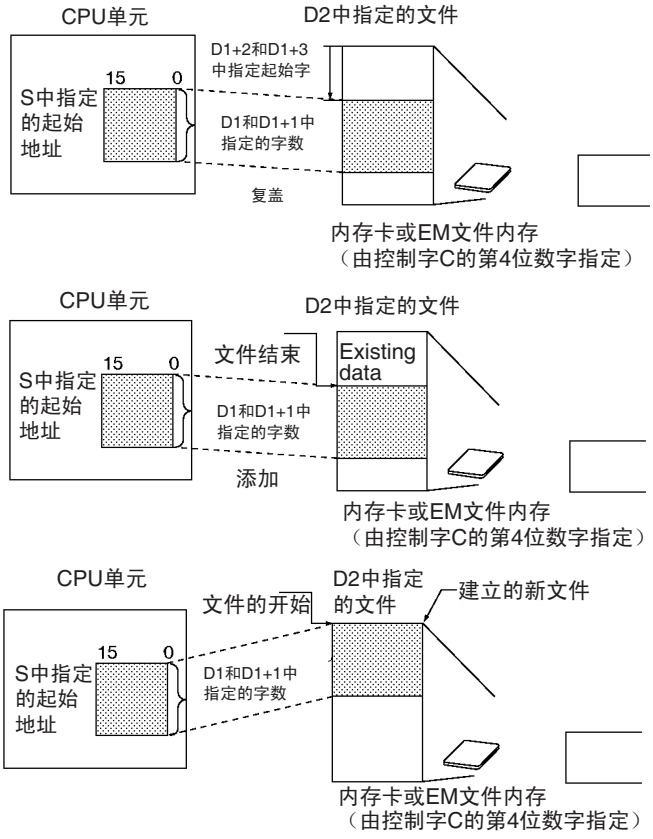
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
网络发送 SEND @SEND 090	SEND(090) S D C S:源首字 D:目的首字 C:控制首字	把数据传输到网络中的一个节点 本地节点 目的节点 15 0 S n:发送的字数 15 0 D n	输出需要

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
网络接收 RECV @RECV 098	RECV(098) S D C S:源首字 D:目的首字 C:控制首字	要求网络中的一个节点传送数据并接收此数据 	输出需要
发布命令 CMND @CMND 490	CMND(490) S D C S:源首字 D:目的首字 C:控制首字	发FINS命令并接收响应 	输出需要

3-24 文件存储指令

<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></tr>
<tr><td>读数据文件
FREAD
@FREAD
700</td><td>FREAD(700)
C
S1
S2
D

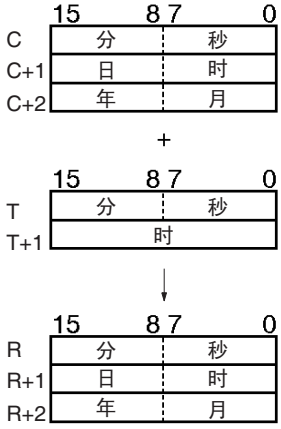
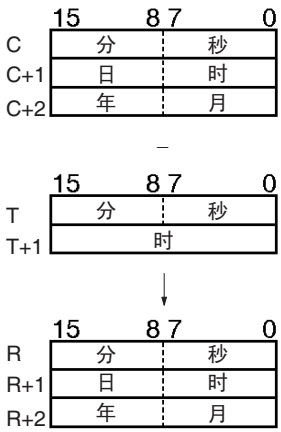
C:控制字
S1:源首字
S2:文件名
D:目的首字</td><td>从文件内存的指定数据文件中把指定的数据或数据量读到CPU单元的指定数据区中。
 
</td><td>输出需要</td></tr>
<tr><td>写数据文件
FWRIT
@FWRIT
701</td><td>FWRIT(701)
C
D1
D2
S

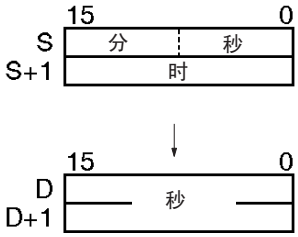
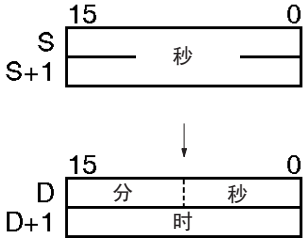
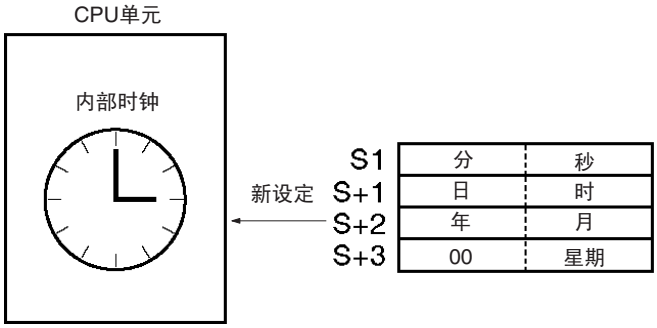
C:控制字
D1:目的首字
D2:文件名
S:源首字</td><td>用CPU单元数据区中指定的数据，复盖或添加在文件内存的指定数据文件中。如果指定文件不存在，就用此文件名建立新文件。
 
</td><td>输出需要</td></tr>
</table>

3-25 显示指令

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
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，并在外围设备 （如编程器）上显示该信息。	输出需要

3-26 时钟指令

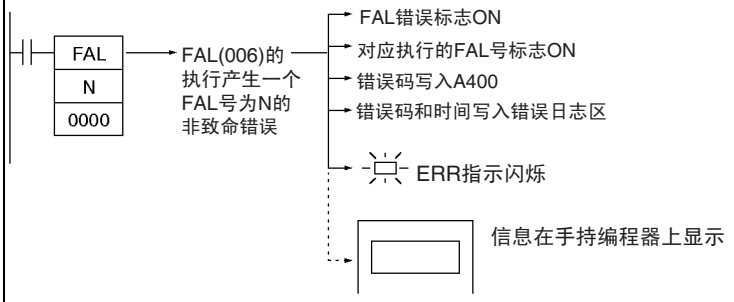
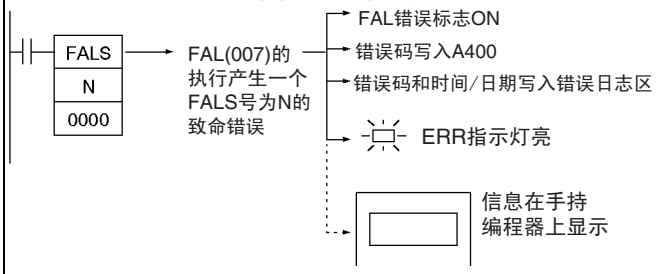
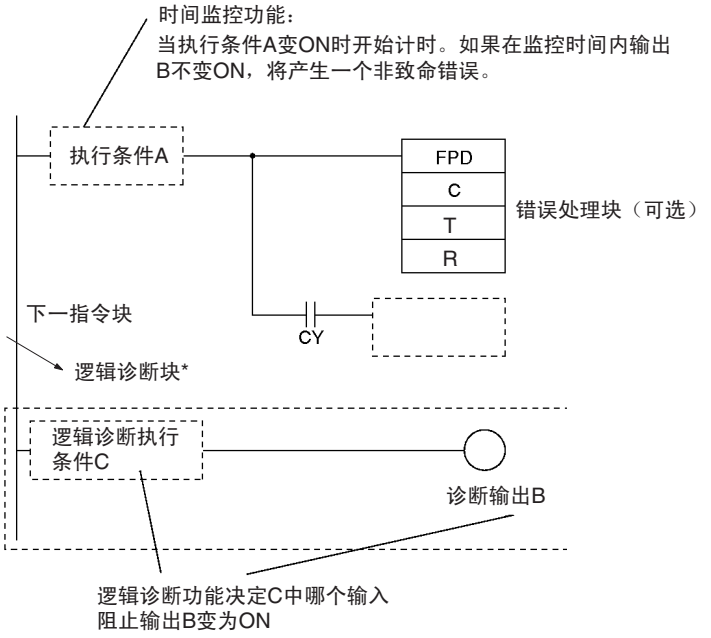
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
日历加法 CADD @CADD 730	CADD(730) C T R C:日历首字 T:时间首字 R:结果首字	把指定字中的日历数据和时间相加 	输出需要
日历减法 CSUB @CSUB 731	CSUB(731) C T R C:日历首字 T: 时间首字 R:结果首字	把指定字中的日历数据减时间数据 	输出需要

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
小时→秒 SEC @SEC 065	SEC(065) S D S:源首字 D:目的首字	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
秒→小时 HMS @HMS 066	HMS(066) S D S:源首字 D:目的首字	把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
时钟调整 DATE @DATE 735	DATE(735) S S:源首字	在指定源字的设定中设置内部时钟设定 	输出需要

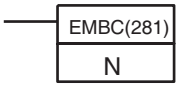
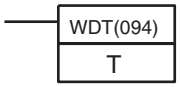
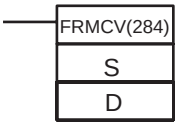
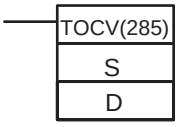
3-27 调试指令

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
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，并存入跟踪内存中。 TRSM(045) 可用于程序中任何地方，且可任意次使用。	输出需要

3-28 故障诊断指令

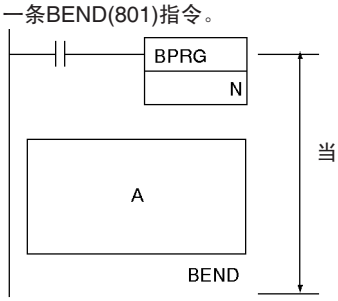
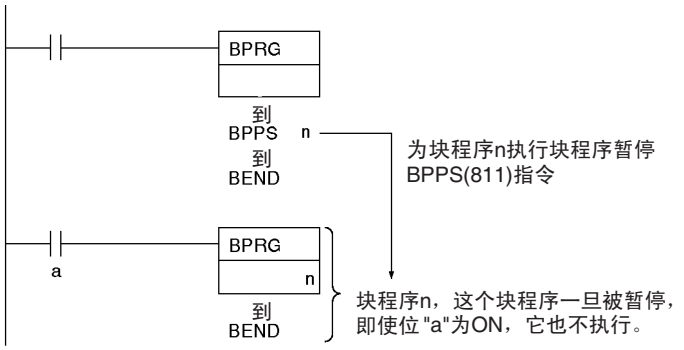
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
故障报警 FAL @FAL 006	FAL(006) N M N: FAL 号 M: 信息首字或产生的错误码 (#0000 ～ #FFFF)	产生或清除用户定义的非致命错误。非致命错误不会使PC停止运行  也生成 （模拟）致命系统错误。	输出需要
严重故障报警 FALS 007	FALS(007) N M N: FAL 号 M: 信息首字或产生的错误码 (#0000 ～ #FFFF)	产生用户定义的致命错误。该错误使PC停止运行  也生成 （模拟）致命系统错误。	输出需要
故障点检测 FPD 269	FPD(269) C T R C:控制字 T:监控时间 R:寄存器字首字	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，并找出哪个输入条件阻止输出变ON。  	输出需要

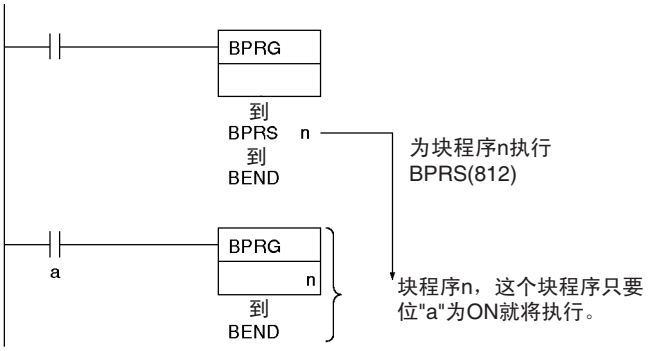
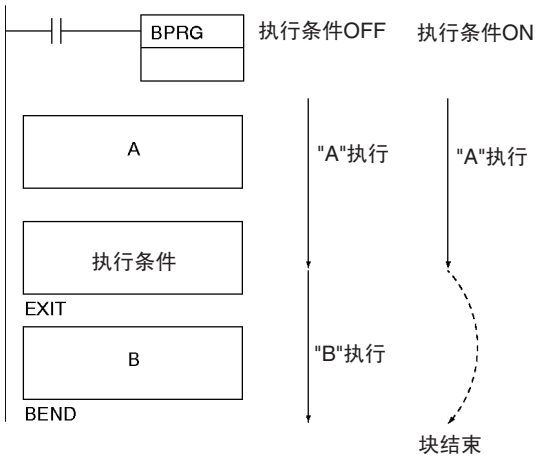
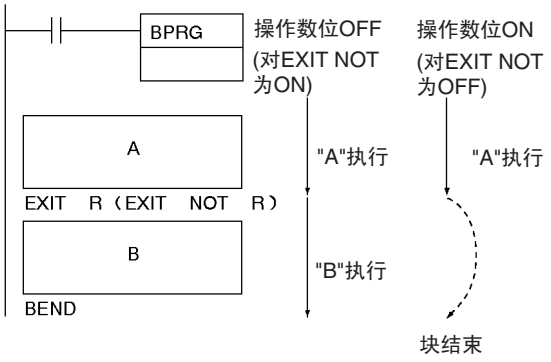
3-29 其它指令

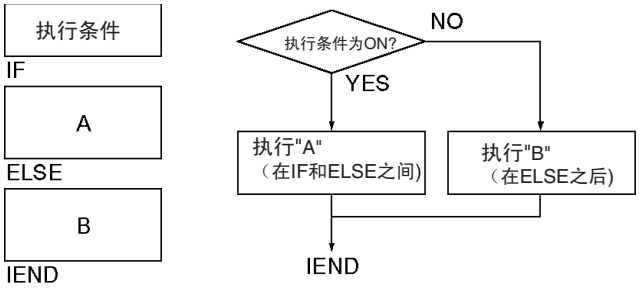
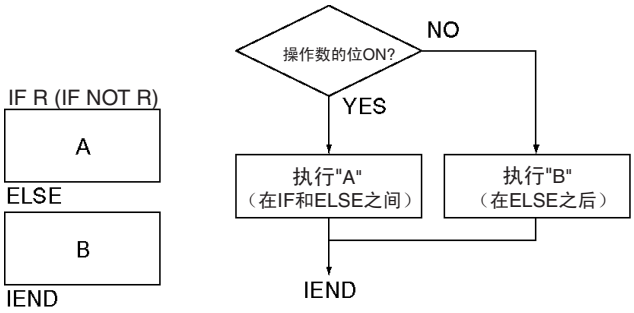
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
置进位 STC @STC 040		置进位标志 (CY)。	输出需要
清进位 CLC @CLC 041		进位标志 (CY) 清零。	输出需要
选择 EM BANK EMBC @EMBC 281	 N: EM Bank	改变当前 EM Bank。	输出需要
延长最大循环时间 WDT @WDT 094	 T:定时器设置	延长最大循环时间，但仅在此指令执行的循环时间。	输出需要
保存条件标志 （仅适用于 CS1-H，CJ1-H，CJ1M 或 CS1D） CCS @CCS 282		保存条件标志位状态。	输出需要
取条件标志 （仅适用于 CS1-H，CJ1-H，CJ1M 或 CS1D） CCL @CCL 283		读取已存储的条件标志位状态。	输出需要
由 CV 转换地址 （仅适用于 CS1-H，CJ1-H，CJ1M 或 CS1D） FRMCV @FRMCV 284	 S:存放CV系列内存地址字 D:目标变址寄存器	将 CV 系列 PLC 内存地址转换成相对应的 CS 系列 PLC 内存地址。	输出需要
地址转换到 CV （仅适用于 CS1-H，CJ1-H，CJ1M 或 CS1D） TOCV @TOCV 285	 S:存放CS系列内存地址的变址寄存器 D:目标字	将 CS 系列 PLC 内存地址转换成相对应的 CV 系列 PLC 内存地址。	输出需要

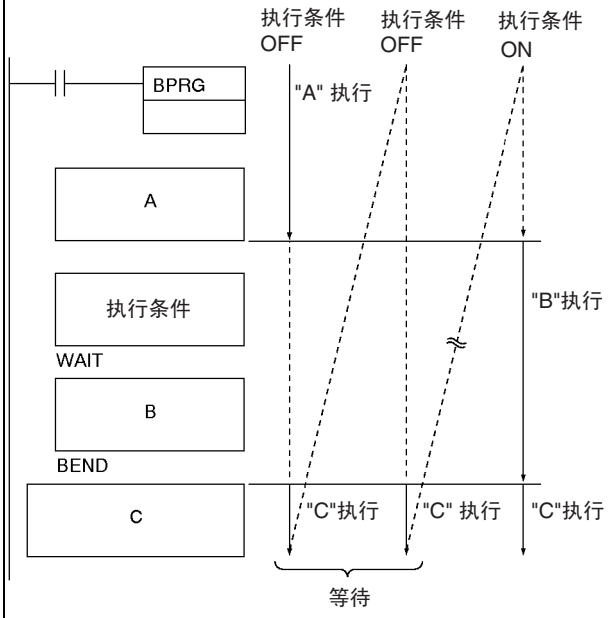
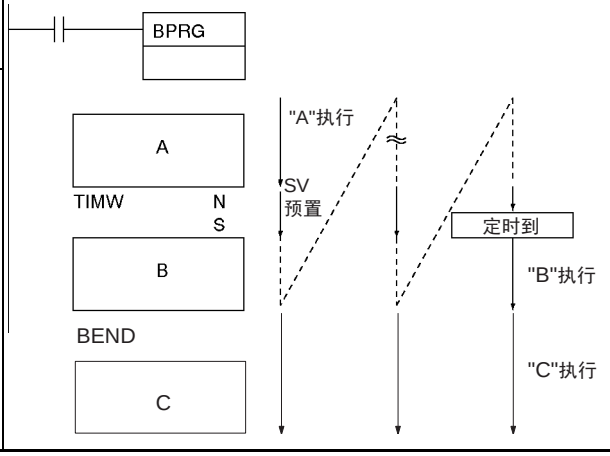
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
禁止外设服务（仅适用于 CS1-H，CJ1-H，CJ1M 或 CS1D） IOSP @IOSP 287	IOSP(287)	在并行处理模式或外设服务优先模式下程序执行时，禁止外设服务。	输出需要
允许外设服务（仅适用于 CS1-H，CJ1-H，CJ1M 或 CS1D） IORS 288	IORS(288)	使那些在并行处理模式下或外设优先模式下被 IOSP(287) 禁止的外设允许服务。	输出不需要

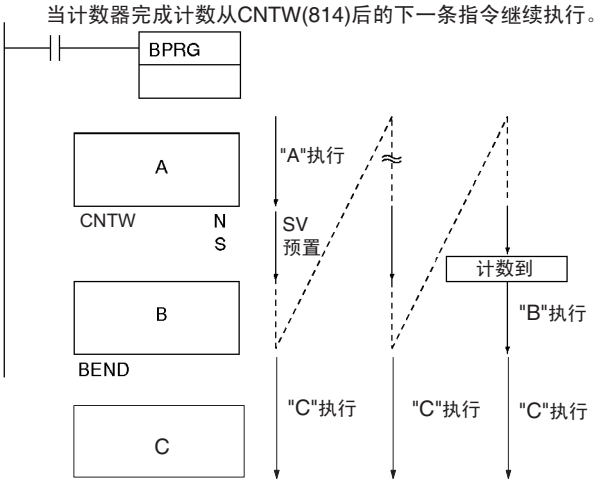
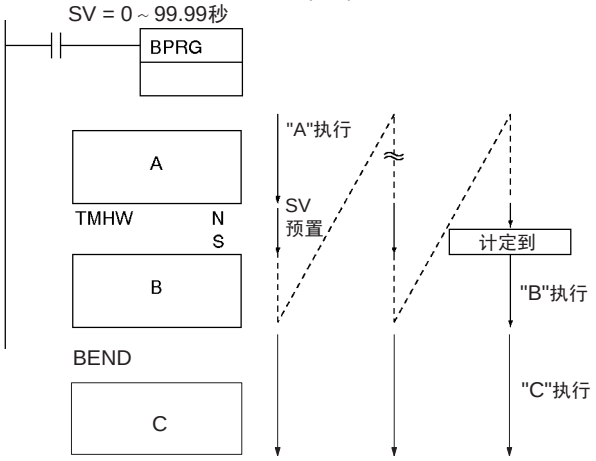
3-30 块程序指令

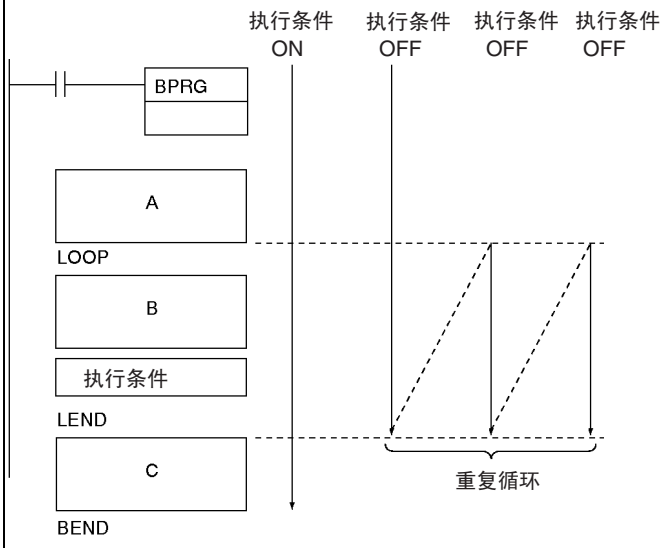
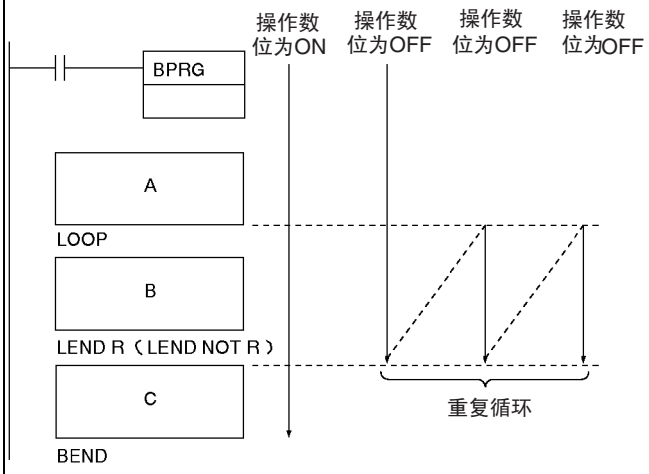
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
块程序开始 BPRG 096	BPRG(096) N N:块程序号	定义一个块程序区。对于每一条BPRG(096)指令，必须对应一条BEND(801)指令。 	输出需要
块程序结束 BEND 801		定义一个块程序区。对于每一条 BPRG(096) 指令，必须对应一条 BEND(801) 指令。	块程序需要
块程序暂停 BPPS 811	BPPS (811) N N:块程序号	从另一个块程序暂停或重新启动指定的块程序。 	块程序需要

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
块程序启动 BPRS 812	BPRS (812) N N:块程序号	从另一个块程序暂停和重新启动指定的块程序。 	块程序需要
条件块退出 EXIT 806	EXIT(806)	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
条件块退出 EXIT 806	EXIT(806) B 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
条件块退出（非） EXIT NOT 806		如果执行条件为 OFF，没有操作数位的 EXIT(806) 将退出程序。	块程序需要

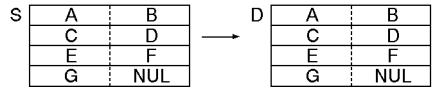
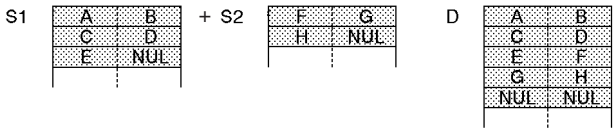
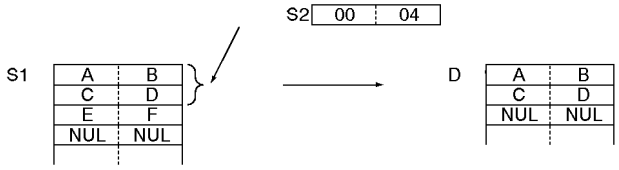
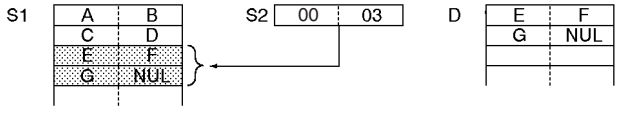
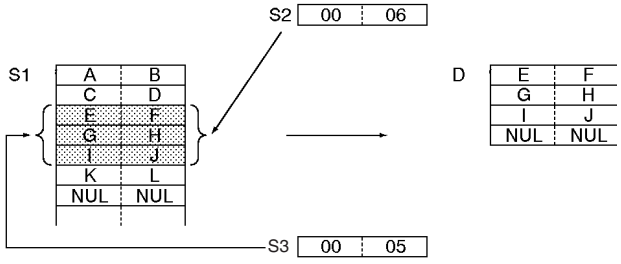
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
条件块程序分支 IF 802	IF (802)	如果执行条件为ON，那么就执行IF(802)和ELSE(803)之间的指令， 如果执行条件为OFF，将执行ELSE(803)和IEND(804)之间的指令。 	块程序需要
条件块程序分支 IF 802	IF (802) B B: 位操作数	如果操作数位为ON，那么就执行IF(802)和ELSE(803)之间的指令， 如果操作数位为OFF，将执行ELSE(803)和IEND(804)之间的指令。 	块程序需要
条件块程序分支 （非） IF NOT 802	IF (802) NOT B B: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

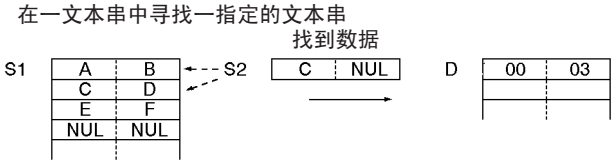
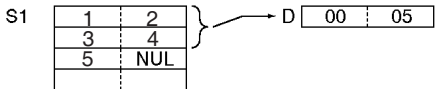
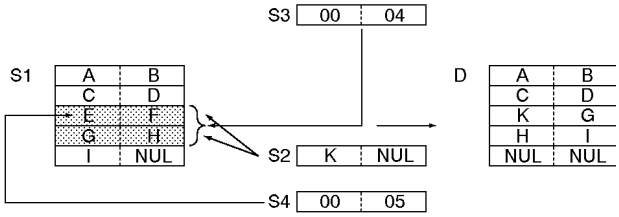
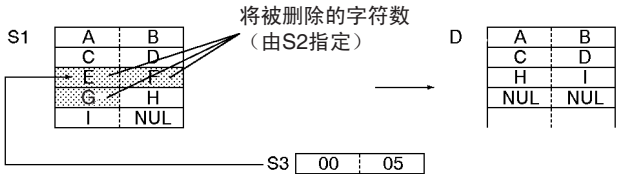
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
一个循环与等待 WAIT 805	WAIT(805)	如果WAIT(805)的执行条件为OFF，那么将跳过块程序中的其余指令。 	块程序需要
一个循环与等待 WAIT 805	WAIT(805) B B: 位操作数	如果操作数位为 OFF （对 WAIT NOT(805) 而言是 ON），那么将跳过块程序中的其余指令。在下一个循环中，除了 WAIT(805) 或 WAIT(805) NOT 外，不执行程序中的其它程序。当操作数位变为 ON （WAIT(805) NOT 为 OFF），执行从 WAIT(805) 或 WAIT(805) NOT 到程序结束之间的指令。	块程序需要
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 外，不执行程序中的其它程序。当操作数位变为 ON （WAIT(805) NOT 为 OFF），执行从 WAIT(805) 或 WAIT(805) NOT 到程序结束之间的指令。	块程序需要
定时器等待 TIMW 813 (BCD) TIMWX 816 （二进制） （仅适用于 CS1-H，CJ1-H，CJ1M 或 CS1D）	TIMW(813) N SV N: 定时器号 SV: 设定值 TIMWX(816) N SV N: 计时器号 SV: 设定值	延时执行块程序的剩余指令，直至定义的时间到。当定时器定时到从TIMW(813)后的下一条指令继续执行。 	块程序需要

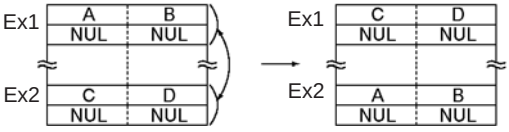
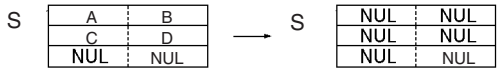
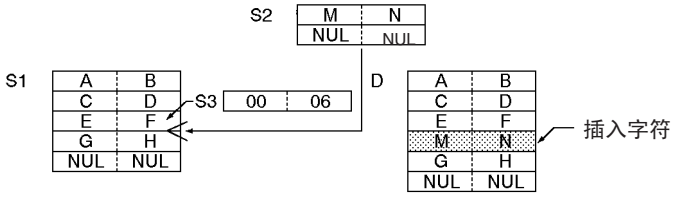
指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
计数器等待 CNTW 814 (BCD)	CNTW(814) N SV N:计数器号 SV:设定值 I:计数输入	延迟执行块程序的剩余指令，直至计数器计数到。 当计数器完成计数从CNTW(814)后的下一条指令继续执行。 	块程序需要
CNTWX 817 （二进制） （仅适用于 CS1-H，CJ1-H，CJ1M 或 CS1D）	CNTWX(817) N SV N: 计数器号 SV: 设定值 I: 计数输入		
高速定时器等待 TMHW 815 (BCD)	TMHW(815) N SV N:定时器号 SV:设定值	延迟执行块程序的剩余指令，直至定时时间到。 当定时时间到，从TMHW(815)后的下一条指令继续执行。 SV = 0 ~ 99.99秒 	块程序需要
TMHWX 818 （二进制） （仅适用于 CS1-H，CJ1-H，CJ1M 或 CS1D）	TMHWX(818) N SV N: 定时器 SV: 设定值		

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
循环 LOOP 809	---	LOOP(809)指定循环程序的开始  	块程序需要
循环结束 LEND 810	LEND(810)	LEND(810) 或 LEND(810) NOT 指定循环结束。当 LEND(810) 或 LEND(810) NOT 执行时，程序将循环回到前面的 LOOP(809) 下一指令执行，直到 LEND(810) 或 LEND(810) NOT 的操作数位分别变为 ON 或 OFF，或直至 LEND(810) 的执行条件变为 ON。	块程序需要
循环结束 LEND 810	LEND (810) B B:位操作数	如果LEND(810)的操作数位为OFF，（对LEND(810) NOT为ON），循环执行从LOOP(809)后的下一条指令处重复开始。如果LEND(810)的操作数位为ON（对LEND(810) NOT为OFF），循环将结束，并且到LEND(810)或LEND(810) NOT后的下一条指令处继续执行。  注　对LEND(810) NOT指令，操作数位的状态相反	块程序需要
循环结束 （非） LEND NOT 810	LEND(810) NOT B:位操作数	LEND(810) 或 LEND(810) NOT 指定循环结束。当 LEND(810) 或 LEND(810) NOT 执行时，程序将循环回到前面的 LOOP(809) 下一指令执行，直到 LEND(810) 或 LEND(810) NOT 的操作数位分别变为 ON 或 OFF，或直至 LEND(810) 的执行条件变为 ON。	块程序需要

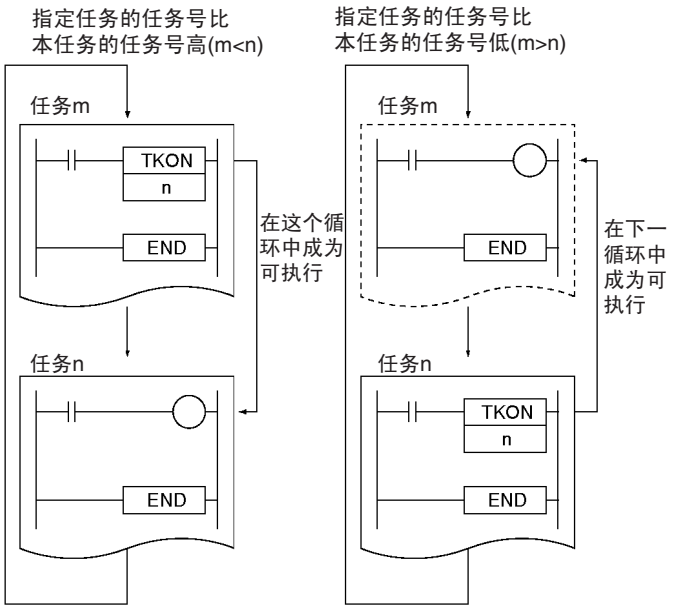
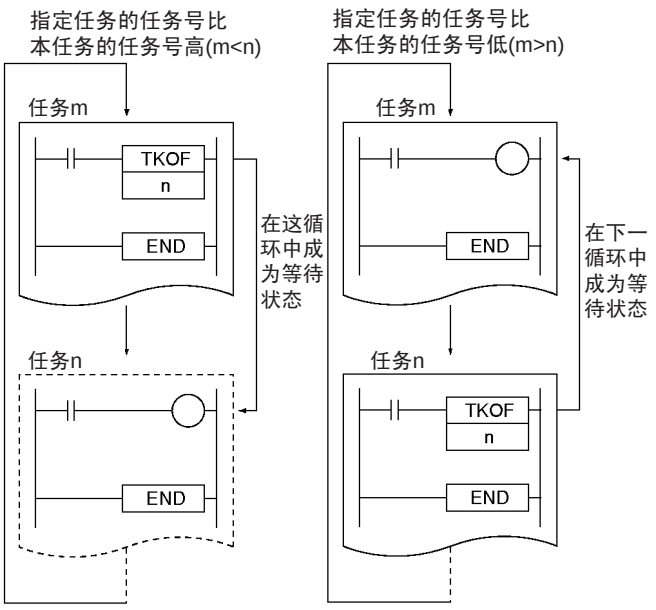
3-31 文本串处理指令

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
串传送 MOV$ @MOV$ 664	MOV$(664) S D S:源首字 D:目的首字	传送一个文本串 	输出需要
链接串 +$ @+$ 656	+$(656) S1 S2 D S1:文本串1 S2:文本串2 D:目的首字	将一个文本串与另一个文本串链接 	输出需要
取左串 LEFT$ @LEFT$ 652	LEFT$(652) S1 S2 D S1:文本串首字 S2:字符数 D:目的首字	从文本串的左边（开始），取指定数量的字符 	输出需要
取右串 RGHT$ @RGHT$ 653	RGHT$(653) S1 S2 D S1:文本串首字 S2:字符数 D:目的首字	从文本串的右边（末尾），读取指定数量的字符 	输出需要
取中间串 MID$ @MID$ 654	MID$(654) S1 S2 S3 D S1:文本串首字 S2:字符数 S3:开始位置 D:目的首字	从文本串中间任何位置，读取指定数量的字符 	输出需要

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
寻找串 FIND @FIND$ 660	FIND$(660) S1 S2 D S1:源文本串首字 S2:寻找文本串首字 D:目的首字	在一文本串中寻找一指定的文本串 	输出需要
计算串长 LEN$ @LEN$ 650	LEN$(650) S D S1:文本串首字 D:目的首字	计算一文本串的长度 	输出需要
取代串 RPLC$ @RPLC$ 661	RPLC$(661) S1 S2 S3 S4 D S1:文本串首字 S2:取代的文本串首字 S3:字符数 S4:开始位置	用一指定的文本串从一指定位置取代一个文本串 	输出需要
删除串 DEL$ @DEL$ 658	DEL$(658) S1 S2 S3 D S1:文本串首字 S2:字符数 S3:开始位置 D:目的首字	从文本串中间删除指定的文本串 	输出需要

指令 助记符 代码	符号 / 操作数	功能	位置 执行条件
交换串 XCHG$ @XCHG$ 665	XCHG$(665) Ex1 Ex2 Ex1:交换字1首字 Ex2:交换字2首字	用一指定文本串替换另一指定文本串 	输出需要
清除串 CLR$ @CLR$ 666	CLR$(666) S S:文本串首字	用零(00 heX）清除整个文本串 	输出需要
插入串 INS$ @INS$ 657	INS$(657) S1 S2 S3 D S1:基础文本串首字 S2:插入文本串首字 S3:开始位置 D:目的首字	在一文本串中间插入一指定的文本串 	输出需要
串比较 LD, AND, OR + =$, <>$, <$, <=$, >$, >=$ 670 (=$) 671 (<>$) 672 (<$) 673 (<=$) 674 (>$) 675 (>=$)	LD Symbol S1 S2 AND Symbol S1 S2 OR Symbol S1 S2 S1:文本串1 S2:文本串2	串比较指令 （=$, <>$, <$, <=$, >$, >=$），根据 ASCII 代码值比较两个文本串。如果比较结果为真，那么就为 LOAD， AND 或 OR 产生一个 ON 的执行条件。	LD：不需要 AND， OR：需要

3-32 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
任务 ON TKON @TKON 820	TKON(820) N N:任务号	使指定的任务号可执行 	输出需要
任务 OFF TKOF @TKOF 821	TKOF(821) N N:任务号	使指定的任务为等待状态 	输出需要

第 4 章
任务

本章描述任务的操作。

4-1 任务特性

4-1-1 概述

CS/CJ 系列控制操作器可按功能、操作设备、过程、开发或其它标准进行划分，每个操作可在一个被称为任务的独立单元内编程。使用任务有如下优点：

1,2,3...

1. 程序可由几个人同时编制每个单独设计的程序部分可很容易地组合成一个单独的用户程序。
2. 程序可被标准化模块

   将以下编程设备功能结合起来，编制一个标准的，而不是为某个特定系统（如机器，设备）编制的模块程序，这就意味着由几个人单独编制的程序可很容易地组合起来。

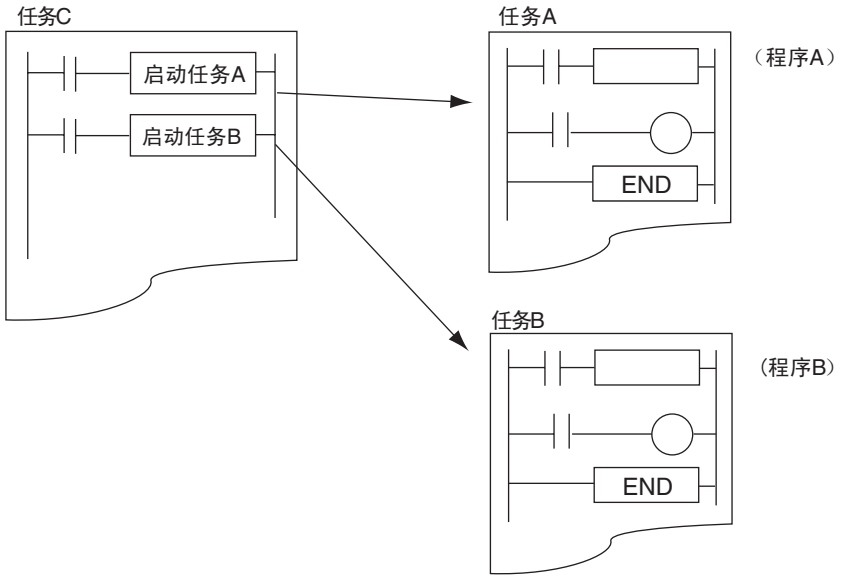
   - 编程使用的符号
   - 符号的全局和局部表示方法
   - 自动分配局部符号地址
3. 改进整体响应

   整体响应得到改进是因为程序系统被分割成各个独立的控制程序和可根据需要运行相应程序。
4. 易于修改和调试
   - 修改效率更高。因为任务可由几个人独立编制，并且每个独立任务都分开修改和调试。
   - 维护简单。为了使程序规范或其它更改，仅需要对相关任务作修改。
   - 调试效率更高。由于符号被设定成局部符号或全局符号，并且局部符号可通过编程器自动分配地址，很容易判定一个地址是局部的还是全局的。程序间的地址仅在调试时检查一次。
5. 易于切换程序

   如果有必要改变操作时，用程序中的一个任务控制指令使特定的任务 （程序）运行有效。

6. 易于理解的用户程序。

   程序以模块化为结构使得程序简单易懂，程序段的处理就像用传统的跳转指令方式。



4-1-2 任务和程序

- 独立的程序一对一地分配给任务，最多有 288 个程序（任务）是可控制的。通常，任务按下例类型划分：
- 循环任务
- 中断任务

注 1. 可产生的最大任务量可达 288 个。其中最多可有 32 个循环任务和 256 个中断任务。每个任务都有自己唯一的标号，循环任务依次为 0 到 31，中断任务为 0 到 255。

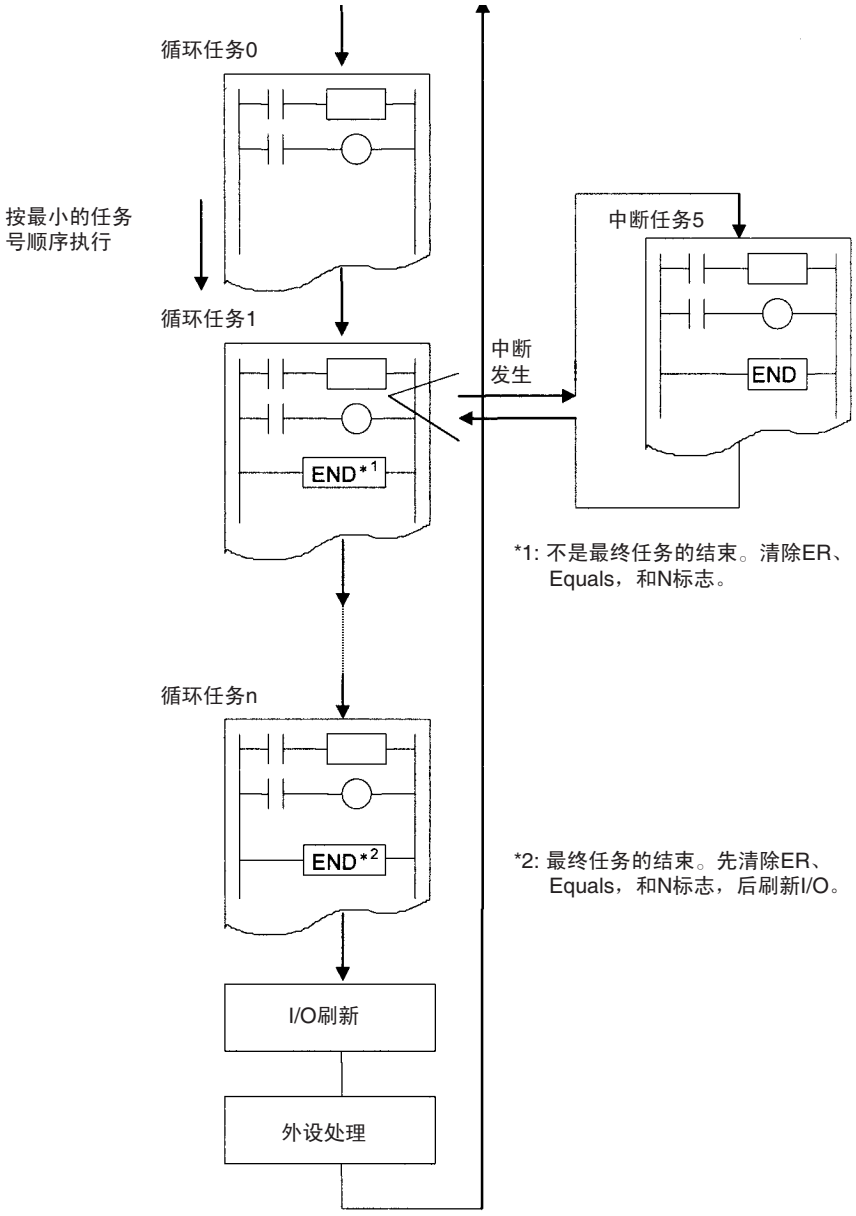
2. 使用 CS1-H， CJ1-H,CJ1M 或 CS1D CPU 时，中断任务 （中断任务 0 ～ 255）用 TKON 指令启动时它们将按循环任务执行，这时它们被称为 “附加循环任务” 。如果使用附加循环任务，则可使用的循环任务数将达 288 个。

3. CJ1 CPU 目前不支持 I/O 中断任务和外部中断任务。因此，CJ1 CPU 可处理的最大任务数为 35 个，即 32 个循环任务和 3 个中断任务。可产生和处理的程序总量也为 35 个。

每个分配给任务的程序必须使用 END(001) 指令结束。仅在所有任务程序执行完后， I/O 端口才被刷新。

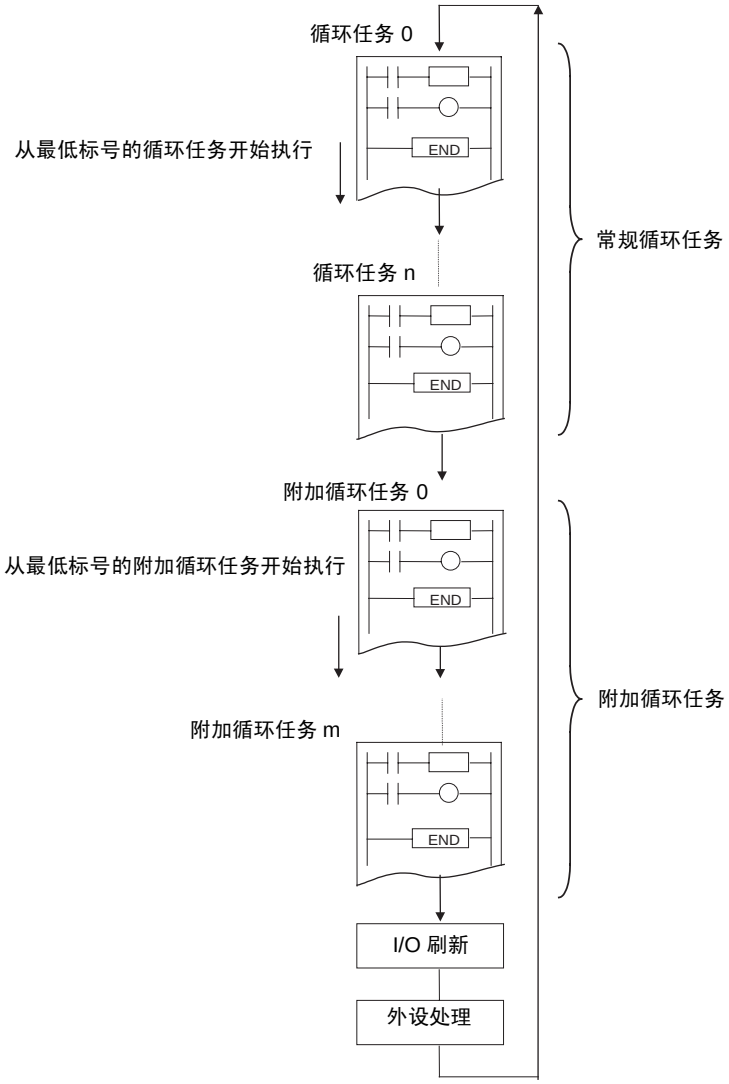
4-1-3 CPU 单元基本操作

CPU 将从标号最低的任务开始依次执行循环任务（包括附加循环任务，仅适用于 CS1-H,CJ1-H,CJ1M 或 CS1D CPU 单元）。如果中断发生，它将终止执行循环任务而去执行中断任务。



注 所有条件标志（ER、CY、Equals、AER、等）和指令条件（内部锁存为 ON 等）在任务开始时被复位。因此，条件标志不可读，而且内部锁存 / 内部锁存清空 （IL/ILC）指令，跳转 / 跳转结束 （JMP/JME）指令，或子程序调用 / 子程序进入 （SBS/SBN）指令不能分开在两个任务间使用。

使用 CS1-H,CJ1-H,CJ1M 或 CS1D CPU 单元时，当中断任务由 TKON 指令启动，它能像循环任务一样执行，这些任务称为 “附加循环任务”，附加循环任务（中断任务标号：0 ～ 255）将在常规的循环任务（循环任务 0 ～ 31）执行后，从标号最低的任务开始依次执行。


循环任务 0
从最低标号的循环任务开始执行
END
循环任务 n
END
常规循环任务
附加循环任务 0
从最低标号的附加循环任务开始执行
END
附加循环任务 m
END
附加循环任务
I/O 刷新
外设处理

4-1-4 任务类型

任务通常划分为循环任务和中断任务。中断任务可进一步划分为断电中断，定时中断， I/O 中断 （仅限于 CS 系列 CPU）和外部中断任务 （仅限于 CS 系列）。中断任务亦可作为附加循环任务执行。

注 在使用 CS1-H,CJ1-H,CJ1M 或 CS1D CPU 单元时，中断任务可在循环任务中通过 TKON 指令启动执行它们，这时任务被称为 “附加循环任务”。

循环任务

一个处于就绪状态的循环任务都是从最低编号开始按序每个周期执行一次 （从程序顶端到 END （001）指令），循环任务的最大总量为 32 个 （循环任务标号：0 ～ 31）。

注 在使用 CS1-H,CJ1-H,CJ1M 或 CS1D CPU 单元时，中断任务（中断任务标号: 0 ～ 255）能像常规循环任务一样执行，如果使用附加循环任务，那么可使用的总的循环任务数为 288 个。

中断任务

如果中断发生，即使当前循环任务 （包括附加循环任务）正在执行，也将转去执行中断任务。在循环周期中，只要中断条件成立，中断任务可在任何时候执行，这包括用户程序执行时、 I/O 口刷新时或外设处理时。

在使用 CS1-H,CJ1-H,CJ1M 或 CS1D CPU 单元时，中断任务能被当作循环任务执行。（CS1D CPU 不支持中断，使用 CS1D CPU 时，中断任务仅能作为附加循环任务执行）

在 CJ1M CPU 单元中，内置的中断输入和高速计数器输入能用来激活中断任务，详情请见有关 CJ 系列内置 I/O 口操作手册。

断电中断任务

电源关中断任务在 CPU 电源被关断时执行，在编程时仅能使用一个断电中断任务 （中断任务标号为 1）。

注 断电中断任务必须在下述时间消逝前或任务被强制退出时执行。

10ms （断电检测延迟时间）

断电检测延迟时间在 PLC 设置时设定。

定时中断任务

定时中断任务在一个基于 CPU 内部计时器设定的固定时间间隔后执行，最多可有两个定时中断任务 （中断任务标号为 2 和 3）

注 设定中断屏蔽 （MSKS(690)）指令是用来为定时中断任务设定中断的指令，中断时间可在 PLC 设置时设为 10ms 或 1.0ms 为单位进行递增。

I/O 中断任务

如果中断输入单元的输入为 ON 时，执行 I/O 中断任务。I/O 中断任务的最大总量为 32 个 （中断任务标号：100 ～ 131），中断输入单元必须安装在 CPU 机架上，就 CJ1-H CPU 单元而言，中断输入单元必须作为紧靠 CPU 的五个单元（槽 0 ～ 4）中的一个与 CPU 相连。而对 CJ1M CPU 单元，中断输入单元必须作为紧靠 CPU 的三个单元 （槽 0 ～ 2）中的一个与 CPU 相连。I/O 中断单元若被安装在其他地方，将不能被用作 I/O 中断任务的要求执行。

CJ1 CPU 不支持 I/O 中断。

外部中断任务

当由特殊 I/O 单元， CPU 总线或内装板 （仅限于 CS 系列）用户程序要求时，将执行外部中断任务。然而，特殊 I/O 单元和 CPU 总线单元必须安装在 CPU 机架上。就 CJ1-H CPU 而言，中断输入单元必须作为紧靠 CPU 的五个单元（槽 0 ～ 4）中的一个与 CPU 相连。而对 CJ1M CPU 单元，中断输入单元必须作为紧靠 CPU 的三个单元 （槽 0 ～ 2）中的一个与 CPU 相连。中断单元若被安装在其它地方，将不能用来产生外部中断。

外部中断任务最多可达 256 个 （中断任务标号：0 ～ 255）。如果外部中断任务与断电中断任务、定时中断任务或 I/O 中断任务标号相同时，中断任务将在其中任何一个条件满足 （即两个条件执行逻辑 “或” 操作）时执行。但一般来说，任务标号不应该重复。

CJ1 CPU 不支持 I/O 中断。

附加循环任务 （仅限于 CS1-H,CJ1-H,CJ1M 或 CS1D CPU）

中断任务能像常规循环任务一样在每个周期内执行。附加循环任务是在常规循环任务 （循环任务标号：0 ～ 31）执行完毕后，从标号最低的任务开始执行。最多可有 256 个附加循环任务 （中断任务标号：0 ～ 255）。然而，循环中断任务与常规循环任务的不同点在于循环中断任务得由 TKON（820）指令启动。此外， TKON(820) 和 TKOF(821) 指令不能在附加循环任务中使用。这表示在一个附加循环任务中不可控制常规循环任务和其它附加循环任务。

如果外部中断任务与断电中断任务、定时中断任务或 I/O 中断任务标号相同时，中断任务将在其中任何一个条件满足 （即两个条件执行逻辑 “或” 操作）时执行。不要使中断任务与一般中断任务和附加循环任务同号。

注

1. 断电任务其优先级最高，当电源关闭时，即使其他中断任务正在执行，也将转去执行断电中断任务。
2. 如果一个中断任务正在执行时发生定时中断，I/O 中断或外部中断，那么这些中断任务只有在当前的中断任务执行完毕后才能执行。如果多个中断任务同时发生，那么将从标号最低的中断任务开始依次执行。
3. 常规循环任务和附加循环任务的区别如下表所示

项目	附加循环任务	常规循环任务
启动激活	不可设置	由 CX-Programmer 设置
使用 TKON/TKOF 指令进入任务	可以	不可以
任务标志	不支持	支持
最初任务执行标志 （A20015）和任务开始标志 （A20014）	不支持	支持
变址 （IR）和数据 （DR）寄存器值	在任务启动时不定义 （与常规中断任务相同）。其值在上一周期内设定，不可读	在运行开始时不定义，其值在上一周期内设定，可读

4. CJ1 CPU 单元不支持 I/O 中断和外部中断任务。

4-1-5 任务执行条件和设置

下表描述了任务的执行条件，相关设置和状态

任务		编号	执行条件	相关设置
循环任务		0 ～ 31	在每个周期中，如果任务处于就绪状态（最初设为启动任务或由 TKON(820) 指令启动），一旦获得执行权就将执行	无
中断任务	断电中断任务	中断任务 1	CPU 电源关断时执行	• 在 PLC 设置时断电中断使能
	定时中断任务 0 和 1	中断任务 2 和 3	根据 CPU 单元内部时钟预先设定的时间间隔每次执行一次	• 定时中断时间（在 0~9999 间）可由中断屏蔽指令（MSKS）设定。 • 在 PLC 设置时，定时中断时间单位可设定为 10ms 或 1.0ms
	I/O 中断任务 00 ～ 31	中断任务 100 ～ 131	当 CPU 机架上的中断输入单元为 ON 时执行	• 利用设置中断屏蔽指令（MSKS）取消指定的输入屏蔽
	外部中断任务 0 ～ 255	中断任务 0 ～ 255	当 CPU 机架上的特殊 I/O 单元中的用户程序或 CPU 总线单元的或内装板（仅限于 CS 系列）用户程序请求时执行	无（一直处于激活状态）
附加循环任务（仅限 CS1-H， CJ1-H， CJ1M 或 CS1D CPU）		中断任务 0 ～ 255	在每个周期中，如果任务处于就绪状态（由 TKON(820) 指令启动），一旦获取执行权就将执行	无（一直处于激活状态）

注 1. 中断输入单元必须安装在 CPU 机架上，就 CJ1-H CPU 而言，中断输入单元必须作为紧靠 CPU 的五个单元（槽 0 ～ 4）中的一个与 CPU 相连。而对 CJ1M CPU 单元，中断输入单元必须作为紧靠 CPU 的三个单元（槽 0 ～ 2）中的一个与 CPU 相连。 I/O 中断单元若被安装在其他地方，将不能被用来要求 I/O 中断任务的执行。

2. 特殊的 I/O 单元和 CPU 总线单元必须安装在 CPU 机架上，就 CJ1-H CPU 而言，单元必须作为紧靠 CPU 的五个单元（槽 0 ～ 4）中的一个与 CPU 相连。而对 CJ1M CPU 单元，单元必须作为紧靠 CPU 的三个单元（槽 0 ～ 2）中的一个与 CPU 相连。单元若被安装在其它地方，将不能用来产生外部中断。

3. 当使用手持编程器对存储单元进行清空操作时，循环任务和中断任务的个数将受到限制。
   - 仅能生成循环任务 0
     使用手持编程器不能生成循环任务 1 ～ 31，但是如果任务已由 CX-Programmer 产生，那么这些任务是可以由手持编程器编辑的。
   - CS 系列仅能生成中断任务 1， 2， 3 和 100 到 131。
     使用手持编程器不能生成中断任务 0、4 到 99 和 132 到 255（在 CJ1M CPU 中，能生成中断任务 140 ～ 143），但是如果任务已由 CX-Programmer 产生，那么这些任务是可以由手持编程器编辑的。

4-1-6 循环任务状态

本节描述了循环任务的状态，包括附加循环任务（仅限于 CS1-H,CJ1-H,CJ1M 或 CS1D CPU）。

循环任务共有四个可能状态：“禁止”状态，“就绪”状态，“运行”状态（可执行）和 “等待”状态。

“禁止”状态 （INI）

处于“禁止”状态下的任务不能执行，在编程（PROGRAM）模式下，所以循环任务都为“禁止”状态。任何循环任务一旦从编程（PROGRAM）模式切换到其它状态，在不返回到编程 （PROGRAM）模式下时，是不能重新返回为“禁止”状态的。

“就绪”状态

当任务处于 “就绪”状态时，任务属性可控。任务属性可由 TASK ON 指令设定为激活任务或在运行操作启动时。

指令激活任务

TASK ON （TKON(820)）指令可用来把指令激活循环任务从 “禁止”和 “等待”状态切换为 “就绪”状态。

操作激活任务

在操作模式从编程（PROGRAM）模式变成运行（RUN）模式或监控(MONITOR) 模式时，一个操作激活循环任务可从“禁止”状态切换为“就绪”状态。这种应用仅对常规循环任务有效。

注 在任务 0 到任务 31 开始操作时，可用一个编程设备设定一个或多个任务为 " 就绪 " 状态。然而这种设置方法对附加循环任务行不通。

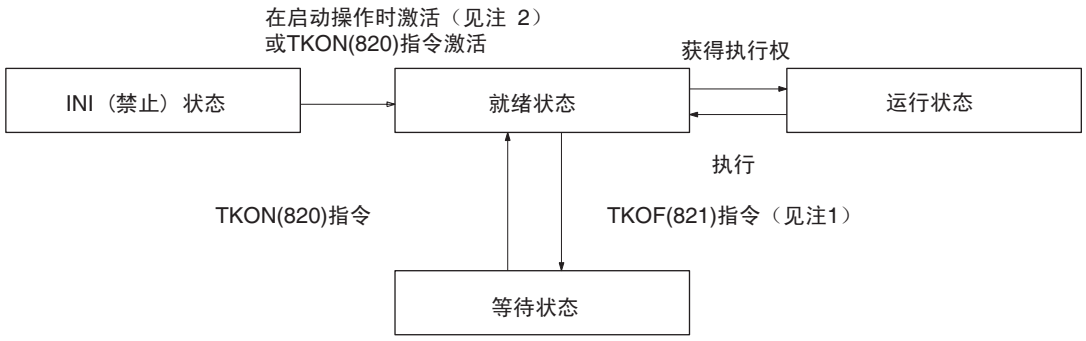
“运行”状态

当处于 “就绪”状态的循环任务获取执行权时，任务将从 “就绪”状态切换为 “运行”状态并执行任务。

“等待”状态

可用 TASK OFF （TKOF(821)）指令把循环任务从 “就绪”状态变为 “等待”状态。

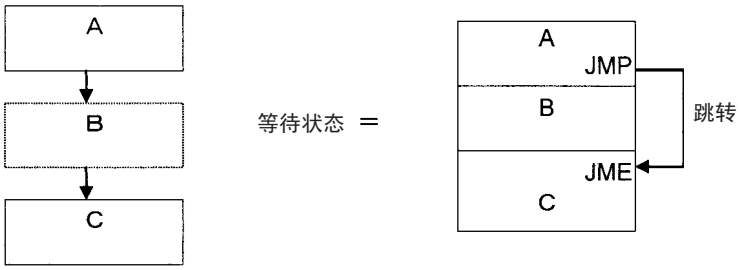
4-1-7 状态变换



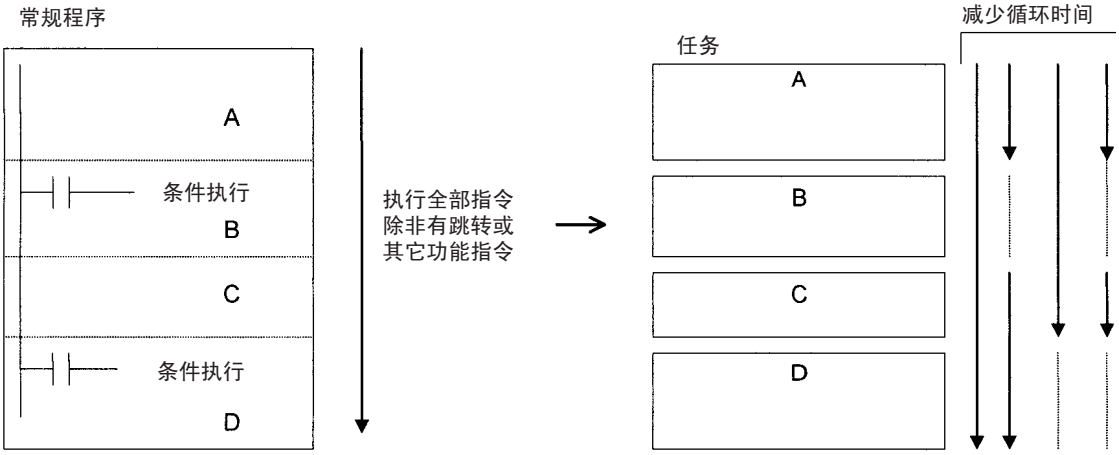
注 1. 使用 TKOF(821) 指令可使处于 “运行”状态下的任务变为 “等待”状态，即使在本任务内执行 TKOF(821) 执行也一样。

2. 在开始操作时激活任务方式仅对常规循环任务有效，对附加循环任务无效。

“等待”状态的效果相当于一个跳转 （JMP/JME）, 处于 “等待”状态的任务的输出将保持不变。



在“等待”状态下不执行指令，因此指令执行时间不增加。不需要执行的程序在任何时候都可以被生成为任务并设为“等待”状态，以缩短循环周期。



注 简言之，就是处于“等待”状态的任务在任务执行时被跳过。改变任务为“等待”状态不需要终止程序。

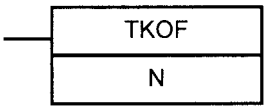
4-2 使用任务

4-2-1 TASK ON 和 TASK OFF 指令

在程序中，TASK ON（TKON（820））和 TASK OFF（TKOF(821)）指令是用来在“就绪”状态与“等待”状态间切换循环任务的（包括附加循环任务）。

注 附加循环任务仅被 CS1-H,CJ1-H,CJ1M 或 CS1D CPU 支持。

N:任务标号 当执行条件为ON时任务进入就绪状态，相应的任务标志位为ON。



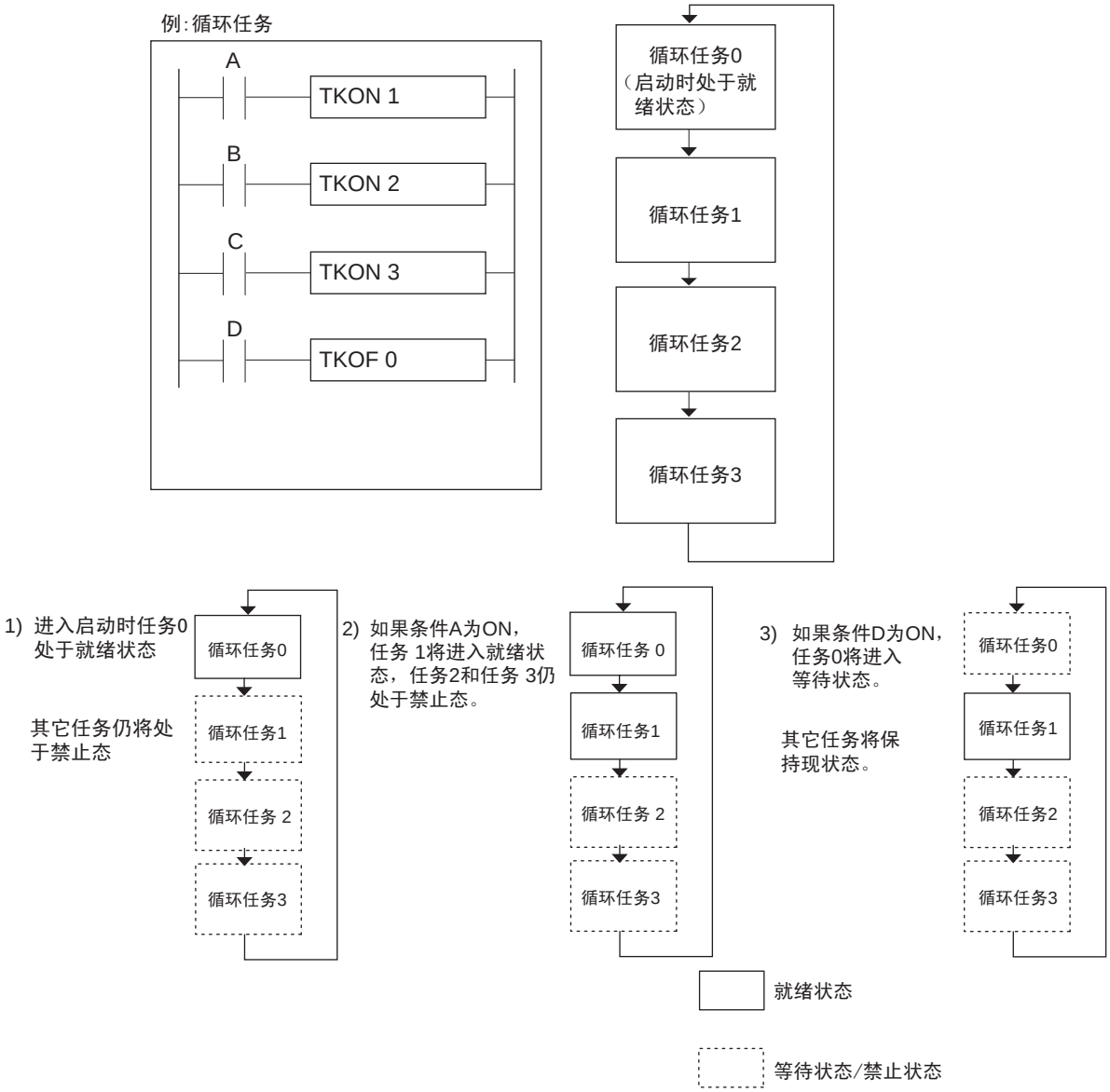
N:任务标号 当执行条件为ON时任务进入等待状态，相应的任务标志位为OFF。

注：附加循环任务无标志位

在任何时候，都可用任务开和任务关指令在“就绪”状态和“等待”状态间对任何循环任务进行状态切换。处于“就绪”状态的循环任务在随后的周期中将保持此状态，同样，在随后的周期中处于“等待”状态的循环任务也将保持状态不变。

