

三菱可编程控制器

MELSEC iQ-R
series

MELSEC iQ-R运动控制器 编程手册(程序设计篇)



-R16MTCPU
-R32MTCPU
-R64MTCPU



安全注意事项


(使用之前务必阅读)

使用本产品前，请仔细阅读本手册及本手册所介绍的关联手册，同时在充分注意安全的前提下正确地操作。

本手册中的注意事项仅记载了与本产品有关的内容。关于可编程控制器系统方面的安全注意事项，请参阅所使用的CPU模块的用户手册。

在“安全注意事项”中，安全注意事项被分为“警告”和“注意”这二个等级。

 警告	表示错误操作可能造成危险后果，导致死亡或重伤事故。
 注意	表示错误操作可能造成危险后果，导致中度伤害、轻伤及设备损失。

注意根据情况不同，即使“注意”这一级别的事项也有可能引发严重后果。

对两级注意事项都须遵照执行，因为它们对于操作人员安全是至关重要的。

请妥善保管本手册以备需要时查阅，并应将本手册交给最终用户。

[设计注意事项]

警告

- I 应在可编程控制器外部设置安全电路，确保外部电源异常或可编程控制器设备故障时，能保证整个系统的安全运行。误输出或误动作可能引发事故。
 - (1) 应在可编程控制器的外部配置异常停止电路、保护电路、正转/反转等相反动作的互锁电路、定位的上限/下限等防止机械损坏的互锁电路。
 - (2) 可编程控制器检测出以下异常状态时，将停止运算，输出将变为以下状态。
 - 电源模块的过电流保护装置或过电压装置动作时将全部输出置为OFF。
 - CPU模块中通过看门狗定时器出错等自诊断功能检测出异常时，根据参数设置，将全部输出保持或置为OFF。
 - (3) 此外，CPU模块无法检测的输入输出控制部分等的异常时，全部输出可能变为ON。此时，应在可编程控制器外部配置失效安全电路，设置安全机构，以保证机械的安全运行。关于安全失效电路的示例，请参阅MELSEC iQ-R模块配置手册的“失效安全电路的思路”。
 - (4) 由于输出电路的继电器或晶体管等的故障，输出可能保持为ON状态或OFF状态。对于可能导致重大事故的输出信号，应在外部设置互锁电路。
- I 输出电路中，由于额定以上的负载电流或负载短路等导致长时间持续过电流的情况下，可能引起冒烟及着火，因此应在外部设置保险丝等的安全电路。
- I 应配置接通可编程控制器本体电源后，再接通外部供应电源的电路。如果先接通外部供应电源，误输出或误动作可能引发事故。
- I 关于网络通信异常时各站的动作状态，请参阅各网络的手册。误输出或误动作可能引发事故。
- I 应在程序中配置互锁电路，以便在将外部设备连接到CPU模块或智能功能模块上对运行中的可编程控制器进行控制(数据更改)时，能始终保证整个系统安全运行。此外，对运行中的可编程控制器进行其它控制(程序更改、参数更改、强制输出、运行状态更改(状态控制))时，应仔细阅读手册，确认足够安全之后再进行操作。如果未认真确认，操作错误可能导致机械损坏或事故。

[设计注意事项]

警告

- | 从外部设备对远程的可编程控制器进行控制时，由于数据通信异常可能无法立即对可编程控制器侧的故障进行处理。应在程序中配置互锁电路的同时，在外部设备与CPU模块之间确定发生通信异常时系统方面的处理方法。
 - | 在模块的缓冲存储器中，请勿对系统区域或禁止写入区域进行数据写入。此外，在从CPU模块对各模块的输出信号之中，请勿输出(ON)禁止使用的信号。如果对系统区域或禁止写入区域进行数据写入，或对禁止使用的信号进行输出，有可能导致可编程控制器系统误动作。关于系统区域或禁止写入区域、禁止使用的信号的详细内容，请参阅各模块的用户手册。
 - | 通信电缆断线的情况下，线路变得不稳定，可能导致多个站网络通信异常。应在程序中配置互锁电路，以便即使发生通信异常也能保证系统安全运行。误输出或误动作可能引发事故。
 - | 对于来自于网络的外部设备的非法访问，需要保证可编程控制器系统安全时，应由用户采取防范措施。此外，对于来自于互联网的外部设备的非法访问，需要保证可编程控制器系统安全时，应采取防火墙等防范措施。
 - | 应在可编程控制器外部设置安全电路，确保外部电源异常或可编程控制器设备故障时，能保证整个系统的安全运行。误输出或误动作可能引发事故。
 - | 对于使用了模块、伺服放大器、伺服电机的具有安全标准(例如机器人等的安全通则等)的系统，应满足安全标准。
 - | 模块、伺服放大器异常时的动作有可能危及系统安全的情况下，应在模块·伺服放大器的外部配置安全电路。
 - | 在模块及伺服放大器的控制电源处于接通状态时，请勿卸下SSCNET III电缆。请勿直视来自模块及伺服放大器的SSCNET III连接器及SSCNET III电缆前端发出的光。如果光线照射到眼睛内，会使眼睛产生刺痛感。(SSCNET III的光源符合JISC6802、IEC60825-1规定的等级1。)
-

[设计注意事项]

注意

- | 请勿将控制线及通信电缆与主电路或动力线捆扎在一起，或使其相互靠得过近。应该彼此相距100mm以上。否则噪声可能导致误动作。
 - | 对灯负载、加热器、螺线管阀等的电感性负载进行控制时，输出OFF→ON时有可能会有大电流(通常的10倍左右)流过，因此应使用额定电流留有余量的模块。
 - | CPU模块的电源OFF→ON或复位时，CPU模块变为RUN状态的时间根据系统配置、参数设置、程序容量等而变动。设计时应做到即使变为RUN状态的时间变动，也能保证整个系统安全运行。
 - | 各种设置的登录中，请勿进行模块安装站的电源OFF及CPU模块的复位。如果在登录中进行模块安装站的电源OFF及CPU模块的复位，闪存内的数据内容将变得不稳定，需要对缓冲存储器中的设置值进行重新设置，再次登录到闪存中。否则可能导致模块故障及误动作。
 - | 从外部设备对CPU模块进行运行状态更改(远程RUN/STOP等)时，应将模块参数的“打开方法设置”设置为“不通过程序OPEN”。将“打开方法设置”设置为“通过程序OPEN”的情况下，从外部设备执行远程STOP时，通信线路将被关闭。此后将无法在CPU模块侧重新打开，也无法从外部设备执行远程RUN。
-

[安装注意事项]

警告

- | 在拆装模块时，必须先将系统使用的外部供应电源全部断开后再进行操作。如果未全部断开，有可能导致触电、模块故障及误动作。
-

[安装注意事项]

注意

- | 为了安全使用，应在(随基板附带的手册)中记载的一般规格的环境中使用可编程控制器。如果在一般规格范围以外的环境中使用，有可能导致触电、火灾、误动作、设备损坏或性能劣化。
 - | 模块安装时，将模块下部的凹槽插入基板的导轨，以导轨的前端为支点，押入直到听见模块上部挂钩发出“咔嚓”声为止。若模块未正确安装，有可能导致误动作、故障或掉落。
 - | 安装没有模块固定用挂钩的模块时，必须将模块下部的凹槽插入基板的导轨，以导轨的前端为支点押入，并用螺栓紧固。若模块未正确安装，有可能导致误动作、故障或掉落。
 - | 在振动较多的环境下使用时，应将模块用螺栓紧固。
 - | 应在规定的扭矩范围内拧紧螺栓。如果螺栓拧得过松，可能导致脱落、短路及误动作。如果螺栓拧得过紧，就会损坏螺栓或模块而导致掉落、短路或误动作。
 - | 扩展电缆应可靠安装到基板的扩展电缆用连接器上。安装后，应确认是否松动。接触不良可能导致误动作。
 - | SD存储卡应压入到安装插槽中可靠安装。安装后，应确认是否松动。接触不良可能导致误动作。
 - | 安装扩展SRAM卡盒时，应可靠压入到CPU模块的卡盒连接用连接器中。安装后应关闭卡盒盖板，确认是否松动。接触不良可能导致误动作。
 - | 请勿直接接触模块、SD存储卡、扩展SRAM卡盒或连接器的导电部位及电子部件。否则可能导致模块故障及误动作。
-

[配线注意事项]

警告

- | 在拆装模块时，必须先将系统使用的外部供应电源全部断开后再进行操作。如果未全部断开，有可能导致触电或模块故障及误动作。
 - | 在安装或配线作业后，进行通电或运行的情况下，必须装好产品附带的端子盖板。若未装好端子盖板，有可能触电。
-

[配线注意事项]

⚠ 注意

- | 必须对FG端子及LG端子采用可编程控制器专用接地(接地电阻小于100Ω)进行接地。否则可能导致触电或误动作。
 - | 压装端子应使用合适的压装端子,并以规定扭矩拧紧。如果使用Y型压装端子,端子螺栓松动的情况下可能导致脱落、故障。
 - | 对模块进行配线时,应确认产品的额定电压及信号排列后正确地进行操作。如果连接了与额定不符的电源或错误配线,可能导致火灾或故障。
 - | 对于外部设备连接用连接器,应使用生产厂商指定的工具进行压装、压接或正确焊接。连接不良的情况下,可能导致短路、火灾或误动作。
 - | 连接器应可靠安装到模块上。接触不良可能导致误动作。
 - | 请勿将控制线及通信电缆与主电路或动力线捆扎在一起,或使其相互靠得过近。应该彼此相距100mm以上。否则噪声可能导致误动作。
 - | 模块上连接的电线及电缆必须纳入导管中或通过夹具进行固定处理。否则由于电缆的晃动或移动、不经意的拉拽等可能导致模块及电缆破损、电缆连接不良而引起误动作。对于扩展电缆,请勿进行剥去包皮的夹具处理。
 - | 连接电缆时,应在确认连接接口类型的基础上正确地操作。如果连接了不同类型的接口或配线错误,可能导致模块或外部设备故障。
 - | 应在规定的扭矩范围内拧紧端子螺栓及连接器的安装螺栓。若螺栓拧得过松,可能引起掉落、短路、火灾或误动作。如果螺栓拧得过紧,就会损坏螺栓或模块而导致掉落、短路、或误动作。
 - | 卸下模块上连接的电缆时,请勿拉拽电缆部分。对于带连接器的电缆,应握住连接模块的连接器进行拆卸。对于端子排连接的电缆,应松开端子排端子螺栓后进行拆卸。如果在与模块相连的状态下拉拽电缆,可能导致误动作或模块及电缆破损。
 - | 应注意防止切屑或配线头等异物掉入模块内。否则有可能导致火灾、故障或误动作。
 - | 为防止配线时配线头等异物混入模块内部,模块上部贴有防止混入杂物的标签。在配线作业中,请勿揭下该标签。系统运行时,必须揭下该标签以利散热。
 - | 可编程控制器应安装在控制盘内使用。至控制盘内安装的可编程控制器电源模块的主电源配线应通过中继端子排进行。此外,电源模块的更换及配线作业应由在触电保护方面受过良好培训的维护作业人员进行操作。关于配线方法,请参阅MELSEC iQ-R模块配置手册。
 - | 系统使用的以太网电缆应符合各模块的用户手册中记载的规格。进行了不符合规格的配线时,将无法保证数据传送正常。
-

[启动・维护注意事项]

警告

- | 请勿在通电的状态下触碰端子。否则有可能导致触电或误动作。
 - | 应正确连接电池连接器。应绝对避免对电池进行充电、拆开、加热、投入火中、短接、焊接、附着液体或使其受到强烈冲击。如果电池处理不当，由于发热、破裂、着火、漏液可能导致人员受伤或火灾。
 - | 在拧紧端子螺栓、连接器安装螺栓或模块固定螺栓以及清洁模块时，必须先将系统使用的外部供应电源全部断开后再进行操作。如果未全部断开，可能导致触电。
-

[启动・维护注意事项]

注意

- | 应在程序中配置互锁电路，以便在将外部设备连接到CPU模块或智能功能模块上对运行中的可编程控制器进行控制(数据更改)时，能始终保证整个系统安全运行。此外，对运行中的可编程控制器进行其它控制(程序更改、参数更改、强制输出、运行状态更改(状态控制))时，应仔细阅读手册，确认足够安全之后再进行操作。如果未认真确认，操作错误可能导致机械损坏或事故。
 - | 从外部设备对远程的可编程控制器进行控制时，由于数据通信异常可能无法立即对可编程控制器侧的故障进行处理。应在程序中配置互锁电路的同时，在外部设备与CPU模块之间确定发生通信异常时系统方面的处理方法。
 - | 请勿拆卸及改造模块。否则有可能导致故障、误动作、人员伤害及火灾。
 - | 使用便携电话及PHS等无线通信设备时，应在所有方向与可编程控制器本体相距25cm以上。否则有可能导致误动作。
 - | 在拆装模块时，必须先将系统使用的外部供应电源全部断开后再进行操作。若未全部断开，有可能导致模块故障或误动作。
 - | 应在规定的扭矩范围内拧紧螺栓。若螺栓拧得过松，有可能导致部件及配线的掉落、短路或误动作。若螺栓拧得过紧，可能会损坏螺栓或模块而导致掉落、短路或误动作。
 - | 产品投入使用后，模块与基板、CPU模块与扩展SRAM卡盒以及端子排的拆装次数不应超过50次(根据IEC61131-2规范)。如果超过了50次，有可能导致误动作。
 - | 产品投入使用后，SD存储卡的安装・拆卸次数不应超过500次。如果超过了500次，有可能导致误动作。
 - | 使用SD存储卡时，请勿触碰露出的卡端子。否则有可能导致故障及误动作。
 - | 使用扩展SRAM卡盒时，请勿触碰电路板上的芯片。否则有可能导致故障及误动作。
 - | 请勿让安装到模块上的电池遭受掉落・冲击。掉落・冲击可能导致电池破损、电池内部漏液。请勿使用遭受过掉落・冲击的电池而应将其废弃。
 - | 控制盘内的启动・保养作业应由在触电保护方面受过良好培训的维护作业人员进行操作。此外，控制盘应上锁，以防止非维护作业人员操作控制盘。
-

[启动・维护注意事项]

⚠ 注意

- l 在接触模块之前，必须先接触已接地的金属等导电物体，释放掉人体等所携带的静电。若不释放掉静电，有可能导致模块故障或误动作。
 - l 试运行前，应将参数的速度限制值设置为较慢的速度，做好发生危险状态时能立即停止的准备之后再行动作确认。
 - l 运行前应进行程序及各参数的确认・调整。机械有可能会有无法预料的动作。
 - l 使用绝对位置系统功能的情况下，新启动时或更换了模块、绝对值对应电机等时，必须进行原点复位。
 - l 应确认制动功能之后再投入运行。
 - l 点检时请勿进行兆欧测试(绝缘电阻测定)。
 - l 维护・点检结束时，应确认绝对位置检测功能的位置检测是否正确。
 - l 控制盘应配锁，以便只有受过电气设备相关培训，具有充分知识的人员才能打开控制盘。
-

[运行注意事项]

⚠ 注意

- l 将个人计算机等外部设备连接到智能功能模块上对运行中的可编程控制器进行控制(特别是数据更改、程序更改、运行状态更改(状态控制))时，应仔细阅读用户手册，确认足够安全之后再进行操作。如果数据更改、程序更改、状态控制错误，有可能导致系统误动作、设备破损及事故。
 - l 将缓冲存储器的设置值登录到模块内的闪存中使用的情况下，登录中请勿进行模块安装站的电源OFF及CPU模块的复位。如果在登录中进行模块安装站的电源OFF及CPU模块的复位，闪存内的数据内容将变得不稳定，需要对缓冲存储器中的设置值进行重新设置，再次登录到闪存中。否则可能导致模块故障及误动作。
 - l 插补运行的基准轴速度指定时，应注意对象轴(第2轴、第3轴、第4轴)的速度有可能大于设置速度(超过速度限制值)。
 - l 试运行及示教等的运行过程中请勿靠近机械。否则可能造成人员伤害。
-

[废弃注意事项]

⚠ 注意

- l 在废弃产品时，应将其作为工业废弃物处理。
 - l 废弃电池时，应根据地方法规将电池与其它废品分开处理。关于欧盟国家内电池规定的详细内容，请参阅MELSEC iQ-R模块配置手册。
-

[运输注意事项]

注意

- l 必须按照运输规定运输含锂电池。关于规定对象机型的详细内容，请参阅MELSEC iQ-R模块配置手册。
 - l 包含有用于木制包装材料的消毒及除虫措施的熏蒸剂的卤素物质(氟、氯、溴、碘等)侵入到三菱电机产品中时可能导致故障。应采取相应措施防止残留的熏蒸剂侵入到三菱电机的产品中。应采取熏蒸剂以外的方法(热处理等)进行处理。此外，消毒及除虫措施应在包装前的木材阶段实施。
-

关于产品的应用

(1) 在使用三菱可编程控制器时，应该符合以下条件：即使在可编程控制器设备出现问题或故障时也不会导致重大事故，并且应在设备外部系统地配备能应付任何问题或故障的备用设备及失效安全功能。

(2) 三菱可编程控制器是以一般工业用途等为目标设计和生产的通用产品。

因此，三菱可编程控制器不应用于以下设备・系统等特殊用途。如果用于以下特殊用途，对于三菱可编程控制器的质量、性能、安全等所有相关责任（包括但不限于债务未履行责任、瑕疵担保责任、质量保证责任、违法行为责任、生产者责任），三菱电机将不负责。

- 面向各电力公司的核电站以及其它发电厂等对公众有较大影响的用途。
- 用于各铁路公司或公用设施目的等有特殊质量保证体系要求的用途。
- 航空航天、医疗、铁路、焚烧・燃料装置、载人移动设备、载人运输装置、娱乐设备、安全设备等预计对人身财产有较大影响的用途。

然而，对于上述应用，如果在限定于具体用途，无需特殊质量（超出一般规格的质量等）要求的条件下，经过三菱电机的判断也可以使用三菱可编程控制器，详细情况请与当地三菱电机代表机构协商。

前言

在此感谢贵方购买了三菱可编程控制器MELSEC iQ-R系列的产品。

本手册是用于使用户了解使用运动控制器时必要的性能规格、投运步骤、配线有关内容的手册。

在使用之前应熟读本手册及关联手册，在充分了解MELSEC iQ-R系列可编程控制器的功能・性能的基础上正确地使用本产品。

将本手册中介绍的程序示例应用于实际系统的情况下，应充分验证对象系统中不存在控制方面的问题。

请妥善保管本手册并将其交给最终用户。

对象模块

R16MTCPU、R32MTCPU、R64MTCPU

EMC指令・低电压指令的对应

关于可编程控制器系统

将符合EMC指令・低电压指令的三菱可编程控制器安装到用户产品上，使其符合EMC指令・低电压指令时，请参阅下述任一手册。

📖 MELSEC iQ-R模块配置手册

📖 安全使用(随基板附带的手册)

符合EMC指令・低电压指令的可编程控制器产品在设备的额定铭牌上印有CE标志。

关于本产品

使本产品符合EMC指令・低电压指令时，请参阅以下手册。

📖 MELSEC iQ-R运动控制器用户手册

目录

安全注意事项	1
关于产品的应用	9
前言	9
EMC指令·低电压指令的对应	9
关联手册	15
术语	16
手册的阅读方法	17
第1章 概要	19
1.1 性能规格	19
运动SFC性能规格	19
运算控制·转换控制规格	20
1.2 运动CPU的程序构成	26
第2章 运动专用顺控程序指令	27
2.1 运动专用顺控程序指令的概要	27
2.2 运动专用顺控程序指令	28
至运动CPU的运动SFC启动请求: M(P).SFCS/D(P).SFCS	29
至运动CPU的伺服程序启动请求: M(P).SVST/D(P).SVST	32
至运动CPU的直接定位启动指令: M(P).SVSTD/D(P).SVSTD	37
至运动CPU的当前值更改指令: M(P).CHGA/D(P).CHGA	47
至运动CPU的指令生成轴的当前值更改指令: M(P).CHGAS/D(P).CHGAS	51
至运动CPU的速度更改指令: M(P).CHGV/D(P).CHGV	55
至运动CPU的指令生成轴的速度更改指令: M(P).CHGVS/D(P).CHGVS	59
至运动CPU的转矩限制值更改指令: M(P).CHGT/D(P).CHGT	64
至运动CPU的位元件的写入: M(P).BITWR/D(P).BITWR	67
至其它机号CPU的中断指令: M(P).GINT/D(P).GINT	70
2.3 注意事项	73
运动专用顺控程序指令中使用的CPU缓冲存储器地址	73
运动专用顺控程序指令的使用块数	75
运动专用顺控程序指令的执行	79
完成状态信息	80
指令的执行顺序	81
第3章 运动SFC程序	83
3.1 运动SFC程序的构成	83
3.2 运动SFC图符号一览	84
3.3 分支·合并图一览	86
3.4 运动SFC程序名	89
3.5 步	90
运动控制步	90
运算控制步	91
子程序调用/启动步	92
清除步	93
3.6 转换	94
3.7 跳转·指针	96
3.8 END	97

3.9	分支·合并	98
	串联转移	98
	选择分支·选择合并	99
	并联分支·并联合并	100
3.10	Y/N转换	102
3.11	运动SFC注释	106
第4章 运算控制程序		108
4.1	运算控制程序	108
4.2	软元件记述	112
4.3	常数记述	114
4.4	标签	115
	种类	115
	数据类型	115
	数组	116
	结构体	117
	注意事项	118
4.5	二项运算	119
	代入: =	119
	加法: +	120
	减法: -	121
	乘法: *	122
	除法: /	124
	余数: %	125
4.6	位运算	126
	位取反(补数): ~	126
	位逻辑积: &	127
	位逻辑和:	128
	位异或: ^	129
	位右移: >>	130
	位左移: <<	131
	符号取反(2的补数): -	132
4.7	标准函数	133
	正弦: SIN	133
	余弦: COS	134
	正切: TAN	135
	反正弦: ASIN	136
	反余弦: ACOS	137
	反正切: ATAN	138
	平方根: SQRT	139
	自然对数: LN	140
	指数运算: EXP	141
	绝对值: ABS	142
	四舍五入: RND	143
	舍去: FIX	144
	进位: FUP	145
	BCD → BIN转换: BIN	146
	BIN → BCD转换: BCD	147
4.8	数据控制	148
	16位整数型标度: SCL	148

	32位整数型标度: DSCL	152
4.9	类型转换	155
	带符号16位整数值转换: SHORT	155
	无符号16位整数值转换: USHORT	156
	带符号32位整数值转换: LONG	158
	无符号32位整数值转换: ULONG	159
	带符号64位浮点值转换: FLOAT	161
	无符号64位浮点值转换: UFLOAT	162
	浮点值的32位 -> 64位转换: DFLT	163
	浮点值的64位 -> 32位转换: SFLT	164
4.10	位软元件状态	165
	ON(常开触点): (无)	165
	OFF(常闭触点): !	166
4.11	位软元件控制	167
	软元件设置: SET	167
	软元件复位: RST	168
	软元件输出: DOUT	169
	软元件输入: DIN	170
	位软元件输出: OUT	171
4.12	逻辑运算	172
	逻辑肯定: (无)	172
	逻辑否定: !	173
	逻辑积: *	174
	逻辑和: +	175
4.13	比较运算	176
	一致: ==	176
	不一致: !=	177
	小于: <	178
	小于等于: <=	179
	大于: >	180
	大于等于: >=	181
4.14	程序控制	182
	条件分支控制: IF~ELSE~IEND	182
	选择分支控制: SELECT~CASE~SEND	184
	次数指定重复控制: FOR~NEXT	186
	重复控制的强制结束: BREAK	188
4.15	运动专用函数	189
	速度更改请求: CHGV	189
	指令生成轴速度更改请求: CHGVS	193
	转矩限制值更改请求: CHGT	196
	目标位置更改请求: CHGP	198
	机器程序运行启动请求: MCNST	205
4.16	高级同步控制专用函数	207
	凸轮数据读取: CAMRD	207
	凸轮数据写入: CAMWR	211
	凸轮自动生成功能: CAMMK	215
	凸轮位置计算: CAMPSCL	226
4.17	视觉系统专用函数	229
	线路打开: MVOPEN	229
	视觉程序装载: MVLOAD	230
	触发发行: MVTRG	231

视觉程序启动: MVPST	232
数据导入: MVIN	233
数据导出: MVOU	235
状态存储软元件复位: MVFIN	237
线路关闭: MVCLOSE	238
任意原生模式指令发送: MVCOM	239
4.18 附加专用函数	242
附加组件调用: MCFUN	242
4.19 其它指令	244
事件任务允许: EI	244
事件任务禁止: DI	245
无处理: NOP	246
块传送: BMOV	247
同一数据块传送: FMOV	249
至缓冲存储器的字数据写入: TO	251
从缓冲存储器中的字数据读取: FROM	253
至对象站缓冲存储器的字数据写入: RTO	255
从对象站缓冲存储器中的字数据读取: RFROM	258
时间等待: TIME	261
4.20 注释文: //	262
第5章 转换程序	263
5.1 转换程序	263
第6章 运动SFC的动作及参数	265
6.1 任务的类型	265
6.2 连续转移数及任务动作	265
连续转移数	265
任务动作	266
6.3 多个任务的执行状态	269
6.4 运动SFC程序的启动方法	271
自动启动	271
通过运动SFC程序的启动	271
通过其它机号的运动专用顺控程序指令进行启动(顺控程序指令: M(P).SFCS/D(P).SFCS)	271
6.5 运动SFC程序的结束方法	272
6.6 运动SFC程序的切换方法	272
6.7 多CPU系统电源断开时、复位时的动作	272
6.8 任务参数	273
6.9 程序参数	275
第7章 运动SFC的功能	279
7.1 运动SFC程序的运行中写入	279
运行中写入操作方法	280
程序的读取/写入	283
7.2 运动SFC程序的监视及调试模式	285
运动SFC程序的监视	285
调试模式	285
附录	286
附1 处理时间	286

运算控制·转换指令处理时间	286
高级同步控制专用函数处理时间	309
凸轮数据展开时间	311
运动专用顺控程序指令处理时间	312
附2 样本程序	313
通过运动SFC程序进行的运动控制示例	313
通过运动SFC程序进行子程序重启时的继续执行示例	319
通过运动SFC程序暂停后的继续执行示例	322
附3 机器出错详细代码	325
修订记录	328
质保	329
商标	330

关联手册

最新的e-Manual、EPUB及手册PDF，请向三菱电机代理店咨询。

手册名称[手册编号]	内容	提供形态
MELSEC iQ-R运动控制器编程手册 (程序设计篇) [IB-0300275CHN](本手册)	介绍了运动SFC的功能、编程及调试等有关内容。	装订产品 e-Manual EPUB PDF
MELSEC iQ-R运动控制器用户手册 [IB-0300267CHN]	介绍了运动CPU模块、SSCNETⅢ电缆及串行ABS同步编码器电缆、故障排除等有关内容。	装订产品 e-Manual EPUB PDF
MELSEC iQ-R运动控制器编程手册 (公共篇) [IB-0300273CHN]	介绍了多CPU系统配置、性能规格、通用参数、辅助/应用功能及出错列表等有关内容。	装订产品 e-Manual EPUB PDF
MELSEC iQ-R运动控制器编程手册 (定位控制篇) [IB-0300277CHN]	介绍了伺服参数、定位指令及软元件一览等有关内容。	装订产品 e-Manual EPUB PDF
MELSEC iQ-R运动控制器编程手册 (高级同步控制篇) [IB-0300269CHN]	介绍了用于进行同步控制的同步控制参数及软元件一览等有关内容。	装订产品 e-Manual EPUB PDF
MELSEC iQ-R运动控制器编程手册 (机器控制篇) [IB-0300309]	介绍了用于进行机器控制的机器控制参数、机器定位数据以及软元件一览等有关内容。	装订产品 e-Manual EPUB PDF

要点

e-Manual是可使用专用工具阅读的三菱电机FA电子书手册。

e-Manual有如下所示特点。

- 希望查找的信息可从多个手册中一次查找(手册横向查找)
- 通过手册内的链接可以参照其它手册
- 通过产品插图的各部件可以阅读希望了解的硬件规格
- 可以对频繁参照的信息进行收藏登录