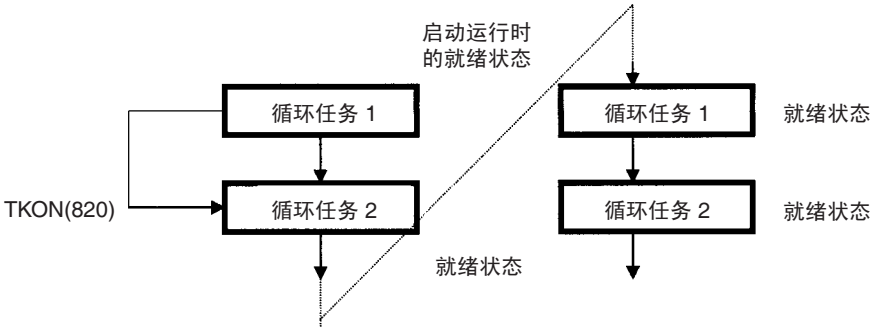
任务开和任务关指令仅能用于循环任务，不可用于中断任务。

注 在每个周期内必须至少要有一个循环任务处于“就绪”状态，否则，任务出错标志位（A29512）将变为 ON， CPU 将停止运行。

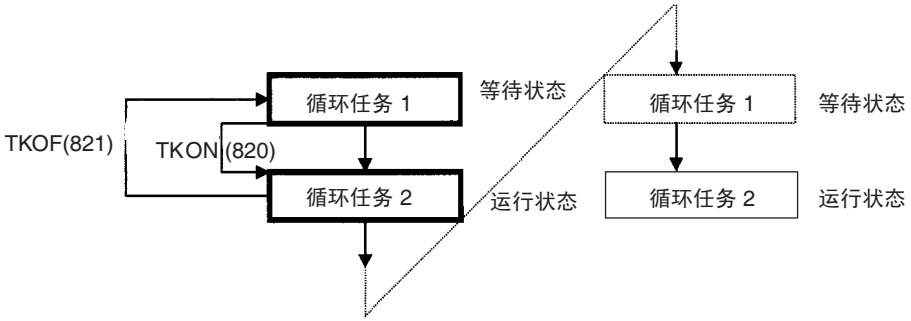


任务与执行周期

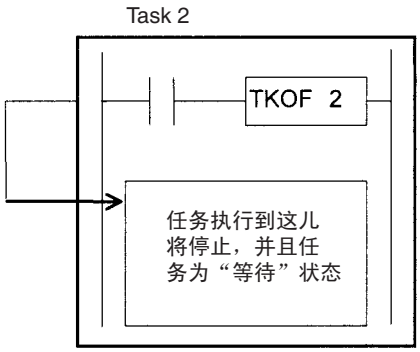
处于“就绪”状态的循环任务（包括附加循环任务）在随后的周期内将保持状态不变。



处于“等待”状态的循环任务在随后的周期内保持状态不变。为了把任务从“等待”状态切换为“就绪”状态，将必须使用 TKON(820) 激活任务。



如果在任务中执行 TKOF(821) 指令，那么任务将在执行到指令时停止执行，并且任务变为 “等待” 状态。



循环任务编号与执行周期 （包括附加循环任务）

如果任务 m 打开任务 n，且 m>n, 那么任务 n 将在下个周期中进入“就绪”状态。

例：如果任务 5 打开任务 2，那么任务 2 将在下个周期中进入 “就绪” 状态。

如果任务 m 打开任务 n 且 m<n, 那么任务 n 将在本周期中进入 “就绪” 状态。

例：如果任务 2 打开任务 5，那么任务 5 将在本周期中进入 “就绪” 状态。

如果任务 m 置任务 n 为 “等待” 状态且 m>n，那么任务 n 将在下个周期中进入 “等待” 状态。

例：如果任务 5 置任务 2 为 “等待” 状态，那么任务 2 将在下个周期中进入 “等待” 状态。

如果任务 m 置任务 n 为 “等待” 状态且 m<n, 那么任务 n 将在本周期中进入 “等待” 状态。

例：如果任务 2 置任务 5 为 “等待” 状态，那么任务 5 将在本周期中进入 “等待” 状态。

I/O 存储器与任务的关系

变址寄存器 （IR） 与数据寄存器 (DR) 在用法上有两种不同方式：1）独立于任务， 2）被所有任务共享 （仅限于 CS1-H,CJ1-H,CJ1M 或 CS1D CPU 单元）

就独立寄存器而言，循环任务 1 中使用的 IR0 与循环任务 2 中使用的 IR0 是不一样的。而对于共享寄存器而言，循环任务 1 中使用的 IR0 与循环任务 2 中使用的 IR0 是相同的。

通过 CX-Programmer 可对寄存器是独立的还是共享的可进行设定。

- I/O 存储器中的其它字和位被所有任务共享。例如：在循环任务 1 与循环任务 2 中 CIO001000 表示同一位。因此，在编程过程中使用 I/O 存储区时必须时刻小心，因为如果值在某个任务中被改变，其它任务中所使用的值将是改变过的值。

I/O 存储器	与任务的关系
CIO，辅助存储器，数据存储器和除 IR 与 DR 以外的所有存储区 （见注 1）	被所有任务共享
变址寄存器 （IR）和数据寄存器 （DR）（见注 2）	被每个任务独立使用

注 1. 当前 EM Bank 也被任务共享。因此，如果当前 EM Bank 标号在循环任务 1 中改变，那么新的当前 EM Bank 标号对循环任务 2 同样有效。

2. 中断任务（包括附加循环任务）启动时不设定 IR 和 DR 值。如果在中断任务中使用 IR 和 DR, 他们的值必须在中断任务中由 MOVR/MOVRW （传输至寄存器和定时器 / 计数器当前值 PV 至寄存器）指令设定。在中断任务执行完毕后， IR 和 DR 值自动返回为中断之前的值。

定时器操作与任务的关系

当任务本身的状态切换或包含的定时器任务的状态变为 “等待”状态或返回为“就绪”状态时，对标号为 0000 ～ 2047 的定时器用 TIM， TIMX， TIMH，TIMHX，TMHH，TMHHX，TIMW，TIMWX，TMHW，TMHWX 进行编程的定时器当前值将会刷新。

如果任务中的 TIM 进入 “等待”状态并返回为 " 就绪 " 状态，当前值为 0 时执行 TIM 指令，那么定时完成标志位将变为 ON。（定时器定时完成标志位仅在指令执行时更新）如果指令在当前值不为 0 时执行，当前值将在任务状态为“就绪”时继续刷新。

- 在任务处于“等待”状态时，对编号为 2048 ～ 4098 的定时器进行编程的定时器当前值将保持不变。

条件标志和任务的关系

在每个任务执行前，所有的条件标志位将复位。因此，在任务 2 中不能读取任务 1 结束时的条件标志位状态。然而，对于 CS1-H， CJ1-H， CJ1M 或 CS1D CPU 单元而言， CCS(282) 和 CCL(283) 指令可以从在程序的其它部分读条件标志状态，即从其它任务中读取条件标志位状态。

注 在使用手持编程器对条件标志位状态进行监控时，手持编程器将显示周期结束时的标志位状态，即周期中最后一个任务结束时的状态。

4-2-2 任务指令限制

同一任务中要求的指令

下列指令必须放在同一任务中，任何企图在两个任务间分割这些指令将导致ER 标志位为 ON, 且指令停止执行。

助记符	指令
JMP/JME	跳转 / 跳转结束
CJP/JME	条件跳转 / 跳转结束
CJPN/JME	条件不满足跳转 / 条件跳转结束
JMP0/JME0	多重跳转 / 跳转结束
FOR/NEXT	FOR/NEXT 循环
IL/ILC	内部锁存 / 内部锁存清除
SBS/SBN/RET	子程序调用 / 子程序进入 / 子程序返回
MCRO/SBN/RET	宏指令 / 子程序进入 / 子程序返回
BPRG/BEND	程序块开始 / 程序块结束
STEP S/STEP	定义步进程序

中断任务中不允许使用的指令

下列指令不允许在中断任务中出现，任何企图在中断任务中使用下列指令都将导致 ER 标志位变为 ON，且指令停止执行。但如果中断任务作为附加循环任务使用时，则可以使用下列指令。

助记符	指令
TKON(820)	任务开启
TKOF(821)	任务关闭
STEP	定义步进
SNXT	定义下一步
STUP	更改串口设置
DI	使中断禁止
EI	使中断有效

在中断任务中，下列指令的操作结果是不确定的：定时器：TIM 和 TIMX(550)，高速定时器：TIMH(015) 和 TIMHX(551)，一毫秒定时器：TMHH(540) 和 TMHHX(542)，累计定时器：TTIM(087) 和 TTIMX(555)，多输出定时器：MTIM(543) 和 MTIMX(544)，长时间定时器：TIML(542) 和 TIMLX(553), 定时器等待：TIMW(813) 和 TIMWX(816)，高速定时器等待：TMHW(815) 和 TMHWX(817)， PID 控制：PID(190)，错误点检测：FPD(269)，更改串行口设置：STUP(237)。

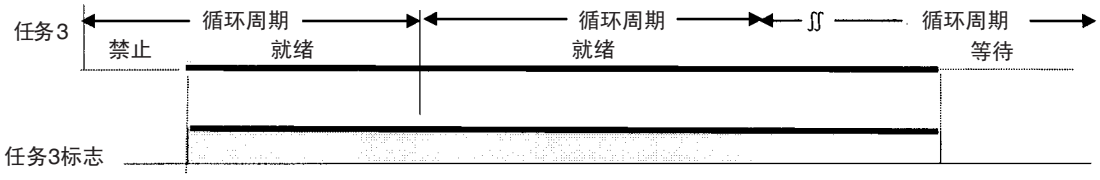
下列指令不能在断电中断任务中使用 （即使使用也将不执行，但出错标志位不会置 ON）。
读数据文件: FREAD(700)，写数据文件: FWRIT(701)，网络发送: SEND(090)，网络接收：RECV(098)，发送命令：CMND(490)，协议宏指令：PMCR(260)。

4-2-3 与任务相关的标志

循环任务相关标志

下列标志位仅在常规循环任务中工作，不能在附加循环任务中工作。

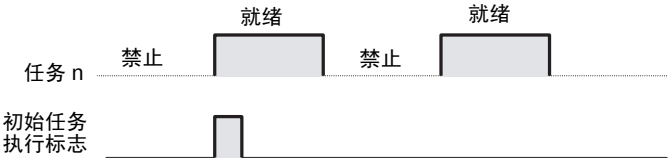
任务标志 （TK00 ～ TK31） 任务标志位在循环任务处于：“就绪”状态时置 ON，在任务处于 “禁止”状态(INI)或“等待”状态时置OFF。标号0～31的任务相应标志为TK00～TK31。



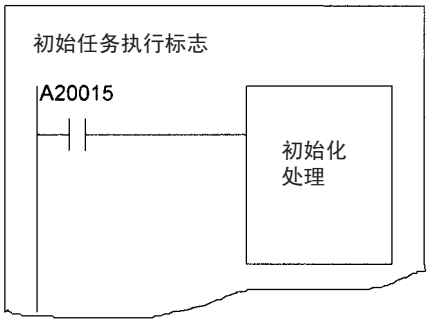
注 任务标志仅能在循环任务中使用，不能在中断任务中使用。就一个中断任务而言，如果在运行开始后执行一个中断任务，A44115 将变为 ON，并且需要最大处理时间的那个中断任务标号将以两位十六进制数保存在 A44100 ～ A44107 中。

初始任务执行标志（A20015）

当循环任务从“禁止”状态切换到“就绪”状态时，初始任务执行标志变为 ON，任务获取执行权，并第一次执行。在任务第一次执行完毕后标志位变为 OFF。



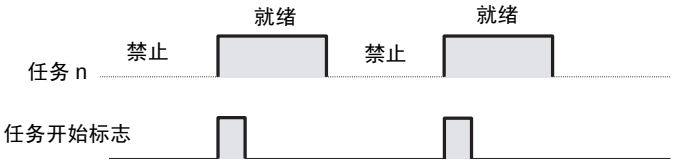
根据初始任务执行标志位可知循环任务是否是第一次执行。当然，这个标志位同时可用来进行任务的初始化处理。



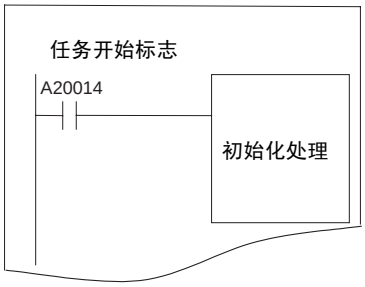
注 尽管使用 TKON(820) 指令可使处于“等待”状态下的循环任务切换回“就绪”状态，但这不能被视为初始执行，初始任务执行标志位 （A20015）不会变为 ON。如果循环任务被从 “运行”状态切换到 “禁止”状态或在获取执行权前被其他任务用 TKOF （821）指令设为 “等待”状态时，初始任务执行标志（A20015）也不变为 ON。

任务开始标志 （A20014）（仅限于 CS1-H,CJ1-H,CJ1M 或 CS1D CPU 单元）

在每次任务周期开始时，任务开始标志位可用来执行初始化处理。无论何时，只要循环任务从 “禁止”状态 （INI）或 “等待”状态切换为 “就绪”状态，其任务开始标志位都将变为 ON。（然而，初始任务执行标志位只有在任务状态从 “禁止”状态 （INI）变为 “就绪”状态时才变为 ON）。



无论何时，只要任务从 “等待”状态变为 “运行”状态，任务开始标志都可用来执行初始化处理，也就是当任务处于 “等待”状态时使用 TKON （820）指令激活任务。



与所有任务相关的标志

任务出错标志 (A29512)

如果发生下列任务错误时，任务出错标志位变为 ON。

- 在一个周期内没有循环任务 （包括附加循环任务）处于 “就绪”状态。
- 分配给一个循环任务 （包括附加循环任务）的程序不存在 （在使用 CX-Programmer 或手持编程器时这种情况不会发生）。
- 处于激活状态的中断任务无指派程序。

程序停止时的任务标号（A294）

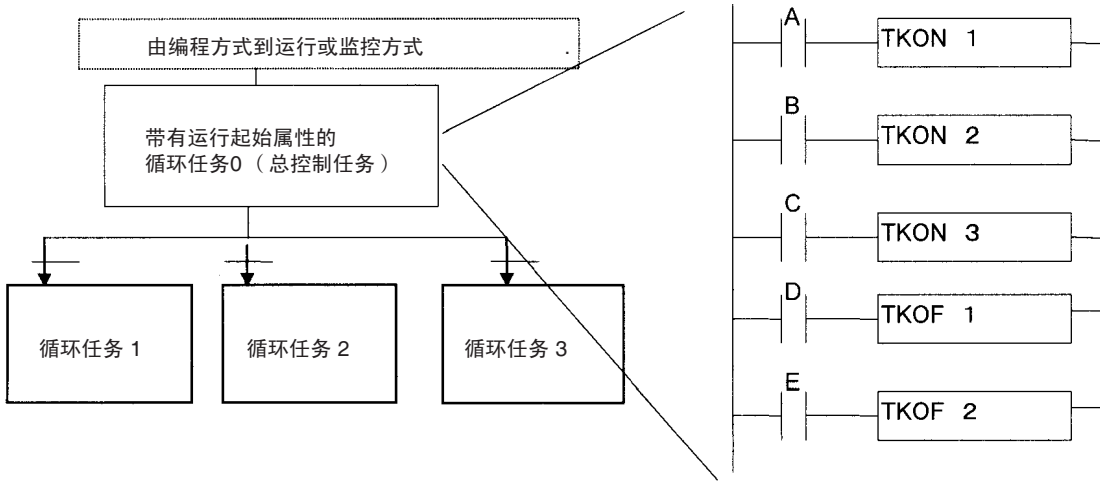
当任务由于程序错误而停止执行时，任务的类型和任务的当前编号将以如下方式保存：

类型	A294
循环任务	0000H ～ 01FH （相应任务标号：0 ～ 31）
中断任务	8000H ～ 80FFH （相应中断任务标号：0 ～ 255）

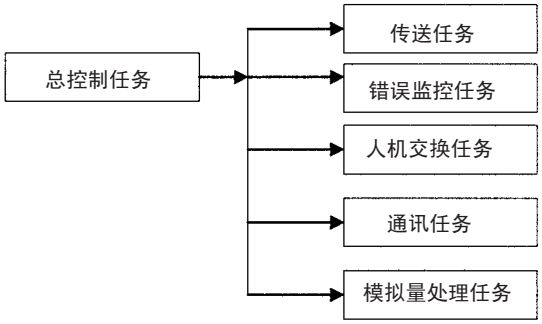
这个信息使得用户可以很容易地判断出在何处发生了致命错误，在致命错误消除后，信息将被清除。任务运行终止处的程序地址保存在 A298 （程序地址的低位 （右边位））和 A299 （程序地址的高位 （左边位））中。

任务举例

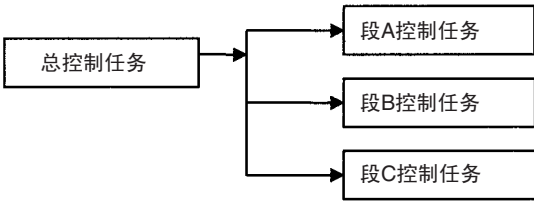
在运行开始时，一个被设为 “就绪”状态的总控制任务通常用来控制其它循环任务 （包括附加循环任务）为 “就绪”或 “等待”状态。当然，任何循环任务在应用需要时也能用来控制其它循环任务为 “就绪”或 “等待”状态。



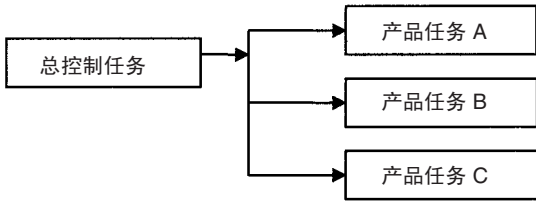
按功能任务分割



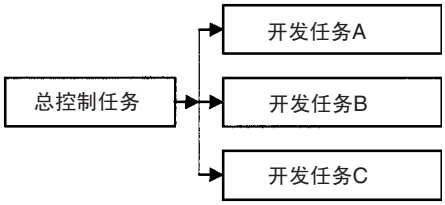
按可控环节任务分割



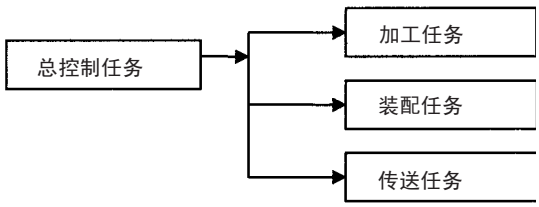
按产品任务分割



按开发任务分割



按处理任务分割



也可以把以上划分方式结合起来分割任务，如按功能和处理方式分割任务。

4-2-4 设计任务

我们建议按如下步骤设计任务。

1,2,3... 1. 使用如下标准考虑分割任务。

a) 归纳出执行与不执行的特定条件；

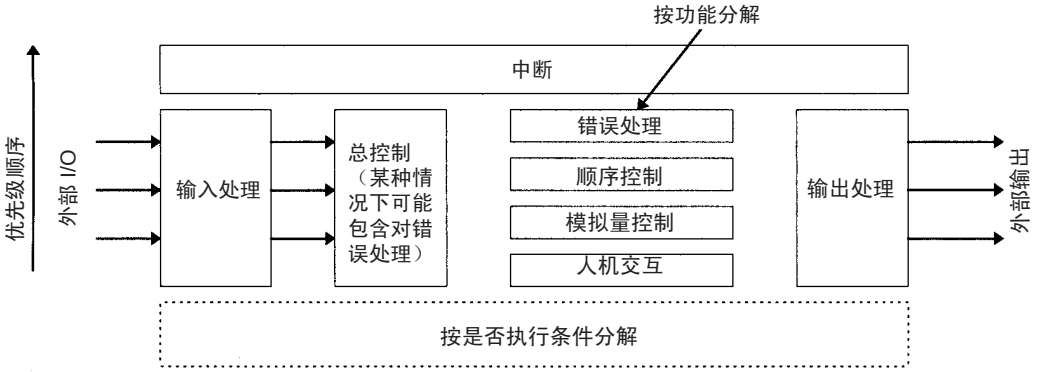
b) 归纳出有无外部 I/O 的情况

c) 概括功能

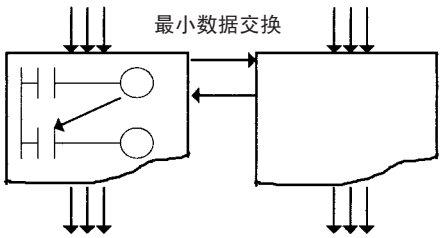
在任务之间为顺续控制、模拟量控制、人机交互、错误处理和其它处理进行数据交换时使其数据量为绝对最小以保证任务间高度的独立性。

d) 归纳出执行的优先级顺序。

把处理分成循环任务和中断任务。



2. 确保以一种能保证独立方式分解和设计程序，并使任务间 （程序）交换的数据量为绝对最小。

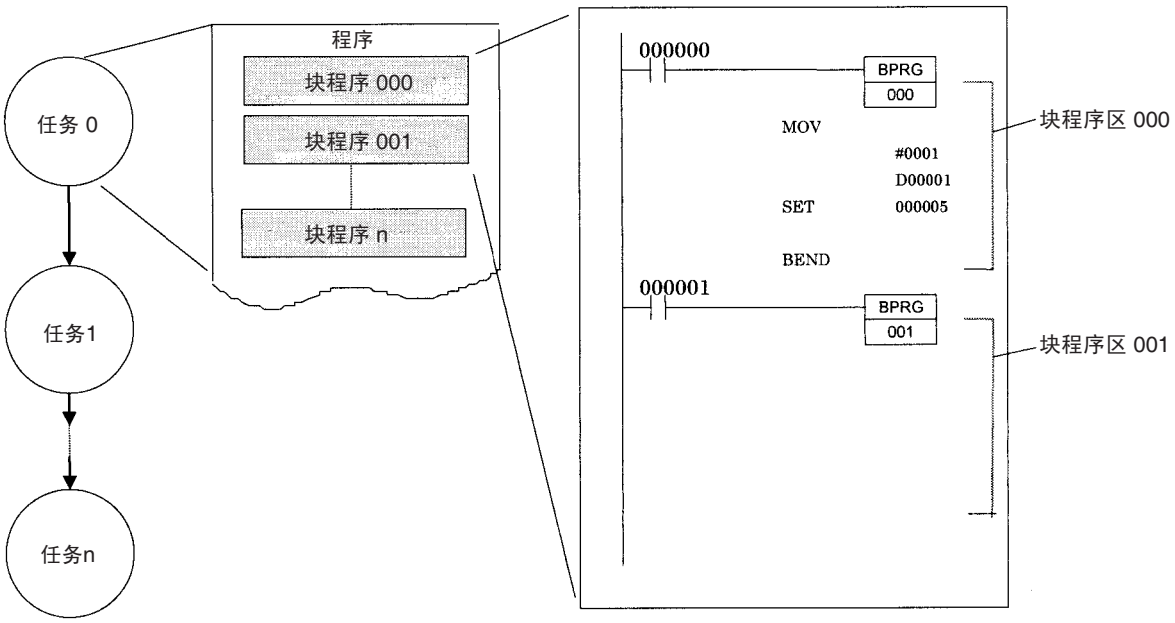


3. 一般情况下，使用一个总控制任务去控制其它任务为“就绪”/“等待”状态。
4. 给优先级最高的任务分配最低的任务标号。
例：给控制任务分配的标号比处理任务的低。
5. 给高优先级的中断任务分配较低的标号。
6. 一个处于“就绪”状态下的任务只要它本身或其它任务没有把它切换为“等待”状态，那么它将在随后的周期中执行。如果处理在任务之间将被分支的话，必须为其它任务插入一个 TKOF(821) 指令。
7. 在执行指令的执行条件中使用初始任务执行标志（A20015）或任务开始标志 （A20014）来初始化任务。
8. 把I/O存储单元分为被所有任务共享的存储单元和仅供独立任务使用的存储单元， 然后，把仅供独立任务通过任务方式使用的 I/O 存储器编组。

任务与块程序的关系

对所有任务而言，在任务中最多可生成 128 个块程序。每个完整的块程序由梯形图控制，但块程序中的指令用助记符表示。换句话说，一个块程序是以梯形图指令和助记符代码相结合的形式表示的。

使用块程序使得写逻辑流程，如条件分支和处理步进变得容易，而用梯形图表示时就可能比较困难。块程序放置在程序层的底部，并且代表任务的较大程序单元可分割成几个像块程序那样的小程序单元，这些单元在相同的执行条件（ON 条件）下运行。



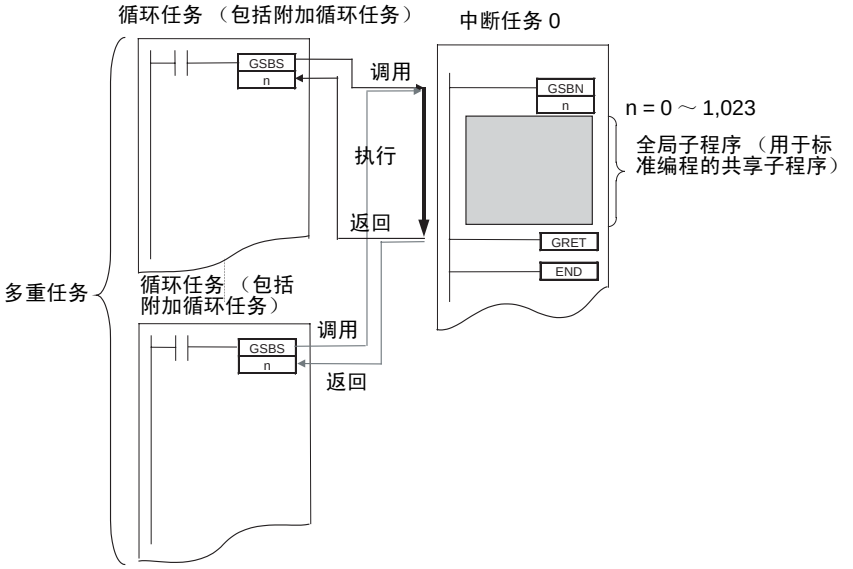
4-2-5 全局子程序

全局子程序可被多个任务调用。他们仅被CS1-H,CJ1-H,CJ1M或CS1D CPU支持。

就 CS1 或 CJ1 CPU 而言，某个任务内的子程序不能被其它任务调用。然而，就 CS1-H,CJ1-H,CJ1M 或 CS1D CPU 而言，在标号为 0 的中断任务中可生成全局子程序，并且这些子程序可被循环任务 （包括附加循环任务）调用。

可用 GSBS 指令来调用一个全局子程序。子程序标号必须在 0 ～ 1023 之间。在标号为 0 的中断任务末尾（就在 END(00) 指令之前）的 GSBN 指令和 GRET 指令之间定义全局子程序。

全局子程序可用来生成一个在任何需要的时候都可调用的标准程序库。



4-3 中断任务

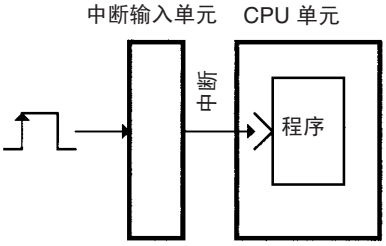
4-3-1 中断任务类型

在一个周期内的任何时候，只要下列条件中的任意一个有效，就执行中断任务。

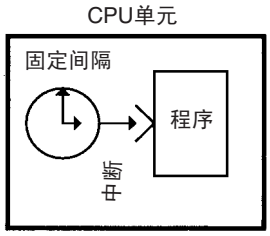
CJ1M CPU 单元上的内置中断输入和高速计数器输入可用来激活中断任务。详情请见有关 *CJ 系列内置 I/O 口*操作手册。

注 CS1D CPU 不支持中断。就 CS1D CPU 而言，中断任务只能被当作附加循环任务使用。

I/O 中断 （仅限于 CS 系列） 在中断输入单元的输入为 ON 时执行 I/O 中断任务。

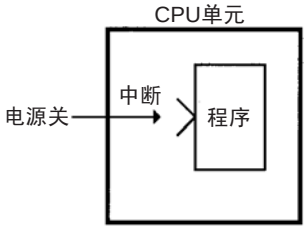


定时中断任务 在固定时间间隔执行定时中断任务。



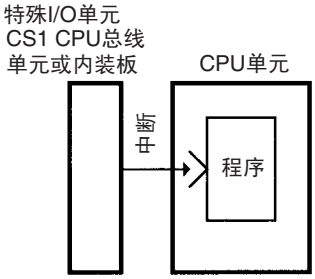
断电中断 当电源关闭时执行断电中断任务。

注 断电中断任务的执行时间必须少于 10ms （断电延迟检测时间）。



外部中断 （仅适用于 CS 系列）

当一个特殊 I/O 单元， CPU 总线单元或内装板 （仅限于 CS 系列）要求中断时，执行外部中断任务。然而，特殊 I/O 单元和 CPU 总线单元必须安装在 CPU 机架上才可要求执行外部中断任务。



中断任务列表

类型	任务标号	执行条件	设置过程	中断数	应用举例
I/O 中断 00 ～ 31	100 ～ 131	CPU 机架上的中断输入单元输入为 ON （见注 1）	使用 MSKS （设定中断）指令为 CPU 机架上的中断输入单元分配输入	32 点	提高特定输入的响应速度
定时中断 0 和 1	2 和 3	定时 （固定时间间隔）	使用 MSKS （设定中断）指令设定中断间隔。见 PLC 设置中的定时中断时间单位	2 点	在固定时间间隔监控运行状态
断电中断	1	当断电时 （在缺省断电检测时间＋电源关断检测延迟时间后）	见断电中断任务和电源关中断检测延迟时间	1 点	处理电源关闭时的突发事件
外部中断 0 ～ 255	0～255	当 CPU 机架上的特殊 I/O 单元与 CPU 总线单元或内装板要求中断时 （见注 2）	无 （一直有效）	256 点	执行特殊 I/O 单元， CPU 总线单元或内装板要求的处理

注 1. 中断输入单元必须安装在 CPU 机架上，就 CJ1-H CPU 而言，中断输入单元必须作为紧靠 CPU 的五个单元 （槽 0 ～ 4）中的一个与 CPU 相连。就 CJ1M CPU 而言，中断输入单元必须作为紧靠 CPU 的三个单元 （槽 0 ～ 2）中的一个与 CPU 相连。I/O 中断单元若被安装在其它地方，将不能被用来要求 I/O 中断任务的执行。

2. 特殊的 I/O 单元和 CPU 总线单元必须被安装在 CPU 机架上，就 CJ1-H CPU 而言，中断输入单元必须作为紧靠 CPU 的五个单元（槽 0 ～ 4）中的一个与 CPU 相连。就 CJ1M CPU 而言，中断输入单元必须作为紧靠 CPU 的三个单元 （槽 0 ～ 2）中的一个与 CPU 相连。中断单元若被安装在其它地方，将不能用来产生外部中断。

3. CJ1 CPU 不支持 I/O 中断和外部中断任务。

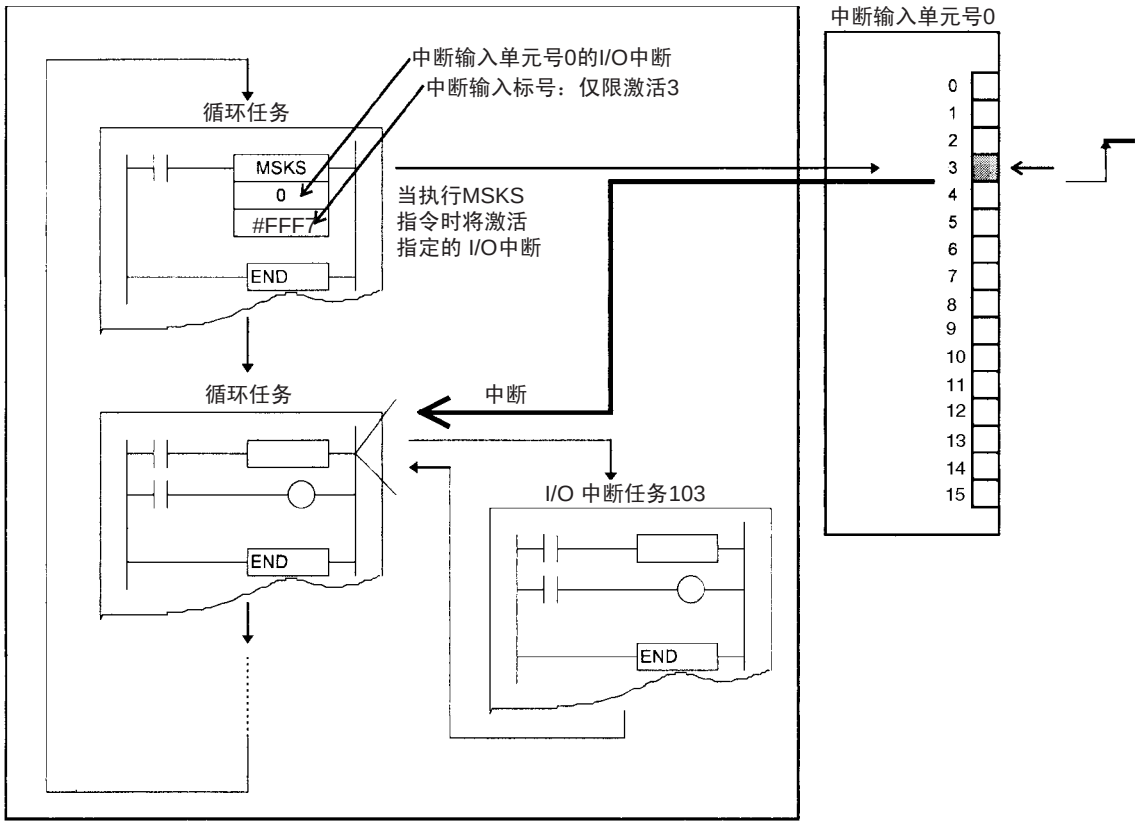
4. CS1D CPU 不支持中断。就 CS1D CPU 而言，中断任务只能作为附加循环任务使用，即无任何其他中断可用。

I/O 中断任务：任务 100 ～ 131

在循环任务开始执行时， I/O 中断任务的缺省状态为 “禁止”状态。为了激活 I/O 中断，要在循环任务中对中断输入单元的中断标号执行 MSKS（设定中断）指令。

例：下面例子演示了单元号为 0 的中断输入单元 （0 号和 1 号两个单元中的左面一个单元）的中断输入 3# 为 ON 时执行 I/O 中断任务 103。

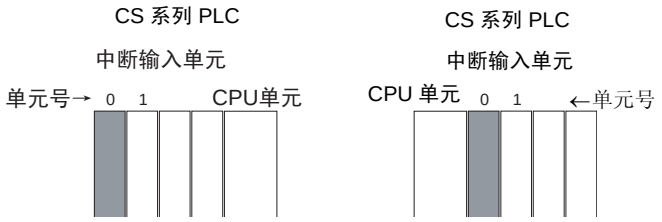
注 不要激活不需要的 I/O 中断任务。如果中断输入被噪音触发且无相应的中断任务可执行时，将导致一个致命错误 （任务错误）并使程序终止。



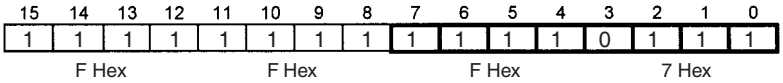
中断输入单元号，输入号和 I/O 中断任务号

中断输入单元号（见注）	输入标号	I/O 中断任务
0	0 ～ 15	100 ～ 115
1	0 ～ 15	116 ～ 131

注 对于 CS 系列 PLC 而言，中断输入单元号从 CPU 机架上的左边开始，从 0 到 1 排列。而对于 CJ 系列 PLC 来说，中断输入单元号从 CPU 单元开始，从 0 到 1 排列。



MSKS 指令的 S 操作数（第二个操作数）：十六进制 FFF7 中的各位相应于中断输入单元的中断输入。中断输入 0 ～ 15 相应于位 0 ～ 15。



定时中断任务：任务 2 和 3

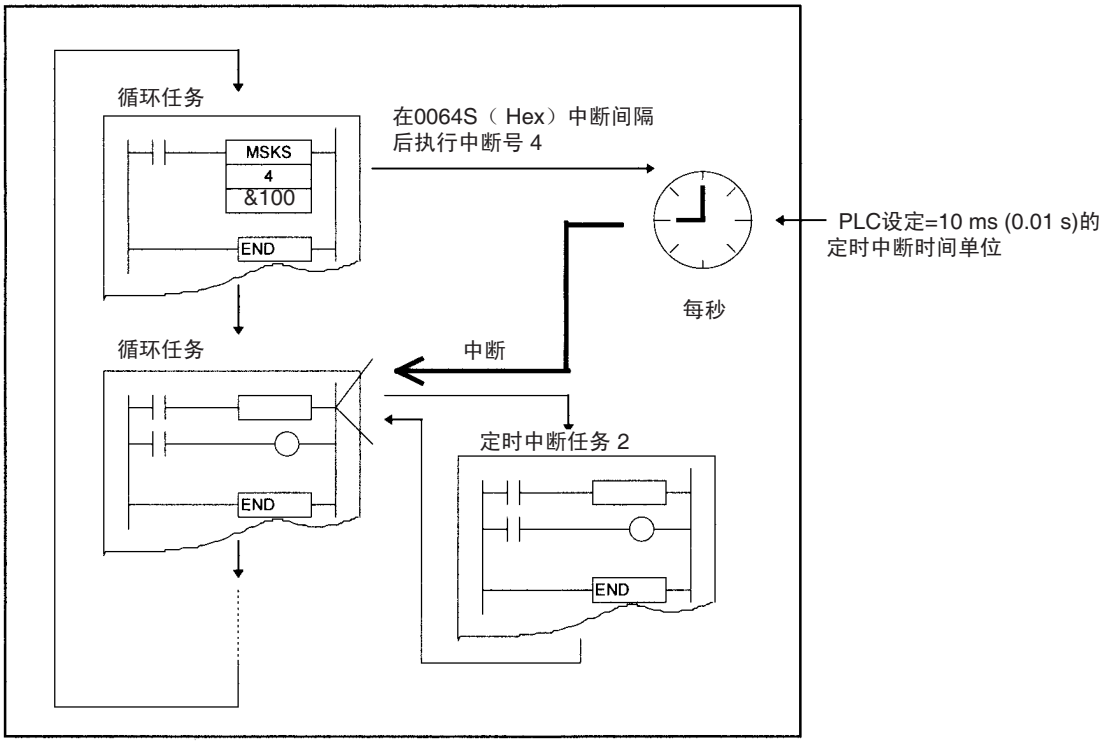
在循环任务开始执行时，定时中断任务在 PLC 设置中的缺省状态为“禁止”状态。执行下面步骤可激活定时中断任务。

1,2,3... 1. 在循环任务中执行 MSKS（设定中断）指令并设定特定预定中断的时间。

2. 在 PLC 设置中设定预定中断时间单位。

注 中断时间的设置值对循环任务有影响，因为中断时间越短，任务执行就越频繁，循环时间就越长。

例：下面例子演示了每隔一秒执行一次定时中断任务 2。



中断号和定时中断任务号

中断号	预定中断号
4	2
5	3

PLC 设置值

地址	名称	描述	设置值	缺省值
195 字中的 0 ～ 3 位	定时中断时间单位	设定定时中断的时间单位，在固定间隔后执行中断任务	00 Hex: 10 ms 01 Hex: 1.0 ms 02 Hex: 0.1 ms（仅限于 CJ1M CPU）	00 Hex

断电中断任务：任务 1

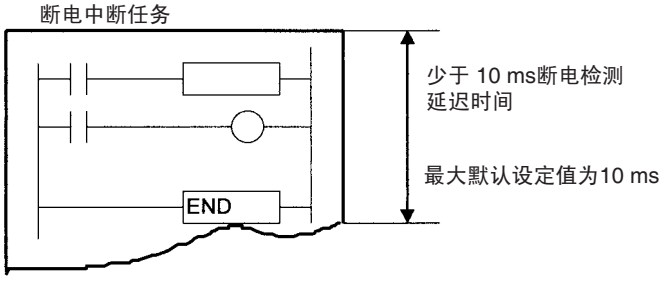
在循环任务开始执行时，断电中断任务在 PLC 设置中的缺省状态为“禁止”状态。

在 PLC 设置中，断电中断任务可激活。

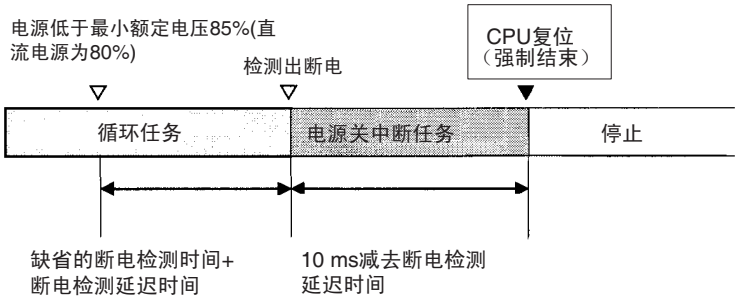
在 PLC 的缺省设置中，断电中断任务在 10ms 后停止执行。即断电中断任务的执行时间必须少于 10ms.

如果在 PLC 设置中设定断电检测延迟时间，那么断电中断任务将在 10ms 减去在 PLC 设置中设定的断电检测延迟时间后停止。在这种情况下，断电中断任务的执行时间必须少于 10ms 减去 PLC 设置中设定的断电检测延迟时间。

例：如果在 PLC 设置中设定的断电检测延迟时间为 4ms，那么执行时间必须少于 10ms ～ 4ms，即 6ms。

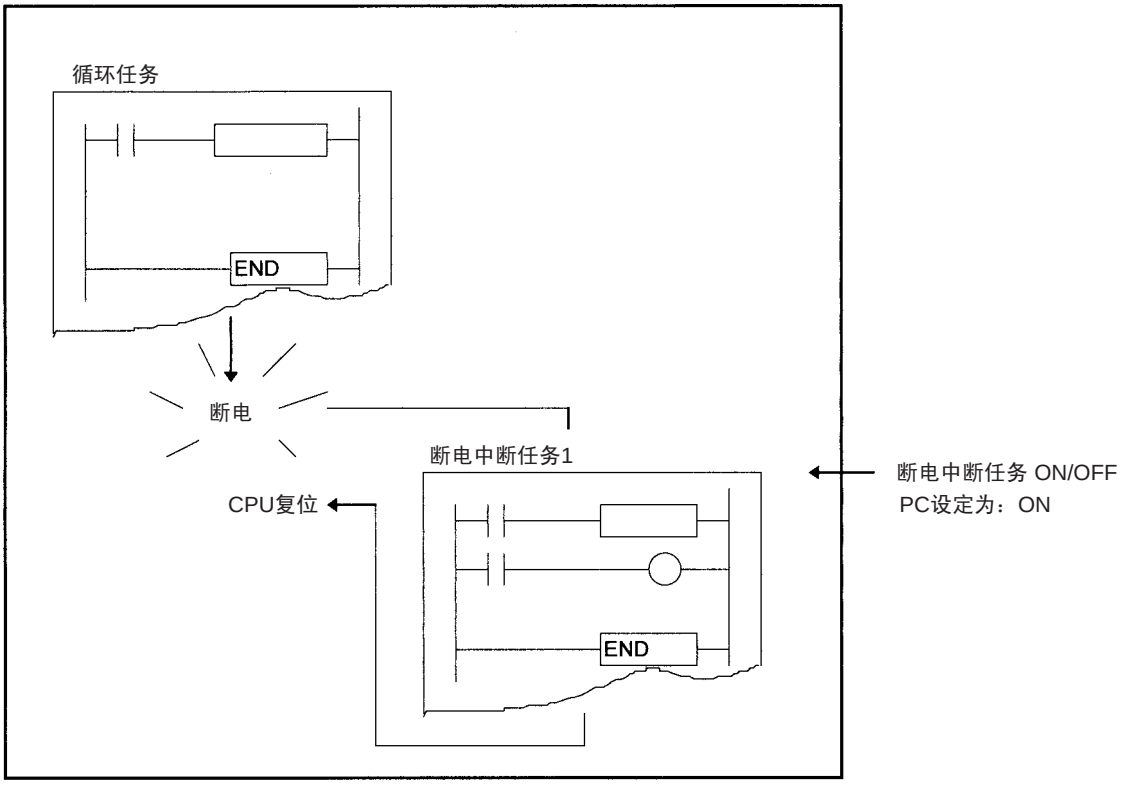


注 当电源供给低于额定电压的 85%（直流电源为 80%）时，认为断电条件，而且断电中断任务真正执行前所用的时间为缺省的断电检测时间（交流电源为 10ms ～ 25ms，直流电源为 2 ～ 5ms 时）加上在 PLC 设置中设定的断电检测延迟时间（0 ～ 10ms）。在这段时间内将执行循环任务。



注 必须确保断电中断任务能在少于 10ms 减去在 PLC 设置中设置的断电检测延迟时间内执行完毕。这段时间消逝后，任务中剩余的指令将不执行。在线编辑过程中，如果断电，断电中断任务不执行。此外，一些指令不可在任何中断任务中使用（详情请见相关编程手册），下列指令不可在断电中断任务中使用：读数据文件：FREAD(700)，写数据文件：FWRIT(701)，网络发送：SEND(090)，网络接收：RECV(098)，传送命令：CMND(490)，传输：TXD(236)，接收：RXD(235)，以及协议宏指令：PMCR(260)。

断电中断任务执行过程



PLC 设置中断电中断任务 （任务号：1）的设定

地址	名称	描述	设定值	缺省值
＋ 255 字中的 15 位	断电中断任务	如果断电，＋ 225 字中的 15 位为 ON，这时开始执行断电中断任务。	0: OFF, 1: ON	0
＋ 255 字中的 0 ～ 7 位	断电检测延迟时间	在断电确认时间加上缺省的断电检测时间 （交流为 10 ～ 25ms，直流为 2 ～ 5ms）后认为断电	00～0A Hex: 0 ～ 10 ms （单位：1ms）	00 Hex

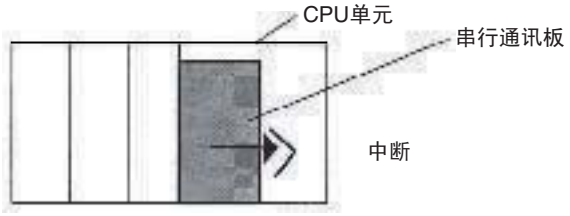
外部中断任务：任务 0 ～ 255

在任何时候都可接收外部中断任务。

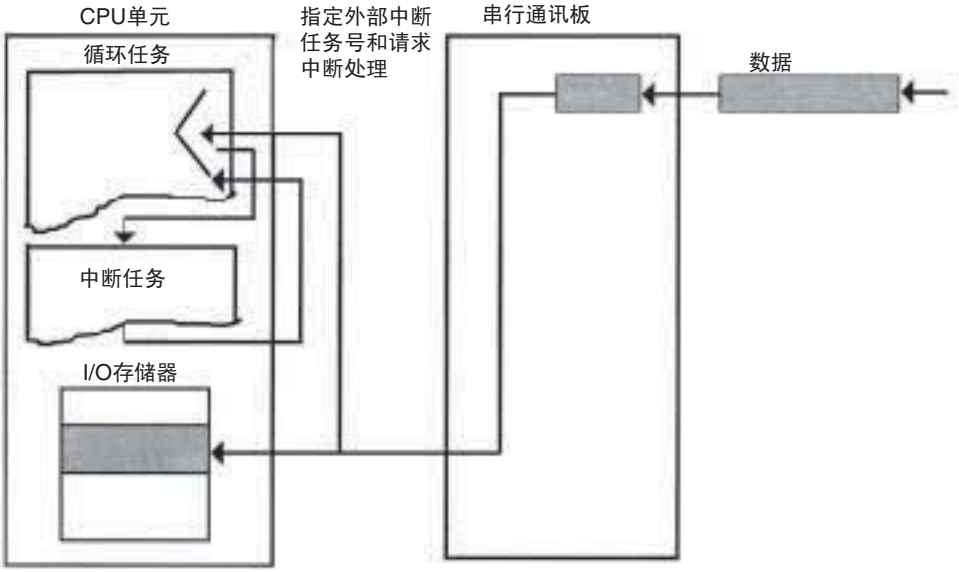
当 PLC 的 CPU 单元包含内装板（仅限于 CS 系列），特殊 I/O 单元或 CPU 总线单元时可执行中断处理。除非程序包含某个特定任务号的外部中断任务，否则不必在 CPU 单元中作设置。

CJ1 CPU 不支持外部中断。

例：下面例子演示了 CS1W-SCB@1 串行通讯板生成一个外部中断。



当串行通讯板的响应通知方式设为中断通知（固定号）或中断通知（接收事件号）时，通讯板在从串口接收到数据并把数据写入 CPU 单元的 I/O 存储区后要求执行一个 CPU 单元内的外部中断任务。



注 1. 当响应通知方式设为中断通知 （固定号）时，通讯板要求执行任务号为预先设定的中断任务。

2. 当响应通知方式设为中断通知 （接收事件号）时，外部中断任务号通过特定的公式计算得出，并且通讯板要求执行任务号为计算得到的任务号的那个中断任务。

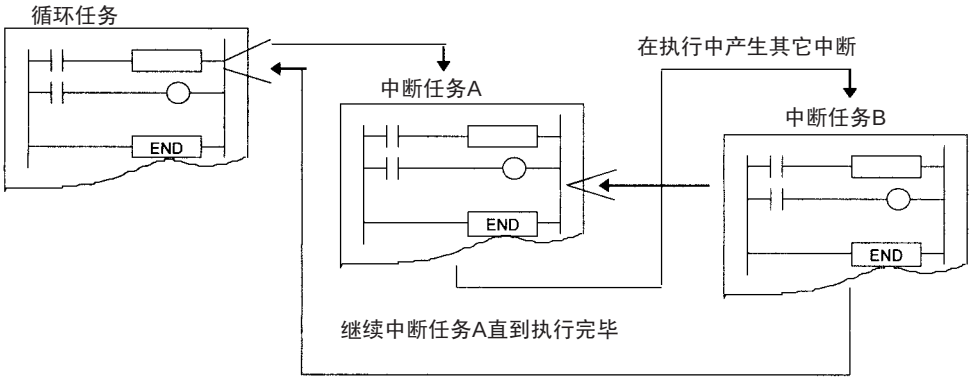
3. 如果外部中断任务（0 ～ 255）与断电中断任务（任务 1），定时中断任务（任务 2 或 3），或 I/O 中断任务 （100 ～ 131）有相同的号时，中断任务将在两个条件（外部中断条件和其它中断条件）中的任何一个满足时执行。因此规定，任务号不可重复。

4-3-2 中断任务优先级

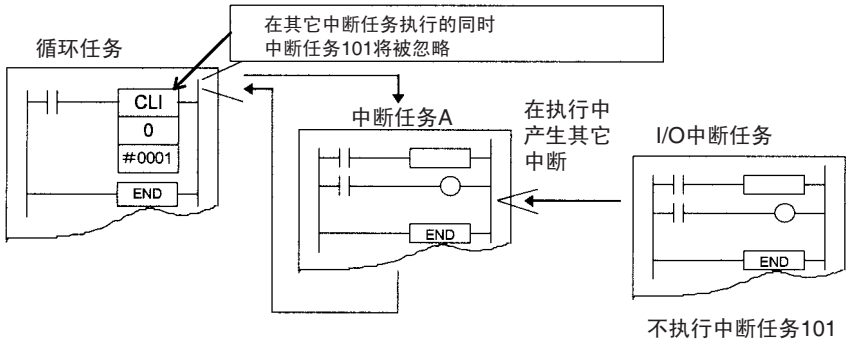
在断电中断任务允许执行时，其它中断任务都将停止执行。在断电中断任务执行完毕后， CPU 复位，被中止的中断任务不再执行。

中断任务执行过程中的中断

如果当一个中断任务正在执行时发生另一个中断，那么这个中断的相应任务将在原先的中断任务执行完毕后才被执行。



注 对于 CS 系列 CPU 而言，当一个中断任务正在执行时发生一个特殊 I/O 中断申请，如果你不想保存和执行这个特殊 I/O 中断任务，可在正在执行的中断任务中执行 CLI（清除中断）指令清除内部自动保存的中断号。但不可取消定时中断和外部中断。



多个中断申请同时出现

除断电中断任务外，无论什么时候，如果多个中断申请同时出现，那么中断任务将按下面的优先级顺序依次执行。

I/O 中断任务（仅限于 CS 系列）> 外部中断任务（仅限于 CS 系列）> 定时中断任务。

在各种类型的中断任务内，如果产生多个同类型中断任务，那么将从标号最低的中断任务开始依次执行。

注 对于所有中断任务而言，仅有一个中断被存储器记录，并且已经在执行的中断不被存储器记录。由于定时中断的优先级最低并且在每个时刻仅能记录一个中断，因此定时中断任务很有可能被跳过而不执行。

4-3-3 中断任务标志和字

最大的中断任务处理时间 （A440）

最大的中断任务处理时间是按二进制数据格式以 0.1ms 为单位存储的，并在运行开始时清除。

处理时间最大的中断任务 （A441）

处理时间最大的中断任务号以二进制数据格式保存。这里，8000H ～ 80FFH 存放对应任务标号 00H ～ FFH。

在运行开始后的第一个中断发生时，A44115 变为 ON。随后的最大中断任务处理时间将以两位十六进制数保存在最右位（低位）中，并且在开始运行时清除。

中断任务出错标志 （非致命错误）（A40213）

如果在 PLC 设置时，中断任务出错检测功能置 ON，当发生中断任务错误时，中断任务出错标志变为 ON。

中断任务出错标志 （A42615） / 产生中断任务出错的任务号 （A42600 ～ A42611）

如果 A40213 为 ON，那么下列数据将保存在 A42615 和 A42600 ～ A42611 中。

A40213	中断任务出错描述	A42615	A42600 ～ 42611
中断任务出错 （如果在 PLC 设置时，中断任务错误检测功能置 ON）	在 C200H 特殊 I/O 单元或 SYSMAC 总线远程 I/O 口刷新过程中，中断任务执行时间超过 10ms （仅限于 CS 系列）	OFF	中断任务号以 12 位二进制数保存（中断任务 0 ～ 255：000H ～ 0FFH）
	在特殊 I/O 单元通过循环 I/O 刷新方式刷新时，试图在中断任务中用 IORF 指令刷新大量 I/O 字	ON	正被刷新的特殊 I/O 单元的单元号以 12 位二进制数保存 （单元号 0 ～ 95：000H ～ 05FH）

程序停止时的任务号（A294）

由于程序错误而使程序停止，此时当前任务号和任务类型将保存在下面区域。

类型	A294
中断任务	8000H ～ 80FFH （相应于中断任务 0 ～ 255）
循环任务	0000H ～ 001FH （相应于任务 0 ～ 31）

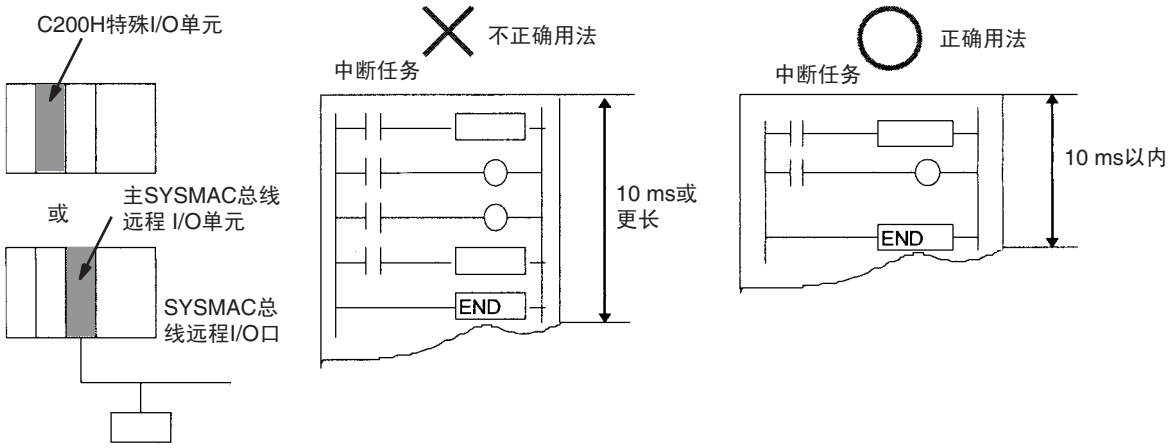
4-3-4 应用注意事项

C200H 特殊 I/O 单元或 SYSMAC 总线长时间执行（仅限于 CS 系列）

当使用 C200H 特殊 I/O 单元或 SYSMAC 总线远程 I/O 单元时必须保证所有中断任务 （I/O，定时，断电，和外部中断任务）在 10ms 内执行。

在 C200H 特殊 I/O 单元或 SYSMAC 总线远程 I/O 口刷新过程中，如果中断任务执行时间超过 10ms，将产生一个中断任务错误， A40206（特殊 I/O 单元出错标志）变为 ON，并且特殊 I/O 单元的 I/O 口刷新停止。然而， CPU 单元将继续运行。

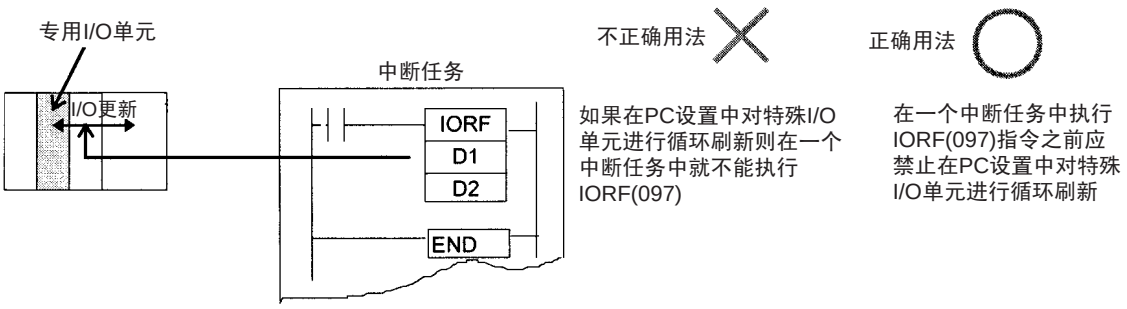
如果在 PLC 设置中，中断任务错误检测功能置 ON，当发生中断任务错误的话，A40213 将变为 ON，并且发生错误的那个中断任务号将保存到 A426（中断任务错误，任务号）中去。然而， CPU 单元将继续运行。



对特殊 I/O 单元执行 IROF 指令

如果必须要在一个中断任务中对某个特殊 I/O 单元执行 IROF （097）指令，那么必须保证在 PLC 设置中，对特殊 I/O 单元进行循环刷新置 OFF （使用单元标号）。

当单元正在使用循环 I/O 刷新方式，或者 I/O 刷新指令 （IROF （097）或者立即刷新指令（！））进行刷新时，如果试图在一个中断任务中使用 IROF（097）指令刷新特殊 I/O 单元，将发生一个中断任务错误。在中断任务错误发生时，如果在 PLC 设置中，中断任务错误检测功能置 ON， A40213 （中断任务出错标志）将变为 ON，并且复制正在进行 I/O 刷新的特殊 I/O 单元的单元标号并把它保存在 A426 （中断任务错误，任务号）中。 CPU 单元继续运行。



注 A426 （中断任务错误，任务号）的最左位 （高位）可用来判定发生了上面中断任务错误中的那一个。（位 15：如果为 0，则代表 10ms 或更长的执行时间错误；如果为 1，则代表多重刷新错误）

PLC 设置值

地址	名称	描述	设定值	缺省值
＋ 128 中的位 14	中断任务错误检测	规定是否进行错误检测。当检测功能被激活，中断错误标志 （A40213）将起作用	0: 激活检测 1: 不激活检测	0

相关辅助区标志 / 字

名称	地址	描述
中断任务错误标志	A40213	在 C200H 特殊 I/O 单元或 SYSMAC 总线远程 I/O 口刷新过程中，如果中断任务执行时间超过 10ms， A40213 变为 ON，但 CPU 将继续运行。 CPU 单元前面面板上的 ERR/ALM 发光二极管亮。（仅限于 CS 系列） 在特殊单元使用循环 I/O 刷新方式进行刷新过程中，如果试图在某个中断任务中用 IROF 对特殊 I/O 单元进行刷新时 A40213 变为 ON。
中断任务错误，任务号	A426	所包含的中断任务号或特殊 I/O 单元号被刷新时。 （当中断任务要求的执行时间为 10ms 或更长时，位 15 为 OFF，当重复刷新特殊 I/O 单元发生后，位 15 为 ON）。

使中断无效

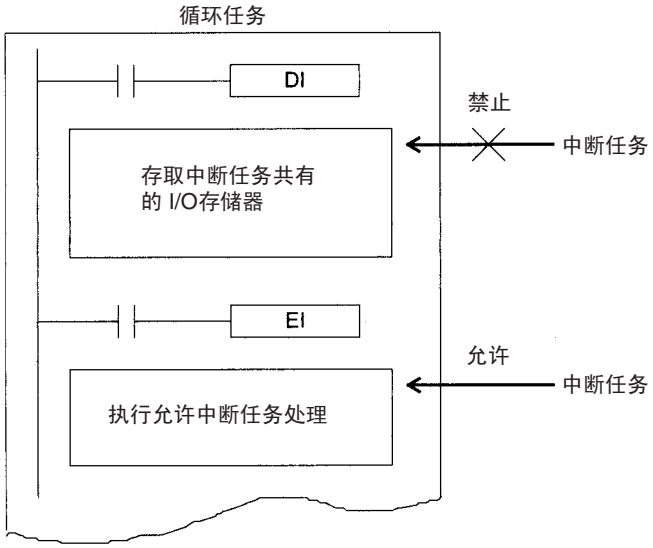
在下列情况下，处理中止并执行中断任务。

- 当某个指令执行时
- 当基本 I/O 单元，CPU 总线单元，内装板（仅限于 CS 系列），或 SYSMAC 总线远程 I/O 单元刷新时
- 当上位链接服务时

循环和中断任务间的数据一致性

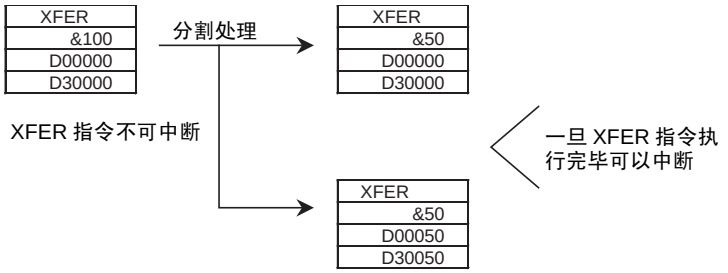
如果循环 （包括附加循环任务）和中断任务对同一 I/O 存储地址进行读写，数据可能不一致性。在循环任务指令存取存储器过程中，执行下列步骤使中断禁止。

- 在用一个循环任务指令进行读写前，立刻使用 DI （中断禁止）指令时中断任务不能执行。
- 在处理完成后立刻使用 EI （中断允许）指令，使中断任务有效。



在要求响应接收与处理的指令 （如网络指令或串行通讯指令）执行过程中，即使用了 DI （693）和 EI （694）指令使中断任务禁止，仍然有可能发生数据一致性的

注 就 CS1-H， CJ1-H， CJ1M 或 CS1D CPU 而言，不能中止位计数 （BCNT），块置位 （BSET），和块传送 （XFER）指令的执行去执行中断任务，即推迟中断响应。在中断任务执行前完成这些指令的执行。为防止这种情况，必须把这些指令的数据处理分成几个部分，下面演示了 XFER 的分割。



4-4 任务的编程工具操作

4-4-1 使用多个循环任务

使用 CX- Programmer 可生成多个循环的任务（包括附加循环任务）。手持编程器不可用来生成新的循环任务。必须保证用 CX- Programmer 为已经生成的程序分配任务类型和任务号。

- 由 CX- Programmer 生成并向 CPU 单元传送的多个循环任务可通过手持编程器进行监控和编辑。
- 手持编程器可通过使用其中的全部清空功能和指定中断任务生成一个循环任务和一个或多个专用中断任务。手持式编程器仅能用来生成中断任务 1（断电中断），2 和 3（定时中断），以及 100 ～ 131（I/O 中断）。然而对于 CJ1M CPU 来说，中断任务 140 ～ 143（为内置输入分配的）也能通过手持式编程器生成。在 PLC 运行开始时启动循环任务 0。

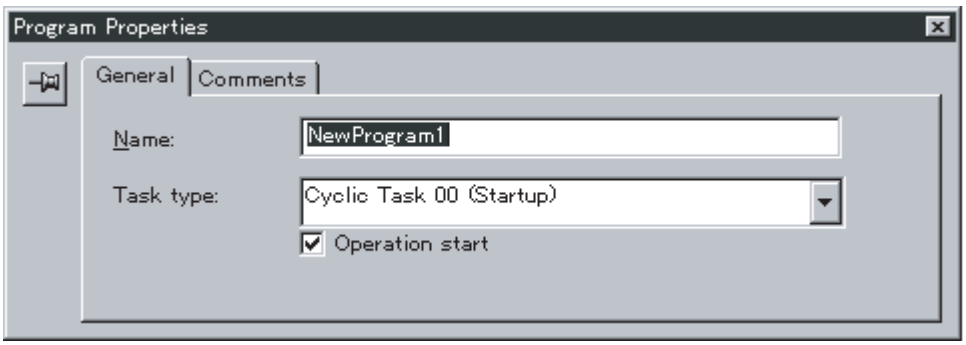
4-4-2 编程工具操作

CX-Programmer

规定每个程序的任务类型和任务号，并定义其属性。

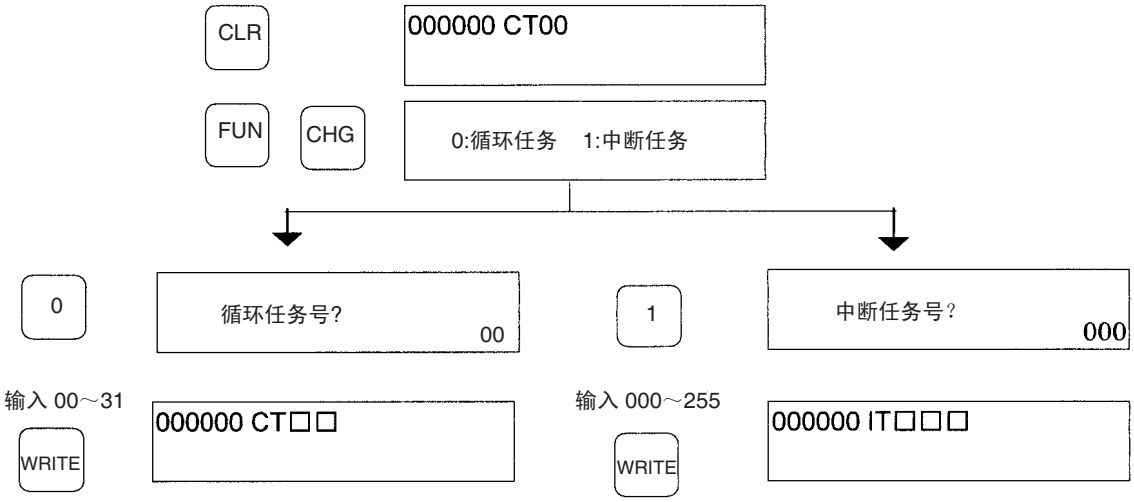
1,2,3...

1. 选择 *View/Properties*，或者单击右键并在弹出菜单上选择 Properties, 显示将要分配给任务的程序。
2. 选择 General 标签，并选择其中的 Task Type 和 Task No 项。对循环任务而言，单击 Operation start 检查框把它置 ON。



手持编程器

手持编程器把任务作为完整的程序进行操作。使用手持编程器通过指定给循环任务的 CT00 ～ CT31 或给中断任务的 IT001 ～ IT255 编号来访问和编辑程序。



注 1. 手持编程器不能生成新的循环任务。

2. CJ 系列 CPU 目前不支持 I/O 中断和外部中断。因此仅有 IT001 ～ IT003 能被指定。

第 5 章
文件存储器功能

本章主要描述使用文件存储器的操作功能。

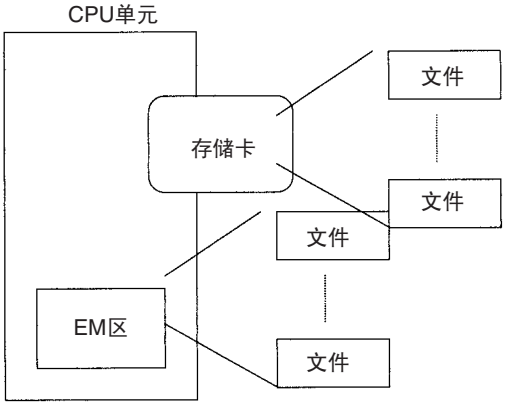
5-1 文件存储器

CS/ CJ 系列 CPU 支持文件存储。下列的介质可用于储存文件的存储器。

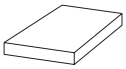
1,2,3... 1. 存储卡
2. 一个专门的 EM 区域称为 EM 文件存储器

注 CJIM CPU 单元中没有 EM 区域 , 因此不能使用 EM 文件存储器。

以上两类存储器均可用来以文件形式存储所有用户程序，I/O 存储区和参数区。



5-1-1 文件存储器的类型

种类	类型	容量	型号	CPU 单元认可文件数据	允许文件操作
存储卡 	闪存	8 M 字节	HMC-EF861	1）所有用户程序 2）指定 I/O 存储器 3）参数区数据（PLC 和其它设置） 见注：4	全部允许 （详细参见 198 页）
		15 M 字节	HMC-EF171		
		30 M 字节	HMC-EF371		
		48 M 字节	HMC-EF571		
EM文件存储器 EM区 Bank 0 Bank 1 Bank n Bank C EM文件存储器	RAM	CPU 单元 EM 区容量 CS 系列 CS1H-CPU67H：832 千字节（Banks 0~C：E0_00000 ～ EC_00000） CJ 系列 CJ1H-CPU66H：448 千字节（Banks 0~6：E0_00000 ～ E6_00000）	从 I/O 存储器中的 EM 区专用存储 Bank 到最末存储 Bank（在 PLC 设置中指定）		启动时的自动传送不能来自 EM 文件存储器中的数据。（详见 198 页）

注 1. 在安装和卸载存储卡时请详细参考第 *5-2 节操作文件*。
2. 首次使用存储器卡或 EM 文件存储器需对其初始化。详细参考 *5-3 文件存储器*使用初始化。
3. HMC-AP001 存储器卡适配器可作为一个储存设备安装在使用存储器卡的个人计算机的一个 PLC 存储器卡槽上。

4. 当使用 CX-Programmer 时 ,CPU 单元能识别符号表（包括 I/O 注解）和注解。当安装了存储器卡 , 则传输的目的是存储器卡 , 否则就是 EM 文件存储器。

存储器卡注意事项

在使用存储器卡前必须确定下列的项目。

格式化

出售前已格式化的存储器卡，在购买后不必再对它们作格式化处理。对曾使用过的存储器要作格式化处理，通常使用 CX-Programmer 或手持编程器在 CPU 单元中处理。

如果存储器卡已被笔记本电脑或其它的计算机直接格式化 , 则 CPU 单元可能不能识别此存储器卡。如此类状况发生 , 即使它在 CPU 单元中被再格式化，也不能够再使用此存储器卡。

根目录下的文件数

存放在存储器卡中根目录下的文件数有一个限定 (正如硬盘有一个限定一样)。虽然限定取决于存储器卡的类型和格式，但是它将被限定在 128 和 512 个文件之间。当应用程序在特定的间隔以 LOG 文件或其它的文件编写时 , 文件应写在子目录下而不要放在根目录下。

在计算机上或使用 CMND(490) 指令可创建子一个子目录。 参照 *3-25-4 DELIVER COMMAND：CS/CJ*系列指令的CMND(490)并参考使用CMND(490)的专门例子

写入次数

一般而言 , 对闪存而言写操作的次数没有限制。然而对于存储器卡 , 在使用说明书中建议，写操作不超过 100，000 次。举例来说 , 如果存储器卡每 10 分钟写入一次 , 则 100,000 写操作相当于可运行 2 年时间。

文件最小长度

如果储存在存储器卡上的 DM 区域中仅包含许多诸如几个字节数据的小文件 ，那么就不可能充分地利用存储器卡容量。举例来说，如果使用 4,096 个字节长度单元的存储器卡 ，不管文件有多大每个文件至少用去 4,096 字节记忆体。如果你把 10 个字节的 DM 区域数据存放在存储器卡上，即使实际的文件大小只有 68 个字节，它同样占用 4,096 字节记忆体。使用类似小文件将会减少存储器卡的利用率。然而如果为提高利用率而减少存储单元长度 ，则数据存取速率将会下降。

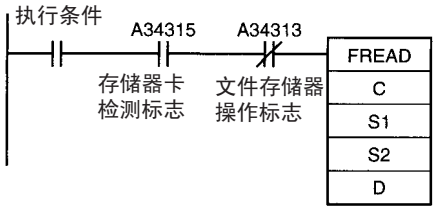
存储器卡的单元分配大小可使用 DOS 中的 CHKDSK 命令来查验。特别的处理过程这里不再详述，更多信息可参考普通计算机有关单元大小分配。

存储器卡存取注意事项

当 PLC 正在存取存储器卡时， CPU 单元上的 "BUSY" 指示灯将会被点亮。请注意下列事项。

1,2,3...