术语

本手册中除了特别标明的情况外，将使用下述术语进行说明。

术语	内容
R64MTCPU/R32MTCPU/R16MTCPU 或运动CPU(模块)	MELSEC iQ-R系列运动控制器的略称
MR-J4(W)-□B	MR-J4-□B/MR-J4W-□B型伺服放大器
MR-J3(W)-□B	MR-J3-□B/MR-J3W-□B型伺服放大器
AMP或伺服放大器	MR-J4-□B/MR-J4W-□B/MR-J3-□B/MR-J3W-□B型伺服放大器系列的总称
RnCPU或可编程控制器CPU	MELSEC iQ-R系列CPU模块的略称
多CPU系统 或运动系统	R系列可编程控制器多CPU系统的略称
CPU _n	多CPU系统中n号机的CPU模块(n=1~4)的略称
设备OS软件	SW10DNC-RMTFW的总称
工程软件包	MT Developer2/GX Works3的总称
MELSOFT MT Works2	运动控制器工程软件SW10DNC-MTW2的总称产品名
MT Developer2	运动控制器工程软件“MELSOFT MT Works2”中包含的编程软件的略称
GX Works3	MELSEC可编程控制器软件包SW10DNC-GXW3的总称产品名
手动脉冲器	手动脉冲发生器的略称
串行ABS同步编码器 或Q171ENC-W8	串行ABS同步编码器(Q171ENC-W8)的略称
SSCNETⅢ/H*1	运动控制器 ↔ 伺服放大器之间高速同步网络
SSCNETⅢ*1	
SSCNETⅢ(/H)	SSCNETⅢ/H、SSCNETⅢ的总称
绝对位置系统	使用了支持绝对位置的伺服电机及伺服放大器的系统的总称
智能功能模块	A/D、D/A转换模块等具有输入输出以外功能的模块的总称
SSCNETⅢ/H起始模块*1	MELSEC-L系列SSCNETⅢ/H起始模块(LJ72MS15)的略称
光分支模块或MR-MV200	SSCNETⅢ/H对应光分支模块(MR-MV200)的略称

*1 SSCNET: Servo System Controller NETwork

手册的阅读方法

关于本手册中使用的数值的表示

n 关于轴No. 的表示

在定位专用信号的介绍中，M3200+20n等的n表示下表的连续轴No. 对应的数值。

轴No.	n	轴No.	n	轴No.	n	轴No.	n	轴No.	n	轴No.	n	轴No.	n	轴No.	n
1	0	9	8	17	16	25	24	33	32	41	40	49	48	57	56
2	1	10	9	18	17	26	25	34	33	42	41	50	49	58	57
3	2	11	10	19	18	27	26	35	34	43	42	51	50	59	58
4	3	12	11	20	19	28	27	36	35	44	43	52	51	60	59
5	4	13	12	21	20	29	28	37	36	45	44	53	52	61	60
6	5	14	13	22	21	30	29	38	37	46	45	54	53	62	61
7	6	15	14	23	22	31	30	39	38	47	46	55	54	63	62
8	7	16	15	24	23	32	31	40	39	48	47	56	55	64	63

- 在R16MTCPU中轴No. 1~16的范围(n=0~15)有效，在R32MTCPU中轴No. 1~32的范围(n=0~31)有效。
- 各轴对应的软元件No. 应按以下方式进行计算。

例

Q兼容配置方式中轴No. 32的情况下

M3200+20n ([Rq. 1140] 停止指令)=M3200+20×31=M3820

M3215+20n ([Rq. 1155] 伺服OFF指令)=M3215+20×31=M3835

但是，M10440+10n等同步编码器轴状态、同步编码器轴指令信号、同步编码器轴监视软元件、同步编码器轴控制软元件的n表示下表的连续同步编码器轴No. 对应的数值。

同步编码器轴No.	n	同步编码器轴No.	n	同步编码器轴No.	n
1	0	5	4	9	8
2	1	6	5	10	9
3	2	7	6	11	10
4	3	8	7	12	11

- 各同步编码器对应的软元件No. 应按以下方式进行计算。

例

Q兼容配置方式中同步编码器轴No. 12的情况下

M10440+10n ([St. 320] 同步编码器轴设置有效标志)=M10440+10×11=M10550

D13240+20n ([Md. 320] 同步编码器轴当前值)=D13240+20×11=D13460

n 关于机器No. 的表示

在定位专用信号的说明中，M43904+32m等的m表示下表所示机器No. 对应的数值。

机器No.	m	机器No.	m
1	0	5	4
2	1	6	5
3	2	7	6
4	3	8	7

- 各机器对应的软元件No. 应按以下方式进行计算。

例

R标准配置方式中机器No. 8的情况下

M43904+32m ([St. 2120] 机器出错检测)=M43904+32×7=M44128

D53168+128m ([Md. 2020] 机器类型)=D53168+128×7=D54064

关于本手册中使用的软元件编号的表示

对于定位专用信号的软元件编号中记载的“[Rq. 1140]停止指令(R: M34480+32n/Q: M3200+20n)”等的R及Q，如下所示，表示所使用的软元件配置方式的软元件编号。未记载R及Q的情况下，在软元件配置方式中将变为通用的软元件编号。

符号	软元件配置方式
R	R标准配置方式
Q	Q兼容配置方式

1 概要

1.1 性能规格

运动SFC性能规格

项目		R64MTCPU/R32MTCPU/R16MTCPU		
运动SFC程序容量	代码合计 (SFC图+运算控制+转换)	4096k字节		
运动SFC程序	SFC程序数	256(No. 0~255)		
	SFC图容量/1程序	最大64k字节(包含SFC图注释)		
	SFC步数/1程序	最大4094步		
	选择分支数/1分支	255		
	并联分支数/1分支	255		
	并联分支的嵌套	最大 4重		
运算控制程序(F/ FS)・转换程序 (G)	运算控制程序数	F(1次执行型)/FS(扫描执行型)合计4096(F/FS0~F/FS4095)		
	转换程序数	4096(G0~G4095)		
	代码容量/1程序	最大约128k字节(65534步)		
	块(行)数/1程序	最大8192块(8步(最小)/1块的情况下)		
	字符数/1块(行)	最大半角1020字符(包括注释)		
	被运算符数/1块	最大510个(被运算符: 常数・字软元件・位软元件)		
	()的嵌套/1块	最大32重		
	记述式	运算控制程序 转换程序	计算式・位条件式・分支/重复处理 计算式・位条件式・比较条件式	
执行规格	同时执行程序数	最大256个		
	同时激活步数	最大256步/全部程序		
	执行任务	普通任务	通过运动CPU的主周期执行	
		事件任务(可屏蔽)	恒定周期	在各恒定周期(0.222ms・0.444ms・0.888ms・1.777ms・3.555ms・7.111ms・14.222ms)执行
			外部中断	运动CPU管理的输入模块的输入16点内, 通过事件任务原因中设置的输入的ON执行
可编程控制器中断	通过来自于可编程控制器的中断指令(D(P).GINT/M(P).GINT)执行			
NMI任务	运动CPU管理的输入模块的输入16点内, 通过NMI任务原因中设置的输入的ON执行			

运算控制 · 转换控制规格

运算控制 · 转换控制规格一览

n 公式

规格		备注
计算式	返回数值的结果。 进行常数、字软元件的间接指定数据的计算。	D100+1, SIN(D100)等
条件式	位条件式	返回真假的结果。 进行位软元件ON/OFF判定。
	比较条件式	返回真假的结果。 进行常数、字软元件的间接指定数据、计算式的比较。

n 位软元件

○：可以；×：不能

软元件	符号	可以访问		可以使用任务			记述示例
		Read	Write	普通	事件	NMI	
输入	X	○	○	○	○	○	X100
输出	Y	○	○				Y100
内部继电器	M	○	○				M20
链接继电器	B	○	○				B3FF
报警器	F	○	○				F0
数据寄存器	D	○	○				D0.A
链接寄存器	W	○	○				W1F.A
运动寄存器	#	○	○				#0.A
特殊继电器	SM	○	○				SM0
特殊寄存器	SD	○	○				SD0.A
CPU缓冲存储器访问软元件	U3E□\G	○	○				U3E0\G200.A
CPU缓冲存储器访问软元件(恒定周期通信区域)	U3E□\HG	○	○				U3E0\HG200.A
模块访问软元件	U□\G	○	○				U0\G10200.A

限制事项

可写入(Write)的位软元件的限制

- 对于软元件X的写入，只有实际输入的软元件以外才可以进行。
- 特殊寄存器、特殊继电器在系统中的用途是确定的。对于用户设置软元件以外请勿进行写入。

n 字软元件

○：可以；×：不能

软元件	符号	可以访问		可以使用任务			记述示例
		Read	Write	普通	事件	NMI	
数据寄存器	D	○	○	○	○	○	D0L
链接寄存器	W	○	○				W1F:F
运动寄存器	#	○	○				0F
特殊寄存器	SD	○	○				SD0
CPU缓冲存储器访问软元件	U3E□\G	○	○				U3E0\G100L
CPU缓冲存储器访问软元件(恒定周期通信区域)	U3E□\HG	○	○				U3E0\HG100L
模块访问软元件	U□\G	○	○				U0\G10100L

限制事项

可写入(Write)的字软元件的限制

- 特殊继电器在系统中的用途是确定的。对于用户设置软元件以外请勿进行写入。

n 数据类型

规格			备注
(无)	16位整数型(带符号)	-32768~32767	K10, D100等
	16位整数型(无符号)	0~65535	
L	32位整数型(带符号)	-2147483648~2147483647	2000000000, W100L等
	32位整数型(无符号)	0~4294967295	
F	64位浮点型(双精度实数型)	IEEE格式	1.23, #10F等

n 常数

规格			备注
K	10进制常数	上述, 通过将数据类型符号 'L'、'.' (小数点) 附加到末尾处, 标明数据类型。省略数据类型时, 视为可处理的最小的类型。	K-100., HOFFL等 'K' 可以省略。
H	16进制常数		

n 实际输入、实际输出的Read/Write响应

规格		备注
输入响应	执行指令时直接Read控制	
输出响应	执行指令时直接Write控制	

运算控制・转换指令一览

○：可以使用；—：不能使用

分类	符号	功能	格式	基本步	可用步		Y/N转换的条件式	详细说明章节
					F/FS	G		
二项运算	=	代入	(D)=(S)	8	○	○	—	119页 代入：=
	+	加法	(S1)+(S2)	7	○	○	—	120页 加法：+
	-	减法	(S1)-(S2)	7	○	○	—	121页 减法：-
	*	乘法	(S1)*(S2)	7	○	○	—	122页 乘法：*
	/	除法	(S1)/(S2)	7	○	○	—	124页 除法：/
	%	剩余	(S1)%(S2)	7	○	○	—	125页 余数：%
位运算	~	位取反(补数)	~(S)	4	○	○	—	126页 位取反(补数)：~
	&	位逻辑积	(S1)&(S2)	7	○	○	—	127页 位逻辑积： &
		位逻辑和	(S1) (S2)	7	○	○	—	128页 位逻辑和：
	^	位异或	(S1)^(S2)	7	○	○	—	129页 位异或：^
	>>	位右移	(S1)>>(S2)	7	○	○	—	130页 位右移：>>
	<<	位左移	(S1)<<(S2)	7	○	○	—	131页 位左移：<<
符号	—	符号取反(2的补数)	-(S)	4	○	○	—	132页 符号取反(2的补数)：-
标准函数	SIN	正弦	SIN(S)	4	○	○	—	133页 正弦：SIN
	COS	余弦	COS(S)	4	○	○	—	134页 余弦：COS
	TAN	正切	TAN(S)	4	○	○	—	135页 正切：TAN
	ASIN	反正弦	ASIN(S)	4	○	○	—	136页 反正弦： ASIN
	ACOS	反余弦	ACOS(S)	4	○	○	—	137页 反余弦： ACOS
	ATAN	反正切	ATAN(S)	4	○	○	—	138页 反正切： ATAN
	SQRT	平方根	SQRT(S)	4	○	○	—	139页 平方根： SQRT
	LN	自然对数	LN(S)	4	○	○	—	140页 自然对数： LN
	EXP	指数运算	EXP(S)	4	○	○	—	141页 指数运算： EXP
	ABS	绝对值	ABS(S)	4	○	○	—	142页 绝对值： ABS
	RND	四舍五入	RND(S)	4	○	○	—	143页 四舍五入： RND
	FIX	舍去	FIX(S)	4	○	○	—	144页 舍去：FIX
	FUP	进位	FUP(S)	4	○	○	—	145页 进位：FUP
BIN	BCD→BIN转换	BIN(S)	4	○	○	—	146页 BCD → BIN 转换：BIN	
BCD	BIN→BCD转换	BCD(S)	4	○	○	—	147页 BIN → BCD 转换：BCD	

分类	符号	功能	格式	基本步	可用步		Y/N转换的条件式	详细说明章节
					F/FS	G		
数据控制	SCL	16位整数型标度	SCL (S1), (S2), (S3), (D)	15	○	○	—	☞ 148页 16位整数型标度: SCL
	DSCL	32位整数型标度	DSCL (S1), (S2), (S3), (D)	15	○	○	—	☞ 152页 32位整数型标度: DSCL
类型转换	SHORT	转换为16位整数型(带符号)	SHORT (S)	4	○	○	—	☞ 155页 带符号16位整数值转换: SHORT
	USHORT	转换为16位整数型(无符号)	USHORT (S)	4	○	○	—	☞ 156页 无符号16位整数值转换: USHORT
	LONG	转换为32位整数型(带符号)	LONG (S)	4	○	○	—	☞ 158页 带符号32位整数值转换: LONG
	ULONG	转换为32位整数型(无符号)	ULONG (S)	4	○	○	—	☞ 159页 无符号32位整数值转换: ULONG
	FLOAT	视为带符号数据, 转换为64位浮点型。	FLOAT (S)	4	○	○	—	☞ 161页 带符号64位浮点值转换: FLOAT
	UFLOAT	视为无符号数据, 转换为64位浮点型。	UFLOAT (S)	4	○	○	—	☞ 162页 无符号64位浮点值转换: UFLOAT
	DFLT	浮点值的32位→64位转换	DFLT (S)	4	○	○	—	☞ 163页 浮点值的32位 → 64位转换: DFLT
	SFLT	浮点值的64位→32位转换	SFLT (S)	4	○	○	—	☞ 164页 浮点值的64位 → 32位转换: SFLT
位软元件状态	无	ON(A触点)	(S)	4	○	○	○	☞ 165页 ON(常开触点): (无)
	!	OFF(B触点)	!(S)	4	○	○	○	☞ 166页 OFF(常闭触点): !
位软元件控制	SET	软元件设置	SET (D)	5	○	○	—	☞ 167页 软元件设置: SET
			SET (D)=(条件式)	8	○	○	—	
	RST	软元件复位	RST (D)	5	○	○	—	☞ 168页 软元件复位: RST
			RST (D)=(条件式)	8	○	○	—	
	DOUT	软元件输出	DOUT (D), (S)	8	○	○	—	☞ 169页 软元件输出: DOUT
	DIN	软元件输入	DIN (D), (S)	8	○	○	—	☞ 170页 软元件输入: DIN
OUT	位软元件输出	OUT (D)=(条件式)	8	○	○	—	☞ 171页 位软元件输出: OUT	
逻辑运算	无	逻辑肯定	(条件式)	0	○	○	○	☞ 172页 逻辑肯定: (无)
	!	逻辑否定	!(条件式)	4	○	○	○	☞ 173页 逻辑否定: !
	*	逻辑积	(条件式)*(条件式)	7	○	○	○	☞ 174页 逻辑积: *
	+	逻辑和	(条件式)+(条件式)	7	○	○	○	☞ 175页 逻辑和: +

分类	符号	功能	格式	基本步	可用步		Y/N转换的条件式	详细说明章节
					F/FS	G		
比较运算	==	一致	(计算式)==(计算式)	7	○	○	○	☞ 176页 一致: ==
	!=	不一致	(计算式)!=(计算式)	7	○	○	○	☞ 177页 不一致: !=
	<	小于	(计算式)<(计算式)	7	○	○	○	☞ 178页 小于: <
	<=	小于等于	(计算式)<=(计算式)	7	○	○	○	☞ 179页 小于等于: <=
	>	大于	(计算式)>(计算式)	7	○	○	○	☞ 180页 大于: >
	>=	大于等于	(计算式)>=(计算式)	7	○	○	○	☞ 181页 大于等于: >=
程序控制	IF~ELSE ~IEND	条件分支控制	IF (S) : ELSE : IEND	IF: 8 ELSE: 5 IEND: 1	○	○	—	☞ 182页 条件分支控制: IF~ELSE~IEND
	SELECT~ CASE~ SEND	选择分支控制	SELECT CASE (S1) : CEND CASE (Sn) : CEND CLELSE : CEND SEND	SELECT: 1 CASE: 8 CEND: 5 CLELSE: 1 SEND: 1	○	○	—	☞ 184页 选择分支控制: SELECT~CASE~SEND
	FOR~NEXT	次数指定重复控制	FOR (D)=(S1) TO (S2) STEP (S3) : NEXT	FOR: 18 NEXT: 15	○	○	—	☞ 186页 次数指定重复控制: FOR~NEXT
	BREAK	重复控制的强制结束	BREAK	5	○	○	—	☞ 188页 重复控制的强制结束: BREAK
运动专用函数	CHGV	速度更改请求	CHGV ((S1), (S2))	7	○	○	—	☞ 189页 速度更改请求: CHGV
	CHGVS	指令生成轴速度更改请求	CHGVS ((S1), (S2))	7	○	○	—	☞ 193页 指令生成轴速度更改请求: CHGVS
	CHGT	转矩限制值更改请求	CHGT ((S1), (S2), (S3))	10	○	○	—	☞ 196页 转矩限制值更改请求: CHGT
	CHGP	目标位置更改请求	CHGP ((S1), (S2), (S3))	11	○	○	—	☞ 198页 目标位置更改请求: CHGP
	MCNST	机器程序运行启动请求	MCNST ((S1), (S2))	7	○	○	—	☞ 205页 机器程序运行启动请求: MCNST
高级同步控制 专用函数	CAMRD	凸轮数据读取	CAMRD (S1), (S2), (n), (D), (S3)	16	○	○	—	☞ 207页 凸轮数据读取: CAMRD
	CAMWR	凸轮数据写入	CAMWR (S1), (S2), (n), (S3), (D)	16	○	○	—	☞ 211页 凸轮数据写入: CAMWR
	CAMMK	凸轮自动生成功能	CAMMK (S1), (S2), (S3), (D)	13	○	○	—	☞ 215页 凸轮自动生成功能: CAMMK
	CAMPSCL	凸轮位置计算	CAMPSCL (S1), (S2), (D)	12	○	○	—	☞ 226页 凸轮位置计算: CAMPSCL

分类	符号	功能	格式	基本步	可用步		Y/N转换的条件式	详细说明章节
					F/FS	G		
视觉系统专用函数	MVOPEN	线路打开	MVOPEN (S1), (S2)	8	○	○	—	☞ 229页 线路打开: MVOPEN
	MVLOAD	视觉程序装载	MVLOAD (S1), (S2)	8	○	○	—	☞ 230页 视觉程序装载: MVLOAD
	MVTRG	触发行	MVTRG (S1), (S2)	8	○	○	—	☞ 231页 触发行: MVTRG
	MVPST	视觉程序启动	MVPST (S1), (S2)	8	○	○	—	☞ 232页 视觉程序启动: MVPST
	MVIN	数据导入	MVIN (S1), (S2), (D), (S3)	15以上	○	○	—	☞ 233页 数据导入: MVIN
	MVOUT	数据导出	MVOUT (S1), (S2), (S3), (S4)	15以上	○	○	—	☞ 235页 数据导出: MVOUT
	MVFIN	状态存储软元件复位	MVFIN (S)	6	○	○	—	☞ 237页 状态存储软元件复位: MVFIN
	MVCLOSE	线路关闭	MVCLOSE (S)	6	○	○	—	☞ 238页 线路关闭: MVCLOSE
	MVCOM	任意原生模式指令发送	MVCOM (S1), (S2), (D), (S3), (S4)	19以上	○	○	—	☞ 239页 任意原生模式指令发送: MVCOM
附加专用函数	MCFUN	附加组件调用	MCFUN (S1), (S2), (D1), (D2)	11以上	○	○	—	☞ 242页 附加组件调用: MCFUN
其它	EI	事件任务允许	EI	1	○	○	—	☞ 244页 事件任务允许: EI
	DI	事件任务禁止	DI	1	○	○	—	☞ 245页 事件任务禁止: DI
	NOP	无处理	NOP	1	○	○	—	☞ 246页 无处理: NOP
	BMOV	块传送	BMOV (D), (S), (n)	12	○	○	—	☞ 247页 块传送: BMOV
	FMOV	同一数据块传送	FMOV (D), (S), (n)	12	○	○	—	☞ 249页 同一数据块传送: FMOV
	TO	缓冲存储器的字数据写入	TO (D1), (D2), (S), (n)	14	○	○	—	☞ 251页 至缓冲存储器的字数据写入: TO
	FROM	缓冲存储器的字数据读取	FROM (D), (S1), (S2), (n)	14	○	○	—	☞ 253页 从缓冲存储器中的字数据读取: FROM
	RTO	对象站缓冲存储器的字数据写入	RTO (D1), (D2), (D3), (S), (n), (D4)	21	○	○	—	☞ 255页 至对象站缓冲存储器的字数据写入: RTO
	RFROM	对象站缓冲存储器的字数据读取	RFROM (D), (S1), (S2), (S3), (n), (D1)	21	○	○	—	☞ 258页 从对象站缓冲存储器中的字数据读取: RFROM
	TIME	时间等待	TIME (S)	8	—	○	—	☞ 261页 时间等待: TIME

运算控制程序・转换程序的1个程序代码容量 概算式

$2 + (2 + 1 \text{块的基本步数合计} + 32 \text{位常数个数} / 1 \text{块} \times 1 + 64 \text{位常数个数} / 1 \text{块} \times 3) \times \text{块数} (\text{步})$
(1步=2字节)

1.2 运动CPU的程序构成

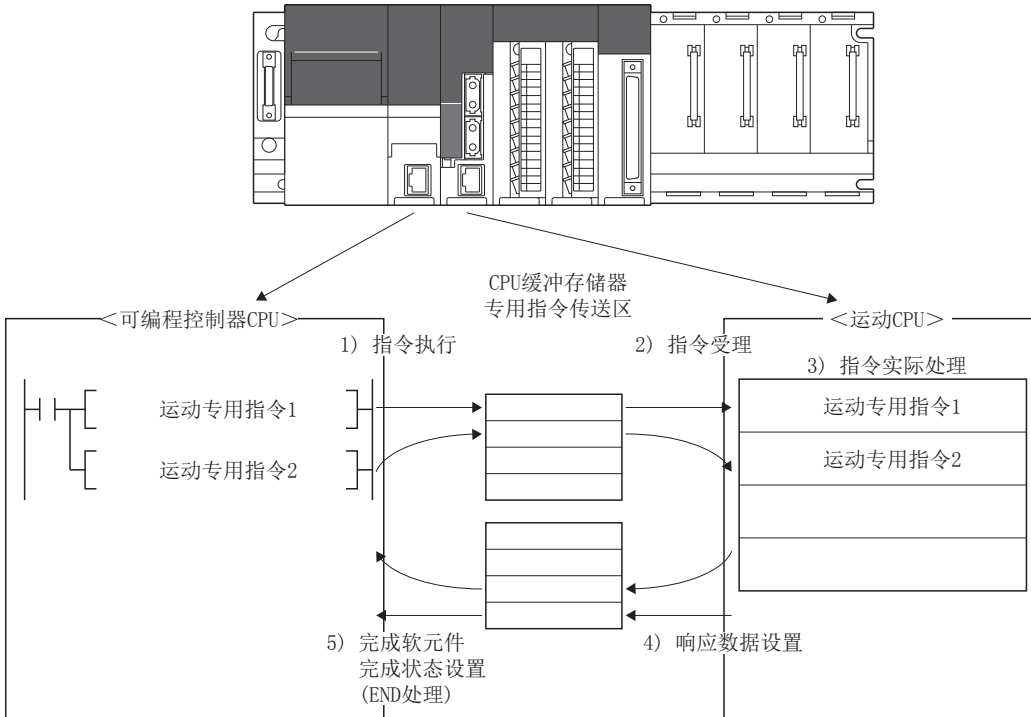
- 通过在可编程控制器CPU的顺控程序中使用运动专用顺控程序指令，通过运动CPU进行以下控制。
 - 运动SFC程序的启动
 - 伺服程序的启动
 - 直接定位(通过顺控程序进行定位控制)
 - 当前值/速度/转矩限制值的更改
 - 事件任务的启动
- 对于运动CPU的程序，以流程图形式的运动SFC进行记述。在定位控制中，将伺服电机的运动控制通过运动SFC程序中的运动控制步指定的伺服程序进行控制。
- 在高级同步控制中，通过设置同步控制用参数，对各输出轴启动同步控制，进行输入轴(伺服输入轴、指令生成轴、同步编码器轴)同步的控制。
- 在机器控制中，根据所使用的机器类型设置机器控制参数，使用软元件中设置的机器定位数据，进行机器的控制。
- 关于可编程控制器CPU中的运动专用顺控程序指令、运动SFC程序、定位控制中的运动控制、高级同步控制中的运动控制、机器控制中的运动控制的详细情况，请参阅以下内容。

项目	参照
可编程控制器CPU中的运动专用顺控程序指令	☞ 27页 运动专用顺控程序指令
运动SFC程序	☞ 83页 运动SFC程序
定位控制中的运动控制(伺服程序)	☞ MELSEC iQ-R运动控制器编程手册(定位控制篇)
高级同步控制中的运动控制(同步控制参数)	☞ MELSEC iQ-R运动控制器编程手册(高级同步控制篇)
机器控制中的运动控制(机器控制参数)	☞ MELSEC iQ-R运动控制器编程手册(机器控制篇)

2 运动专用顺控程序指令

2.1 运动专用顺控程序指令的概要

通过运动专用顺控程序指令，可以进行从可编程控制器CPU等其它机号CPU访问运动CPU的软元件数据访问及程序启动等。运动专用顺控程序指令通过CPU缓冲存储器或CPU缓冲存储器(恒定周期通信区域)中的系统区域中设置的CPU之间专用指令传送区域进行传送。运动专用顺控程序指令的动作概要如下图所示。



指令使用的存储器区域及运动CPU侧受理指令的周期如下所示。


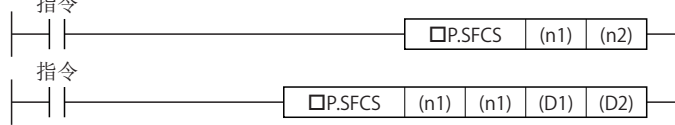

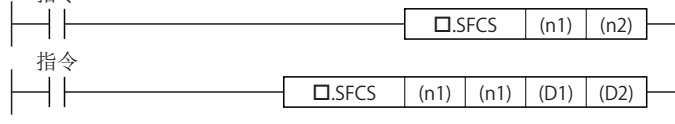
指令	使用存储器	运动CPU侧受理周期
M(P).□指令	CPU缓冲存储器	非恒定周期(即时)
D(P).□指令	CPU缓冲存储器(恒定周期通信区域)	恒定周期(CPU之间通信周期)

2.2 运动专用顺控程序指令

可对运动CPU执行的运动专用顺控程序指令的一览如下表所示。

指令		内容	参阅目标
M(P). □	D(P). □		
M(P). SFCS	D(P). SFCS	指定的运动SFC程序的启动请求	☞ 29页 至运动CPU的运动SFC启动请求： M(P). SFCS/D(P). SFCS
M(P). SVST	D(P). SVST	指定的伺服程序的启动请求	☞ 32页 至运动CPU的伺服程序启动请求： M(P). SVST/D(P). SVST
M(P). SVSTD	D(P). SVSTD	直接定位启动请求	☞ 37页 至运动CPU的直接定位启动指令： M(P). SVSTD/D(P). SVSTD
M(P). CHGA	D(P). CHGA	指定轴的当前值更改请求	☞ 47页 至运动CPU的当前值更改指令：M(P). CHGA/ D(P). CHGA
M(P). CHGAS	D(P). CHGAS	指定的指令生成轴的当前值更改请求	☞ 51页 至运动CPU的指令生成轴的当前值更改指令： M(P). CHGAS/D(P). CHGAS
M(P). CHGV	D(P). CHGV	指定轴的速度更改请求	☞ 55页 至运动CPU的速度更改指令：M(P). CHGV/ D(P). CHGV
M(P). CHGVS	D(P). CHGVS	指定的指令生成轴的速度更改请求	☞ 59页 至运动CPU的指令生成轴的速度更改指令： M(P). CHGVS/D(P). CHGVS
M(P). CHGT	D(P). CHGT	指定轴的转矩限制值更改请求	☞ 64页 至运动CPU的转矩限制值更改指令： M(P). CHGT/D(P). CHGT
M(P). DDWR	D(P). DDWR	将本机CPU的软件数据写入到其它机号运动CPU的软件	☞ MELSEC iQ-R编程手册(指令/通用FUN/FB篇)
M(P). DDRD	D(P). DDRD	将其它机号运动CPU的软件数据读取到本机CPU的软件	
M(P). BITWR	D(P). BITWR	将位操作写入到其它机号运动CPU的位软件	☞ 67页 至运动CPU的位软件的写入： M(P). BITWR/D(P). BITWR
M(P). GINT	D(P). GINT	其它机号运动CPU的事件任务执行请求	☞ 70页 至其它机号CPU的中断指令：M(P). GINT/ D(P). GINT