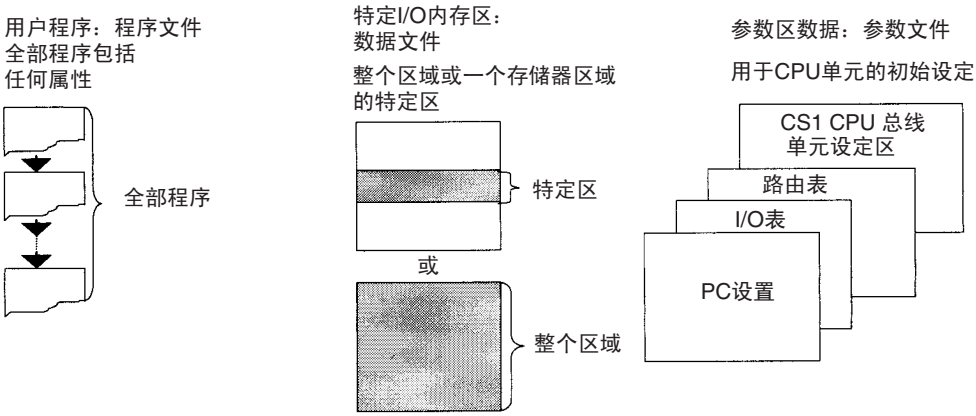
1. 当 "BUSY" 指示灯点亮时 , 不可将 CPU 单元的电源关断 , 否则，存储器卡就会损坏而无法使用。
2. 当 "BUSY" 指示灯点亮时 , 不可拆卸 CPU 单元上的存储器卡。先关断存储器卡电源开关，并在 "BUSY" 指示灯熄灭后才能拆卸存储器卡。否则，存储器卡就会损坏而无法使用。
3. 插入存储器卡时必须把带有标签的一面朝右，不要随意无方向的插入存储器卡。否则会损坏存储器卡或 CPU 单元。
4. 在储器卡被插入之后，需等待数秒时间以便 CPU 单元识别。当接通电源或插入存储器卡之后立刻访问存储器卡，必须把存储器卡的确认标志 (A34315) 常开触点作为输入条件编入程序中，如下图所示。



5-1-2 文件数据

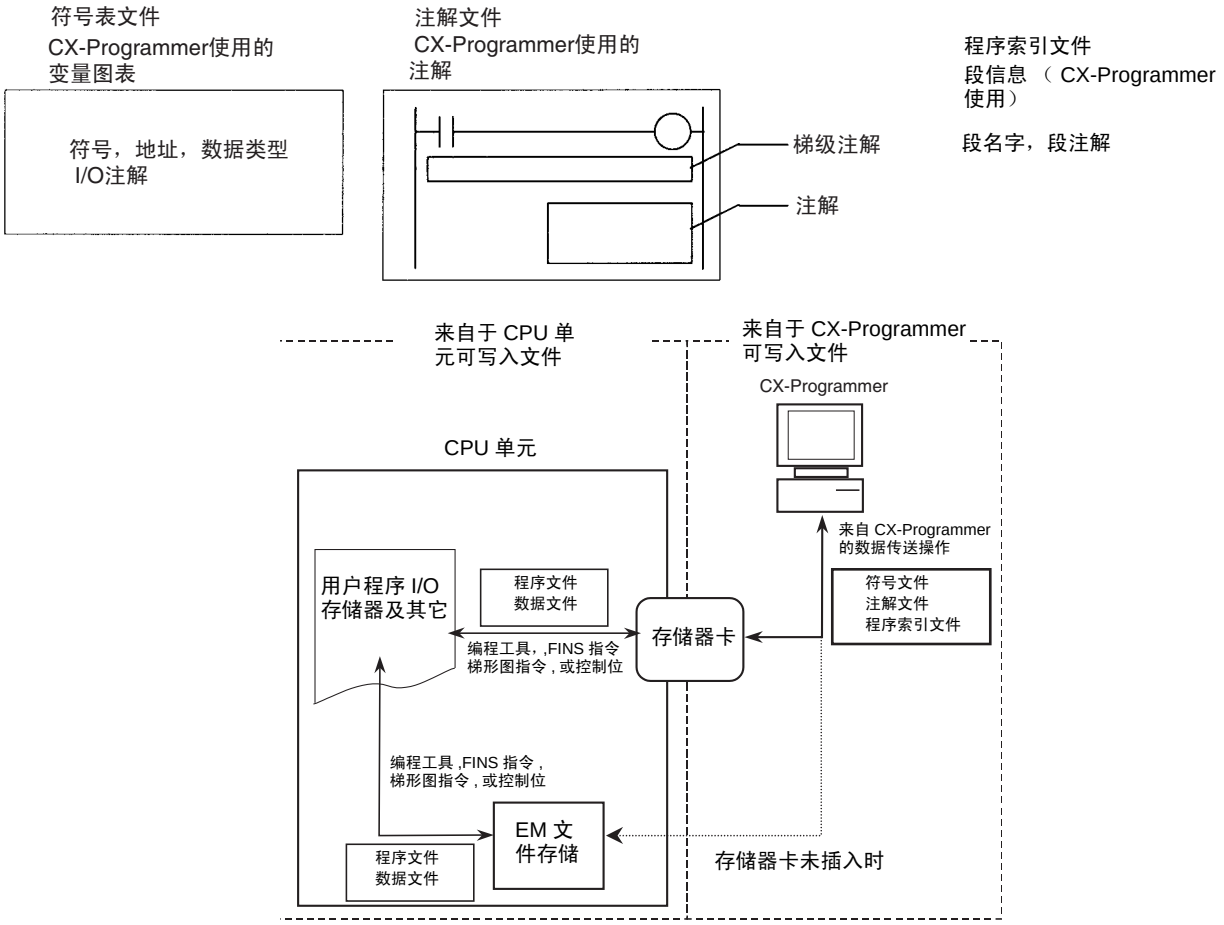
通过编程工具 （CX-Programmer 或手持编程器），用 FINS 命令、梯形图指令、或特别控制位把下列文件写入 CPU 单元的存储器中：

- 程序文件
- 数据文件
- 参数文件



注 下列三类文件也可被 CX-Programmer 进行写操作。

- 符号表文件
- 注解文件
- 程序索引文件



注 符号表 （符号，地址和 I/O 注解）可当作来自 CX-Programmer 的文件处理。

文件	文件名	扩展名	内容
符号表文件	SYMBOLS	.SYM	全局和区域符号
注解文件	COMMENTS	.CMT	梯级注解和注释 (annotations)
程序索引文件	PROGRAM	.IDX	段名字和段注解

通过 CX-Programmer 可为对象进行数据传输操作，在 CPU 单元和存储器卡，或 EM 文件存储器之间传输上述所有的文件 （符号表文件 , 注解文件 , 程序索引文件）。从 2.0 版本开始可支持程序索引文件传送。使用 CX-Programmer1.2 版或更新版同样可在 CX-Programmer、计算机随机储存器 （RAM）和一个数据储存设备之间传送符号表文件和注解文件。

CX-Programmer也可用来保存扩展名为.STD文件中的独立数据区的数据。（这些文件在启动时不能自动传送。所有的参数区域必须保存在一个文件中以便在启动时能自动传送）。

5-1-3 文件

在 DOS 中可对文件格式化，因此它可在带有 Windows 操作系统的计算机上当常规文件使用。

文件可通过文件名和扩展名加以识别，如下所示。文件名可使用下列的字符写成：字母 A 到 Z, 数字 0 到 9,！ ,&,$,#,`,{,},-,^,(,), 和 _。

下列的字符不能用于文件名：,,.,/,￥,?,*,",:,； ,<,>,=,+, 空格。

文件扩展名取决于储存文件的类型。数据文件也有扩展名，如 IOM， TXT，CSV 或 IOR 。（在 EV1 之前 的 CS 系列 CS1 CPU 单元不支持 TXT， CSV 和 IOR 这些扩展名）。程序文件的扩展名为 OBJ，参数文件扩展名为 STD。存储器中的文件可放置在指定在目录中，一个目录路径中最多可达 5 个子目录（包括根目录）。

文件类型，名称和扩展名

CPU 单元可处理 （读和写） 3 类文件。

- **通用文件**
  这些文件可通过编程工具，FINS 命令，指令或辅助区控制位操作进行访问（读和写）。文件名可以自由地由用户定义。
- **启动时自动传送文件**
  打开电源时这些文件将自动地从存储器卡传送到CPU 单元。文件名被指定在批命令 AUTOEXEC 或 ATEXEC@@ 中。
- **备份文件 （CS 系列中在 EV1 之前的 CS1 CPU 单元不支持）**
  这些文件可通过备份功能在存储器卡和 CPU 单元之间传送。文件名被指定在 BACKUP@@ 中。

通用文件

下表所示为通用文件的文件名和扩展。

类型	名称[1]	扩展名	描述	注解	
数据文件（见注 1）	********	.IOM	I/O 存储器指定区域	• 位于一个区域的字单元 （16 个位）从开始到结束数据字。 • 可以是 CIO,HR,WR,AR,DM 或 EM 区。	二进制
		.TXT			文本格式[2]（非分隔或 tab 分隔）
		.CSV			CSV 格式[2]（逗号分隔）
程序文件（见注 1）	********	.OBJ	所有用户程序	• 所有循环、中断任务和一个 CPU 单元中的任务数据。	
参数区域文件	********	.STD	PLC 设置，I/O寄存表、路由表， CPU 总线单元设置[3]等。	• 包含对一个 CPU 单元的所有的初始设定。 • 用户不必区分在文件中的参数数据的类型。	

注 1. 由上述 “********”表示的文件名最多有 8 位 ASCII 字符构成。

2. TXT 和 CSV 文件格式；CS 系列中，在 EV1 之前的 CS1 CPU 单元不支持。

3. 一个 CPU 总线单元的设置实例将是一个数据链接表。其它数据设置请参阅特殊单元的操作手册。

启动时文件自动传输

文件列指示的文件必须存放在存储器卡中，以便这些文件在启动时能自动传送。

类型	名称[1]	扩展名	描述	注释	文件
数据文件	AUTOEXEC	.IOM	I/O 存储器数据 （包含指定的在以 D20000 开始的数据字数）。	• 以D20000开始的DM数据储存在命名为 AUTOEXEC.IOM 的文件中。 • 启动时，所有在文件中的数据传送至以 D20000 开始的 DM 区。 • 当启动时使用自动启动传输功能，这个文件不一定非要在存储器卡中。	---
	ATEXECDM	.IOM	I/O 存储器数据[2] （包含指定的在以 D00000 开始的数据字数）。	• 以D20000开始的DM数据储存在命名为 AUTOEXEC.IOM 的文件中。 • 启动时，所有在文件中的数据传送至以 D00000 开始的 DM 区。 • 当启动时使用自动启动传输功能，这个文件不一定非要在存储器卡中。 注 如果 ATEXECDM.IOM 与 AUTOEXEC. IOM 文件中的 DM 数据有重叠，ATEXECDM.IOM 文件中的数据优先级比后者高。	---
	ATEXECE@	.IOM	EM 区数据 （bank@ ）[2] （包含指定的在以 E@ _0000 开始的数据字数）。	• 以 E@ _0000 开始的 EM bank@ 数据储存在命名为 AUTOEXEC@.IOM 的文件中。最大的bank号取决于所采用的CPU 单元的型号。 • 启动时，所有在文件中的数据传送至以 E@ _0000 开始的 EM bank 。 • 当启动时使用自动启动传输功能，这个文件不一定非要在存储器卡中。	---
程序文件	AUTOEXEC	.OBJ	所有用户程序	• 即使在启动时指定为自动传输，这个文件不一定非要在存储器中。 • CPU 单元中的所有循环和中断任务以及任务数据。	需要
参数区文件	AUTOEXEC	.STD	PLC 设置，I/O 登记表、路由表， CPU 总线单元设置[3] 等。	当启动时使用自动启动传输功能，这个文件就必须在存储卡上。 包括 CPU 单元上的所有初始设置。 用户不必区分文件中的参数数据类型。 在启动时 , 初始设定数据将自动保存在 CPU 单元的特定区域内。	需要

注 1. 必须保证启动时自动传输文件的文件名为 AUTOEXEC 或 ATEXEC@@。

2. ATEXECDM.IOM 和 ATEXECE@.IOM 文件：CS 系列中在 EV1 之前的 CS1 CPU 不支持这些文件。

3. 一个 CPU 总线单元的设置实例是一个数据链接表。其它设置数据参阅有关操作手册。

文件备份 （CS 系列中在 EV1 之前的 CS1 CPU 不支持）

在备份操作过程中，当数据向存储器卡传送或获取时，将自动生成下列表格中的文件。

类型	名称 [1]	扩展名	描述	注解
数据文件	BACKUP	.IOM	分配给特殊 I/O 单元， CPU 总线单元，和内装板的 DM 字（仅限于 CS 系列）	• 包含从 D20000 ～ D32767 DM 数据。 • 如果在备份时从存储卡上读取数据，文件必须要在存储卡上。
	BACKUPIO	.IOR	I/O 存储数据区	• 包含所有在 CIO，WR，HR 和 AR 数据区中的数据以及定时 / 计数器的完成标志和 PV[2]。 • 如果在备份时从存储卡上读取数据，文件必须要在存储卡上。
	BACKUPDM	.IOM	通用 DM 区	• 包含从 D00000 ～ D19999 DM 数据。 • 如果在备份时从存储卡上读取数据，文件必须要在存储卡上。
	BACKUPE@	.IOM	通用 EM 区	包括 EM Bank 中地址为 E _0000 至 E _32767 范围内所有数据 （最大的 Bank 号取决于所采用的 CPU 单元的型号）。 如果在备份时从存储卡上读取数据，文件必须要在存储卡上。 • 当数据备份到存储卡时，每个 EM Bank 内的所有数据将被自动写入独立的文件中。
程序文件	BACKUP	.OBJ	所有用户程序	• 包括 CPU 单元中所有的循环和中断任务程序以及任务数据。 • 如果在备份时从存储卡上读取数据，文件必须要在存储卡上。
参数区域文件		.STD	PLC 设置，I/O 登记表、路由表、 CPU 总线单元设置 [3] 等。	• 包括一个 CPU 单元的所有初始设定。 • 用户不必区分文件中参数数据的类型。 • 如果在备份时从存储卡上读取数据，文件必须要在存储卡上。
单元 / 板备份文件 （仅限 CS1-H,CJ1-H, 或 CJ1M CPU 单元）	BACKUP@@ （其中 是正在复制中的单元板的地址）	.PRM	指定单元或板	• 控制某个单元或板中的备份数据。详情参考 *5-2-6 简单备份*功能。

注 1. 必须确认用于备份的文件的文件名为 BACKUP@@。

2. 在启动时从存储卡上读取的 CIO 区，WR 区，定时 / 计数器完成标志和 PV，以及强制置位 / 强制复位的数据将被清除。在 PLC 设置启动时的 IOM 保持位状态和强制保持位状态时， PV 数据将保留下来。

3. 一个 CPU 总线单元的设置实例是一个数据链接表。其它设置数据参阅有关操作手册。

目录

CS/CJ 系列 PLC 能够访问子目录下的文件，但手持式编程器只能访问根目录下的文件。目录路径的最大长度为 65 个字符。必须保证用象 Windows 这样的程序在存储器卡上生成子目录时，路径长度不能超过其最大字符数。

文件大小

以字节表示的文件大小可通过下表中所列的公式计算得出。

文件类型	文件大小
数据文件 (.IOM)	（字数 ×2） +48 字节 例如：整个 DM 区 （D00000-D32767） （32,768 字 ×2） +48=65,584 字节
数据文件 (.TXT 或 .CSV)	文件大小取决于分隔符和回车符使用数。分隔符码是 1 个字节，而回车符码是 2 个字节的。 例 1：无分隔字 , 没有回车 123456789ABCDEF012345678 占用 24 个字节。 例 2：每 2 个区分隔一个字，一个回车。 1234,5678↵ 9ABC,DEF0↵ 1234,5678↵ 占用 33 个字节。 例 3：每 2 个区分隔两个字，一个回车。 56781234,DEF01234↵ 56781234↵ 占用 29 个字节。
程序文件 (.OBJ)	（使用的步数 ×4） +48 字节 （见注）
参数文件 (.STD)	16,048 字节

注 在程序文件中，步数通过把全部 UM 步数减去可利用的 UM 步数来计算得出。这些值在 CX-Programmer 的对比报告（Cross-Reference）中给出。详情参考 *CX-Programmer 用户手册*。

数据文件

通用文件

1,2,3... 1. 通用数据文件的文件扩展名为 IOM， TXT 或 CSV 。（TXT 和 CSV 文件 : CS 系列中在 EV1 之前的 CS1 CPU 不支持这些文件）。

扩展名	数据格式	内容		字 / 字段
.IOM	二进制	CS/CJ 系列数据格式		---
.TXT（见注）	无分隔字	ASCII 格式	这个格式通过把 I/O 存储器的单字段 （4 个 16 进制数）转换为 ASCII 码，生成不分隔排列字段。记录可使用回车分隔。	1 字
	无分隔双字		这个格式通过把 I/O 存储器的双字段 （8 个 16 进制数）转换为 ASCII 码，生成不分隔排列字段。记录可使用回车分隔。	2 字
	制表符分隔字		这个格式通过把 I/O 存储器的单字段 （4 个 16 进制数）转换为 ASCII 码，并用制表符分隔字段。记录可使用回车分隔。	1 字
	制表符分隔双字		这个格式通过把 I/O 存储器的双字段 （8 个 16 进制数）转换为 ASCII 码，并用制表符分隔字段。记录可使用回车分隔。	2 字
.CSV（见注）	逗号分隔字		这个格式通过把 I/O 存储器的单字段 （4 个 16 进制数）转换为 ASCII 码，并用逗号分隔字段。记录可使用回车分隔。	1 字
	逗号分隔双字		这个格式通过把 I/O 存储器的双字段 （8 个 16 进制数）转换为 ASCII 码，并用逗号分隔字段。记录可使用回车分隔。	2 字

注 a) 读 / 写 TXT 和 CSV 数据文件：
TXT 和 CSV 数据文件只能通过 FREAD(700) 和 FWRIT(701) 进行读写。

b) 有关字符的注意事项：
如果 TXT 或 CSV 文件包含的不是十六进制数 （0 ～ 9， A ～ F，或 a ～ f），则不能把数据正确地写到 I/O 存储器中去。

c) 字段长度注意事项：
在使用单字时 , 如果 TXT 或 CSV 文件字段不是 4 个十六进制数，则数据就不能够顺利写入 I/O 存储器。同样在使用双字时 , 如果 TXT 或 CSV 文件字段不是 8 个十六进制数，则数据也不能够顺利写入 I/O 存储器。

d) 存储顺序：
在使用单字时 ,I/O 存储器数据被转换成 ASCII 并以一个字段从最低到最高的 I/O 存储器地址按顺序储存。
在使用单字时 ,I/O 存储器数据被转换成 ASCII 并以一个字段从最低到最高的 I/O 存储器地址按顺序储存。（2 字段以内数据 , 首先储存高地址字，然后储存低地址字）。

e) 分隔符：
没有分隔符时，字段被连续排列然后储存。当被逗号分隔时，在储存之前，逗号被插入字段之间。当被制表符分隔时，在储存之前，制表符代码被插入字段之间。
当分隔符 （逗号或制表符）被 FREAD(700) 指定，数据则以一字段的 （逗点或制表符）分隔形式被读取。

f) 回车：
不用回车时，数据被连续排列。当使用回时，回车代码被插入在指定字段数目之后。如果在文件中使用回车，
FREAD(700)/FWRIT(701) 指令就不能指定文件开头 （字存取）的偏移量。

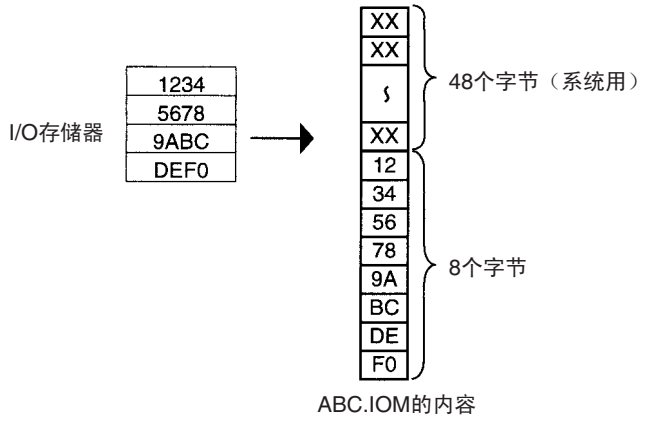
g) 字段数：
文件中数据的总量取决于在 FWRIT(701) 指令中指定的字段数（写入条目数）以及字 / 字段的数量。使用单字时为 1 字 / 字段，双字则为 2 字 / 字段。

2. 数据文件不指明保存的是什么数据，如保存的是哪个存储区域的数据。因此必须给文件一个可表示其内容的文件名以便于文件管理，如下例所示。
例：D00100.IOM, CIO0020.IOM
即使写入到数据文件的原始数据 (IOM ， TXT 或 CSV) 是来自不相同的区域，文件开始的数据总是写入到指定的 I/0 存储器中的起始地址。例如：如果通过一个编程工具把某个文件中的 CIO 数据写入 DM 区， CPU 单元的 DM 区读取这些数据时并不会出现任何关于区域不同的提示。

注 包含十六进制数（0 ～ 9，A ～ F）的 TXT 和 CSV 格式的数据文件允许 I/O 存储器中的数字数据与电子数据表程序进行数据交换。

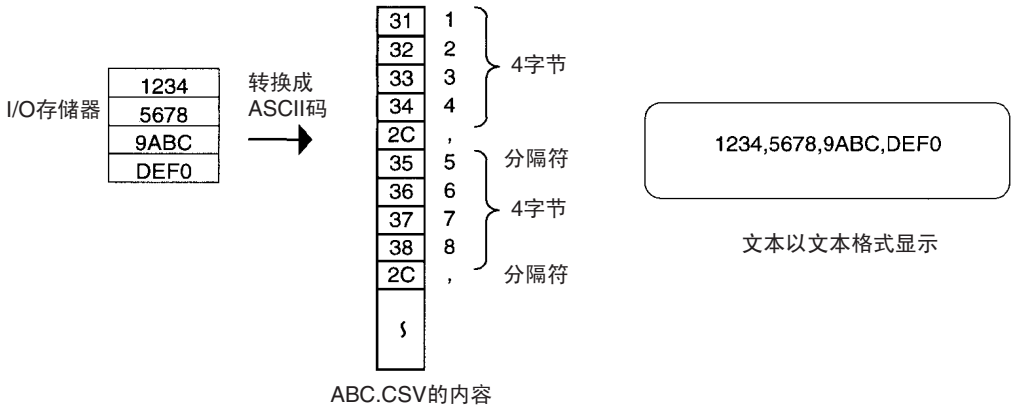
IOM 数据文件结构

下面的例子给出了 I/O 存储器中包含四个字的数据文件 （ABC.IOM）的二进制数据结构：1234H， 5678H， 9ABCH，和 DEF0H。然而用户在通常操作时并不需要考虑数据格式。



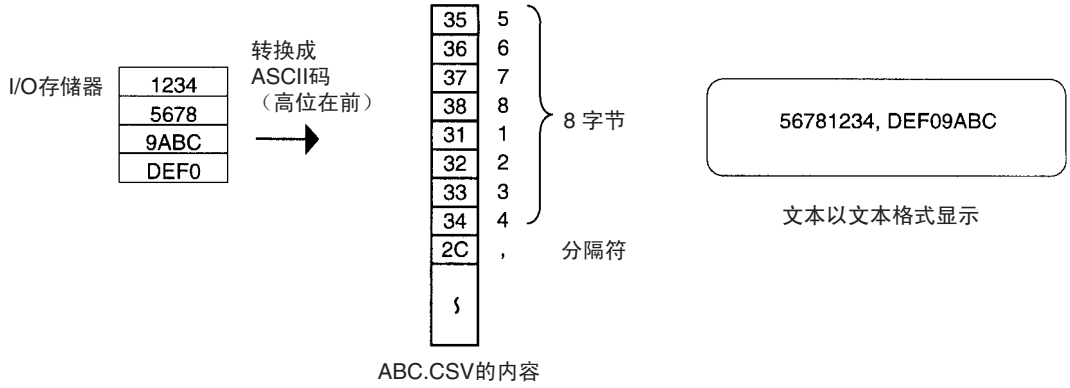
CSV/TXT 数据文件结构（单字）

下面的例子给出了 I/O 存储器中包含四个字的单字段 CSV 数据文件 (ABC.CSV) 的数据结构：1234H， 5678H， 9ABCH 和 DEF0H。单字段 TXT 文件的结构与它相同。



CSV/TXT 数据文件结构（双字）

下面的例子给出了 I/O 存储器中包含四个字的双字段 CSV 数据文件 (ABC.CSV) 的结构：1234 Hex, 5678 Hex，9ABC Hex 和 DEFO Hex。双字段 TXT 文件的结构与它相同。



用电子表格软件创建数据文件

使用电子表格 （如：Microsoft Excel）程序软件创建 TXT 和 CSV 数据文件须按下列规程进行。

- 设定单元内容为数字或字符。
- 若为单字段，则每个单元输入 4 个字符；若为双字段，每个单元输入 8 个字符。举例来说，单字段时应输入 000A，不能仅输入 A。
- 确定每单元输入的只能是十六进制字符（0 ～9，A～F或a～f）。其它字符和代码一概不能使用。

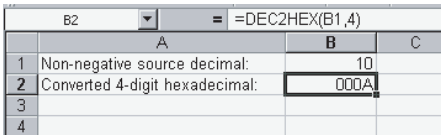
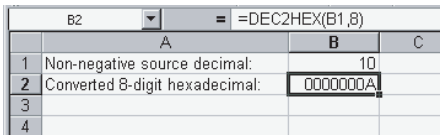
若要把十六进制数保存在 I/O 存储器中，把电子数据表的十进制输入转换成十六进制是很有帮助的。按照下列步骤进行十六进制转换。

1,2,3...

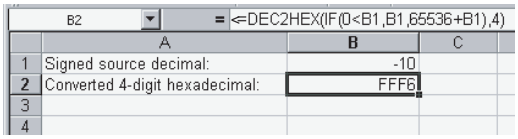
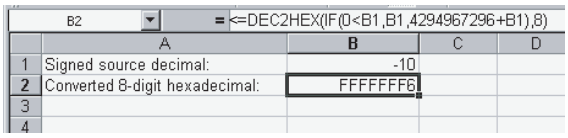
1. 从工具菜单中选择 *Add-Ins...*。
2. 从 *Add-Ins* 菜单中选择 Analysis ToolPak。
3. 从插入功能单元菜单中选择 *Function*。
4. 从工程分类字段选择 *DEC2HEX* （数值 , 数字）。
5. 若需转换 4 位十六进制数时 , 输入下列数值变量：IF（0<=Cell location,cell location,65535+cell location）

   若需转换 8 位十六进制数时 , 输入下列数值变量：IF（0<=Cell location,cell location,429467296+ceil location）

• 例 1：非负数十进位值的输入。

项目	将无符号十进制转换成 4- 十六进制数	将无符号十进制转换成 8- 十六进制数
功能使用	DEC2HEX(*cell_location*,4)	DEC2HEX(*cell_location*,8)
例	输入十进制数10，并转换成4-十六进制数000A。 	输入十进制10，并转换成8-十六进制数0000000A。 

• 例 2：带符号十进制数值的输入。

项目	将带符号十进制转换成 4- 十六进制数	将带符号十进制转换成 8- 十六进制数
功能作用	DEC2HEX(IF(0<=*cell_location*,*cell_location*,65536+*cell_location*),4)	DEC2HEX(IF(0<=*cell_location*,*cell_location*,4294967296+*cell_location*),8)
例	输入十进制数-10，并转换成4-十六进制数FFF6。 	输入十进制数-10并转换成8-十六进制数FFFFFFF6 

启动时自动传输数据文件

如果使用启动时的自动传输功能，有三种类型的文件在启动时进行自动传输。

- AUTOEXEC.IOM：分配给特殊 I/O 单元和内装板的 DM 字。
  在电源打开时，文件的内容自动传输给以 D20000 开始的 DM 区。
- ATEXECDM.IOM：通用 DM 字。
  在电源打开时，文件的内容自动传输给以 D00000 开始的 DM 区。
- ATEXECE@.IOM：通用 EM 字。
  在电源打开时，文件的内容自动传送给以 E@_00000 开始的 EM 区。

当生成上面所列文件时，一定要指明上面所示的首地址（D20000， D00000，或 E@_00000），并确保文件尺寸不能超过指定数据区的容量。

每个文件中的所有数据都是传输到以指定首地址（D20000， D00000 或 E@_00000）开始的区域中去。

注 1. 当通过编程工具（手持编程器或 CX-Programmer）生成 AUTOEXEC.IOM，ATEXECDM.IOM， ATEXECE@.IOM 文件时，一定要指明适当的首地址（D20000， D00000，或 E@_00000）并确保文件尺寸不能超过 DM 区或指定 EM Bank 的容量。即使指定了可能导致错误数据覆盖 DM 区或 EM Bank 的那部分内容的另一开始字，文件的内容总是传输到以适当的首地址（D20000，D00000 或 E@_00000）开始的区域中去。进一步讲，如果超过了 DM 区或 EM Bank 的容量（当通过 CX-Programmer 进行设置时可能出现这种情况），当超过 DM 区容量后，余下的数据写到 EM Bank0 中去；当某个 EM Bank 容量满后，余下的数据写到下面的 EM Bank 中去。

2. 当使用 CX-Programmer 时，你能指定一个大于 DM 区最大地址 D32767 或 EM 区最大地址 E@_32767 的数据文件。如果 AUTOEXEC.IOM 文件越过了 DM 区的范围，则所有余下的数据将写到以 E0_00000 开始的 EM 区中去，并且还是按照存储器地址和起止存储单元顺序进行写入。启动时也能把数据自动传送到 DM 区或 EM 区。同样，如果 ATEXECE@.IOM 文件大于 EM Bank，余下的数据将写到随后的 EM Bank 中去。
3. 特殊 I/O 单元， CPU 总线单元和内装板单元（仅限于 CS 系列）的系统设置可通过使用不同的 AUTOEXEC.IOM 文件进行更改，这些文件中含有为专用 I/O 单元区 （D20000 ～ D29599）， CPU 总线单元区 （D30000 ～ D31599），以及内装板区（D32000 ～ D32099）设定的不同的设置值。存储器卡可为特殊的 I/O 单元，CPU 总线单元，和不同系统或器件的内装板（仅限于 CS 系列）创建系统安装数据库。

数据文件备份

备份功能可生成如下所述的四种数据文件。

为了备份数据，必须把 CPU 单元上的 DIP 开关上脚 7 置 ON，脚 8 置 OFF，插入存储器卡，并且按下存储器卡上电源按钮并保持三秒钟。四种备份文件（BACKUP.IOM, BACKUPIO.IOR,BACKUPDM.IOM, 以及 BACKUPE@.IOM）将自动生成并写到存储卡中去。

四种备份文件只能由备份功能使用，而且三种文件 （BACKUP.IOM, BACKUPDM.IOM 和 BACKUPE@.IOM）可由编程工具操作创建（BACKUPIO.IOR 不能由编程工具操作创建。

5-1-4 文件操作步骤的描述

下表概括了能用于文件读写的六种方法。

读：从文件存储器到 CPU 单元。
写：从 CPU 单元到文件存储器。

操作步骤	介质	文件名	描述	所有程序	数据区数据（见注 3）	参数区数据
编程工具 （包括手持编程器）	存储器卡 EM 文件存储器	任何有效的文件名	读	可以	可以	可以
			写	可以	可以	可以
			其它操作（见注 2）	可以	可以	可以
FINS 命令 （见注 1）	存储器卡 EM 文件存储器	任何有效的文件名	读	可以	可以	可以
			写	可以	可以	可以
			其它操作（见注 2）	可以（见注 4）	可以	可以
FREAD(700) 和 FWRIT(701) 指令	存储器卡 EM 文件存储器	任何有效的文件名	从文件读出数据	不可以	可以	不可以
			写入数据到文件	不可以	可以	不可以

操作步骤	介质	文件名	描述	所有程序	数据区数据（见注 3）	参数区数据
在操作时用辅助区控制位操作代取整个程序 （CS 系列中在 EV1 之前的 CS1 CPU 不支持）	存储器卡	任何有效的文件名	读	可以	不可以	不可以
启动时自动传送	存储器卡	AUTOEXEC 或 ATEXEC@@	读	可以	可以	可以
			写	不可以	不可以	不可以
备份操作 （CS 系列中在 EV1 之前的 CS1 CPU 不支持）	存储器卡	BACKUP@@	读	可以	可以	可以
			写	可以	可以	可以

注
1. 用于文件存储操作的 FINS 命令可由一个连接到 Host Link 的主计算机，连在网络上的其它 PLC （使用 CMND(490)），或本地 PLC 的程序 （使用 CMND(490)）上发送出去。（就 CS 系列中在 EV1 之前的 CS1 CPU 而言，在同一个 CPU 单元上文件存储操作不能使用 CMND(490)，因为这个 CPU 正在执行文件存储操作）。
2. 其它的操作：文件存储器格式化，读文件数据，写文件数据，更改文件名，读文件存储器数据 , 删除文件，拷贝文件 , 创建子目录。
3. 用 TXT 或 CSV 格式的数据文件仅可用 FREAD(700) 和 FWRIT(701) 指令访问。但不能通过编程工具访问。
4. 版本 1.2 和更高版本的 CX-Programmer 可用来在计算机的 RAM 与存储设备间传输程序文件 （.OBJ）。

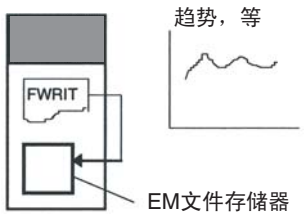
5-1-5 应用

文件存储器可在下列应用中使用。

数据文件

在这个应用中， DM 区设定数据 （为特殊 I/O 单元， CPU 总线单元，以及内装板（仅限于 CS 系列）设定的）保存在存储卡中。如果文件命名为 AUTOEXEC.IOM，那么在电源接通时，文件中保存的设定值将自动传输。

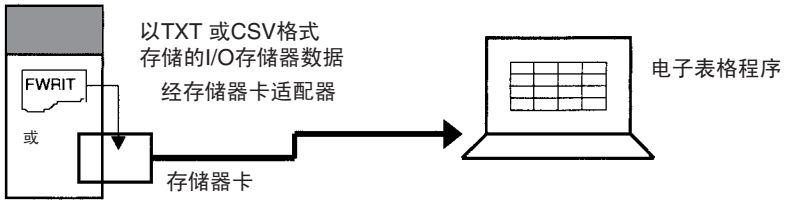
在这一应用中，在程序运行期间所产生的运算数据 （趋势，质量控制和其它的数据）使用写数据文件指令 （FWRIT （701））储存在 EM 文件存储器中。



注 经常被存取的数据，如趋势数据，最好保存在 EM 文件存储器中，而不要保存在存储卡上。

ASCII 数据文件 (.TXT 和 .CSV)

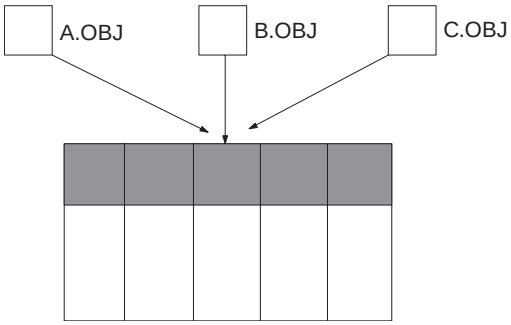
以 TXT 或 CSV 格式保存在存储卡上的结果数据可以通过存储卡适配器传送到个人计算机上去，并使用电子数据表程序进行编辑。（CS 系列中在 EV1 之前的 CS1 CPU 不支持这个功能）。



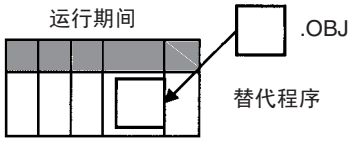
相反，像特殊 I/O 单元设定值数据可由电子数据表程序 TXT 或 CSV 格式生成，保存在存储卡上，并且使用 FREAD(700) 指令读到 CPU 单元中去 （CS 系列中在 EV1 之前的 CS1 CPU 不支持这个功能）。

程序文件 (.OBJ)

在这个应用中，控制不同过程的程序保存在各自的存储卡上。整个 PLC 配置（程序， PLC 设置，等）可通过插入一个不同的存储卡更改，并在启动时使用自动传输功能。

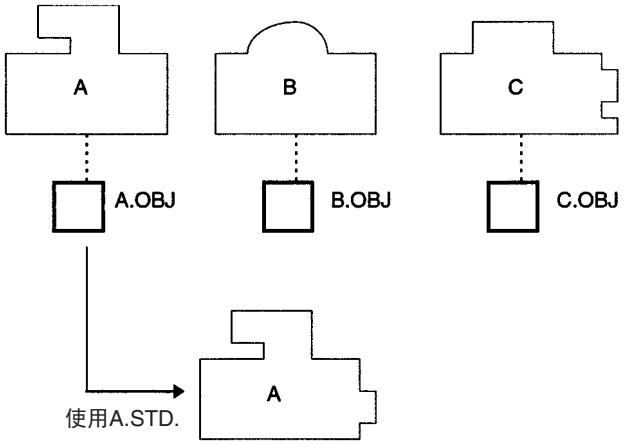


在运行过程中，整个程序可以由程序本身 （不使用编程工具）通过使用一个辅助区控制位来替代。（CS 系列中在 EV1 之前的 CS1 CPU 不支持这个功能）。



参数区文件 (.STD)

在这个应用中， PLC 设置，路由表， I/O 表，以及其它特定设备和机器的有关数据保存在存储卡上。仅需要变换存储卡就能把数据传送到其它设备和机器上去。



备份文件

无需编程工具，备份功能可用来在存储卡上储存所有的 CPU 单元数据 （ 包括全部 I/O 存储器，程序和参数区）。一旦 CPU 单元数据出现问题，可用备份数据立刻恢复。（CS 系列中在 EV1 之前的 CS1 CPU 不支持这个功能）。

符号表文件

CX-Programmer 可把程序符号和 I/O 注释保存在 SYMBOLS.SYM 文件中，并存储到存储卡或 EM 文件存储器中。

注释文件

CX- Programmer 可把程序梯级注释保存在 COMMENTS.CMT 命名的注释文件中，并存储到存储卡或 EM 文件存储器中。

5-2 操作文件

可使用下列的步骤和方法，以文件的方式实现读 , 写和其它工作。

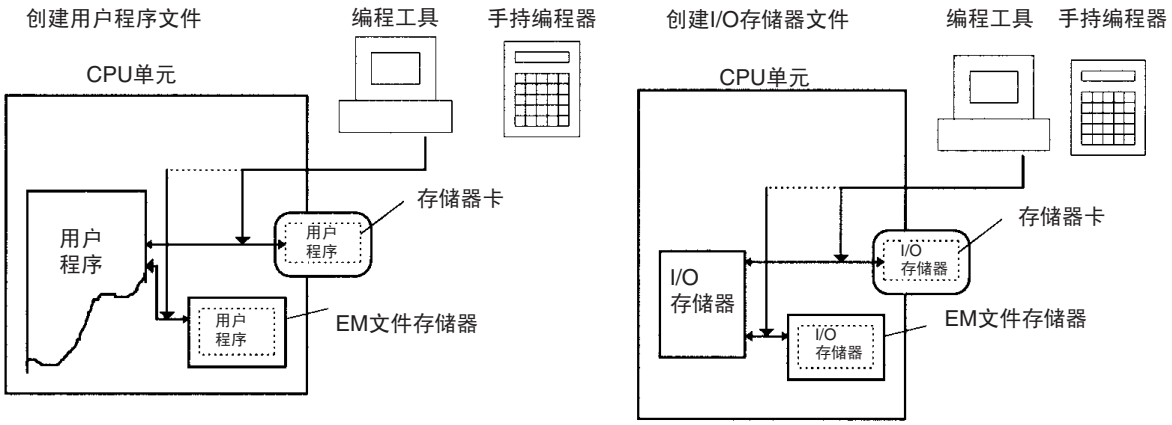
- 编程器
- FINS 命令
- FREAD(700), FWRIT(701) 和 CMND(490) 指令。用户程序中 CMND(490)（CS 系列中在 EV1 之前的 CS1 CPU 不支持这个功能）。
- 采用辅助区控制位替代所有程序。（CS 系列中在 EV1 之前的 CS1 CPU 不支持这个功能）。
- 启动时自动传送
- 备份功能 （CS 系列中在 EV1 之前的 CS1 CPU 不支持这个功能）。

5-2-1 编程工具 （ 包括手持编程器）

下列的操作可利用编程装置完成。

操作	CX-Programmer	手持编程器
读文件（从文件存储器传送至 CPU 单元）	可以	可以
写文件（从 CPU 单元传送至文件存储器）	可以	可以
文件比较 （在文件存储器与 CPU 单元之间比较）	不可以	可以

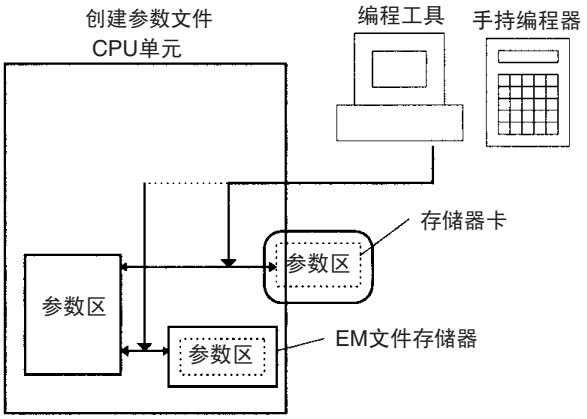
操作		CX-Programmer	手持编程器
格式化文件存储器	存储器卡	可以	可以
	EM 文件	可以	可以
更改文件名		可以	不可以
读文件存储器数据		可以	不可以
删除文件		可以	可以
复制文件		可以	不可以
删除 / 创建子目录		可以	不可以



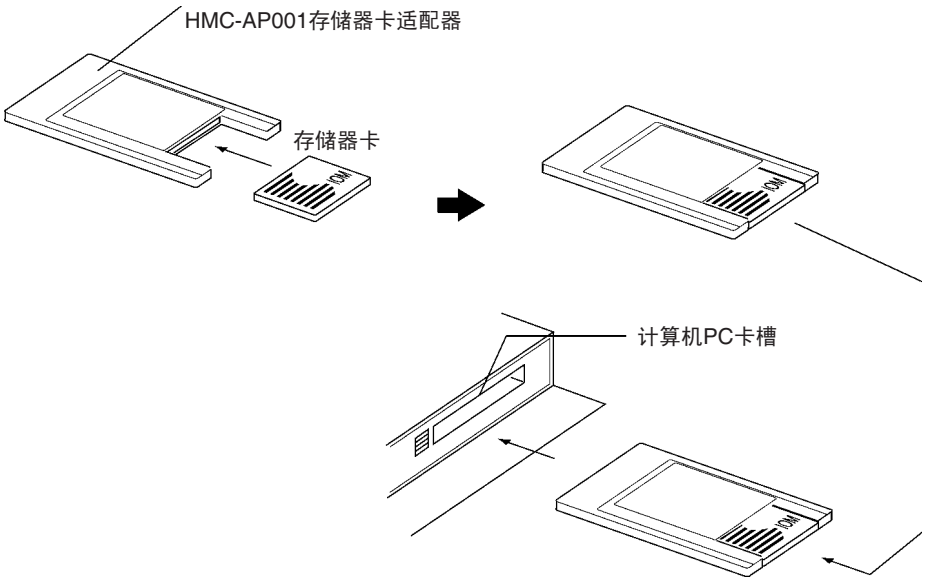
注 1. 可使用 Windows 浏览器生成任何所需卷标。

2. 可使用 Windows 对文件存储器快速格式化。一旦存储器卡格式化发生差错 , 还可用 Windows 普通方式重新格式化。

3. 时间和日期可用文本格式通过 CPU 单元同步时钟传递至 CPU 单元文件存储器。



存储卡可以安装在带有 HMCAP001 存储卡适配器的计算机 PC 卡槽中（卡槽分开）。安装在计算机中的存储卡允许卡文件被其它的程序访问，如 Windows 浏览器。



CX-Programmer

可使用下列的步骤进行文件存储操作。

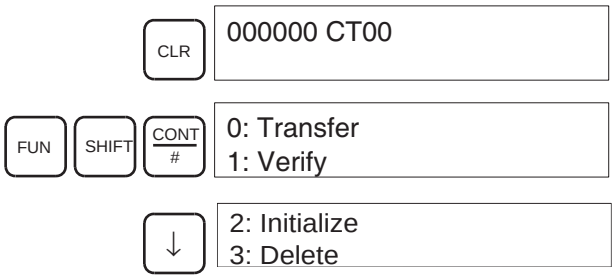
1,2,3...

1. CPU单元在线情况下，双击对象管理窗口的存储卡图标，显示存储卡窗口。
2. 从 CPU 单元向文件存储器传送 , 在对象管理工作区选择程序区域 ,I/0 存储区域或参数区域 , 选择从文件存储器传送 , 然后选择向存储器卡或向 EM 文件存储器传送。

或 从文件存储器向 CPU 单元传送 , 则选取文件存储器中的文件然后将它拖曳到对象区域 ,I/O 存储区域 , 或参数区域，并将其释放。

注 使用对象传输操作在 CX-Programmer 上生成并读取符号表文件（SYMBOLS.SYM）注释文件 （COMMENTS.CMT）。

手持编程器



可进行下表操作

项目 1	项目 2	项目 3	项目 4	项目 5
0：传送	0：PLC 到存储卡	选择 OBJ, CIO, HR, WR, AR, DM, EM 或 STD	设置传送首和末地址	介质类型，文件名
	1：存储卡到 PLC	选择 OBJ, CIO, HR, WR, AR, DM, EM 或 STD	设置传送首和末地址	介质类型，文件名
1：校验		选择 OBJ, CIO, HR, WR, AR, DM, EM 或 STD	设置传送首和末地址	介质类型，文件名

项目 1	项目 2	项目 3	项目 4	项目 5
2：初始化		进入 9713（存储器卡）或 8426（EM 文件存储器）	---	---
3：删除		选择 OBJ, CIO, HR, WR, AR, DM, EM 或 STD	介质类型，文件名	---

注 文件类型由下表所示。

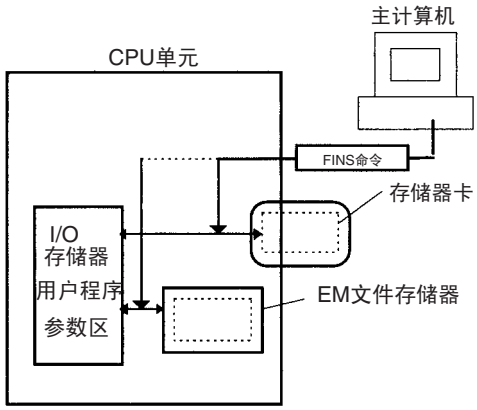
符号	文件类型	
OBJ	程序文件 (.OBJ)	
CIO	数据文件 (.IOM)	CIO 区
HR		HR 区
WR		WR 区
AR		辅助区
DM		DM 区
EM0_		EM 区
STD	参数文件 (.STD)	

5-2-2 FINS 命令

CPU 单元在接收到特有的 FINS 命令后，能进行下列的文件存储操作。这类似于编程工具的功能。

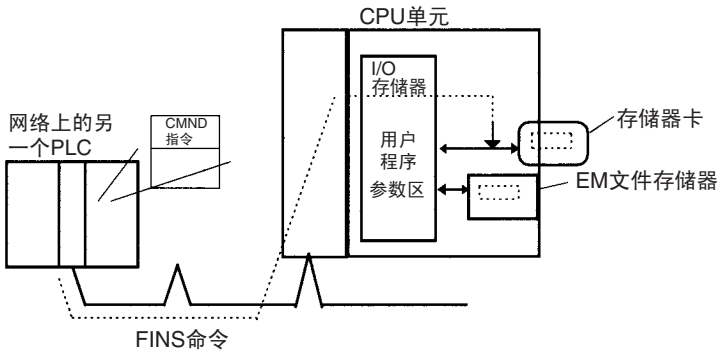
经 Host Link 的 FINS 命令

连接经由一个 Host Link 系统的计算机能在主链电路与终端器之间传送出一个 FINS 命令。

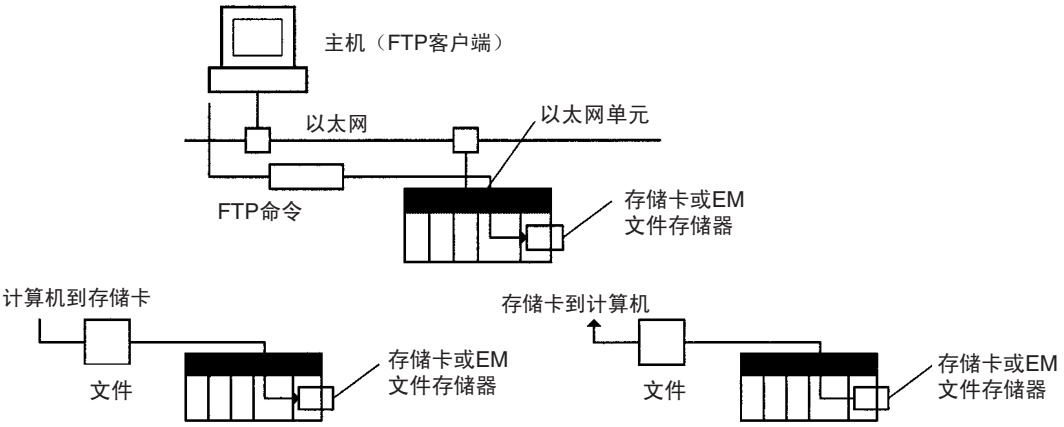


FINS 指令来自于其它 PLC 网络

网络中的其它 PLC 可利用 CMND(490) 发送 FINS 指令。



注 连接以太网中的计算机可通过以太网访问在 CPU 单元上的文件存储器 （存储卡或 EM 文件存储器）。只要主计算机具备 FTP 用户端功能以及 CS/ CJ 系列的 PLC 具备 FTP 服务器的功能，就可以改变文件中的数据。



下表所示的 FINS 命令可运行各种功能，包括对文件的访问。

命令	含义	描述
2201 Hex	读文件名	读文件存储器数据
2202 Hex	读单个文件名	从一个单文件中指定位置读取一个指定长度的文件数据
2203 Hex	写单个文件	向一个单文件中指定位置写入一个指定长度的文件数据
2204 Hex	文件存储器格式化	文件存储器格式化 （初始化）
2205 Hex	删除文件	删除文件存储器中的一个指定文件
2207 Hex	文件复制	将文件从一个文件存储器复制到另一个文件存储器
2208 Hex	更改文件名	更改文件名
220A Hex	存储区文件传送	在 I/O 存储器与文件存储器之间传送或比较数据
220B Hex	参数区文件传送	在参数区与文件存储器之间传送或比较数据
220C Hex	程序区文件传送	在 UM （用户存储器）区与文件存储器之间传送或比较数据
2215 Hex	创建 / 删除子目录	创建 / 删除子目录

注 用 220 A 、220 B、220 C 和 2203 命令将 CPU 单元内部时钟的时间注明在文件存储器中文件的创建日期。

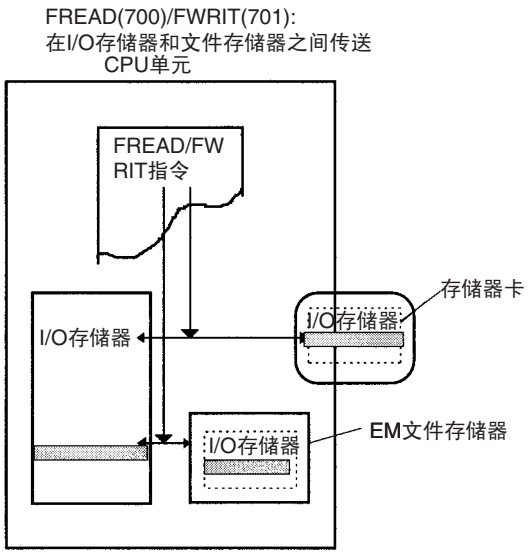
5-2-3 FREAD(700), FWRIT(701) 和 CMND(490)

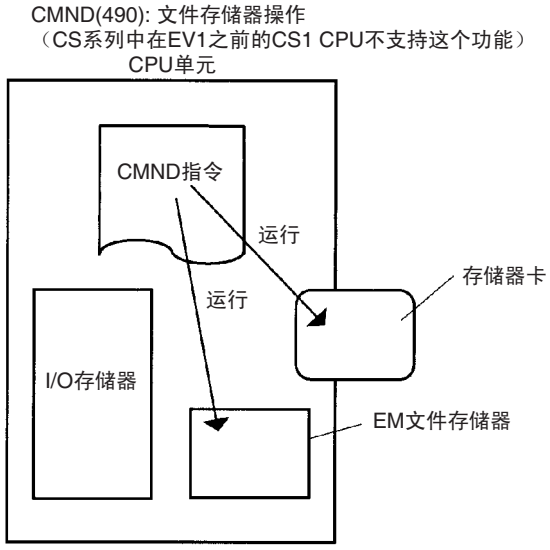
FWRIT(701) （写数据文件）指令可用来创建一个存储在存储器卡或 EM 文件存储器中的包含特殊 I/O 存储器数据的文件。它也可在已存在文件的任何地方加入或覆盖内容。

FREAD(700) （读数据文件）指令可从存储器卡或 EM 文件存储器中的数据文件指定处读取 I/O 存储器数据，也可将数据写入 I/O 存储器的指定部分。它可从指定文件中的任一处读取数据。

注 这些指令不能传送指定文件，但是能从文件指定起始点开始传送指定量的数据。

为进行文件操作，用 CMND(490) （发布命令）指令可向 CPU 单元本身发送 FINS 命令。可在存储器卡或 EM 文件存储器中运行文件格式化、清除、拷贝、更改文件名等操作 （CS 系列中在 EV1 之前的 CS1 CPU 不支持这个功能）。





FREAD(700)/ FWRIT(701) 指令

FREAD(700) 和 FWRIT(701) 指令用于在 I/O 存储器和文件存储器之间传送数据。所有的 CJ CPU 单元能传送二进制数（.IOM 文件），V1 CPU 单元也能传送 ASCII 文件 （.TXT 和 .CSV 文件）。

名称	助记符	描述
读数据文件	FREAD(700)	到指定 I/O 存储器读指定数据文件的数据或数据元素
写数据文件	FWRIT(701)	用指定 I/O 存储器区中的数据创建一个指定的数据文件。

传输 ASCII 文件 （EV1 前的 CS 系列 CS1 CPU 单元不支持）

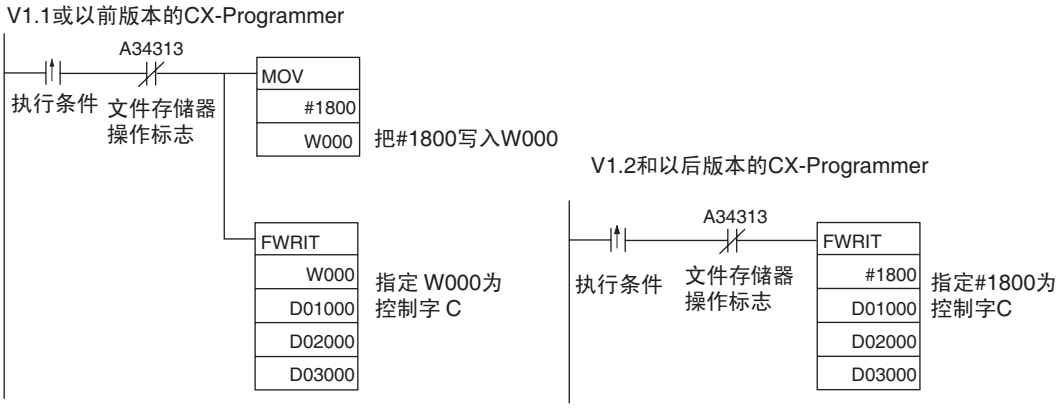
ASCII 文件也可以二进制文件传输，这样，指令控制字操作数 （C）的第 3 和第 4 位字节指明了在回车之间传送文件的类型和文件数。

C 中的位	设定	编辑工具的限制
12 ～ 15	数据的类型 0：二进制 （.IOM） 1：无分隔字 （.TXT） 2：双字长无分隔字 （.TXT） 3：逗号分隔字 （.CSV） 4：双字长逗号分隔字 （.CSV） 5：表格分隔字 （.TXT） 6：双字长表格分隔字 （.TXT）	如果使用 V1.1 或以前的版本的 CX-Programmer，只能用 0HEX （.IOM 文件）直接定义。 如果使用 V1.2 或以后的版本的 CX-Programmer （或者是手持式编程器），控制字位可在 0 ～ 6 HEX 间设定。
08 ～ 11	回车符 0：无回车 8：每 10 个字段一个回车 9：每 1 个字段一个回车 A：每 2 个字段一个回车 B：每 4 个字段一个回车 C：每 5 个字段一个回车 D：每 16 个字段一个回车	如果使用 V1.1 或以前的版本的 CX-Programmer （包括手持式编程器），只能用 0 HEX 直接定义。 如果使用 V1.2 或以后的版本的 CX-Programmer，控制字位可在 0 HEX 和 8 ～ D HEX 间设定。

V1.1 或以前版本的 CX-Programmer
间接设定控制字

当使用 V1.1 或以前的版本的 CX-Programmer，如果输入一个常数定义数据类型和回车处理的控制字，ACSII 码文件不能用 FREAD（700）和 FWRIT（701）指令传输。如果使用一个常数，只能传输无回车分隔的二进制文件。

然而，通过间接设定控制字，可用 FREAD （700）和 FWRIT （701）指令传输 ACSII 码文件。把所需控制字设定到一个字中，并指定这个字在 FREAD （700）和 FWRIT （701）指令中作为控制字，如下左下图所示。



注 用 FWRIT （701）指令，将 CPU 单元中内部时钟标注生成在文件存储器的数据文件日期。

因为在任一时刻只有一个文件存储操作可执行，所以，当下列文件存储操作在运行时 FREAD （700）和 FWRIT （701）不能执行：

1,2,3...

1. FREAD （700）和 FWRIT （701）的执行。
2. 执行 CMND （490）将 FINS 命令传送到 CPU 单元本身。
3. 通过辅助区域控制位操作替换整个程序。
4. 一个简单的备份操作的执行。

将文件存储器操作标志（A34313）用于文件存储器指令的互斥控制，以防止它们在另一个文件存储器正在操作期间被执行。

如果指定的文件包含有错的数据类型或文件数据出错，执行 FREAD （700）时，文件读出错标志 （A34310）将变为 ON，指令将不能执行。对于文本或 CSV 文件，字符码必须是十六进制数据，定界符必须为每 4 个数字（对于字数据）和每 8 个数字 （对于双字数据）定位。数据将被读出，直到在某个点上检测到非法字符为止。

相关的辅助位 / 字

名称	地址	操作
存储卡类型	A34300～A34302	指明任何一种已安装好的存储卡类型。
EM 文件存储器格式化错误标志	A34306	在第一个分配给文件存储器的 EM Bank 中发生格式化出错时，为 ON。 当格式化正常完成时，为 OFF。
存储卡格式化错误标志	A34307	当存储卡没被格式化或有一个格式化出错发生时变为 ON。
文件写错误标志	A34308	写入文件有一个错误出现时变为 ON。
文件不能写标志	A34309	由于文件是写保护或没有足够的存储器空间，数据不能写入时变为 ON。
文件读错误标志	A34310	由于数据出错或包含了错的数据类型，数据不能读出时变为 ON。
无文件标志	A34311	由于特定文件不存在，数据不能读出时变为 ON。
文件存储器操作标志	A34313	在以下的任何情况下变为 ON： 使用 CMND （490）， CPU 单元处理 FINS 命令并传送给自身。 执行 FREAD （700）或 FWRITE （701）。 使用辅助区域控制位，程序正在被重写。 正在执行一个简单的备份操作。
存取文件标志	A34314	当文件数据正被存取时变为 ON
存储卡检测标志	A34315	存储卡已检测，标志位为 ON。 （EV1 前的 CS 系列 CS1 CPU 单元不支持）
传输项目数	A346 ～ A347	这些字指明将要传输的字或字段的数目 （32 位）。 当传输一个二进制的文件 （.IOM）时，这个数在每次读出一个字后减一。 当传输一个文本或 CSV 文件时，这个数在每次传输一个字段后减一。

CMND(490)：发布命令

可以使用 CMND（490）发布 FINS 命令到本地 CPU 单元，执行文件存储器操作，如：格式化或删除文件。

当发布文件存储器 FINS 命令到本地 CPU 时，在 CMND（490）的控制字中作出以下设定：

1,2,3...

1. 在 C+2 中设定目的网址 00 （本地网络）。
2. 在 C+3 中设定目的单元地址 00 （PLC 的 CPU 单元）和目的网点 00 （在本地网点内）。
3. 在 C+4 中设定重试次数 0 （重试次数设定是无效的，所以设定为 0）。

与文件存储器相关的 FINS 命令

有关 FINS 命令的信息请参考第 *5-2-2FINS 命令* 。

注 还有其它与文件存储器相关的可以执行的 FINS 命令，它们没有在下表中列出。有关 FINS 命令更详细内容请参阅*通信命令参考手册 （W342）*。

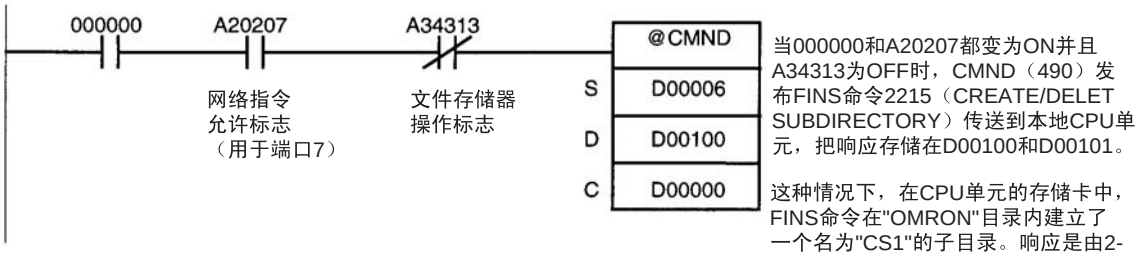
如果另一个 CMND（490）指令正在另一个 CPU 单元执行，CMND（490）就不能在本地 CPU 单元执行，执行 FREAD （700）或 FWRITE （701），程序将被辅助区域控制位操作所替换，或执行一个简单的备份操作。确认把文件存储器操作标志（A34313）作为一个常闭条件，以防止在另一个文件存储器操作正在进行时执行 CMND （490）指令。

对于本地 CPU 单元，如果 CMND （490）不能执行，出错标志将变为 ON。

相关的辅助位 / 字

名称	地址	操作
文件存储器操作标志	A34313	在以下的任何情况下变为 ON： • CPU 单元处理 FINS 命令并用 CMND （490）传送给自身。 • 执行 FREAD （700）或 FWRITE （701）。 • 使用辅助区域控制位，程序被重写。 • 执行一个简单备份操作
存储卡检测标志	A34315	存储卡已检测，标志位为 ON。（EV1 前的 CS 系列 CS1 CPU 单元不支持）

以下示例说明了如何使用 CMND （490）在存储卡中建立一个子目录。



当000000和A20207都变为ON并且A34313为OFF时，CMND（490）发布FINS命令2215（CREATE/DELET SUBDIRECTORY）传送到本地CPU单元，把响应存储在D00100和D00101。

这种情况下，在CPU单元的存储卡中，FINS命令在"OMRON"目录内建立了一个名为"CS1"的子目录。响应是由2-字节命令码（2215）和2--字节响应码所组成。

		15 8	7 0	
S:	D00006	22	15	命令码：2215Hex（CREATE/DELETE SUBDIRECTORY）
S+1:	D00007	80	00	磁盘号：8000Hex (存储卡)
S+2:	D00008	00	00	参数：0000 Hex (建立子目录)
S+3:	D00009	43	53	子目录名：CS1 □□□□□. □□□ （□：一个空格）
S+4:	D00010	31	20	
S+5:	D00011	20	20	
S+6:	D00012	20	20	
S+7:	D00013	2E	20	
S+8:	D00014	20	20	
S+9:	D00015	00	06	目录长度：0006 Hex （6个字符）
S+10:	D00016	5C	4F	目录路径：\ OMRON
S+11:	D00017	4D	52	
S+12:	D00018	4F	4E	

		15 8	7 0	
C:	D00000	00	1A	命令数据的字节数：001A Hex（26个字节）
C+1:	D00001	00	04	响应数据的字节数：0004 Hex (4个字节)
C+2:	D00002	00	00	目的地址：0000 Hex (本地网)
C+3:	D00003	00	00	00 Hex (本地网点)和00 Hex(CPU单元)
C+4:	D00004	07	00	被要求响应，通信端口7， 0 重试
C+5:	D00005	00	00	响应监控时间：FFFF Hex (6,553.5 s)

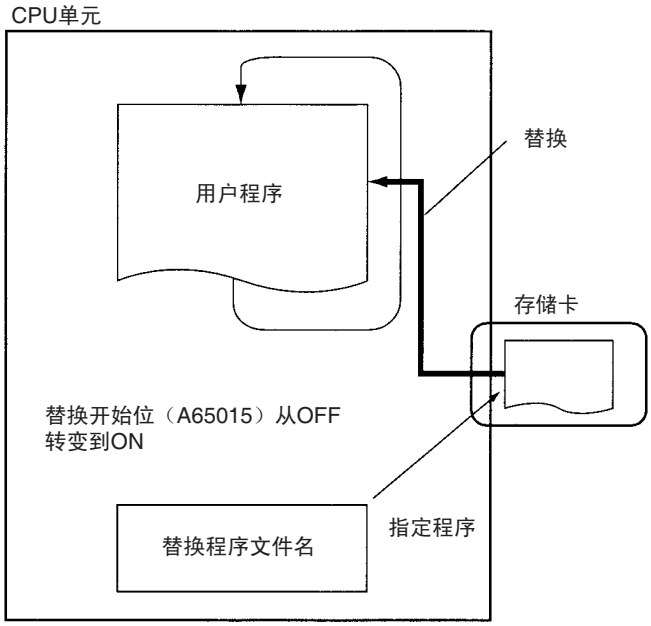
注 除了与文件存储器操作相关的指令，还有其它可以传送到本地 PLC 的 FINS 命令列在上表中。必须使用文件存储器操作标志，以防止同时也执行这些其它 FINS 命令。

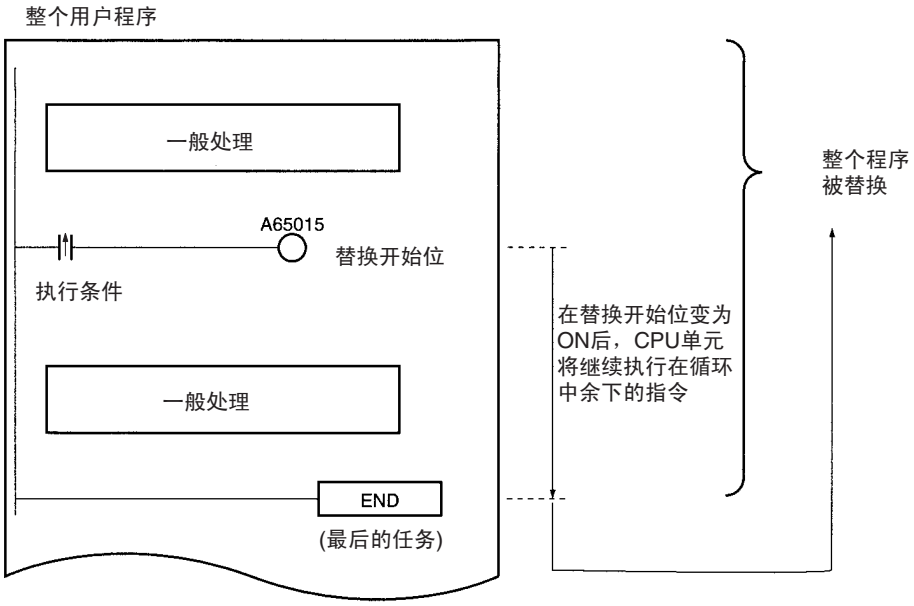
5-2-4 操作期间整个程序的替换

（EV1 前的 CS 系列 CS1 CPU 单元不支持）

过使替换开始位 （A65015）变为 ON，操作 （RUN 或 MONITOR 方式）期间可以替换整个程序。指定文件将从存储卡中读出，并且它将替换，即在当前循环结束时它将替换可执行的程序。为了在操作期间替换程序，替换程序密码

（在 A651 中）和程序文件名 （A654 ～ A657）必须预先复制好，指定文件必须在存储卡中存在。





通过从编程设备使替换开始位变为 ON，当程序执行停止 （PROGRAM 方式）时，也可替换程序。

注 从 EM 文件存储器不能读出替换程序文件。

在程序的任何位置（程序地址）可以使替换开始位（A65015）变为 ON。当替换开始位从 OFF 转变到 ON 时， CPU 单元将继续执行在循环中余下的指令。

在程序被替换期间将不执行程序。程序被替换后，操作将重新开始（就象 CPU 单元从 PROGRAM 方式切换到 RUN 或 MONITOR 方式那样

在每个循环的结束点（此时替换开始位已从 OFF 转变到了 ON，即在程序的最后一个任务 （END （001）执行以后）替换程序。

注 1. 经过程序替换以后，如果你想保持 I/O 存储器数据的状态，那么就使 IOM 保持位 （A50012）变为 ON。

经过程序替换以后，如果你想保持强制置位和强制复位位的状态，那么就使强制状态保持位 （A50013）变为 ON。

2. 在程序替换以前，如果 IOM 保持位 （A50012）变为 ON，程序替换以后，I/O 存储器位的状态将保持不变。请确认，外部负载将用相同的 I/O 存储器数据正确地运行。

同样，在程序替换以前如果强制状态保持位 （A50013）变为 ON，程序替换以后，强制置位和强制复位位的状态将保持不变。请确认，外部负载将用相同的强制置位和强制复位位正确地运行。

替换文件

在程序文件名称 （A654 ～ A657）中规定的程序文件将从存储卡中读出，并将在循环结束时 （此时替换开始位 （A65015）从 OFF 转变到 ON）替换现存的程序。

文件	文件名和扩展名	特定的替换文件名 (********)
程序文件	********.OBJ	在程序替换之前，将替换文件名写到 A654 ～ A657

程序替换的条件要求

在操作期间为了能替换程序，要求满足以下一些条件：

- 程序密码 （A5A5）已经写入 A651。
- 在程序文件名字 （A654 ～ A657）中规定的程序文件存在于存储卡的根目录中。
- 存储卡已被 CPU 单元检测。（A34315 ON）。
- 没有致命出错发生。
- 没有执行文件存储器操作。（A34313 OFF）。
- 数据没有写入程序区域。
- 访问权有效。（如：数据没有正在从 CX-Programmer 传输到 PLC）。

注 程序可以在任何一种操作方式下传输。

在程序替换期间的 CPU 操作

在程序替换期间， CPU 单元的操作将如下：

- 程序执行：停止
- 周期监控：无监控

在程序替换期间和以后，操作继续

当 IOM 保持位 （A50012）变为 ON，在以下存储器区域的数据将保持：CIO 区域，工作区域 （W），计时器完成标志 （T），变址寄存器 （IR），数据寄存器 （DR），和当前 EM Bank 号。

注 在程序替换期间，定时器当前值 （PV）将被清除。

传输程序时，如果 IOM 保持位为 ON，在程序替换之前正被输出的负载在替换以后将继续被输出。确认在程序替换以后，外部负载将正确地运行。

如果强制状态保持位（A50013）为 ON，程序替换后仍将保持强制置位和强制复位位的状态。

中断将被屏蔽

如果执行数据跟踪，它将被终止。
指令条件 （互锁，退出和块程序执行）将被初始化。
不管 IOM 保持位是 ON 还是 OFF，将初始化微分标志。

程序替换以后的操作

循环任务的状态取决于它们的操作开始特性。（它们的状态是跟原来应有的状况 （如果 PLC 从 PROGRAM 切换到 RUN/MONITOR 方式时它的状态）相同的）。

在程序从新执行后的一个循环，第一循环标志 （A20011）变为 ON。（它的状态是跟原来应有的状况 （如果 PLC 从 PROGRAM 切换到 RUN/MONITOR 方式时它的状况）相同的）。

程序替换所需时间

整个程序的大小	在 PLC 中设定外设服务时间	程序替换所需的大约时间
60 K 千步	默认 （4% 的周期时间）	6 s
250 K 千步		25 s

相关的辅助位 / 字

名称	地址	操作
文件存储器操作标志	A34313	在以下的任何情况下变为 ON： 使用 CMND （490） CPU 单元已将 FINS 命令传送到自身。 执行 FREAD （700）或 FWRITE （701）。 使用辅助区域控制位 （A65015）程序被重写。 执行一个简单的备份操作。
存储卡检测标志 （EV1 前的 CS 系列 CS1 CPU 单元不支持）	A34315	存储卡已检测，标志位为 ON。
IOM 保持位	A50012	当这个位为 ON，在程序替换中 I/O 存储器的内容保持不变。
强制状态保持位	A50013	当这个位为 ON，在程序替换中强制置位和强制复位的状态保持不变。
替换完成码 （EV1 前的 CS 系列 CS1 CPU 单元不支持）	A65000 ～ A65007	用于正常程序替换的码 （A65014 OFF）： 01 Hex：程序文件 （.OBJ）替换程序 用于未完成程序替换的码 （A65014 ON）： 00 Hex：发生一个致命出错 01 Hex：发生一个存储器出错 11 Hex：程序被写保护 12 Hex：在 A651 中的程序密码不正确 21 Hex：一个存储卡没安装 22 Hex：指定文件不存在 23 Hex：指定文件太长 （存储器出错） 31 Hex：以下的一种操作已执行： • 执行文件存储器操作 • 程序写入 • 操作方式改变
替换出错标志 （EV1 前的 CS 系列 CS1 CPU 单元不支持）	A65014	A65015 由 OFF 变为 ON 后，在替换程序期间产生一个出错时，标志位变为 ON。 当下个周期 A65015 再从 OFF 变为 ON 时，标志位变为 OFF。

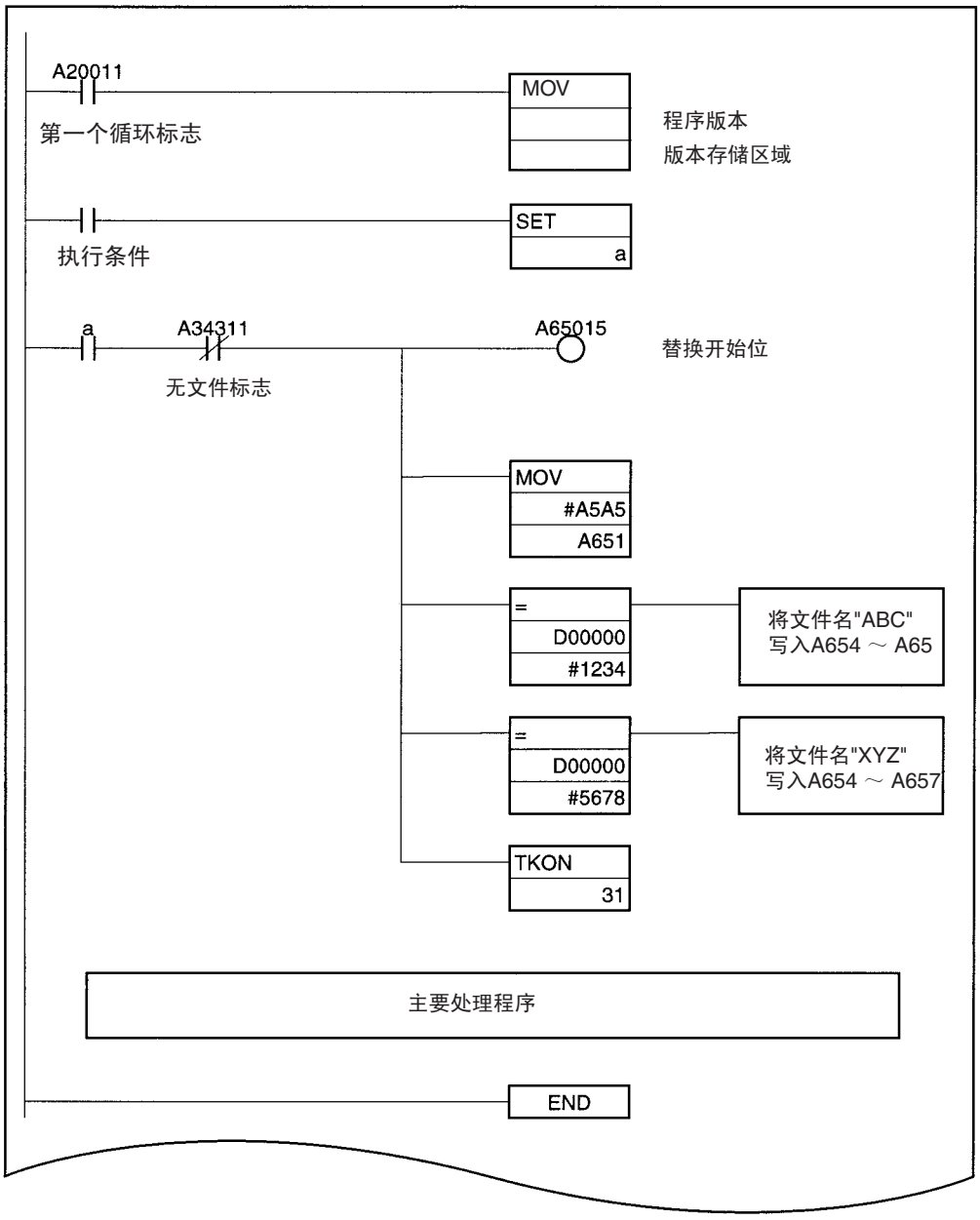
名称	地址	操作
替换开始位 （EV1 前的 CS 系列 CS1 CPU 单元不支持）	A65015	如果通过将程序密码 （A651）设定到 A5A5 Hex 使这个位起用，当这一位从 OFF 变为 ON 时，开始程序替换。在程序替换期间不能再次使这个位从 OFF 变为 ON。 当程序替换结束时 （正常或有一个出错）或电源接通时，这位自动变到 OFF。 这位的状态可以从编程设备， PT，或主计算机中读出，从而确定程序替换是否已完成。
程序密码（EV1 前的CS 系列CS1 CPU 单元不支持）	A651	将密码写入这个字，允许程序替换。 A5A5 Hex：允许替换开始位 （A65015）。 其它值：禁止替换开始位 （A65015）。 当程序替换结束时 （正常或有一个出错）或电源接通时，这个位自动变到 OFF。
程序文件名 （EV1 前的 CS 系列 CS1 CPU 单元不支持）	A654 ～ A657	开始程序替换以前，用ASCII的字写出替换程序文件的文件名。只要写出8个字符的文件名；扩展名OBJ将自动加上。从A654（最高位开始）依次写出字符。如果文件名少于8个字符，以空格码（20 Hex）添满余下的字节。在文件名内不能包括零字符或空格。 下表显示了程序文件ABC.OBJ的数据： 15 0 A654 41 42 A655 43 20 A656 20 20 A657 20 20

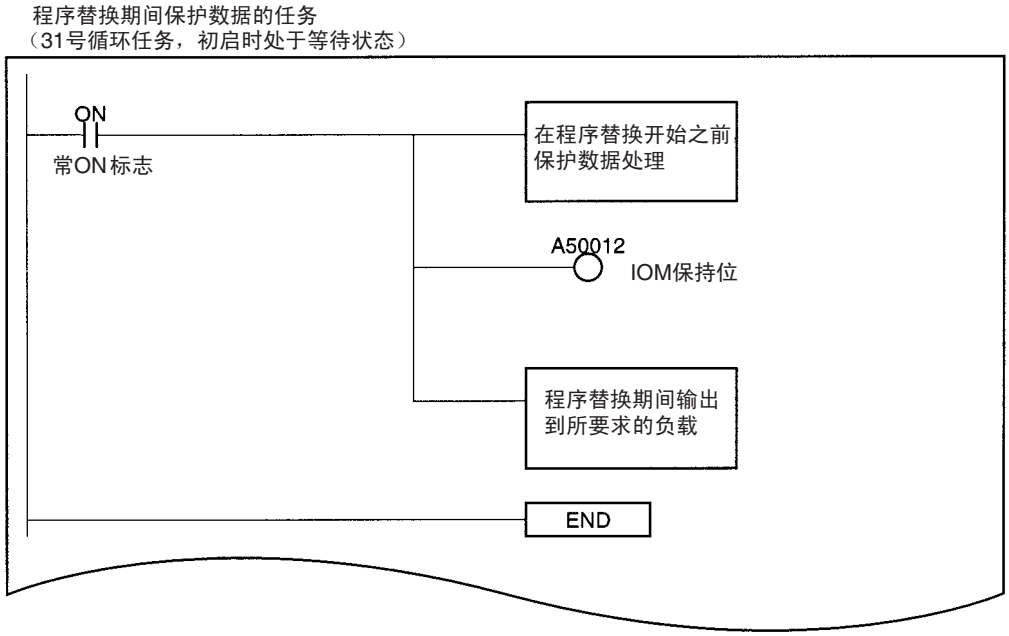
程序示例 1

将文件名 ABC.OBJ 和 XYZ.OBJ 存储在存储卡中，根据 D00000 的值选择一个程序或选择其它。当要选择 ABC.OBJ 时，将 D00000 设定为 #1234 ；或当要选定 XYZ.OBJ 时，将 D00000 设定为 #5678。

在程序替换或 IOM 保持位处理之前，开始和执行另一个任务，以完成所要求的处理。

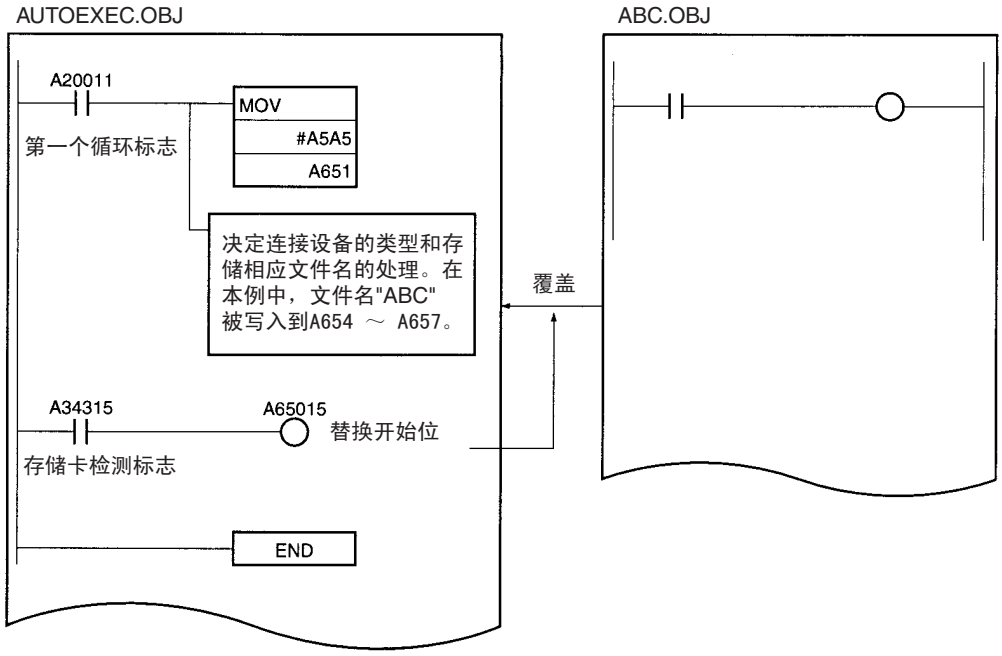
主要任务（0号循环任务）





程序示例 2

存储程序文件到几个设备中，初启时自动传输程序文件 （AUTOEXEC.OBJ）在存储卡中。当 PLC 接通，初启时自动传输的文件被读出，随后，不同设备的程序被一个程序文件替换。



5-2-5 在初启时自动传输

当电源接通时，用初启时自动传输功能将存储卡中的用户程序，参数，和 I/O 存储器数据读到 CPU 单元。
以下文件可以自动读到 CPU 单元存储器中。

注 不能使用这个功能来读 EM 文件存储器内容。

文件	文件名	初启时	自动传输的要求
程序文件	AUTOEXEC.OBJ	这个文件的内容被自动传输，并覆盖包括 CPU 单元任务属性的整个用户程序。	要求在存储卡
数据文件	AUTOEXEC.IOM	DM 字分配给特殊 I/O 单元， CPU 总线单元，以及内装板（仅限于 CS 系列）。 当电源接通时，这个文件的内容自动传输到在 D20000 开始的 DM 区域 （见注 1）	不要求在存储卡中
	ATEXECDM.IOM	通用 DM 字 当电源接通时，这个文件的内容自动传输到在 D00000 开始的 DM 区域 （EV 1 前的 CS 系列 CS1 CPU 单元不支持）（见注 1）	
	ATEXECE@.IOM	通用 DM 字 当电源接通时，这个文件的内容自动传输到在 E@_00000 开始的 EM 区域 （EV1 前的 CS 系列 CS1 CPU 单元不支持）	
参数区域文件	AUTOEXEC.STD	在 CPU 单元中，这个文件的内容自动传输，并覆盖所有的初始设定数据。	要求在存储卡

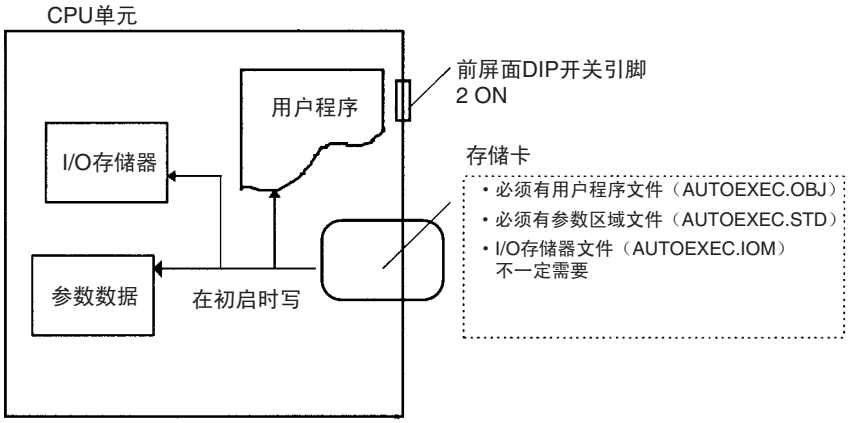
注 1. 如果包含在 AUTOEXEC.IOM 和 ATEXECDM.IOM 中的数据重叠，在 ATEXECDM.IOM 中的数据将覆盖任何从 AUTOEXEC.IOM 传输过来的重叠数据，因为 ATEXECDM.IOM 是稍后写入。

2. 程序文件 （AUTOEXEC.IOM）和参数文件 （AUTOEXEC.STD）必须在存储器卡中。没有这些文件，不能自动传输，并将产生一个存储器出错，A40115 （存储器出错标志：致命出错）将变为 ON。（I/O 存储器文件（AUTOEXEC.IOM）不必一定存在）。

3. 可以从编程工具 （手持编程器或 CX-Programmer）建立 AUTOEXEC.IOM， ATEXECDM.IOM，和 ATEXECE@.IOM 文件，分别带有除了 D20000， D00000，和 E@_00000 的开始地址。无任如何，数据是以正确开始地址开始写出，但不必指定其它的开始地址。

4. 如果 DIP 开关引脚 7 变为 ON，引脚 8 变为 OFF，使用简单的备份功能，即使引脚 2 也变为 ON，简单备份功能仍将进行。在这种情况下，BACKUP □□文件将传输到CPU单元，但在初启时，自动传输的文件将不传输（EV1 前的 CS 系列 CS1 CPU 单元不支持）。

5. 初启时，自动传输功能可以与程序替换功能同时使用。替换开始位（A65015）可以在程序中变为 ON，而这个程序在初启时被自动传输，被另一个程序替换。



步骤

1,2,3...

1. 断开 PLC 电源。
2. 接通 CPU 单元前屏面上的 DIP 开关引脚 2。确认引脚 7 和 8 都是 OFF。

   注　在初启自动传输功能进行过程中，简单备份功能仍将继续，所以确认引脚 7 和 8 都是 OFF。
3. 插入含有由 CX-Programmer 建立的用户程序文件 （AUTOEXEC.OBJ），参数区域文件 （AUTOEXEC.STD），和 /I/O 存储器文件（AUTOEXEC.IOM，ATEXECDM.IOM，和 ATEXECE@.IOM）存储卡。（程序文件和参数区域文件必须在存储卡中。 I/O 存储器文件可以选择）。
4. 接通 PLC 电源。

注　**在初启时自动传输失败**

如果在初启时自动传输失败，将出现存储器出错， A40115 变为 ON， CPU 单元将停止运行。如果出现一个错误，可断开电源，清除错误。（这种错误只能用断电才能清除）。

CPU 单元前屏面上的 DIP 开关

引脚	名称	设定
2	在初启时执行自动传输引脚	ON：在初启时执行自动传输 OFF：在初启时不执行自动传输
7 和 8	简单备份引脚	两条引脚都断开。

相关辅助字 / 位

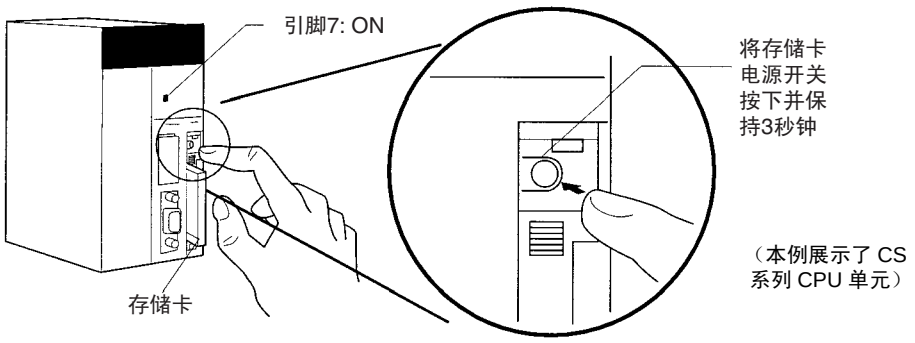
名称	地址	设定
存储器出错标志 (Fatal error)	A40115	在存储器中出现错误时，或当电源接通时从存储卡自动传输出现错误时 （在初启时自动传输）变为 ON。 CPU 单元停止运行，并且在 CPU 单元前部的 ERR/ALM 指示器闪亮。 注：在初启时自动传输期间如果出现错误， A40309 将变为 ON。（此时不能清除这个错误）。
存储卡初启传输出错标志	A40309	在初启时选定自动传输及自动传输期间出现错误时，变为ON。（DIP 开关引脚 2 为 ON）。如果传输出错或存储卡没有安装，就出现一个错误。 注：通过断开电源可以清除这个错误。（电源接通期间不能清除这个错误）。

5-2-6 简单备份功能

EV1 前的 CS 系列 CS1 CPU 单元不支持这个功能。

数据从 CPU 单元备份到存储卡

为了备份数据，可以在 CPU 单元的 DIP 开关上使引脚 7 为 ON，将存储卡电源开关按下并保持 3 秒钟。备份功能将自动建立备份文件并将它们写入存储卡。备份文件包含程序，参数区域数据和 I/O 存储器数据。在任何操作方式下可以执行这个功能。



将数据重新从存储卡存储到 CPU 单元

将备份文件重新存储到 CPU 单元，检查：引脚 7 是 ON，断开 PLC 的电源然后再接通。备份文件包含的程序，参数区域数据和 I/O 存储器数据将从存储卡读到 CPU 单元。

注 1. 备份功能将替换初启功能的自动传输功能，因此当 PLC 接通时，即使 DIP 开关的引脚 2 为 ON，备份文件仍将被读到 CPU 单元。

2. 如果 DIP 开关的引脚 1 变为 ON （写保护程序存储器），数据将不会从存储卡中读到 CPU 单元。

3. 通过备份功能从存储卡中读出备份文件时，除非已在辅助区域和 PLC 设定中进行了必要的设定，I/O 存储器的状态和强制置位 / 强制复位的状态将被清除。

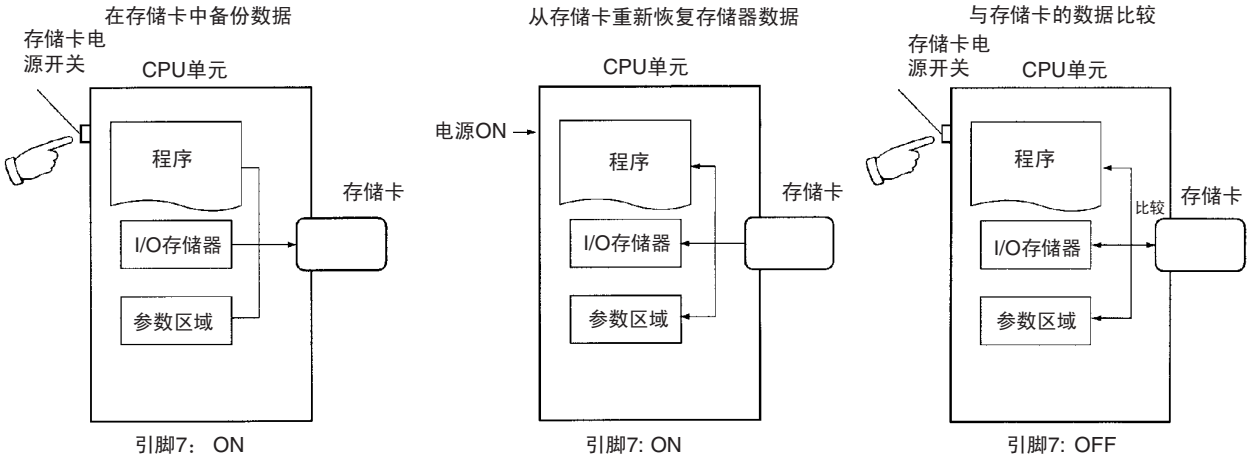
   当备份文件被写入时，如果 IOM 保持位 （A50012）为 ON，并且 PLC 设置设定为在初启时保持 IOM 保持位状态，当数据从存储卡中读出时，I/O 存储器数据状态将保持。

当备份文件被写入时，如果强制状态保持位（A50013）为 ON，并且 PLC 设置设定为在初启时保持强制状态保持位状态，当数据从存储卡中读出时，强制置位和强制复位位的状态将保持。

4. 执行简单备份操作以后， CS1-H， CJ1-H，或 CJ1M CPU 单元将保持 PROGRAM 方式，当电源保持接通，操作方式不能改变到 MONITOR 或 RUN 方式。完成备份操作以后，将 PLC 单元的电源断开，改变引脚 7 的设定，然后将电源重新接通。

存储卡和 CPU 单元中的数据比较

为了将存储卡中的备份文件与 CPU 单元中的数据相比较，将 CPU 单元的 DIP 开关上的引脚 7 OFF，将存储卡电源开关按下并保持 3 秒钟。备份功能将存储卡中的程序，参数区域数据和 I/O 存储器数据与 CPU 单元中的相应数据进行比较。此功能在任何操作方式下都可以执行。



下表给出了简单备份操作的小结。

备份操作	引脚状态	过程
	引脚 7	
从 CPU 单元将数据备份到存储卡	ON	将存储卡电源开关按下并保持 3 秒钟
从存储卡将数据重新存储到 CPU 单元	ON	将 PLC 电源断开并重新接通 （见注 1）
比较 CPU 单元和存储之间的数据	OFF	将存储卡电源开关按下并保持 3 秒钟

注 1. 有关读、写和比较操作结果的详细内容，请参考*第223页使用指示器验证备份操作*。

2. 有关存储卡备份操作所需时间的说明，请参考*第 5-3-2 章存储卡的操作过程*。

备份文件

数据文件

文件名和扩展名	数据区域和存储地址的范围		从 I/O 存储器备份到存储卡（建立文件）	从存储卡重新存储到 I/O 存储器	存储卡与 I/O 存储器比较		重新存储数据时所需的文件
CPU 单元	CS/CJ				CS1/CJ1	CS1-H/CJ1-H	
BACKUP.IOM	DM	D20000 ～ D32767	有	有	有	---	需要在存储卡中
BACKUPIO.IOR	CIO	0000 ～ 6143（包括强制位状态）	有	---[4]	有	---	需要在存储卡中
	WR	W000 ～ W511（包括强制位状态）	有	---[4]	有	---	
	HR	H000 ～ H511	有	有	有	---	
	AR	A000 ～ A447	有	---	---	---	
		A448 ～ A959	有	有	有	---	
	定时器[1]	T0000 ～ T4095	有	有[4]	有	---	
	计数器[1]	C0000 ～ C4095	有	有	有	---	
BACKUPDM.IOM	DM	D00000 ～ D19999	有	有	有	---	需要在存储卡中
BACKUPE@.IOM[2,3]	EM	E@_00000 ～ E@_32767	有	有	有	---	需要在存储卡中（必须与CPU单元相配）

注 1. 完成标志和 PV 已被备份。

2. □代表的是 Bank 号， Bank 数取决于所使用的 CPU 单元。

   在存储卡中的 BACKUPE@.IOM 文件被重新存储到 CPU 单元时，文件读出的次序是以在 CPU 单元中的 0 号 Bank 开始，至 CPU 单元最大号 Bank 结束。如果备份的 Bank 数超过了 CPU 单元中的存储单元数，其余 BACKUPE@.IOM 文件将不被读出。相反地，如果备份的 Bank 数小于 CPU 单元中的 Bank 数， CPU 单元中那些余下的 EM Bank 将保持不变。

   如果一个 BACKUPE@.IOM 文件消失（例如：0，1，2，4，5，6），只有连续文件将被读出。在这种情况下，数据仅被读到 0 号， 1 号和 2 号存储单元。

3. EM 区域数据将作为二进制数据备份。已经转变成文件存储器的 EM Bank 内容将和未转变的 EM Bank 内容一起被备份。

   仅当 BACKUPE@.IOM 文件连续时，并且备份 EM Bank 数与 CPU 单元的 Bank 数相配时， EM 文件存储器可以被存储到另一个 CPU 单元的 EM 区域。如果 BACKUPE@.IOM 文件不连续，或 EM Bank 数与 CPU 单元的 Bank 数不相配时， EM 文件存储器将复原到它的未格式化状态，这样在文件存储器中文件将失效。（常规 EM 区域 Bank 内容将被正常读出）。

4. 通常情况下，当 PLC 接通电源及 BACKUPIO.IOR 是从存储卡中读出时，CIO 区域，WR 区域，定时器完成标志，定时器 PV 的内容和强制置位 / 强制复位位的状态将被清除。

   在写入备份文件时，如果 IOM 保持位 （A50012）为 ON，并且 PLC 设置设定在初启时保持 IOM 保持位的状态，那么当数据是从存储卡中读出时，I/O 存储器数据的状态将保持。

当写入备份文件时，如果强制状态保持位（A50013）为 ON，并且 PLC 设置设定在初启时保持强制状态保持位状态，那么当数据从存储卡中读出时，强制置位和强制复位位的状态将保持。

程序文件

文件名和扩展名	内容	从 I/O 存储器备份到存储卡（建立文件）	从存储卡重新存储到 I/O 存储器	存储卡与 I/O 存储器比较	重新存储数据时所需的文件
CPU 单元	CS/CJ				
BACKUP.OBJ	整个用户程序	有	有	有	需要在存储卡中

参数文件

文件名和扩展名	内容	从 I/O 存储器备份到存储卡（建立文件）	从存储卡重新存储到 I/O 存储器	存储卡与 I/O 存储器比较	重新存储数据时所需的文件
CPU 单元	CS/CJ				
BACKUP.STD	PLC 设置 I/O 登记表 路由表 CPU 总线单元设置等等	有	有	有	需要在存储卡中

单元 / 板备份文件 （仅限于 CS1-H、 CS1D、 CJ1-H 或 CJ1M CPU 单元）

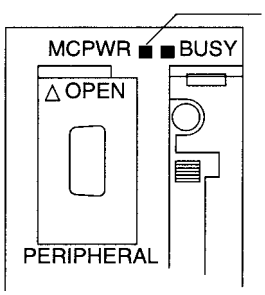
文件名和扩展名	内容	从 I/O 存储器备份到存储卡（建立文件）	从存储卡重新存储到 I/O 存储器	存储卡与 I/O 存储器比较	重新存储数据时所需的文件
CPU 单元	仅限于 CS1-H, CJ1-H 或 CJ1M CPU 单元				
BACKUP@@.PRM （这里 @@ 是将被备份的单元 / 板的单元地址）	从单元或具有特定单元地址的板中备份数据 （特定内容取决于单元或板）	有	有	有	需要在存储卡中 （见注 2）

注 1. 单元地址如下：
CPU 总线单元：单元号 +10 Hex
特殊 I/O 单元：单元号 +20 Hex
内装板：E1 Hex

2. 当数据从存储卡传输到 I/O 存储器时，即使这个文件丢失， CPU 单元中也不产生出错；但是如果数据没有重新存储，在单元或板中将产生一个出错。有关单元或板的出错的详细内容请参考特殊单元或板的操作手册。

用指示器验证备份操作

存储卡电源状态 （MCPWR）指示器表示了简单备份操作是否正常完成。



（本例展示的是 CS 系列 CPU 单元）

备份操作	正常完成 （见注 1）	发生错误	
	MCPWR 状态	MCPWR 状态	出错
从 CPU 单元备份数据到存储卡	亮→在存储卡电源开关被压下期间保持亮→闪亮一次→在写的时候亮→数据写完后暗	亮→在存储卡电源开关被压下期间保持亮→保持闪亮→在写的时候亮→数据写完后暗	在下列错误出现时不能建立文件： 存储卡没有足够容量（见注 2） CPU 单元的存储器出错 I/O 总线出错 （当数据写到单元或板上，仅限于 CS1-H， CS1D 或 CJ1-H CPU 单元）
重新将数据从存储卡到 CPU 单元	当电源接通时亮 → 闪亮一次 → 在读的时候亮 → 数据读完后暗	当电源接通时亮 → 闪亮五次 → 变成暗	在下列错误出现时不能读出数据： 存储卡中的程序超过 CPU 单元的容量 所要求的备份文件在存储卡中不存在 由于写保护程序不能写入（DIP 开关的引脚 1 为 ON）。
		当电源接通时亮 → 闪亮一次 → 在读的时候亮 → 闪亮三次 → 数据读完后暗	警告：下列错误出现时仍将读出数据： EM 文件和 CPU 单元 EMBank 不相配 （不连续 Bank 号或最大 Bank 号不相配）

备份操作	正常完成 （见注 1）	发生错误	
	MCPWR 状态	MCPWR 状态	出错
比较 CPU 单元和存储卡之间的数据	亮→在存储卡电源开关被压下期间保持亮→ 闪亮一次→在写的时候亮→数据比较后暗	亮→在存储卡电源开关被压下期间保持亮→保持闪亮→当存储卡电源开关被压下时亮	下列比较错误可能出现 （见注 3）： 存储卡和 CPU 单元数据不一致 所要求的备份文件在存储卡中不存在 EM 文件和 CPU 单元 EM Bank 不相配 （不连续 Bank 号或最大 Bank 号不相配） CPU 单元的存储器出错 I/O 总线出错 （当比较数据写到单元或板，仅限于 CS1-H， CS1D，或 CJ1-H CPU 单元）
为三个备份操作所共有	---	读： 闪亮五次 → 变暗 写成比较： 保持闪亮 → 当存储卡电源开关被压下时亮	存储卡存取出错 （格式化出错或读 / 写出错）

注 1. 备份操作正常完成，MCPWR 指示器变成 OFF 时，存储卡的电源将关断。如果再次使用存储卡，按下存储卡电源开关供电，并执行所要求的操作。

2. 当从 CS1-H，CS1D，CJ1-H 或 CJ1M CPU 单元中对一个简单的备份操作写入数据时，在 A397 （简单备份写容量）可以检测出存储卡容量不足出错。在写操作执行以后，如果 A397 中含有任何除了 0000 Hex 的值，这个值将指出在存储卡中所要求的容量 （以千字节为单位）。

3. 使用 CS1-H，CS1D，CJ1-H 或 CJ1M CPU 单元，用于单元和板的备份文件也将作比较。

相关的辅助位 / 字

名称	地址	操作
文件存储器操作标志	A34313	在以下的任何执行情况下为 ON，当执行完成时变为 OFF。 • 存储卡检测 • 对于本地 CPU 单元执行 CMND 指令 • FREAD/FWRITE 指令 • 通过特定控制位作程序替换 • 简单备份操作 当这个标志变为 ON 时，不可能将数据写入存储卡或验证存储卡的内容。
EM 文件存储器开始 Bank	A344	当 CPU 单元开始从存储卡中读出时，它参考这个值。如果 BACKUPE □ .IOM 文件的最大 EM Bank （从 0 计的最大连续存储单元号）与 CPU 单元的最大 Bank 相匹配，基于这个字的值， EM 区域将被格式化。如果最大 EM Bank 不匹配， EM 区域将恢复到它未格式化的状态。
网络通信指令允许标志（仅限于 CS1-H,CJ1-H,CJ1M 或 CS1D CPU 单元）（见注）	A20200 ～ A20207	• 当写或比较存储卡数据开始时，标志位变为 OFF。 • 当写或比较存储卡数据已经完成时，标志位变为 ON。 如果所有的网络通信指令允许标志变为 OFF，当存储卡写或比较操作开始，并且如果有一个错误出现那么单元和板的数据将不能写或比较。
网络通信完成码（仅限于 CS1-H,CJ1-H,CJ1M 或 CS1D CPU 单元）（见注）	A203 ～ A210	当执行存储卡写或比较操作时，提供单元或板的通信结果。
网络通出错标志（仅限于 CS1-H,CJ1-H,CJ1M 或 CS1D CPU 单元）（见注）	A21900 ～ A21907	• 当执行存储卡写或比较操作，单元或板通信出错发生时，变为 ON。 • 当执行存储卡写或比较操作，单元或板通信无出错发生时，保持 OFF （或变成 OFF）。
简单备份写容量（仅限于 CS1-H,CJ1-H,CJ1M 或 CS1D CPU 单元）（见注）	A397	当对一个简单的备份操作写失效时，以千字节提供存储卡所需求的数据容量，指明由于容量不足发生了写出错。 0001 ～ FFFF Hex：写出错 （指出要求的存储卡容量 （1 到 65,535 千字节之间））（当成功地执行写操作时，清除为 0000 Hex） 0000 Hex：写正常完成。

注 这些标志与 CS1-H，CJ1-H，CJ1M，或 CS1D CPU 单元有关，因为在对存储卡进行写或比较数据时 CPU 单元将自动使用现有的通信端口。

单元和板数据备份

这个功能仅限于 CS1-H,CJ1-H,CJ1M 或 CS1D CPU 单元。

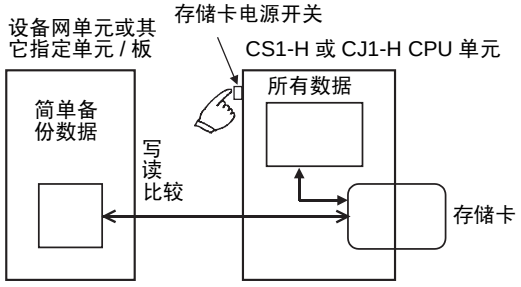
介绍

对于简单备份操作通过 CS1 和 CJ1 CPU 单元从 CPU 单元备份以下数据：

用户程序，参数区域，整个 I/O 存储器。除了以上数据，从 CS1-H， CJ1-H，CJ1M，或 CS1D CPU 单元也可备份以下数据：安装在 PLC 上的特殊单元和板中的数据。

概要

当简单备份操作用于 CS1-H，CJ1-H，CJ1M 或 CS1D CPU 单元时，单元 / 板备份文件包含写入存储卡中的指定单元和板的数据。对于每一个单元和板的数据分别备份。



应用

这个功能可以用于对整个 PLC 备份数据，包括 CPU 单元，设备网单元，串行通信单元 / 板，等。它也可以用于单元替换。

单元 / 板备份文件

使用文件：BACKUP □□ .PRM 使每个单元和板的数据存储在存储卡中。这里 " □□ " 是以十六进制为单位的单元或板的单元地址。

注 单元地址如下：
CPU 总线单元：单元号 +10 Hex
特殊 I/O 单元：单元号 +20 Hex
内装板：E1 Hex

当从存储卡读出或比较存储卡数据时，也可以使用这些文件。

可应用的单元和板

对于要备份数据的单元和板，单元和板必须也支持备份功能。有关支持的详细内容请参阅单元和板的操作手册。
以下是 2001 年 7 月制造的单元和板，支持备份功能。

单元 / 板	型号	备份数据 （仅限于 CS1-H 或 CJ1-H CPU 单元）
设备网单元	CS1W-DRM21-V1 CJ1W-DRM21	设备参数 （所有数据在单元中的 EEPROM 中）。 （虽然这是和由单元或设备网配置 （2.0 版本）支持的备份功能从存储卡得到的备份数据相同，但是没有文件兼容性）。
串行通信单元	CS1W-SCU21-V1 CJ1W-SCU41	协议宏数据 （包括标准系统协议和在单元或板的闪存器中用户定义的协议）
串行通信板	CS1W-SCB21-V1	
	CS1W-SCB41-V1	

注 上面所列的单元和板的数据对于简单备份操作将自动备份，没有设定可用于是否包括这些数据。

步骤

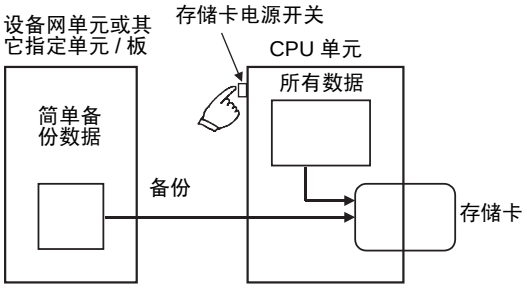
不管数据是否从指定的单元和板中进行备份，简单备份操作的步骤是相同的（包括写，读，和比较）。

■ 备份数据

1,2,3... 1. 在 CPU 单元的 DIP 开关上将引脚 7 接通 （ON）。

2. 将存储卡电源开关按下，并保持 3 秒钟。

单元和板的备份数据将在一个文件中建立，并与其它备份数据一起存储在存储卡中。



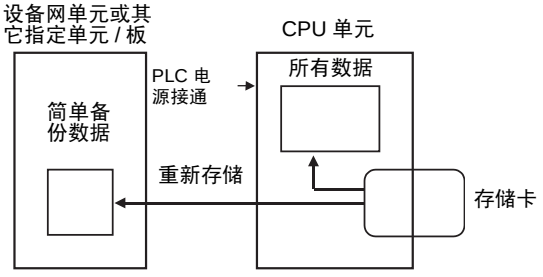
按下电源开关时，MCPWR 指示器将闪亮一次，而在写操作期间一直亮着。如果写正常完成，则变成 OFF。

■ 重新存储数据

1,2,3... 1. 在 CPU 单元的 DIP 开关上将引脚 7 接通 （ON）。

2. 将 PLC 接通电源。备份文件将重新存储到单元和板中。

单元和板的备份数据将重新从存储卡存储到单元和板中。



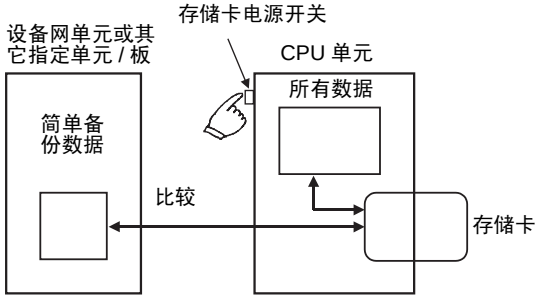
当电源接通时， MCPWR 指示器将闪亮一次，而在读操作期间一直亮着，如果读操作正常完成，则变成 OFF。

■ 比较数据

1,2,3... 1. 在 CPU 单元的 DIP 开关上将引脚 7 断开 （OFF）。

2. 将存储卡电源开关按下，并保持 3 秒钟。

在存储卡中的备份数据将与单元和板中的数据比较。



按下电源开关时， MCPWR 指示器将闪亮一次，而在比较操作期间一直亮着，如果比较正常完成且数据相同，则变为 OFF。

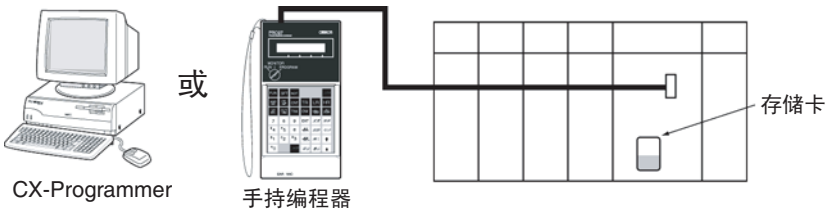
注 在进行以上操作之前，确认单元和板运行正常。如果单元和板运行不正常，就不能执行写，读，和比较操作。

5-3 使用文件存储器

5-3-1 初始化介质

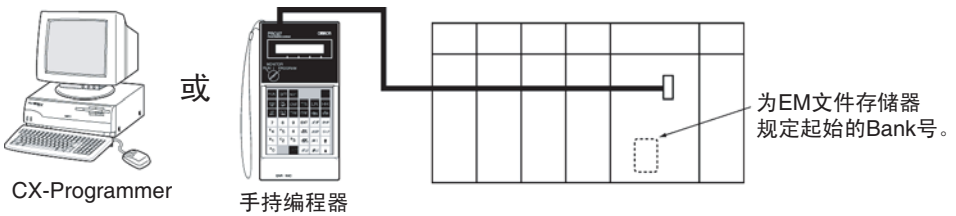
存储卡

1,2,3... 1. 使用编程工具 （如：手持编程器）使存储卡初始化。

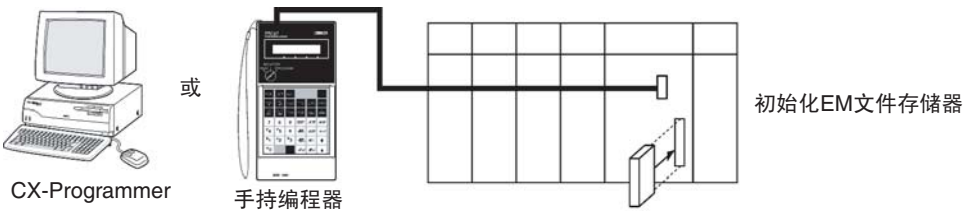


EM 文件存储器

1,2,3... 1. 使用编程工具 （如：手持编程器），在 PLC 设置中设定 EM 文件存储器设定，起用 EM 文件存储器，然后为 EM 文件存储器设定规定的 Bank 号（从 0 ～ C Hex）。



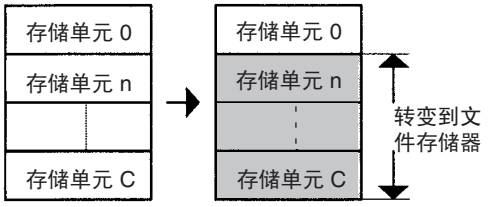
2. 使用 FINS 命令或除了手持编程器之外的编程工具来初始化 EM 文件存储器。



初始化专用的 EM 文件存储器

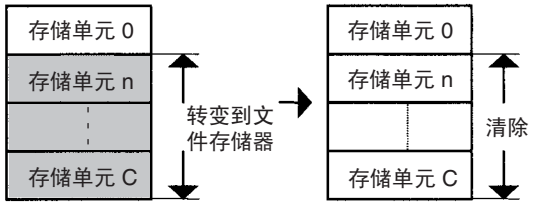
一个特定的 EM Bank 可以从普通的 EM 转变成文件存储器。

注 CJ 系列 CPU 单元的最大存储单元号是 6 。



1. 在PLC设置中设定n。
2. 在n开始使用编程工具或FINS命令格式化。
3. “n”存储在A344.

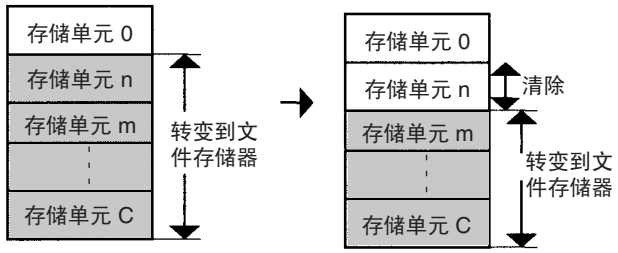
用于文件存储器 EM 可以重新存储到普通 EM 状态。



1. 在PLC设置中设定n。
2. 如果编程工具或FINS命令用于格式化，在n点开始的存储器将清除到0000 Hex。
3. FFFF Hex将存储在A344，用于指明那里没有EM文件存储器。

注：1. 在这个时候当前的任何文件数据将被删除。
2. 对于CJ系列CPU单元只能规定0～6号Bank。

用于文件存储器的开始 Bank 号可以改变。



1. 在PLC设置中将n改到m。
2. 使用编程工具或FINS 将存储单元开始点改到m点用于文件存储器。

注：Bank0至m～1将被清除为0000 Hex。

3. m将存储在A344。

注：1. 在这个时候当前的任何文件数据将被删除。
2. 对于CJ系列CPU单元只能规定0～6号Bank。

PLC 设置

地址	名称	说明	初始设定
136	EM 文件存储器开始 Bank	0000 Hex：无 0080 Hex：在 0 号存储单元开始 008C hex：C 号存储单元 从指定存储单元号开始的 EM 区域将转变成文件存储器。 （对于 CJ 系列 CPU 单元只能规定 0 ～ 6 号存储单元）	0000 Hex

相关特殊辅助中继站

名称	地址	说明
EM 文件存储器开始 Bank	A344	在那个时候，真正开始 EM 文件存储器区域的 Bank 号将被存储。从开始到最后的 Bank 号之间的 EM 文件将被转变成文件存储器。 FFFF Hex 将指明那里没有 EM 文件存储器。

使用 CX-Programmer 读 / 写符号表和注释

使用以下步骤将建立于 CX-Programmer 的符号或注解传递到存储卡或 EM 文件存储器 （和从存储卡或 EM 文件存储器中传出）。

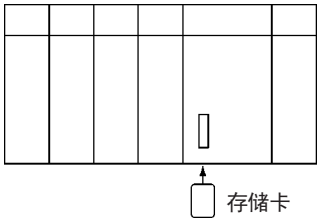
1,2,3...

1. 将格式化后的存储卡放入 CPU 单元或格式化后的 EM 文件存储器。
2. 将 CX-Programmer 联机。
3. 从 PLC *菜单*选择传送，随后到 *PLC 或从 PLC*。
4. 选择符号或注解作为数据传递。

5-3-2 存储卡操作步骤

使用编程工具

1,2,3... 1. 将存储卡插入 PLC 单元中。

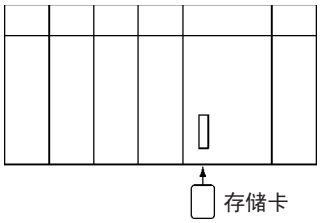


2. 使用编程工具初始化存储卡。

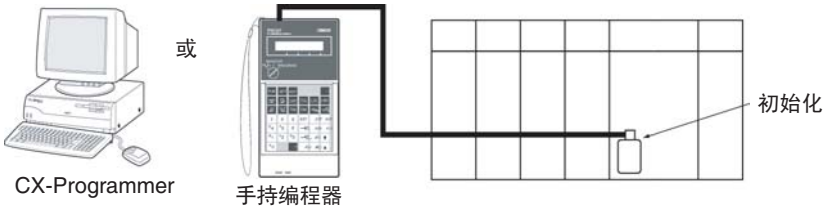
3. 使用编程工具命名 CPU 单元的数据（用户程序，I/O 存储器，参数区域），然后将数据保存到存储卡中。（使用编程工具将存储卡文件读到 CPU 单元）。

在初启时自动传递文件

1,2,3... 1. 将存储卡插入 PLC 单元中。（已经初始化）

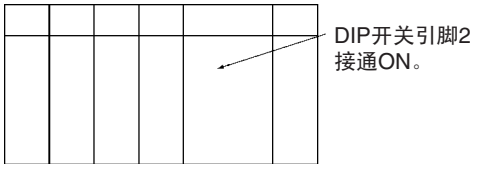


2. 使用编程工具将在初启时自动传递的文件写入存储卡。这些文件包括程序文件 （AUTOEXEC.OBJ），参数区域文件 （AUTOEXEC.STD），和 I/O 存储器文件 （AUTOEXEC.IOM 或 ATEXEC □□ .IOM）。



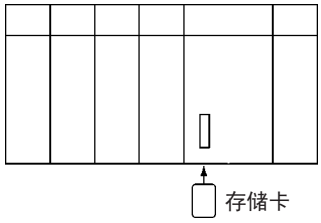
注 用户程序和参数区域文件必须在存储卡中。

3. 将 PLC 电源断开 （OFF）。
4. 将 DIP 开关引脚 2 接通 （ON）（在初启时自动传递）。



注 如果引脚 7 ON 并且引脚 8 OFF，允许执行备份功能并将替代初启时的自动传递功能。（将引脚 7 和 8 断开 （OFF），在初启时执行自动传递）。

5. 将存储卡插入 PLC 单元中。



6. 将 PLC 电源接通 （ON），读取文件。

使用 FREAD(700)/FWRIT(701)/CMND(490)

1,2,3...

1. 将存储卡插入 PLC 单元中。（已经初始化）
2. 在指定 I/O 存储器区域，使用 FWRITE（701）命名文件，并将文件保存到存储卡。

注 包含有 TXT 或 CSV 数据文件的存储卡可被安装到带有 HMC-AP001 存储卡适配器的个人计算机的 PLC 卡槽中，使用标准微软视窗功能将数据文件读入扩展页程序。（EV1 前的 CS 系列 CS1 CPU 单元不支持）。

3. 使用 FREAD （700）将存储卡中的文件读到 CPU 单元的 I/O 存储器中。

用 CMND（490）指令发布 FINS 命令到本地 CPU 单元，可以执行存储卡文件操作。（EV1 前的 CS 系列 CS1 CPU 单元不支持）。

操作期间替换程序

1,2,3...

1. 将存储卡插入 PLC 单元中。（已经初始化）
2. 在 A651 中写入程序密码 （A5A5 Hex），在 A654 到 A657 中写入程序文件名。
3. 将替换开始位 （A65015）由 OFF 变到 ON。

简单备份功能

有 3 种备份操作：将数据备份到存储卡，从存储卡重新存储数据，与存储卡比较数据。

将 CPU 单元的数据备份到存储卡

1,2,3...

1. 将存储卡插入 PLC 单元中。（已经初始化）
2. 把 CPU 单元的 DIP 开关上引脚 7 接通 （ON），引脚 8 断开 （OFF）。
3. 将存储卡电源开关按下，并保持 3 秒钟。
4. 确认 MCPWR 指示器闪亮一次然后熄灭。（备份数据期间如果有其它变化，表示发生了错误）。

将存储卡数据重新存储到 CPU 单元

1,2,3...

1. 将含有备份文件的存储卡插入 PLC 单元中。
2. 把 CPU 单元的 DIP 开关上引脚 7 接通 （ON），引脚 8 断开 （OFF）。
3. 当 PLC 接通时，将重新存储备份文件
4. 确认 MCPWR 指示器闪亮一次后熄灭。（重新存储备份数据期间如果有其它变化，表示发生了错误）。

比较存储卡和 CPU 单元中的数据

1,2,3...

1. 将含有备份文件的存储卡插入 PLC 单元中。
2. 把 CPU 单元的 DIP 开关上引脚 7 断开 （OFF），引脚 8 接通 （ON）。
3. 将存储卡电源开关按下，并保持 3 秒钟。
4. 如果 MCPWR 指示器闪亮一次后熄灭，表示数据相配。

注 写或比较数据期间，如果有出错， MCPWR 指示器将闪亮。当存储卡电源开关被按下， MCPWR 指示器不再闪烁而呈点亮状态。

下表给出了在 RUN 方式下，周期为 10 ms 的一个 20 千步的程序备份操作时所需要的时间：

方式	备份	重新存储	比较
PROGRAM	约 50 s	约 30 s	约 7 s
RUN	约 5 min	约 2 min	约 7 s

下表给出了在 RUN 方式下，周期为 10 ms 的一个 30 千步的程序备份操作时所需要的时间：

方式	备份	重新存储	比较
PROGRAM	约 50 s	约 30 s	约 7 s
RUN	约 5 min 30 s	约 2 min 40 s	约 7 s

下表给出了在 RUN 方式下，周期为 12 ms 的一个 250 千步的程序备份操作时所需要的时间：

方式	备份	重新存储	比较
PROGRAM	约 1 min 30 s	约 1 min 30 s	约 20 s
RUN	约 13 min	约 7 min 30 s	约 20 s

建立变量表和注释文件

在存储卡或 EM 文件存储器中，使用以下 CX-Programmer 操作步骤，建立变量表文件或注解文件。

1,2,3...

1. 将格式化后的存储卡插入 CPU 单元或格式化 EM 文件存储器。
2. 将 CX-Programmer 联机。
3. 从 PLC *菜单*选择传送，随后到 *PLC 或从 PLC*。
4. 选择符号或注解作为数据传递。

注 如果存储卡安装在 CPU 单元中，数据只能在存储卡中传递。（不能用EM 文件存储器）。

5-3-3 EM 文件存储器的操作过程

使用编程工具

1,2,3...

1. 使用 PLC 设置，指定转变到文件存储器的 EM 起始存储单元。
2. 使用编程工具初始化 EM 文件存储器。
3. 使用编程工具命名 CPU 单元数据 （用户程序， I/O 存储器。参数区域），然后将数据存放到 EM 文件存储器中。
4. 使用编程工具将 EM 文件存储器中的数据读到 CPU 单元。

使用 FREAD(700)/FWRIT(701)/CMND(490)

1,2,3...

1. 使用 PLC 设置，指定转变到文件存储器的 EM 起始存储单元。
2. 使用编程工具初始化 EM 文件存储器。
3. 使用 FWRIT （701）命名 I/O 存储器指定区域中的文件，然后将文件存放到 EM 文件存储器中。
4. 使用 FREAD （700）将 EM 文件存储器中的文件读到 CPU 单元的 I/O 存储器中。

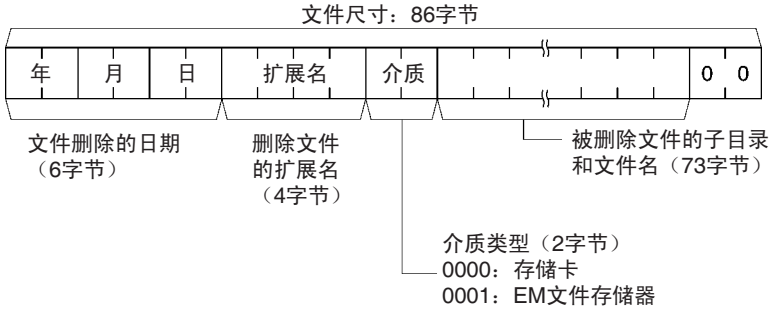
用 CMND（490）指令发布 FINS 命令到本地 CPU 单元，就可以执行 EM 文件存储器操作。

访问文件存储器期间电源中断

在 CPU 访问文件存储器 （存储卡或 EM 文件存储器）期间，如果电源中断，正被更新的文件可能不能正确地重写。此时，受其影响的文件在下一次电源接通时被系统自动删除。相应的文件删除通告标志（存储卡为 A39507，EM 文件存储器为 A39506）将变为 ON。在下一次电源断开时，这个标志将变为 OFF。

当一个文件被删除时，在存储卡或 EM 文件存储器的根目录中将建立一个删除记录文件 （DEL_FILE.IOM）。用 CX-Programmer 或 FREAD （700）可以阅读这个删除记录文件，并检查以下信息：被删除文件的数据，存在的文件存储器 （介质）的类型，子目录，文件名，和扩展名。当需要时，重新建立或重新复制被删除的文件。

下图表示了删除记录文件的结构。



第 6 章
高级功能

本章对以下的一些高级功能进行详细地描述：周期 / 高速处理功能，变址寄存器功能，串行通信功能，启动和维护功能，诊断和调试功能，编程工具功能，以及基本的 I/O 单元输入响应时间设定。

6-1 循环时间 / 高速处理

本节中将描述下列功能：

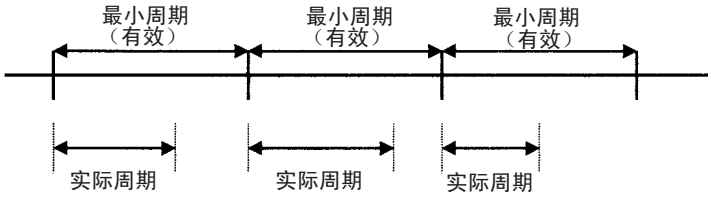
- 最小周期功能
- 最大周期功能 （警戒周期）
- 周期监控
- 快速响应输入
- 中断功能
- I/O 刷新方法
- 屏蔽特殊 I/O 单元的定期刷新
- 提高数据链接和其它（CPU）总线单元数据的刷新响应时间（仅限于 CS1-H， CJ1-H， CJ1M 或 CS1D CPU）。
- 通过后台执行处理的方式来减少周期内的变动 （仅限于 CS1-H， CJ1-H，CJ1M 或 CS1D CPU）。

6-1-1 最小周期

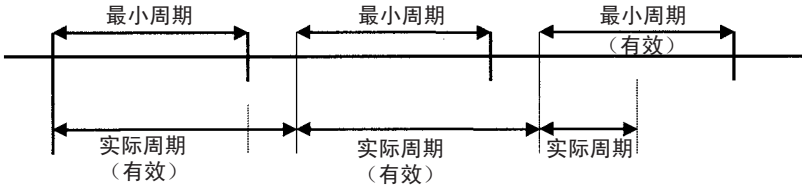
在 CS/CJ 系列的可编程控制器 （PLC）上可以设定一个最小 （或固定）周期（见注）。通过重复具有固定周期的程序可以消除 I/O 响应时间的变化。

注 采用并行处理模式亦可对 CS1-H，CJ1-H，CJ1M 或 CS1D 型中央处理器 （CPU）设定固定周期。

最小周期（1 至 32，000ms）在可编程控制器（PLC）设置中以 1 毫秒单位计。



如果实际周期比最小周期长，则最小周期功能无效且周期会各不相同。



可编程控制器 （PLC）的设置

地址	名称	设定值	缺省值
208 位：0 至 15	最小周期	001 至 7D00：1 至 32， 000ms （单位：1 毫秒）	0000 （无最小值）

6-1-2 最大周期 （警戒周期）

若是周期 （见注）超出最大周期设定值，其周期过长标记 （A40108）将变为 ON 状态，且可编程序控制器停止运行。

注 在此，当对 CS1-H，CJ1-H 或 CJ1M CPU 单元采用并行处理模式时，此周期为程序执行时间。

PLC 设定

地址	名称	设定值	缺省值
209 位：15	警戒周期 设定	0：初始值 (1s) 1：位 0 至 14	0001 (1 s)
209 位：0 至 14	警戒周期设定 （当 15 设为 1 时，即完成启用）	001 至 FA0:10 至 40,000ms （以 10ms 为单位）	

辅助区域标志及说明

名称	地址	说明
周期过长标志	A40108	如果周期超过警戒周期设定值， A40108 将呈“ON”状态，且中央处理器 （CPU）停止运行当对 CS1-H， CJ1-H 或 CJ1M CPU 单元采用并行处理模式时，“周期”即为程序执行时间。

注 在对 CS1-H， CJ1-H 或 CJ1M CPU 单元采用并行处理模式时，若外设服务周期超过 2 秒则将出错且中央处理器 （CPU）停止运行。此时， A40515 （外设服务周期超时标志）变为 ON 状态。

6-1-3 周期监控

在每一周期，最大周期和当前周期存储辅助区域内。对 CS1-H，CJ1-H 或 CJ1M CPU 单元采用并行处理模式，其程序执行时间将被存储起来。

辅助区域标志及说明

名称	地址	说明
最大周期 （对执行并行处理模式的 CS1-H，CJ1-H 或 CJ1M CPU 单元而言即：程序执行时间）	A262 和 A263	每个循环以 32 位二进制方式在下述范围内存储： 0 至 429,496,729.5 ms，以 0.1 毫秒计 （0 至 FFFF FFFF）
当前周期 （对执行并行处理模式的 CS1-H，CJ1-H 或 CJ1M CPU 单元而言即：程序执行时间）	A264 和 A265	每个循环以 32 位二进制方式在下述范围内存储： 0 至 429,496,729.5 ms，以 0.1 毫秒 （0 至 FFFF FFFF）

可使用一台编程工具 （CX-Programmer 或手持编程器）读取最近 8 次循环的平均周期。

减少循环时间

可对 CS/CJ 系列的可编程控制器采用下述有效方式来减少其循环时间：

1,2,3... 1. 将当前不执行的任务置于 “等待”状态。

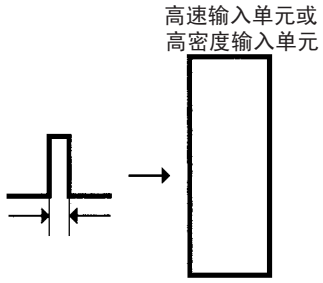
2. 跳过含 JMP （004）和 JME （005）指令且暂时不执行的程序部分。

对执行并行处理模式 CS1-H， CJ1-H CPU 单元，每一个服务周期，其外设服务循环时间将存储在 A268 （外设服务周期）内。

6-1-4 高速输入

当你希望接收到的脉冲比循环时间短时，使用 CS1W-IDP01 高速输入单元或使用 C200H-ID501/ID215 和 C200H-MD501/MD115/MD215 高密度 I/O 单元的高速输入。

高速输入时用 C200H 高密度输入单元得到脉宽 （ON 状态）为 1ms 或 4ms 的脉冲，用 CS1W-IDP01 高速输入单元时可得到脉宽为 0.1ms 的脉冲。



CS1W-IDP01: 0.1 ms
CJ1W-IDP01: 0.05 ms
C200H-ID501/ID215/MD501/MD115/MD215: 4 ms

6-1-5 中断功能

在下述状态下可执行中断任务。更详细的内容可参考*第 4-3 章中断任务*章节。

注 CS1D CPU 单元不支持中断功能。在 CS1D CPU 单元中，中断任务仅被用作附加的循环任务，即，没有任何其它中断任务的方式可使用。

I/O 中断 （中断任务 100 ～ 131）

当从中断输入单元接收到相应的输入信息（信号的上升沿，或对于 CS/CJ 系列中断输入单元可在信号的上升或下降沿）时，可执行一种 I/O 中断任务。

定时中断 （中断任务 2 ～ 3）

在一定的时间间隔执行中断任务。

断电中断 （中断任务 1）

当断电时执行断电中断任务。

外部中 （中断任务 0 ～ 255）

当终端是从特殊 I/O 单元， CPU 总线单元或内装板接收到指令时，执行外部中断任务。

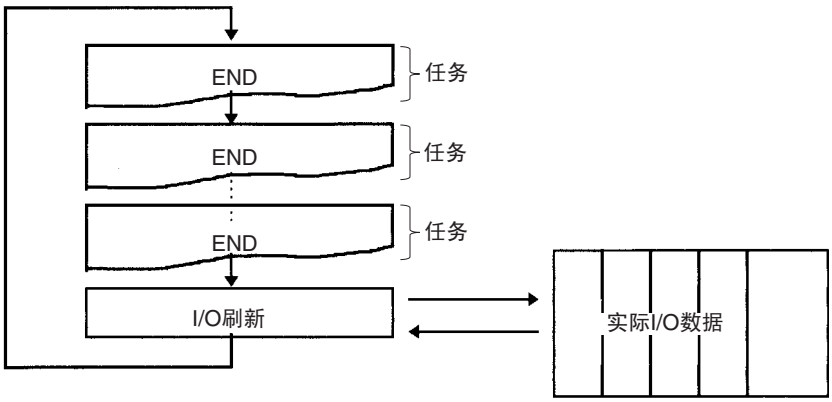
注 在 CJ1M CPU 单元中的内置中断输入和高速计数器输入可被用来激活中断任务。详细内容请参考 *CJ 系列的内置输入 I/O 操作手册*。

6-1-6 I/O 刷新方法

CS/CJ系列CPU单元中有三种方法对基本的I/O单元和特殊的I/O单元刷新数据: 周期刷新，立即刷新，和执行 IORF(097) 刷新。

1. 周期刷新

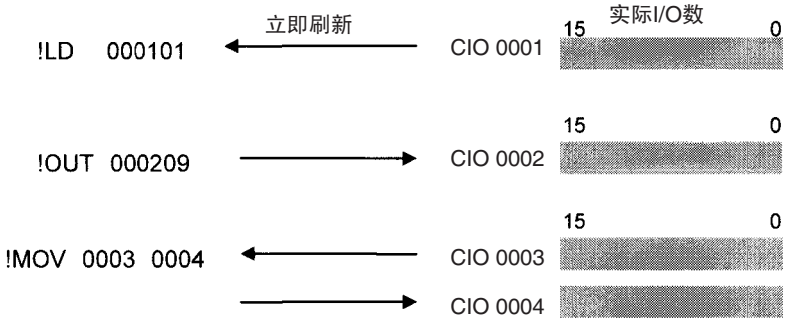
当可执行任务中所有的指令都执行完毕后， I/O 会被刷新。（在 PLC 设置中可对个别特殊的 I/O 单元设定不进行周期刷新）



2. 立即刷新

当 I/O 区域的一个地址在一条指令的立即刷新变量中被指定作为一个操作数时，那么当该条指令被执行时，该操作数的数据将被刷新，立即刷新指令可刷新位于基本 I/O 单元中的数据。

在 CJ1M CPU 单元中的内置 I/O 也可以执行立即刷新。

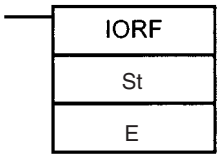


注
1. 当指令包含一位操作数，那么该位操作数包括的整个字将被刷新。当指令包含一个（计算机）字操作数，该（计算机）字将被刷新。
2. 在指令执行之前将刷新输入数据和源数据，指令执行完毕后，将刷新输出数据和目的数据。
3. 立即刷新变量的执行时间要长于指令中的常规变量，因此周期将延长。详细内容请参阅操作手册中第 *10-5 章指令执行时间和步数*。

3. IORF(097) 和 DLNK(226) 的执行

■ IORF(097)：I/O 刷新

根据执行指令可运用 IORF(097) 来刷新一系列的 I/O 字，IORF(097) 可刷新位于基本 I/O 单元和特殊 I/O 单元的数据。



下述实例显示了运用 IORF(097) 刷新 I/O 数据的 7 个字。

计算时输入和输出需要高速响应时，在计算指令的前后运用 IORF(097)。

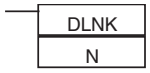
注 IORF(097) 需要相对长的指令执行时间，并且执行时间的增长量与需刷新的字的数目成正比，因此它会显著的延长周期。详细内容请参阅 10-5 指令执行时间和步骤次序。

■ DLNK(226) ：CPU 总线单元 I/O 刷新 （仅限于 CS1-H，CJ1-H，CJ1M，或 CS1D CPU 单元）

DLNK(226) 可用于特殊单元号的一种 CPU 总线单元的数据刷新。可刷新以下数据：

- 位于 CIO 区域单元中的字
- 位于 DM 区域单元中的字
- 对单元特殊的数据 （见注）

注 对 CPU 总线单元特殊的数据包括：控制链接单元或 SYSMAC LINK 单元的数据链接，以及设备网络单元的远程 I/O 数据链接。



例：

在左方的指令将刷新位于 CIO 区域 DM 区域单元中的字，以及对单元号为 1 的 CPU 总线单元特殊的数据。

应用事例：在一个长周期下，控制器链接数据链接的刷新间隔将非常长。对于控制器链接单元通过执行 DLNK （226）可以缩短间隔，从而增加数据链接刷新的频率。

6-1-7 禁止特殊 I/O 单元周期刷新功能

根据设置于单元前方的单元号，在特殊 I/O 单元区域 （CIO 2000-CIO 2959）的 10 个字，分配到每个特殊的 I/O 单元中。在 I/O 刷新期间的每个循环，数据在这个区域和 CPU 单元之间刷新，但对个别单元可以在 PLC 设置中禁止这种周期刷新功能。

禁止周期刷新功能有三种基本原因：

1,2,3...