至运动CPU的运动SFC启动请求：M(P).SFCS/D(P).SFCS

指令符号	执行条件	顺控程序*1
MP.SFCS DP.SFCS		
M.SFCS D.SFCS		

*1 □=指令符号(M(P).SFCS: M, D(P).SFCS: D)

设置数据

n 可用软元件

○：可以设置；△：部分可以设置

设置数据*3	可用软元件											
	内部软元件 (系统、用户)		文件寄存器		链接直接软元件 J□\□		模块访问软元件 U□\G□		变址 寄存器 Z□	常数		其它
	位	字	位	字	位	字	位	字		10进制 K、16进 制H	实数字 符串	
(n1)		○		○						○		
(n2)		○		○						○		
(D1)*1		△*2		△*2								
(D2)*1		△*2		△*2								

*1 (D1)、(D2) 两方省略的情况下可以省略。

*2 不能使用局部软元件。

*3 对设置数据(n1)~(D2)可进行变址修饰。(常数除外)

n 内容、范围、设置方、数据类型

设置数据	设置内容	设置范围	设置方	数据类型
(n1)	对象机号CPU的起始输入输出编号÷16 实际指定的值如下所示。*2 2号机：3E1H；3号机：3E2H；4号机：3E3H	3E1H~3E3H	用户	BIN16位
(n2)	启动的运动SFC程序No.	0~255	用户	BIN16位
(D1)*1	完成软元件 • (D1+0)：通过指令的受理处理完成使其1个扫描ON的软元件 (D1+1)：通过指令的受理异常完成使其1个扫描ON的软元件(异常完成时，D1+0也变为ON)	—	系统	位
(D2)*1	存储完成状态的软元件	—	系统	字

*1 (D1)、(D2) 两方省略的情况下可以省略。

*2 在多CPU构成中，不能将运动CPU设置为1号机。

功能

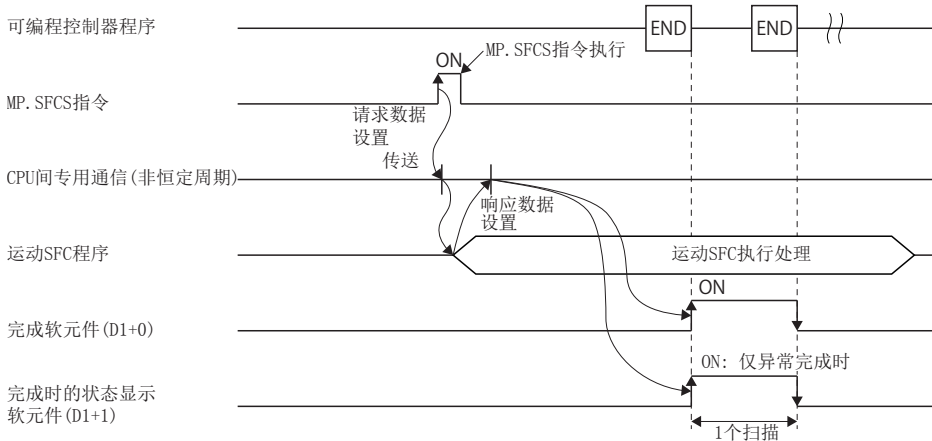
n 控制内容

进行(n2)中指定的程序No.的运动SFC程序的启动请求。启动的运动SFC程序可以进行普通任务执行、事件任务执行、NMI任务执行的设置。

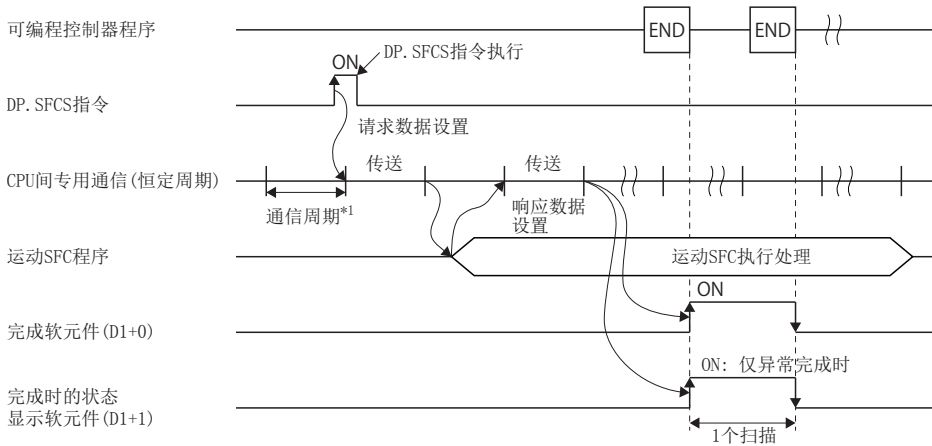
n 动作时机

执行了MP、SFCS指令、DP、SFCS指令时CPU之间的动作概要如下所示。

• MP、SFCS指令



• DP、SFCS指令



*1 在GX Works3的[系统参数]⇒[多CPU设置]中设置

出错

• 以下情况下将异常结束，完成状态存储软件(D2)中指定的软件中将存储出错代码。省略了完成状态存储软件(D2)的情况下将不进行出错检测而变为无处理，应加以注意。

完成状态*1 (出错代码) (H)	出错原因	处理方法
0010	从可编程控制器CPU至运动CPU的指令请求超出了允许值。	确认程序，修改为正确的顺控程序。
2100	同时通过M(P)、SFCS/D(P)、SFCS进行的从可编程控制器CPU至运动CPU的指令请求超过了65个指令，运动CPU无法处理。	
2200	启动的运动SFC程序No. 超出了0~255的范围。	

*1 0000H(正常)

• 以下情况下将运算出错，“最新自诊断出错(SM0)”将ON，出错代码将被存储到“最新自诊断出错代码(SD0)”中。

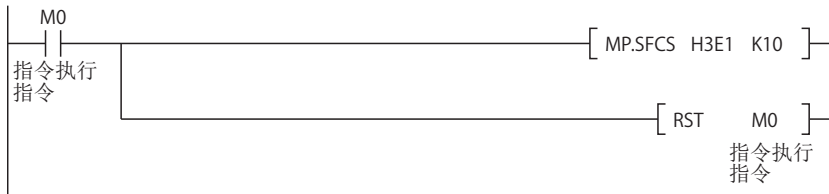
出错代码(H)*1	出错原因	处理方法
2800	指定的其它机号CPU模块的起始输入输出编号(以16进制数4位表示时的前3位)超出了3E0H~3E3H的范围时。	确认程序，修改为正确的顺控程序。
2801	指定的其它机号CPU模块有误。 • 指定了被设置为保留的机号时。 • 指定了未安装的机号时。	

*1 0000H(正常)

程序示例

n 以下为M0变为ON时启动运动CPU(2号机)的SFC程序No. 10的程序

- (例1) 省略了完成软元件、完成状态情况下的程序



- (例2) 使用了完成软元件、完成状态情况下的程序



至运动CPU的伺服程序启动请求：M(P).SVST/D(P).SVST

指令符号	执行条件	顺控程序*1
MP.SVST DP.SVST		
M.SVST D.SVST		

*1 □=指令符号(M(P).SVST: M, D(P).SVST: D)

设置数据

n 可用软元件

○：可以设置；△：部分可以设置

设置数据*3	可用软元件											
	内部软元件 (系统、用户)		文件寄存器		链接直接软元件 J□\□		模块访问软元件 U□\G□		变址 寄存器 Z□	常数		其它
	位	字	位	字	位	字	位	字		10进制 K、16进 制H	实数字 字符串	
(n1)		○		○						○		
(S1)		○		○							○	
(n2)		○		○						○		
(D1)*1	△*2		△*2									
(D2)*1		△*2		△*2								

*1 (D1)、(D2) 两方省略的情况下可以省略。

*2 不能使用局部软元件。

*3 对设置数据(n1)~(D2)可进行变址修饰。(常数除外)

n 内容、范围、设置方、数据类型

设置数据	设置内容	设置范围	设置方	数据类型
(n1)	对象机号CPU的起始输入输出编号+16 实际指定的值如下所示。*2 2号机：3E1H；3号机：3E2H；4号机：3E3H	3E1H~3E3H	用户	BIN16位
(S1)	启动的轴No. (“Jn”)*3 R64MTCPU: J1~J64 R32MTCPU: J1~J32 R16MTCPU: J1~J16	1~64	用户	字符串
(n2)	执行的伺服程序No.	0~4095	用户	BIN16位
(D1)*1	完成软元件 • (D1+0)：通过指令的受理处理完成使其1个扫描ON的软元件 • (D1+1)：通过指令的受理异常完成使其1个扫描ON的软元件(异常完成时，D1+0也变为ON)	—	系统	位
(D2)*1	存储完成状态的软元件	—	系统	字

*1 (D1)、(D2) 两方省略的情况下可以省略。

*2 在多CPU构成中，不能将运动CPU设置为1号机。

*3 “n”表示对应于轴No.的数值。(n=1~64)

功能

n 控制内容

- 进行(n2)中指定的程序No. 的伺服程序的启动请求。
- 为了避免对同一机号运动CPU的同一轴执行多个指令，需要通过CPU缓冲存储器上的启动受理标志及用户软元件置入互锁条件。

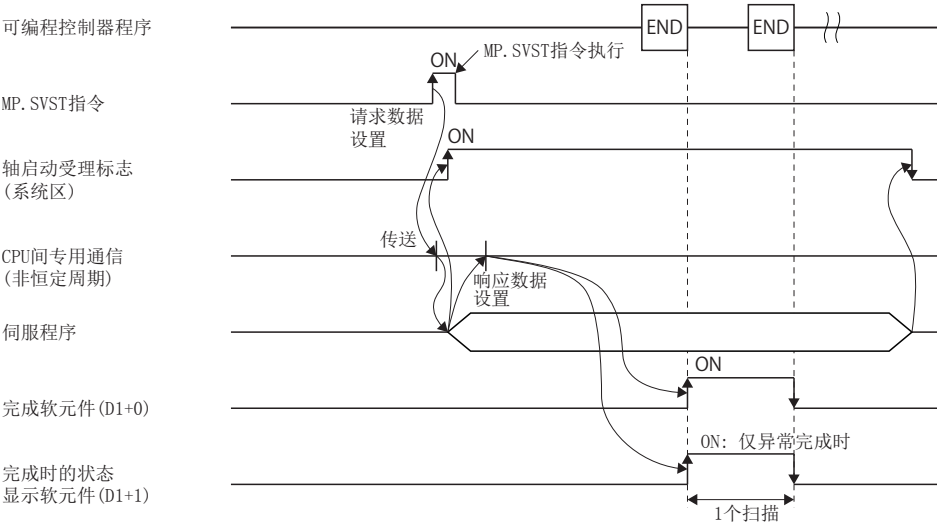
要点

关于启动受理标志的详细内容，请参阅启动受理标志(系统区域)。(P.73页 启动受理标志(系统区域))

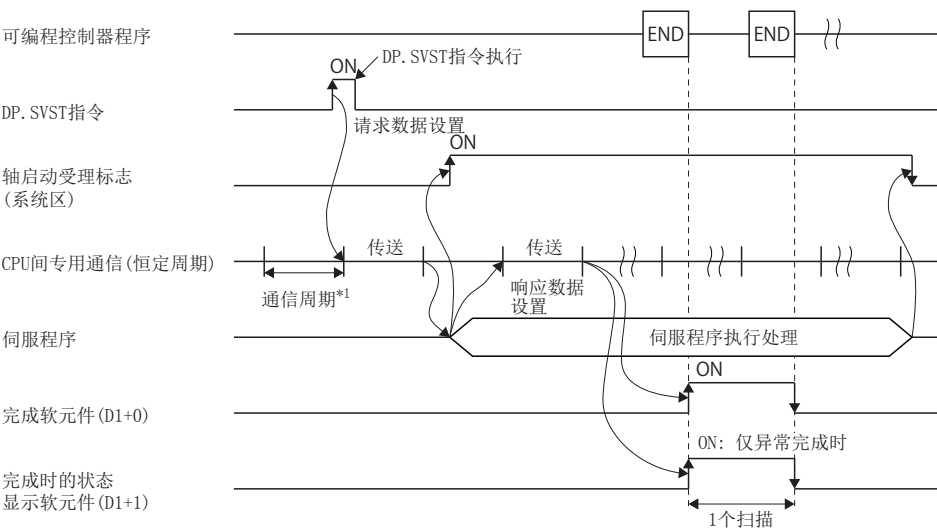
n 动作时机

执行了MP. SVST、DP. SVST指令时CPU之间的动作概要如下所示。

• MP. SVST指令



• DP. SVST指令



*1 在GX Works3的[系统参数]⇒[多CPU设置]中设置

n 启动轴的设置

对于(S1)中设置的启动轴，将J+轴No. 以字符串“ ”进行设置。

运动CPU	(S1)设置范围
R64MTCPU	1~64
R32MTCPU	1~32
R16MTCPU	1~16

最多可以设置12轴，设置多个轴的情况下，按下述示例进行设置。J设置为大写字母或小写字母，启动的轴No. 使用伺服网络设置中设置的轴No.。关于伺服网络设置，请参阅以下手册。

📖 MELSEC iQ-R运动控制器编程手册(公共篇)

此外，启动的轴编号无需为升序。

例

设置多个轴(轴1、轴2、轴10、轴11)的情况下

- “J1J2J10J11”
- “j1j2j10j11”
- “J1 J2 J10 J11”

对(S1)中指定的启动轴与启动对象伺服程序内指定的轴No. 进行校验，指定轴一致时，启动伺服程序。指定轴不一致的情况下，将变为轻度出错(出错代码：19FBH)而不启动。(S1)中指定了空白字符串“ ”的情况下，将不进行轴No. 的校验，启动伺服程序中指定的轴。伺服程序中间接指定轴No. 的情况下，应在(S1)中指定“ ”。

要点

[(S1)中指定了“ ”的情况下]

省略轴No. 的情况下，启动受理标志及指令生成轴用启动受理标志变为ON的时机将迟于设置了轴No. 的情况下。

(☞ 73页 启动受理标志(系统区域))

- 设置了轴No. 的情况下：指令受理时
- 省略了轴No. 的情况下：指令分析时

[(S1)中指定轴No. 的情况下]

通过M(P).SVST/D(P).SVST启动START指令的情况下，即使未设置所有的启动轴，START指令内各伺服程序指令的启动轴内，分别指定了1个以上轴No. 的情况下将不变为出错而进行启动。

n 轴启动受理标志(系统区域)

对象机号CPU缓冲存储器内的启动受理标志地址中，将存储启动受理标志的完成状态。

CPU缓冲存储器地址 (10进制数)	内容																									
204H (516) 205H (517) 206H (518) 207H (519)	<p>启动受理标志按64轴存储在对应的各位中。 实际位设置如下所示。</p> <ul style="list-style-type: none"> • R64MTCPU: J1~J64 • R32MTCPU: J1~J32 • R16MTCPU: J1~J16 <p>OFF: 可以启动受理 ON: 不能启动受理</p> <table border="1"> <thead> <tr> <th></th> <th>b15</th> <th>b2</th> <th>b1</th> <th>b0</th> </tr> </thead> <tbody> <tr> <td>204H(516) 地址编码</td> <td>J16</td> <td>••••••••</td> <td>J2</td> <td>J1</td> </tr> <tr> <td>205H(517) 地址编码</td> <td>J32</td> <td>••••••••</td> <td>J18</td> <td>J17</td> </tr> <tr> <td>206H(518) 地址编码</td> <td>J48</td> <td>••••••••</td> <td>J34</td> <td>J33</td> </tr> <tr> <td>207H(519) 地址编码</td> <td>J64</td> <td>••••••••</td> <td>J50</td> <td>J49</td> </tr> </tbody> </table>		b15	b2	b1	b0	204H(516) 地址编码	J16	••••••••	J2	J1	205H(517) 地址编码	J32	••••••••	J18	J17	206H(518) 地址编码	J48	••••••••	J34	J33	207H(519) 地址编码	J64	••••••••	J50	J49
	b15	b2	b1	b0																						
204H(516) 地址编码	J16	••••••••	J2	J1																						
205H(517) 地址编码	J32	••••••••	J18	J17																						
206H(518) 地址编码	J48	••••••••	J34	J33																						
207H(519) 地址编码	J64	••••••••	J50	J49																						
20EH (526) 20FH (527) 210H (528) 211H (529)	<p>指令生成轴的启动受理标志按64轴存储在对应的各位中。 实际位设置如下所示。</p> <ul style="list-style-type: none"> • R64MTCPU: J1~J64 • R32MTCPU: J1~J32 • R16MTCPU: J1~J16 <p>OFF: 可以启动受理 ON: 不能启动受理</p> <table border="1"> <thead> <tr> <th></th> <th>b15</th> <th>b2</th> <th>b1</th> <th>b0</th> </tr> </thead> <tbody> <tr> <td>20EH(526) 地址编码</td> <td>J16</td> <td>••••••••</td> <td>J2</td> <td>J1</td> </tr> <tr> <td>20FH(527) 地址编码</td> <td>J32</td> <td>••~••••••</td> <td>J18</td> <td>J17</td> </tr> <tr> <td>210H(528) 地址编码</td> <td>J48</td> <td>••••••••</td> <td>J34</td> <td>J33</td> </tr> <tr> <td>211H(529) 地址编码</td> <td>J64</td> <td>••••~••••</td> <td>J50</td> <td>J49</td> </tr> </tbody> </table>		b15	b2	b1	b0	20EH(526) 地址编码	J16	••••••••	J2	J1	20FH(527) 地址编码	J32	••~••••••	J18	J17	210H(528) 地址编码	J48	••••••••	J34	J33	211H(529) 地址编码	J64	••••~••••	J50	J49
	b15	b2	b1	b0																						
20EH(526) 地址编码	J16	••••••••	J2	J1																						
20FH(527) 地址编码	J32	••~••••••	J18	J17																						
210H(528) 地址编码	J48	••••••••	J34	J33																						
211H(529) 地址编码	J64	••••~••••	J50	J49																						

出错

- 以下情况下将异常结束，完成状态存储软元件(D2)中指定的软元件中将存储出错代码。省略了完成状态存储软元件(D2)的情况下将不进行出错检测而变为无处理，应加以注意。

完成状态*1 (出错代码)(H)	出错原因	处理方法
0010	从可编程控制器CPU至运动CPU的指令请求超出了允许值。	确认程序，修改为正确的顺控程序。
2100	从可编程控制器CPU至运动CPU的指令请求M(P).SVST/D(P).SVST/M(P).SVSTD/D(P).SVSTD/M(P).CHGA/D(P).CHGA/M(P).CHGAS/D(P).CHGAS合计同时有257个指令以上，运动CPU无法处理。	
2201	执行的伺服程序No.超出了0~4095的范围。	
2202	M(P).SVST/D(P).SVST指令中设置的轴No.不正确。	

*1 0000H(正常)

- 以下情况下将运算出错，“最新自诊断出错(SM0)”将ON，出错代码将被存储到“最新自诊断出错代码(SD0)”中。

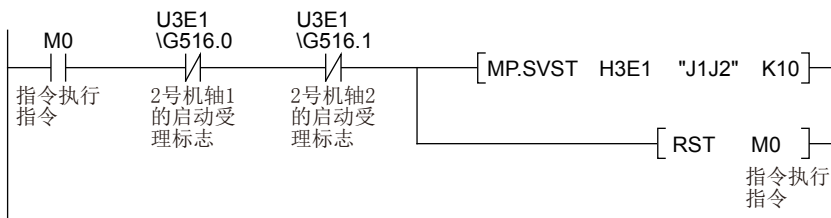
出错代码(H)*1	出错原因	处理方法
2800	指定的其它机号CPU模块的起始输入输出编号(以16进制数4位表示时的前3位)超出了3E0H~3E3H的范围时。	确认程序，修改为正确的顺控程序。
2801	指定的其它机号CPU模块有误。 <ul style="list-style-type: none"> • 指定了被设置为保留的机号时。 • 指定了未安装的机号时。 	

*1 0000H(正常)

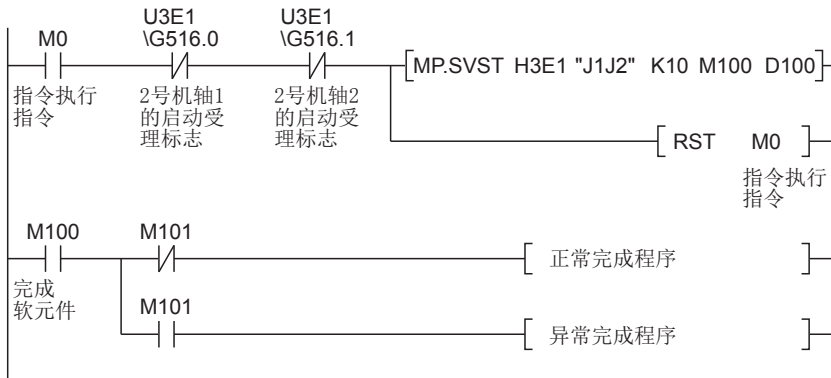
程序示例

n 以下为M0变为ON时，对运动CPU(2号机)的轴1、轴2进行伺服程序No. 10的启动请求的程序

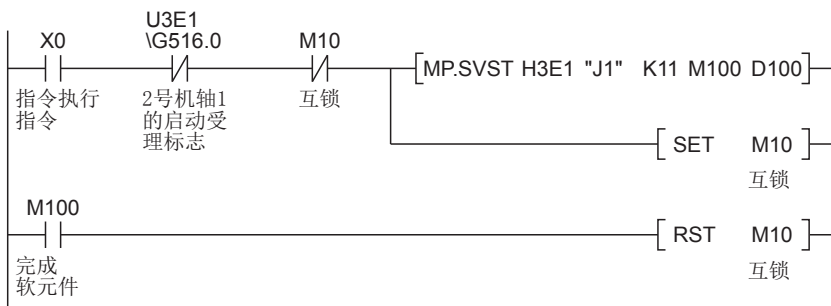
- (例1) 省略了完成软元件、完成状态情况下的程序



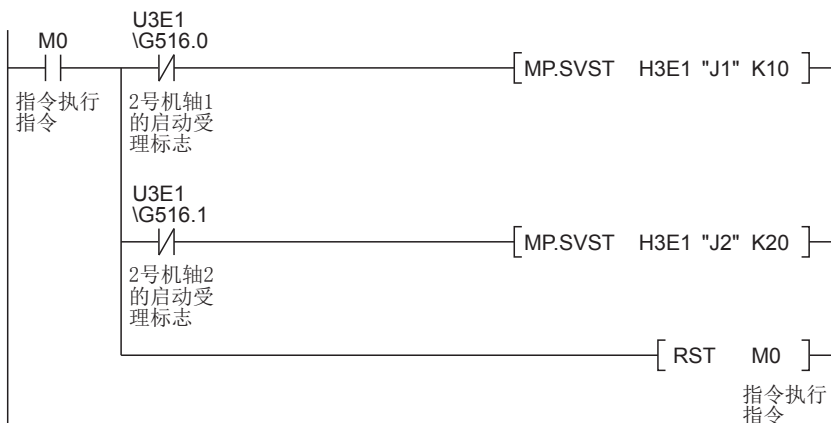
- (例2) 使用了完成软元件、完成状态情况下的程序




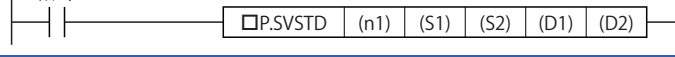

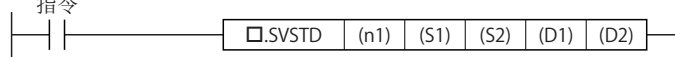
n X0为ON期间，对运动CPU(2号机)的轴1进行伺服程序No. 11的连续启动的程序



n M0变为ON时，同时对运动CPU(2号机)的轴1进行伺服程序No. 的启动请求，对轴2进行伺服程序No. 20的启动请求的的程序



至运动CPU的直接定位启动指令：M(P).SVSTD/D(P).SVSTD

指令符号	执行条件	顺控程序*1
MP.SVSTD DP.SVSTD		指令 
M.SVSTD D.SVSTD		指令 

*1 □=指令符号(M(P).SVSTD: M, D(P).SVSTD: D)

设置数据

n 可用软元件

○：可以设置；△：部分可以设置

设置数据*2	可用软元件											
	内部软元件 (系统、用户)		文件寄存器		链接直接软元件 J□\□		模块访问软元件 U□\G□		变址 寄存器 Z□	常数		其它
	位	字	位	字	位	字	位	字		10进制 K、16进 制H	实数字 字符串	
(n1)		○		○						○		
(S1)		△*1		△*1								
(S2)		○		○								
(D1)											△	
(D2)	△*1		△*1									

*1 不能使用局部软元件。

*2 对于设置数据(D1)以外，可以进行变址修饰。(常数除外)

n 内容、范围、设置方、数据类型

设置数据	设置内容	设置范围	设置方	数据类型
(n1)	对象机号CPU的起始输入输出编号+16 实际指定的值如下所示。*1 2号机：3E1H；3号机：3E2H；4号机：3E3H	3E1H~3E3H	用户	BIN16位
(S1)	存储了控制数据的本机的起始软元件	☞ 37页 控制数据	用户	字
(S2)	存储了定位数据区域的本机的起始软元件	☞ 38页 定位数据区域	用户	字
(D1)	以下对象机号运动CPU的软元件 • 固定位置停止指令中使用的对象机号运动CPU的位软元件 • 存储位置跟踪地址的对象机号运动CPU的字软元件 字符串应通过“ ”围住。 使用上述以外的情况下，应指定空白字符串“ ”	—	用户	字符串
(D2)	完成软元件 • (D2+0)：通过指令完成使其1个扫描ON的软元件 • (D2+1)：通过指令的异常完成使其1个扫描ON的软元件(异常完成时，D2+0也变为ON)	—	系统	位

*1 在多CPU构成中，不能将运动CPU设置为1号机。



n 控制数据

软元件(S1)	项目	设置数据	设置范围	设置方
+0	完成状态	存储指令完成时的状态。 0：出错無(正常完成) 0以外：出错代码	—	系统
+1	定位数据点数	将定位数据区域容量以字单位进行设置。	14~2044	用户

n 定位数据区域

软元件(S2)*1	名称		内容	设置范围	
+0	定位类型/定位点数(R)		启动连续轨迹控制(CPSTART)的情况下设置定位点数。 连续轨迹控制(CPSTART)以外时设置“0”。	0: 启动连续轨迹控制(CPSTART)以外 1~128: 启动连续轨迹控制(CPSTART)	
+1	定位用数据项目设置		设置定位启动中使用的定位用数据。以位单位设置有效/无效。	0: 无效 1: 有效 (☞ 42页 定位用数据项目设置)	
+2	定位用数据项目1		以2字设置定位用数据的设置值。	定位用数据项目设置中“1: 有效”的数据项目的范围 (☞ 42页 定位用数据项目设置)	
+3	:				
⋮	⋮				
+(2V)	定位用数据项目(V)				
+(2V+1)					
+(2V+2)	第1轴数据	轴No.	设置轴No.。	R64MTCPU: 1~64、1001~1064*2 R32MTCPU: 1~32、1001~1032*2 R16MTCPU: 1~16、1001~1016*2	
+(2V+3)		用户不能使用		应设置为0。	0
+(2V+4)		定位点1	[Da. 2]控制方式/[Da. 29]插补轴速度指定	设置控制方式及插补轴速度指定方法。	☞ 43页 定位点数据
+(2V+5)			[Da. 10]M代码	设置M代码。 对螺旋插补时的直线轴设置齿距数。	M代码: 0~32767 齿距数: 0~999
+(2V+6)			[Da. 11]停留时间	设置停留时间。	0~5000[ms]
+(2V+7)			运行中转矩限制值*3	设置各点的转矩限制值。	1~10000[×0.1%]
+(2V+8)			[Da. 8]指令速度	设置定位速度。	mm: 1~600000000×10 ⁻² [mm/min] inch: 1~600000000×10 ⁻³ [inch/min] degree: 1~2147483647×10 ⁻³ [degree/min]*4 pulse: 1~2147483647[pulse/s]
+(2V+9)					
+(2V+10)			[Da. 6]定位地址/移动量	设置定位地址/移动量。	-2147483648~2147483647 [任意单位]
+(2V+11)					
+(2V+12)			[Da. 7]圆弧地址	设置设置圆弧插补时的辅助点/中心点地址/半径。	-2147483648~2147483647 [任意单位]
+(2V+13)					
⋮		⋮		⋮	⋮
+(2V+10R-6)		定位点(R)	[Da. 2]控制方式/[Da. 29]插补轴速度指定	“定位类型/定位点数”中设置的定位点将生效。 设置了“0”的情况下,变为仅“定位点1”。	☞ 43页 定位点数据
+(2V+10R-5)			[Da. 10]M代码		M代码: 0~32767 齿距数: 0~999
+(2V+10R-4)	[Da. 11]停留时间		0~5000[ms]		
+(2V+10R-3)	运行中转矩限制值*3		1~10000[×0.1%]		
+(2V+10R-2)	[Da. 8]指令速度		mm: 1~600000000×10 ⁻² [mm/min] inch: 1~600000000×10 ⁻³ [inch/min] degree: 1~2147483647×10 ⁻³ [degree/min]*4 pulse: 1~2147483647[pulse/s]		
+(2V+10R-1)					
+(2V+10R)	[Da. 6]定位地址/移动量		-2147483648~2147483647 [任意单位]		
+(2V+10R+1)					
+(2V+10R+2)	[Da. 7]圆弧地址		-2147483648~2147483647 [任意单位]		
+(2V+10R+3)					