1. 因为安装太多的特殊 I/O 单元导致周期太长时，可禁止特殊 I/O 单元的周期刷新功能。
2. 如果 I/O 刷新时间太短时，单元的内部处理可能跟不上，此时特殊 I/O 单元出错标识（A40206）将变为 ON，且特殊 I/O 单元将不能正常运行。在这种情况下，通过在 PLC 设置中延长最小周期设定，或禁止特殊 I/O 单元的周期 I/O 刷新。
3. 当特殊 I/O 单元在中断任务中执行 IORF （097）指令时可刷新数据，就可禁止周期刷新。如果对同一个单元同时进行周期刷新和用 IORF （097）刷新，将产生中断任务出错，且中断任务出错标识将显示 “ON”。

当周期刷新被禁止时，特殊 I/O 单元的数据可以通过带有 IORF （097）指令的程序运行而刷新。

PLC 设置

对特殊 I/O 单元 0 ～ 95 周期刷新禁止位，直接与地址 226 ～ 231 中的这 96 个位相对应。

地址	名称	设置	默认
226 位 0	对特殊 I/O 单元 0 禁止周期刷新位	0：起用 1：禁止	0 （起用）
:	:	:	:
231 位 15	对特殊 I/O 单元 95 禁止周期刷新位	0：起用 1：禁止	0 （起用）

6-1-8 改进 CPU 总线单元数据刷新响应

只有 CS1-H，CJ1-H，CJ1M，或 CS1D CPU 单元支持这个功能。通常，在程序执行后的 I/O 刷新期间，CPU 总线单元的数据链路和其它特殊数据是随着位于单元中的 CIO 和 DM 区域字的刷新而刷新。

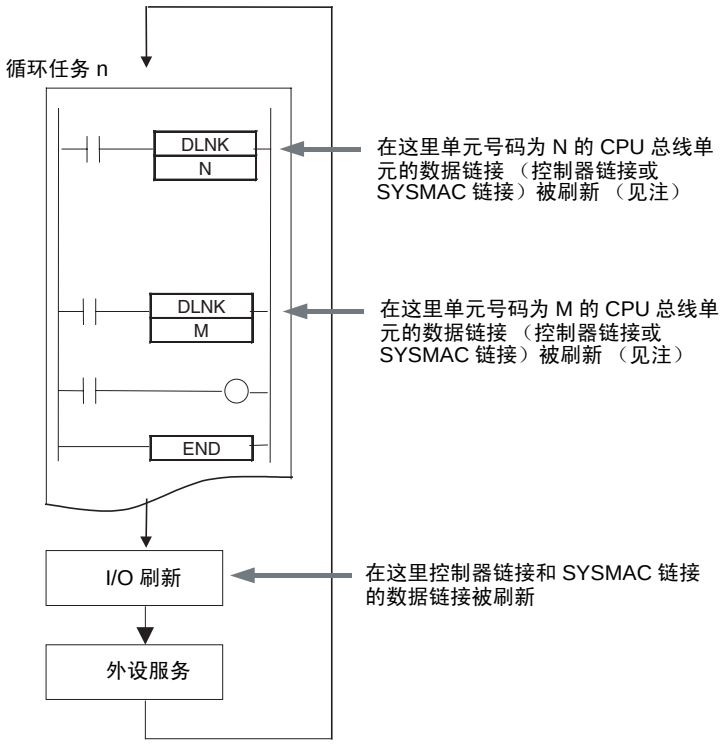
下表中列出了一些 CPU 总线单元特殊数据的实例。

单元	特殊数据
控制器链接单元和 SYSMAC LINK 单元	控制器链接和 SYSMAC LINK 数据链接 （包括自动链接和用户设定链接）
CS/CJ 系列设备网络单元	设备网络远程 I/O 通信 （包括固定配置和用户设定配置）

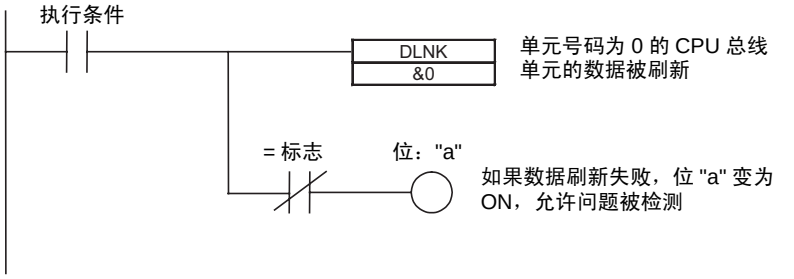
下列功能可以用来改进在 CS1-H，CJ1-H，CJ1M，或 CS1D CPU 单元中特殊 CPU 总线单元数据的刷新响应。

- 运用并行处理方式或高速指令缩短周期。（CJ1M CPU 单元不支持并行处理方式）
- 通过规定特殊 CPU 总线单元的单元号码，执行 DLNK（226）刷新（DLNK（226）在程序中可多次使用）。

注 1. 当数据链接被刷新时，较长的周期（如：100 ms）会增加间隔时间。这种情况下可使用 DLNK（260），可参阅下列实例。



注 如果对 CPU 总线单元执行 DLNK（226）出现占线刷新数据，数据将不能刷新并且相等标志将变为 OFF。通常，相等标识必须按如下编程以确保刷新正常完成。

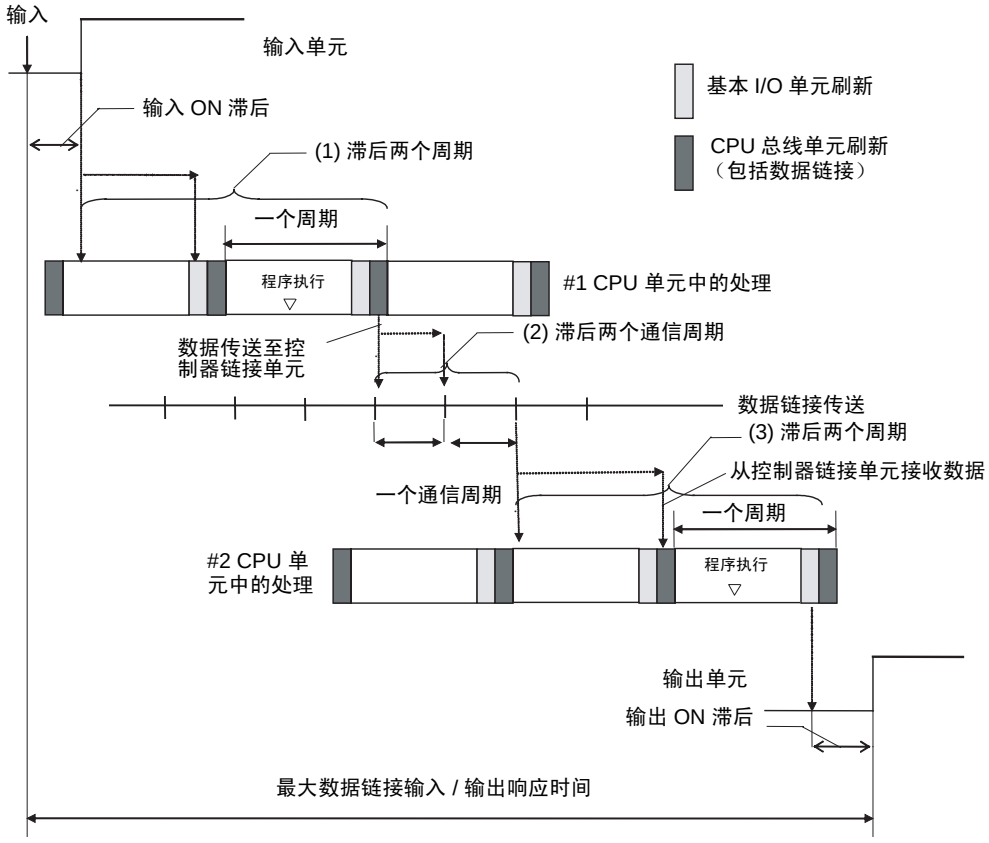


2. 对基本的 I/O 单元和特殊 I/O 单元用 IORF（097）刷新数据。DLNK（226）用于刷新 CPU 总线单元（位于此单元的 CIO 和 DM 区域字和特殊数据）。

6-1-9 最大数据链接 I/O 响应时间

正常处理

下图说明了当不使用 DLNK（226）指令时，数据链接 I/O 响应时间最长的情况下的数据流动状况。



上图中显示有三点出现了处理滞后，增加了数据链接 I/O 响应时间。

1,2,3...

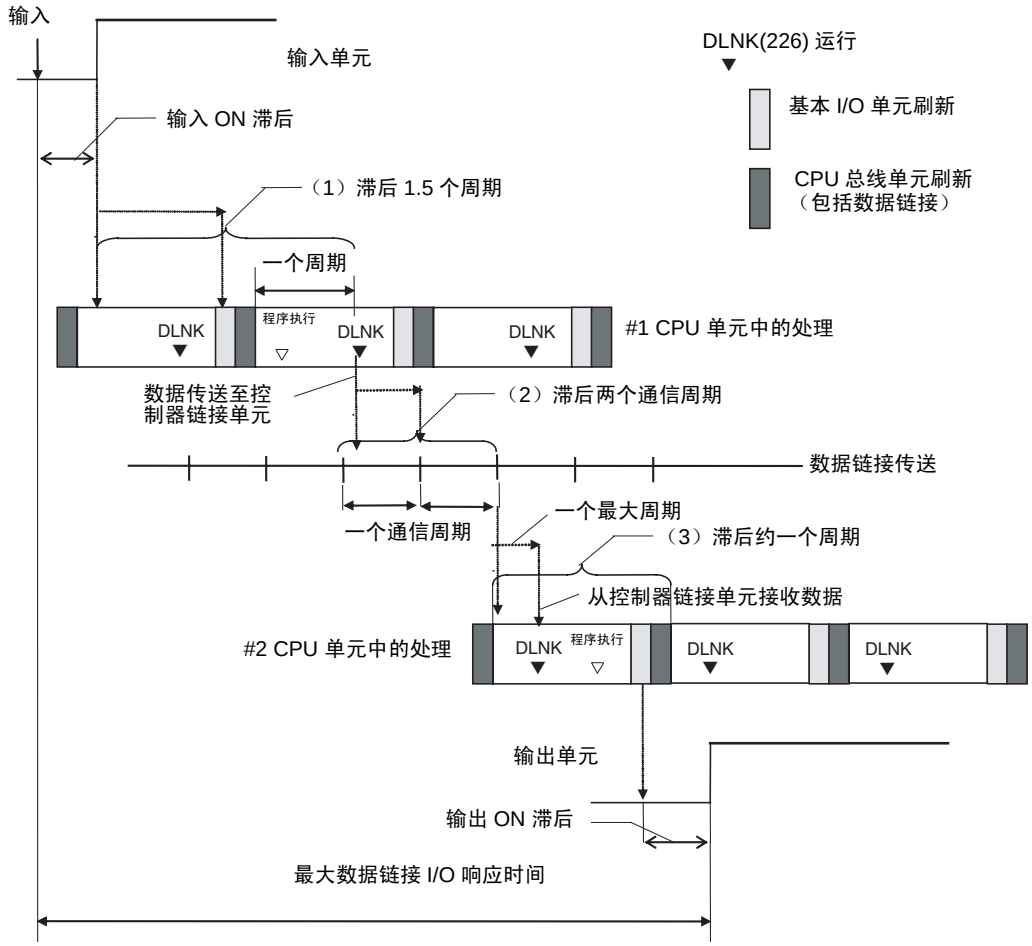
1. 恰好在 I/O 刷新以后，输入传到 CPU （#1 CPU 单元），导致输入读入到 PLC 的时间滞后一个周期。程序执行后刷新 CPU 总线单元，导致总计滞后两个周期。
2. 恰好在 PLC 通过交换节点令牌以后，数据交换发生，在数据传递到数据链接处理之前导致最多滞后一个通信周期。接收到交换节点标志后，同样最多滞后一个通信周期，导致总计最多滞后两个通信周期。
3. 数据交换后，在数据链接处理中传递的数据传到 PLC（#2 CPU 单元），因此这个数据直到下一次数据交换时才会被读入到 PLC，导致最多滞后一个周期。程序执行后刷新 CPU 总线单元，导致总计滞后两个周期。

最大数据链接 I/O 响应时间的方程式如下：

输入 ON 滞后	1.5 ms
在 #1 CPU 单元 PLC 的周期 x 2	25 ms × 2
通信周期 x2	10 ms × 2
在 #2 CPU 单元 PLC 的周期 x 2	20 ms × 2
输出 ON 滞后	15 ms
总计 （数据链接 I/O 响应时间）	126.5 ms

使用 DLNK(226)

下图说明了使用 DLNK（226）时，数据流产生的最大数据链接 I/O 响应时间。



上图中显示有三点出现了处理滞后，增加了数据链接 I/O 响应时间。

注 本例中，假设在两个CPU单元的程序中，DNLK（226）是放置在其它指令之后。

1,2,3...

1. 恰好在 I/O 刷新以后，输入传到 CPU （#1 CPU 单元），导致输入读入到 PLC 的时间滞后一个周期。CPU 总线单元在程序执行期间刷新，使总计滞后缩短为约 1.5 个周期。
2. 恰好在 PLC 通过交换节点令牌以后，数据交换发生，在数据传递到数据链接处理之前导致最多滞后一个通信周期。接收到交换节点标志后，同样最多滞后一个通信周期，导致总计最多滞后两个通信周期。
3. I/O 刷新后，在数据链接处理中传递的数据传到 PLC （#2 CPU 单元），但用 DLNK （226）刷新数据，因此这个数据被读入到 PLC，导致滞后不超过一个周期。程序执行后刷新基本的 I/O 单元，导致总计滞后约一个周期。

最大数据链接 I/O 响应时间的方程式如下：

输入 ON 滞后	1.5 ms	---
在 #1 CPU 单元 PLC 的周期 x1.5	25 ms × 1.5	快速时 12.5ms (25 ms x 0.5)
通信周期 x 2	10 ms × 2	---
在 #2 CPU 单元 PLC 的周期 x1	20 ms × 1	快速时 20 ms (20 ms x 1)
输出 ON 滞后	15 ms	---
总计 （数据链接 I/O 响应时间）	94 ms	快速时 32.5 ms （快了 26%）

6-1-10 后台执行功能

使用后台执行功能可以减少周期的波动。只有 CS1-H，CJ1-H，或 CJ1M CPU 单元具有这个后台执行功能。

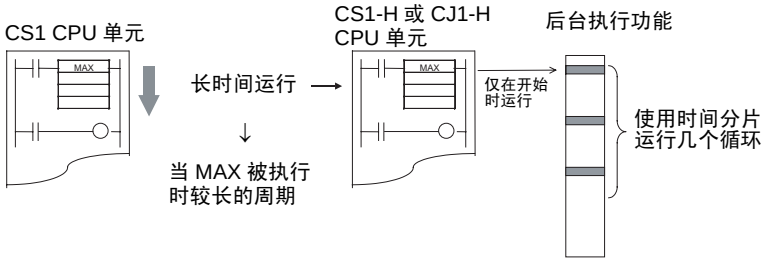
表格数据处理 （如数据搜索）和文本字符串处理 （如文本字符串搜索）运行需要时间，由于需要大量的时间运行处理将使周期产生大幅度的波动。

然而，在 CS1-H， CJ1-H，或 CJ1M CPU 单元 （见注）中可使用后台执行功能执行下述指令分几个循环，进行可控制周期的波动。PLC 设置可在指令的每种类型中设定后台执行功能。

注 CS1D CPU 单元不支持后台执行功能。

- 表格数据处理指令
- 文本字符串处理指令
- 数据移位指令 （ 仅限于移位寄存器）

对上述指令设定后台执行功能可以暂时帮助控制周期波动。



应用

当减少对周期的影响比数据处理的速度更重要时，后台执行功能可被用于大量数据处理 （如数据编译或数据处理），这种需求仅在特殊的时候 （如：一天一次）要求这种处理。

过程

1,2,3...

1. 设定 PLC 设置，对需求指令启用后台执行功能。
2. 在 PLC 设置中为了使用后台执行功能设定通信端口号 （逻辑端口号）。这个端口号在后台执行中用于所有的指令。

注 一个端口被用于所有的后台执行功能。一条指令的后台执行功能可能不启用，如果后台执行功能已经在另一个指令下作用。使用通信端口起用标志控制指定后台执行功能的指令，因此在同一时间只运行一个指令。

3. 如果用于指定后台执行功能的一个指令被执行，运行只在运行条件满足的那个循环中启动，并且运行将不能在这同一的循环中完成。
4. 当后台执行功能起动，那个端口的通信端口起用标志将变为 OFF。
5. 后台执行功能将在几个循环中运行。
6. 处理结束后，那个端口的通信端口起用标志将变为 ON。这时就可以在后台执行中执行另一个指令。

应用指令

■ 表格数据处理指令

指令	助记码	功能代码
数据搜索	SRCH	181
交换字节	SWAP	637
寻找最大值	MAX	182
寻找最小值	MIN	183
求和	SUM	184
帧校验和	FCS	180

■ 正文串处理指令

指令	助记码	功能代码
串传送	MOV$	664
链接串	+$	656
取左串	LEFT$	652
取右串	RIGHT$	653
取中间串	MID$	654
寻找串	FIND$	660
串长	LEN$	650
取代串	RPLC$	661
删除串	DEL$	658
交换串	XCHG$	665
清除串	CLR$	666
插入串	INS$	657

■ 数据移位指令

指令	助记码	功能代码
异步移位寄存器	ASFT	017

正常执行和后台执行指令的区别

正常指令执行和后台指令执行的区别列于下面：

■ 输出到变址寄存器 （IR）

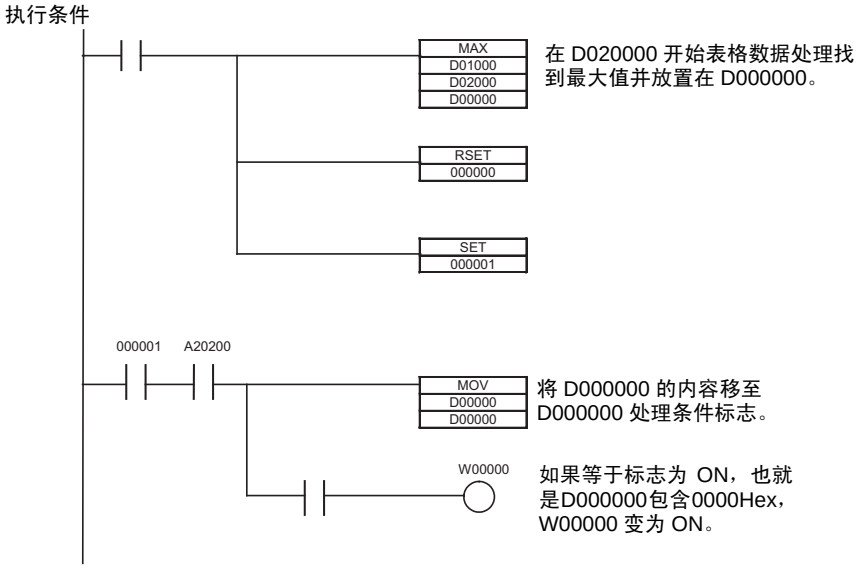
如果执行 MAX （182）或 MIN （183），把包含有最大或最小值字的 I/O 存储器变换地址输出到一个变址寄存器，地址将不会输出到变址寄存器而是输出到 A595 和 A596。为了将地址保存在一个变址寄存器中，须用一个数据移动指令（如：MOVL（498））将在 A595 和 A596 中的地址复制到一个变址寄存器上。

■ 条件标志

在后台处理的以下指令的执行不会更新条件标志。为了存取条件标志状态，执行一个以同样方式影响条件标志的指令，如下例所示，随后存取条件标志。

例：

MOV （021）与 MAX （182）以相同方式影响等于标志与负标志，也就是说，当内容为 0 等于标志变为 ON，如果 MSB 是 “ON”时负标志变为 ON。这样 MOV （021）可被用来复制 MAX （182）的结果，在相同地址处理条件标志，因此状态可以被存取。



■ 输出到变址寄存器 IR00

如果执行 SRCH （181）把包含有匹配值 （如果有几个，选第一个字）字的 I/O 存储器变换地址输出到一个变址寄存器，地址将不会输出到变址寄存器而是输出到 A595 和 A596。

■ 对 SRCH （181）输出到数据寄存器 （DR）

如果执行 SRCH （181）把匹配数据输出到一个数据寄存器，这个数据不会输出到数据寄存器而是输出到 A597。

■ 匹配文本字符串

如果 SRCH（181）找到匹配数据，它不会将等于标志变为 ON，而是把 A59801 变为 ON。

■ 指令出错

如果在后台处理的一个指令发生一个指令执行出错或不合法存取出错， ER 或 AER 标志不会变为 ON，而是 A39510 变为 ON。A39510 将一直为 ON，直到在后台处理下一条指令。

■ 对 MAX （182）或 MIN （183）输出到数据寄存器 （DR）

如果 MAX （182）或 MIN （183）和指定为最小或最大值的输出字的一个数据寄存器一起执行，将发生一个指令执行出错，且 ER 标志将变为 ON。

PLC 设置

字	数	名称	设定	默认和更新时间
198	15	表格数据指令后台执行	0：不在后台执行 1：在后台执行	0：不在后台执行操作开始
	14	文本字符串指令后台执行	0：不在后台执行 1：在后台执行	
	13	数据移位指令后台执行	0：不在后台执行 1：在后台执行	
	00～03	后台执行的通信端口号	0 ～ 7 Hex：0 ～ 7 通信端口（内部逻辑端口）	0 Hex：端口 0 操作开始

辅助区域标志与字

名称	地址	说明
通信端口允许标志	A20200 ～ A20207	应用相应端口号执行网络工作指令 （SEND， RECV，CMND，或 PMCR），或在相应端口号 （仅限于 CS1-H， CJ1-H，或 CJ1M CPU 单元）运行后台执行时，位 00 ～ 07 对应通信端口号 0 ～ 7 变为 ON。 在 CS1-H，CJ1-H，或 CJ1M CPU 单元对存储卡运用简单的备份操作进行写或比较操作时，将自动分配通信端口，在操作期间相应标志变为 ON，当操作完成后，变为 OFF。
通信端口出错标志	A21900 ～ A21907	网络工作指令 （SEND， RECV， CMND，或 PMCR）执行期间发生出错时变为 ON。位 00 ～ 07 对应通信端口号 0 ～ 7。 在 CS1-H，CJ1-H，或 CJ1M CPU 单元对存储卡运用简单的备份操作进行写或比较操作时，将自动分配通信端口。如果一个错误发生相应标志变为 ON，当简单备份操作正常结束后，变为 OFF。
通信端口完成码	A203 ～ A210	网络工作指令 （SEND， RECV， CMND，或 PMCR）执 行完成后，这些字包含了相应端口号的完成码。当后台执行完成后，内容将被清除。（仅限于 CS1-H， CJ1-H，或 CJ1M CPU 单元）。字 A203 ～ A210 对应通信端口 0-7。 在 CS1-H，CJ1-H，或 CJ1M CPU 单元对存储卡运用简单的备份操作进行写或比较操作时，将自动分配通信端口，一个完成码将存储在相应的字中。

名称	地址	说明
后台执行 ER/AER 标志	A39510	在后台执行的一条指令执行出错或不合法存取错误发生时，变为 ON。当电源接通或操作起动时，变为 OFF。
后台执行 IR00 输出	A595 和 A596	当在后台一条指令执行的输出是指定一个变址寄存器时，这些字接收输出。 IR00 没有输出。 范围：0000 0000 ～ FFFF FFFF Hex。 低 4 位数字：A595，高 4 位数字：A596
后台执行 DR00 输出	A597	当在后台一条指令执行的输出是指定一个数据寄存器时这些字接收输出。 DR00 没有输出。 范围：0000 ～ FFFF Hex。
后台执行等于标志输出	A59801	为在后台执行的 SRCH （181）找到匹配数据后，这个标志变为 ON。

注 CPU 单元中通信端口 （内部逻辑端口）被用于后台执行和以下指令。

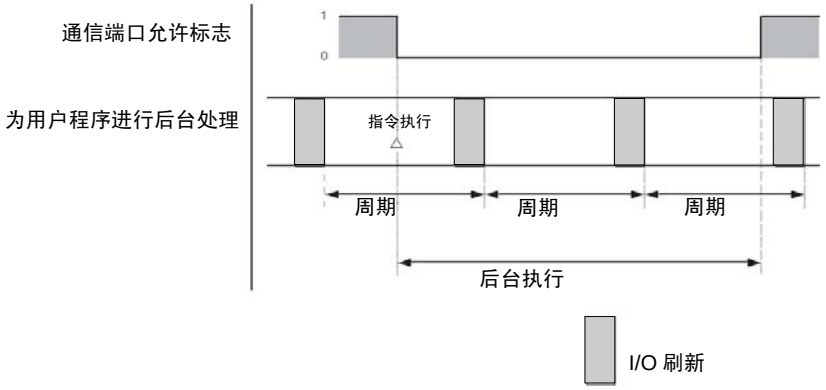
- SEND （090）， RECV （098），和 CMND （490）（网络工作通信指令）
- PMCR(260) （协议宏）

后台指令和以上指令不能在同一端口同时执行。使用通信端口允许标志来确保在任一时间在每个端口仅仅执行一种指令。

注 一端口的通信端口允许标志变为 ON 时，如果对这一端口有一条指令是指定在后台执行， ER 标志将变为 ON，并且将不执行后台指令。

通信端口允许标志

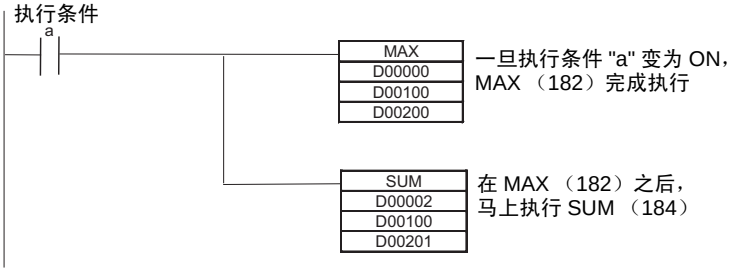
当处理未在端口中进行，通信端口允许标志为 ON，当处理正在端口中进行，通信端口允许标志为 OFF。



程序示例 1：

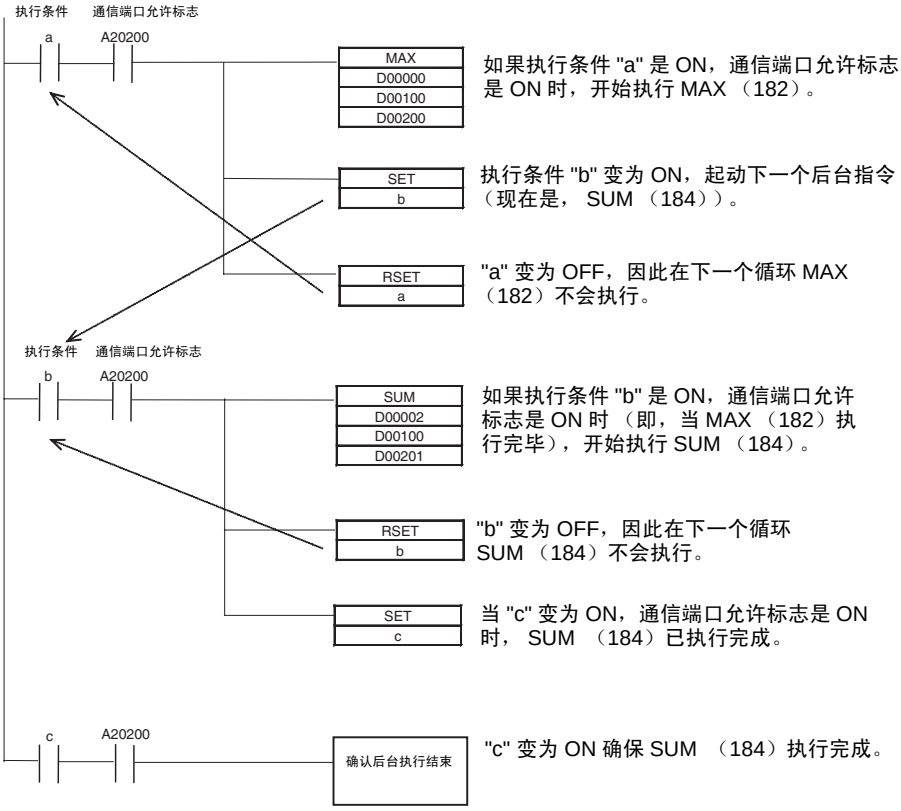
■ 无后台执行的常规程序

如下所示，当指令执行后，完成处理。



■ 带后台执行程序

具有后台执行功能后，改变了程序，因此仅当特定的通信端口允许标志变为 ON 时（即：仅当在后台执行时，或网络工作通信时还没使用该端口），MAX(182) 才能执行。同样，输入状态受到 SET 和 RESET 指令控制，确保以正确的次序执行处理。（在下列实例中通信端口 0 用于后台执行）。

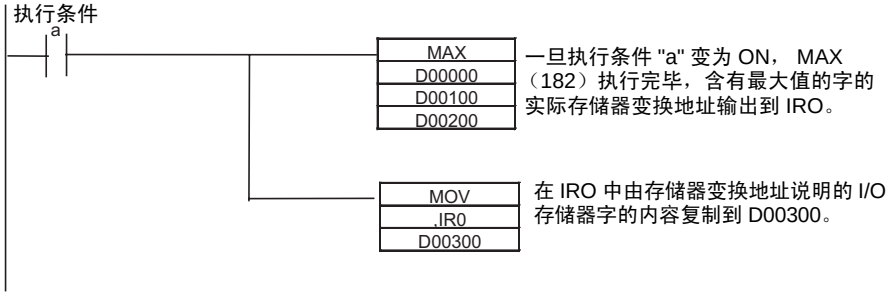


程序示例 2：

本例说明了当变址寄存器的输出被指定（如：MAX（182），MIN（183），和 SRCH（181））时的后台执行。

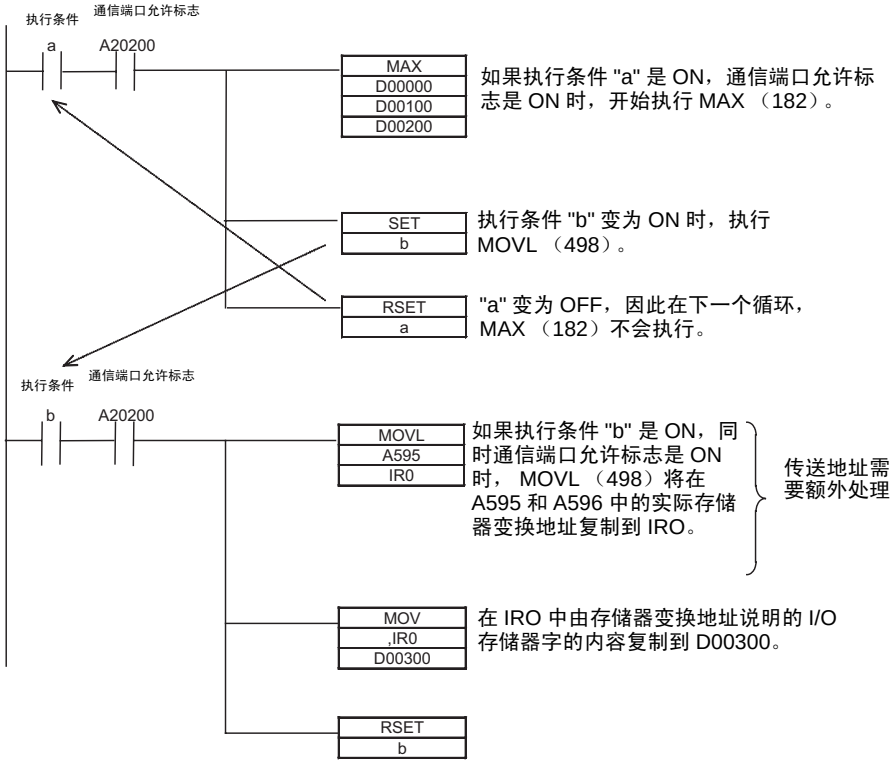
■ 无后台执行的常规程序

如下所示，含有最大值字的实际存储器变换地址输出到一个变址寄存器。



■ 带后台执行程序

具有后台执行功能，含有最大值的字的实际存储器变换地址输出到 A595 和 A596。因此使用 MOVL（498）将实际存储器变换地址传送到变址寄存器。



6-1-11 任务之间的共享变址寄存器和数据寄存器

仅在 CS1-H，CJ1-H，CJ1M，或 CS1D CPU 单元中支持任务之间共享变址寄存器和数据寄存器（IR/DR）。常规设置是单独的寄存器分配一个任务。当前的设置可以在 A09914 中确认。

注 1. 在两个或多个任务之间如果需要相同内容，可使用共享变址寄存器和数据寄存器消除在任务之间存和取寄存器内容的需求。在 *CS 系列操作手册*（W339）或 *CJ 系列操作手册*（W393）关于变址寄存器篇章，查阅有关存取寄存器内容的信息。

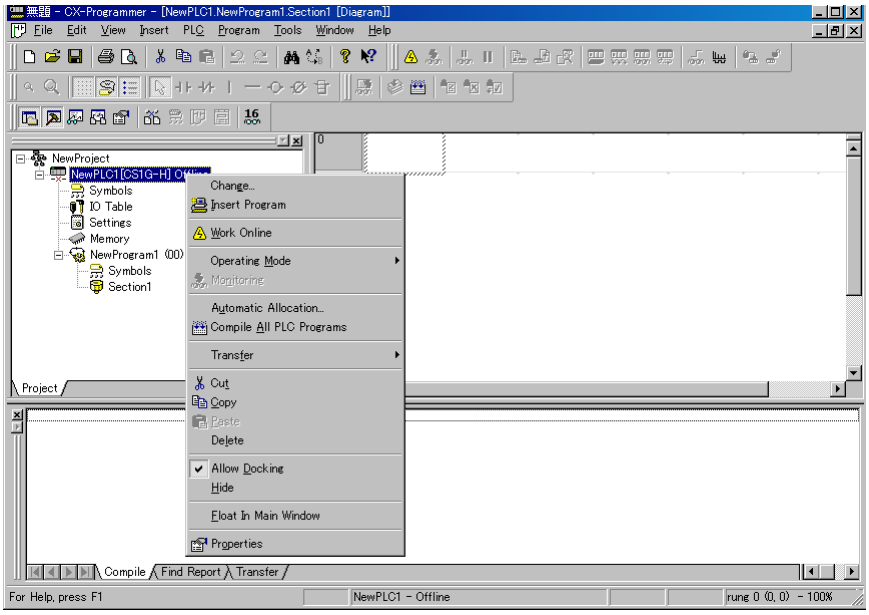
2. 当变址和数据寄存器被分享时，任务之间的切换时间将有所加快。如果寄存器没被使用或如果每一项任务对单独的寄存器没有特别的要求，这时推荐设置共享寄存器。

设置方法

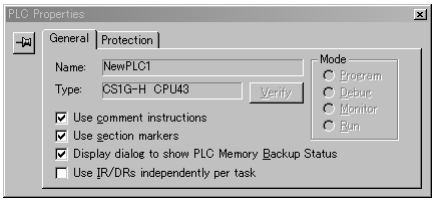
使用 CX-Programmer 设置共享变址和数据寄存器。不能用手持编程器设置。

1,2,3...

1. 在 CX-Programmer 目标栏中选择 PLC （PLC），点击鼠标右键。



2. 选择特性。将显示下列对话框。



3. 如果对每一项任务需要单独的变址和数据寄存器，每一项任务单独地使用 IR/DR，作一记号“ √ ”。使用共享变址和数据寄存器的所有任务，将“ √ ”去掉。

辅助区域标志与字

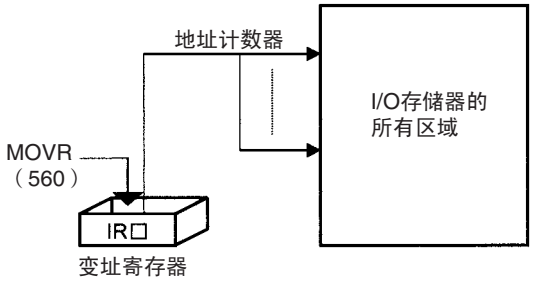
名称	地址	说明
任务之间的 IR/DR 操作	A09914	说明任务之间变址和数据寄存器是否共享 0：每一项任务单的独寄存器 （默认） 1：所有任务共享寄存器

6-2 变址寄存器

6-2-1 什么是变址寄存器?

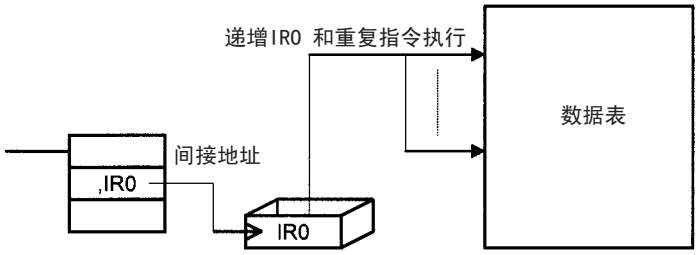
变址寄存器功能如同地址计数器（指针）用来规定存储器地址，这个地址是 I/O 存储器中的绝对存储器地址。使用 MOVR （560）或 MOVRW （561）在变址寄存器中存储 PLC 存储器地址后，在其它指令中输入这一变址寄存器，作为一个操作数使存储的 PLC 存储器地址成为非直接地址。

变址寄存器的优势在于他们能够指定在 I/O 存储器中 （包括定时器和计数器 PV）的任何一个位或字。



6-2-2 变址寄存器应用

当变址寄存器与循环 （如 FOR- NEXT 循环）指令共同使用时，是一种有用的工具。变址寄存器的内容可以非常容易地递增，递减和偏置，因此一个循环中少量的指令可以非常有效地处理连续数据的表格。



基本操作

通常以以下步骤使用变址寄存器：

1,2,3...

1. 在一个变址寄存器中使用 MOVR（560）存储所需的数或字的 PLC 存储器地址。
2. 在任何指令中把变址寄存器规定为一个操作数，成为所需的数或字的非直接地址。
3. 偏置或递增原始 PLC 存储器地址（参见下方）使地址计数器指向另一个地址。
4. 继续步骤 2 和 3，执行地址的任一位数上的指令。

偏置，递增和递减地址

下表给出了间接地址的各种变化情况。

变化	语句
间接寻址	,IR@
固定偏置的间接寻址	常数， IR@ （包括一个正负常数）

变化	语句
带有 DR 偏置的间接寻址	DR@,IR@
自动递增的间接寻址	递增量 1: ,IR@+ 递增量 2: ,IR@++
自动递减的间接寻址	递减量 1: ,–IR@ 递减量 2: ,– –IR@

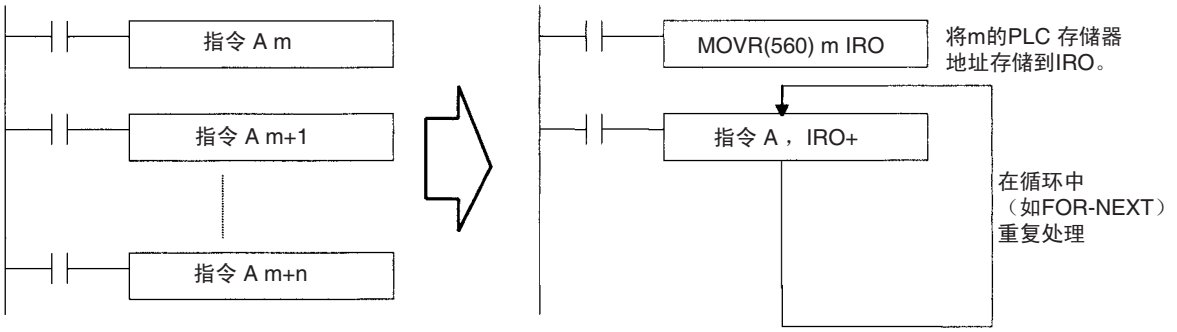
变址寄存器直接寻址的指令

使用下述指令用变址寄存器直接寻址。

DOUBLE SIGNED BINARY ADD WITHOUT CARRY: +L(401), DOUBLE SIGNED BINARY SUBTRACT WITHOUT CARRY: –L(411), DOUBLE INCREMENT BINARY: ++L(591), and DOUBLE DECREMENT BINARY: – –L(593)

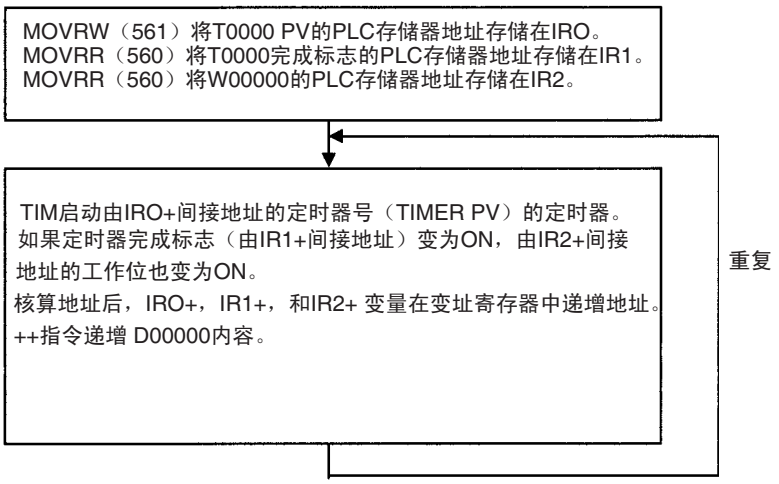
例 1

下例显示了在一个程序循环中用变址寄存器可替换一长串的指令。在本例中，指令 A 重复了 n+1 次来重复一些操作 （如读入和比较表中的值）。

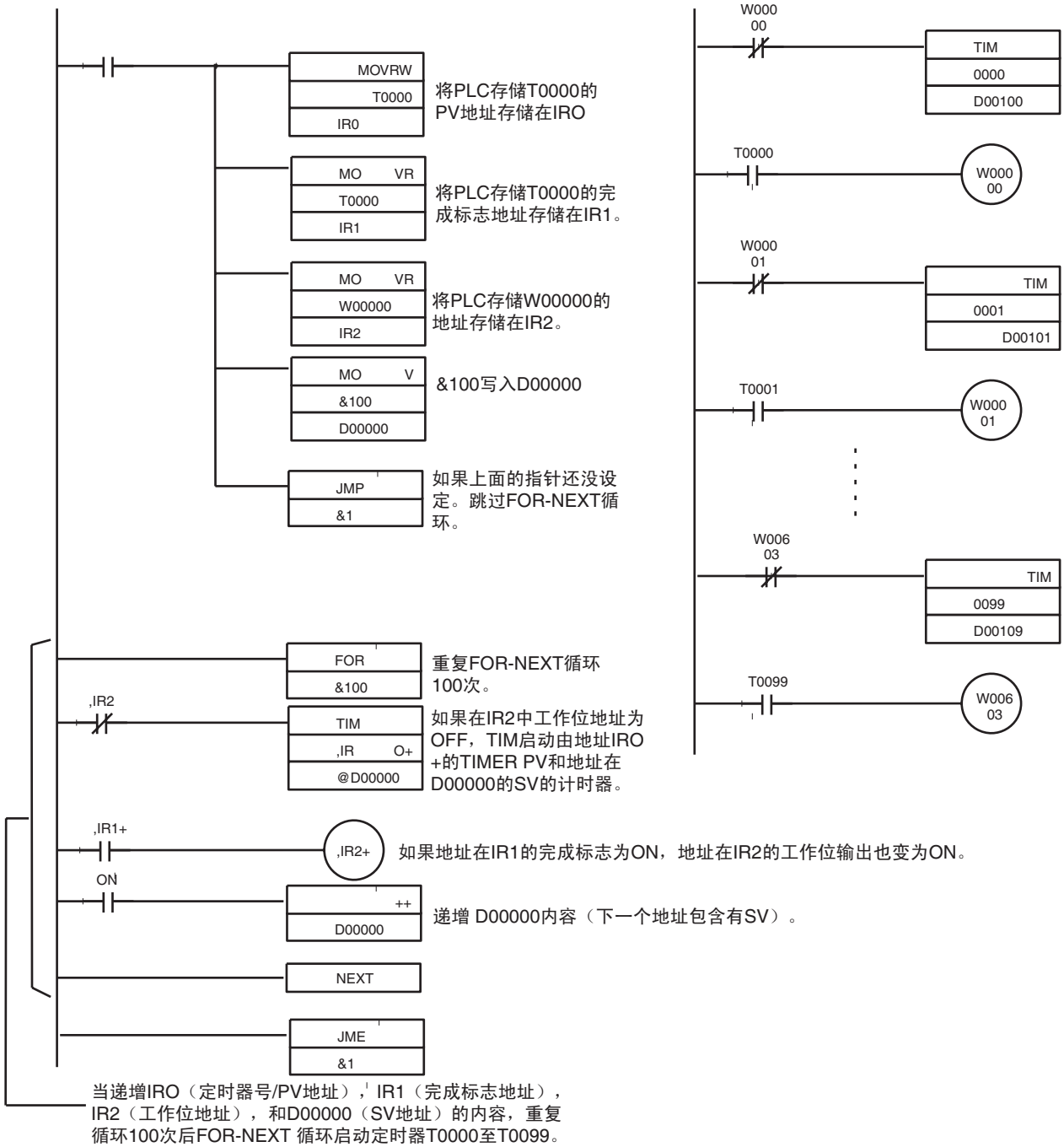


例 2

下例中在 FOR-NEXT 循环中，使用变址寄存器和储存在 D00100 ～ D00109 的 SV 定义和启动 100 个定时器 （T0000 ～ T099）。每一个定时器的定时器号和完成标志由变址寄存器规定，因此当变址寄存器在每一次重复时递增 1 时循环重复进行。



在左边的 11 条指令的子程序与右边的 200 条指令的子程序等效。



变址寄存器的直接寻址

只能用下表中的指令用变址寄存器直接寻址。

指令组	指令名称	助记符	基本功能
数据传送指令	传送到寄存器	MOVR(560)	在变址寄存器中存储数或字的 PLC 存储器地址
	定时器 / 计数器 PV 传送到寄存器	MOVRW(561)	
表格数据处理指令	设置记录位置	SETR(635)	
	获得记录号	GETR(636)	输出存储在变址寄存器中的 PLC 存储器地址
数据传送指令	双字传送	MOVL(498)	变址寄存器之间传送。被用于交换和比较。
	双字数据交换	XCGL(562)	
比较指令	与双字等于	=L(301)	
	与双字不等于	<>L(306)	
	与双字小于	<L(311)	
	与双字小于等于	<=L(316)	
	与双字大于	>L(321)	
	与双字大于等于	>=L(326)	
	双字比较	CMPL(060)	
递增 / 递减指令	双字二进制递增	++L(591)	通过递增，递减或偏置内容使变址寄存器的 PLC 存储器地址改变。
	双字二进制递减	– –L(593)	
符号数学指令	无进位双字带符号二进制加法	+L(401)	
	无进位双字带符号二进制减法	–L(411)	
特殊指令	由 CV 地址转化	FRMCV(284)	在 CV 系列和 CS/CJ 系列地址之间改变实际 PLC 存储器地址。（仅限于 CS1-H，CJ1-H，或 CJ1M CPU 单元）。
	转化到 CV 地址	TOCV(285)	

注 用于倍长操作数 （即，由 "L" 结尾）的指令被用于 IRO ～ IR15 变址寄存器，因为每个寄存器包含两个字。

6-2-3 与变址寄存器相关的处理

CS/CJ 系列 CPU 单元数据表格处理指令具有变址寄存器的求反功能。这些指令可大致分成栈处理和表格处理指令。

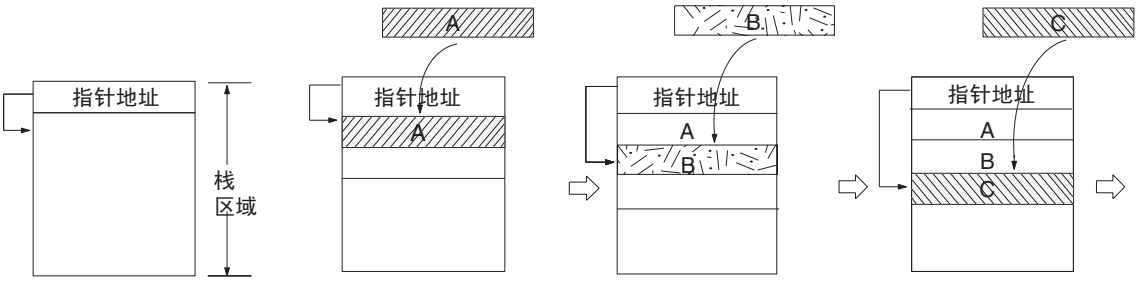
处理	目的	指令
堆栈处理	FIFO （先进先出）操作或 LIFO （后进先出）数据表格和在数据表格中读，写，插入，删除，或数据项目计数	SSET （630）， PUSH （632）， FIFO （633）， LIFO （634）和， SREAD （639）， SWRITE （640）， SINS （641）， SDEL （642）， SNUM （638）（仅限于 CS1-H， CJ1-H，或 CJ1M CPU 单元）

处理			目的	指令
表格处理	一个字记录表格 (范围指令)	基本处理	在范围中找出下列值：校验和，特定值，最大值，或最小值	FCS(180)， SRCH(181)， MAX(182)，MIN(183)，和 SUM(184)
		特殊处理	执行其它各种表格处理，如：比较或分类。	将下列指令与变址寄存器共用，如：SRCH （181）， MAX （182） MIN （183），和比较指令
	多个字记录表格 （记录表指令）		对几个字长的记录中的数据处理	将下列指令与变址寄存器共用，如：DIM （631）， SETR （635） GETR （636），和比较指令。

栈处理

栈指令作用于被称为栈的特殊定义的数据表格。以先进先出（FIFO）或后进先出 （LIFO）的条件，可以从栈中取得数据。

必须将 I/O 存储器的一个特定区域定义为栈。栈的第一个字表示的是栈的长度和栈的指针。每一次数据写入栈，在指明数据应该存储的下一个地址时，栈的指针递增。

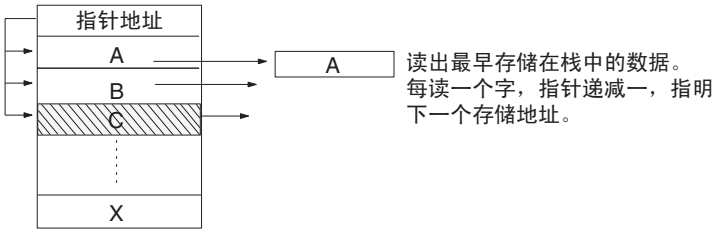


上图表示了数据加入以前指针数据的状态。

注 实际上，栈的最初两个字包含了栈中最后一个字的 PLC 存储器地址，下一个字包含栈指针。

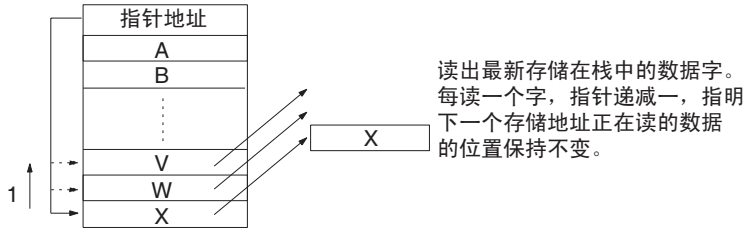
FIFO(先进先出) 处理

下图显示了先进先出 （FIFO）栈的操作。



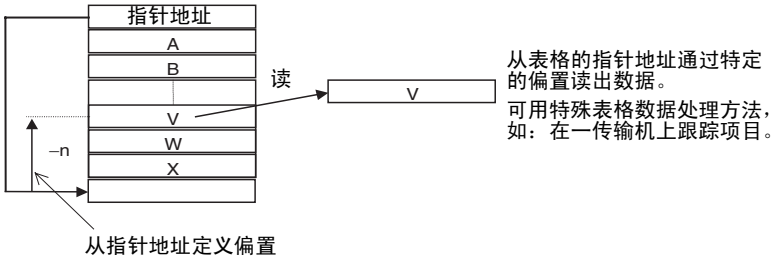
LIFO （后进先出 ）处理

下图显示了后进先出 （LIFO）栈的操作。



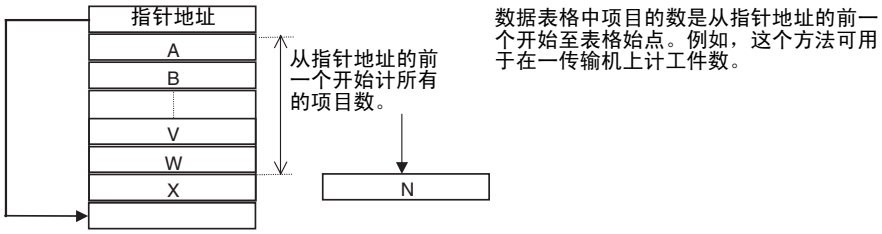
特殊表格数据处理

可以读，写，插入，或删除表格中的每一项。下图显示了读的实例。



表格数据计数

下图显示了数据表格中如何对数据计数。



栈指令

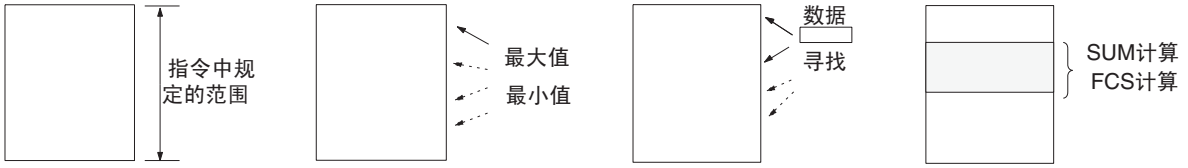
下表列出栈指令和它们的功能。栈一般应用在以下方面：自动仓库系统的货架信息处理，测试结果处理，和传输机上工件信息的管理。

指令	功能
SSET(630)	定义栈区域
PUSH(632)	存储数据在栈中的下一个可用的字中。
FIFO(633)	以先进先出条件从栈中读出数据。
LIFO(634)	以后进先出条件从栈中读出数据。
SREAD(639)	从表格中读出特定项目 （仅限于 CS1-H， CJ1-H，或 CJ1M CPU 单元）
SWRITE(640)	特定项目写入表格 （仅限于 CS1-H， CJ1-H，或 CJ1M CPU 单元）
SINS(641)	在表格中插入一特定项目 （仅限于 CS1-H， CJ1-H，或 CJ1M CPU 单元）
SDEL(642)	从表格中删除一特定项目 （仅限于 CS1-H， CJ1-H，或 CJ1M CPU 单元）
SNUM(638)	计表格中的项目数 （仅限于 CS1-H， CJ1-H，或 CJ1M CPU 单元）

表格处理 （范围指令）

范围指令作用于字范围中，这范围可认为是具有单字记录的表格。这些指令执行一些基本操作，如：在范围中找出最大值或最小值，在范围中寻找一特定值，或计算和或 FCS。

结果字（包含有最大值，最小值，寻找数据，等等）的 PLC 存储器地址自动存储在 IRO。在新近的一些指令中 （如：MOV （021）读入字的内容）变址寄存器 （IRO）可被用作一个操作数，或进行其它处理。



下表列出范围指令和它们的功能。

指令	功能	说明
SRCH(181)	寻找数据	在特定范围内发现寻找的数据，将含有那个值的字的 PLC 存储器地址输　出到 IRO。
MAX(182)	找最大值	在特定范围内发现最大值，将含有那个值的字的 PLC 存储器地址输出到 IRO。
MIN(183)	找最小值	在特定范围内发现最小值，将含有那个值的字的 PLC 存储器地址输出到 IRO。
SUM(184)	计算和	在特定范围中计算数据的和。
FCS(180)	计算校验和	在特定范围中计算数据的帧校验和。

在 FOR-NEXT 循环中变址寄存器可以与其它指令 （如比较指令）共用，在字的范围内执行更复杂的操作。

表格处理 （记录表格处理）

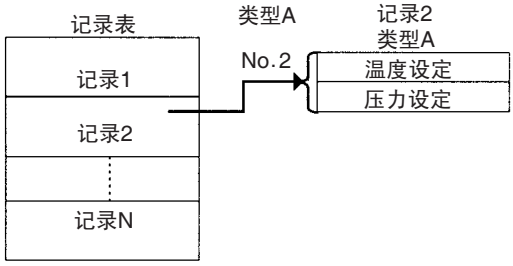
记录表格指令作用于专门定义的等长记录组成的数据表格。记录可通过记录数存取以便于处理。

指令	功能	说明
DIM(631)	定义一记录表。	说明每个记录的长度和记录数。
SETR(635)	设定记录位置。	在特定变址寄存器中写入特定记录的位置 （记录开始的 PLC 存储器地址）。
GETR(636)	获得记录位置。	回到记录的记录号，它包含有特定变址寄存器的 PLC 存储器地址。

注 通过变址寄存器使记录数和字地址相关。在 SETR（635）中规定一个记录数，用于在变址寄存器中存储那个记录开始时的 PLC 存储器地址。当需要记录中的数据时，给那个变址寄存器加上所需的偏置量来存取记录中的任一个字。

与变址寄存器一起使用记录表格指令执行以下各种操作：读 / 写记录数据，寻找记录，记录数据分类，比较记录数据，和运用记录数据进行计算。

记录表格典型应用是将一种产品的不同类型制造数据 （如温度和压力设定）以记录格式存储，并且通过改变记录数就可从一种类型变到另一种。



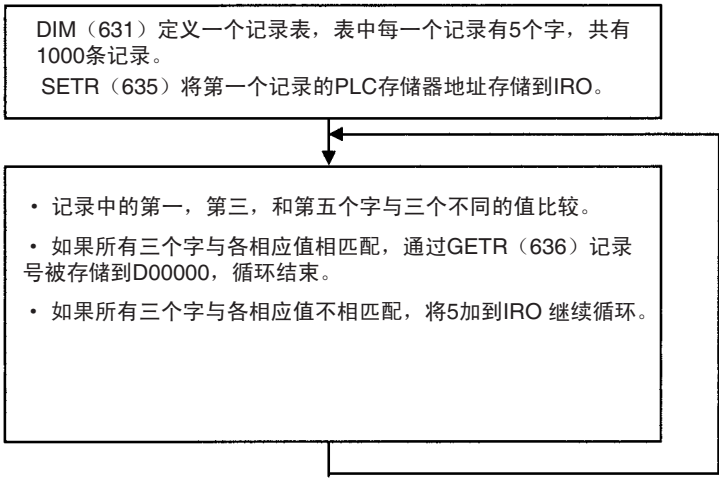
一般，以下述步骤使用记录表格：

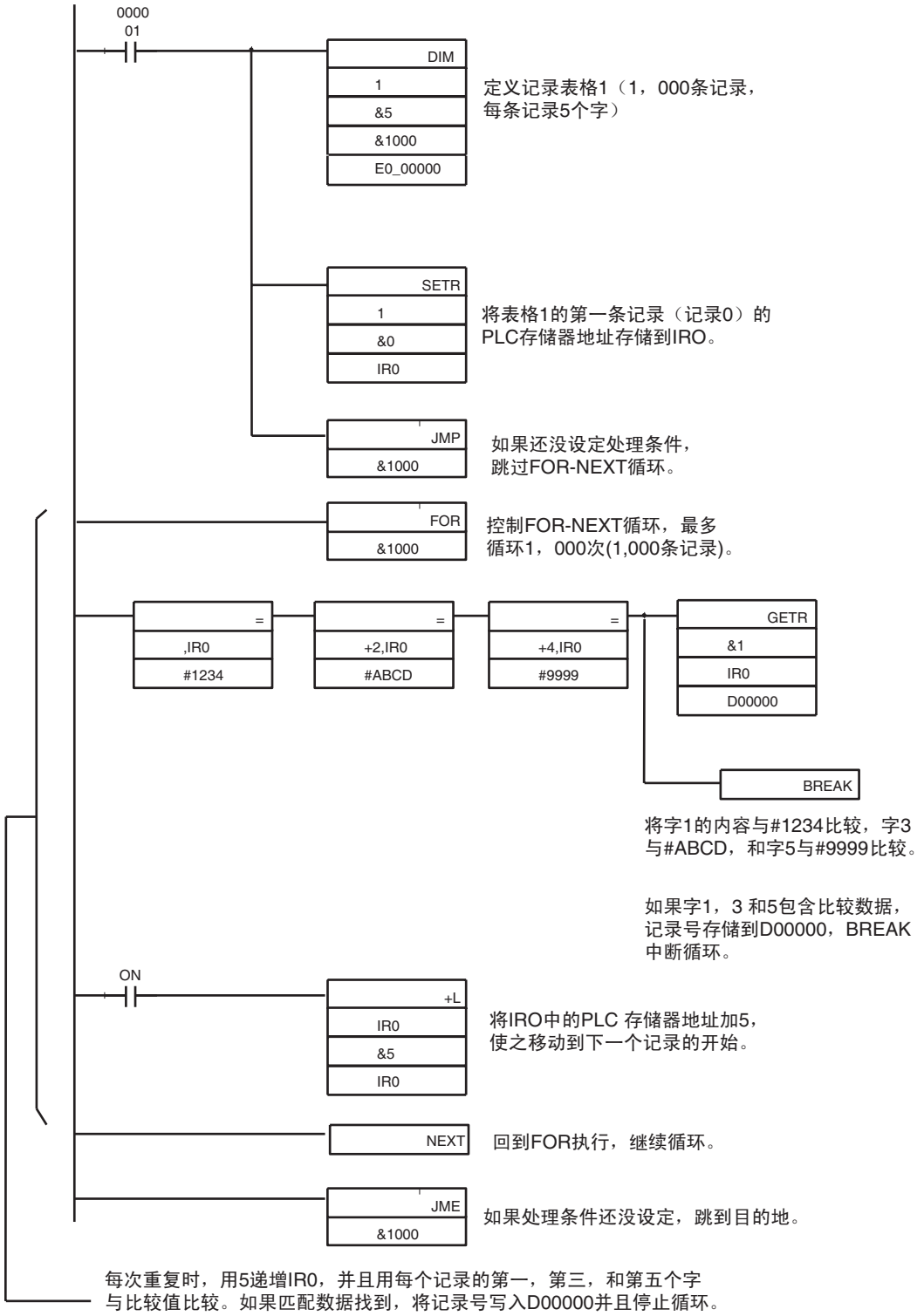
1,2,3...