软元件 (S2)*1	名称	内容	设置范围	
+ (2V+10R+4)	第2轴 数据	轴No.	R64MTCPU: 1~64、1001~1064*2 R32MTCPU: 1~32、1001~1032*2 R16MTCPU: 1~16、1001~1016*2	
+ (2V+10R+5)		用户不能使用	0	
+ (2V+10R+6)		定位点1	[Da. 2]控制方式/[Da. 29] 插补轴速度指定	☞ 43页 定位点数据
+ (2V+10R+7)			[Da. 10]M代码	M代码: 0~32767 齿距数: 0~999
+ (2V+10R+8)			[Da. 11]停留时间	0~5000[ms]
+ (2V+10R+9)			运行中转矩限制值*3	1~10000[×0.1%]
+ (2V+10R+10)			[Da. 8]指令速度	mm: 1~600000000×10 ⁻² [mm/min] inch: 1~600000000×10 ⁻³ [inch/min] degree: 1~2147483647×10 ⁻³ [degree/min]*4 pulse: 1~2147483647[pulse/s]
+ (2V+10R+11)				
+ (2V+10R+12)			[Da. 6]定位地址/移动量	-2147483648~2147483647 [任意单位]
+ (2V+10R+13)			[Da. 7]圆弧地址	-2147483648~2147483647 [任意单位]
+ (2V+10R+14)				
+ (2V+10R+15)				
⋮		⋮	⋮	
+ (2V+20R-4)		第3轴 数据	定位点(R)	☞ 43页 定位点数据
+ (2V+20R-3)			[Da. 2]控制方式/[Da. 29] 插补轴速度指定	M代码: 0~32767 齿距数: 0~999
+ (2V+20R-2)	[Da. 10]M代码		0~5000[ms]	
+ (2V+20R-1)	[Da. 11]停留时间		1~10000[×0.1%]	
+ (2V+20R)	运行中转矩限制值*3		mm: 1~600000000×10 ⁻² [mm/min] inch: 1~600000000×10 ⁻³ [inch/min] degree: 1~2147483647×10 ⁻³ [degree/min]*4 pulse: 1~2147483647[pulse/s]	
+ (2V+20R+1)	[Da. 8]指令速度		-2147483648~2147483647 [任意单位]	
+ (2V+20R+2)	[Da. 6]定位地址/移动量		-2147483648~2147483647 [任意单位]	
+ (2V+20R+3)	[Da. 7]圆弧地址		-2147483648~2147483647 [任意单位]	
+ (2V+20R+4)				
+ (2V+20R+5)				
+ (2V+20R+6)	第3轴 数据	轴No.	R64MTCPU: 1~64、1001~1064*2 R32MTCPU: 1~32、1001~1032*2 R16MTCPU: 1~16、1001~1016*2	
+ (2V+20R+7)	用户不能使用	0		
+ (2V+20R+8)	定位点1	[Da. 2]控制方式/[Da. 29] 插补轴速度指定	☞ 43页 定位点数据	
+ (2V+20R+9)		[Da. 10]M代码	M代码: 0~32767 齿距数: 0~999	
+ (2V+20R+10)		[Da. 11]停留时间	0~5000[ms]	
+ (2V+20R+11)		运行中转矩限制值*3	1~10000[×0.1%]	
+ (2V+20R+12)		[Da. 8]指令速度	mm: 1~600000000×10 ⁻² [mm/min] inch: 1~600000000×10 ⁻³ [inch/min] degree: 1~2147483647×10 ⁻³ [degree/min]*4 pulse: 1~2147483647[pulse/s]	
+ (2V+20R+13)				
+ (2V+20R+14)		[Da. 6]定位地址/移动量	-2147483648~2147483647 [任意单位]	
+ (2V+20R+15)		[Da. 7]圆弧地址	-2147483648~2147483647 [任意单位]	
+ (2V+20R+16)				
+ (2V+20R+17)				
⋮	⋮	⋮		
+ (2V+30R-2)	第3轴 数据	定位点(R)	☞ 43页 定位点数据	
+ (2V+30R-1)		[Da. 2]控制方式/[Da. 29] 插补轴速度指定	M代码: 0~32767 齿距数: 0~999	
		[Da. 10]M代码		

软元件(S2)*1	名称		内容	设置范围
+ (2V+30R)	第3轴 数据	定位点(R)	[Da. 11]停留时间	0~5000 [ms]
+ (2V+30R+1)			运行中转矩限制值*3	1~10000 [x0.1%]
+ (2V+30R+2)			[Da. 8]指令速度	mm: 1~600000000×10 ⁻² [mm/min]
+ (2V+30R+3)				inch: 1~600000000×10 ⁻³ [inch/min]
+ (2V+30R+4)			[Da. 6]定位地址/移动量	degree: 1~2147483647×10 ⁻³ [degree/min]*4
+ (2V+30R+5)				pulse: 1~2147483647 [pulse/s]
+ (2V+30R+6)				-2147483648~2147483647
+ (2V+30R+7)			[任意单位]	-2147483648~2147483647
			[任意单位]	
+ (2V+30R+8)	第4轴 数据	轴No.		R64MTCPU: 1~64、1001~1064*2 R32MTCPU: 1~32、1001~1032*2 R16MTCPU: 1~16、1001~1016*2
+ (2V+30R+9)		用户不能使用		0
+ (2V+30R+10)		定位点1	[Da. 2]控制方式/[Da. 29]插补轴速度指定	 43页 定位点数据
+ (2V+30R+11)			[Da. 10]M代码	M代码 : 0~32767 齿距数: 0~999
+ (2V+30R+12)			[Da. 11]停留时间	0~5000 [ms]
+ (2V+30R+13)			运行中转矩限制值*3	1~10000 [x0.1%]
+ (2V+30R+14)			[Da. 8]指令速度	mm: 1~600000000×10 ⁻² [mm/min]
+ (2V+30R+15)			inch: 1~600000000×10 ⁻³ [inch/min]	
+ (2V+30R+16)			degree: 1~2147483647×10 ⁻³ [degree/min]*4	
+ (2V+30R+17)			pulse: 1~2147483647 [pulse/s]	
+ (2V+30R+18)			[Da. 6]定位地址/移动量	-2147483648~2147483647
+ (2V+30R+19)			[任意单位]	-2147483648~2147483647
			[任意单位]	-2147483648~2147483647
			[任意单位]	[任意单位]
⋮		⋮	⋮	⋮
+ (2V+40R)		定位点 (R)	[Da. 2]控制方式/[Da. 29]插补轴速度指定	 43页 定位点数据
+ (2V+40R+1)			[Da. 10]M代码	M代码: 0~32767 齿距数: 0~999
+ (2V+40R+2)			[Da. 11]停留时间	0~5000 [ms]
+ (2V+40R+3)			运行中转矩限制值*3	1~10000 [x0.1%]
+ (2V+40R+4)			[Da. 8]指令速度	mm: 1~600000000×10 ⁻² [mm/min]
+ (2V+40R+5)	inch: 1~600000000×10 ⁻³ [inch/min]			
+ (2V+40R+6)	degree: 1~2147483647×10 ⁻³ [degree/min]*4			
+ (2V+40R+7)	pulse: 1~2147483647 [pulse/s]			
+ (2V+40R+8)	[Da. 6]定位地址/移动量		-2147483648~2147483647	
+ (2V+40R+9)	[任意单位]		-2147483648~2147483647	
		[任意单位]	[任意单位]	

*1 定位数据区域的说明中，“+(2V+10R+3)”等的“R”及“V”表示下述项目对应的偏置的数值

- R: 定位点
- V: 定位数据用项目

各项目对应的偏置的数值应按下述方式计算。

(例) 定位用数据项目为“5”、定位点为“20”的情况下

$$+(2V+10R+3)=+(2\times 5+10\times 20+3)=+213$$

*2 使用指令生成轴时。

*3 连续轨迹控制时，仅VPF/VPR、VVF/VVR有效。在其它控制方式中设置转矩限制的情况下，应在定位数据项目指定中设置启动时转矩限制值。不使用运行中转矩限制值的情况下应设置为“0”。通过VVF/VVR启动时设置了了转矩限制及运行中转矩限制二者的情况下，启动时转矩限制将变为有效。

*4 degree轴速度10倍指定有效时的设置范围为1~2147483647×10⁻²[degree/min]。

功能

n 控制内容

- 根据(S2)中指定的本机的软元件以后的定位数据区域中存储的数据，进行至运动CPU的直接定位启动请求。
- 为了避免对同一机号运动CPU的同一轴执行多个指令，需要通过CPU缓冲存储器上的启动受理标志及用户软元件置入互锁条件。

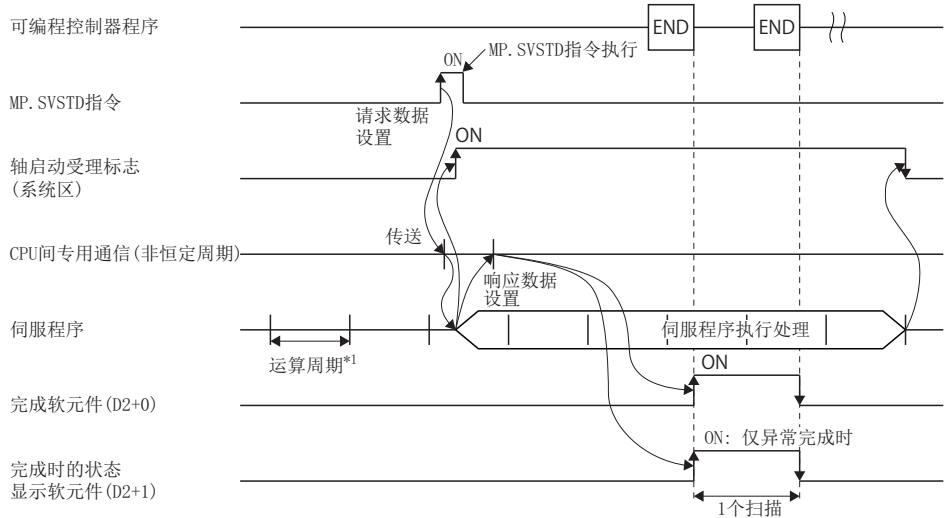
要点

关于启动受理标志的详细内容，请参阅启动受理标志(系统区域)。(P.73页 启动受理标志(系统区域))

n 动作时机

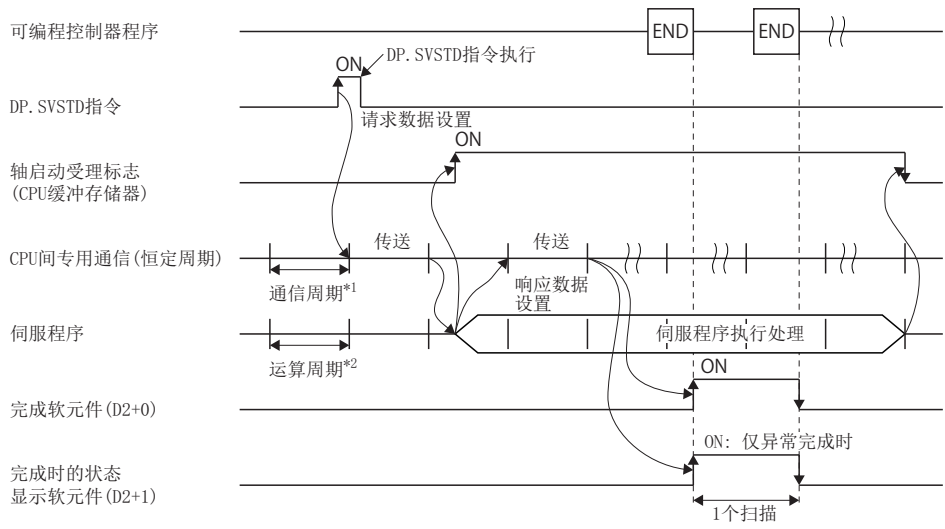
执行了MP. SVSTD指令、DP. SVSTD指令时CPU之间的动作概要如下所示。

- MP. SVSTD指令



*1 在[运动CPU通用参数]⇒[基本设置]中设置

- DP. SVSTD指令



*1 在GX Works3的[系统参数]⇒[多CPU设置]中设置

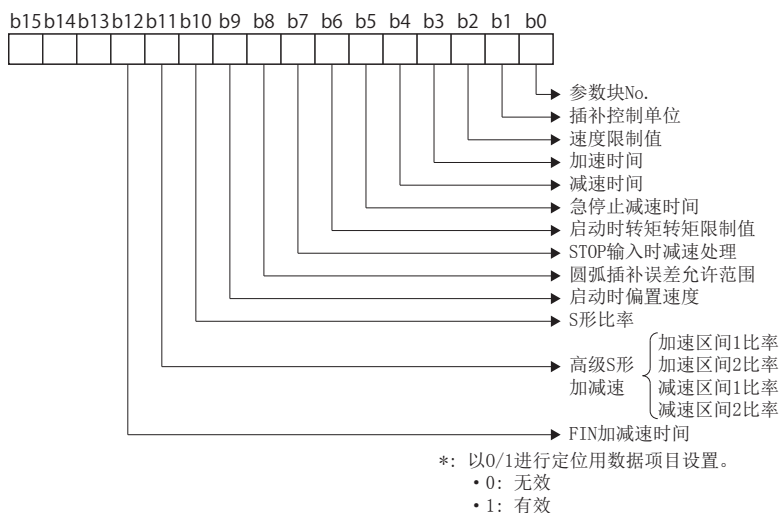
*2 在[运动CPU通用参数]⇒[基本设置]中设置

n 定位类型/定位点数

启动连续轨迹控制 (CPSTART) 的情况下，设置定位点数(1~128)。但是，设置的定位点数不能超过控制数据 (S1+1) 中设置的定位数据区域点数(14~2044)。启动连续轨迹控制 (CPSTART) 以外的情况下设置为“0”。定位点仅设置为1点。

n 定位用数据项目设置

- 设置执行指令时使用的定位用数据项目。将设置的项目的位设置为ON(1:有效)。对于设置为OFF(0:无效)的项目，使用参数块No. 1的数据进行定位启动。



- 将设置为ON(1:有效)的数据从“定位用数据项目1”开始依次填充对齐进行配置。关于设置的数据内容，请参阅以下手册。

📖 MELSEC iQ-R运动控制器编程手册(定位控制篇)

对于定位用数据项目，1个项目中使用2字。但是，高级S字加减速时使用4个项目(8字)。

例

设置了参数块No.、加速时间、高级S字加减速的情况下

偏置	名称	设置值
(S2)+1	定位用数据项目指定	0809H
(S2)+2	定位用数据项目1	参数块No.
(S2)+3		
(S2)+4	定位用数据项目2	加速时间
(S2)+5		
(S2)+6	定位用数据项目3	高级S字加减速 加速区间1比率
(S2)+7		
(S2)+8	定位用数据项目4	高级S字加减速 加速区间2比率
(S2)+9		
(S2)+10	定位用数据项目5	高级S字加减速 减速区间1比率
(S2)+11		
(S2)+12	定位用数据项目6	高级S字加减速 减速区间2比率
(S2)+13		

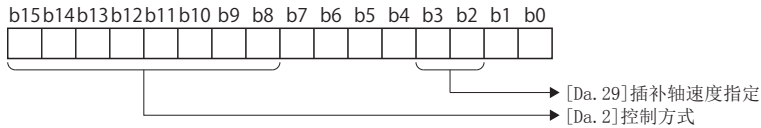
- 根据启动的控制方式，可以设置的数据项目有所不同。指定了不能使用的数据项目的情况下，将被忽略。此外，设置了二个排他的数据项目的情况下，定位用数据项目指定中后面的位项目将优先。

例

设置了S字比率及高级S字加减速二者的情况下，S字比率项目将被忽略。

n 定位点数据

设置进行定位控制情况下的“[Da. 2]控制方式”及“[Da. 29]插补轴速度指定”。



• [Da. 29]插补轴速度指定

设置值	内容
0或1	合成速度
2	基准轴速度
3	长轴速度

• [Da. 2]控制方式

定位控制		指令符号	控制内容	设置值	备注
直线插补控制	1轴	ABS-1	绝对1轴直线定位	01H	n 插补控制时*1 • 进行插补运行的情况下，1~4轴数据中，设置为插补轴(基准轴以外的)“[Da. 2]控制方式”应设置为“NOP”。对于螺旋插补的直线轴，应设置为“ABS-1”或“INC-1”。控制方式设置不正确的的情况下，将发生轻度出错(出错代码：19FCH)而不启动。 n 连续轨迹控制(CPSTART)时 • “[Da. 2]控制方式”中可设置的控制方式为直线插补控制(ABS-1~4、INC-1~4)、圆弧插补控制(ABS/INC)、螺旋插补控制(ABS/INC)。如果设置了其它控制方式，将发生轻度出错(出错代码：19FCH)而不启动。 • 在3轴以上的连续轨迹控制中，进行圆弧插补控制及螺旋插补控制的情况下，停止轴(插补对象以外的轴)的“[Da. 2]控制方式”中应设置为“0”。 • “[Da. 29]插补速度指定”的设置将被忽略，变为常时“合成速度”。
		INC-1	增量1轴定位	02H	
	2轴	ABS-2	绝对2轴直线定位	0AH	
		INC-2	增量2轴定位	0BH	
	3轴	ABS-3	绝对3轴直线定位	15H	
		INC-3	增量3轴定位	16H	
	4轴	ABS-4	绝对4轴直线定位	1AH	
		INC-4	增量4轴定位	1BH	
圆弧插补控制	辅助点指定	ABS ↗	绝对辅助点指定圆弧插补	0DH	
		INC ↗	增量辅助点指定圆弧插补	0EH	
	中心点指定	ABS ↖	绝对中心点指定圆弧插补 CW	0FH	
		ABS ↘	绝对中心点指定圆弧插补 CCW	10H	
		INC ↖	增量中心点指定圆弧插补 CW	11H	
		INC ↘	增量中心点指定圆弧插补 CCW	12H	
	半径指定	ABS ↙	绝对半径指定圆弧插补 CW180°以下	30H	
		ABS ↘	绝对半径指定圆弧插补 CW180°以上	31H	
		ABS ↖	绝对半径指定圆弧插补 CCW180°未満	32H	
		ABS ↗	绝对半径指定圆弧插补 CCW180°以上	33H	
		INC ↙	增量半径指定圆弧插补 CW180°以下	34H	
		INC ↘	增量半径指定圆弧插补 CW180°以上	35H	
螺旋插补控制	辅助点指定	ABH ↗	绝对辅助点指定螺旋插补	20H	
		INH ↗	增量辅助点指定螺旋插补	21H	
	中心点指定	ABH ↖	绝对中心点指定螺旋插补 CW	22H	
		ABH ↘	绝对中心点指定螺旋插补 CCW	23H	
		INH ↖	增量中心点指定螺旋插补 CW	24H	
		INH ↘	增量中心点指定螺旋插补 CCW	25H	
	半径指定	ABH ↙	绝对半径指定螺旋插补 CW180°以下	28H	
		ABH ↘	绝对半径指定螺旋插补 CW180°以上	29H	
ABH ↖		绝对半径指定螺旋插补 CCW180°以下	2AH		
ABH ↗		绝对半径指定螺旋插补 CCW180°以上	2BH		
INH ↙		增量半径指定螺旋插补 CW180°以下	2CH		
INH ↘		增量半径指定螺旋插补 CW180°以上	2DH		
定距进给控制	1轴	FEED1	1轴定距进给控制	03H	
		FEED2	2轴直线插补定距进给控制	0CH	
		FEED3	3轴直线插补定距进给控制	17H	

定位控制		指令符号	控制内容	设置值	备注
速度控制(I)	正转	VF	速度控制(I)正转启动	04H	
	反转	VR	速度控制(I)反转启动	05H	
速度控制(II)	正转	VVF	速度控制(II)正转启动	70H	
	反转	VVR	速度控制(II)反转启动	71H	
速度·位置切换控制	正转	VPF	速度·位置切换控制正转启动	06H	<ul style="list-style-type: none"> 不支持VPSTART指令。 通过M(P).SVSTD/D(P).SVSTD指令执行VPF/VPR指令后,通过运动SFC程序执行VPSTART指令时,将发生轻度出错(出错代码:19FCH)。
	反转	VPR	速度·位置切换控制反转启动	07H	
固定位置停止速度控制	正转	PVF	固定位置停止速度控制正转启动	73H	<ul style="list-style-type: none"> (D1)中指定使用固定位置停止指令的运动CPU的位软元件。(D1)中未指定有效的软元件时,完成状态(S1+0)中将存储出错代码(2225)且不启动。指定了超出范围的编号的情况下,将发生轻度出错(出错代码:19FCH)。 加减速时间是在定位用数据项目设置的“加速时间(b4)”中设置。如果未设置,完成状态(S1+0)中将存储出错代码(2224)且不启动。
	反转	PVR	固定位置停止速度控制反转启动	74H	
位置跟踪控制		PFSTART	位置跟踪控制启动	75H	在(D1)中指定存储位置跟踪地址的运动CPU的字软元件。(D1)中未指定有效的软元件时,完成状态(S1+0)中将存储出错代码(2225)且不启动。指定了超出范围的编号的情况下,将发生轻度出错(出错代码:19FCH)。
原点复位		ZERO	原点复位启动	76H	根据原点复位数据进行原点复位。“[Da.2]控制方式”以外的数据将被忽略。
高速振荡		OSC	高速振荡	77H	对于高速振荡的设置项目(开始角、振幅、频率),应在以下定位点的数据中进行设置。 <ul style="list-style-type: none"> 振幅: [Da.6]定位地址/移动量 开始角: [Da.7]圆弧地址 频率: [Da.8]指令速度
NOP指令		NOP	无处理	80H	

*1 基准轴及插补轴中设置的数据项目如下所示。

◎: 必须设置

○: 根据需要设置(不需要时“—”)

△: 根据控制方式“—”或“◎”

—: 无需设置(设置值将被忽略)

设置项目		基准轴的设置项目	插补轴的设置项目	螺旋插补的直线轴的设置项目
同一定位点No.	[Da.2]控制方式	◎	◎(指定NOP)	◎(指定ABS-1或INC-1)
	[Da.6]定位地址/移动量	◎	◎	◎
	[Da.7]圆弧地址	△(仅圆弧插补、螺旋插补的圆弧轴◎)	△(仅圆弧轴◎)	—
	[Da.8]指令速度	◎	—	—
	[Da.10]M代码	○	—	—
	[Da.11]停留时间	○	—	◎
	[Da.29]插补轴速度指定	○	—	—

n轴启动受理标志(系统区域)

对象机号CPU缓冲存储器内的启动受理标志地址中，将存储启动受理标志的完成状态。

CPU缓冲存储器地址 (10进制数)	内容																														
204H(516) 205H(517) 206H(518) 207H(519)	<p>启动受理标志按64轴存储在对应的各位中。 实际位设置如下所示。</p> <ul style="list-style-type: none"> • R64MTCPU: J1~J64 • R32MTCPU: J1~J32 • R16MTCPU: J1~J16 <p>OFF: 可以启动受理 ON: 不能启动受理</p> <table border="1"> <thead> <tr> <th></th> <th>b15</th> <th></th> <th>b2</th> <th>b1</th> <th>b0</th> </tr> </thead> <tbody> <tr> <td>204H(516)地址编码</td> <td>J16</td> <td>••••••••</td> <td>J2</td> <td>J1</td> <td></td> </tr> <tr> <td>205H(517)地址编码</td> <td>J32</td> <td>••••••••</td> <td>J18</td> <td>J17</td> <td></td> </tr> <tr> <td>206H(518)地址编码</td> <td>J48</td> <td>••••••••</td> <td>J34</td> <td>J33</td> <td></td> </tr> <tr> <td>207H(519)地址编码</td> <td>J64</td> <td>••••••••</td> <td>J50</td> <td>J49</td> <td></td> </tr> </tbody> </table>		b15		b2	b1	b0	204H(516)地址编码	J16	••••••••	J2	J1		205H(517)地址编码	J32	••••••••	J18	J17		206H(518)地址编码	J48	••••••••	J34	J33		207H(519)地址编码	J64	••••••••	J50	J49	
	b15		b2	b1	b0																										
204H(516)地址编码	J16	••••••••	J2	J1																											
205H(517)地址编码	J32	••••••••	J18	J17																											
206H(518)地址编码	J48	••••••••	J34	J33																											
207H(519)地址编码	J64	••••••••	J50	J49																											
20EH(526) 20FH(527) 210H(528) 211H(529)	<p>指令生成轴的启动受理标志按64轴存储在对应的各位中。 实际位设置如下所示。</p> <ul style="list-style-type: none"> • R64MTCPU: J1~J64 • R32MTCPU: J1~J32 • R16MTCPU: J1~J16 <p>OFF: 可以启动受理 ON: 不能启动受理</p> <table border="1"> <thead> <tr> <th></th> <th>b15</th> <th></th> <th>b2</th> <th>b1</th> <th>b0</th> </tr> </thead> <tbody> <tr> <td>20EH(526)地址编码</td> <td>J16</td> <td>••••••••</td> <td>J2</td> <td>J1</td> <td></td> </tr> <tr> <td>20FH(527)地址编码</td> <td>J32</td> <td>••~••••••</td> <td>J18</td> <td>J17</td> <td></td> </tr> <tr> <td>210H(528)地址编码</td> <td>J48</td> <td>••••~••••</td> <td>J34</td> <td>J33</td> <td></td> </tr> <tr> <td>211H(529)地址编码</td> <td>J64</td> <td>••••~••••</td> <td>J50</td> <td>J49</td> <td></td> </tr> </tbody> </table>		b15		b2	b1	b0	20EH(526)地址编码	J16	••••••••	J2	J1		20FH(527)地址编码	J32	••~••••••	J18	J17		210H(528)地址编码	J48	••••~••••	J34	J33		211H(529)地址编码	J64	••••~••••	J50	J49	
	b15		b2	b1	b0																										
20EH(526)地址编码	J16	••••••••	J2	J1																											
20FH(527)地址编码	J32	••~••••••	J18	J17																											
210H(528)地址编码	J48	••••~••••	J34	J33																											
211H(529)地址编码	J64	••••~••••	J50	J49																											

出错

- 以下情况下将异常结束，完成状态存储软元件(S1+0)中指定的软元件中将存储出错代码。

完成状态*1 (出错代码)(H)	出错原因	处理方法
0010	从可编程控制器CPU至运动CPU的指令请求超出了允许值。	确认程序，修改为正确的顺控程序。
2100	从可编程控制器CPU至运动CPU的指令请求M(P).SVST/D(P).SVST/M(P).SVSTD/D(P).SVSTD/M(P).CHGA/D(P).CHGA/M(P).CHGAS/D(P).CHGAS合计同时有257个指令以上，运动CPU无法处理。	
2209	M(P).SVSTD/D(P).SVSTD指令中设置的轴No.不正确。	
2220	M(P).SVSTD/D(P).SVSTD指令的指令区域不足。(通过M(P).SVSTD/D(P).SVSTD指令动作中的指令超过了128个指令)	
2221	M(P).SVSTD/D(P).SVSTD指令的定位数据区域点数不正确。	
2222	M(P).SVSTD/D(P).SVSTD指令的指令仅为NOP或停止轴设置。	
2224	在M(P).SVSTD/D(P).SVSTD指令中必须进行定位数据项目指定的指令中，未进行项目设置。	
2225	M(P).SVSTD/D(P).SVSTD指令中字符串数据(D1)不正确。	

*1 0000H(正常)

• 以下情况下将运算出错，“最新自诊断出错(SM0)”将ON，出错代码将被存储到“最新自诊断出错代码(SD0)”中。

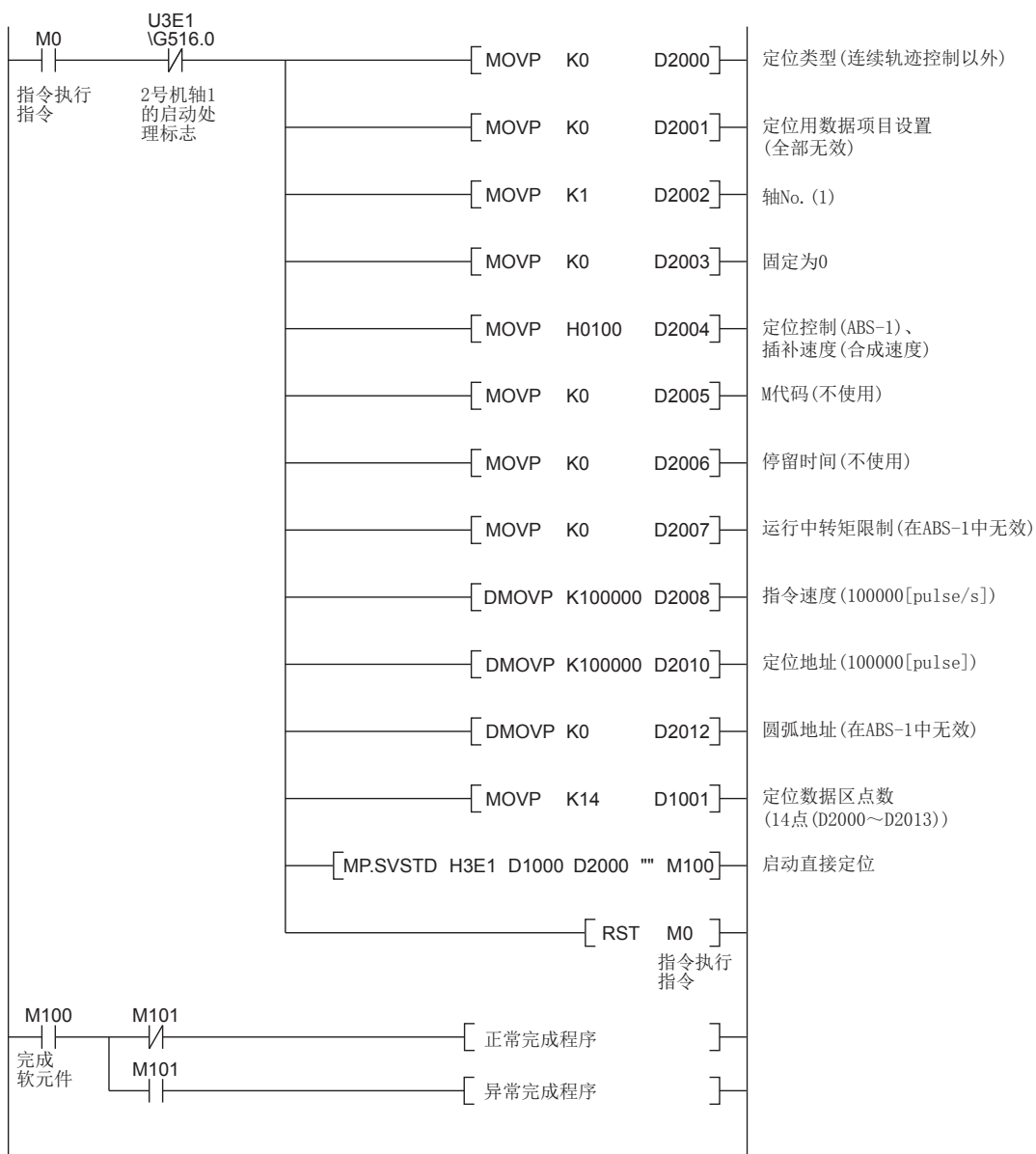
出错代码(H)*1	出错原因	处理方法
2800	指定的其它机号CPU模块的起始输入输出编号(以16进制数4位表示时的前3位)超出了3E0H~3E3H的范围时。	确认程序，修改为正确的顺控程序。
2801	指定的其它机号CPU模块有误。 • 指定了被设置为保留的机号时。 • 指定了未安装的机号时。	

*1 0000H(正常)

程序示例

n 以下为M0变为ON时，在运动CPU(2号机)中进行相当于以下伺服程序的直接定位启动请求的程序

ABS-1	
轴 1,	100000 pulse
速度	100000 pulse/s



至运动CPU的当前值更改指令：M(P).CHGA/D(P).CHGA

指令符号	执行条件	顺控程序*1
MP. CHGA DP. CHGA		
M. CHGA D. CHGA		

*1 □=指令符号 (M(P).CHGA: M, D(P).CHGA: D)

设置数据

n 可用软元件

○：可以设置；△：部分可以设置

设置数据*3	可用软元件											
	内部软元件 (系统、用户)		文件寄存器		链接直接软元件 J□\□		模块访问软元件 U□\G□		变址 寄存器 Z□	常数		其它
	位	字	位	字	位	字	位	字		10进制 K、16进 制H	实数字 字符串	
(n1)		○		○						○		
(S1)		○		○							○	
(n2)		○		○						○		
(D1)*1	△*2		△*2									
(D2)*1		△*2		△*2								

*1 (D1)、(D2) 两方省略的情况下可以省略。

*2 不能使用局部软元件。

*3 对设置数据 (n1)~(D2) 可进行变址修饰。(常数除外)

n 内容、范围、设置方、数据类型

设置数据	设置内容	设置范围	设置方	数据类型
(n1)	对象机号CPU的起始输入输出编号+16 实际指定的值如下所示。*2 2号机：3E1H；3号机：3E2H；4号机：3E3H	3E1H~3E3H	用户	BIN16位
(S1)	进行当前值更改的轴No. (“Jn”)*3 R64MTCPU: J1~J64 R32MTCPU: J1~J32 R16MTCPU: J1~J16	1~64	用户	字符串
(n2)	更改的当前值的设置	-2147483648~ 2147483647	用户	BIN32位
(D1)*1	完成软元件 • (D1+0)：通过指令的受理处理完成使其1个扫描ON的软元件 • (D1+1)：通过指令的受理异常完成使其1个扫描ON的软元件(异常完成时，D1+0也变为ON)	—	系统	位
(D2)*1	存储完成状态的软元件	—	系统	字

*1 (D1)、(D2) 两方省略的情况下可以省略。

*2 在多CPU构成中，不能将运动CPU设置为1号机。

*3 “n”表示对应于轴No. 的数值。(n=1~64)

n 控制内容

- 将(S1)中指定的轴(停止的轴)的当前值更改为(n2)中指定的当前值。
- 为了避免对同一机号运动CPU的同一轴执行多个指令，需要通过CPU缓冲存储器上的启动受理标志及用户软元件置入互锁条件。

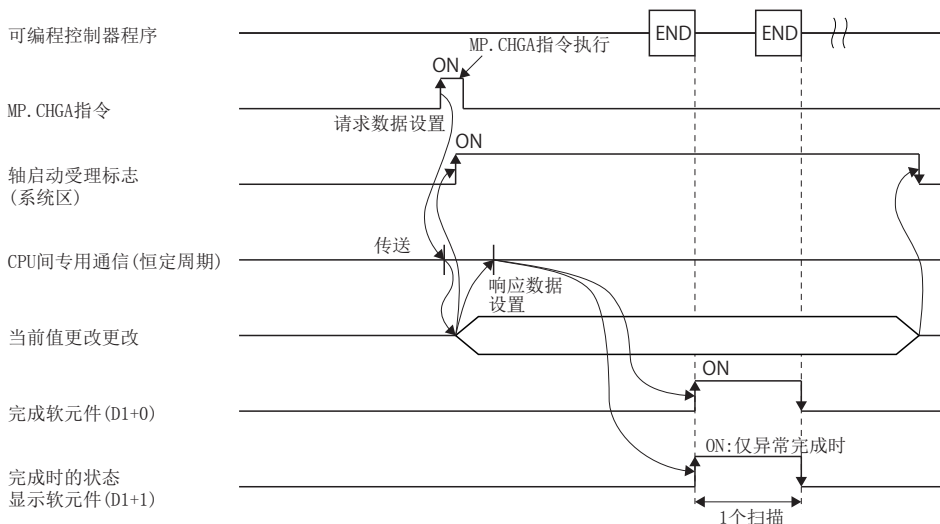
要点

关于启动受理标志的详细内容，请参阅启动受理标志(系统区域)。(P.73页 启动受理标志(系统区域))

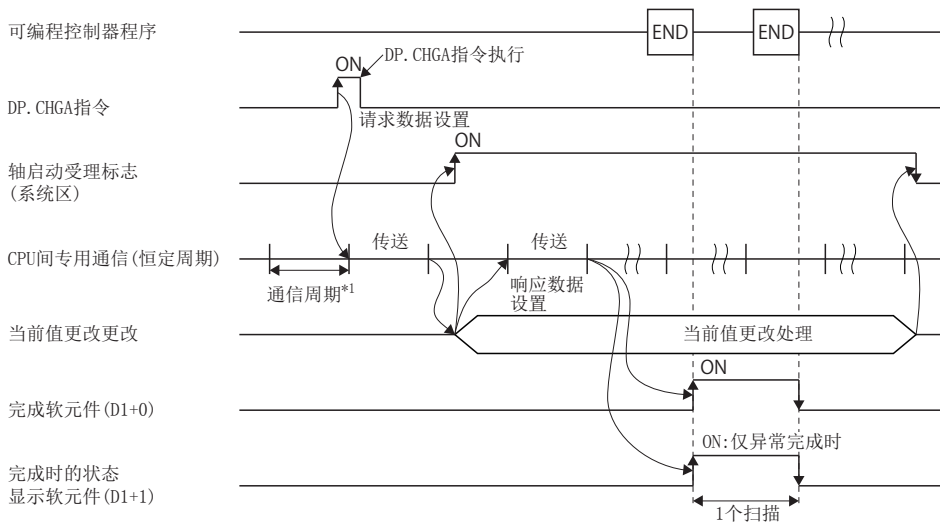
n 动作时机

通过MP. CHGA指令、DP. CHGA指令指定指令轴No. “Jn” 执行时CPU之间的动作概要如下所示。

- MP. CHGA指令



- DP. CHGA指令



*1 在GX Works3的[系统参数]⇒[多CPU设置]中设置

n 进行当前值更改的轴的设置

对于(S1)中设置的进行当前值更改的轴，通过字符串“ ”设置J+轴No.。

运动CPU	(S1)设置范围
R64MTCPU	1~64
R32MTCPU	1~32
R16MTCPU	1~16

可设置的轴数仅为1轴。J设置为大写字母或小写字母，启动的轴No. 使用伺服网络设置中设置的轴No.。关于伺服网络设置，请参阅以下手册。

📖 MELSEC iQ-R运动控制器编程手册(公共篇)

n 轴启动受理标志(系统区域)

指定轴No. “Jn” 执行时，对象机号CPU缓冲存储器内的启动受理标志地址中将存储启动受理标志的完成状态。

CPU缓冲存储器地址 (10进制数)	内容																														
204H (516) 205H (517) 206H (518) 207H (519)	启动受理标志按64轴存储在对应的各位中。 实际位设置如下所示。 • R64MTCPU: J1~J64 • R32MTCPU: J1~J32 • R16MTCPU: J1~J16 OFF: 可以启动受理 ON: 不能启动受理																														
	<table border="1" style="width: 100%; text-align: center;"> <tr> <td></td> <td>b15</td> <td></td> <td>b2</td> <td>b1</td> <td>b0</td> </tr> <tr> <td>204H (516) 地址编码</td> <td>J16</td> <td>••••••••</td> <td>J2</td> <td>J1</td> <td></td> </tr> <tr> <td>205H (517) 地址编码</td> <td>J32</td> <td>••••••••</td> <td>J18</td> <td>J17</td> <td></td> </tr> <tr> <td>206H (518) 地址编码</td> <td>J48</td> <td>••••••••</td> <td>J34</td> <td>J33</td> <td></td> </tr> <tr> <td>207H (519) 地址编码</td> <td>J64</td> <td>••••••••</td> <td>J50</td> <td>J49</td> <td></td> </tr> </table>		b15		b2	b1	b0	204H (516) 地址编码	J16	••••••••	J2	J1		205H (517) 地址编码	J32	••••••••	J18	J17		206H (518) 地址编码	J48	••••••••	J34	J33		207H (519) 地址编码	J64	••••••••	J50	J49	
	b15		b2	b1	b0																										
204H (516) 地址编码	J16	••••••••	J2	J1																											
205H (517) 地址编码	J32	••••••••	J18	J17																											
206H (518) 地址编码	J48	••••••••	J34	J33																											
207H (519) 地址编码	J64	••••••••	J50	J49																											

出错

- 以下情况下将异常结束，完成状态存储软元件(D2)中指定的软元件中将存储出错代码。省略了完成状态存储软元件(D2)的情况下将不进行出错检测而变为无处理，应加以注意。

完成状态*1 (出错代码)(H)	出错原因	处理方法
0010	从可编程控制器CPU至运动CPU的指令请求超出了允许值。	确认程序，修改为正确的顺控程序。
2100	从可编程控制器CPU至运动CPU的指令请求M(P).SVST/D(P).SVST/M(P).SVSTD/P(P).SVSTD/M(P).CHGA/D(P).CHGA/M(P).CHGAS/D(P).CHGAS合计同时有257个指令以上，运动CPU无法处理。	
2203	M(P).CHGA/D(P).CHGA指令中设置的轴No. 不正确。	

*1 0000H(正常)

- 以下情况下将运算出错，“最新自诊断出错(SM0)”将ON，出错代码将被存储到“最新自诊断出错代码(SDO)”中。

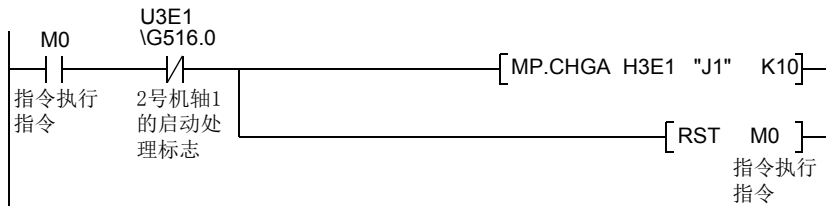
出错代码(H)*1	出错原因	处理方法
2800	指定的其它机号CPU模块的起始输入输出编号(以16进制数4位表示时的前3位)超出了3E0H~3E3H的范围时。	确认程序，修改为正确的顺控程序。
2801	指定的其它机号CPU模块有误。 • 指定了被设置为保留的机号时。 • 指定了未安装的机号时。	

*1 0000H(正常)

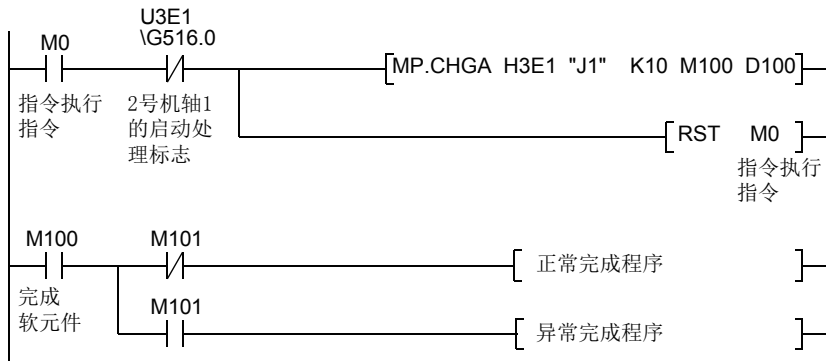
程序示例

n 以下为M0变为ON时将运动CPU(2号机)的轴1的当前值更改为10的程序


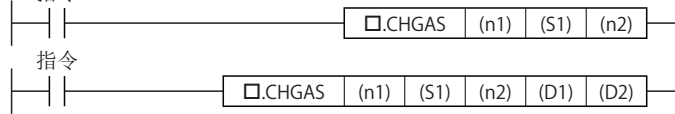

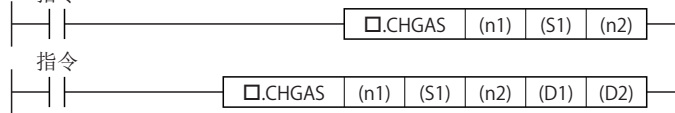
- (例1) 省略了完成软元件、完成状态情况下的程序



- (例2) 使用了完成软元件、完成状态情况下的程序



至运动CPU的指令生成轴的当前值更改指令：M(P).CHGAS/D(P).CHGAS

指令符号	执行条件	顺控程序*1
MP. CHGAS DP. CHGAS		指令 
M. CHGAS D. CHGAS		指令 

*1 □=指令符号 (M(P).CHGAS: M, D(P).CHGAS: D)

设置数据

n 可用软元件

○：可以设置；△：部分可以设置

设置数据*3	可用软元件											
	内部软元件 (系统、用户)		文件寄存器		链接直接软元件 J□\□		模块访问软元件 U□\G□		变址 寄存器 Z□	常数		其它
	位	字	位	字	位	字	位	字		10进制 K、16进 制H	实数字 字符串	
(n1)		○		○						○		
(S1)		○		○							○	
(n2)		○		○						○		
(D1)*1	△*2		△*2									
(D2)*1		△*2		△*2								

*1 (D1)、(D2) 两方省略的情况下可以省略。

*2 不能使用局部软元件。

*3 对设置数据 (n1)~(D2) 可进行变址修饰。(常数除外)

n 内容、范围、设置方、数据类型

设置数据	设置内容	设置范围	设置方	数据类型
(n1)	对象机号CPU的起始输入输出编号+16 实际指定的值如下所示。*2 2号机：3E1H；3号机：3E2H；4号机：3E3H	3E1H~3E3H	用户	BIN16位
(S1)	进行指令生成轴的当前值更改的轴No. (“Jn”)*3 R64MTCPU: J1~J64 R32MTCPU: J1~J32 R16MTCPU: J1~J16	1~64	用户	字符串
(n2)	更改的当前值的设置	-2147483648~ 2147483647	用户	BIN32位
(D1)*1	完成软元件 • (D1+0)：通过指令的受理处理完成使其1个扫描ON的软元件 • (D1+1)：通过指令的受理异常完成使其1个扫描ON的软元件(异常完成时，D1+0也变为ON)	—	系统	位
(D2)*1	存储完成状态的软元件	—	系统	字

*1 (D1)、(D2) 两方省略的情况下可以省略。

*2 在多CPU构成中，不能将运动CPU设置为1号机。

*3 “n”表示对应于轴No. 的数值。(n=1~64)

n 控制内容

- 将(S1)中指定的指令生成轴(停止的轴)的当前值更改为(n2)中指定的当前值。
- 为了避免对同一机号运动CPU的同一轴执行多个指令，需要通过CPU缓冲存储器上的启动受理标志及用户软元件置入互锁条件。

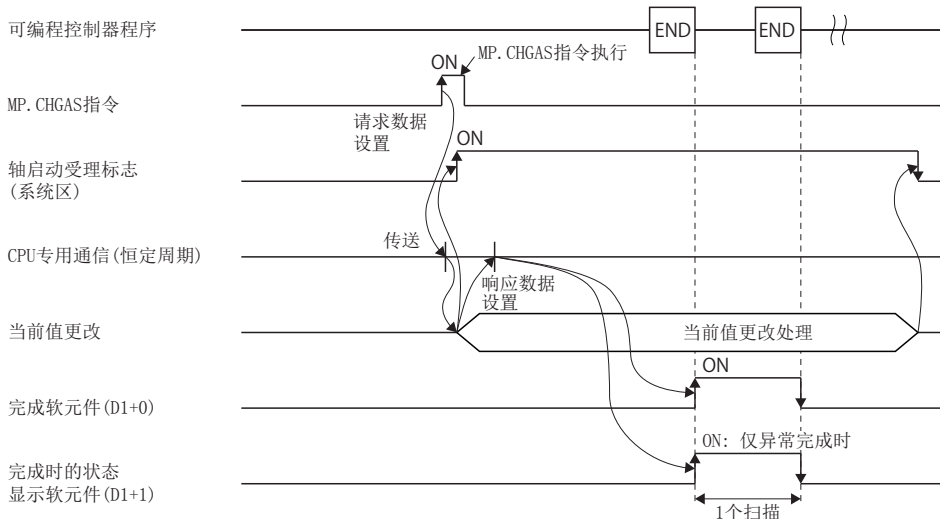
要点

关于启动受理标志的详细内容，请参阅启动受理标志(系统区域)。(P.73页 启动受理标志(系统区域))

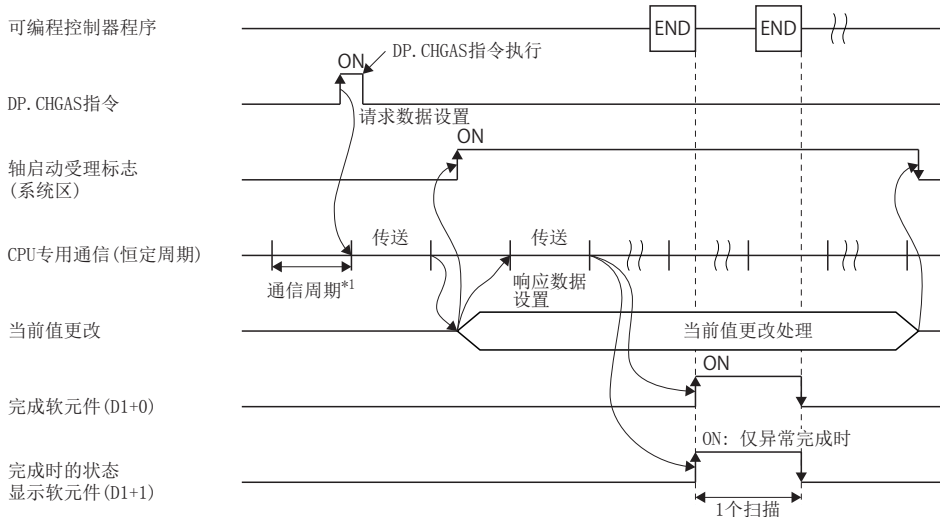
n 动作时机

通过MP. CHGAS指令、DP. CHGAS指令指定轴No. “Jn” 执行时的CPU之间的动作概要如下所示。

- MP. CHGAS指令



- DP. CHGAS指令



*1 在GX Works3的[系统参数]⇒[多CPU设置]中设置

n 进行当前值更改的轴的设置

对于(S1)中设置的进行当前值更改的轴，通过字符串“ ”设置J+轴No.。

运动CPU	(S1) 设置范围
R64MTCPU	1~64
R32MTCPU	1~32
R16MTCPU	1~16

可设置的轴数仅为1轴。J设置为大写字母或小写字母，启动的轴No. 使用指令生成轴参数中设置的轴No.。关于指令生成轴参数，请参阅以下手册。

📖 MELSEC iQ-R运动控制器编程手册(高级同步控制篇)

n 轴启动受理标志(系统区域)

指定轴No. “Jn” 执行时，对象机号CPU缓冲存储器内的启动受理标志地址中将存储启动受理标志的完成状态。

CPU缓冲存储器地址 (10进制数)	内容																														
20EH (526) 20FH (527) 210H (528) 211H (529)	<p>指令生成轴的启动受理标志按64轴存储在对应的各位中。 实际位设置如下所示。</p> <ul style="list-style-type: none"> • R64MTCPU: J1~J64 • R32MTCPU: J1~J32 • R16MTCPU: J1~J16 <p>OFF: 可以启动受理 ON: 不能启动受理</p> <table border="1" style="margin-left: 40px;"> <thead> <tr> <th></th> <th>b15</th> <th></th> <th>b2</th> <th>b1</th> <th>b0</th> </tr> </thead> <tbody> <tr> <td>20EH (526) 地址编码</td> <td>J16</td> <td>••••••••</td> <td>J2</td> <td>J1</td> <td></td> </tr> <tr> <td>20FH (527) 地址编码</td> <td>J32</td> <td>••••••••</td> <td>J18</td> <td>J17</td> <td></td> </tr> <tr> <td>210H (528) 地址编码</td> <td>J48</td> <td>••••••••</td> <td>J34</td> <td>J33</td> <td></td> </tr> <tr> <td>211H (529) 地址编码</td> <td>J64</td> <td>••••••••</td> <td>J50</td> <td>J49</td> <td></td> </tr> </tbody> </table>		b15		b2	b1	b0	20EH (526) 地址编码	J16	••••••••	J2	J1		20FH (527) 地址编码	J32	••••••••	J18	J17		210H (528) 地址编码	J48	••••••••	J34	J33		211H (529) 地址编码	J64	••••••••	J50	J49	
	b15		b2	b1	b0																										
20EH (526) 地址编码	J16	••••••••	J2	J1																											
20FH (527) 地址编码	J32	••••••••	J18	J17																											
210H (528) 地址编码	J48	••••••••	J34	J33																											
211H (529) 地址编码	J64	••••••••	J50	J49																											

出错

- 以下情况下将异常结束，完成状态存储软元件(D2)中指定的软元件中将存储出错代码。省略了完成状态存储软元件(D2)的情况下将不进行出错检测而变为无处理，应加以注意。

完成状态*1 (出错代码)(H)	出错原因	处理方法
0010	从可编程控制器CPU至运动CPU的指令请求超出了允许值。	确认程序，修改为正确的顺控程序。
2100	从可编程控制器CPU至运动CPU的指令请求M(P). SVST/D(P). SVST/M(P). SVSTD/P(P). SVSTD/M(P). CHGA/D(P). CHGA/M(P). CHGAS/D(P). CHGAS合计同时有257个指令以上，运动CPU无法处理。	
2207	M(P). CHGAS/D(P). CHGAS指令中设置的轴No. 不正确。	

*1 0000H(正常)

- 以下情况下将运算出错，“最新自诊断出错(SM0)”将ON，出错代码将被存储到“最新自诊断出错代码(SDO)”中。

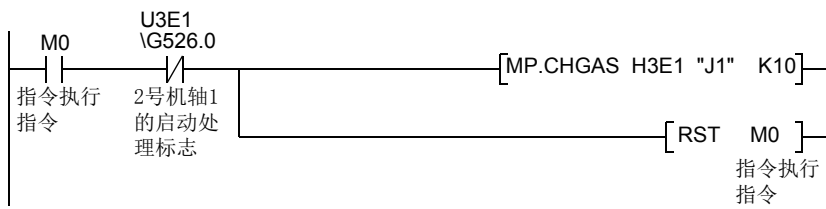
出错代码(H)*1	出错原因	处理方法
2800	指定的其它机号CPU模块的起始输入输出编号(以16进制数4位表示时的前3位)超出了3E0H~3E3H的范围时。	确认程序，修改为正确的顺控程序。
2801	指定的其它机号CPU模块有误。 • 指定了被设置为保留的机号时。 • 指定了未安装的机号时。	

*1 0000H(正常)

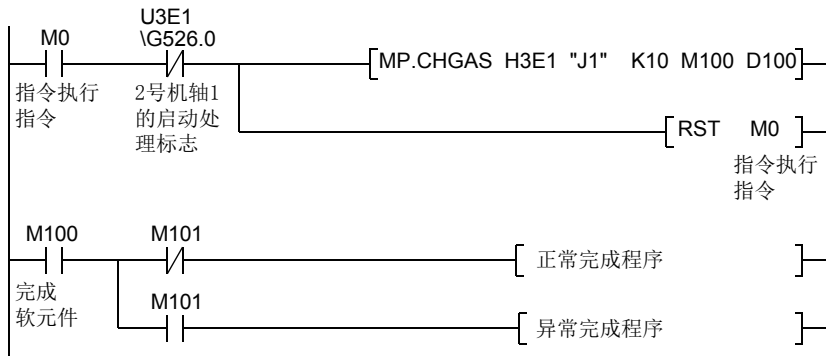
程序示例

n 以下为M0变为ON时将运动CPU(2号机)的轴1的当前值更改为10的程序

- (例1) 省略了完成软元件、完成状态情况下的程序



- (例2) 使用了完成软元件、完成状态情况下的程序



至运动CPU的速度更改指令：M(P).CHGV/D(P).CHGV

指令符号	执行条件	顺控程序*1
MP.CHGV DP.CHGV		
M.CHGV D.CHGV		

*1 □=指令符号(M(P).CHGV; M, D(P).CHGV; D)

设置数据

n 可用软元件

○：可以设置；△：部分可以设置

设置数据*3	可用软元件											
	内部软元件 (系统、用户)		文件寄存器		链接直接软元件 J□\□		模块访问软元件 U□\G□		变址 寄存器 Z□	常数		其它
	位	字	位	字	位	字	位	字		10进制 K、16进 制H	实数字 字符串	
(n1)		○		○						○		
(S1)		○		○							○	
(n2)		○		○						○		
(D1)*1	△*2		△*2									
(D2)*1		△*2		△*2								