1. 用 DIM（631）定义记录表格的指令，用 SETR（635）在变址寄存器中设定一个记录的 PLC 存储器地址。
2. 在变址寄存器中偏置或递增 PLC 存储器地址，对记录中的字进行读或比较。
3. 在变址寄存器中偏置或递增 PLC 存储器地址从而转到另一个记录。
4. 如果需要，重复步骤 2 和 3。

例

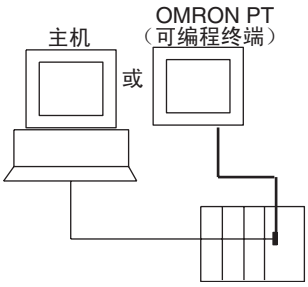
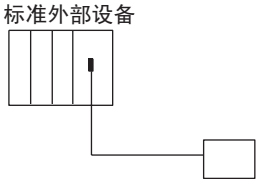
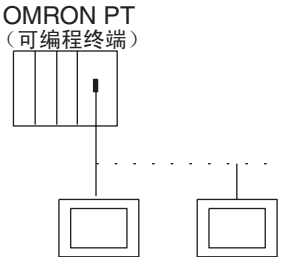
下例中应用变址寄存器和记录表格指令，在每个记录中与字 1，3 和 5 比较三个值。如果发现一个匹配，记录数被存储在 D00000。

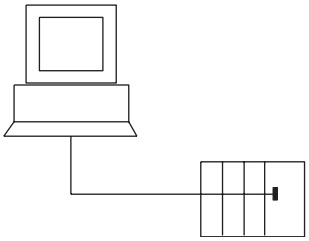
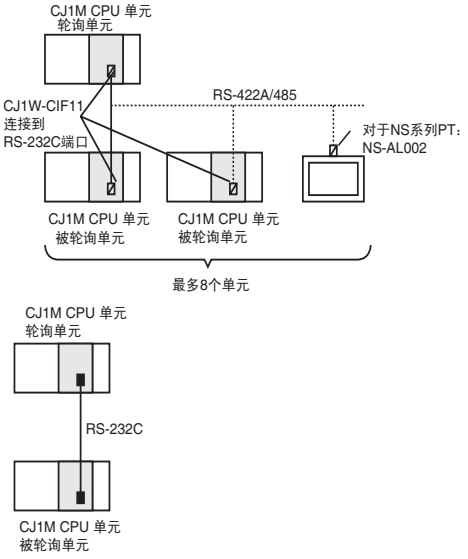



0000
01
DIM
1
&5
&1000
E0_00000
定义记录表格1（1，000条记录，
每条记录5个字）
SETR
1
&0
IR0
将表格1的第一条记录（记录0）的
PLC存储器地址存储到IRO。
JMP
&1000
如果还没设定处理条件，
跳过FOR-NEXT循环。
FOR
&1000
控制FOR-NEXT循环，最多
循环1，000次(1,000条记录)。
=
,IR0
#1234
=
+2,IR0
#ABCD
=
+4,IR0
#9999
GETR
&1
IR0
D00000
BREAK
将字1的内容与#1234比较，字3
与#ABCD，和字5与#9999比较。
如果字1，3 和5包含比较数据，
记录号存储到D00000，BREAK
中断循环。
ON
+L
IR0
&5
IR0
将IRO中的PLC 存储器地址加5，
使之移动到下一个记录的开始。
NEXT
回到FOR执行，继续循环。
JME
&1000
如果处理条件还没设定，跳到目的地。
每次重复时，用5递增IR0，并且用每个记录的第一，第三，和第五个字
与比较值比较。如果匹配数据找到，将记录号写入D00000并且停止循环。

6-3 串行通信

CS/CJ 系列 CPU 单元支持以下串行通信功能。主机链接通信和无协议通信将在本章详细讨论。

协议	连接	说明	端口	
			外设	RS-232C
上位机链接		1) 各种控制命令（如：I/O 存储器的读和写，改变操作模式，和通过从主机到 CPU 单元发布主机链接命令或 FINS 命令执行位强迫设定 / 复位。 2) 也可以从 CPU 单元发送 FINS 命令到主机传递数据或信息。 使用主机链接通信监控数据（如：在 PLC 中的操作状态，错误信息，和质量数据）或发送数据（如：将生产计划信息发送到 PLC）。	允许	允许
无协议通信		与连接到 RS-232C 端口标准设备的通信，无命令响应格式。通过执行程序从传送端口传递数据或在接受端口中读出数据，而不是使用 TDX（236）和 RDX（235）指令。	不允许	允许
1:N 或 1:1 NT 链接		在 CPU 单元中不使用通信程序，数据可以在 PT 下进行交换。	允许	允许

协议	连接	说明	端口	
			外设	RS-232C
外设总线	编程工具 （非手持式编程器） 	使用编程工具 （而不是手持式编程器）可具有高速度通信。 （不能通过调制解调器远程编程）	允许	允许
串行 PLC 链接 （仅限于 CJ1M）		不超过 10 个字的每一个单元可被不超过 9 个 CPU 单元共享 （包括一个轮询单元和 8 个被轮询单元。 可将一个 RS-422 转换器连接到每个 CPU 单元的 RS-232C 端口，通过 RS-422A/485 进行通信，或通过一个 RS-232C 连接两个 CPU 单元可以相互通信。 串行 CPU 链接也可包括 PT[作为被轮询单元通过 NT 链接 （1：N） 与 CJ1M CPU 单元相联]	不允许	允许

这里，我们将讨论上位机链接通信和非协议通信。

6-3-1 上位机链接通信

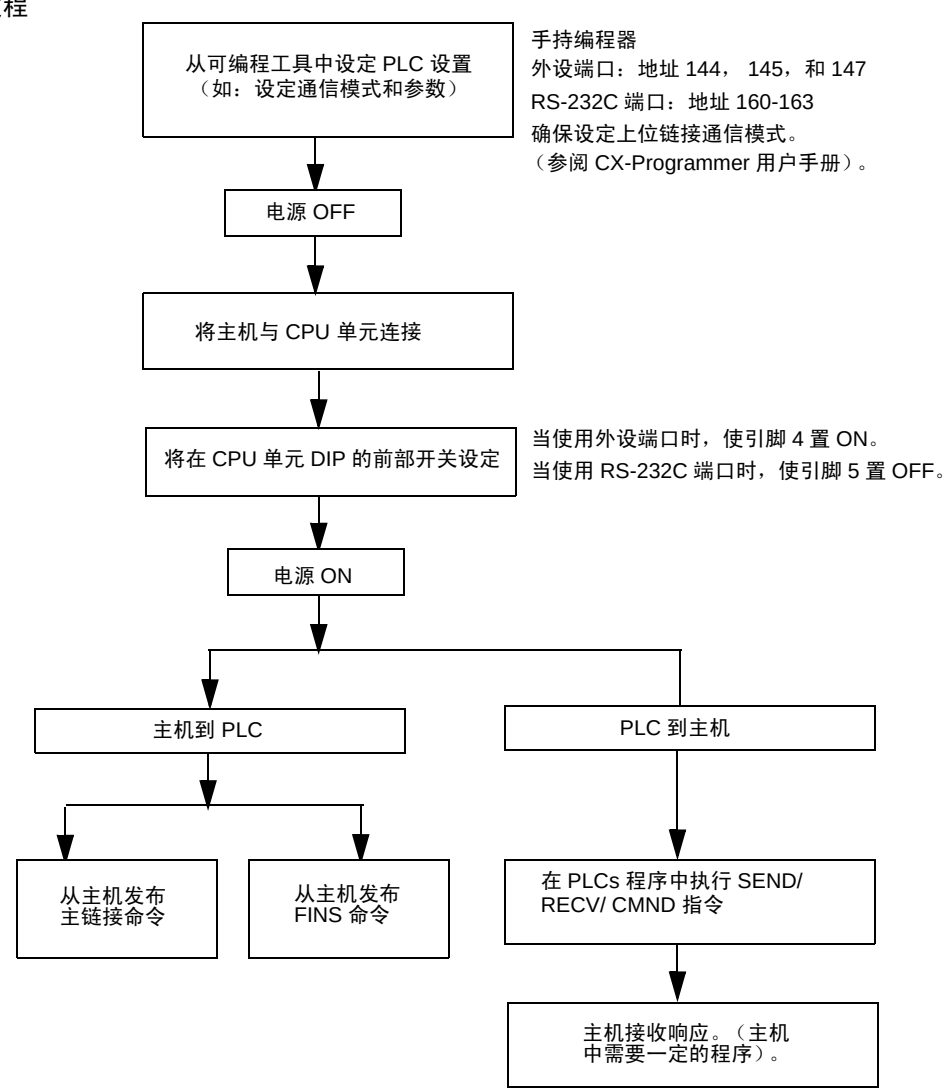
下表显示了在CS/CJ PLC中已有的上位机链接通信功能。选择一种最适合你的应用的方法。

命令流程	命令类型	通信方法	配置
主机 PLC	主链接命令 主链接命令	在主机中建立帧，并向 PLC 发布命令。接收 PLC 的响应。 应用： 当通信主要是从主机传递到 PLC 时，使用这个方法。	以1：1或1：N系统直接连接主机 或 命令
	FINS命令[1] （使用主链接首标和终止） FINS 首标 终止	在主机中建立帧，并向 PLC 发布命令。接收 PLC 的响应。 应用： 当通信主要是从主机传递到在网络中的 PLC 时，使用这个方法。	以1：1或1：N系统直接连接主机 或 命令
			通过主机与网络中的其它PLC通信（从主机链接转换到网络协议）。 命令
PLC 主机	FINS命令[2] （使用主链接首标和终止） FINS 首标 终止	用 CPU 单元中的 SEND/RECV/CMND 指令发布帧。接收主机的响应。 应用： 当通信主要是从 PLC 传递到主机，传递状态信息（如：错误信息）时，使用这个方法。	在1：1 系统直接连接主机。 SEND/RECV/CMND 命令
			通过主机与网络中的其它PLC通信（从主机链接转换到网络协议）。 SEND/RECV/CMND 命令

注 1. 在命令从主机发出之前， FINS 命令必须附加一个主链接首标和终止。

2. 从 PLC 发出的 FINS 命令已附加了一个主链接首标和终止。主机中必须有一个程序用来分析 FINS 命令和回复合适的响应。

过程



主链接命令

下表列出上位机链接命令。更详细的请参阅 *C 系列主链接单元系统手册*（W143）。

首标码	名称	功能
RR	CIO AREA READ	从指定的字开始，读出 CIO 区域指定数量的字的内容。
RL	LINK AREA READ	从指定的字开始，读出链接区域指定数量的字的内容。
RH	HR AREA READ	从指定的字开始，读出保持区域指定数量的字的内容。
RC	PV READ	从指定的定时器 / 计数器开始，读出指定数量的定时器 / 计数器 PV（当前值）的内容。
RG	T/C STATUS READ	从指定的定时器 / 计数器开始，读出指定数量的定时器 / 计数器完成标帜的状态。
RD	DM AREA READ	从指定的字开始，读出 DM 区域指定数量的字的内容。
RJ	AR AREA READ	从指定的字开始，读出辅助区域指定数量的字的内容。

首标码	名称	功能
RE	EM AREA READ	从指定的字开始，读出 EM 区域指定数量的字的内容。
WR	CIO AREA WRITE	从指定的字开始，将指定数据 （仅限于字单元）写入 CIO 区域。
WL	LINK AREA WRITE	从指定的字开始，将指定数据 （仅限于字单元）写入链接区域。
WH	HR AREA WRITE	从指定的字开始，将指定数据 （仅限于字单元）写入保持区域。
WC	PV WRITE	从指定的计时器 / 计数器开始，写指定数量的定时器 / 计数器的 PV （当前值）。
WD	DM AREA WRITE	从指定的字开始，将指定数据 （仅限于字单元）写入 DM 区域。
WJ	AR AREA WRITE	从指定的字开始，将指定数据 （仅限于字单元）写入辅助区域。
WE	EM AREA WRITE	从指定的字开始，将指定数据 （仅限于字单元）写入 EM 区域。
R#	SV READ 1	在指定的定时器 / 计数器指令的 SV 中，读出 4 位 BCD 常数或字地址。
R$	SV READ 2	寻找在指定程序地址开始的指定的定时器 / 计数器指令，并在 SV 中读出 4 位常数或字地址。
R%	SV READ 3	寻找在指定程序地址开始的指定的定时器 / 计数器指令，并在 SV 中读出 4 位 BCD 常数或字地址。
W#	SV CHANGE 1	在指定的定时器 / 计数器指令的 SV 中改变 4 位 BCD 常数或字地址 。
W$	SV CHANGE 2	寻找在指定程序地址开始的指定的定时器 / 计数器指令，并在 SV 中改变 4 位常数或字地址。
W%	SV CHANGE 3	寻找在指定程序地址开始的指定的定时器 / 计数器指令，并在 SV 中改变 4 位常数或字地址。
MS	STATUS READ	读出 CPU 单元的操作状态 （操作模式，强迫设定 / 重设定状态，致命差错状态）。
SC	STATUS CHANGE	改变 CPU 单元的操作模式。
MF	ERROR READ	在 CPU 单元中读出和清除差错 （一般和致命）。
KS	FORCE SET	强制设置指定位。
KR	FORCE RESET	强制复位指特定位。
FK	MULTIPLE FORCE SET/RESET	强制设置，强制复位，或清除指定位的强迫状态。
KC	FORCE SET/RESET CANCEL	取消所有强制设置和强制复位的位强制状态。
MM	PLC MODEL READ	读出 PLC 的模式类型。
TS	TEST	主机传递的一个数据块的返回，不改变 。
RP	PROGRAM READ	用机器语言 （目标码）读出 CPU 单元的用户程序区域的内容。
WP	PROGRAM WRITE	写机器语言 （目标码）程序，从主机传递到 CPU 单元的用户程序区域。
MI	I/O TABLE GENERATE	用实际 I/O 表建立一个登记 I/O 表。
QQMR	COMPOUND COMMAND	将所需的位和字登记在表中。
QQIR	COMPOUND READ	从 I/O 存储器中读出登记的字和位。
XZ	ABORT （仅限于命令）	中止当前正在处理的主链接命令。

首标码	名称	功能
**	INITIALIZE （仅限于命令）	所有与主机连接的 PLC 的传递控制过程初始化。
IC	未定义的命令 （仅限于响应）	如果没有识别出命令的首标码，这个响应是返回。

FINS 命令

下表列出 FINS 命令。更详细的请参阅 FINS 命令参考手册 （W227）。

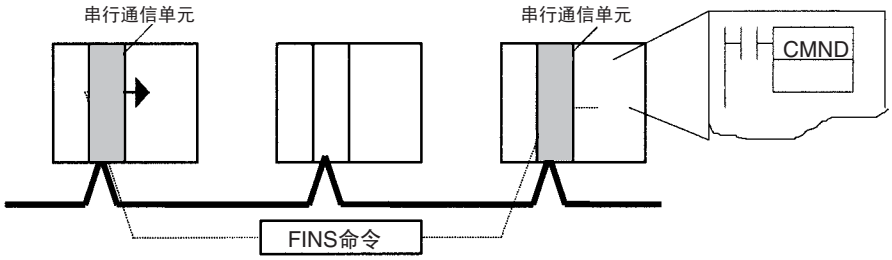
类型	命令码		名称	功能
I/O 存储器区域存取	01	01	MEMORY AREA READ	从 I/O 存储器区域读出连续数据。
	01	02	MEMORY AREA WRITE	将连续数据写入 I/O 存储器区域。
	01	03	MEMORY AREA FILL	I/O 存储器的指定范围中填入相同数据。
	01	04	MULTIPLE MEMORY AREA READ	从 I/O 存储器区域读出非连续数据。
	01	05	MEMORY AREA TRANSFER	从 I/O 存储器区域的一部分连续数据复制和传递到另一部分中去。
参数区域存取	02	01	PARAMETER AREA READ	从参数区域读出连续数据。
	02	02	PARAMETER AREA WRITE	将连续数据写入参数区域。
	02	03	PARAMETER AREA FILL	参数区域的指定范围中填入相同数据。
程序区域存取	03	06	PROGRAM AREA READ	从用户程序区域读出数据。
	03	07	PROGRAM AREA WRITE	将数据写入用户程序区域。
	03	08	PROGRAM AREA CLEAR	清除用户程序区域的指定范围。
执行控制	04	01	RUN	将 CPU 单元切换到 RUN， MONITOR，或 DEBUG 模式。
	04	02	STOP	将 CPU 单元切换到 PROGRAM 模式。
结构读出	05	01	CONTROLLER DATA READ	读出 CPU 单元信息。
	05	02	CONNECTION DATA READ	读出特定单元的模式数。
状态读出	06	01	CONTROLLER STATUS READ	读出 CPU 单元状态信息。
	06	20	CYCLE TIME READ	读出平均，最大，和最小周期。
时钟存取	07	01	CLOCK READ	读出时钟。
	07	02	CLOCK WRITE	设定时钟。
信息存取	09	20	MESSAGE READ/CLEAR	读出 / 清除信息和 FAL （S）信息。
访问权利	0C	01	ACCESS RIGHT ACQUIRE	如果没有其它设备占用它，获得访问权。
	0C	02	ACCESS RIGHT FORCED ACQUIRE	即使另一种设备目前占用它，获得访问权。
	0C	03	ACCESS RIGHT RELEASE	不管那个设备占用它，释放访问权。
错误存取	21	01	ERROR CLEAR	清除错误和错误信息。
	21	02	ERROR LOG READ	读出错误运行记录。
	21	03	ERROR LOG CLEAR	清楚错误运行记录，指针指向零。

类型	命令码		名称	功能
文件存储器	22	01	FILE NAME READ	读出文件存储器的文件信息。
	22	02	SINGLE FILE READ	从一个文件的指定点读出指定量的数据
	22	03	SINGLE FILE WRITE	从一个文件的指定点写入指定量的数据
	22	04	FILE MEMORY FORMAT	格式文件存储器。
	22	05	FILE DELETE	从文件存储器中删除指定文件。
	22	07	FILE COPY	在一个文件存储器中，或在一个系统中的两个文件存储器之间复制一个文件。
	22	08	FILE NAME CHANGE	改变文件名。
	22	0A	I/O MEMORY AREA FILE TRANSFER	在 I/O 存储器区域和文件存储器之间传递或比较数据。
	22	0B	PARAMETER AREA FILE TRANSFER	在参数区域和文件存储器之间传递或比较数据。
	22	0C	PROGRAM AREA FILE TRANSFER	在程序区域和文件存储器之间传递或比较数据。
	22	15	CREATE/DELETE DIRECTORY	建立或删除一个目录
强迫状态	23	01	FORCED SET/RESET	指定位的强制置位，强制复位，或清除强制状态。
	23	02	FORCED SET/RESET CANCEL	取消所有强制置位，强制复位位的强制状态。

信息通信功能

列于上表中的 FINS 命令也可以通过网络从其它 PLC 传递到 CPU 单元。当通过网络传递 FINS 命令时注意以下几点。

- 为了传递 FINS 命令，CPU 总线单元（如：控制器链接单元或以太网单元）必须安装在本地 PLC 和目的地 PLC 上。
- 从 CPU 单元的程序中使用 CMND （490）发布 FINS 命令。
- FINS 命令最多可传递到 3 个网站。网站可以是相同类型或不同类型。



有关信息通信功能的更详细的内容请参阅 CPU 总线单元操作手册。

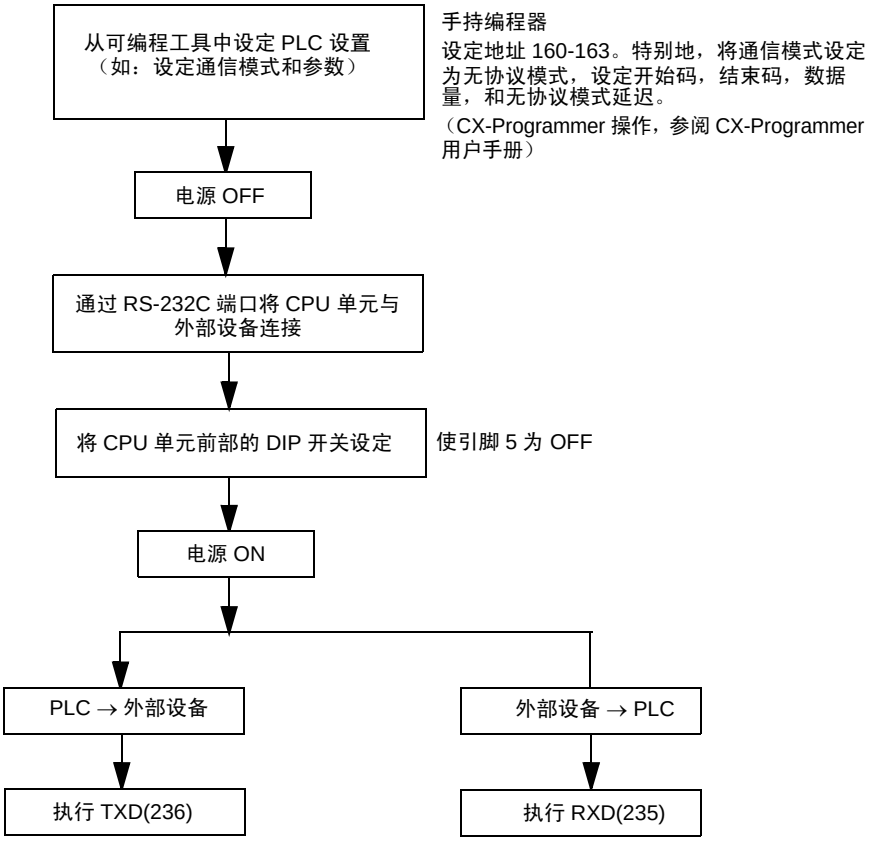
6-3-2 无协议通信

下表列出 CS/CJ PLC 中已有的无协议通信功能。

传递方向	方法	数据最大量	帧格式		其它功能
			开始码	结束码	
数据传递 （PLC → 外部设备）	在程序中执行 TXD(236)	256 字节	是：00 ～ FF 否：无	是： 00 ～ FF 或 CR+LF 否：无	传送延迟时间（TXD 执行和从指定端口传送数据之间的延迟）：0 ～ 99， 990ms（单位：10ms）
数据接收 （外部设备 → PLC）	在程序中执行 RXD(235)	256 字节			---

注 可以在 PLC 设置中（地址 162）规定传递延迟或“无协议模式延迟”。这种设定在 TXD （236）执行和从指定端口数据的传送之间引起延迟不超过 30 秒。

步骤



信息帧格式

数据可以位于开始码和结束码之间，通过 TXD （236）传递， RXD （235）可以接收具有相同格式的帧。当用 TXD （236）传递时，仅传递 I/O 存储器中的数据；当用 RXD （235）接收时，仅接收存储在 I/O 存储器中的数据。无协议模式中，可以传递的字节不超过 256 （包括开始和结束码）。

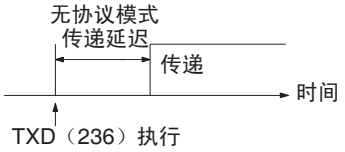
下表给出在无协议模式中用于传递和接收的信息格式。PLC 设置中设定的开始码和结束码决定了格式。

开始码设定	结束码设定		
	否	是	CR+LF
否	数据 (*数据*：最大 256 字节)	*数据* +ED (*数据*：最大 255 字节)	*数据* +CR+LF (*数据*：最大 254 字节)
是	ST+ *数据* (*数据*：最大 255 字节)	ST+ *数据* +ED (*数据*：最大 254 字节)	ST+ *数据*：+CR+LF (*数据*：最大 253 字节)

- 当使用的开始码超过一个时，第一个开始码有效。
- 当使用的结束码超过一个时，第一个结束码有效。

注 1. 如果将传递的数据包含有结束码，数据传递会在中途停止。这种情况下，将结束码改成 CR+LF。

2. TXD （236）执行后， PLC 设置中有一种设定 （地址 162：无协议模式延迟）将引起数据传递的延迟。



有关 TXD （236）和 RXD （235）更详细的内容，请参考 *CJ 系列可编程控制器编程手册*。

6-3-3 NT 链接 （1：N 模式）

CS/CJ 系列中，使用 NT 链接（1：N 模式）在 PT（可编程终端）可实现通信。

注 使用 1：1 模式 NT 链接协议不能通信。

通过使用 PT 系统菜单和下列 PLC 设置设定 （CS 系前 EV1 CS1 CPU 单元不支持该功能），除了上述 标准 NT 链接之外，可得到高速 NT 链接。然而，高速 NT 链接仅限于 NT31 （C） V2 或 NT631 （C） V2 PT。

PLC 设置

通信端口	手持编程器设定地址	名称	设定内容	默认值	其它条件
外设端口	144 位：8 ～ 11	串行通信模式	02 Hex：NT 链接（1:N 模式）	00 Hex：主链接	CPU 单元 DIP 开关上的引脚 4 置 ON
	145 位：0 ～ 7	波特率	00 ～ 09 Hex：标准 NT 链接 0A Hex：高速 NT 链接 （见注 1）	00 Hex：标准 NT 链接	
	150 位：0 ～ 3	NT 链接模式最大单元数	0 ～ 7 Hex	0 Hex（No. 0 最大单元）	---
RS-232C 端口	160 位：8 ～ 11	串行通信模式	02 Hex：NT 链接（1:N 模式）	00 Hex：主链接	CPU 单元 DIP 开关上的引脚 5 置 OFF
	161 位：0 ～ 7	波特率	00 ～ 09 Hex：标准 NT 链接 0A Hex：高速 NT 链接 （见注 1）	00 Hex：标准 NT 链接	
	166 位：0 ～ 3	NT 链接模式最大单元数	0 ～ 7 Hex	0 Hex（No. 0 最大单元）	---

注 用 CX-Programmer 设定时，设定波特率 115， 200 bps （比特 / 秒）。

PT 系统菜单

设定 PT 如下：

1,2,3...

1. 在 PT 单元，系统菜单下的存储器开关菜单，从 Comm.A 方法或 Comm.B 方法中选择 NT 链接 （1：N）。
2. 按下 SET 按键式开关，将 Comm. 速度设定为高速。

6-3-4 串行 PLC 链接 （仅限于 CJ1M CPU 单元）

概述

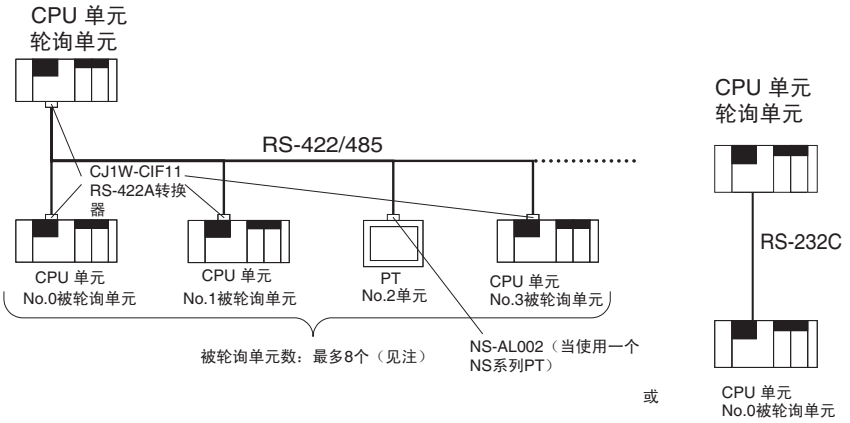
只有 CJ1M CPU 单元支持串行 PLC 链接。通过内置 RS-232C 端口 （不要求专门编程），允许数据在 CJ1M CPU 单元之间交换。串行链接字（CIO3100 ～ CIO 3199）位于存储器中。RS-232C 连接可用在 CPU 单元之间，或 RS-422A/485连接可将RS-232C-RS-422A/485转换器连接到RS-232C端口。应用CJ1W-CIF11 RS-422A 转换器，可在 RS-232C 和 RS-422A/485 之间转换。

设定为 NT 链接 （1：N）通信的 PT，可以同时用于相同的网络。被轮询的 PT 使用网络与轮询 CPU 单元在一个 NT 链接（1：N）中通信。然而， PT 连接以后，与 PT's 单元数相对应的在串行 PLC 链接字中的地址没有定义。

说明

项目	说明
连接方法	通过 CPU 单元的 RS-232C 端口， RS-232C 或 RS-422A/485 连接
已分配数据区域	串行 PLC 链接字： CIO 3100 ～ CIO3199 （每个 CPU 单元可分配字不超过 10 个）
单元数	最多 9 个单元，包括一个轮询单元和八个被轮询单元 （PT 可放置在一个 NT 链接 （1：N）的相同网络中，但它必须作为八个被轮询单元中的一个）。

系统结构



注 当一个设定于串行 PLC 链接通信的 PT 是在相同的网络时，最多 8 个单元（包括 PT 和被轮询单元）可以与轮询单元连接。

数据刷新方法

可用下面两个方法刷新数据。

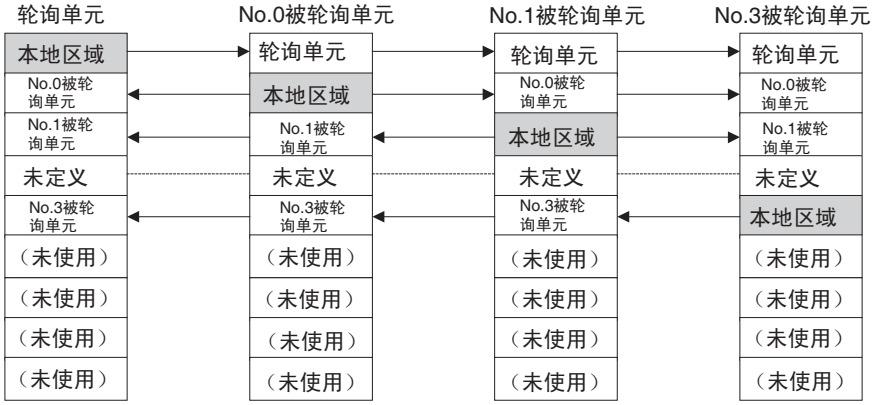
- 完整链接方法
- 轮询单元链接方法

完整链接方法

串行 PLC 链接中所有节点的数据同时是轮询单元和被轮询单元的映射。（唯一例外是：分配给已连接的 PT 的单元号的地址和被轮询单元的地址在网络中是不存在的。这些数据在所有节点上是未定义的）。

例：完整链接方法，最高单元数：3

下图中，被轮询单元 No.2 是在网络中的 PT 或一个不存在单元，因此分配给被轮询单元 No.2 的区域在所有节点上是未定义的。

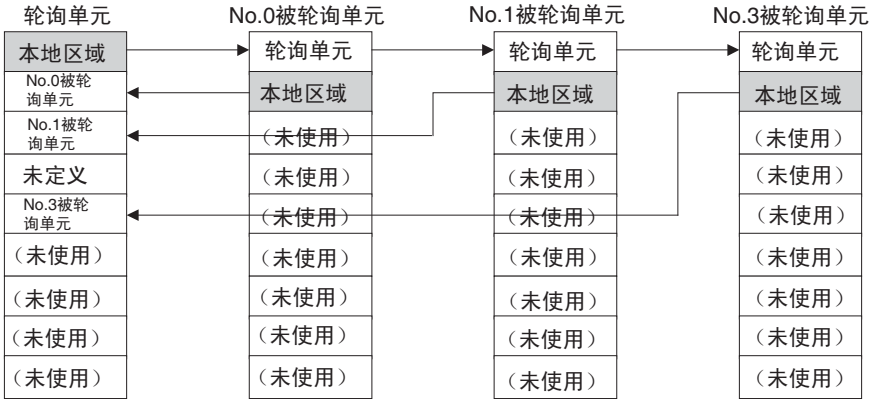


轮询单元链接方法

串行 PLC 链接中所有被轮询单元的数据都映射到轮询单元，并且每一个被轮询单元都仅映射轮询单元的数据。轮询单元链接方法的优势在于：分配给本地被轮询单元数据的地址与每个被轮询单元的地址相同，允许通过普通梯形编程存取数据。在网络中不存在的分配给 PT 单元数或被轮询单元的区域仅在轮询单元中未被定义）。

例：轮询单元链接方法，最高单元数：3

下图中，被轮询单元 No.2 是在网络中的 PT 或一个不存在单元，因此在轮询单元中的相应区域是未定义的。



分配字

完整链接方法

链接字	1 字	2 字	3 字	～	10 字
轮询单元	CIO 3100	CIO 3100 ～ CIO 3101	CIO 3100 ～ CIO 3102		CIO 3100 ～ CIO 3109
No. 0 被轮询单元	CIO 3101	CIO 3102 ～ CIO 3103	CIO 3103 ～ CIO 3105		CIO 3110 ～ CIO 3119
No. 1 被轮询单元	CIO 3102	CIO 3104 ～ CIO 3105	CIO 3106 ～ CIO 3108		CIO 3120 ～ CIO 3129
No. 2 被轮询单元	CIO 3103	CIO 3106 ～ CIO 3107	CIO 3109 ～ CIO 3111		CIO 3130 ～ CIO 3139
No. 3 被轮询单元	CIO 3104	CIO 3108 ～ CIO 3109	CIO 3112 ～ CIO 3114		CIO 3140 ～ CIO 3149
No. 4 被轮询单元	CIO 3105	CIO 3110 ～ CIO 3111	CIO 3115 ～ CIO 3117		CIO 3150 ～ CIO 3159
No. 5 被轮询单元	CIO 3106	CIO 3112 ～ CIO 3113	CIO 3118 ～ CIO 3120		CIO 3160 ～ CIO 3169
No. 6 被轮询单元	CIO 3107	CIO 3114 ～ CIO 3115	CIO 3121 ～ CIO 3123		CIO 3170 ～ CIO 3179
No. 7 被轮询单元	CIO 3108	CIO 3116 ～ CIO 3117	CIO 3124 ～ CIO 3126		CIO 3180 ～ CIO 3189
未使用	CIO 3109 ～ CIO 3199	CIO 3118 ～ CIO 3199	CIO 3127 ～ CIO 3199		CIO 3190 ～ CIO 3199

轮询单元链接方法

链接字	1 字	2 字	3 字	～	10 字
轮询单元	CIO 3100	CIO 3100 ～ CIO 3101	CIO 3100 ～ CIO 3102		CIO 3100 ～ CIO 3109
No. 0 被轮询单元	CIO 3101	CIO 3102 ～ CIO 3103	CIO 3103 ～ CIO 3105		CIO 3110 ～ CIO 3119
No. 1 被轮询单元	CIO 3101	CIO 3102 ～ CIO 3103	CIO 3103 ～ CIO 3105		CIO 3110 ～ CIO 3119
No. 2 被轮询单元	CIO 3101	CIO 3102 ～ CIO 3103	CIO 3103 ～ CIO 3105		CIO 3110 ～ CIO 3119
No. 3 被轮询单元	CIO 3101	CIO 3102 ～ CIO 3103	CIO 3103 ～ CIO 3105		CIO 3110 ～ CIO 3119
No. 4 被轮询单元	CIO 3101	CIO 3102 ～ CIO 3103	CIO 3103 ～ CIO 3105		CIO 3110 ～ CIO 3119
No. 5 被轮询单元	CIO 3101	CIO 3102 ～ CIO 3103	CIO 3103 ～ CIO 3105		CIO 3110 ～ CIO 3119
No. 6 被轮询单元	CIO 3101	CIO 3102 ～ CIO 3103	CIO 3103 ～ CIO 3105		CIO 3110 ～ CIO 3119
No. 7 被轮询单元	CIO 3101	CIO 3102 ～ CIO 3103	CIO 3103 ～ CIO 3105		CIO 3110 ～ CIO 3119
未使用	CIO 3102 ～ CIO 3199	CIO 3104 ～ CIO 3199	CIO 3106 ～ CIO 3199		CIO 3120 ～ CIO 3199

步骤

串行 PLC 链接操作，在 PLC 设置中根据以下设定。

轮询单元的设定

1,2,3...

1. 将 RS-232C 通信端口的串行通信模式设定至串行 PLC 链接（轮询单元）。
2. 链接方法设定为完整链接方法或轮询单元链接方法。
3. 设定链接字的数 （每个单元 <10 字）。
4. 串行 PLC 链接中设定最大单元数 （0 ～ 7）。

被轮询单元的设定

1,2,3...

1. 将RS-232C通信端口的串行通信模式设定至串行PLC链接（被轮询单元）。
2. 设定串行 PLC 链接被轮询单元的单元号。

PLC 设置

轮询单元的设定

项目		PLC 地址		设定值	默认	刷新时间
		字	位			
RS-232C 端口设定	串行通信模式	160	11 ～ 08	8 hex：串行 PLC 链接轮询单元	0 hex	每个循环（除了执行 STUP(237) 指令时立即刷新之外）
	端口波特率	161	07 ～ 00	00 hex：标准 0A hex：高速	00 hex	
	链接方法	166	15	0：完整链接 1：轮询单元链接	0	
	链接字的数		07 ～ 04	1 ～ A hex	0 hex （见注）	
	最高单元数		03 ～ 00	0 ～ 7 hex	0 hex	

注　当默认设定为 0 hex 时，自动分配 10 字 （A hex）。

轮询单元的设定

项目		PLC 地址		设定值	默认	刷新时间
		字	位			
RS-232C 端口设定	串行通信模式	160	11 ～ 08	7 hex：串行 PLC 链接轮询单元	0 hex	每个循环（除了执行 STUP(237) 指令时立即刷新之外）
	端口波特率	161	07 ～ 00	00 hex：标准 0A hex：高速	00 hex （见注）	
	被轮询单元单元数	167	03 ～ 00	0 ～ 7 hex	0 hex	

注　默认波特率为 38.4 kbps。

相关辅助区域标志

名称	地址	详细内容	读 / 写	刷新时间
RS-232C 端口通信错误标志	A39204	在 RS-232C 端口当通信错误发生时 变为 ON。 1：错误 0：正常	读	• 电源接通时清除。 • 在 RS-232C 端口当通信错误发生时变为 ON。 • 当端口重新起动时变为 OFF。 • 外设总线模式 NT 链接模式取消 。
带有 PT 标志的 RS-232C 端口通信 （见注）	A39300 ～ A39307	在 RS-232C 端口用于 NT 链接模式 时，与单元执行通信相应的位为 ON。位 00 ～ 07 相对应的单元数分别是 0 ～ 7。 1：通信 0：不通信	读	• 电源接通时清除。 • 在 NT 链接模式或串行 PLC 链接模式，通过 RS-232C 端口，与 PT/ 被轮询单元的单元号相对应的位为 ON。 • 位 00 ～ 07 相对应的单元号分别是 0 ～ 7。
RS-232C 端口重新启动位	A52600	这个位变为 ON 时重新起动 RS-232C 端口	读 / 写	• 电源接通时清除。 • 重新起动 RS-232C 端口时变为 ON，（当通信是在外设总线模式时例外）。 注： 当重新起动的处理完成后，（取决于系统设置）位可能自动变为 OFF。
RS-232C 端口错误标志	A52800 ～ A52807	RS-232C 端口错误发生时，相应的错误码被存储。 位 00：未使用 位 01：未使用 位 02：奇偶错误 位 03：分帧错误 位 04：过速错误 位 05：超时错误 位 06：未使用 位 07：未使用	读 / 写	• 电源接通时清除。 • RS-232C 端口错误发生时，相应的错误码被存储。 • 当 RS-232C 端口重新起动时，根据系统的设置，可能清除标志。 • 外设总线模式期间禁止。 • NT 链接模式下，仅在位 05 有效时 （超时错误）。 串行 PLC 链接模式下，仅以下位有效。 在轮询单元错误： 位 05：超时错误 • 在被轮询单元 CHECK 错误： 位 05：超时错误 位 04：过速错误 位 03：分帧错误
RS-232C 端口设定改变标志	A61902	RS-232C 端口的通信条件改变时，变为 ON。 1：改变 0：不改变	读 / 写	• 电源显示 ON 时清除。 • RS-232C 端口的通信条件设定改变时，变为 ON。 • 当执行 CHANGE SERIAL PORTSETUP 指令 （STUP （237））时，变为 ON。 • 当设定的改变完成后，再变为 OFF。

注 如同现存的 NT （1：N）链接方法，通过读出 RS-232C 端口通信 （A393 数 00 ～ 07 对应单元号 0 ～ 7 的），从轮询单元（CPU 单元）读串行 PLC 链接中 PT 的状态检查有无通信。

6-4 改变定时器 / 计数器当前值 PV 的刷新方式

6-4-1 概述

以前，CS1 CPU 单元仅使用 BCD 执行定时器 / 计数器当前值 PV 刷新方式。因此，所有定时器 / 计数器设定时用 BCD 值输入。其它 CPU 单元（见注 1 和 2）既可用 BCD 方式也可用二进制方式来刷新定时器和计数器指令的当前值 （见注 3）。

当使用二进制时，定时器 / 计数器设定时间可从以前的 0 ～ 9999 扩展到 0 ～ 65535。使用其它指令计算的二进制数据可被用于设定定时器 / 计数器的值。当定时器 / 计数器设定值被指定为一个地址（间接规定）时，定时器 / 计数器 PV 刷新方式也可以被定。（作为 BCD 方式或二进制方式的设定将决定地址字的内容是采用 BCD 值还是二进制值）。

然而，对于 BCD 方式和二进制方式的指令操作数具有差异，因此在改变定时器 / 计数器 PV 刷新方式之前，要检查和了解 BCD 和二进制方式之间的区别。

注 1. 现在，除了 CS1 CPU 单元之外还有：
- CS1-H CPU 单元
- CJ1-H CPU 单元
- CJ1M CPU 单元

2. 对于2002年5月31日前制造的CS1-H/CJ1-H CPU 单元，所带有的定时器/计数器 PV 刷新方式设定为二进制方式，当助记码是由手持编程器监控时，二进制的助记符显示为助记符或 BCD 指令（例：TIMX #0000 &16 显示的是 TIM #0000&16），但是在二进制方式下执行。
3. 只有 3.0 版 CX - Programmer 具有 PV 刷新方式的选择功能。2.1 版或更低版的 CX - Programmer，或手持编程器都不支持方式选择。
4. 2.1 版或更低版的 CX - Programmer 的 CPU 不能读入二进制方式指令的用户程序，但能读入用 BCD 方式指令设定的用户程序。

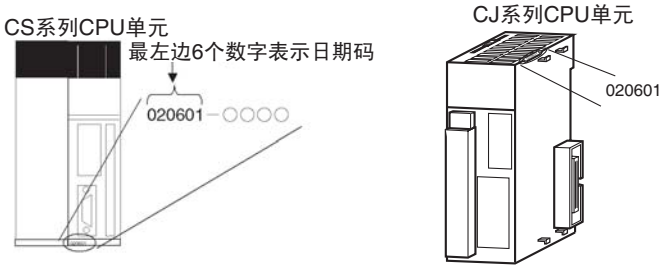
6-4-2　功能说明

项目	详细内容		
定时器 / 计数器 PV刷新方式设定方法	必须使用 3.0 版 CX - Programmer 设定 （2.1 版或更低版的 CX - Programmer 不支持方式选择）。 在 3.0 版 CX - Programmer 的 PLC 属性中设定。		
支持功能的 CPU 单元	批号 No.020601 （2002 年 6 月 1 日制造）或以后（见注 1）中的 CS1-H/CJ1-H CPU 单元，和 CJ1M CPU 单元。		
方式	BCD 方式	二进制方式	
助记码	同以前的模式相同 例：TIM	将 X 加到 BCD 模式助记码上 例：TIMX	
功能码	同以前的模式相同	新码	
PV/SV 范围	#0000 ～ #9999	&0 ～ &65536	#0000 ～ #FFFF
编程工具的 PV 显示（3.0 版 CX- Programmer 或手持编程器）	BCD 例：#0100	10 进制 例：&100	十六进制 例：#64

注　对于 2002 年 5 月 31 日前制造的 CS1-H/CJ1-H CPU 单元，所带有的定时器 / 计数器 PV 刷新方式设定为二进制方式，当助记符是由手持编程器监控时，二进制的助记符显示为助记符或 BCD 指令（例：TIMX #0000 &16 显示的是 TIM #0000&16），但是在二进制方式下执行。

检查 CPU 单元批号

1,2,3...　1. 批号印在 （CS 系列）前屏面底部或在 （CJ 系列）单元的顶部的右角，是由年、月、和日的后两位数组成，排列次序如下图所示。
例：020601 （2002 年 6 月 1 日制造）

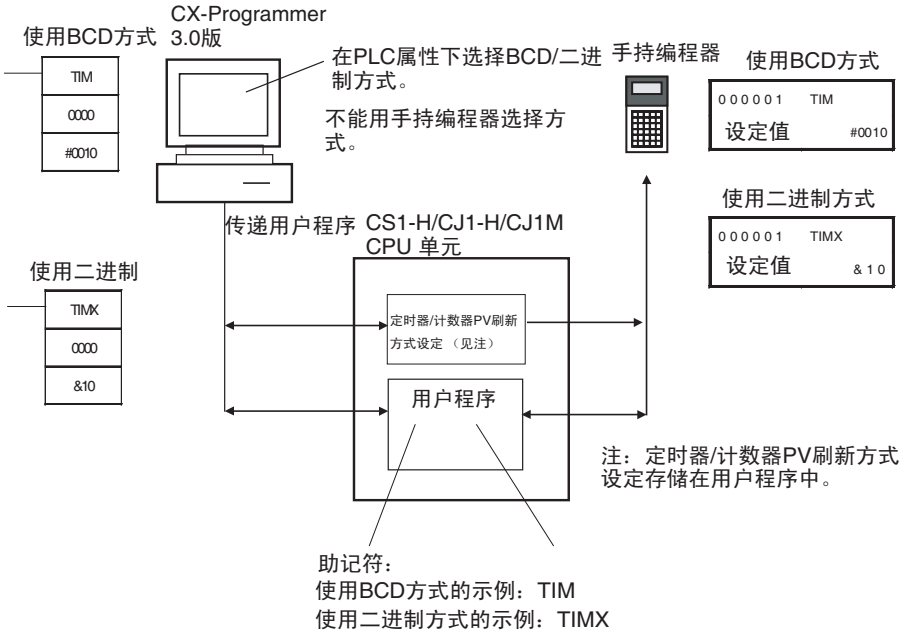


2. 通过将 CX - Programmer 联机，打开 I/O 桌面窗口，选择*单元信息 CPU 单元*，检查选定的方式。将如上图一样的格式表示批号，即：以上述次序，由年、月、和日的后两位数组成。

6-4-3 BCD 方式 / 二进制方式的选择和确认

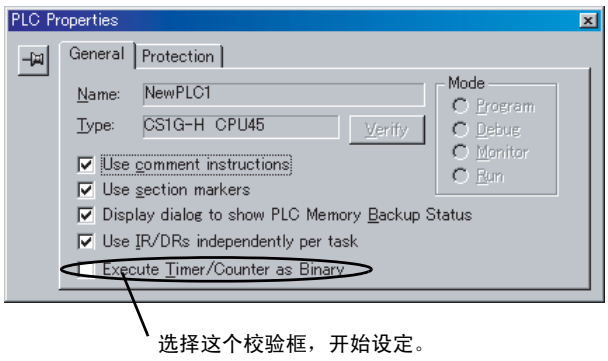
当编写新的程序时，BCD 方式 / 二进制方式的选择是在 3.0 版 CX-Programmer 的 PLC 特性设定中进行。

注 只有 3.0 版或更高版的 CX - Programmer 具有 BCD 方式 / 二进制方式的选择功能。 2.1 版或更低版的 CX - Programmer 不支持方式选择。



BCD 方式 / 二进制方式的选择

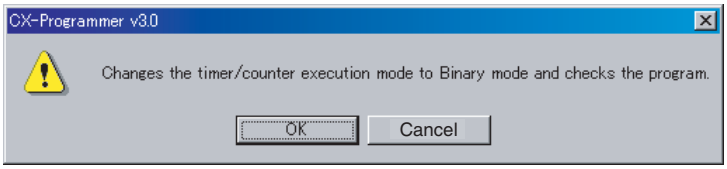
1,2,3... 1. 选定 PLC 名称，点击鼠标右键，选定 *PLC 特性*。



2. 点击通用 General 标记，并且选择*定时器 / 计数器在二进制方式* 下执行。
   - 不选择 （默认）：BCD 方式
   - 选择：二进制方式

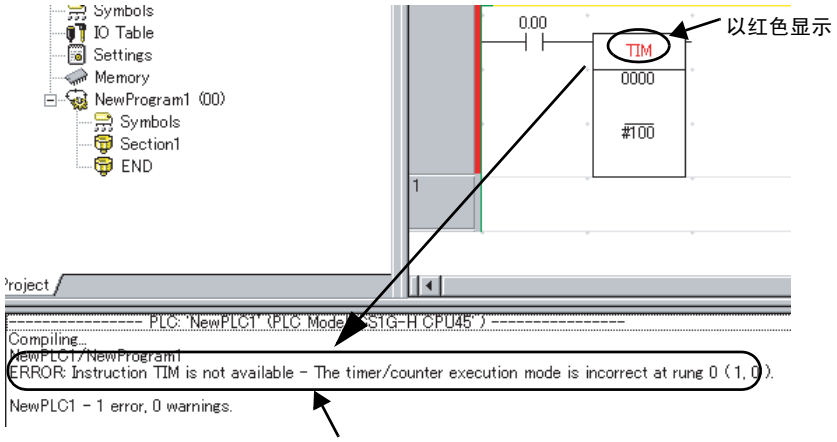
   当用户程序从 CX-Programmer 传递到 CPU 单元时，在 PLC 特性下设定的定时器 / 计数器 PV 刷新方式设定值将存储在 CPU 单元的用户存储器中。

改变设定时，将自动显示下面的对话框图。



点击 OK 按钮，执行程序检查。程序检查结果显示在输出窗口。

例：尽管方式已改变到二进制方式，但是一直在使用 TIM 指令。



程序检查结果显示在输出窗口

例：由于定时器 / 计数器操作方式不同，因此不能使用 TIM。

BCD 方式 / 二进制方式的确认

在辅助区域的 A09915（定时器 / 计数器 PV 刷新方式标志）可被用来检查正在运行的 CPU 单元是 BCD 方式还是在二进制方式。

名称	地址	详细内容
定时器 / 计数器 PV 刷新方式标志	A09915	0：BCD 方式 1：二进制方式

6-4-4　BCD 方式 / 二进制方式助记符和数据

BCD 方式 / 二进制方式助记符

二进制方式助记符是用 BCD 助记符加上后缀 X 表示。

例：TIMER 指令的助记符

BCD 方式：TIM

二进制方式：TIMX

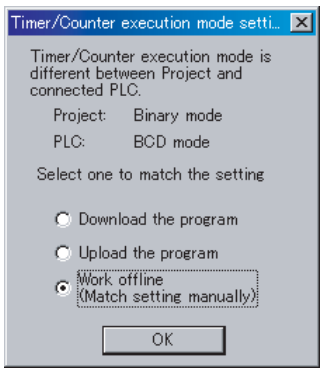
BCD 方式 / 二进制方式数据的显示

PLC 特性	输入和显示符号的意义	设定范围	例：定时器号：0000，设定值：10 s
BCD 方式	符号 # 表示指令值（使用 BCD 方式时，BCD 码值）	#0000 ～ #9999 或 #00000000 ～ #99999999	TIM 0000 #0010
二进制方式	符号 & 表示十进制值	&0 ～ &65535 或 &0 ～ &4294967295	TIMX 0000 &10
	符号 # 表示指令值（使用 BCD 方式时十六进制值）	#0000 ～ #FFFF 或 #0000 ～ #FFFFFFFF	TIMX 0000 #A

注　在 BCD 或二进制方式下使用 CX-Programmer 时，如果输入的数值不包括输入 / 显示符号 # 或 & 表示的常数（例：TIM 0000 0010），那么定时器 / 计数器设定值将作为一个地址输入（例：在 CIO 字 0010 中的一个值被作为设定值）。

6-4-5　限制

- 在同一 CPU 单元中不能同时使用 BCD 方式和二进制方式。
- 当使用手持编程器建立新的用户程序或清除存储器时，定时器/计数器PV刷新方式固定为 BCD 方式。
- 当使用 3.0 版 CX-Programmer 将 CPU 单元联机时，将自动使用设定值。该设定值用于定时器 / 计数器 PV 刷新方式的设定，存储在 CPU 单元的用户存储器中。如果 CPU 设定与 CX-Programmer 项目的设定不同，将会出现错误，此时将不能联机。并显示以下信息。



选择改变 CPU 单元设定来适应 CX-Programmer 对象，或改变 CX-Programmer 器对象相应的设定与 CPU 单元相适应。

- CPU 单元以二进制方式设定时，2.1 版或更低版的 CX-Programmer 不能读出用户程序，但以 BCD 方式设定时可以读出。

- 当一个错误的定时器 / 计数器 PV 刷新方式指令输入时，CX-Programmer 和手持编程器操作之间的区别如下：
  - CX-Programmer：
    如果输入一条指令用的方式与在 PLC 特性下设定的定时器 / 计数器 PV 刷新方式不同，就会出现错误。
    例 ：在对象中的 PLC 被设定为二进制方式，如果输入 TIM 作为助记码，那么出现一个错误。当设定的是 BCD 方式，如果输入 TIMX 作为助记码，那么也出现一个错误。
  - 手持编程器：
    如果给一条指令输入一功能码，用的方式与在 PLC 单元中设定的定时器 / 计数器 PV 刷新方式不同，将自动改变助记符使其与在 PLC 单元中设定的定时器 / 计数器 PV 刷新方式一致。

6-4-6 指令与操作数

指令

指令类型	名称	助记码	
		BCD 方式	二进制方式
定时器和计数器指令	定时器 (100 ms)	TIM	TIMX(550)
	高速定时器 (10 ms)	TIMH(015)	TIMHX(551)
	1 毫秒定时器 (1 ms)	TMHH(540)	TMHHX(552)
	累加定时器 (100 ms)	TTIM(087)	TTIMX(555)
	长定时器 (100 ms)	TIML(542)	TIMLX(553)
	多路输出定时器 (100 ms)	MTIM(543)	MTIMX(554)
	计数器	CNT	CNTX(546)
	可逆计数器	CNTR(012)	CNTRX(548)
	定时器 / 计数器复位	CNR(545)	CNRX(547)
块程序指令	定时器等待 (100 ms)	TIMW(813)	TIMWX(816)
	高速定时器等待 (10 ms)	TMHW(815)	TMHWX(817)
	计数器等待	CNTW(814)	CNTWX(818)

指令和操作数

定时器和计数器指令

定时器 (100 ms)

指令名称	BCD 方式	二进制方式
助记码	TIM	TIM**X**(550)
S (定时器设定值)	#0000 ～ #9999 (BCD)	&0 ～ &65535 (十进制) 或 #0000 ～ #FFFF (十六进制)
设定时间 (单元：0.1 s)	0 ～ 999.9 s	0 ～ 6,553.5 s

高速定时器 (10 ms)

指令名称	BCD 方式	二进制方式
助记码	TIMH(015)	TIMH**X**(551)
S (定时器设定值)	#0000 ～ #9999 (BCD)	&0 ～ &65535 (十进制) 或 #0000 ～ #FFFF (十六进制)
设定时间 (单元：0.01 s)	0 ～ 99.99 s	0 ～ 655.35 s

1 毫秒定时器 (1 ms)

指令名称	BCD 方式	二进制方式
助记码	TMHH(540)	TMHH**X**(552)
S (定时器设定值)	#0000 ～ #9999 (BCD)	&0 ～ &65535 (十进制) 或 #0000 ～ #FFFF (十六进制)
设定时间 (单元：0.001 s)	0 ～ 9.999 s	0 ～ 65.535 s

累加定时器 (100 ms)

指令名称	BCD 方式	二进制方式
助记码	TTIM(087)	TTIM**X**(555)
S (定时器设定值)	#0000 ～ #9999 (BCD)	&0 ～ &65535 (十进制) 或 #0000 ～ #FFFF (十六进制)
设定时间 (单元：0.1 s)	0 ～ 999.9 s	0 ～ 6,553.5 s

长定时器 (100 ms)

指令名称	BCD 方式	二进制方式
助记码	TIML(542)	TIMLX(553)
S, S+1 (定时器设定值)	#00000000 ～ #99999999 (BCD)	&0 ～ &4294967295 (十进制) 或 #0000 ～ #FFFFFFFF（十六进制)
设定时间 (单元：0.1 s)	0 ～ 999.9 s	0 ～ 6,553.5 s

多路输出定时器 (100 ms)

指令名称	BCD 方式	二进制方式
助记码	MTIM(543)	MTIM**X**(554)
S～ S～7 (每一个定时器设定值)	#0000 ～ #9999 (BCD)	&0 ～ &65535 或 #0000 ～ #FFFF (十六进制)
设定时间 (单元：0.1 s)	0 ～ 999.9 s	0 ～ 6,553.5 s

计数器

指令名称	BCD 方式	二进制方式
助记码	CNT	CNT**X**(546)
S (计数器设定值)	#0000 ～ #9999 (BCD)	&0 ～ & 65535 (十进制) 或 #0000 ～ #FFFF (十六进制)
设定	0 ～ 9,999 次	0 ～ 65,535 次

可逆计数器

指令名称	BCD 方式	二进制方式
助记码	CNTR(012)	CNTR**X**(548)
S (计数器设定值)	#0000 ～ #9999 (BCD)	&0 ～ &65535 (十进制) 或 #0000 ～ #FFFF (十六进制)
设定	0 ～ 9,999 次	0 ～ 65,535 次

定时器 / 计数器复位

指令名称	BCD 方式	二进制方式
助记码	CNR(545)	CNR**X**(547)

块程序指令

定时器等待 (100 ms)

指令名称	BCD 方式	二进制方式
助记码	TIMW(813)	TIMW**X**(816)
S (定时器设定值)	#0000 ～ # 9999 (BCD)	&0 ～ &65535 (十进制) 或 #0000 ～ #FFFF (十六进制)
设定时间 (单元：0.1 s)	0 ～ 999.9 s	0 ～ 6,553.5 s

高速定时器等待 (10 ms)

指令名称	BCD 方式	二进制方式
助记码	TMHW(815)	TMHW**X**(817)
S (定时器设定值) 单元：0.01 s	#0000 ～ #9999 (BCD)	&0 ～ &65535 (十进制) 或 #0000 ～ #FFFF (十六进制)
设定时间 (单元：0.01 s)	0 ～ 999.9 s	0 ～ 655.35 s

计数器等待

指令名称	BCD 方式	二进制方式
助记码	CNTW(814)	CNTW**X**(818)
S (计数器设定值)	#0000 ～ #9999 (BCD)	&0 ～ &65535 (十进制) 或 #0000 ～ #FFFF (十六进制)
设定	0 ～ 9,999 次	0 ～ 65,535 次

6-5 采用定时中断作为高精度定时器 （仅限于 CJ1M）

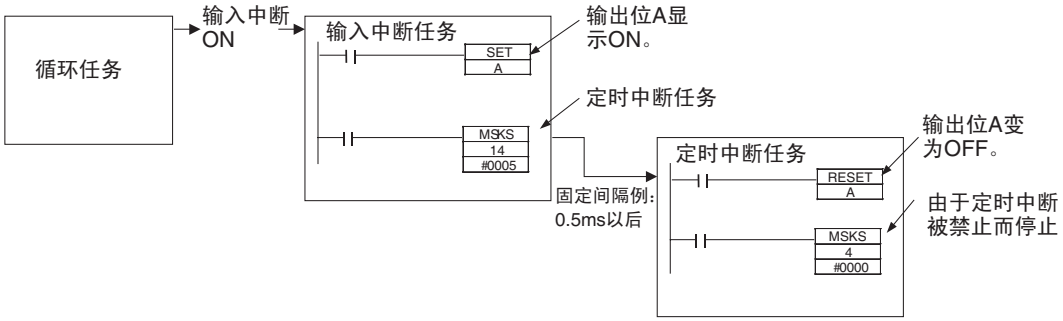
使用 CJ1M CPU 单元，下述功能允许采用定时中断作为高精度定时器。

- 定时中断定时器可以 0.1ms 为单位输入 （高精度间隔定时器）。
- 采用 MSKS （690）指令 （设定第一次中断时间）可以重新设定 （即：重新启动）
- 使用 MSKR（692）指令（读出间隔定时器 PV）可以读出间隔定时器 PV。

这些功能的应用以下面的示例说明。一个高精度的单步定时器，输入位变为 ON 作为一个信号，引起输出数变为 ON，随后间隔一定时间后变为 OFF。
例：

1,2,3...

1. 当内置输入位变为 ON 时，输入中断任务启动。
2. 在输入中断任务中输出位 A 变为 ON，执行 MSKS （690）指令使定时中断重置启动。
3. 一定时间的间隔后，定时中断任务启动，在定时中断任务中的输出位 A 变为 OFF，执行 MSKS （690）指令停止定时中断。



6-5-1 以 0.1ms 为单位设定定时中断

使用 PLC 设置的定时中断单元时间设定和 MSKS（690）指令，设定定时中断时间。

CJ1M CPU 单元中，在最小间隔时间 0.5ms 和最大间隔时间 999.9ms 之间可以以 0.1 ms 为单位设定定时中断时间。

PLC 设置

项目	PLC 地址		设定值	默认	刷新时间
	字	位			
定时中断单元时间设定	195	00 ～ 03	0 hex：10ms 单元 1 hex：1ms 单元 2 hex：0.1ms 单元 （仅限于 CJ1M CPU 单元）	0 hex	操作开始时

6-5-2 用 MSKS （690）定义复位启动

当使用 CJ1M CPU 单元和 MSKS（690）指令启动定时中断时，启动中断之前可以对内部定时器复位 （称为复位启动）。

可用这个方法规定第一次中断的时间，而不需使用 CLI （691）指令。

通过使用 MSKS （690）指令设定定时中断时间 （两个中断之间的间隔），启动定时中断。然而，执行 MSKS （690）指令后，只有规定了 CLI （691）指令，第一次定时中断任务启动 （第一次中断开始时间）之前所需的时间才是一定的。因此，CJ1M CPU 单元提供了一个内部定时器重新启动功能，允许设定第一次中断的时间，而不需使用 CLI （691）指令。

MSKS （690）指令操作数 （仅在定时中断指定时）

操作数	设定值
N (中断标识符)	4：定时中断 0，正常设定 （内部定时器未复位）
	5：定时中断 1，正常设定 （内部定时器未复位）
	14：定时中断 0，复位启动 （仅限于 CJ1MCPU 单元）
	15：定时中断 1，复位启动 （仅限于 CJ1MCPU 单元）

6-5-3 用 MSKR （692）读出内部定时器当前值 PV

CJ1M CPU 单元允许读出内部定时器的 PV，它是定时中断时间的量度。这个时间既可从定时中断开始点，也可从前一个中断点读出。通过执行 MSKR （692）指令读出内部定时器 PV。时间单位取决于在 PLC 设置中设定的定时中断单位时间 （与设定定时中断时间同样方法）。

MSKS （692）操作数 （仅在定时中断指定时）

操作数	设定值
N (中断标识符)	4：定时中断 0，读出定时中断时间 （设定值）
	5：定时中断 1，读出定时中断时间 （设定值）
	14：定时中断 0，读出内部定时器 PV （仅限于 CJ1MCPU 单元）
	15：定时中断 1，读出内部定时器 PV （仅限于 CJ1MCPU 单元）

6-6 启动设定和维护

本节描述以下一些与启动和维护有关的功能。

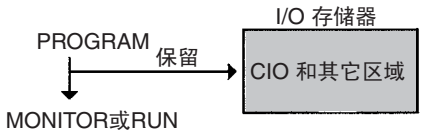
- 热启动 / 热停止功能
- 启动方式设定
- 电源 OFF 检测延迟设定
- 取消电源 OFF 中断
- RUN 输出
- 时钟
- 程序保护
- 远程编程和监控
- 闪存器
- 设定启动条件

6-6-1 热启动 / 热停止功能

操作方式改变

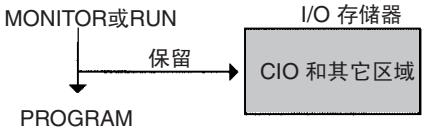
热启动

当 CPU 单元从 PROGRAM 方式转到 RUN/MONITOR 方式开始程序执行时，IOM 保持位 （A50012）变为 ON，保留在 I/O 存储器中的所有数据 * 。



热停止

当 CPU 单元从 RUN/MONITOR 方式转到 PROGRAM 方式停止程序执行时，IOM 保持位 （A50012）为 ON 时，在 I/O 存储器中的所有数据 * 仍被保留。



注 * 在方式改变 （PROGRAM↔RUN/MONITOR）期间， I/O 存储器的下述区域将被清除，除非 IOM 保持位为 ON：CIO 区域（I/O 区域，数据链接区域，CPU 总线单元区域，特殊 I/O 单元区域，内装板区域，SYSMAC BUS 区域，I/O 终端区域，设备网 （CompoBus/D）区域，和内部 I/O 区域），工作区域，定时器完成标志，和定时器 PV （ 只在 CS 系列 CPU 单元中支持：内装板区域，SYSMAC BUS 区域， I/O 终端区域）。

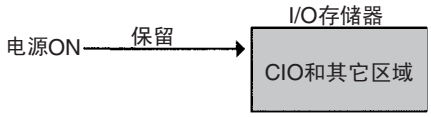
辅助区域标志和字

名称	地址	说明
IOM 保持位	A50012	在操作方式改变 （PROGRAM↔RUN/MONITOR）时，而这个位为 ON 时，将保留 I/O 存储器的所有内容。

在 IOM 保持位为 ON，程序执行停止时，将保留从输出单元的所有输 出。程序重新启动时，输出具有程序停止之前所具有的相同状态。
（当 IOM 保持位为 OFF 时，在输出被清除以后，将执行指令）。

PLC 电源 ON

当 PLC 接通电源（OFF → ON）时，为了保留 I/O 存储器中所有的数据 *，IOM 保持位必须为 ON 同时必须在 PLC 设置中 （地址 80，启动时 IOM 保持位状态）被保护。



辅助区域标志和字

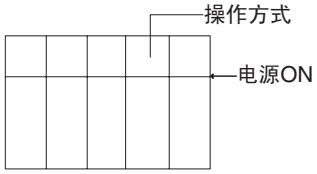
名称	地址	说明
IOM 保持位	A50012	在操作方式改变 （PROGRAM↔RUN/MONITOR）时，而这个位是 ON 时，将保留 I/O 存储器的所有内容。

PLC 设置

手持编程器地址	名称	地址	说明
80 位 15	启动时 IOM 保持位状态	0：电源接通时， IOM 保持位清 0 1：电源接通时，保留 IOM 保持位	0 (清除)

6-6-2 启动方式设定

CPU 单元起始操作方式 （当电源接通时）可以在 PLC 设置中设定。



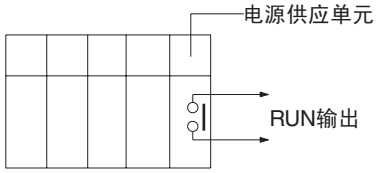
PLC 设置

手持编程器地址	名称	意义	地址	说明
81	启动方式	在启动时使用规定的操作方式	PRCN：手持编程器的方式开关 PRG：PROGRAM 方式 MON：MONITOR 方式 RUN：RUN 方式	PRCN：手持编程器的方式开关

注 如果启动方式是在 PRCN （手持编程器的方式开关）中设定，但手持编程器没有连接，CPU 单元将以 RUN 方式启动。当电源接通时，从默认值改变 PLC 设置，启动 MONITOR 方式或 PROGRAM 方式。（然而，在相同情况下， CS 系列 CS1 CPU 单元将以 PROGRAM 方式启动）。

6-6-3 RUN 输出

有些电源供应单元（C200HW-PA204R，C200HW-PA209R，CJ1W-PA205R）配备 RUN 输出。当 CPU 单元是在 RUN 方式或 MONITOR 方式下操作时，这个输出点是 ON （接通）；而当 CPU 单元是在 PROGRAM 方式下操作时，这个输出点是 OFF （断开）。



使用 RUN 输出可建立一个外部安全电路，如：一个急停电路，以防止外部电源提供电源给输出单元，除非 PLC 在运行。

注 使用的电源供应单元无 RUN 输出时，可通过对永远 ON 标志 （A1）（Always ON Flag(A1)）编程，从一个输出单元对一个输出点作为一个执行条件，从而建立 （与 RUN 输出）等效的输出。

! 注意 在 PLC 的电源供应之前，如果输出单元的外部电源接通，在 PLC 刚一接通的瞬时，输出单元将会出错。为了防止任何出错，加上一个外部电路，在电源供应到 PLC 之前，它能防止输出单元的外部电源接通。建立一个象上面所描述的失效保险电路，确保仅在 PLC 是在 RUN 方式或 MONITOR 方式下操作时，电源才通过外部电源供电。

6-6-4 电源 OFF 检测延迟设定

通常情况下，电源供应电压下降到最小额定电压 （DC 电源供应的 80%）的 85% 以后约 10 ～ 25ms 将检测为电源中断。这是在 PLC 设置（地址 225 位 0 ～ 7，电源 OFF 检测延迟设定）中的设定，它能增加这个时间在 10ms 以下（对于 DC 电源供应，小于 2ms）。
在电源 OFF 中断任务允许执行时，当确认电源断开时将执行电源 OFF 中断任务，否则 CPU 将复位并且停止操作。

相关设定

地址	名称	意义	设定	默认
CIO 256，位 00 ～ 07	电源 OFF 检测延迟	设定检测电源中断之前的延迟时间	00 ～ 0A (Hex)：0 ～ 10 ms	00(Hex) 0 ms

6-6-5 禁止电源 OFF 中断

仅有 CS1-H， CJ1-H， CJ1M，或 CS1D CPU 单元具有这个功能。

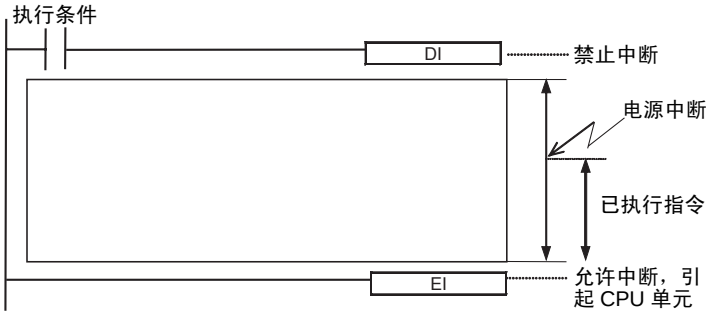
用 CS1-H，CJ1-H，CJ1M，或 CS1D CPU 单元，程序区域将免受电源 OFF 中断影响，因此即使在 CPU 单元电源供应中断前它们仍可以运行。这个功能是通过 DISABLE INTERRUPTS （DI （693） ）和 ENABLE INTERRUPTS （EI （694））指得到的。

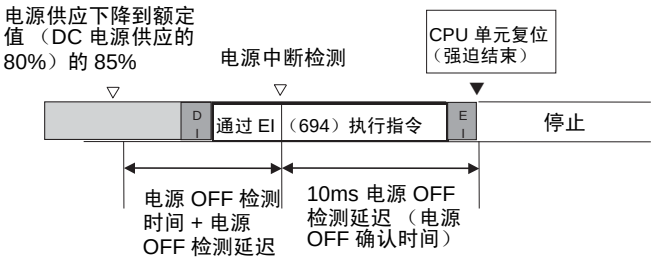
这个功能可以使用指令组 （如：这套指令必须成组执行），因此在下一次电源接通时，不会用中间存储的数据开始执行操作。

步骤

1,2,3...