*1 (D1)、(D2)两方省略的情况下可以省略。

*2 不能使用局部软元件。

*3 对设置数据(n1)~(D2)可进行变址修饰。(常数除外)

n 内容、范围、设置方、数据类型

设置数据	设置内容	设置范围	设置方	数据类型
(n1)	对象机号CPU的起始输入输出编号+16 实际指定的值如下所示。*2 2号机：3E1H；3号机：3E2H；4号机：3E3H	3E1H~3E3H	用户	BIN16位
(S1)	进行当前值更改的轴No. (“Jn”)*3 R64MTCPU: J1~J64 R32MTCPU: J1~J32 R16MTCPU: J1~J16	1~64	用户	字符串
(n2)	更改的速度的设置	*4	用户	BIN32位
(D1)*1	完成软元件 • (D1+0)：通过指令的受理处理完成使其1个扫描ON的软元件 • (D1+1)：通过指令的受理异常完成使其1个扫描ON的软元件(异常完成时，D1+0也变为ON)	—	系统	位
(D2)*1	存储完成状态的软元件	—	系统	字

*1 (D1)、(D2)两方省略的情况下可以省略。

*2 在多CPU构成中，不能将运动CPU设置为1号机。

*3 “n”表示对应于轴No.的数值。(n=1~64)

*4 根据设置单位，(n2)的设置范围如下所示。

设置单位	(n2)设置范围
mm	-600000000~600000000($\times 10^{-2}$ [mm/min])
inch	-600000000~600000000($\times 10^{-3}$ [inch/min])
degree	-2147483647~2147483647($\times 10^{-3}$ [degree/min])*5
pulse	-2147483647~2147483647[pulse/s]

*5 degree轴速度10倍指定有效时的设置范围为-2147483647~2147483647($\times 10^{-2}$ [degree/min])

功能

n 控制内容

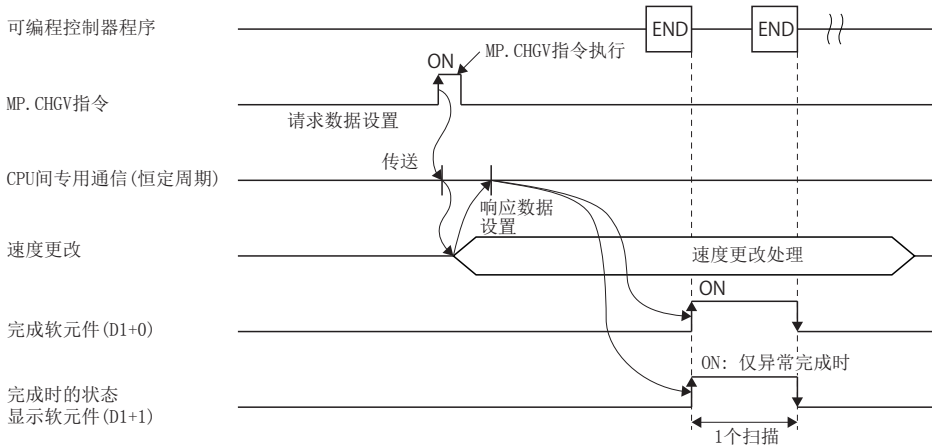
- 将定位中以及JOG运行中的(S1)中指定的轴的速度更改为(n2)中指定的速度。
- 没有速度更改中相关CPU缓冲存储器上的互锁用信号。对同一机号运动CPU的同一轴执行了多个指令时，将被更改为后执行的指令中指定的速度。
- 通过设置(S1)中指定的轴的加减速时间更改参数，可以更改速度更改时的加减速时间。关于加减速时间更改参数及加减速时间更改功能，请参阅以下手册。

📖 MELSEC iQ-R运动控制器编程手册(定位控制篇)

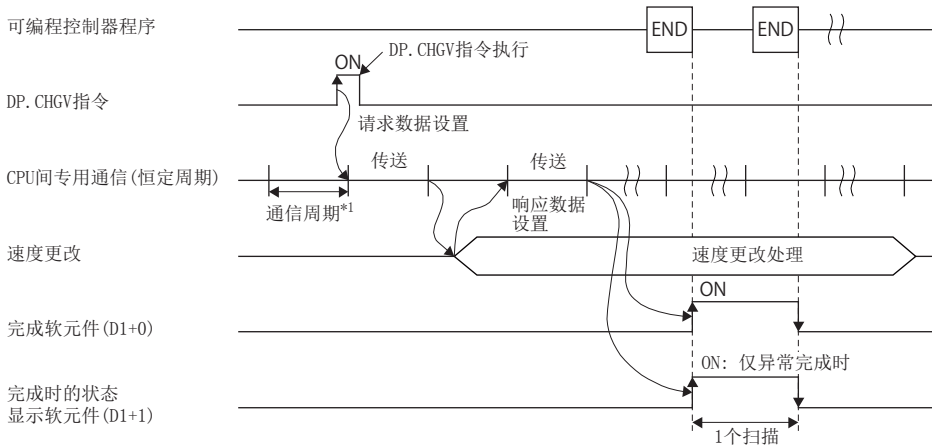
n 动作时机

执行了MP. CHGV指令、DP. CHGV指令时CPU之间的动作概要如下所示。

- MP. CHGV指令



- DP. CHGV指令



*1 在GX Works3的[系统参数]⇒[多CPU设置]中设置

n 进行速度更改的轴的设置

对于(S1)中设置的进行速度更改的轴，将J+轴No.以字符串“ ”进行设置。

运动CPU	(S1)设置范围
R64MTCPU	1~64
R32MTCPU	1~32
R16MTCPU	1~16

可设置的轴数仅为1轴。J设置为大写字母或小写字母，启动的轴No.使用伺服网络设置中设置的轴No.。关于伺服网络设置，请参阅以下手册。

📖 MELSEC iQ-R运动控制器编程手册(公共篇)

出错

- 以下情况下将异常结束，完成状态存储软元件(D2)中指定的软元件中将存储出错代码。省略了完成状态存储软元件(D2)的情况下将不进行出错检测而变为无处理，应加以注意。

完成状态*1 (出错代码)(H)	出错原因	处理方法
0010	从可编程控制器CPU至运动CPU的指令请求超出了允许值。	确认程序，修改为正确的顺控程序。
2204	M(P).CHGV/D(P).CHGV指令中设置的轴No.不正确。	

*1 0000H(正常)

- 以下情况下将运算出错，“最新自诊断出错(SM0)”将ON，出错代码将被存储到“最新自诊断出错代码(SD0)”中。

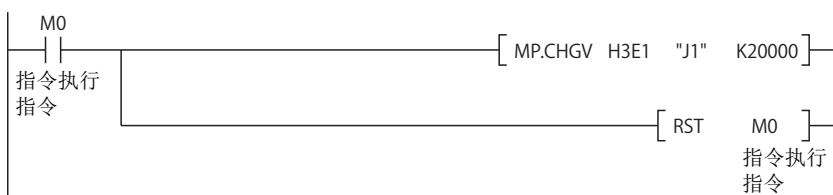
出错代码(H)*1	出错原因	处理方法
2800	指定的其它机号CPU模块的起始输入输出编号(以16进制数4位表示时的前3位)超出了3E0H~3E3H的范围时。	确认程序，修改为正确的顺控程序。
2801	指定的其它机号CPU模块有误。 <ul style="list-style-type: none"> 指定了被设置为保留的机号时。 指定了未安装的机号时。 	

*1 0000H(正常)

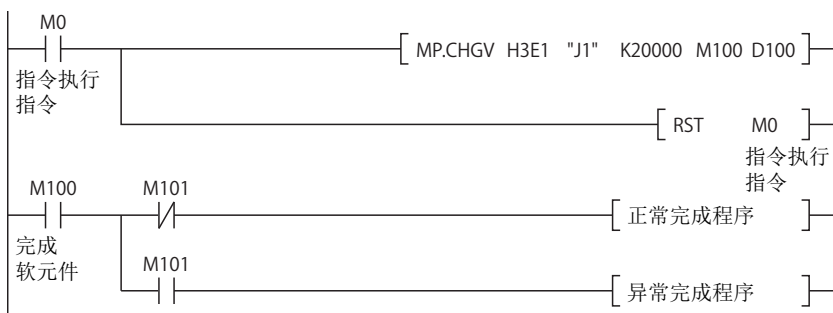
程序示例

n 以下为M0变为ON时将运动CPU(2号机)的轴1的定位速度更改为20000的程序

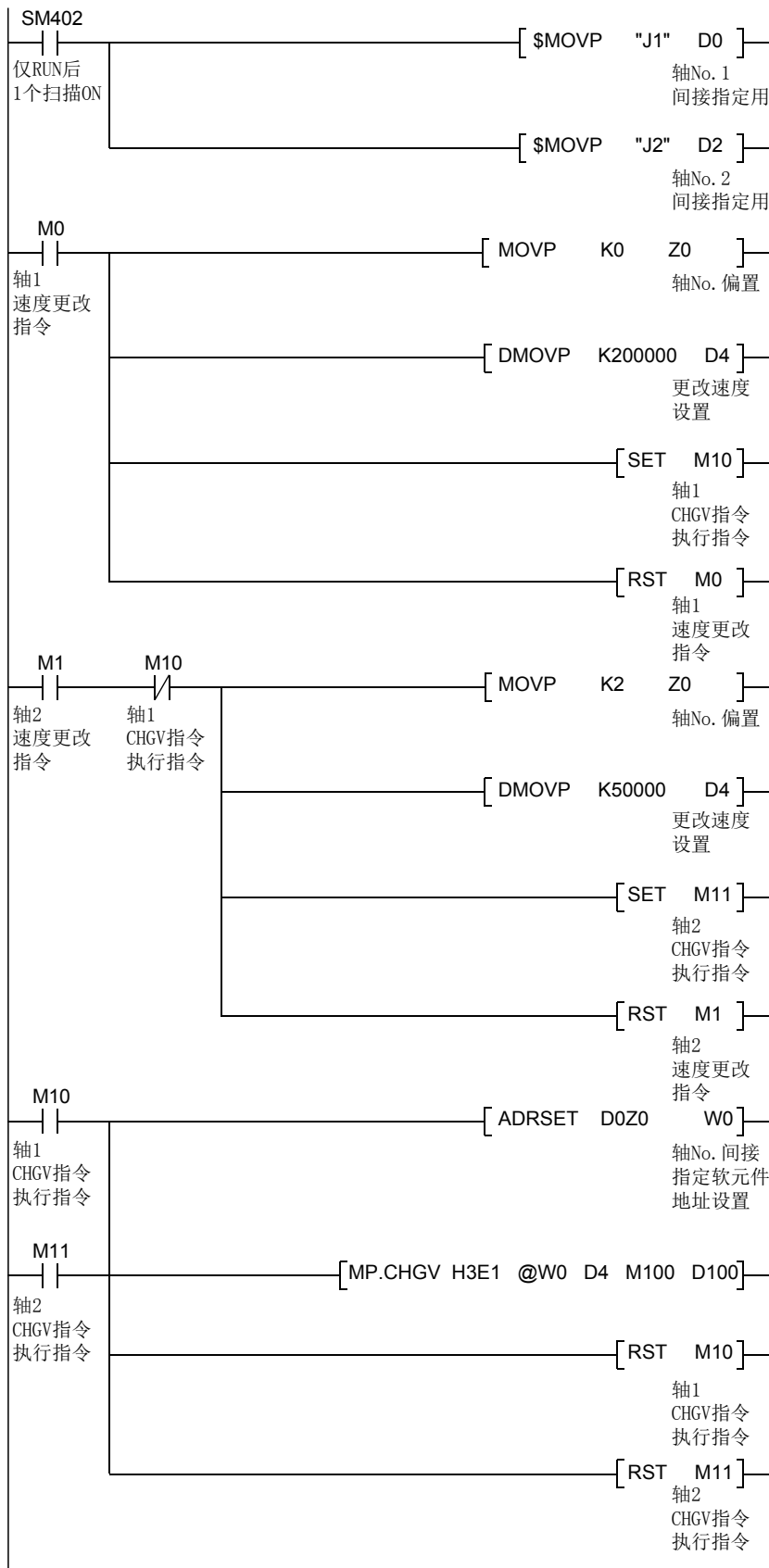
- (例1) 省略了完成软元件、完成状态情况下的程序




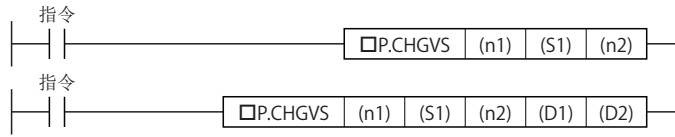

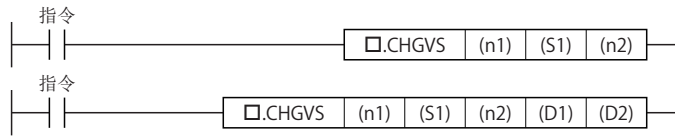
- (例2) 使用了完成软元件、完成状态情况下的程序



n 以下为将轴No. 设置为间接指定方式时，M0变为ON时将运动CPU(2号机)的轴1的定位速度更改为200000，M1变为ON时将轴2的定位速度更改为50000的程序



至运动CPU的指令生成轴的速度更改指令：M(P).CHGVS/D(P).CHGVS

指令符号	执行条件	顺控程序*1
MP.CHGVS DP.CHGVS		
M.CHGVS D.CHGVS		

*1 □=指令符号(M(P).CHGVS: M, D(P).CHGVS: D)

设置数据

n可用软元件

○: 可以设置; △: 部分可以设置

设置数据*3	可用软元件											
	内部软元件 (系统、用户)		文件寄存器		链接直接软元件 J□\□		模块访问软元件 U□\G□		变址 寄存器 Z□	常数		其它
	位	字	位	字	位	字	位	字		10进制 K、16进 制H	实数字 字符串	
(n1)		○		○						○		
(S1)		○		○							○	
(n2)		○		○						○		
(D1)*1	△*2		△*2									
(D2)*1		△*2		△*2								

*1 (D1)、(D2)两方省略的情况下可以省略。

*2 不能使用局部软元件。

*3 对设置数据(n1)~(D2)可进行变址修饰。(常数除外)

n 内容、范围、设置方、数据类型

设置数据	设置内容	设置范围	设置方	数据类型
(n1)	对象机号CPU的起始输入输出编号÷16 实际指定的值如下所示。*2 2号机：3E1H；3号机：3E2H；4号机：3E3H	3E1H~3E3H	用户	BIN16位
(S1)	进行指令生成轴的速度更改的轴No. (“Jn”)*3 R64MTCPU：J1~J64 R32MTCPU：J1~J32 R16MTCPU：J1~J16	1~64	用户	字符串
(n2)	更改的速度的设置	*4	用户	BIN32位
(D1)*1	完成软元件 • (D1+0)：通过指令的受理处理完成使其1个扫描ON的软元件 • (D1+1)：通过指令的受理异常完成使其1个扫描ON的软元件(异常完成时，D1+0也变为ON)	—	系统	位
(D2)*1	存储完成状态的软元件	—	系统	字

*1 (D1)、(D2) 两方省略的情况下可以省略。

*2 在多CPU构成中，不能将运动CPU设置为1号机。

*3 “n”表示对应于轴No.的数值。(n=1~64)

*4 根据设置单位，(n2)的设置范围如下所示。

设置单位	(n2)设置范围
mm	-600000000~600000000($\times 10^{-2}$ [mm/min])
inch	-600000000~600000000($\times 10^{-3}$ [inch/min])
degree	-2147483647~2147483647($\times 10^{-3}$ [degree/min])*5
pulse	-2147483647~2147483647[pulse/s]

*5 degree轴速度10倍指定有效时的设置范围为-2147483647~2147483647($\times 10^{-2}$ [degree/min])

功能

n 控制内容

- 将定位中及JOG运行中的(S1)中指定的指令生成轴的速度更改为(n2)中指定的速度。
- 没有速度更改中相关CPU缓冲存储器上的互锁用信号。对同一机号运动CPU的同一轴执行了多个指令时，将被更改为后执行的指令中指定的速度。
- 通过设置(S1)中指定的轴的加减速时间更改参数，可以更改速度更改时的加减速时间。关于加减速时间更改参数，请参阅以下手册。

📖MELSEC iQ-R运动控制器编程手册(高级同步控制篇)

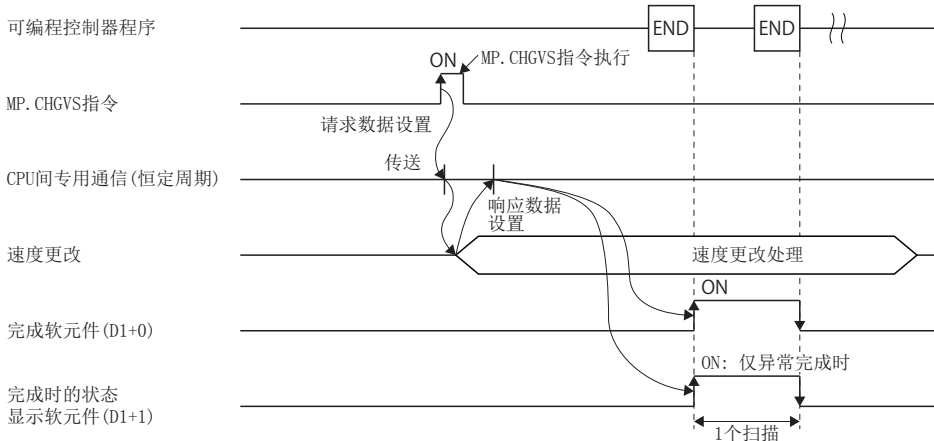
关于加减速时间更改功能，请参阅以下手册。

📖MELSEC iQ-R运动控制器编程手册(定位控制篇)

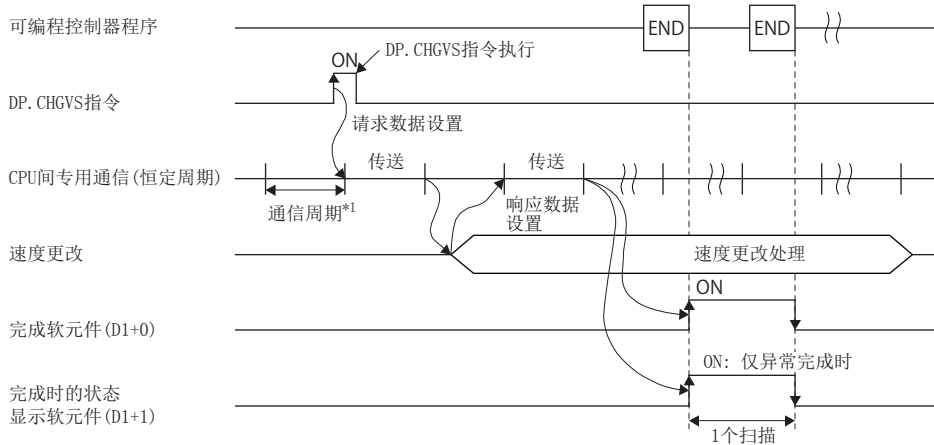
n 动作时机

执行了MP. CHGVS指令、DP. CHGVS指令时CPU之间的动作概要如下所示。

• MP. CHGVS指令



• DP. CHGVS指令



*1 在GX Works3的[系统参数]⇒[多CPU设置]中设置

n 进行速度更改的轴的设置

对于(S1)中设置的进行速度更改的轴，将J+轴No. 以字符串“ ”进行设置。

运动CPU	(S1) 设置范围
R64MTCPU	1~64
R32MTCPU	1~32
R16MTCPU	1~16

可设置的轴数仅为1轴。J设置为大写字母或小写字母，启动的轴No. 使用指令生成轴参数中设置的轴No.。关于指令生成轴参数，请参阅以下手册。

📖 MELSEC iQ-R运动控制器编程手册(高级同步控制篇)

出错

- 以下情况下将异常结束，完成状态存储软元件(D2)中指定的软元件中将存储出错代码。省略了完成状态存储软元件(D2)的情况下将不进行出错检测而变为无处理，应加以注意。

完成状态*1 (出错代码)(H)	出错原因	处理方法
0010	从可编程控制器CPU至运动CPU的指令请求超出了允许值。	确认程序，修改为正确的顺控程序。
2208	M(P).CHGVS/D(P).CHGVS指令中设置的轴No.不正确。	

*1 0000H(正常)

- 以下情况下将运算出错，“最新自诊断出错(SM0)”将ON，出错代码将被存储到“最新自诊断出错代码(SD0)”中。

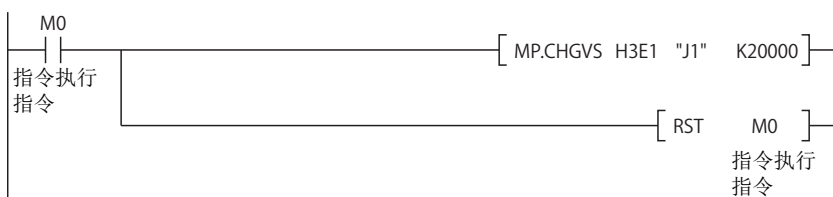
出错代码(H)*1	出错原因	处理方法
2800	指定的其它机号CPU模块的起始输入输出编号(以16进制数4位表示时的前3位)超出了3E0H~3E3H的范围时。	确认程序，修改为正确的顺控程序。
2801	指定的其它机号CPU模块有误。 <ul style="list-style-type: none"> 指定了被设置为保留的机号时。 指定了未安装的机号时。 	

*1 0000H(正常)

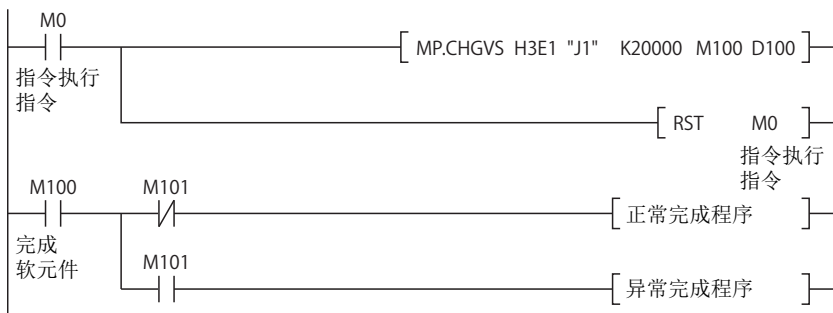
程序示例

n 以下为M0变为ON时将运动CPU(2号机)的轴1的定位速度更改为20000的程序

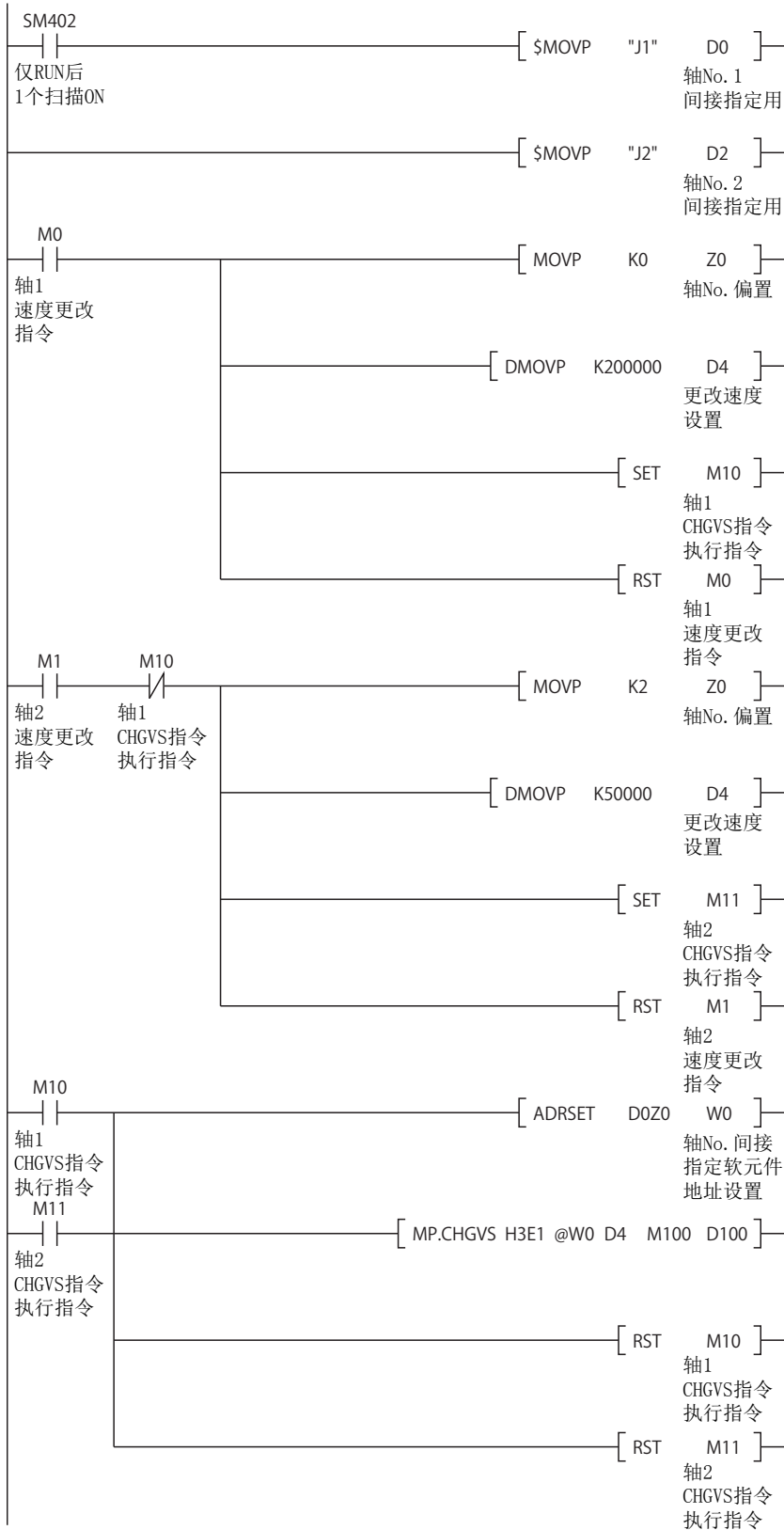
- (例1) 省略了完成软元件、完成状态情况下的程序



- (例2) 使用了完成软元件、完成状态情况下的程序



n 以下为将轴No. 设置为间接指定方式时，M0变为ON时将运动CPU(2号机)的轴1的定位速度更改为200000，M1变为ON时将轴2的定位速度更改为50000的程序



至运动CPU的转矩限制值更改指令：M(P).CHGT/D(P).CHGT

指令符号	执行条件	顺控程序*1
MP. CHGT DP. CHGT		
M. CHGT D. CHGT		

*1 □=指令符号(M(P).CHGT: M, D(P).CHGT: D)

设置数据

n 可用软元件

○: 可以设置; △: 部分可以设置

设置数据*3	可用软元件											
	内部软元件 (系统、用户)		文件寄存器		链接直接软元件 J□\□		模块访问软元件 U□\G□		变址 寄存器 Z□	常数		其它
	位	字	位	字	位	字	位	字		10进制 K、16进 制H	实数字 符串	
(n1)		○		○						○		
(S1)		○		○							○	
(n2)		○		○						○		
(n3)		○		○						○		
(D1)*1	△*2		△*2									
(D2)*1		△*2		△*2								

*1 (D1)、(D2) 两方省略的情况下可以省略。

*2 不能使用局部软元件。

*3 对设置数据(n1)~(D2)可进行变址修饰。(常数除外)

n 内容、范围、设置方、数据类型

设置数据	设置内容	设置范围	设置方	数据类型
(n1)	对象机号CPU的起始输入输出编号÷16 实际指定的值如下所示。*3 2号机: 3E1H; 3号机: 3E2H; 4号机: 3E3H	3E1H~3E3H	用户	BIN16位
(S1)	进行当前值更改的轴No. (“Jn”)*4 R64MTCPU: J1~J64 R32MTCPU: J1~J32 R16MTCPU: J1~J16	1~64	用户	字符串
(n2)*1	更改的正方向转矩限制值的设置(×0.1[%])	1~10000 (×0.1[%])	用户	BIN16位
(n3)*1	更改的负方向转矩限制值的设置(×0.1[%])	1~10000 (×0.1[%])	用户	BIN16位
(D1)*2	完成软元件 • (D1+0): 通过指令的受理处理完成使其1个扫描ON的软元件 • (D1+1): 通过指令的受理异常完成使其1个扫描ON的软元件(异常完成时, D1+0也变为ON)	—	系统	位
(D2)*2	存储完成状态的软元件	—	系统	字

*1 无需更改正方向或负方向的转矩限制值的情况下, 通过设置为-1, 设置的方向将延续上次的转矩限制值。

*2 (D1)、(D2) 两方省略的情况下可以省略。

*3 在多CPU构成中, 不能将运动CPU设置为1号机。

*4 “n”表示对应于轴No. 的数值。(n=1~64)

功能

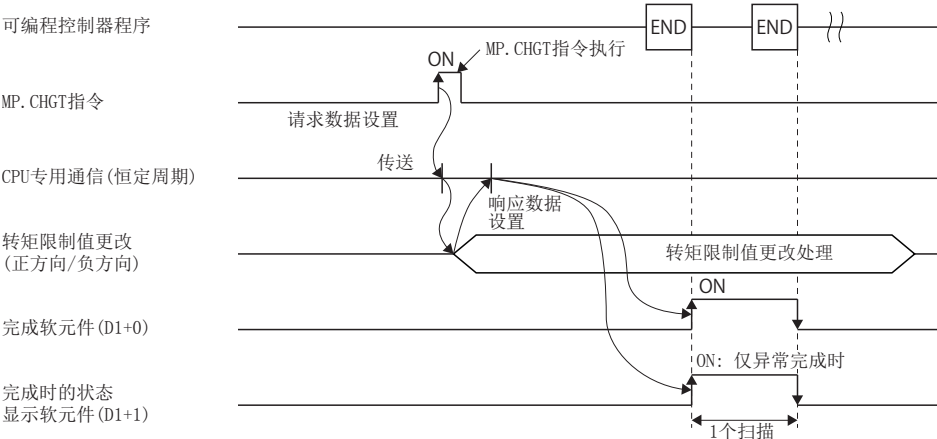
n 控制内容

- 与运行中/停止中无关，将(S1)中指定的轴的转矩限制值更改为(n2)中指定的正方向转矩限制值及(n3)中指定的负方向转矩限制值。
- 没有轴的转矩更改状态相关的互锁用信号。对同一机号运动CPU的同一轴执行了多个指令时，将被更改为后执行的指令中指定的转矩。
- 关于伺服程序中指定的转矩限制值与转矩限制值更改指令的关系，请参阅以下手册。

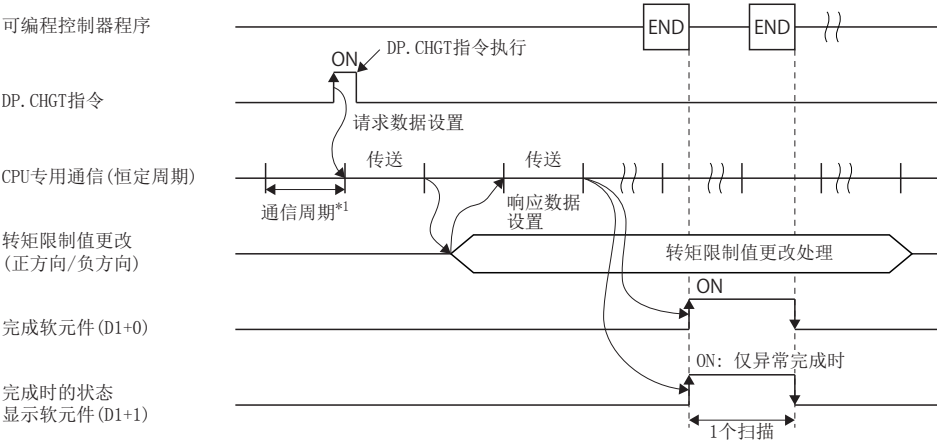
n 动作时机

执行了MP. CHGT指令、DP. CHGT指令时CPU之间的动作概要如下所示。

- MP. CHGT指令



- DP. CHGT指令



*1 在GX Works3的[系统参数]⇒[多CPU设置]中设置

n 进行转矩限制值更改的轴的设置

对于(S1)中设置的进行转矩限制值更改的轴，将J+轴No. 以字符串“ ”进行设置。

运动CPU	(S1) 设置范围
R64MTCPU	1~64
R32MTCPU	1~32
R16MTCPU	1~16

可设置的轴数仅为1轴。J设置为大写字母或小写字母，启动的轴No. 使用伺服网络设置中设置的轴No.。关于伺服网络设置，请参阅以下手册。

出错

- 以下情况下将异常结束，完成状态存储软元件(D2)中指定的软元件中将存储出错代码。省略了完成状态存储软元件(D2)的情况下将不进行出错检测而变为无处理，应加以注意。

完成状态*1 (出错代码)(H)	出错原因	处理方法
0010	从可编程控制器CPU至运动CPU的指令请求超出了允许值。	确认程序，修改为正确的顺控程序。
2205	M(P).CHGT/D(P).CHGT指令中设置的轴No.不正确。	

*1 0000H(正常)

- 以下情况下将运算出错，“最新自诊断出错(SM0)”将ON，出错代码将被存储到“最新自诊断出错代码(SD0)”中。

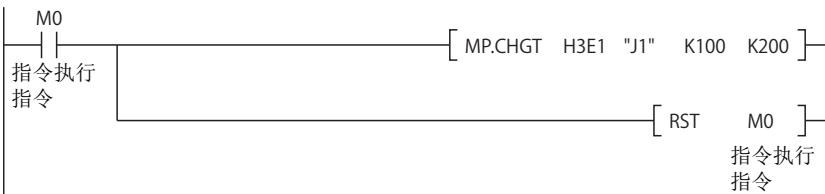
出错代码(H)*1	出错原因	处理方法
2800	指定的其它机号CPU模块的起始输入输出编号(以16进制数4位表示时的前3位)超出了3E0H~3E3H的范围时。	确认程序，修改为正确的顺控程序。
2801	指定的其它机号CPU模块有误。 <ul style="list-style-type: none"> 指定了被设置为保留的机号时。 指定了未安装的机号时。 	

*1 0000H(正常)

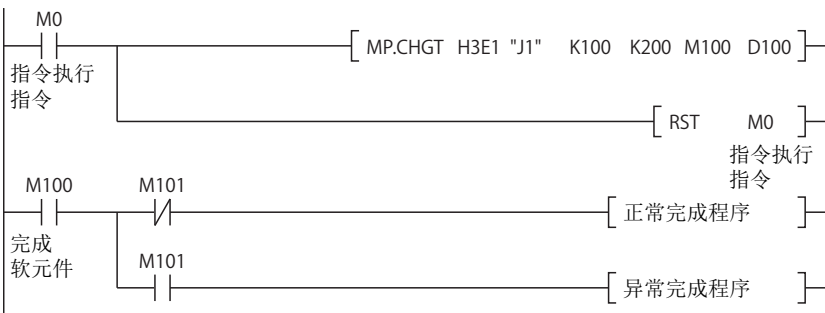
程序示例

n 以下为M0变为ON时将运动CPU(2号机)的轴1的转矩限制值更改为正方向10.0%、负方向20.0%的程序


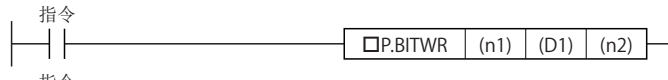
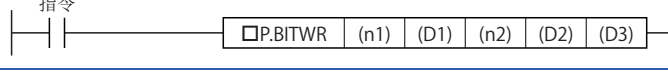

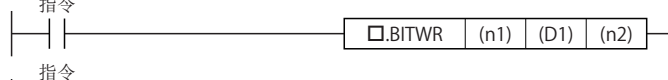
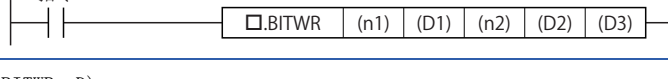
- (例1) 省略了完成软元件、完成状态情况下的程序



- (例2) 使用了完成软元件、完成状态情况下的程序



至运动CPU的位软元件的写入：M(P).BITWR/D(P).BITWR

指令符号	执行条件	顺控程序*1
MP.BITWR DP.BITWR		 
M.BITWR D.BITWR		 

*1 □=指令符号(M(P).BITWR: M, D(P).BITWR: D)

设置数据

n可用软元件

○：可以设置；△：部分可以设置

设置数据*3	可用软元件											
	内部软元件 (系统、用户)		文件寄存器		链接直接软元件 J□\□		模块访问软元件 U□\G□		变址 寄存器 Z□	常数		其它
	位	字	位	字	位	字	位	字		10进制 K、16进 制H	实数字 字符串	
(n1)		○		○						○		
(D1)		○		○							○	
(n2)		○		○						○		
(D2)*1	△*2		△*2									
(D3)*1		△*2		△*2								

*1 (D2)、(D3)两方省略的情况下可以省略。

*2 不能使用局部软元件。

*3 对设置数据(n1)~(D3)可进行变址修饰。(常数除外)

n内容、范围、设置方、数据类型

设置数据	设置内容	设置范围	设置方	数据类型
(n1)	对象机号CPU的起始输入输出编号+16 实际指定的值如下所示。*2 2号机：3E1H；3号机：3E2H；4号机：3E3H	3E1H~3E3H	用户	BIN16位
(D1)	进行位操作的对象机号运动CPU的位软元件(包括字软元件的位指定)	—	用户	字符串
(n2)	位操作方法 实际指定的值如下所示。 OFF: 0 ON: 1	0~1	用户	BIN16位
(D2)*1	完成软元件 • (D2+0)：通过指令的受理处理完成使其1个扫描ON的软元件 • (D2+1)：通过指令的受理异常完成使其1个扫描ON的软元件(异常完成时，D2+0也变为ON)	—	系统	位
(D3)*1	存储完成状态的软元件	—	系统	字

*1 (D2)、(D3)两方省略的情况下可以省略。

*2 在多CPU构成中，不能将运动CPU设置为1号机。

功能

n 控制内容

- 多CPU系统构成时，根据(n2)的位操作方法，将对象机号的(n1)的(D1)中指定的位软元件置为ON或OFF。

要点

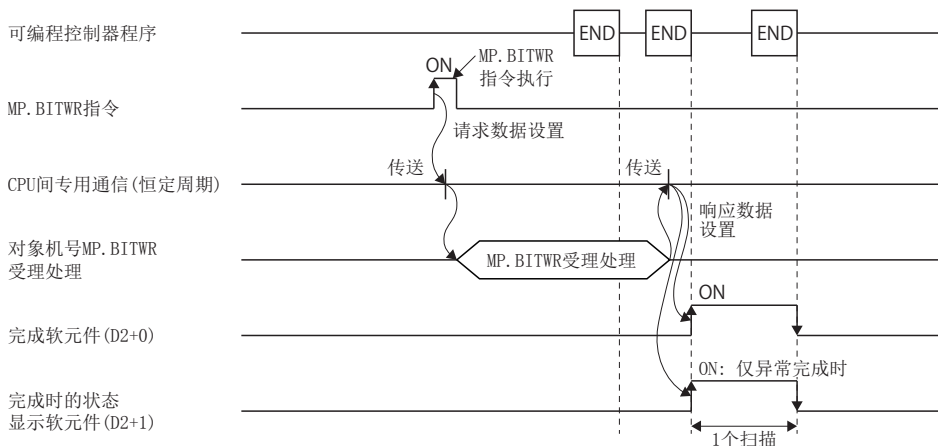
读取对象机号CPU的位软元件的情况下，应使用M(P). DDRD/D(P). DDRD指令。关于M(P). DDRD/D(P). DDRD指令的详细内容，请参阅以下手册。

📖 MELSEC iQ-R编程手册(指令/通用FUN/FB篇)

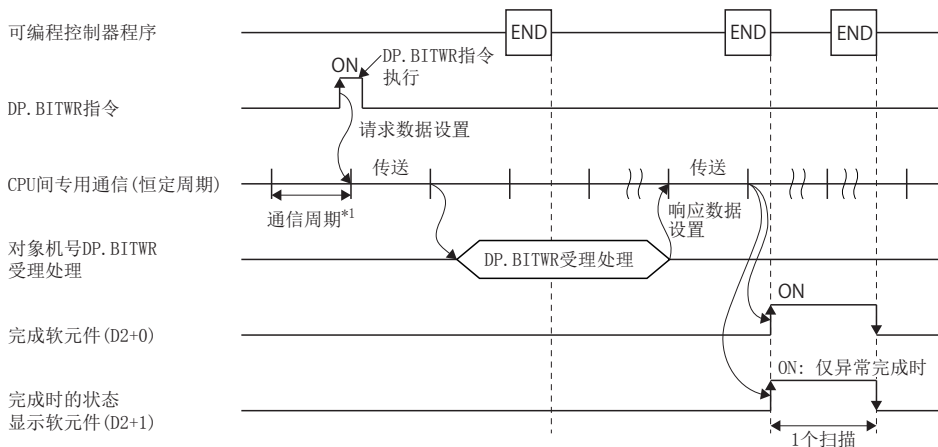
n 动作时机

执行了MP. BITWR指令、DP. BITWR指令时CPU之间的动作概要如下所示。

- MP. BITWR指令



- DP. BITWR指令



*1 在GX Works3的[系统参数]⇒[多CPU设置]中设置

出错

- 以下情况下将异常结束，完成状态存储软元件(D3)中指定的软元件中将存储出错代码。省略了完成状态存储软元件(D3)的情况下将不进行出错检测而变为无处理，应加以注意。

完成状态*1 (出错代码)(H)	出错原因	处理方法
0010	从可编程控制器CPU至运动CPU的指令请求超出了允许值。	确认程序，修改为正确的顺控程序。
2230	M(P).BITWR/D(P).BITWR指令中字符串数据(D1)不正确。	
2231	M(P).BITWR/D(P).BITWR指令中设置的位操作方法超出了0~1的范围。	

*1 0000H(正常)

- 以下情况下将运算出错，“最新自诊断出错(SM0)”将ON，出错代码将被存储到“最新自诊断出错代码(SD0)”中。

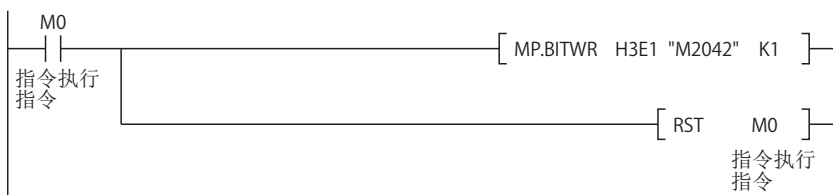
出错代码*1	出错原因	处理方法
2800	指定的其它机号CPU模块的起始输入输出编号(以16进制数4位表示时的前3位)超出了3E0H~3E3H的范围时。	确认程序，修改为正确的顺控程序。
2801	指定的其它机号CPU模块有误。 <ul style="list-style-type: none"> 指定了被设置为保留的机号时。 指定了未安装的机号时。 	
2802	其它机号CPU模块不支持M(P).BITWR/D(P).BITWR指令。	

*1 0000H(正常)

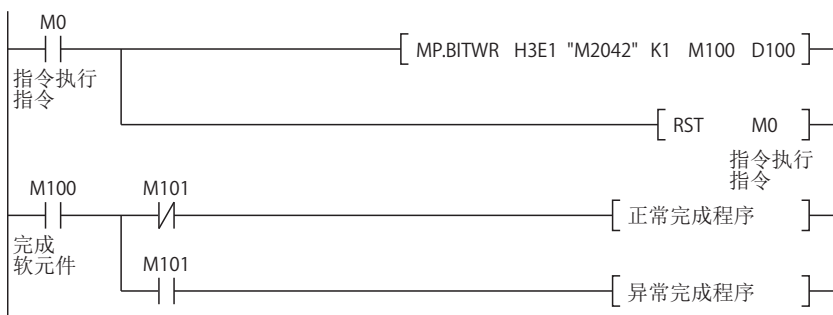
程序示例

n 以下为M0变为ON时将运动CPU(2号机)的M2042置为ON的程序

- (例1) 省略了完成软元件、完成状态情况下的程序



- (例2) 使用了完成软元件、完成状态情况下的程序



至其它机号CPU的中断指令：M(P).GINT/D(P).GINT

指令符号	执行条件	顺控程序*1
MP.BITWR DP.BITWR		
M.BITWR D.BITWR		

*1 □=指令符号(M(P).GINT: M, D(P).GINT: D)

设置数据

n 可用软元件

○：可以设置；△：部分可以设置

设置数据*3	可用软元件											
	内部软元件 (系统、用户)		文件寄存器		链接直接软元件 J□\□		模块访问软元件 U□\G□		变址 寄存器 Z□	常数		其它
	位	字	位	字	位	字	位	字		10进制 K、16进 制H	实数字 字符串	
(n1)		○		○						○		
(n2)		○		○						○		
(D1)*1	△*2		△*2									
(D2)*1		△*2		△*2								

*1 (D1)、(D2) 两方省略的情况下可以省略。

*2 不能使用局部软元件。

*3 对设置数据(n1)~(D2)可进行变址修饰。(常数除外)

n 内容、范围、设置方、数据类型

设置数据	设置内容	设置范围	设置方	数据类型
(n1)	对象机号CPU的起始输入输出编号÷16 实际指定的值如下所示。*2 2号机：3E1H；3号机：3E2H；4号机：3E3H	3E1H~3E3H	用户	BIN16位
(n2)	中断指针编号	0~15	用户	BIN16位
(D1)*1	完成软元件 • (D1+0)：通过指令的受理处理完成使其1个扫描ON的软元件 • (D1+1)：通过指令的受理异常完成使其1个扫描ON的软元件(异常完成时，D1+0也变为ON)	—	系统	位
(D2)*1	存储完成状态的软元件	—	系统	字

*1 (D1)、(D2) 两方省略的情况下可以省略。

*2 在多CPU构成中，不能将运动CPU设置为1号机。

功能

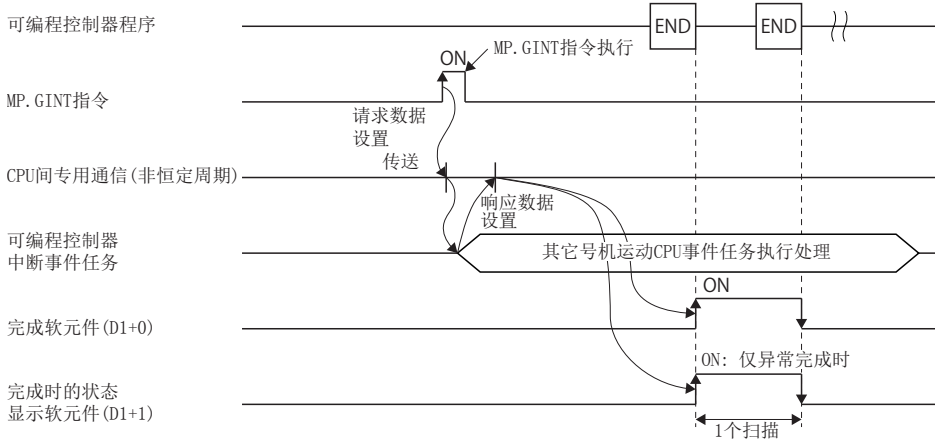
n 控制内容

- 对(n1)中指定的运动CPU，发生可编程控制器CPU的(n2)中指定的编号的“事件任务的可编程控制器中断”。
- 运动CPU侧为DI(中断禁止)中的情况下，不能执行事件处理。执行事件处理时，应预先执行EI(中断允许)指令。

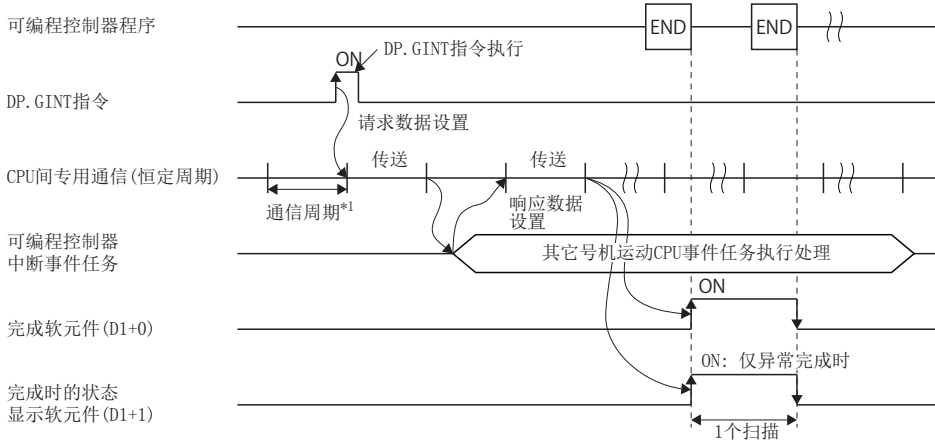
n 动作时机

执行了MP. GINT指令、DP. GINT指令时CPU之间的动作概要如下所示。

• MP. GINT指令



• DP. GINT指令



*1 在GX Works3的[系统参数]⇒[多CPU设置]中设置

出错

• 以下情况下将异常结束，完成状态存储软元件(D2)中指定的软元件中将存储出错代码。省略了完成状态存储软元件(D2)的情况下将不进行出错检测而变为无处理，应加以注意。

完成状态*1 (出错代码)(H)	出错原因	处理方法
0010	从可编程控制器CPU至运动CPU的指令请求超出了允许值。	确认程序，修改为正确的顺控程序。
2282	M(P). GINT/D(P). GINT指令中设置的断指No. 超出了0~15的范围。	

*1 0000H(正常)

• 以下情况下将运算出错，“最新自诊断出错(SM0)”将ON，出错代码将被存储到“最新自诊断出错代码(SD0)”中。

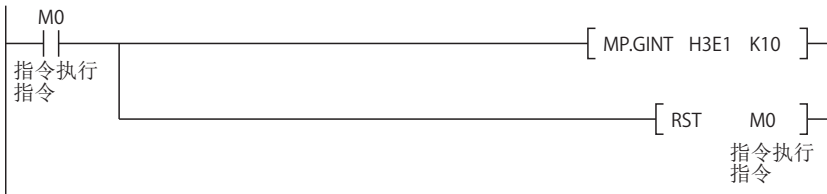
出错代码(H)*1	出错原因	处理方法
2800	指定的其它机号CPU模块的起始输入输出编号(以16进制数4位表示时的前3位)超出了3E0H~3E3H的范围时。	确认程序，修改为正确的顺控程序。
2801	指定的其它机号CPU模块有误。 • 指定了被设置为保留的机号时。 • 指定了未安装的机号时。	

*1 0000H(正常)

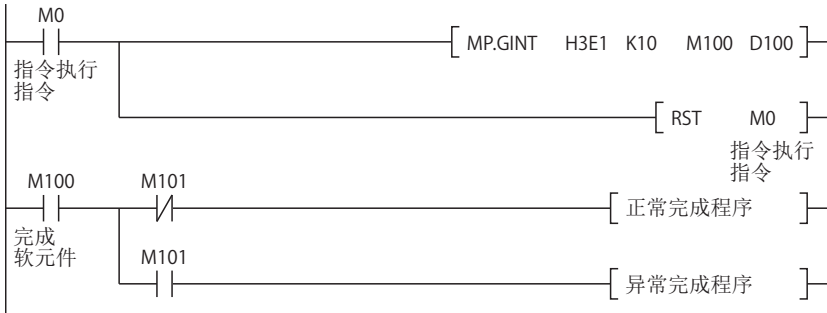
程序示例

n 以下为M0变为ON时使运动CPU(2号机)发生中断指针编号10的中断的程序

- (例1) 省略了完成软元件、完成状态情况下的程序



- (例2) 使用了完成软元件、完成状态情况下的程序



2.3 注意事项

运动专用顺控程序指令中使用的CPU缓冲存储器地址

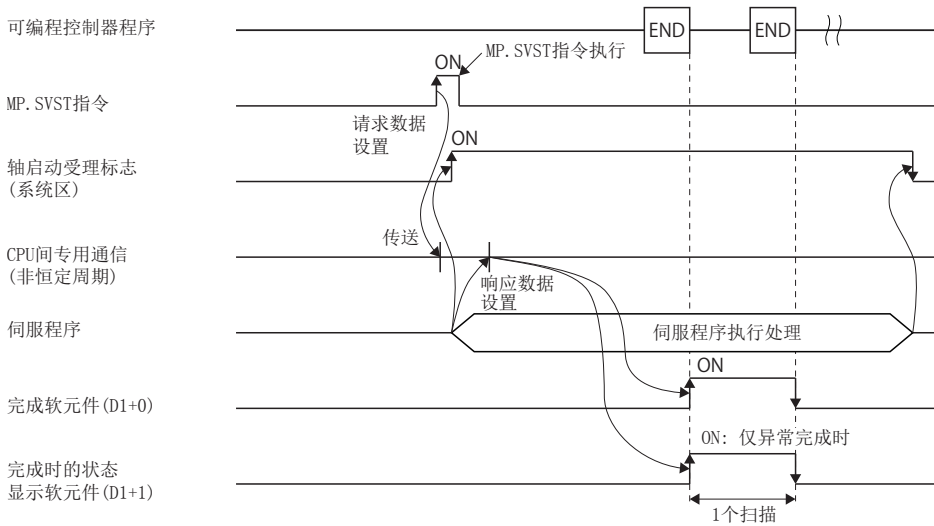
相当于定位专用信号的信息将被输出到运动CPU的CPU缓冲存储器上的系统区域(U3E□\G0~2k点)中。执行运动专用顺控程序指令时，应对本区域的信号采取互锁。

启动受理标志(系统区域)

各标志的状态将被存储到下述地址中。

CPU缓冲存储器地址 (10进制数)	内容																									
204H(516) 205H(517) 206H(518) 207H(519)	<p>启动受理标志按64轴存储在对应的各位中。 实际位设置如下所示。</p> <ul style="list-style-type: none"> • R64MTCPU: J1~J64 • R32MTCPU: J1~J32 • R16MTCPU: J1~J16 <p>OFF: 可以启动受理 ON: 不能启动受理</p> <table border="1"> <thead> <tr> <th></th> <th>b15</th> <th>b2</th> <th>b1</th> <th>b0</th> </tr> </thead> <tbody> <tr> <td>204H(516)地址编码</td> <td>J16</td> <td>••••••••</td> <td>J2</td> <td>J1</td> </tr> <tr> <td>205H(517)地址编码</td> <td>J32</td> <td>••••••••</td> <td>J18</td> <td>J17</td> </tr> <tr> <td>206H(518)地址编码</td> <td>J48</td> <td>••••••••</td> <td>J34</td> <td>J33</td> </tr> <tr> <td>207H(519)地址编码</td> <td>J64</td> <td>••••••••</td> <td>J50</td> <td>J49</td> </tr> </tbody> </table>		b15	b2	b1	b0	204H(516)地址编码	J16	••••••••	J2	J1	205H(517)地址编码	J32	••••••••	J18	J17	206H(518)地址编码	J48	••••••••	J34	J33	207H(519)地址编码	J64	••••••••	J50	J49
	b15	b2	b1	b0																						
204H(516)地址编码	J16	••••••••	J2	J1																						
205H(517)地址编码	J32	••••••••	J18	J17																						
206H(518)地址编码	J48	••••••••	J34	J33																						
207H(519)地址编码	J64	••••••••	J50	J49																						
20EH(526) 20FH(527) 210H(528) 211H(529)	<p>指令生成轴的启动受理标志按64轴存储在对应的各位中。 实际位设置如下所示。</p> <ul style="list-style-type: none"> • R64MTCPU: J1~J64 • R32MTCPU: J1~J32 • R16MTCPU: J1~J16 <p>OFF: 可以启动受理 ON: 不能启动受理</p> <table border="1"> <thead> <tr> <th></th> <th>b15</th> <th>b2</th> <th>b1</th> <th>b0</th> </tr> </thead> <tbody> <tr> <td>20EH(526)地址编码</td> <td>J16</td> <td>••••••••</td> <td>J2</td> <td>J1</td> </tr> <tr> <td>20FH(527)地址编码</td> <td>J32</td> <td>••~••••••</td> <td>J18</td> <td>J17</td> </tr> <tr> <td>210H(528)地址编码</td> <td>J48</td> <td>••~••••••</td> <td>J34</td> <td>J33</td> </tr> <tr> <td>211H(529)地址编码</td> <td>J64</td> <td>••~••••~••</td> <td>J50</td> <td>J49</td> </tr> </tbody> </table>		b15	b2	b1	b0	20EH(526)地址编码	J16	••••••••	J2	J1	20FH(527)地址编码	J32	••~••••••	J18	J17	210H(528)地址编码	J48	••~••••••	J34	J33	211H(529)地址编码	J64	••~••••~••	J50	J49
	b15	b2	b1	b0																						
20EH(526)地址编码	J16	••••••••	J2	J1																						
20FH(527)地址编码	J32	••~••••••	J18	J17																						
210H(528)地址编码	J48	••~••••••	J34	J33																						
211H(529)地址编码	J64	••~••••~••	J50	J49																						