1. 用 A530 ～ A5A5Hex 对电源 OFF 中断设定禁止，使禁止电源 OFF 中断功能有效。
2. 在 PLC 设置中使禁止电源 OFF 中断功能有效 （这是默认设定）。
3. 对被保护的程序部分使用 DI （693）禁止中断，在这个程序部分以后，使用 EI （694）使允许中断。在执行 DI （693）和 EI （694）之间的指令期间即使电源中断发生，在电源 OFF 中断之前完成 DI （693）和 EI （694）之间的所有指令。





相关设定

名称	地址	意义
电源 OFF 中断设定禁止	A530	在执行 EI （694）之前，允许使用 DI （693）使电源 OFF 中断处理禁止 （除执行电源 OFF 中断任务之外）。 A5A5 Hex：允许使用 DI （693）使电源 OFF 中断处理禁止。 任何其它值：禁止使用 DI （693）指令禁止 OFF 中断处理。

6-6-6 时钟功能

CS 系列 PLC 具有以下时钟功能：

- 发生电源中断时的时间监控
- PLC 接通时的时间监控
- PLC 接通总时间的监控

注 CS 系列 CS1 CPU 单元发货时没有装入备用电池，当安装电池后， CPU 单元的内部时钟将显示 00/01/01 00：00：00 或可能其它值。使用时钟功能，连接电池，接通电源，使用编程设备（手持编程器或 CX-Programmer）或使用 FINS 命令（07 02， CLOCK WRITE）设定时间和日期。一旦设定， CPU 单元的内部时钟就开始运行。

辅助区域标志和字

名称	地址	功能
时钟数据	A35100 ～ A35107	秒：00 ～ 59 (BCD)
	A35108 ～ A35115	分：00 ～ 59 (BCD)
	A35200 ～ A35207	小时：00 ～ 23 (BCD)
	A35208 ～ A35215	每月的日期：00 ～ 31 (BCD)
	A35300 ～ A35307	月：00 ～ 12 (BCD)
	A35308 ～ A35315	年：00 ～ 99 (BCD)
	A35400 ～ A35407	每周日期： 00：星期天， 01：星期一， 02：星期二，03：星期三， 04：星期四， 05：星期五， 06：星期六
启动时间	A510 和 A511	含有电源接通的时间
电源中断时间	A512 和 A513	含有电源最后一次中断的时间
总的电源接通时间	A523	含有 PLC 接通的总的时间 （以二进制）以 10 小时为单位。

相关指令

指令	名称	功能
SEC(065)	小时→秒	在小时 / 分 / 秒格式之间转换时间数据到一个仅以秒为单位的等效时间。
HMS(066)	秒→小时	将秒的数据转换到小时 / 分 / 秒格式的等效时间。
CADD(730)	日历加法	将时间添加到指定的日历数据字上。
CSUB(731)	日历减法	将时间从指定的日历数据字上减去。
DATE(735)	时钟调整	在指定的源字设定中改变内部时钟设定。

6-6-7 程序保护

CS/CJ 系列用户程序可以是写保护和完全保护 （读 / 写保护）。

使用 DIP 开关进行写保护

把 CPU 单元的 DIP 开关的引脚 1 置 ON，可以对用户程序写保护。当这个引脚为 ON 时，将不可能用编程工具 （包括手持编程器）改变用户程序。在工作场所，这个功能可以防止在不注意时程序被重写。

在写保护时，仍然可以对程序进行读和显示。

使用密码读 / 写保护

使用 CX-Progtammer 可以阻止对用户程序区域的访问。受保护的程序可防止未经许可的程序复制和知识产权的损失。使用编程工具为程序保护设定一个密码，这样，整个程序就不能进行存取。

注
1. 如果忘记密码，PLC 内的程序就不能传送到计算机中。将密码作一个记录，并保存在一个安全的地方。
2. 如果忘记密码，程序就不能从计算机传送到 PLC 中。如果没有输入密码，程序也不能从计算机传送到 PLC 中。

密码保护

1,2,3...

1. 在联机或未联机时用如下步骤注册一个密码：
   a) 选择 PLC, 从视图菜单 （View Menu）中选择*特性 （Properties）*。
   b) 从 PLC 特性对话框中选择*保护 （Protection）*，并输入密码。
2. 在联机时用如下步骤设定密码：
   a) 选择 *PLC*, 密码*保护 (Password Protection)* ，然后设定 （Set）。显示程序保护设定对话框。
   b) 点击 OK 按钮。

用户程序数据确认

通过检查 A090 ～ A097 的内容能确认 CS1-H，CJ1-H，CJ1M，或 CS1D CPU 单元中所建立的数据、程序和参数。

辅助区域字

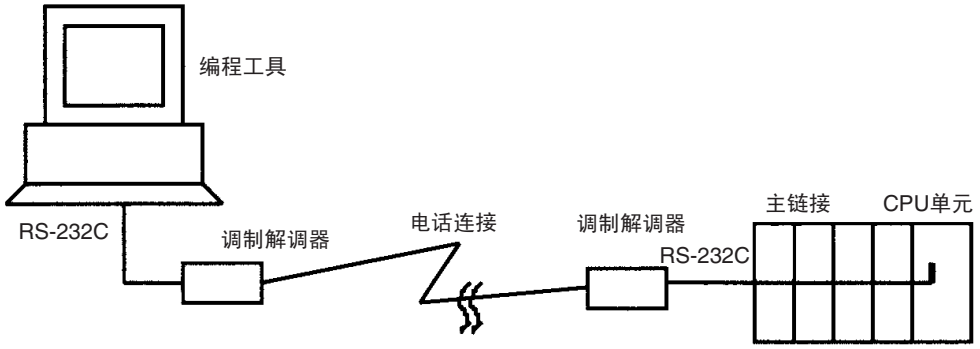
名称	地址	说明	
用户程序数据	A090 ～ A093	最新重写的用户程序时间和日期在存储器中以 BCD 给出。	
		A09000 ～ A09007	秒 (00 ～ 59 BCD)
		A09008 ～ A09015	分 (00 ～ 59 BCD)
		A09100 ～ A09107	小时 (00 ～ 23 BCD)
		A09108 ～ A09115	每月的日期 (01 ～ 31 BCD)
		A09200 ～ A09207	月 (01 ～ 12 BCD)
		A09208 ～ A09215	年 (00 ～ 99 BCD)
		A09300 ～ A09307	日期 (00 ～ 06 BCD) 每周日期： 00 ：星期天 01：星期一 02：星期二 03：星期三 04：星期四 05：星期五 06：星期六
参数数据	A094 ～ A097	最新重写的参数时间和日期在存储器中以 BCD 出中给。格式同上面给出的用户程序数据相同。	

6-6-8 远程编程和监控

通过调制解调器或控制器链接网络可以远程编程和监控 CS/CJ 系列 PLC。

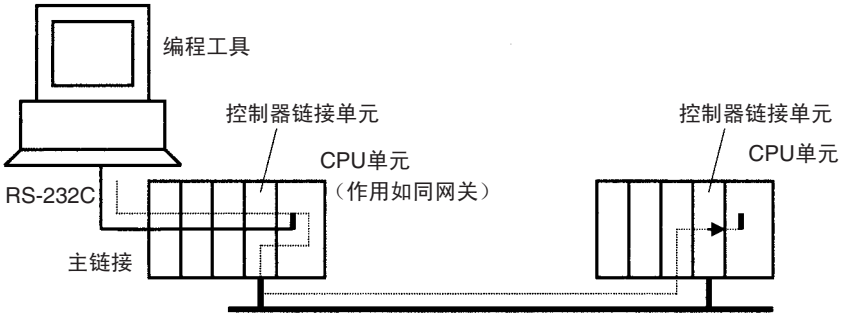
1,2,3... 1. 调制解调器连接

通过调制解调器可执行主链接功能，它允许一定距离的 PLC 进行操作监控，数据传送，甚至可以通过电话对远距离 PLC 程序进行联机编辑。所有编程工具的联机操作都可在这个连接中执行。



2. 控制器链接网络连接

控制器链接或以太网络中的 PLC 可通过主链接编程和监控。所有编程工具的联机操作都可在这个连接中执行。



6-6-9 单元简介

从 CX-Programer 中，可以读出有关 CS/CJ 系列单元的以下信息。

- 制造信息（批号，序号，等。）：当单元出现问题时，可以方便地向 OMRON 公司提供的信息。
- 单元信息（类型，方式号，正确的机架 / 槽位置）：得到安装信息的一种便捷的方法。
- 用户定义文本（最多 256 字）；可在存储卡中记录维护必须的信息（单元检查历史，制造线号，和其它应用信息）。

6-6-10 闪存器

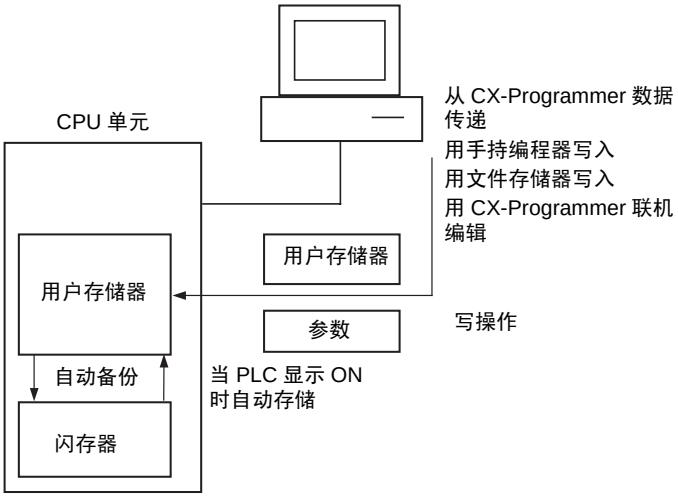
这个功能仅限于 CS1-H， CJ1-H， CJ1M，或 CS1D CPU 单元。

使用 CS1-H，CJ1-H，CJ1M，或 CS1D CPU 单元，闪存器中会自动备份任何时候在 CPU 单元中写入或更改的用户程序和参数。

下列数据会自动备份：用户程序，参数 （包括 PLC 设置，注册 I/O 表，路由表，和 CPU 总线单元数据，如数据链接表）。

无任何时在 CPU 单元中写入的用户程序或参数将自动备份数据，包括从 CX-Programer 数据传递操作，用手持式编程器写入数据，联机编辑，从存储卡或 EM 文件存储器数据传递等。

在启动时，写入闪存器的用户程序和参数数据自动传递到CPU 单元中的用户存储器中。



注 1. 数据写入到闪存器时，在 CPU 单元前面的 BKUP 指示器将闪亮。从编程设备或文件存储器传递数据，或执行联机编辑以后，在备份操作结束之前（即：在 BKUP 指示器熄灭之前），不要将 CPU 单元的电源切断。

2. 仅对联机编辑且 CPU 单元配有电池时，在上述条件下，即使电源在备份操作结束之前断开（如：在 BKUP 指示器亮着时）， CPU 单元将重新启动，尽管在 1 分钟内被要求启动 CPU 单元。在这种情况下（即使 CPU 单元中配有电池）总是确保在关闭电源之前完成备份操作，不然 CPU 单元将在无电源下工作一定时间。

备份数据所需的时间 （BKUP 指示器亮的时间）取决于用户程序的大小，如下表所示。

用户程序大小	备份处理时间		
	MONITOR 方式		PROGRAM 方式
	0.4 ms 循环时间 （例）	10.0 ms 循环时间 （例）	
10 千步	2 s	8 s	1 s
60 千步	11 s	42 s	6 s
250 千步	42 s	170 s	22 s

注 1. 当电源接到 CPU 单元时， BKUP 指示器点亮。

2. 备份数据可能需要不超过 1 分钟的时间，取决于执行的联机编辑类型。

! 注意 当用户程序和参数数据写入 CPU 单元中时， CS1-H， CJ1-H， CJ1M，和 CS1DCPU 单元会自动将他们备份到闪存器中。然而， I/O 存储器（包括 DM，EM，HR 区域）不写入闪存器。DM，EM，和 HR 区域在电源中断期间使用电池保持。如果电池出错，电源中断后这些区域的内容将不准确。如果 DM，EM，和 HR 区域的内容被用于控制外部输出，无任何时电池出错标志（A40204）变为 ON，都要防止不正确的输出。

注 当备份数据从 CX-Programer 传递操作而不是通常的数据传递时 （PLC/ 传递（*PLC/Transfer*）），通过 CX-Programer 备份状态将显示在存储器备份状态窗口。为了显示这个窗口，在 PLC 特性中检查“设定显示备份状态”对话框，从视图菜单中选择窗口。对于正常传递操作，在程序和其它数据传递状态之后，备份状态将显示在传递窗口中。

辅助区域标志

名称	地址	意义
闪存器出错标志	A40310	当闪存器失败时，变为 ON。

6-6-11 启动条件设定

这个功能仅限于 CS1-H， CJ1-H， CJ1M，或 CS1D CPU 单元。

电源接通后，有些单元和内装板要求很大的时间启动，影响了 CPU 单元的启动时间。可以设定 PLC 设置，这样 CPU 在启动时不初始化这些单元。

这个设定应用在 ITNC-EIS01-CST 和 ITNC-EIX01-CST 开放网络控制器 CS1 总线接口单元中。（目前没有内装板如 2001 年 10 月那样的应用）。

通过设定启动条件和内装板控制这个功能，设定在下表中说明。

启动条件	PLC 设置	
	启动条件 （手持编程器地址 83，位 15）	内装板设定 （手持编程器地址 84，位 15）
不等待所有的单元和内装板启动	1：允许不等待运行	1：不等待特定的内装板
不等待所有的单元（等待内装板）启动	1：允许不等待运行	0 ：启动前等待所有的内装板
启动前等待所有的单元和内装板	0 ：总是等待所有的单元 / 内装板	所有

注 CS1 CPU 单元中，直到所有的单元和内装板完成启动处理后， CPU 单元才会启动。

PLC 设置

手持编程器地址		名称	设定	默认	CPU 单元刷新时间
字	位				
83	15	启动条件	0：等待单元和内装板 1：不等待	0：等待	电源 ON
84	15	内装板设定	0：等待所有内装板 1：不等待特定的内装板	0：等待	电源 ON

启动条件

0：如果有一个或更多个特定内装板或单元还没有完成启动处理， CPU 单元将在 MONITOR 或 PROGRAM 方式中等待，等待所有的单元和内装板完成启动处理。

1：即使有一个或更多个特定内装板或单元还没有完成启动处理， CPU 单元将在 MONITOR 或 PROGRAM 方式中继续处理并启动。然而，内装板的操作也取决于以下设定。

内装板设定

这个设定仅用于当启动条件设定在1时，不等待特定的单元和内部板允许启动。如果启动条件设定在 0 时，忽略这个设定。

0：如果有一个或更多个特定内装板还没有完成启动处理， CPU 单元将在 MONITOR 或 PROGRAM 方式中等待，等待所有的内装板完成启动处理。

1：即使有一个或更多个特定内装板还没有完成启动处理， CPU 单元将在 MONITOR 或 PROGRAM 方式中继续处理并启动。

6-7 诊断功能

本节对以下的诊断和调试功能给出简要的概述。

- 错误记录
- 输出断开功能
- 故障报警功能 (FAL(006) 和 FALS(007))
- 故障点检测 (FPD(269)) 功能

6-7-1 错误记录

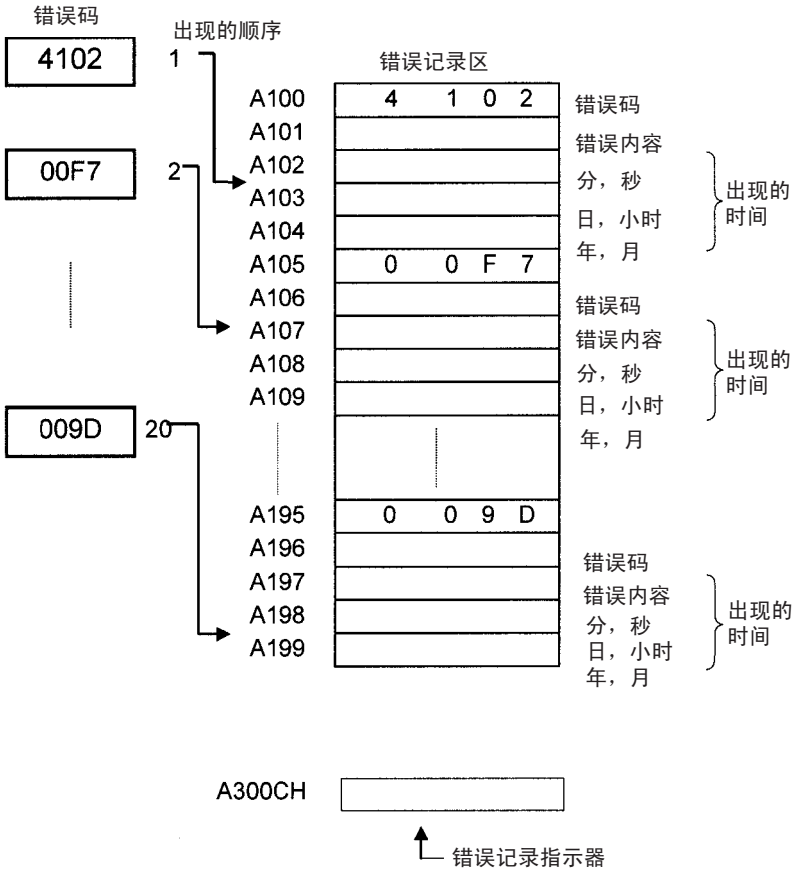
在CS/CJ 系列PLC中每出现一个错误，CPU单元在错误记录区中存储一条错误信息。出误信息包括错误码（存储在 A400），出错内容和出错误时间。在错误记录中可以最多可存储 20 个记录。

除系统产生的错误外， PLC 记录用户定义的 FAL （006）和 FALS （007）错误，这使系统工作状态的跟踪变得容易。

详细内容请参考 *CS/CJ 系列操作手册*有关故障分析章节。

注 程序中执行 FAL（006）或 FALS（007）时将产生用户定义错误。这些指令的执行条件构成了用户定义错误条件。FAL（006）产生一个非致命错误，而 FALS（007）产生一个停止程序执行的致命错误。

当出错超过 20 次时，最早存储的错误数据 （在 A100 ～ A104）将被删除，余下的 19 条记录向下移一条记录位置，最新的记录被存储在 A195 ～ A199。



记录的数目以二进制存储在错误记录指示器中 （A300）。当错误超过 20 后，指示器不会递增。

6-7-2 输出断开功能

当错误发生时，作为紧急措施，通过使输出断开位 （A50015）接通 （ON），输出单元的所有输出将断开（OFF）。操作方式仍将保持在 RUN 或 MONITOR 方式，但所有输出将断开。

注 通常 （当 IOM 保持位 =OFF），当操作方式从 RUN/MONITOR 方式转变到 PROGRAM 方式时，输出单元的所有输出将断开。可以使用输出断开位来断开所有输出，而不用转变到 PROGRAM 方式和停止执行程序。

设备网络的应用注意 CS1W-DRM21 或 CJ1W-DRM21 使用主控功能时，所有从属输出将断开（OFF）。当使用从属功能时，所有到主控的输入都将断开。然而，使用 C200HW-DRM21-V1，从属输出不会断开。

6-7-3 故障报警功能

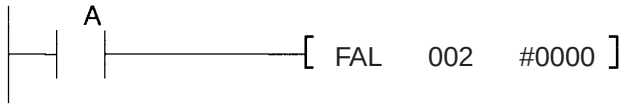
FAL （006）和 FALS （007）指令产生用户定义错误。FAL （006）产生一个非致命错误，而 FALS （007）产生一个停止程序执行的致命错误。

当满足用户定义错误条件（FAL （006）或 FALS （007）执行条件）时，故障报警指令将执行，同时进行以下处理。

1,2,3...

1. FAL 错误标志 （A40215）或 FALS 错误标志 （A40106）变为 ON。
2. 相应的错误码写入 A400。
3. 错误码和发生的时间存储在错误记录中。
4. 位于 CPU 单元前面的错误指示器将闪亮或点亮。
5. 如果 FAL （006）执行完毕， CPU 单元将继续操作。
   如果 FALS （007）执行完毕， CPU 单元将停止操作。（程序执行停止）。

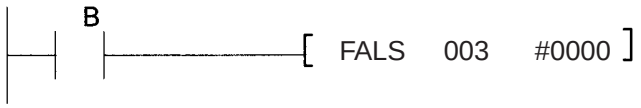
FAL(006) 的操作



当执行条件 A 变为 ON，产生 2 号 FAL 错误， A40215 （FAL 错误标志）变为 ON， A36002 （FAL 2 号标志）变为 ON。程序执行继续。

通过执行 FAL 号 00 的 FAL （006）或从编程工具 （包括手持编程器）中执行错误读 / 清除操作，可以清除 FAL （006）产生的错误。

FALS(007) 的操作



当执行条件 B 变为 ON，产生 3 号 FALS 错误，A40106（FALS 错误标志）变为 ON。程序执行停止。

通过消除错误的原因和从编程工具 （包括手持编程器）中执行错误读 / 清除操作，可以清除 FALS （007）产生的错误。

6-7-4 故障点检测

FPD （269）进行时间监控和逻辑诊断。在一个特定的监控时间内，如果诊断输出不变为 ON，时间监控功能产生一个非致命错误。逻辑诊断功能指明哪个输入可防止诊断输出变为 ON。

<u>时间监控功能</u>

在一个特定的监控时间内，如果诊断输出不变为 ON，执行 FPD （269）并使进位标志为 ON，起动计时。进位标志对一个错误处理块作为执行条件编程。FPD （269）也可用于产生一个所需 FAL 号的非致命 FAL 错误编程。

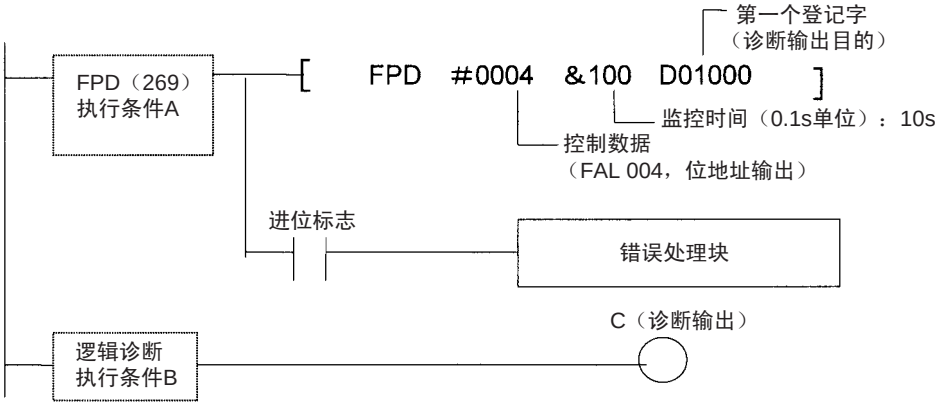
产生一个 FAL 错误时，当前信息将被记录并在编程设备上显示。可设定 FPD （269）在信息之前输出逻辑诊断的结果 （防止诊断输出变为 ON 的位地址）。

对诊断输出变为 ON 和设定监控时间，可使用教学功能自动决定实际所需的时间。

逻辑诊断功能

FPD （269）决定是那个输入位引起诊断输出保持 OFF，并且输出那个位的地址。输出可以设定到位地址输出 （PLC 存储器地址）或信息输出 （ASCII）。

- 如果选定位地址输出，位的 PLC 存储器地址可传送到一个变址寄存器中，在以后的处理中可以间接寻址变址寄存器。
- 如果选定信息输出，位地址将以一个可在编程工具中显示的ASCII信息登记记录。



时间监控：

输入 A 后 10 秒不管输出 C 是否变为 ON，监控运行。如果 C 在 10 秒内没有变为 ON，将检测出故障，进位标志显示 ON。进位标志执行错误处理块程序。同时产生 004 号 FAL 的 FAL 错误 （非致命）。

逻辑诊断：

FPD （269）决定在块 B 中是那个输出位防止输出 C 继续 ON。那个位地址输出到 D01000 和 D01001。

辅助区域标志和字

名称	地址	操作
错误码	A400	出现一个错误时，它的错误码存储在 A400 。
FAL 错误标志	A40215	执行 FAL （006）时，为 ON。
FALS 错误标志	A40106	执行 FALS （007）时，为 ON。
执行 FAL 号标志	A360 ～ A391	当一个 FAL （006）或 FALS （007）错误发生时，相应标志变为 ON。
错误记录区域	A100 ～ A199	错误记录区域包含有最近发生的 20 个错误信息。
错误记录指示器	A300	作为从错误记录区域 （A100）开始点的一个偏置，当一个错误发生时，错误记录指针递增 1，指出下一个错误记录将记录的位置。
错误记录指示器复位位	A50014	使这位为 ON，复位错误记录指示器 (A300) ～ 00。
FPD 教学位	A59800	执行 FPD （269）时。当你希望自动设定监控时间时，使这个位变为 ON。

6-7-5 模拟系统错误

这个功能仅限于 CS1-H， CJ1-H， CJ1M，或 CS1D CPU 单元。

可使用 FAL（006）和 FALS（007）有意地产生致命或非致命系统错误。系统调试时，可使用这个方法在可编程终端 （PT）或其它操作接口上测试显示信息。

使用以下步骤：

1,2,3...

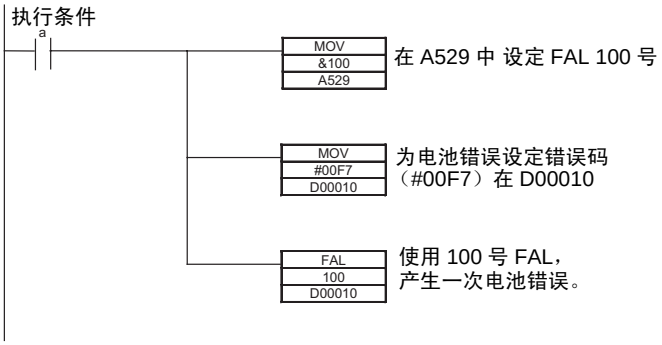
1. 在 A529 将 FAL 或 FALS 号设定为模拟使用。（当 FAL （006）和 FALS （007）模拟错误时，使用 A529）。
2. 将 FAL 或 FALS 号设定为模拟使用，作为 FAL （006）或 FALS （007）的第一个操作数。
3. 将错误码和错误设定为模拟，作为 FAL （006）或 FALS （007）的第二个操作数 （二个字）。指定 FAL （006）为非致命错误，指定 FALS （007）为致命错误。

为了模拟多个系统错误，如上所述多次使用 FAL（006）或 FAL（007）指令。

辅助区域标志和字

名称	地址	操作
用于系统错误模拟的 FAL/FALS	A529	设定一个虚 FAL/FALS 号，用于模拟系统错误。 0001 ～ 01FF Hex：FAL/FALS 号 1 ～ 511 0000 或 0200 ～ FFFF Hex：无 FAL/FALS 号用于系统模拟错误。

电池错误示例



注 使用同实际系统错误的相同的方法清除模拟系统错误。详细内容请参考 *CS 系列操作手册或 CJ 系列操作手册*。使用 FAL （006）和 FALS （007）模拟的所有系统错误都可通过再次启动电源清除。

6-7-6 禁止用户定义 FAL 错误的错误记录存储

这个功能仅限于 CS1-H， CJ1-H， CJ1M，或 CS1D CPU 单元。

PLC 设置提供了一个设定，它可防止用 FAL （006）和 FPD （269）时间监控产生的用户定义 FAL 错误记录到错误记录（A100 ～ A199）中。即使使用了这个设定， FAL 错误仍将产生，同时输出以下信息：A40215（FAL 错误标志），A360 ～ A391 （执行 FAL 号），和 A400 （错误码）。

仅当系统 FAL 出错需要存储在错误记录时使用这个功能，如：当程序使用 FAL （006）时产生许多用户定义出错，这些错误将很快充满错误记录。

PLC 设置

手持编程器地址		名称	设定	默认	CPU 单元刷新时间
字	位				
129	15	设定用户 FAL 存储	0：在出错记录中记录用户定义的 FAL 出错。 1：在出错记录中不记录用户定义的 FAL 出错	0：记录	无任何时都执行 FAL（006）（每个循环）

注 即使使用了以上设定来防止用户定义 FAL 出错被记录，以下项目还将存储在出错记录中。

- 用户定义致命出错 (FALS(007))
- 系统非致命出错
- 系统致命出错
- 用户模拟系统非致命出错 (FAL(006))
- 用户模拟系统致命出错 (FALS(007))

6-8 CPU 处理方式

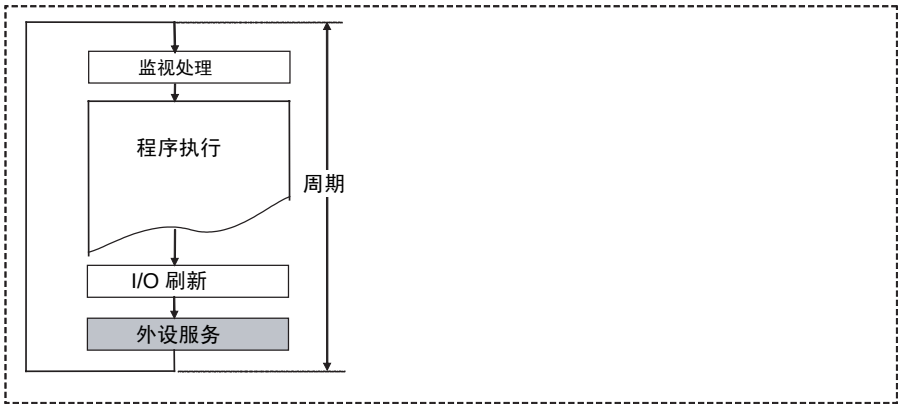
6-8-1 CPU 处理方式

通常情况下，外设服务 （见注）在每次循环结束时 （在 I/O 刷新以后）执行一次，每次服务时间可以是周期的 4% 或用户定义的时间。这样服务外设的时间不可能比循环时间快，而且由于外设服务时间的需要而使循环时间增加。

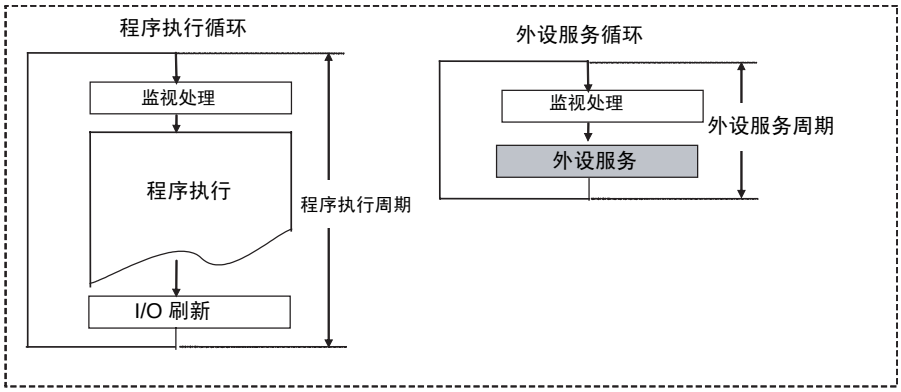
然而， CS1-H 或 CJ1-H CPU 单元支持并行处理方式，使程序执行过程与外设服务并行执行。特别当需要延长外设服务时，这些方式加快外设服务和缩短循环时间。（CJ1M 和 CS1D CPU 单元不具备并行处理方式）。

注 外设服务包括在外部设备要求下的不定期服务，如：对于特殊 I/O 单元， CPU 总线单元，和内装板 （仅限于 CS 系列）的事件服务 （例 ：FINS 命令的通信），以及对外设和 RS-232C 端口的通信端口服务 （但不包括用于 CPU 总线单元的数据链接和其它特殊 I/O 刷新）。

一般方式



并行处理方式



并行处理方式

有两种不同的并行处理方式：同步存储器访问并行处理和异步存储器访问并行处理。

■ 异步存储器访问并行处理

这个方式中，外设服务 I/O 存储器访问与程序执行的 I/O 存储器访问是不同步的。换言之，所有外设服务与程序执行并行执行，包括存储器访问。当外设服务量很重时，这个方式对于程序执行和事件处理执行是最快的（与其它方式比较）。

■ 同步存储器访问并行处理

这个方式中，外设服务 I/O 存储器访问与程序执行不是并行执行的，而是在程序执行以后执行（就如同是在一般执行方式下），例如在 I/O 刷新以后执行。所有其它外设服务与程序执行并行执行。

这个方式相比于一般执行方式下的程序执行和事件处理都要快。由于对外设服务刷新 I/O 另需时间，因此程序执行周期将大于异步存储器访问并行处理。

一般处理，异步存储器访问并行处理和同步存储器访问并行处理的扫描周期和外设服务响应列于下表中。（这些值适用于：由基本指令组成的周期为 10ms, 带有一个以太网的程序。这些值仅供参考，它们会随着系统的不同而变化）。

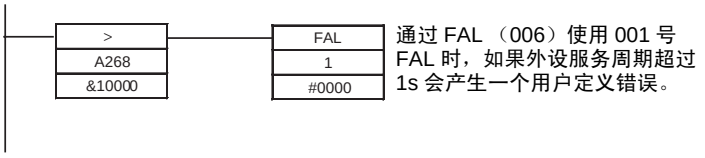
项目	一般方式	异步存储器访问并行处理	同步存储器访问并行处理
周期	任务设定为 1	0.9	0.9
外设服务	任务设定为 1	0.4	1.0

注 1. 外设服务包括用于特殊 I/O 单元，CPU 总线单元，和内装板（仅限于 CS 系列）的事件服务（例：FINS 命令的通信），以及对外设和 RS-232C 端口的通信端口服务 （但不包括用于 CPU 总线单元的数据链接和其它特殊 I/O 刷新）。

2. 版本 1 或更高的 CS1 CPU 单元和 CS1-H 或 CJ1-H CPU 单元也支持外设服务优先方式 （在一个固定的程序执行扫描周期执行外设服务）。它比一般处理方式提供更快的外设服务，但是程序执行将变慢。然而，事件响应将没有并行处理方式快。因此在处理时，任何当赋予事件响应优先权时，应该使用异步存储器访问并行处理。

3. 当使用并行处理时，CPU 单元中会发生外设服务周期超时出错（如在下面 a）和 b) 所描述的）。如果发生这个出错，编程工具中的显示将指出周期时间太长， A40515（外设服务周期超时）将变为 ON，操作将停止（致命错误）。

   a) 如果外设服务周期超过 2.0s，将发生一个周期超时错误。可以在 A268 中监控外设服务周期，在发生之前检测出可能的出错。例如，使用 001 号 FAL 时，如果外设服务周期超过 1s （即：如果 A268 的内容超过 2710Hex （10000 十进制））将会产生一个用户定义错误。



A268
&10000
FAL
1
#0000
通过 FAL （006）使用 001 号 FAL 时，如果外设服务周期超过 1s 会产生一个用户定义错误。

   b) 如果指令执行周期 （即，指令执行时间）处理时间太短时，也将发生外设服务周期超时出错。在一般执行方式下，这个时间存储在 A266 和 A267 中。作为一条准则：如果指令执行时间为 2ms 或更短，将发生外设服务周期超时出错，这时不能使用并行处理方式。当仅调试部分程序时（它将引起非常短的指令执行时间），使用一般方式来防止发生这种出错。

当用户的应用程序在并行处理方式下运行时，应该断开手持编程器。手持式编程器将被分配服务时间，从而增加对手持编程器按键的响应，这将增加外设服务时间和降低并行处理的有效性。

PLC 设置

PLC 设置中规定处理方式。

手持编程器地址		名称	设定	默认	CPU 单元刷新时间
字	位				
219	08 ～ 15	CPU 处理方式	00 Hex：一般方式 01 Hex：同步存储器访问并行处理 02 Hex：异步存储器访问并行处理 05 ～ FF Hex：对外设服务优先权方式下，时间片程序执行时间（以 1ms 递增， 5 ～ 255ms）。 0 3 和 04 Hex 的设定无定义（非法），将引起 PLC 设置出错 （非致命）。	00 Hex：一般方式	开始操作

辅助区域标志和字

名称	地址	操作
外设服务周期超时	A40515	外设服务周期超时 2 s 时变为 ON。停止操作。
外设服务周期	A268	当使用并行处理方式中的一种 （同步或异步存储器访问）并且 PLC 是在 RUN 或 MONITOR 方式下，将保留外设服务周期。这个时间以二进制表示，从 0.0 ～ 20000.0 （以 0.1ms 递增）
指令执行时间 （程序执行所有时间片时间和外设服务所有时间片时间的总和）	A266 和 A267	一般方式下，仅包括指令执行时间。这个时间以 32 位二进制值存储。 00000000 ～ FFFFFFFF Hex （单位：0.1ms）（0 ～ 429,496,729.5 ms） A266 ：最低位字 A267 ：最高位字

异步存储器访问并行处理

程序执行

监视		I/O 总线检查和其它处理 0.3 ms
指令执行时间		所有指令执行时间之和
最小周期计算		最小程序执行周期的处理时间
循环服务	I/O 刷新	每个 I/O 单元刷新时间× 单元数
	CPU 总线单元特殊 I/O 刷新	每个特殊 I/O 单元刷新时间× 单元数
外设服务	文件访问	在 PLC 设置中设定的外设服务时间 （默认：4% 的周期时间）

外设服务

监视		电池检查，用户程序存储器检查，等。 0.2 ms
外设服务	特殊 I/O 单元的事件服务	包括访问 I/O 存储器的事件服务（见注） 每个服务时间最多 1s
	CPU 总线单元的事件服务	
	外设端口服务	
	RS-232C 端口服务	
	内部板的事件服务 （仅限于 CS 系列）	
	使用的 （包括后台执行）通信端口（内部逻辑端口）的事件服务	

注 访问 I/O 存储器的事件服务包括 1）服务接受到的任何访问 I/0 存储器的 FINS 命令 （带有从 01Hex 开始的通用码的 I/O 存储器读 / 写命令或带有 23 Hex 开始的通用码的强迫置位 / 复位命令）和 2）服务接受到的任何访问 I/0 存储器的 C-mode 命令 （不包括使用外设或 RS-232C 端口的 NT 链接）

同步存储器访问并行处理

程序执行

监视		I/O 总线检查和其它处理 0.3 ms
指令执行时间		所有指令执行时间之和
最小周期计算		最小程序执行周期的处理时间
循环服务	I/O 刷新	每个 I/O 单元刷新时间× 单元数
	CPU 总线单元特殊 I/O 刷新	每个特殊 I/O 单元刷新时间× 单元数
外设服务	文件访问	在 PLC 设置中设定的外设服务时间 （默认：4% 的周期时间）
	事件服务要求 I/O 存储器访问 （见注）	

外设服务

监视		电池检查，用户程序存储器检查，等。 0.2 ms
外设服务	特殊 I/O 单元的事件服务	包括访问 I/O 存储器的事件服务（见注） 每个服务时间最多 1s
	CPU 总线单元的事件服务	
	外设端口服务	
	RS-232C 端口服务	
	内部板的事件服务 （仅限于 CS 系列）	
	使用的 （包括后台执行）通信端口（内部逻辑端口）的事件服务	

注 访问 I/O 存储器的事件服务包括 1）服务接受到的任何访问 I/0 存储器的 FINS 命令 （带有从 01Hex 开始的通用码的 I/O 存储器读 / 写命令或带有 23 Hex 开始的通用码的强迫置位 / 复位命令）和 2）服务接受到的任何访问 I/0 存储器的 C-mode 命令 （不包括使用外设或 RS-232C 端口的 NT 链接）

6-8-2 并行处理方式和最小周期

在使用并行处理方式时，如果规定了最小周期，在程序执行后将插入一段等待时间直至达到最小周期要求的时间，但外设服务将继续。

6-8-3 异步存储器访问的并行处理中的数据同时性

使用异步存储器访问的并行处理时，在以下情况下，数据可能不并行处理。

- 使用通信命令从 I/O 存储器中读取超过一个字时，这些字中包含的数据可能不是同一时间点的数据。
- 在指令执行期间，如果一条指令读取超过内存中的一个 I/O，同时执行外设服务，这些字中包含的数据可能不是同一时间点的数据。
- 在 I/O 存储器中如果相同的字被超过一条指令（这些指令在一个程序中的不同位置）读取，并且在这指令执行之间执行外设服务，那么这个字中所含有的数据可能不一致。

在需要时，可以使用以下步骤来确保数据的同时性。

1. 使用同步存储器访问的并行处理。
2. 在程序需要的地方使用 IOSP （287）禁止外设服务，然后再次使用 IORS （288）允许外设服务。

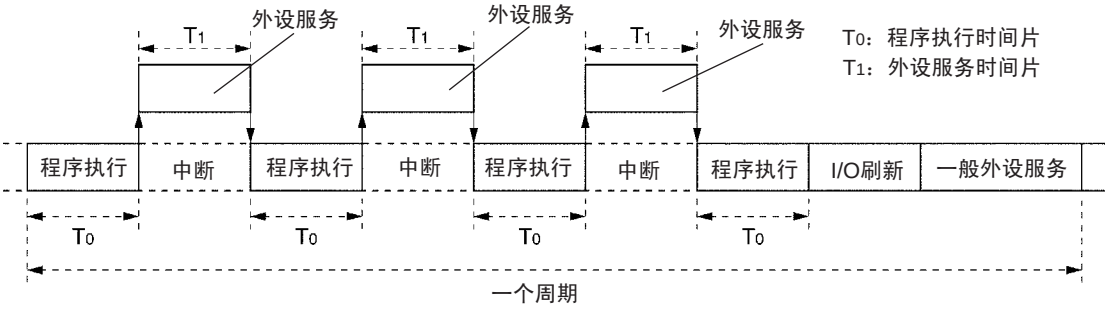
6-9 外设服务优先方式

通常情况下，RS-232C 端口，外设端口，内装板（仅限于 CS 系列），CPU 总线单元，和特殊 I/O 单元的外设服务仅在 I/O 刷新以后，周期结束时服务一次。周期时间的 4% 或用户定义的时间分配给每个服务。然而，在一个周期内，有一个方式可用来允许定期的服务。这个方式 （称为外设服务优先方式）是在 PLC 设置中设定的。

注 外设服务优先方式可在 CJ 系列 CPU 单元或 CS 系列 CPU 单元中使用，但 CS 系列 CS1 CPU 单元必须有批号 001201 □□□□或以后产品 （制造日期为 2000 年 12 月 1 日，或以后）。（CS1D 不支持外设服务优先方式）。

6-9-1 外设服务优先方式

如果设定外设服务优先方式，程序执行在一规定的时间会被中断，执行规定的服务，随后程序再恢复执行。在整个程序执行中不断重复这个过程。正常外设服务仍将在 I/O 刷新以后执行。



外设服务优先方式可被用于与一般外设服务一起对指定的端口或单元执行定期的服务。这就使得有些应用程序成为可能，这些应用程序要求给予外设服务高于程序执行的优先权，如要求快速响应主监控机的过程控制应用程序。

- 最多可以指定5个单元或端口具有优先服务权。CPU总线单元和CS/CJ特殊 I/O 单元通过它们的单元号来指定。
- 在外设服务的每个时间片中只能执行一个单元或端口。在指定时间期限前如果服务已完成，程序执行立即继续，下一个单元或端口直到外设服务的下一个时间片才被服务。然而，在同一周期内相同的单元或端口有可能被服务多次。
- 单元或端口的被服务顺序是根据 CPU 单元检测它们的顺序而定。

注 1. 尽管下列指令用于通信端口，即使使用外设服务优先权，它们将在执行周期中只执行一次：
RXD(235) (RECEIVE)
TXD(236) (TRANSMIT)

2. 使用外设服务优先权时，如果通过通信命令读出超过一个字，不能保证读出的数据同时发生。

3. 使用外设服务优先权时，CPU 单元可能超过最大周期时间。最大周期时间是在 PLC 设置中设定，如同周期时间监视的设定。如果周期时间超过设定的周期时间监视，周期时间太长标志（A40108）将变为 ON，PLC 操作将停止。如果使用外设服务优先权时，在 A264 和 A265 中的当前周期时间必须监控，按照要求调整周期时间监视（地址：+209）。（设定范围是 10 ～ 40,000ms, 递增量 10ms，默认值 1s）。

PLC 设置设定

为使用外设服务优先方式，在 PLC 设置中必须做出以下设定。

- 程序执行时间片：5 ～ 255ms，以 1ms 递增。
- 外设服务时间片：1.0 ～ 25.5ms，以 0.1ms 递增。
- 优先服务的单元和 / 或端口：CPU 总线单元 （根据单元号）
  CS/CJ特殊I/O单元（根据单元号）
  内装板 （仅限于 CS 系列）
  RS-232C 端口
  外设端口

手持编程器中的地址		设定	默认	功能	新设定的效果
字	位				
219	08 ～ 15	00 05 ～ FF (Hex)	00	00：禁止优先方式服务 05 ～ FF：指令执行的时间片 (5 ～ 255 ms ，以 1ms 递增)	操作开始时取得任务有效 （操作期间不能更改）
	00 ～ 07	00 ～ FF (Hex)	00	00：禁止优先方式服务 01 ～ FF：外设服务的时间片 (0.1 ～ 25.5 ms，以 0.1ms 递增)	
220	08 ～ 15	00 10 ～ 1F 20 ～ 2F E1 FC FD (Hex)	00	00：禁止优先方式服务 10 ～ 1F：CPU 总线单元单元号 + 10 (Hex) 20 ～ 7F：CS/CJ 特殊 I/O 单元单元号 + 20 (Hex) E1：内装板 FC：RS-232C 端口 FD：外设端口	
	00 ～ 07		00		
221	08 ～ 15		00		
	00 ～ 07		00		
222	08 ～ 15		00		

- 根据 PLC 设置的设定，操作和出错如下表所示。
- 不能用 CX-Programmer 设定。

条件			CPU 单元运行	PLC 设置出错
外设服务时间片	指令执行时间片	指定的单元和端口		
01 ～ FF：(0.1 ～ 25.5 ms)	05 ～ FF：(5 ～ 255 ms)	设定全部正确	外设服务优先方式	无
		00 和正确设定		
		正确，但冗余设定		
		一些非法设定	正确设定的项目外设服务优先方式	产生
		全部 00 设定	正常操作	产生
		00 和非法设定		
		所有非法设定		
00	00	---	正常操作	无
任何其它		---	正常操作	产生

注 如果在 PLC 设置中检测出一个出错， A40210 将变为 ON，同时产生一个非致命出错。

辅助区域信息

如果为程序执行和外设服务设定了时间片，所有程序执行和外设服务片时间的总和将存储在 A266 和 A267。对时间片做出合适的调整时，可参考这个信息。不使用外设服务优先权时，将存储程序执行时间。将使用这个值决定合适的时间片设定。

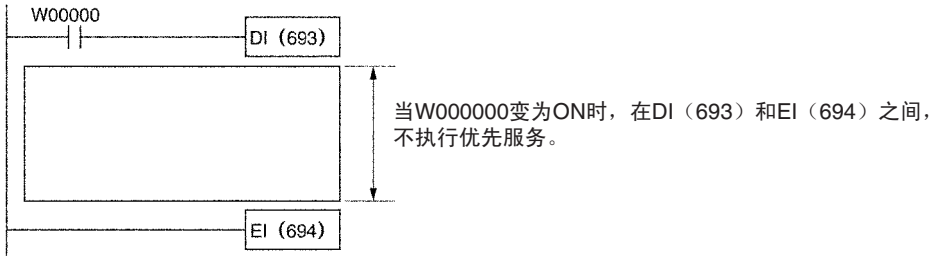
字	内容	意义	刷新
A266 和 A267	00000000 ～ FFFFFFFF Hex (0～4294967295十进制)	所有程序执行和外设服务片时间的总和。 0.0～429，496，729.5ms (0.1ms递增) A267（最高位字节） A266（最低位字节） 用32位二进制值（8数字十六进制）存储值	每个循环刷新内容，在操作开始时清除。

6-9-2 临时禁止优先方式服务

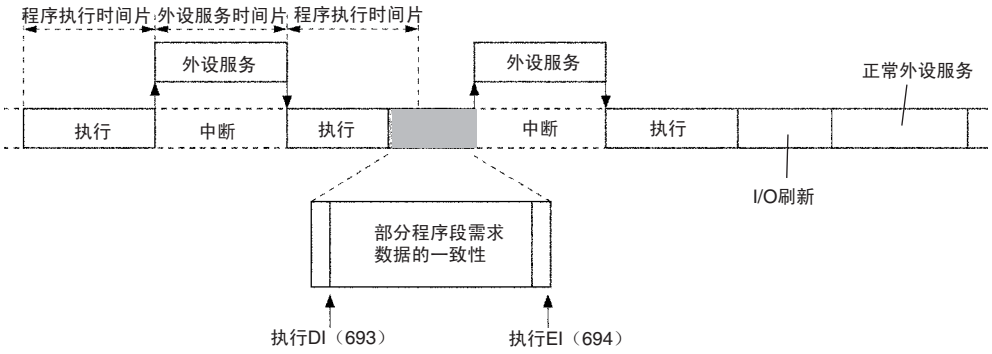
如果使用外设服务优先方式，在下述时间内不能保证数据同时性。

- 使用通信命令时，从外设服务读出一个以上字。数据可能在不同外设服务时间片期间读出，引起数据不一致。
- 在程序中使用了有很长执行时间的指令，如 .，当传递大量 I/O 存储器数据时。由于外设服务传递操作可能中断，引起数据不一致。在下面情况下数据可能不一致：程序正在写的字但在写完之前从外设读取数据，或程序正在读的字但在读完之前从外设写入数据。
- 在存储器中两条指令访问相同字。如果这些字是在执行这两条指令之间从外设写入，那么这两条指令将从存储器中读出不同的数值中断。

必须确保数据同时性时，对于 CS1 或 CJ1 CPU 单元使用禁止和允许中断指令（DI（693）和 EI（694）），在程序所要求的部分防止优先服务，如下例所示。对于 CS1-H，CJ1-H，或 CJ1M CPU 单元，可使用禁止外设服务和允许外设服务 指令 （IOSP （287）和 IORS （288））。



操作



注 1. DI（693）和 IOSP（287）不仅禁止优先服务中断，也包含其它所有中断，包括 I/O，定时和外部中断。执行循环任务后 （执行 END （001）以后）除非首先执行 CLI（691）来清除这些中断，否则所有产生的中断将被执行。

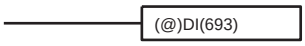
2. 在执行 EI （694）或 IORS （288）， END （100）之前，或 PLC 操作停止之前，用 DI （693）或 IOSP （287）禁止中断是有效的。这样就不能建立一个超过任务或周期结束的程序片段。当在一个以上的循环或任务中必须禁止中断，在每个循环任务中使用 DI （693）和 EI （694）或 IOSP （287）和 IORS （288）指令。

CS1 和 CJ1 CPU 单元

DI(693)

执行 DI （693）时禁止所有中断 （除了电源中断任务的中断外），包括优先服务中断、 I/O 中断、定时中断和外部中断。当它们已经被禁止时，如果执行 DI （693）指令，中断仍然保持禁止状态。

符号



适用的程序区域

区域	应用性
块程序区域	可以
步程序区域	可以
子程序	可以
中断任务	不可以

条件标志

标志	标记	操作
出错标志	ER	如果在一个中断任务中执行 DI （693），标志变为 ON，否则为 OFF 状态。

EI(694)

EI （694）执行时允许所有中断 （除了电源中断任务的中断外），包括优先服务中断、 I/O 中断、定时中断和外部中断。当它们已经使能时，如果执行 EI （694）指令，中断仍然保持使能状态。

符号

EI(694)不要求执行条件

适用的程序区域

区域	应用性
块程序区域	可以
步程序区域	可以
子程序	可以
中断任务	不可以

条件标志

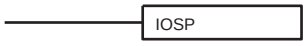
标志	标号	操作
出错标志	ER	如果在一个中断任务中执行 EI （694），变为 ON

CS1-H， CJ1-H，和 CJ1M CPU 单元

IOSP(287)

执行 IOSP （287）时，禁止外设服务。当它们已经被禁止时，如果执行 IOSP （287，）外设服务仍然保持禁止状态。

符号



适用的程序区域

区域	应用性
块程序区域	可以
步程序区域	可以
子程序	可以
中断任务	不可以

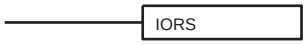
条件标志

标志	标号	操作
出错标志	ER	如果在一个中断任务中执行 IOSP （287），变为 ON，否则为 OFF 状态。

IORS(288)

执行 IOSP （288）时，允许被 IOSP （287）禁止的外设服务。当它们已经使能许时，如果执行 IORS （288）指令，外设服务仍然保持使能状态。

符号



适用的程序区域

区域	应用性
块程序区域	可以
步程序区域	可以
子程序	可以
中断任务	不可以

条件标志

标志	标号	操作
出错标志	ER	如果在一个中断任务中执行 IORS （288），变为 ON

6-10 无电池

CS 系列和 CJ 系列 PLC 可以在无电池 （或一个已用完的电池）情况下操作。无电池操作的使用过程取决于以下因素。

- CPU 单元
- 不管 I/O 存储器 （例：CIO 区域）是否被保持
- 不管 DM 和 EM 区域在启动时是否被初始化
- 不管 DM 和 EM 区域是否是在用户程序中被初始化

以上区别的归类如下表所示。

CPU 单元	没有保持的 I/O 存储器			保持 I/O 存储器
	在启动时没有初始化 DM 和 EM 区	启动时初始化 DM 和 EM 区域		
		从用户程序	不从用户程序	
CS1-H，CJ1-H，或 CJ1M	使用一般操作 （使用闪存器）或存储卡		在启动时从存储卡使用自动传输 （DIP 开关的引脚 2 为 ON）。	使用任何方法都不可能。必须安装一节电池。
CS1 或 CJ1	在启动时从存储卡使用自动传输 （DIP 开关的引脚 2 为 ON）。			

注 1. 使用无电池操作时，可忽略无电池操作使用的方法。在 PLC 设置中取消对电池低电压的检测。

2. 如果电池没有连接或电池已用完，对 CPU 单元操作就有以下的限定。当然，这与 CPU 单元是否在使用无关。
   - I/O 存储器（包括 HR，DM，和 EM 区域）的内容有可能得不到正确的保持。因此，设定 PLC 设置区：当电源接通 ON 时，I/O 存储器保持标志（A50012）和强制状态保持标志 （A50013）的状态设定为不保持。
   - 不能使用时钟功能。在A351-A354中的时钟数据和在A510和A511中的初启时间都将不可靠。从 CPU 单元写入存储卡中的文件上的文件数据也将不可靠。
   - 在初启时下述数据都为零：电源 ON 时间 （A523），电源中断时间（A512 和 A513），和电源中断次数 （A514）。
   - 在 A100 ～ A199 中的出错记录区域将不被保持。
   - 初启时当前 EM 区将永远是零。
   - 初启时在 EM 文件存储器中将没有文件留下，不能使用文件存储器功能。必须在 PLC 设置中重新设定 EM 文件存储器，要使用 EM 文件存储器必须重新格式化。

CS1-H， CJ1-H， CJ1M，或 CS1D CPU 单元

CS1-H，CJ1-H，CJ1M，或 CS1D CPU 单元在一般操作下，可以执行无电池操作。用户程序和参数数据会自动备份到 CPU 单元的闪存器中，在初启时会自动从闪存器重新存储。在这种情况下，I/O 存储器将不被保持，DM 和 EM 区域必须从用户程序中初始化。

在初启时通过从存储卡中自动传输数据（就象对于 CS1 CPU 单元那样），CS1-H，CJ1-H，CJ1M，或 CS1D CPU 单元也可以执行无电池操作。（配有存储卡后，也能包括 DM 和 EM 区域数据）。

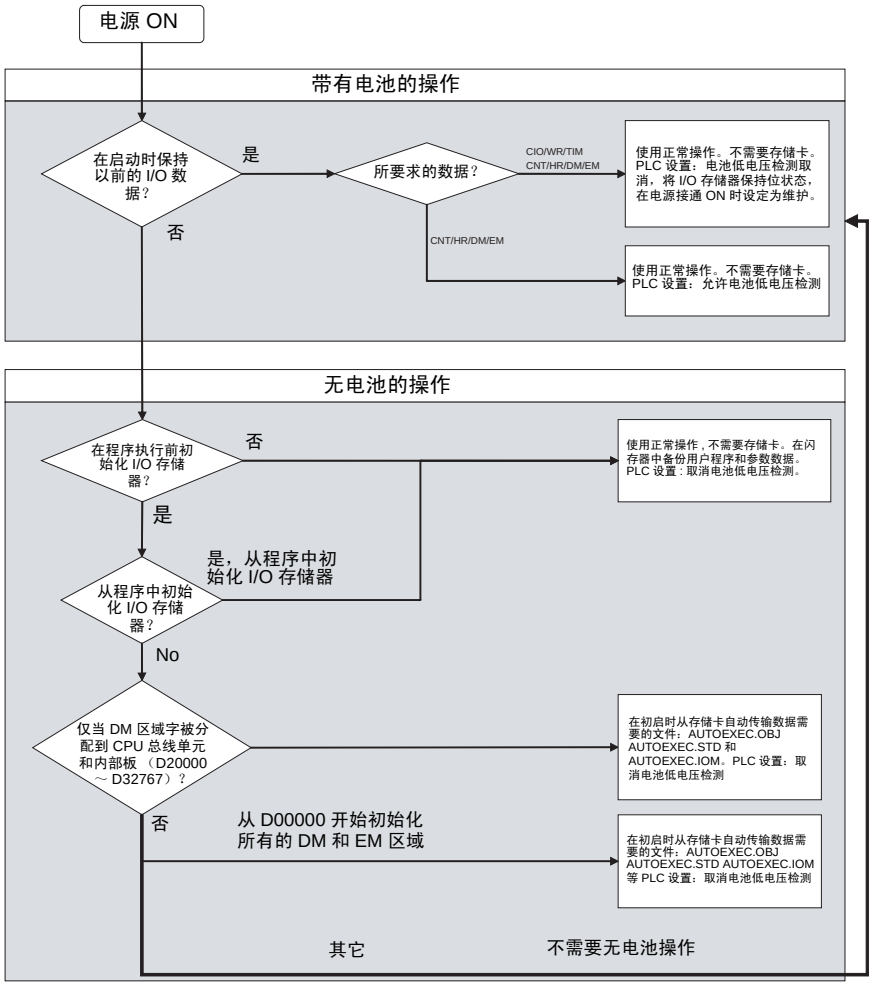
CJ1 和 CJ1 CPU 单元

在初启时通过从存储卡中自动传输数据， CS1 和 CJ1 CPU 单元也可以执行无电池操作。在这种情况下，I/O 存储器将不被保持。（配有存储卡后，也能包括 DM 和 EM 区域数据）。

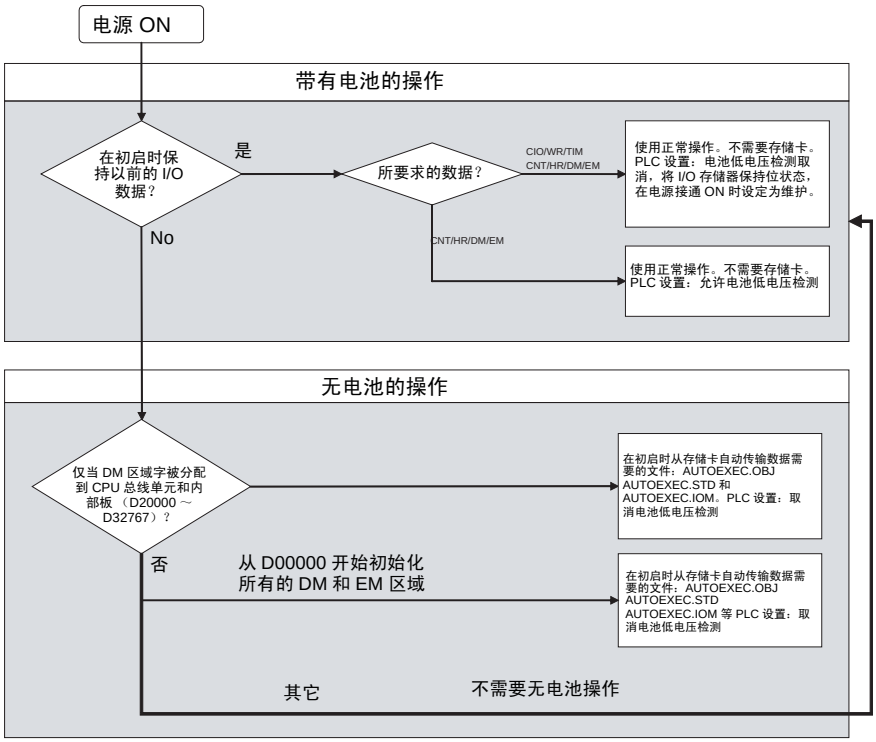
步骤

下面这张流程图表示了两种类型 CPU 单元的运行过程。

CS1-H， CJ1-H， CJ1M，或 CS1D CPU 单元



CS1 和 CJ1 CPU 单元

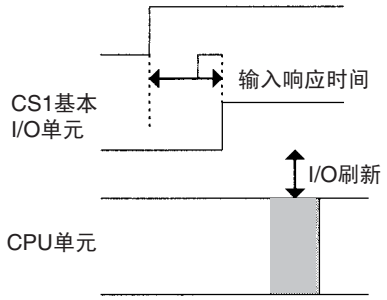


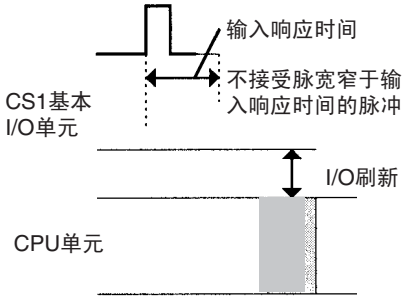
6-11 其它功能

6-11-1 I/O 响应时间设定

对 CS/CJ 基本 I/O 单元的输入响应时间可通过机架和槽号设定。增加输入响应时间以减少抖动和噪声的影响。减小输入响应时间（但保持脉宽大于周期）可接受窄的输入脉冲。

注 CS 系列 CPU 单元中，在一些 C200H 高密度 I/O 单元中已有的高速输入或高速输入单元中，窄于1周期的脉宽可以输入。详细内容请参考*6-1-4 高速输入*章节。





PLC 设置

CS/CJ PLC （机架 0 槽 0 到机架 7 槽 9）中 80 槽的输入响应时间可以在地址 10 到 49 中用 80 个字节设定。

手持编程器地址	名称	设定 (Hex)	默认 (Hex)
10 位 0 ～ 7	对机架 0、槽 0， CS/CJ 基本 I/O 单元的输入响应时间	00: 8 ms 10: 0 ms 11: 0.5 ms 12: 1 ms 13: 2 ms 14: 4 ms 15: 8 ms 16: 16 ms 17: 32 ms	00 (8 ms)
:	:	:	:
49 位 8 ～ 15	对机架 7、槽 9， CS/CJ 基本 I/O 单元的输入响应时间	同上	00 (8 ms)

6-11-2 I/O 区域分配

在扩展机架 （CS/CJ 扩展机架和 C200H 扩展 I/O 机架）中可以使用编程设备设定 I/O 分配的第一个字。这个功能允许每个机架的 I/O 分配区域固定在 CIO 0000 ～ CIO 0999 范围中。（第一个字的分配由机架号决定）。

第 7 章
程序传送、试运行操作和调试

本章将描述用于程序传输到 CPU 单元的处理，以及用于测试和调试程序的功能。

7-1 程序传送

当 CPU 单元在 PROGRAM 模式时，使用编程工具将程序、PLC 设置、I/O 内存数据和 I/O 注释传输到 CPU 单元。

CX-Programmer 的程序传输过程

1,2,3...

1. 选择 *PLC*， *TRANSFER*，然后 *TO PLC*。将显示 “下载选项对话框”。
2. 从下面内容中选择传输的选项：程序，PLC 设定，I/O 表，符号，注释和程序索引。

   注 仅当 I/O 表和注释在 CPU 地址中的存储卡中存在时，才可以被选定。
3. 点击 OK 按钮。

使用以下的任一种方法可以传输数据。

- 当电源接通时，自动传输。

当电源接通时，存储卡中的AUTOEXEC.OBJ文件将被读入到CPU单元。（DIP开关上的引脚 2 必须为 ON）。

- 操作期间程序替换

当 CPU 单元正在工作时，用程序将 AR 区中的替换开始位（A65015）置 ON，现存的程序文件可被在辅助区域规定的程序文件所替换。详细内容请参考*第 5 章文件存储功能*。

7-2 试运行操作

7-2-1 强制置位 / 复位

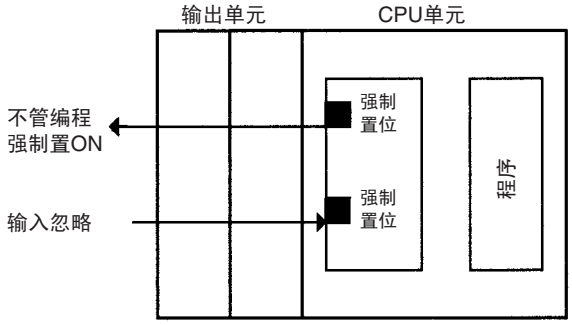
编程工具可以指定位强制置位 （ON）或复位 （OFF）将 （CIO 区域，辅助区域， HR 区域，和定时器 / 计数器完成标志位）。强制状态将比程序或 I/O 刷新时的状态的输出具有更高的优先权。这个状态不能通过指令重写，在它被编程设备清除以前，不管程序或外部输入的状态，它将被存储。

在试运行操作期间，使用强制置位 / 复位操作进行强制输入和输出或在调试期间强制一定的条件。

既可在 MONITOR 也可在 PROGRAM 方式下，执行强制置位 / 复位操作，但不能在 RUN 方式下执行。

注 同时使强制状态保持位 （A50013）和 IOM 保持位 （A50012）变为 ON，当转变操作方式时，能保持强制置位或复位位的状态。

使强制状态保持位 （A50013）和 IOM 保持位 （A50012）变为 ON，并在初启设定 PLC 设置时，设定强制状态保持位，保持强制状态保持位保持的状态。当电源断开时，保持已经强制置位或复位位的状态。



下述区域可以强制置位和复位。

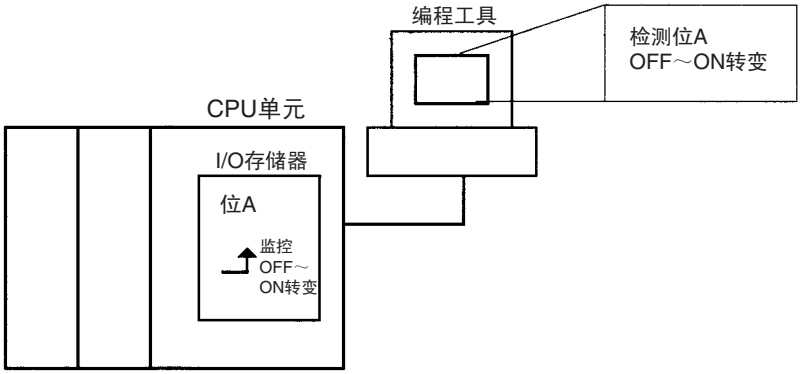
CIO （I/O 位，数据链接位， CPU 总线单元位，特殊 I/O 单元位，内装板位，SYSMAC BUS 位，光 I/O 单元位，工作位），WR 区域，定时器完成标志，HR 区，计数器完成标志。（仅限于CS系列CPU单元支持内装板，SYSMAC BUS，和 I/O 终端区域）。

编程工具操作

- 为强制置位 / 复位选择位。
- 选择强制置位或复位。
- 清除强制状态 （包括同时清除所有强制状态）。

7-2-2 微分监视

当 CPU 单元检测到由编程工具设定的位已从 OFF 转变到 ON 或从 ON 转变到 OFF，在微分监视完成标志（A50809）中将指明结果。当为微分监视设定的条件满足时，标志将变为 ON。编程工具将在屏幕上监控和显示这些结果。



CX-Programmer 编程工具的操作

1,2,3...

1. 右点击微分监视选定位。
2. 从 PLC 菜单点击*微分监视*。将显示微分监视对话框。
3. 点击*上升*或*下降*。
4. 点击开始按钮。当检测到指定的变化时，蜂鸣器将响起，并且计数值将递增。
5. 点击停止按钮。停止微分监视。

相关辅助位 / 字

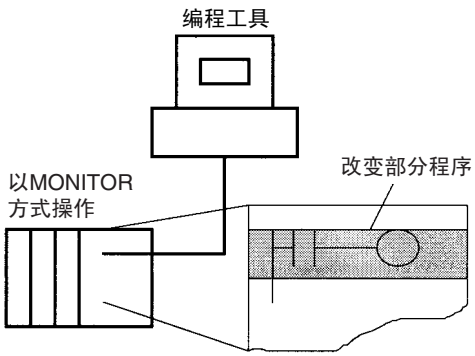
名称	地址	说明
微分监视完成标志	A50809	在微分监视期间，当满足微分监视条件时，变为 ON。 注：微分监视开始时，标志将清除。

7-2-3 在线编辑

当 CPU 单元是在 MONITOR 或 PROGRAM 方式时，直接从编程设备的 CPU 单元中使用在线编辑功能来增加或改变部分程序。手持编程器一次在一条指令中增加或改变，CX-Programmer 一次可以编辑一个或多个程序部分。因此这个功能适用于程序的小修改而不用停止 CPU 单元运行。

对于不同的编辑任务，可以同时从两台以上计算机运行的 CX-Programmer（也可以从一个手持编程器）在线编辑。

在线编程



当在 MONITOR 方式下对 CPU 单元中程序在线编辑，周期将增加一到几个周期时间。

为了在线编辑以后在闪存器中备份数据，CS1-H，CJ1-H，CJ1M 和 CS1D CPU 单元的周期也将增加。在这个期间 BKUP 指示器将闪亮。备份的进度显示在 CX-Programmer 上。每循环的增加量列于下表中。

CPU 单元	每个周期中增加量	
	在线编辑	备份到闪存器
EV1 前 CS1 CPU 单元	90 ms max.	不支持
EV1 或以后 CS1 CPU 单元	12 ms max.	
CS1-H CPU 单元		4% 或周期
CS1 CPU 单元		不支持
CJ1-H CPU 单元		4% 或周期
CJ1M CPU 单元		

使用 CS1-H，CJ1-H，CJ1M 或 CS1D CPU 单元，可以连续编辑的数目是有限的。实际数目取决于执行的编辑的类型，但可以参考以下信息。

CJ1M-CPU@@： 40 个编辑
CS1G-CPU@@H/CJ1G-CPU@@H：160 个编辑
CS1H-CPU@@H/CJ1H-CPU@@H：400 个编辑

如果超过限制数目，在 CX-Programmer 或手持编程器上将显示一条信息，在 CPU 单元完成备份数据之前不能继续编辑。

任务大小和周期延长

当使用版本 1 或更高版本的 CS1 CPU 单元， CS1-H CPU 单元， CS1D CPU 单元， CJ1 CPU 单元，或 CJ1M CPU 单元，由于在线编辑几乎不受正在编辑的任务 （程序）的大小的影响，周期的时间长度将延长。

当使用 EV1 以前的 CS1 CPU 单元时，正在编辑的任务的大小将决定程序停止在线编辑的时间长短。通过将程序分成小的任务，使用在线编辑功能时，周期延长的时间量将比其它的 PLC 方式缩短。

注意

在MONITOR方式下，当使用在线编辑程序重写时，周期将比正常情况来得长，因此必须确认延长的时间量没有超过在 PLC 设置中设定的周期监控时间。如果它确实超过了监控时间，那么将发生周期时间超时， CPU 单元将停止。在改变到RUN 或MONITOR方式前，首先选定PROGRAM方式来重新起动CPU单元。

注 如果被在线编辑的任务包含一个块程序，那么以前的执行数据（如等待（WAIT）或暂停状态）由于在线编辑将被清除，同时下一个操作将从头开始。

从 CX-Programmer 执行在线编辑

1,2,3...

1. 显示将被编辑的程序部分。
2. 选择将被编辑的指令。
3. 选择*程序*，*在线编辑*和*开始*。
4. 编程指令。
5. 选择程序，*在线编辑*和*传送改变*。指令将被检查，如果没有错误，他们将被传输到 CPU 单元。指令将在 CPU 单元中重写，在这个时候将增加周期。

! **注意** 仅在证实延长的周期不影响操作后，才继续在线编辑。如果周期太长时，输入信号可能不能输入。

临时禁止在线编辑

可以在一个循环中禁止在线编辑，确保在那个循环中机器控制的响应特性。通过编程设备的在线编辑将在一个循环中禁止，在这个循环期间接收到的任何在线编辑要求将被保持到下一个循环。

通过使在线编辑禁止位 （A52709）变为 ON 和将在线编辑禁止位确认器（A52700-A52707）设定到 A5 可以禁止在线编辑。当完成这些设定，并接收到在线编辑要求后，在线编辑将为等待状态，并且在线编辑等待标志 （A20110）变为 ON。

在线编辑禁止位 （A52709）变为 OFF 时，将执行在线编辑，在线编辑处理标志 （A20111）变为 ON，并且在线编辑等待标志 （A20110）将显示 OFF。当在线编辑完成后，在线编辑处理标志 （A20111）变为 OFF。

当在线编辑正在执行期间，使在线编辑禁止位 （A52709）变为 ON，可以临时禁止在线编辑。同样，在线编辑等待标志 （A20110）将为 ON。

在第一个在线编辑要求处于等待状态期间，如果收到第二个要求，将不记录第二个要求并将产生一个出错。

在线编辑也可以禁止，用于防止随机在线编辑。如上所述，通过使在线编辑禁止位 （A52709）变为 ON 和将在线编辑禁止位确认器 （A52700 ～ A52707）设定到 A5，可以禁止在线编辑。

用编程设备允许在线编辑

如果不能从程序允许在线编辑，可以用 CX-Programmer 允许在线编辑。

1,2,3...

1. 用手持编程器执行在线编辑

   如果用手持编程器执行在线编辑，在线编辑等待状态不能清除，手持编程器被锁住，并不能操作。

   这种情况下，将 CX-Programmer 连接到另一个串行端口并使在线编辑禁止位 （A52709）变为 OFF，执行在线编辑。手持编程器将重新允许操作。

2. 用 CX-Programmer 执行在线编辑

   在在线编辑处于等待状态时，如果继续操作， CX-Programmer 可能离线。如果这种情况发生，将计算机重新连接到 PLC，并使在线编辑禁止位（A52709）变为 OFF。

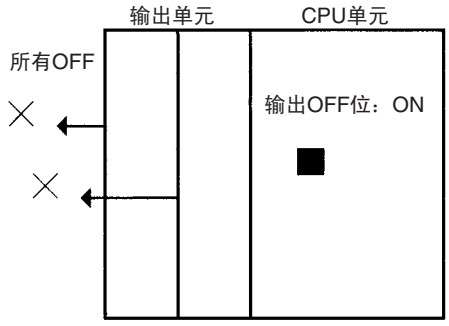
相关辅助位 / 字

名称	地址	说明
在线编辑禁止位确认器	A52700 ～ A52707	确认在线编辑禁止位 （A52709） 不是 5A：在线编辑禁止位未确认 5A：在线编辑禁止位确认
在线编辑禁止位	A52709	禁止在线编辑，使这个位变为 ON，将在线编辑禁止位确认器 （A52700 ～ A52707）设定到 A5。
在线编辑等待标志	A20110	为 ON，由于在线编辑被禁止，在线编辑处于等待状态。
在线编辑处理标志	A20111	为 ON，在线编辑处理正被执行。

输出关断

如果通过 OUT 指令或用编程工具使输出 OFF 位（A50015）变为 ON，所有输出单元的所有输出将变为 OFF （这也适用于 CJ1M CPU 单元中的脉冲输出或内置通用输出应用），在 CPU 单元前面的 INH 指示器将点亮。

不管电源是断开还是接通，输出 OFF 位的状态将保持。



7-2-4 数据跟踪

数据跟踪功能使用下列计时方法的任何一种，对指定 I/O 存储器数据进行采样，它将采样数据存储在跟踪内存中，随后可用编程工具和检查这些数据。

- 指定采样时间 （10 ～ 2,550ms ，以 10 ms 为单位）
- 每一个采样周期
- 当执行跟踪存储器采样指令 （TRSM）时

可在I /O存储器中指定最多31位和6个字对其采样。跟踪内存容量是4，000字。

基本步骤

1,2,3...

1. 当使用 CX-Programmer 设定参数并在采样命令开始执行后，采样将开始。
2. 当采样触发条件满足后，采样数据 （上述步骤 1 以后）将被跟踪，稍延迟后 （见注 1）数据将被存储在跟踪存储器中。
3. 跟踪存储器数据将被采样，跟踪结束。

注 延迟值：定义了从跟踪开始位（A50814）变为 ON 开始向前在采样到跟踪内存的数据中有多少个采样周期的偏移量。设定范围如下表所示。

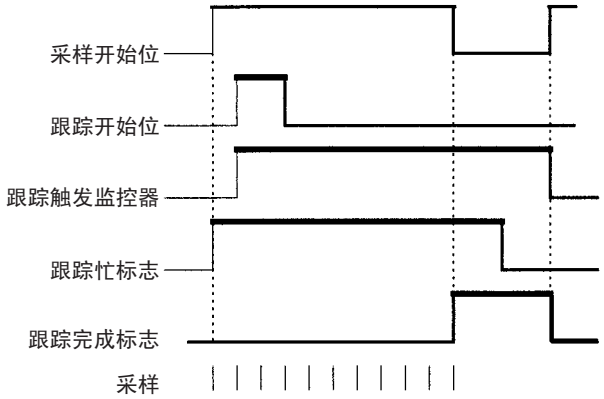
采样字的号	设定范围
0	–1999 ～ 2000
1	–1332 ～ 1333
2	–999 ～ 1000
3	–799 ～ 800
4	–665 ～ 666
5	–570 ～ 571
6	–499 ～ 500

正延迟：根据延迟设定，延迟存储数据。

负延迟：根据设定的延迟，存储前面的数据。

例：以 10 ms 为周期的采样，延迟值为 –30 ms，就得到 –30 x 10 = 300 ms，在跟踪位开始触发前 300 ms 的数据将被存储。

注 用编程工具使采样开始位（A50815）变为 ON。不能用用户程序使这位变为 ON。



以下跟踪可以执行。

定时数据跟踪

定时数据跟踪将以固定时间间隔对数据采样数据。以 10 ms 为单位，指定采样时间为 10 ms 至 2,550ms。在用户程序中不要使用 TRSM 指令，并确保设定的采样周期大于 0。

单循环数据跟踪

在整个周期所有任务完成后，单循环数据跟踪将对 I/O 刷新数据采样在用户程序中不要使用 TRSM 指令，并确保设定的采样周期大于 0。

通过 TRSM 数据跟踪

当执行跟踪存储器采样指令 （TRSM）时，执行一次采样。在一个程序中使用两次以上 TRSM 指令时，在跟踪触发条件满足以后，每执行一次 TRSM 指令就采样一次。

数据跟踪步骤

使用以下步骤执行跟踪。

1,2,3...

1. 使用 CX-Programmer 设定跟踪参数：采样数据的地址，采样周期和触发条件。
2. 使用 CX-Programmer 或使采样开始位 （A50815）置 ON 开始采样。
3. 使跟踪触发条件有效。
4. 结束跟踪。
5. 使用 CX-Programmer 读出*跟踪数据*。
   a) 从 PLC 菜单选择*数据跟踪*。
   b) 从执行菜单选择*选择*。
   c) 从执行菜单选择*执行*。
   d) 从执行菜单选择读。

相关辅助位 / 字

名称	地址	说明
采样开始位	A50815	使用编程设备使这个位变为 ON，开始采样。这个位必须由编程设备置 ON。 不要从用户程序使这个位变为 ON 和 OFF。 注：当数据跟踪完成后，这个位将被清除。
跟踪开始位	A50814	触发条件满足，这位变为 ON 时，跟踪触发将被监控，同时采样数据。这位允许以下跟踪。 1) 定时跟踪 （在 10 ～ 2,555ms 之间以固定间隔跟踪） 2)TRSM 指令跟踪 （当执行 TRSM 指令时跟踪） 3) 一个循环跟踪 （所有循环任务执行结束时跟踪）

名称	地址	说明
跟踪触发监控标志	A50811	在跟踪开始位已经变为 ON 后，当跟踪触器满足时，这个标志变为 ON。采样开始位变为 ON，当采样再次开始时，这个标志将变为 OFF。
跟踪忙标志	A50813	根据采样开始位，当开始采样时，这个标志变为 ON；跟踪完成后，变为 OFF。
跟踪完成标志	A50812	在跟踪操作期间，当跟踪触发条件满足后，如果跟踪存储器满，这个位变为 ON；当下一个采样操作开始时，变为 OFF。

附录 A

PLC 比较表：

CJ 系列， CS 系列， C200HG/HE/HX, CQM1H,CVM1 和 CV 系列可编程序控制器

功能比较

项目			CJ 系列	CS 系列	C200HX/HG/HE	CVM1/CV 系列	CQM1H
基本特性	容量	I/O 点数	2,560 点	5,120 点	1,184 点	6,144 点	512 点
		程序容量 (VM)	120 K 步 一步大约为一个字。操作手册第 10~5 章后面指令执行时间和部数内容。	250K 步一步大约为一个字。详情参见操作手册 10~5 节后面指令执行时间和步数内容	2 K 字 （Z 为 63.2K 字）	62 K 字	15.2 K 字
		最大 (DM)	32 K 字	32 K 字	6 K 字	24 K 字	6 K 字
		I/O 位	160 个字 （2,560 位）	320字（5,120位）	40 字（640 位）	128 字 （2,048 位）	32 字（512 位）
		(WR)	2,644 个字 （42,304 位）+ WR: 512 个字（8,192 位）=3,156 个字 （50,496 位）	2,644 个字（42,304 位）+WR：512 个字（8,192 位）=3,156 个字（50,496 位）	408 字（6,528 位）	168 字 （2,688 位）+400 字（6,400 位）	158 字（2,528 位）
		(HR)	512 字 （8,192 位）	512 字 (8,192 位)	100 字 （1,600 位）	300 字 （4,800 位）最多：1，400 字（12,400 位）	100 字（1,600 位）
		最多允许扩充数据存储器容量 (EM)	32 K 字 x 7 个存储单元	32K 字 x13 个存储单元	6 K字x 3个存储单元 （Z 为 6K 字 x 16 个存储单元）	32K 字 x8 个存储单元（可选择）	6 K 字
		计时 / 计数器数量	各 4,096 个	各 4,096 个	总计：512	1,024 点	总计：512
	处理速度	基本指令 (LD)	CJ1: 0.08 μs min. CJ1-H: 0.02 μs min. CJ1M: 0.1 μs min.	CS1: 0.04 μs min. CS1-H: 0.02 μs min.	0.104 μs min.	0.125 μs min.	0.375 μs min.
		特殊指令 (MOV)	CJ1: 0.25 μs min. CJ1-H: 0.18 μs min. CJ1M: 0.3 μs min.	CS1: 0.25 μs min. CS1-H: 0.18 μs min.	0.417 μs min.	4.3 μs min.	17.7 μs
		系统通用时间	CJ1: 0.5 ms CJ1-H: 一般模式 0.3 ms 并行处理模式 0.2 ms CJ1M: 0.5 ms min.	CS1：0.5msCS1－H：一般模式 0.3ms 并行处理模式 0.2ms	0.7 ms	0.5 ms	0.7 ms
		在线编辑延迟 （写）	CJ1: 大约 12 ms CJ1-H: CPU4@ 约 11 ms 和 CPU6 约 8 ms CJ1M: 约 14 ms	CS1: 大约 12 ms CS1-H：CPU4 口约 11ms 和 CPU6 约 8ms	80 ms （Z 为 160 ms）	500 ms	典型值：250 ms

项目		CJ 系列	CS 系列	C200HX/HG/HE	CVM1/CV 系列	CQM1H
结构	螺丝固定	否	是	是	是	否
	DIN 导轨固定	是	是	是	否	是
	底板	否	是	是	是	否
	尺寸 （H x D, mm）	90 x 65	130 x 123	130 x 118	250 x 100	110 x 107
（单元 / 机架）数	I/O 单元	40 单元	89 单元 （包括从机架）	10 或 16 单元	64 单元 （8 机架 x 8 单元）	16 单元
	CPU 总线单元	16 单元	16 单元	无	16 单元	无
	扩展 I/O 机架	3 机架	7 机架	3 机架	7 机架	1 机架
任务功能		有	有	无	无	无
CPU 处理模式（程序执行和外设服务）	一般模式	有	有	---	---	---
	外设服务优先模式	有	有	---	---	---
	带同步内存访问的并行处理	CJ1: 无 CS1-H: 有 CJ1M: 无	CS1: 无 CS1-H: 有	无	无	无
	带异步内存访问的并行处理	CS1: 无 CJ1-H: 有 CJ1M: 无	CS1: 无 CS1-H: 有	无	无	无
I/O 刷新方式	周期刷新	有	有	有	有	有
	定时刷新	无	无	无	有	无
	过零刷新	无	无	无	有	无
	立即刷新	有	有	无	有	无
	带 IORF 指令立即刷新	有	有	有	有	有
时钟功能		有	有	有	有	有 （需存储器合）
RUN 输出		有 （取决于电源模块）	有 （取决于电源模块）	有 （取决于电源模块）	有	无
开始工作模式 （当手持编程器未连接，PLC 缺省设置情况下）		RUN 模式	CS1: 编程模式 CS1-H: RUN模式	RUN 模式	RUN 模式	编程模式
电源中断处理禁止		CJ1: 无 CJ1-H: 有 CJ1M: 有	CS1: 无 CS1-H: 有	无	无	无
无电池运行		CJ1: 存储器卡 CJ1-H: 存储器卡或闪存器 CJ1M:存储器卡或闪存器	CS1: 存储器卡 CS1-H: 存储器卡或闪存器	存储器卡	存储器卡	存储器卡
自动备份到闪存器		CJ1: 无 CJ1-H: 有 CJ1M: 有	CS1: 无 CS1-H: 有	无	无	无
重新启动继续		无	无	无	有	无

项目			CJ 系列	CS 系列	C200HX/HG/HE	CVM1/CV 系列	CQM1H
外部存贮器	介质		存储器卡（闪存 ROM）	存储器卡（闪存 ROM）	存储器卡（EEPROM, EPROM）	存储器卡（RAM,EEPROM, EPROM）	存储器卡（ROM,EEPROM, EPROM）
	容量		48 M 字节	48 M 字节	4 ～ 32K 字（Z 为 4 ～ 64k 字）	32 ～ 512k 字（ROM：64 ～ 512K 字节）EEPROM：64 ～ 128k 字节 EPROM：0.5 ～ 1M 字节	4 ～ 16 K 字
	内容		程序， I/O 存储 器，参数	程序， I/O 存储器，参数	程序， I/O 存储器，参数	程序，I/O 存储器，参数	程序，只读 DM，参数
	读 / 写方式		编程装置，用户程序（文件存储器指令）或主连接	编程装置，用户程序（文件存储器指令）或主连接	接通 SR 位 (ON)	编程装置，用户程序（文件存储器指令）主连接或存储器卡写入器	接通 SR 位 (ON)
	文件格式		二进制	二进制	二进制	二进制	二进制
	当作文件的扩充数据存储器		有（除 CJ1M CPU 单元外）	有	无	无	无
	开始启动时程序自动传输		有	有	有	有	有
内装板			无	串行通信板	串行通信板	无	串行通信板
内置式串行口			有 (RS-232C x 1)	有 (RS-232C x 1)	有 (RS-232Cx1)	有（RS-232C或 RS-422 x 1）	由 （RS-232C x 1）
串行通信	外设口	外设总线	有	有	有	有	有
		Host Link (SYSMAC WAY)	有	有	有	无 （可与外设接 D 连接）	有
		无协议	无	无	有	无	有
		NT 链接	有	有	无	无	无
	CPU单元置式 RS-232C 口	外设总线	有	有	有	无	无
		Host Link (SYSMAC WAY)	有	有	有	有	有
		无协议	有	有	有	无	有
		NT 链接	有 (1:N)	有 (1:N)	有	无	有 (1:1)
		串行 PLC 链接	有 （仅 CJ1M）	无	无	无	无
	在通信板上 RC-232C 或 RS-422/ RS-485	外设总线	无	无	有	无	无
		Host Link (SYSMAC WAY)	No	有， WG,MP 和 CR 命令不支持	有， CR 命令不支持	有， WG 和 MP 命令不支持	有， CR 命令不支持
		无协议	无	无	有	无	有
		NT 链接	无	有	有	无	有 （1:1 和 1:N）
		协议宏	无	有	有	无	有
		CompoWay/F Master	无	有 （采用协议宏）	有 （采用协议宏）	无	有 （采用协议宏）

项目		CJ 系列	CS 系列	C200HX/HG/HE	CVM1/CV 系列	CQM1H
中断	I/O 中断	有 （最多 2 个中断输入单元：32 点，在 CJ1MCPU 单元上加 4 点内置式 I/O）（CJ1 CPU 不支持 I/O 中断）	有 （最多 4 个或 2 个中断输入单元：32 点）	有 （最多 2 个中断输入单元：16 点）	有 （最多 4 个中断输入单元：32 点）	有 （4 个内置式 CPU 总线单元）
	定时中断	有	有	有	有	有
	单步定时器中断	无	无	无	无	有
	计数方式输入中断	有 （仅 CJ1M CPU 单元）	无	无	无	有
	高速定时器中断	有 （仅 CJ1M CPU 单元）	无	无	无	有
	外部中断	有 （CJ1 CPU 单元不支持外部中断）	有	无	无	无
	来之于通信板中断	无	有	有	无	无
	电源接通中断	无	无	无	有	无
	电源断开中断	有	有	无	有	无
	中断响应时间	0.17ms （CJ1M CPU 单元上内置 I/O：0.12MS）	C200H 系列 I/O 单元：1msCJ 系列 I/O：0.1ms	1 ms	---	大约 0.1 ms
PLC 设置区		无用户地址 （仅可用编程装置包括手持式编程器可进行设定）	无用户地址 （仅可用编程装置包括手持式编程器可进行设定）	固定 DM 区位置：DM6600 ～ DM6655，DM6550 ～ DM6559 可用手持编程器进行设定。	无用户地址 （仅可用编程装置包括手持编程器可进行设定）	固定 DM 区位置：DM6600 ～ DM6655 可用手持编程器进行设定。

项目			CJ 系列	CS 系列	C200HX/HG/HE	CVM1/CV 系列	CQM1H
初始化设定	I/O	对基本 I/O 单元输入响应时间	在 PLC 设置中设定	在 PLC 设置中设定	无	无	在 PLC 设置中设定
		机架首址	用编程装置在 I/O 表中设定（但机架号排列固定）	用编程装置在 I/O 表中设定（但机架号排列固定）	无	在 PLC 设置中设定（但机架序号可设定）	无
		SYSMAC 光学总线主 I/O 单元首址	无	无	无	在 PLC 设置中设定	无
		I/O 错误校验运行	无	无	无	在 PLC 设置中设定	无
	存储器	用户存储器保护	DIP 开关设定	DIP 开关设定	DIP 开关设定	由主开关设定决定	DIP 开关设定
		保持区	无	无	无	在 PLC 设置中设定	无
		用户致命错误的保持 I/O 字（除电源故障外）	无	无	无	在 PLC 设置中设定	无
		当电源接通 PLC 时，用 IOM 保持位保存存储器。	在 PLC 设置中设定	在 PLC 设置中设定	在 PLC 设置中设定	在 PLC 设置中设定	在 PLC 设置中设定
		当电源接通时，用强制状态保持位保存存储器。	在 PLC 设置中设定	在 PLC 设置中设定	在 PLC 设置中设定	在 PLC 设置中设定	在 PLC 设置中设定
		DIP 开关状态监控	有	有	有	无	有
	指令	设置间接 DM 数据为 BCD 或二进制	直接输入允许	直接输入允许	无	在 PLC 设置中设定	无
		JMP(0) 指令多次使用	多次使用已允许	多次使用已允许	无	在 PLC 设置中设	无
		指令出错（继续或停止）操作	在 PLC 设置中设	在 PLC 设置中设	无	无	无
		后台执行	CJ1: 无 CJ1-H: 有 CJ1M: 有	CS1: 无 CS1-H: 有	无	无	无
	文件存储器	启动时自动传输	DIP 开关设定决定（从存储器卡中自动读取）	DIP 开关设定决定（从存储器卡中自动读取）	DIP 开关设定决定（从存储器卡中自动读取）	由 PLC 设置中设定或由 DIP 开关设定（从存储器卡中自动读取）	DIP 开关设定决定（从存储器卡中自动读取）
		转换或 EM 文件	在 PLC 设置中设定	在 PLC 设置中设定	无	无	无
	中断	中断响应	无	无	在 PLC 设置中设定（C200H/高速响应）	无	无
		错误检测	在 PLC 设置中设定	在 PLC 设置中设定	在 PLC 设置中设定	无	无
		在 I/O 中断程序执行时，保持 I/O 中断	无	无	无	在 PLC 设置中设定	无
		电源断开中断允许 / 禁止	在 PLC 设置中设定	在 PLC 设置中设定	无	在 PLC 设置中设定	无
		定时中断时间间隔设定	在 PLC 设置中设定 (10ms,1.0ms)（对 CJ1M CPU 单元仅为 1.0ms）	在 PLC 设置中设定 (10 ms, 1.0 ms)	在 PLC 设置中设定	在 PLC 设置中设定 (10 ms, 1 ms, 0.5 ms)	无

项目			CJ 系列	CS 系列	C200HX/HG/HE	CVM1/CV 系列	CQM1H
初始化设定（续）	电源	再启动继续位保持	无	无	无	在 PLC 设置中设定	无
		启动模式	在 PLC 设置中设定	在 PLC 设置中设定	在 PLC 设置中设定	在 PLC 设置中设定	在 PLC 设置中设定
		启动条件设定	CJ1: 无 CJ1-H: 有 CJ1M: 有	CS1: 无 CS1-H: 有	无	无	无
		启动跟踪	无	无	无	在 PLC 设置中设定	无
		电池电压低测试	在 PLC 设置中设定	在 PLC 设置中设定	在 PLC 设置中设定	在 PLC 设置中设定	在 PLC 设置中设定
		瞬间电源中断时间	无	无	无	在 PLC 设置中设定	无
		电源断延时时间检测	在 PLC 设置中设定	在 PLC 设置中设定	PLC 设置时设定（检测电源开后继续运行的时间）	无	无
		作为致命 / 非致命电源中断错误	无	无	无	在 PLC 设置中设定	无
	循环	I/O 刷新	无	无	PLC 设置中设定（仅限于特殊 /O 单元）	在 PLC 设置中设定	无
		恒定循环时间	PLC 设置中设定 (1 ～ 32,000Ms)	PLC设置中设定 (1 ～ 32,000Ms)	PLC 设置中设定 (1 ～ 9,999ms)	PLC 设置中设定 (1 ～ 32,000Ms)	PLC 设置中设定 (1 ～ 9,999ms)
		监控循环时间	PLC 设置中设定（10 ～ 40,000ms，初始化设定：固定 1,000ms）	PLC 设置中设定（10 ～ 40,000ms，初始化设定：固定 1,000ms）	PLC 设置中设定 (0 ～ 99) 单位：1s,10ms,100ms（初始化设定：固定 120ms）	PLC 设置中设定（10 ～ 40,000ms，初始化设定：固定 1,000ms）	PLC 设置中设定 (0 ～ 99) 单位：1s,10ms,100ms（初始化设定：固定 120ms）
		检测循环时间结束不允许	无	无	PLC 设置中设定	无	PLC 设置中设定
		异步指令执行和外设服务	无	无	无	PLC 设置中设定	No
	串行通信	RS-232C 口通信设定	DIP 开关设定自动检测或由 PLC 设置	DIP 开关设定自动检测或由 PLC 设置	DIP 开关设定自动检测或由 PLC 设置	DIP 开关设定自动检测或由 PLC 设置	DIP 开关设定自动检测或由 PLC 设置
		外设口通信设定	在 PLC 设置中设定	在 PLC 设置中设定	在 PLC 设置中设定	由 DIP 开关设定	在 PLC 设置中设定
		通信板通信设定	无	无	PLC 设置	无	PLC 设置
	CPU 处理模式	并行处理模式	CJ1: 无 CJ1-H: 有 CJ1M: 无	CS1: 无 CS1-H: 有	无	无	无
		外设处理优先模式	有	有	无	无	无
	其它外设服务	服务时间	在 PLC 设置中设定（外设服务时间固定）	在 PLC 设置中设定（外设服务时间固定）	在 PLC 设置中设定（内置式 RS-232C 口，通信板外设口）	无	在 PLC 设置中设定（内置式 RS-232C 口，通信板外设口）
		测量 CPU 总线单元服务间隔	无	无	无	在 PLC 设置中设定	无
		特殊 I/O 单元循环刷新停止	在 PLC 设置中设定	在 PLC 设置中设定	在 PLC 设置中设定	无	无
		CPU 总线连接应用	无	无	无	在 PLC 设置中设定	无

项目			CJ 系列	CS 系列	C200HX/HG/HE	CVM1/CV 系列	CQM1H
初始化设定（续）	手持编程器	手持编程器语言	DIP 开关设定	CS1：DIP 开关设定 CS1-H：由手持编程器设定	DIP 开关设定	无	DIP 开关设定
	错误	错误记录区	无 （固定）	无 （固定）	无 （固定）：DM 6001 ～ DM 6030	PLC 设置中设定	无 （固定）：DM 6569 ～ DM 6599
		在错误记录中不登记用户定义故障错误	CJ1: 无 CJ1-H: 有 CJ1M: 有	CS1: 无 CS1-H: 有	无	无	无
	运行	CPU 等待	无	无	无	PLC 设置中设定	无
辅助区域	条件标志	ER, CY, <, >, = 始终 ON/OFF 标志	输入使用符号，如：ER	输入使用符号，如：ER	有	有	有
		时钟脉冲	输入使用符号，如：0.1 s	输入使用符号，如：0.1 s	有	有	有
	服务	CPU 服务禁止位	无	无	无	有	无
		连接设备码	无	无	无	有	无
		外设处理循环时间	无	无	无	有	无
		CPU 总线单元服务间隔	无	无	无	有	无
		外设连接到 CPU 允许 / 禁止	无	无	无	有	无
		主连接 /NT 连接服务禁止位	无	无	无	有	无
		定时刷新禁止位	无	无	无	有	无
		外设服务禁止位	无	无	无	有	无
		内部板一般用途监控区	无	有	有	无	有
		循环时间结束	有	有	有	有	有
	任务	首任务标志	有	有	无 （仅有首任务扫描标志）	无 （仅有首任务扫描标志）	无 （仅有首任务扫描标志）
	调试	在线编辑禁止标志	有	有	有 (AR)	无	无
		在线编辑等待标志	有	有	有 (AR)	无	无
		输出断开位	有	有	有	有	有
		强制状态保持位	有	有	有	有	有
	文件存储器	文件存储器指令标志	有	有	无	有	无
		EM 文件存储器格式错误标志	有 （除 CJ1M CPU 单元）	有	无	无	无
		EM 文件格式起始存储器单元	有 （除 CJ1M CPU 单元）	有	无	无	无
	存储器	DIP 开关状态标志	有 （脚 6）	有 （脚 6）	有 （AR，脚 6）	无	有 （AR，脚 6）
		IOM 保持位	有	有	有	有	有
	中断	最大子程序 / 作用处理时间	有	有	有	无	无
		中断任务错误标志	有	有	有	无	无

项目				CJ 系列	CS 系列	C200HX/HG/HE	CVM1/CV 系列	CQM1H
辅助区域（续）	错误	错误记录存储区 / 点		有	有	否	有	否
		错误码		有	有	有	有	有
	初始化设定	PLC 设置初始化		否	否	有	否	有
	通信	PLC 连接操作等级标志		有 （PLC 连接辅助区域位）	有 （PLC 连接辅助区域位）	有 (AR)	否	否
	电源	电源中断标志		否	否	否	有	否
		电源中断时间		否	否	否	有	否
		电源接通时间		有	有	否	有	否
		在电源中间时时间（包括电源断开）		有	有	否	有	有
		瞬间电源中断次数		有 （电源中断次数）	有 （电源中断次数）	有 （电源中断次数）	有	有 （电源中断次数）
		总计电源接通时间		有	有	否	否	否
分配方式		格式		分配根据按秩序连接单元所需的字数	分配根据单元所需的字数和跳过的空槽数	固定字分配：每一单元自动分配一个字	分配根据单元所需的字数和跳过的空槽数	分配根据按秩序连接单元所需的字数
		组 2 高密度 I/O 单元分配		无	与基本 I/O 单元相同	组 2 分配区在 AR 区域 （位置取决于板前开关状态）	无	无
		字保留方式		改变 CX － Progrmmer 生成的 I/O 表	改变 CX － Progrmmer 生成的 I/O 表	和空槽生成 I/O 表或改变 CX － Progrmmer 生成的 I/O 表	虚拟 I/O 单元或改变 CX － Progrmmer 生成的 I/O 表	启动时自动分配
		特殊 I/O 单元分配	CIO 区	根据单元号在特殊 I/O 单元区分配 10 个字一个单元，总计 96 个单元。	根据单元号在特殊 I/O 单元区分配 10 个字一个单元，总计 96 个单元。	根据单元号在特殊 I/O 单元区（IR 区）分配，10 个字一个单元，总计 16 个单元。	与基本 I/O 单元一样：（随单元不同而不同） 2 ～ 4 字分配在 I/O 区。	与基本 I/O 单元一样：（随单元不同而不同） 1,2 或 4 个字分配在 I/O 区。
			DM 区	根据单元号在 D20000 ～ D29599 内，每个单元 100 个字，共计 96 个单元。	根据单元号在 D20000 ～ D29599 内，每个单元 100 个字，共计 96 个单元。	分配在 DM1000 ～ DM1999 和 DM2000 ～ DM2599, 每个单位 100 个字，共计 16 个单元。	无	无
		CPU 总线单元 / CPU 总线单元分配	CIO 区	根据单元号， 25 个字一个单元，共计 16 个单元，分配在 CPU 总线单元区。	根据单元号， 25 个字一个单元，共计 16 个单元，分配在 CPU 总线单元区。	无	根据单元号，25 个字一个单元，共计 16 个单元，分配在 CPU 总线单元区。	无
			DM 区	根据单元号，每单元 100 个字，共计 16 个单元，分配在 D30000 ～ D315900 内。	根据单元号，每单元 100 个字，共计 16 个单元，分配在 D30000 ～ D315900 内。	无	根据单元号，每单元 100 个字，共计 16 个单元，分配在 D30000 ～ D315900 内。	无

项目			CJ 系列	CS 系列	C200HX/HG/HE	CVM1/CV 系列	CQM1H
I/O 存储器	CIO 区		有	有	有	有	有
	WR 区		有	有	无	无	无
	暂存继电器区		有	有	有	有	有
	辅助区		有	有	有	有	有
	特殊区		无	无	有	无	有
	连接区		有（数据连接区）	有（数据连接区）	有（数据连接区）	无	有
	C200H 特殊 I/O 单元区		有	有	有（CIO 区）	无	无
	内置 I/O 区		有（仅 CJ1M CPU 单元内置 I/O 带有）	无	无	无	无
	串行 PLC 连接区		有（仅 CJ1M CPU 单元带有）	无	无	无	无
	DM 区		有	有	有	有	有
	扩展数据存储器 (EM) 区		有（地址包括存储器号可被指定）（CJ1M CPU 单元不支持）	有（地址包括存储器号可被指定）	有（Z 可指定地址但存储器不能指定）	有（地址包括存储器号不能指定；存储器必须更换，需要 EM 单元）	有（无存储器）
	计时 / 计数器区		有	有	有	有	有
	变址寄存器		有	有	无	有	无
	数据寄存器		有	有	无	有	无
	强制位 / 复位区	CIO 区	有	有	有	有	无
		WR 区	有	有	无	无	有
		保持区	有	有	有	无	无
		辅助区	无	无	有	无	有
		SR 区	无	无	无	无	无
		链接区	无	无	有	无	无
		计时 / 计数器区	有（标志）	有（标志）	有（标志）	有（标志）	有（标志）
		DM 区	无	无	无	无	无
		EM 区	无	无	无	无	无

项目			CJ 系列	CS 系列	C200HX/HG/HE	CVM1/CV 系列	CQM1H
指令变化间接寻址	上升沿微分（执行一次）		有（用 @ 指定）	有（用 @ 指定）	有（用 @ 指定）	有（用↑指定）	有（用 @ 指定）
	下降沿微分（执行一次）		有（用 % 指定）	有（用 % 指定）	无（用 DIFD 指令替代）	有（用↓指定）	无（用 DIFD 指令获得此功能）
	立即刷新		有（用 ! 指定）	有（用 ! 指定）	无（用 IORF 指令取代）	有（用 ! 指定）	无（用 IORF 指令获得此功能）
	对 DM/EM 间接寻址	BCD 模式	有 (0000 ～ 9999) 用星号注	有 (0000 ～ 9999) 用星号注	有 (0 ～ 9999)	有 (0 ～ 9999)	有 (0000 ～ 9999) 用星号注
		二进制模式	有（00000 ～ 32767）用 @ 注：0000 ～ 7FFF Hex：00000 ～ 31767 8000 ～ FFFF Hex：00000 ～ 32767 下一个存储器	有（00000 ～ 32767）用 @ 注：0000 ～ 7FFF Hex：00000 ～ 31767 8000 ～ FFFF Hex：00000 ～ 32767 下一个存储器	无	有，但仅为间接寻址，采用 PLC 存储器地址	无

指令比较

项目		助记符	CJ 系列	CS 系列	C200HX/HG/HE	CVM1/CV 系列	CQM1H
顺序输入指令	取 / 与 / 或	LD/ AND/ OR	有	有	有	有	有
	块与 / 块或	AND LD/OR LD	有	有	有	有	有
	非	NOT	有	有	有	有	无
	条件接通	UP	有	有	No	有 (*1)	无
	条件断开	DOWN	有	有	No	有 (*1)	无
	位测试	TST/ TSTN	有 （位位置用二进制）定义：0000 ～ 000Fhex	有 （位位置用二进制）定义：0000 ～ 000Fhex	有 （位位置用 BCD 吗定义）(*2)	有 （位位置用 BCD 吗定义）(*1)	无
顺序输出指令	输出	OUT	有	有	有	有	有
	暂存	TR	有	有	有	有	有
	保持	KEEP	有	有	有	有	有
	上升 / 下降沿微分	DIFU/ DIFD	有 (LD↑, AND↑，OR↑) (LD↓， AND↓，OR↓)	有 (LD↑, AND↑，OR↑) (LD↓， AND↓，OR↓)	有 (DIFU/DIFD)	有(LD↑, AND↑，OR↑)/ (LD↓，AND↓，OR↓)	有 (DIFU/DIFD)
	位置和复位	SET/ RSET	有	有	有	有	有
	多位置位 / 复位	SETA/ RSTA	有 （用二进制定义起始位和位数）	有 （用二进制定义起始位和位数）	无	(*1) （用 BCD 定义起始位和位数）	无
	单个位置位 / 复位	SET/ RSTB	CJ1: 无 CJ1-H: 有 CJ1M: 有	CS1: 无 CS1-H: 有	无	无	无
	单个位输出	OUTB	CJ1: 无 CJ1-H: 有 CJ1M: 有	CS1: 无 CS1-H: 有	无	无	无
顺序控制指令	结束 / 空操作	END/ NOP	有	有	有	有	有
	联锁 / 联销清除	IL/ILC	有	有	有	有	有
	跳转 / 跳转结束	JMP/ JME	有 （BCD：0 ～ 1023 定义跳转号）	有 （BCD：0 ～ 1023 定义跳转号）	有 （BCD：0 ～ 99 定义跳转号）	有 （BCD：0 ～ 99 定义跳转号）	有 （BCD：0 ～ 99 定义跳转号）
	条件跳转	CJP/ CJPN	有 （BCD：0 ～ 1023 定义跳转号）	有 （BCD：0 ～ 1023 定义跳转号）	无	有 （BCD：0 ～ 999 定义跳转号）(*1)	无
	多路跳转 / 跳转结束	JMP0/ JME0	有	有	无	无 （但 PLC 设置时可设定允许跳转号零为多路跳转）	无
	FOR/NEXT 循环	FOR/ NEXT	有	有	无	无	无
	循环退出	BREAK	有	有	无	无	无

项目		助记符	CJ 系列	CS 系列	C200HX/HG/HE	CVM1/CV 系列	CQM1H
定时器和计数器指令	定时器	TIM (BCD)	有	有	有	有	有
		TIMX (binary)	有 (*4)	有 (*4)	无	无	无
	高速定时器	TIMH (BCD)	有	有	有	有	有
		TIMHX (binary)	有 (*4)	有 (*4)	无	无	无
	1MS 定时器	TMHH (BCD)	有	有	无	无	无
		TMHHX (binary)	有 (*4)	有 (*4)	无	无	无
	累加定时器	TTIM (BCD)	有	有	有	有	有
		TTIMX (binary)	有 (*4)	有 (*4)	无	无	无
	长定时器	TIML (BCD)	有	有	无	有	无
		TIMLX (binary)	有 (*4)	有 (*4)	无	无	无
	多路输出定时器	MTIM (BCD)	有	有	无	有	无
		MTIMX (binary)	有 (*4)	有 (*4)	无	无	无
	计数器	CNT (BCD)	有	有	有	有	有
		CNTX (binary)	有 (*4)	有 (*4)	无	无	无
	可逆计数器	CNTR (BCD)	有	有	有	有	有
		CNTRX (binary)	有 (*4)	有 (*4)	无	无	无
	复位定时器 / 计数器	CNR (BCD)	有 （仅复位计时或计数器）	有 （仅复位计时或计数器）	无	有 （也在 CIO 指定区域清零）	无
		CNRX (binary)	有 (*4)	有 (*4)	无	无	无
比较指令	符号比较	=, <, etc.	有 （支持所有 LD， OR 和 AND）	有 （支持所有 LD， OR 和 AND）	有 (*2) （仅支持 AND）	有 (*1) （仅支持 AND）	无
	比较 / 双字比较	CMP/ CMPL	有	有	有	有 (*3)	有
	带符号二进制比较 / 带符号双字二进制比较	CPS/ CPSL	有	有	有	有 (*1)	有
	块比较	BCMP	有	有	有	有	有
	扩展块比较	BCMP2	有 （仅 CJIM CPU 单元）	无	无	无	无
	表比较	TCMP	有	有	有	有	有
	多重比较	MCMP	有	有	有	有	有
	相等	EQU	无	无	无	有	无
	区域范围比较	ZCP/ ZCPL	CJ1: 无 （可用比较指令获得） CJ1-H: 有 CJ1M: 有	CS1: 无（可用比较指令获得） CS1-H: 有	有	无	无 （可用比较指令获得）

项目		助记符	CJ 系列	CS 系列	C200HX/HG/HE	CVM1/CV 系列	CQM1H
数据传送指令	传送	MOV	有	有	有	有	有
	双字传送	MOVL	有	有	无	有	无
	取反传送	MVN	有	有	有	有	有
	双字取反传送	MVNL	有	有	无	有	无
	数据交换	XCHG	有	有	有	有	有
	双字数据交换	XCGL	有	有	无	有	无
	快速传送	MOVQ	无	无	无	有	无
	块传送	XFER	有（用二进制 0 ～ 65535 指定要传送的字数）	有（用二进制 0 ～ 65535 指定要传送的字数）	有（用 BCD 0 ～ 6144 指定要传送的字数）	有（用 BCD 0 ～ 9999 指定要传送的字数）	有（用 BCD 0 ～ 9999 指定要传送的字数）
	块设定	BSET	有	有	有	有	有
	位传送	MOVB	有（用二进制指定源和目的位置）	有（用二进制指定源和目的位置）	有（用 BCD 指定源和目的位置）	有（用 BCD 指定源和目的位置）	有（用 BCD 指定源和目的位置）
	多位传送	XFRB	有	有	有	有 (*1)	有
	字节传送	MOVD	有	有	有	有	有
	单字分配	DIST	有（用其它指令可作栈操作功能。偏置值用二进制 0 ～ 65535 指定）	有（用其它指令可作栈操作功能。偏置值用二进制 0 ～ 65535 指定）	有（可用作栈操作功能。偏置值用 BCD：0 ～ 8999 指定）	有（用其它指令可作栈操作功能。偏置值用 BCD：0 ～ 9999 指定）	有（用其它指令可作栈操作功能。偏置值用 BCD：0 ～ 9999 指定）
	数据采集	COLL	有（用其它指令可作栈操作功能。偏置值用二进制 0 ～ 65535 指定）	有（用其它指令可作栈操作功能。偏置值用二进制 0 ～ 65535 指定）	有（可用作栈操作功能。偏置值用 BCD：0 ～ 7999 指定）	有（用其它指令可作栈操作功能。偏置值用 BCD：0 ～ 9999 指定）	有（可用作栈操作功能。偏置值用 BCD：0 ～ 7999 指定）
	EM 块在存储器间传送	BXFR	无（用 XFEREM 区值接寻址，可达到传送 65，535 个字功能）	无（用 XFEREM 区值接寻址，可达到传送 65，535 个字功能）	无	有 (*1)	无
	EM 块传送	XFR2	无	无	有	无	无
	EM 存储器传送	BXF2	无	无	有	无	无
	传送到寄存器	MOVR	有（对间接 DM/EM 无地址指定）	有（对间接 DM/EM 无地址指定）	无	有（对间接 DM/EM 地址指定）	无
	传送计时 / 计数器当前值到寄存器	MOVRW	有	有	无	无（使用 MOVR 仅对完成标志）	无

项目		助记符	CJ 系列	CS 系列	C200HX/HG/HE	CVM1/CV 系列	CQM1H
数据移位指令	移位寄存器	SFT	有	有	有	有	有
	可逆移位寄存器	SFTR	有	有	有	有	有
	异步移位寄存器	ASFT	有	有	有	有	有
	字移位	WSFT	有 （与 CV 同：3 个操作数）	有 （与 CV 同：3 个操作数）	有	有	有
	算术左 / 右移	ASL/ ASR	有	有	有	有	有
	左 / 右循环	ROL/ ROR	有	有	有	有	有
	单字节左 / 右移位	SLD/ SRD	有	有	有	有	有
	N 位数据左 / 右移位	NSFR/ NSFL	有 （移位数据和起始用二进制指定）	有 （移位数据和起始用二进制指定）	无	有 （移位数据和起始用 BCD 指定） (*1)	无
	左 / 右移几位双字左 / 右几移位	NASL/ NASR, NSLL/ NSRL	有 （用二进制指定移位位数）	有 （用二进制指定移位位数）	无	有 （移位数据和起始用 BCD 指定）(*1)	无
	双字左 / 右移	ASLL/ ASRL	有	有	无	有	无
	双字循环左 / 右移	ROLL/ RORL	有	有	无	有	无
	无进位循环左 / 右移位双字无进位循环左 / 右移位	RLNC/ RRNC, RLNL/ RRNL	有	有	无	有 (*1)	无
递增和递减指令	BCD 递增 / 递减	++B/– –B (INC/ DEC)	有 (++B/– –B)	有 (++B/– –B)	有 (INC/DEC)	有 (INC/DEC)	有 (INC/DEC)
	双字 BCD 递增 / 递减	++BL/– –BL (INCL/ DECL)	有 (++BL/– –BL)	有 (++BL/– –BL)	无	有 (INCL/DECL)	无
	二进制递增 / 递减	++/– – (INCB/ DECB)	有 C 进位，借位时 CY 为 ON(++L/ －－ L)	有 C 进位，借位时 CY 为 ON(++L/ －－ L)	无	有	无
	双字二进制递增 / 递减	++L/– – L INBL/ DCBL)	有 C 进位，借位时 CY 为 ON(++L/ －－ L)	有 C 进位，借位时 CY 为 ON(++L/ －－ L)	无	有	无
四则运算指令			有	有	有	有	有

项目		助记符	CJ 系列	CS 系列	C200HX/HG/HE	CVM1/CV 系列	CQM1H
转换指令	BCD →二进制 / 双字 BCD →双字二进制	BIN/ BINL	有	有	有	有	有
	二进制→ BCD/ 双字二进制→双字 BCD	BCD/ BCDL	有	有	有	有	有
	二进制求补 / 双字二进制求补	NEG/ NEGL	有 （与 CV 同但对源 8000Hex UP 不变 ON）	有 （与 CV 同但对源 8000Hex UP 不变 ON）	有	有	有
	16 位→ 32 位带符号二进制	SIGN	有	有	无	有	无
	数据评码	MLPX	有	有	有	有	有
	数据编码	DMPX	有 （与 CVM1 － V2 同：可指定最低位为 ON）	有 （与 CVM1 － V2 同：可指定最低位为 ON）	有 （仅最高位为 ON）	有 （与 CVM1 － V2 同：可指定最低位为 ON）	有 （仅最高位为 ON）
	ASCII 码转换	ASC	有	有	有	有	有
	ASCII → HEX	HEX	有	有	有	有 (*1)	有
	列→行 / 行→列	LINE/ COLM	有 （用二进制指定位的位置）	有 （用二进制指定位的位置）	有 （用 BCD 指定位的位置）	有 （用 BCD 指定位的位置）	有 （用 BCD 指定位的位置）
	带符号 BCD →二进制双字带符号 BCD →二进制	BINS/ BISL	有	有	无	有 (*1)	无
	带符号二进制→ BCD 双字带符号二进制→ BCD	BCDS/ BDSL	有	有	无	有 (*1)	无
逻辑指令	逻辑与 / 逻辑或 / 异或异或非	ANDW, ORW, XORW, XNRW	有	有	有	有	有
	双字逻辑与 / 双字逻辑或 / 双字异或双字异或非	ANDL, ORWL, XORL, XNRL	有	有	无	有	无
	求反 / 双字求反	COM/ COML	有	有	有 （仅 COM 指令）	有	有 （仅 COM 指令）
特殊算术指令	BCD 平方根	ROOT	有	有	有	有	有
	二进制平方根	ROTB	有	有	无	有 (*1)	无
	算术处理	APR	有	有	有	有	有
	浮点数除法	FDIV	有	有	有	有	无
	位计数	BCNT	有 （计字的数和计数结果用二进制表示：0 ～ FFFF Hex）	有 （计字的数和计数结果用二进制表示：0 ～ FFFF Hex）	有 （计字的数和计数结果用 BCD 表示：1 ～ 6656）	有 （计字的数和计数结果用 BCD 表示：0 ～ 9999，但 0 为错误）	有 （计字的数和计数结果用 BCD 表示：1 ～ 6656）

项目		助记符	CJ 系列	CS 系列	C200HX/HG/HE	CVM1/CV 系列	CQM1H
浮点数运算指令	浮点数→ 16 位 /32 位二进制 16 位 /32 位二进制→ 浮点数	FIX/ FIXL, FLT/ FLTL	有	有	无	有 (*1)	有
	浮点数加 / 减 / 乘 / 除	+F, –F, *F, /F	有	有	无	有 (*1)	有
	度→ 弧度弧度→度	RAD, DEG	有	有	无	有 (*1)	有
	正弦 / 余弦 / 正切 / 反正弦 / 反余弦 / 反正切	SIN, COS, TAN, ASIN, ACOS, ATAN	有	有	无	有 (*1)	有
	平方根	SQRT	有	有	无	有 (*1)	有
	指数	EXP	有	有	无	有 (*1)	有
	对数	LOG	有	有	无	有 (*1)	有
	指数幂	PWR	有	有	无	无	无
	浮点数十进制比较	Exam-ples: =F, <>F	CJ1: 无 CJ1-H: 有 CJ1M: 有	CS1: 无 CS1-H: 有	无	无	无
	浮点数十进制 → 文本字符串	FSTR, FVAL	CJ1: 无 CJ1-H: 有 CJ1M: 有	CS1: 无 CS1-H: 有	无	无	无
双精度浮点数转换和计算指令	同上面单精度浮点数转换和计算指令一样	Exam-ple: FIXD	CJ1: 无 CJ1-H: 有 CJ1M: 有	CS1: 无 CS1-H: 有	无	无	无
表格数据处理指令	堆栈设置	SSET	有 （栈控制的四个字字数定义用二进制：5 ～ 65535）	有 （栈控制的四个字字数定义用二进制：5 ～ 65535）	无	有 （栈控制的四个字字数定义用 BCD：3 ～ 9999）	无
	压入栈	PUSH	有	有	无	有	无
	先进先出	FIFO	有	有	无	有	无
	后进先出	LIFO	有	有	无	有	无
	寻找最大值 / 寻找最小值	MAX, MIN	有 （控制数据字段两个字。表长定义用二进制：1 ～ FFFF）	有 （控制数据字段两个字。表长定义用二进制：1 ～ FFFF）	有 （控制数据字段－个字。表长定义用 BCD：1 ～ 999）	有 （控制数据字段－个字。表长定义用 BCD：1 ～ 999）	有 （控制数据字段－个字。表长定义用 BCD：1 ～ 999）
	数据搜索	SRCH	有 （表长定义用二进制：1 ～ FFFF. PLC 存储器地址输出到 IRO。相配的数可写入 DRO）	有 （表长定义用二进制：1 ～ FFFF. PLC 存储器地址输出到 IRO。相配的数可写入 DRO）	有 （表长定义用 BCD：1 ～ 6556。PLC 存储器地址输出到 C ＋ 1。相配的数不能输出到 DRO）	有 （表长定义用 BCD：1 ～ 9999。PLC 存储器地址输出到 IRO。相配的数不能输出到 DRO）	有 （表长定义用 BCD：1 ～ 6556。PLC 存储器地址输出到 C ＋ 1。相配的数不能输出到 DRO）
	帧校验和	FCS	有	有	有	无	有
	求和	SUM	有 （与 C200HX/HG/HE 相同：可对字节和字求和）	有 （与 C200HX/HG/HE 相同：可对字节和字求和）	有 （可对字节和字求和）	有 （仅对字求和）	有 （可对字节和字求和）
	交换字节	SWAP	有 （可用于数据通信和其它应用）	有 （可用于数据通信和其它应用）	无	无	无
	定位记录表	DIM	有	有	无	无	无
	设置记录位置	SETR	有	有	无	无	无
	获得记录位置	GETR	有	有	无	无	无

项目		助记符	CJ 系列	CS 系列	C200HX/HG/HE	CVM1/CV 系列	CQM1H
数据控制指令	标度	SCL	有	有	有	无	有
	标度 2	SCL2	有	有	无	无	有
	标度 3	SCL3	有	有	无	无	有
	PID 控制	PID	有（当 PV ＝ SV 时，输出转换可在 0% ～ 50% 之间。 PID 的采样周期用二进制定义）	有（当 PV ＝ SV 时，输出转换可在 0% ～ 50% 之间。 PID 的采样周期用二进制定义）	有（PID 和采样周期用 BCD 定义）	有（PID 和采样周期用 BCD 定义） (*1)	有（PID 和采样周期用 BCD 定义）
	带自动调整 PID 控制	PIDAT	CJ1: 无 CJ1-H: 有 CJ1M: 有	CS1: 无 CS1-H: 有	无	无	无
	限位控制	LMT	有	有	无	有 (*1)	无
	静带控制	BAND	有	有	无	有 (*1)	无
	静域控制	ZONE	有	有	无	有 (*1)	无
	平均值	AVG	有（扫描数的定义用二进制）	有（扫描数的定义用二进制）	有（扫描数的定义用 BCD）	无	有（扫描数的定义用 BCD）
子程序指令	子程序调用 / 子程序进入 / 子程序返回	SBS, SBN, RET	有（子程序号用 BCD：0 ～ 1023 定义）	有（子程序号用 BCD：0 ～ 1023 定义）	有（子程序号用 BCD：0 ～ 255 定义）	有（子程序用 BCD：0 ～ 999 定义）	有（子程序号用 BCD：0 ～ 255 定义）
	宏	MCRO	有（子程序号用 BCD：0 ～ 1023 定义）	有（子程序号用 BCD：0 ～ 1023 定义）	有（子程序号用 BCD：0 ～ 255 定义）	有（子程序用 BCD：0 ～ 999 定义） (*1)	有（子程序号用 BCD：0 ～ 255 定义）
	全局子程序指令	GSBS, GSBN, RET	CJ1: 无 CJ1-H: 有 CJ1M: 有	CS1: 无 CS1-H: 有	无	无	无
中断控制指令	设置中断屏蔽	MSKS	有	有	无（所有中断处理用 INT）	有	无（所有中断处理用 INT）
	清除中断	CLI	有	有	无	有	无
	读中断屏蔽	MSKR	有	有	无	有	无
	禁止中断	DI	有	有	无	无	无
	允许中断	EI	有	有	无	无	无
	允许计时器	STIM	无	无	无	无	有

项目		助记符	CJ 系列	CS 系列	C200HX/HG/HE	CVM1/CV 系列	CQM1H
高速计数器 / 脉冲输出指令	控制模式	INI	有 (*5)	无	无	无	有
	读当前值	PRV	有 (*5)	无	无	无	有
	设比较标	CTBL	有 (*5)	无	无	无	有
	设脉冲	PULS	有 (*5)	无	无	无	有
	设频率	SPED	有 (*5)	无	无	无	有
	加速度控制	ACC	有 (*5)	无	无	无	有
	位置控制	PLS2	有 (*5)	无	无	无	有
	原点搜索	ORG	有 (*5)	无	无	无	无
	PWM 输出	PWM	有 (*5)	无	无	无	有
步指令	步定义和步开始	STEP/ SNXT	有	有	有	有	有
基本 I/O 单元指令	I/O 刷新	IORF	有	有（也用于 C200H 组 2 高密度 I/O 单元。包括组 2 高密度 I/O 刷新功能（MPRF））	有（也用于 C200H 组 2 高密度 I/O 单元和特殊 I/O 单元）	有	有
	7 段译码	SDEC	有	有	有	有	有
	组 2 高密度 I/O 刷新	MPRF	无	无	有	无	无
	+ 键输入	TKY	无	无	有	无	有
	十六进制的键输入	HKY	无	无	有	无	有
	数字开关输入	DSW	无	无	有	无	有
	矩阵输入	MTR	无	无	有	无	无
	7 段显示输入	7SEG	无	无	有	无	有
特殊 I/O 单元指令	特殊 I/O 单元读和特殊 I/O 单元写（I/O 读和 I/O 写）	IORD/ IOWR (READ/ WRIT)	IORD/IOWR（达 96 个单元。不再用于发 FINS 命令）	IORD/IOWR（达 96 个单元。不再用于发 FINS 命令）	IORD/IOWR	READ/WRIT	无
	I/O 读 2 和 I/O 写 2	RD2/ WR2	无	无	无	有 (*1)	无
文本串处理指令	串传送	MOV$	有	有	无	无	无
	连接串	+$	有	有	无	无	无
	取左串	LEFT$	有	有	无	无	无
	取右串	RGHT$	有	有	无	无	无
	取中间串	MID$	有	有	无	无	无
	寻找串	FIND$	有	有	无	无	无
	计算串长	LEN$	有	有	无	无	无
	取代串	RPLC$	有	有	无	无	无
	删除串	DEL$	有	有	无	无	无
	交换串	XCHG$	有	有	无	无	无
	清除串	CLR$	有	有	无	无	无
	插入串	INS$	有	有	无	无	无

项目		助记符	CJ 系列	CS 系列	C200HX/HG/HE	CVM1/CV 系列	CQM1H
串行通信指令	接收	RXD	有（存储的字节数用二进制定义）（仅用于 CPU 单元的 RS－232C 口。不能用于串行通信单元，或 CPU 单元的外设口）	有（存储的字节数用二进制定义）（仅用于 CPU 单元的 RS－232C 口。不能用于串行通信单元，或 CPU 单元的外设口）	有（存储的字节数用 BCD 定义）（用于外设口，RS－232C 口，或在 CPU 单元中的通信板）。	无	有（存储的字节数用 BCD 定义）（用于外设口，RS－232C 口，或在 CPU 单元中的通信板）。
	发送	TXD	有（存储的字节数用二进制定义）（仅用于 CPU 单元的 RS－232C 口。不能用于串行通信单元，或 CPU 单元的外设口）（未经请求的通信不能使用主连接 EX 命令）	有（存储的字节数用二进制定义）（仅用于 CPU 单元的 RS－232C 口。不能用于串行通信单元，或 CPU 单元的外设口）（未经请求的通信不能使用主连接 EX 命令）	有（存储的字节数用 BCD 定义）（用于外设口，RS－232C 口，或在 CPU 单元中的通信板）（未经请求的通信不能使用主连接 EX 命令）	无	有（存储的字节数用 BCD 定义）（用于外设口，RS－232C 口，或在 CPU 单元中的通信板）（未经请求的通信不能使用主连接 EX 命令）
	修改串行口设置	STUP	有（10 个字设定） 可用于串行通信单元。	有（10 个字设定） 可用于串行通信单元。	有（5 个字设定）	无	有（5 个字设定）
	协议宏	PMCR	有（用二进制定义顺序号，四个操作数，可指定目的单元地址和串行口号）	有（用二进制定义顺序号，四个操作数，可指定目的单元地址和串行口号）	有（用 BCD 定义顺序号，三个操作数）	无	有（用 BCD 定义顺序号，三个操作数）
	PCMCIA 卡宏	CMCR	无	无	有	无	无
网络指令	网络发送 / 网络接收	SEND/ RECV	有（通过主链接连接用于主计算机。不能用于串行通信单元或 CPU 单元的 RS－232C 口）。	有（通过主链接连接用于主计算机。不能用于串行通信单元或 CPU 单元的 RS－232C 口内装板）。	有（通过主链接连接，不能用于主计算机）	有（通过主链接连接，可用于主计算机）	有（通过主链接连接，不能用于主计算机）
	发布命令	CMND	有（通过主链接连接用于主计算机。不能用于串行通信单元或 CPU 单元的 RS－232C 口）。	有（通过主链接连接用于主计算机。不能用于串行通信单元或 CPU 单元的 RS－232C 口内装板）。	无	有（通过主链接连接，可用于主计算机）	有（通过主链接连接，不能用于主计算机）
文件存储指令	读数据文件 / 写数据文件	FREAD/ FWRIT	有	有	无	有 (FILR/FILW)	无
	读程序文件	FILP	无	无	无	有	无
	改变步程序	FLSP	无	无	无	有	无
显示指令	显示信息	MSG	有（信息用）	有（信息用）	有（信息用 CR 结束）	有（信息用 CR 结束）	有（信息用 CR 结束）
	显示长信息	LMSG	无	无	有（信息用 CR 结束）	无	无
	I/O 显示	IODP	无	无	无	有	无
	终端模式	TERM	无	无	有	无	无

项目		助记符	CJ 系列	CS 系列	C200HX/HG/HE	CVM1/CV 系列	CQM1H
时钟指令	日历加法	CADD	有	有	无	有	无
	日历减法	CSUB	有	有	无	有	无
	小时→秒	SEC	有	有	有	有	有
	秒→小时	HMS	有	有	有	有	有
	时钟调整	DATE	有	有	无	有 (*1)	无
调试指令	跟踪内存采样	TRSM	有	有	有	有	有
	记录轨迹	MARK	无	无	无	有 （用 BCD 定义标记号）	无
故障诊断指令	故障报警 / 严重故障报警	FAL/ FALS	有 （信息用 NUL 结束，文本串存储字节由高位到低位，然后字由低位到高位，故障号用二进制指定）。	有 （信息用 NUL 结束，文本串存储字节由高位到低位，然后字由低位到高位，故障号用二进制指定）。	有 （信息用 CR 结束，文本串存储字节由高位到低位，然后字由低位到高位，故障号用 BCD 码指定）。	有 （信息用 CR 结束，文本串存储字节由高位到低位，然后字由低位到高位，故障号用 BCD 码指定）。	有 （信息用 CR 结束，文本串存储字节由高位到低位，然后字由低位到高位，故障号用 BCD 码指定）。
	故障点检测	FPD	有 （信息用 NUL 结束，文本串存储字节由高位到低位，然后字由低位到高位，故障号用二进制指定）。	有 （信息用 NUL 结束，文本串存储字节由高位到低位，然后字由低位到高位，故障号用二进制指定）。	有 （信息用 CR 结束，文本串存储字节由高位到低位，然后字由低位到高位，故障号用 BCD 码指定）。	有 （信息用 CR 结束，文本串存储字节由高位到低位，然后字由低位到高位，故障号用 BCD 码指定）。 (*1)	有 （信息用 CR 结束，文本串存储字节由高位到低位，然后字由低位到高位，故障号用 BCD 码指定）。
其它指令	置进位 / 复进位	STC/ CLC	有	有	有	有	有
	取标志 / 存标志	CCL, CCS	CJ1: 无 CJ1-H: 有 CJ1M: 有	CS1: 无 CS1-H: 有	无	有	无
	延长最大循环时间	WDT	有	有	有	有 (*1)	有
	循环时间	SCAN	无	无	有	无	无
	取寄存器 / 存寄存器	REGL, REGS	无	无	无	有	无
	选择 EM 存储器	EMBC	有	有	有	有	无
	扩充 DM 读	XDMR	无	无	有	无	无
	EM 间接寻址	IEMS	无	无	有	无	无
	允许访问 / 禁止访问	IOSP, IORS	无	CS1: 无 CS1-H: 有	无	有	无
	CV-CS 地址转换指令	FRMCV TOCV	CJ1: 无 CJ1-H: 有 CJ1M: 有	CS1: 无 CS1-H: 有	无	无	无

项目		助记符	CJ 系列	CS 系列	C200HX/HG/HE	CVM1/CV 系列	CQM1H
块程序指令		BPRG/ BEND, IF/ ELSE/ IEND, WAIT, EXIT, LOOP/ LEND, BPPS/ BPRS, TIMW, CNTW, TMHW	有	有	无	有 (*1)	无
任务控制指令	任务 ON/ 任务 OFF	TKON/ TKOF	有	有	无	无	无

注 *1：仅由 CVM1(V2)CPU 支持。
*2：仅由 CPU 口口 Z 型号支持。
*3：由 CV1M 版本 2 支持，继续在相同程序运行。
*4：除 CS1 和 CJ1 CPU 单元。
*5：仅为 CJ1M CPU 带内置式 I/O。一些操作数与 CQM1H 使用的不同。

附录 B
与原上位机链接系统不同之处

用 CS/CJ 系列串行通信板和单元 （CS 系列）所生成的上位机链接系统和其它 PLC 产品系列 CPU 单元所生成的上位机链接系统相比有不同之处。这些不同点在本节中描述。

RS-232C 端口

当从已用的上位机链接系统改成在 CS/CJ 系列 CPU 单元的 RS-232C 端口，串行通信板（CS 系列），或（CS1H/G-CPU □□ RS-232C 端口，CS1W-SCU21 端口，CS1W-SCB21 端口，CS1W-SCB41 端口 1 或 CJW-SCU41 端口 2) 串行通信单元，下列不同之处需考虑。

以前产品	模块型号	CS/CJ 系列产品所需变更	
		接线	其它
C 系列上位机链接单元	3G2A5-LK201-E C500-LK203 3G2A6-LK201-E	连接器已由 25 针改为 9 针。 CS/CJ 系列产品不支持 ST1，ST2 和 RT 信号，不需要和它们相连。	对系统 ST1， ST2 和 RT 同步下列变更是必须的。 同步传送将不再有效。 用 CS/CJ 系列产品可作全－双工传送，但主计算机的通信程序，硬件或两者都得作相应变化。 对 ST1， ST2 和 RT 不同步的系统需作下列改变。 要通信的设置相同 （如波特率），采用主计算机不改变程序是可能的。然而，要允许帧中字符长度不同或不同的 CS/CJ 命令定义，改变程序是必须的 （见注）。
	C200H-LK201	连接器已由 25 针改为 9 针	只要通信的设置相同 （如波特率），采用主计算机不改变程序是可能的。然而，要允许帧中字符长度不同或不同的 CS/CJ 命令定义，改变程序是必须的 （见注）。
C 系列到 CPU 单元	SRM1 CPM1 CPM1A CQM1-CPU@@-E C200HS-CPU@@-E C200HX/HG/HE-CPU@@-E C200HW-COM@@-E	连接线上不作任何改动。	只要通信的设置相同 （如波特率），采用主计算机不改变程序是可能的。然而，要允许对不同的 CS/CJ 命令定义，改变程序是必须的。

以前产品	模块型号	CS/CJ 系列产品所需变更	
		接线	其它
CVM1或CV系列CPU 单元	CVM1/CV-CPU@@-E	不需对接线作任何改动	只要通信的设置相同 （如波特率），采用主计算机不改变程序是可能的。然而，对不同的 CS/CJ 命令定义，改变程序是必须的。
CVM1或CV系列主链接单元	CV500-LK201	端口 1： 连接器已从 25 针改为 9 针。 端口 2 设置为 RS-232C： SG 信号脚由 7 脚改为 9 脚。	**用 CD 系列产品作半－双工传送，下列改变是必须的。** 当使用 SEND，RECV 或 CMND 从 PLC 初始化通信或从主计算机在发送命令同步问题中，检查系统同步。如果需要，切换到全－双工传送状态。 **不用 CD 系列产品作全－双工传送，下列改变是必须的。** 只要通信的设置相同 （如波特率），半－双工传送使用主计算机不作程序更改是可能的。然而，要允许对不同的 CS/CJ 命令定义，改变程序是必须的。

注 当使用 C 模式命令，C 系列上位机链接单元和 CS/CJ 系列串行通信板 / 单元每帧 (即字符串长度) 读 / 写的字数不同。如果以前用于 C 系列上位机链接单元主计算机程序用于 CS/CJ 系列 PLC，功能就会出错。在使用前先检查主计算机程序，对每一需修改的地方作修改，以控制不同帧字符长度。详情参见 *CS/CJ 系列通信命令参考手册 (W342)*。

RS-422A/485 端口

当从已用的上位机链接系统改成在 CS 串行通信板 （CS1W-SCB41 端口 2）或 CJ 系列串行通信单元 （CJ1W-SCU41 端口 1）上的 RS-422A/485 端口，下列不同之处需考虑。

以前产品	模块型号	CS/CJ 系列产品所需变更	
		接线	其它
C 系列上位机链接单元	3G2A5-LK201-E C200H-LK202 3G2A6-LK202-E	31 脚线改接如下所示： SDA: 9 脚到 1 脚 SDB: 5 脚到 2 脚 RDA: 6 脚到 6 脚 RDB: 1 脚到 8 脚 SG: 3 脚 不连接 FG: 7 脚 连接器外壳	只要通信的设置相同 （如波特率），采用主计算机不改变程序是可能的。然而，要允许帧中字符长度不同或不同的 CS/CJ 命令定义，改变程序是必须的 （见注）。
C200HX/HG/HE 通信板	C200HW-COM@@-E	连接线上不作任何改动。	只要通信的设置相同 （如波特率），采用主计算机不改变程序是可能的。然而，要允许不同的 CS/CJ 命令定义，改变程序是必须的。

以前产品	模块型号	CS/CJ 系列产品所需变更	
		接线	其它
CVM1 或 CV 系列 CPU 单元	CVM1/CV-CPU@@-E	不需要对接线作任何改动	只要通信的设置相同（如波特率），采用主计算机不改变程序是可能的。然而，要允许不同的 CS/CJ 命令定义，改变程序是必须的。
CVM1 或 CV 系列上位机链接单元	CV500-LK201		

注 当使用 C 模式命令， C 系列上位机链接单元和 CS/CJ 系列串行通信板 / 单元每帧（即字符串长度）读 / 写的字数不同。如果以前用于 C 系列上位机链接单元主计算机程序用于 CS/CJ 系列 PLC，功能就会出错。在使用前先检查主计算机程序，对每一需修改的地方作修改，以控制不同帧字符长度。详情参见 *CS/CJ 系列通信命令参考手册 (W342)*。