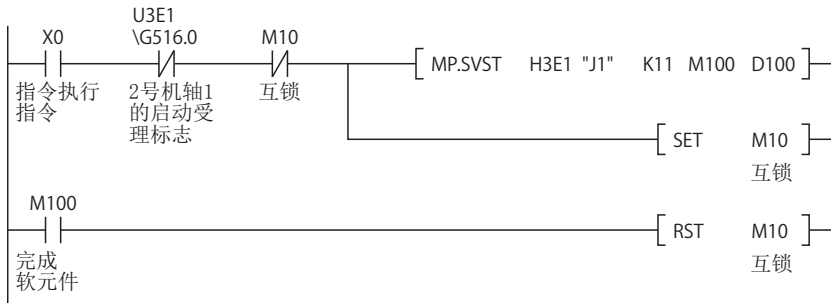
如下图所示，运动CPU中指令受理后启动受理标志将被设置。



从可编程控制器CPU中执行指令后至运动CPU中指令受理完成为止启动受理标志不变为ON。因此，为了避免在同一轴中重复启动出错，避免对同一轴执行以下运动专用顺控程序指令，应根据需要通过用户软元件置入互锁条件。

n 程序示例

- 以下为X0处于ON期间，对运动CPU(2号机)的轴1进行伺服程序No. 11的连续启动的程序



运动专用顺控程序指令的使用块数

多CPU之间专用指令传送区域块数

n CPU之间专用指令传送区域数

对于运动专用顺控程序指令中使用的多CPU之间专用指令传送区域，通过以16字为最小单位的块进行管理。各机号中使用的块数如下表所示。

多CPU个数	每个指令执行对象机号的CPU之间专用指令传送区域数
2个	599块
3个	299块
4个	199块

n 运动专用顺控程序指令的使用块数

对于各运动专用顺控程序指令，通过可编程控制器CPU执行了指令后，在完成软元件变为ON之前，使用下述块数的CPU之间专用指令传送区域。运动专用顺控程序指令中使用的块数如下表所示。

指令	使用块数
M(P).SFCS/D(P).SFCS	1
M(P).SVST/D(P).SVST	$1 + \lceil (1 + \lceil j + 2 \rceil) \div 16 \rceil^{*1}$
M(P).SVSTD/D(P).SVSTD	$1 + \lceil (n + \lceil m + 2 \rceil + 1) \div 16 \rceil^{*1}$
M(P).CHGA/D(P).CHGA	2
M(P).CHGAS/D(P).CHGAS	2
M(P).CHGV/D(P).CHGV	2
M(P).CHGVS/D(P).CHGVS	2
M(P).CHGT/D(P).CHGT	2
M(P).DDWR/D(P).DDWR	3^{*2}
M(P).DDRD/D(P).DDRD	2^{*3}
M(P).BITWR/D(P).BITWR	$1 + \lceil (1 + \lceil j + 2 \rceil) \div 16 \rceil^{*1}$
M(P).GINT/D(P).GINT	1

*1 通过计算公式计算。

j=(S1)中指定的轴No.的字符数

n=(S1+1)中指定的定位数据点数

m=(D1)中指定的字符串数据的字符数

$\lceil \rceil$ 内=小数点以下进位

*2 写入数据数为11字以下的情况下，使用块数将变为2。

*3 读取数据数为13字以下的情况下，使用块数将变为2。

n 动作示例

在各自的完成软元件变为ON之前同时执行了M(P).SVST指令(字符数8字符)的12指令，M(P).DDWR指令(写入数据数12字)的12指令的情况下

使用块数如下：

$2(M(P).SVST指令使用块数) \times 12(M(P).SVST指令发行数) + 2(M(P).DDWR指令使用块数) \times 12(M(P).DDWR指令发行数) = 48$ (合计使用块数)。

多CPU之间专用指令同时发行数

用于与各CPU通信的CPU之间专用指令传送区域中，没有多CPU之间专用指令最大使用块数设置(执行指令的可编程控制器CPU的特殊寄存器(SD796~SD799))的设置值以上的空余块的情况下，将变为无法受理运动专用顺控程序指令状态(执行数超过允许值)。此时，如果对对象机号执行运动专用指令，执行时指定的完成状态中将被设置异常完成状态0010H。省略了完成软元件的情况下将变为无处理，应加以注意。

通过使用多CPU之间专用指令使用块信息(执行指令的可编程控制器CPU的特殊继电器(SM796~SM799))，可以置入防止执行数超出允许值的互锁条件。

- 可编程控制器CPU的特殊继电器

软元件编号	名称	内容	详细内容	设置方
SM796	多CPU之间专用指令使用块信息 (1号机用)	OFF: 块预留 ON: 未能预留SD796中设置的块数	多CPU之间专用指令中使用的专用指令传送区域的剩余块数小于SD796~SD799中指定的块数时将变为ON。 执行指令时ON。END处理时区域有空余时OFF。	系统(指令执行时/END处理时)
SM797	多CPU之间专用指令使用块信息 (2号机用)	OFF: 块预留 ON: 未能预留SD797中设置的块数		
SM798	多CPU之间专用指令使用块信息 (3号机用)	OFF: 块预留 ON: 未能预留SD798中设置的块数		
SM799	多CPU之间专用指令使用块信息 (4号机用)	OFF: 块预留 ON: 未能预留SD799中设置的块数		

- 可编程控制器CPU的特殊寄存器

软元件编号	名称	内容	详细内容	设置方
SD796	多CPU之间专用指令最大使用块数设置 (1号机用)	根据多CPU系统配置的CPU个数其范围*1如下所示。 2个配置时: 2~599 3个配置时: 2~299 4个配置时: 2~199 (默认: 2)	指定多CPU之间专用指令的最大使用块数。对对象机号执行了多CPU之间专用指令时，专用指令传送区域的空余块数小于本寄存器的设置值的情况下，将SM796~SM799置为ON。作为多CPU之间专用指令的连续执行互锁信号使用。	用户 (RUN后1个扫描时)
SD797	多CPU之间专用指令最大使用块数设置 (2号机用)			
SD798	多CPU之间专用指令最大使用块数设置 (3号机用)			
SD799	多CPU之间专用指令最大使用块数设置 (4号机用)			

*1 设置了超出范围的值的的情况下，将以多CPU系统配置的各范围的最大值执行动作。

要点

在执行运动专用顺控程序指令的时刻，

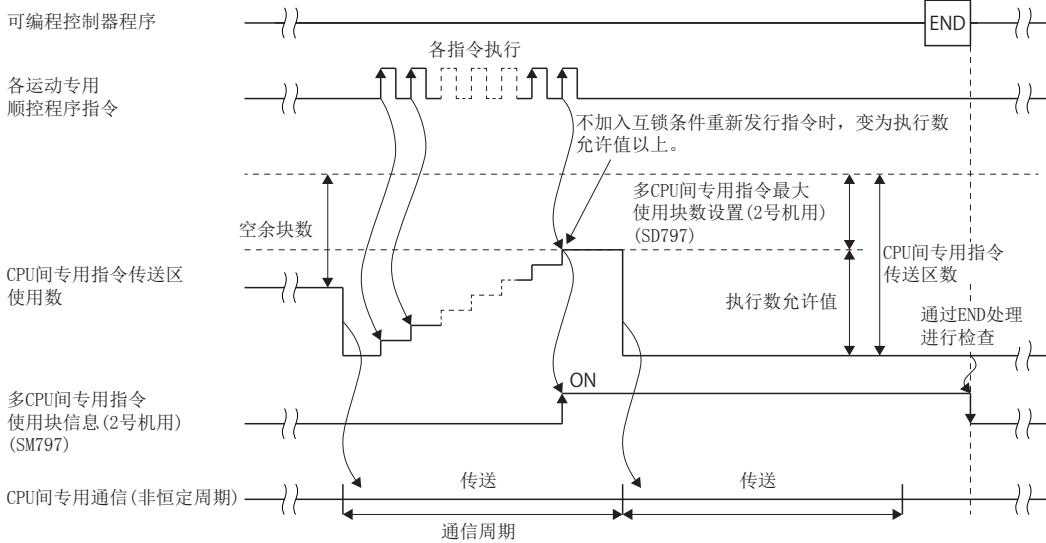
多CPU之间专用指令最大使用块数设置(SD796~799) ≤ 专用指令传送区域的空余块数 < 指令中使用的块数

的情况下，在该扫描中不执行指令(变为无处理)，在下1个扫描中再次执行指令。

通过“多CPU之间专用指令使用块信息(SM796~SM799)”置入互锁条件的情况下，应在“多CPU之间专用指令最大使用块数设置(SD796~SD799)”中设置大于执行指令中使用的块数的值。

n 动作时机

执行各运动专用指令后，多CPU之间专用指令使用块信息变为ON时的动作概要如下所示。



n 动作示例

多CPU个数为4个时将多个D(P). DDWR指令(传送数据数80字)在各自的完成软元件ON之前依次执行的情况下将各项目的使用块数进行以下设置后，

- CPU之间专用指令传送区域数：199块(初始值)
- 多CPU之间专用指令最大使用块数设置2号机(SD797)：2(初始值)
- D(P). DDWR指令使用块数：6

通过在恒定周期通信周期(0.888ms)内发行33个D(P). DDWR指令，使用块数如下所示，

$$6(\text{D(P). DDWR指令使用块数}) \times 33(\text{D(P). DDWR指令发行数}) = 198(\text{合计使用块数})$$

空余块如下所示，

$$199(\text{CPU之间专用指令传送区域数}) - 198(\text{合计使用块数}) = 1(\text{空余块数})$$

由此，

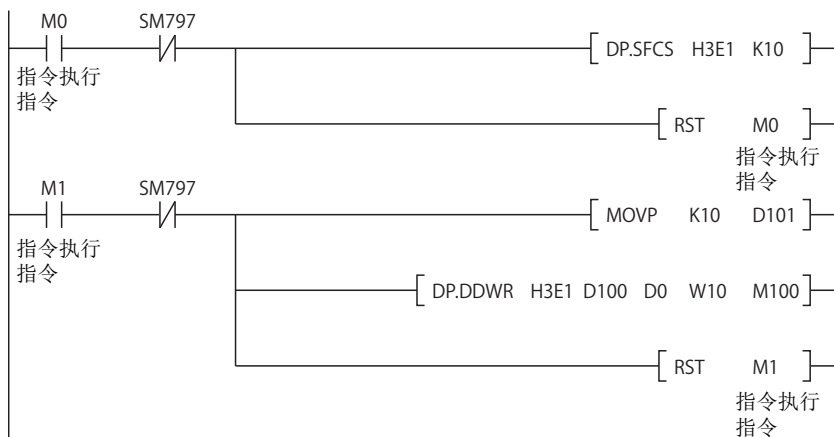
$$1(\text{空余块数}) < 2(\text{多CPU之间专用指令最大使用块数设置(2号机用)(SD797)})$$

由于空余块数小于“多CPU之间专用指令最大使用块数设置(2号机用)(SD797)”，因此“多CPU之间专用指令使用块信息(2号机用)(SM797)”变为ON。

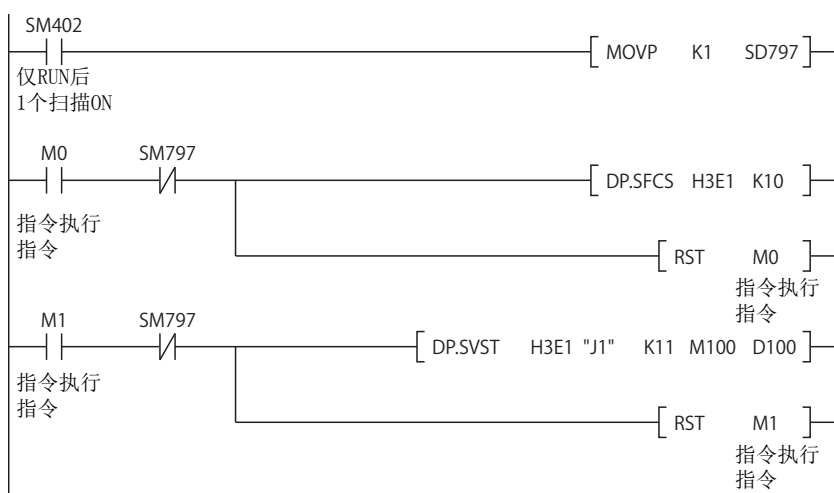
在此状态下执行新指令时，执行数将超过允许值，但通过将“多CPU之间专用指令使用块信息(2号机用)(SM797)”作为互锁条件置入，可以避免执行数超出允许值。

n 程序示例

- 以下为执行DP.DDWR(使用块数2)的情况下, 将SD797设置为2(初始值), 将SM797作为互锁条件置入的程序



- 以下为不执行D(P).DDWR/D(P).DDRD的情况下, 将SD797设置为1, 将SM797作为互锁条件置入的程序



多CPU之间专用指令的同时受理允许数

在运动CPU中, 可同时受理下表中所示的指令数。

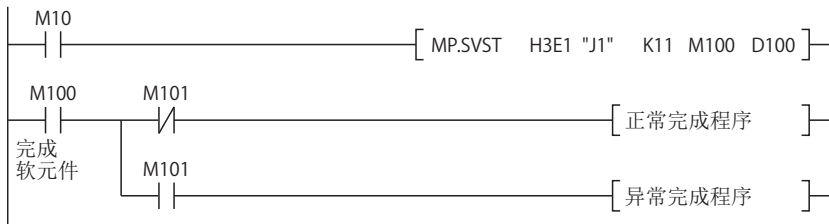
指令	同时受理允许数
M(P).SFCS/D(P).SFCS	64
M(P).SVST/D(P).SVST	合计256个指令
M(P).SVSTD/D(P).SVSTD	
M(P).CHGA/D(P).CHGA	
M(P).CHGAS/D(P).CHGAS	
M(P).CHGV/D(P).CHGV	无限制*1 (最后执行的指令的指令速度及转矩限制值有效)
M(P).CHGVS/D(P).CHGVS	
M(P).CHGT/D(P).CHGT	无限制*1 (按发行顺序处理)
M(P).DDWR/D(P).DDWR	
M(P).DDRD/D(P).DDRD	
M(P).BITWR/D(P).BITWR	
M(P).GINT/D(P).GINT	

*1 同时执行数取决于可编程控制器CPU侧的CPU之间专用指令空余块数。

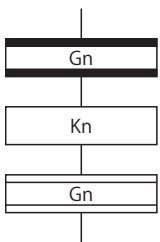
即使CPU之间专用指令传送区域中有充足可用空间, 通过可编程控制器CPU执行了大于上述指令数的指令的情况下, 运动CPU也不能受理指令。此时完成状态信息中将被设置(2100H), 变为异常完成。

运动专用顺控程序指令的执行

- 在恒定周期执行型程序及中断程序中也可执行运动专用顺控程序指令。但是，完成软元件将通过脉冲被输出。指定了完成软元件的情况下，如果在指令完成的扫描中未执行程序，通过指令完成的完成软元件的1个扫描ON将无法识别。



- 对于局部软元件及各程序的文件寄存器，不能用于下述软元件。
 - 各指令的完成软元件及完成状态
 - M(P). DDRD/D(P). DDRD指令的D1(存储读取数据的本机的起始软元件)
- 在运动CPU中执行运算控制步(Fn/FSn)的运动专用函数、伺服程序(Kn)的情况下，应根据需要使用以下的WAIT转换(Gn)，在用户程序中置入互锁条件。



完成状态信息

运动专用顺控程序指令完成时，完成状态中存储的代码如下表所示。省略了完成状态存储软元件的情况下将不检测出错而变为无处理，应加以注意。

完成状态 (出错代码) (H)	出错原因
0	正常完成
0010	从可编程控制器CPU至运动CPU的指令请求超出了允许值*2。
1001*1	指定的软元件是CPU中不能使用的软元件。或者超出了软元件范围。
1080*1	M(P). DDWR/D(P). DDWR指令中设置的写入数据点数不正确。
1081*1	M(P). DDRD/D(P). DDRD指令中设置的读取数据点数不正确。
2000*1	指定了运动CPU中无法解读的指令。
2100*1	n 使用M(P). SFCS/D(P). SFCS指令时 同时通过M(P). SFCS/D(P). SFCS进行的从可编程控制器CPU至运动CPU的指令请求超过了65个指令，运动CPU无法处理。 n 使用M(P). SVST/D(P). SVST/M(P). SVSTD/D(P). SVSTD/M(P). CHGA/D(P). CHGA/M(P). CHGAS/D(P). CHGAS指令时 从可编程控制器CPU至运动CPU的指令请求M(P). SVST/D(P). SVST/M(P). SVSTD/D(P). SVSTD/M(P). CHGA/D(P). CHGA/M(P). CHGAS/ D(P). CHGAS合计同时有257个指令以上，运动CPU无法处理。
2200*1	启动的运动SFC程序No. 超出了0~255的范围。
2201*1	执行的伺服程序No. 超出了0~4095的范围。
2202*1	M(P). SVST/D(P). SVST指令中设置的轴No. 不正确。
2203*1	M(P). CHGA/D(P). CHGA指令中设置的轴No. 不正确。
2204*1	M(P). CHGV/D(P). CHGV指令中设置的轴No. 不正确。
2205*1	M(P). CHGT/D(P). CHGT指令中设置的轴No. 不正确。
2207*1	M(P). CHGAS/D(P). CHGAS指令中设置的轴No. 不正确。
2208*1	M(P). CHGVS/D(P). CHGVS指令中设置的轴No. 不正确。
2209*1	M(P). SVSTD/D(P). SVSTD指令中设置的轴No. 不正确。
2220*1	M(P). SVSTD/D(P). SVSTD指令的指令区域不足。(通过D(P). SVSTD/M(P). SVSTD指令动作中的指令超过了128个指令)
2221*1	M(P). SVSTD/D(P). SVSTD指令的定位数据区域点数不正确。
2222*1	M(P). SVSTD/D(P). SVSTD指令的指令仅为NOP或停止轴设置。
2224*1	在M(P). SVSTD/D(P). SVSTD指令中必须进行定位数据项目指定的指令中，未进行项目设置。
2225*1	M(P). SVSTD/D(P). SVSTD指令中字符串数据(D1)不正确。
2230*1	M(P). BITWR/D(P). BITWR指令中字符串数据(D1)不正确。
2231*1	M(P). BITWR/D(P). BITWR指令中设置的位操作方法超出了0~1的范围。
2282*1	M(P). GINT/D(P). GINT指令中设置的中断指针No. 超出了0~15的范围。

*1 运动CPU中检测出出错代码。

*2 允许值根据CPU的安装个数而有所不同。

指令的执行顺序

从可编程控制器CPU对运动CPU发送数据后，使用该数据进行控制的情况下有以下方法。

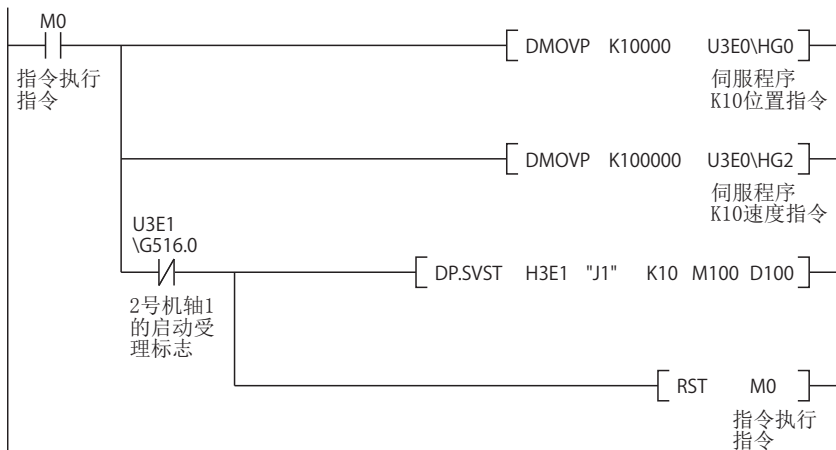
在CPU缓冲存储器(恒定周期通信区域)中写入数据后执行的方法

将数据从可编程控制器CPU中写入本机的CPU缓冲存储器(恒定周期通信区域)后，执行运动专用顺控程序指令时，可以使用该数据进行控制。

n 程序示例

以下为从可编程控制器CPU(1号机)对CPU缓冲存储器(恒定周期通信区域)U3E0\HG0~U3E0\HG3写入数据后，通过DP.SVST指令启动伺服程序(定位)的程序

可编程控制器CPU顺控程序



运动CPU侧伺服程序

```
[ K10: 实 ]
1 INC-1
轴 1,      U3E0\HG0 μm
速度      U3E0\HG2 mm/min
```

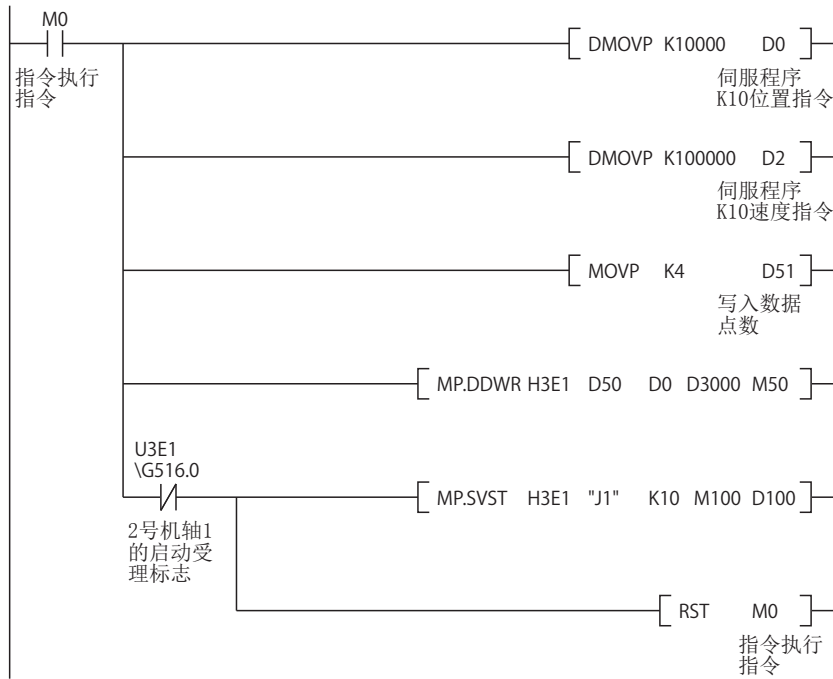
通过M(P). DDWR/D(P). DDWR指令写入数据后执行的方法

通过M(P). DDWR/D(P). DDWR指令将数据从可编程控制器CPU中写入运动CPU后，执行运动专用顺控程序指令时，可以使用该数据进行控制。

n 程序示例

以下为从可编程控制器CPU(1号机)对运动CPU(2号机)的D3000~D3003通过MP. DDWR指令写入数据后，通过MP. SVST指令启动伺服程序(定位)的程序

可编程控制器CPU侧顺控程序



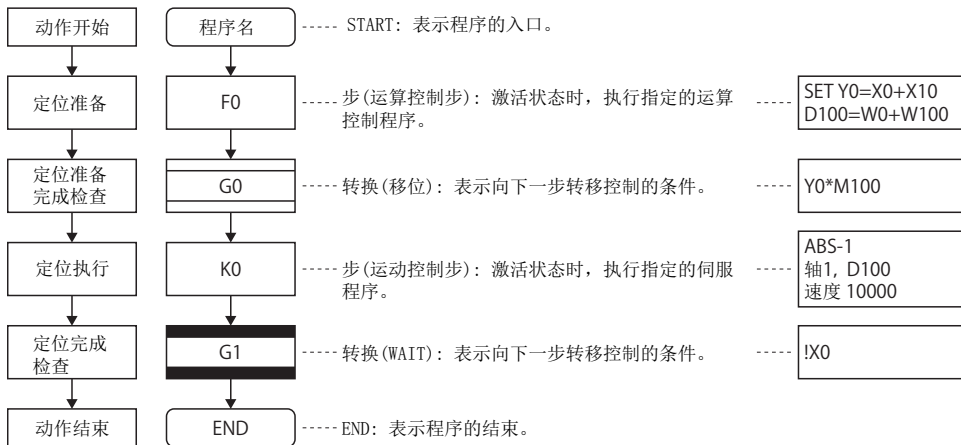
运动CPU侧伺服程序

[K10: 实]	
1 INC-1	
轴 1,	D3000 μm
速度	D3002 mm/min

3 运动SFC程序

3.1 运动SFC程序的构成

运动SFC程序由如下所示的START、步、转换、END等组合构成。



启动的上述运动SFC程序的动作如下所示。

1. 步 (F0) 变为激活状态, 执行步 (F0) 中指定的动作 (定位准备)。这样处于激活状态的步称为激活步。
2. 对转换 (G0) 中指定的条件是否成立 (能否启动定位程序) 进行检查, 通过条件成立激活步 (F0) 将变为非激活状态, 下一个步 (K0) 将变为激活状态。(启动伺服程序K0)
3. 在转换 (G1) 中, 对步 (K0) 的动作完成 (伺服程序K0的定位完成) 进行检查, 通过动作完成 (条件成立) 将控制转移至下一个步。
4. 按上述 (1)~(3) 所示通过激活步转移执行控制, 通过END结束。

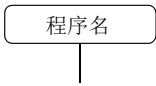

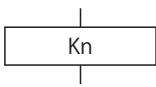
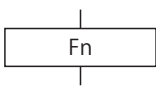

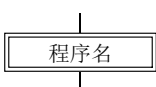
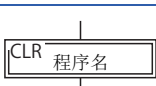
关于上述运动SFC程序的执行时机的详细内容, 请参阅任务动作。(☞ 266页 任务动作)

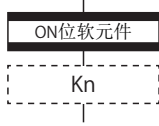
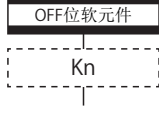
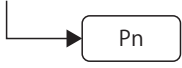
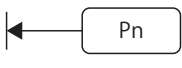
要点

全部运动SFC程序合计中, 同时变为激活步的步数为256个以下。超过256个的情况下, 将发生轻度出错 (SFC) (出错代码: 33FEH)。运动SFC程序的各符号如下所示。
F/FS: 运算控制; K: 定位控制; G: 判断

3.2 运动SFC图符号一览

运动SFC程序的构成要素及部件如下所示。运动SFC程序通过这些部件通过有向线连接，表示动作顺序、转移控制。

分类	名称	符号 (代码容量(字节))	功能
程序开始/ 结束	START	 (0)	<ul style="list-style-type: none"> 将程序的入口以程序名表示。 进行子程序调用时，指定该程序名。 1个程序中仅限1个。
	END	 (8)	<ul style="list-style-type: none"> 表示程序的结束(出口)。 进行了子程序调用的情况下，返回至调用源程序。 1个程序内可设置多个，也可无设置。
步	运动控制步	 (8)	进行伺服程序Kn(K0~K4095)的启动。
	1次执行型运算控制步	 (8)	执行1次运算控制程序Fn(F0~F4095)。
	扫描执行型运算控制步	 (8)	重复执行运算控制程序FSn(FS0~FS4095)直至下一个转移条件成立为止。
	子程序调用/启动步	 (8)	<ul style="list-style-type: none"> GSUB的后面为WAIT的情况下，变为“子程序调用”并将控制转移至指定的程序。通过执行END，将控制返回至调用源程序。 GSUB的后面为WAIT以外的情况下，变为“子程序启动”，启动指定程序后，转移至下一步(后面)。同时执行启动源程序及启动目标程序，启动目标程序通过执行END而结束。
	清除步	 (8)	<ul style="list-style-type: none"> 中止执行中的指定程序，结束。结束后，通过再次启动，从初始(开始步)开始。 指定程序为“子程序调用”中的情况下，子程序的执行也将中止。 指定程序为“子程序启动”后的情况下，子程序的执行不中止。 对进行了“子程序调用”的子程序执行了清除的情况下，指定子程序的执行将中止，返回至调用源程序，转移至下一步。

分类	名称	符号 (代码容量(字节))	功能
转换	移位(预读转移)	 (8)	<ul style="list-style-type: none"> 之前为运动控制步的情况下, 不等待运动的动作完成, 通过转移条件Gn(G0~G4095)成立转移至下一个步。 之前为运算控制步的情况下, 执行运算后, 通过转移条件成立转移至下一个步。 之前为子程序调用/启动步的情况下, 不等待子程序的动作完成, 通过转移条件成立转移至下一个步。
	WAIT	 (8)	<ul style="list-style-type: none"> 之前为运动控制步的情况下, 等待运动的动作完成, 通过转移条件Gn(G0~G4095)成立转移至下一个步。 之前为运算控制步的情况下, 执行运算后, 通过转移条件成立转移至下一个步。(变为与移位相同的动作。) 之前为子程序调用/启动步的情况下, 等待子程序的动作完成, 通过转移条件成立转移至下一个步。
	WAITON	 (14)	<ul style="list-style-type: none"> 进行下一个运动控制步的启动准备, 通过指定位软元件ON立即发出指令。 必须与运动控制步以1对1进行成对设置。
	WAITOFF	 (14)	<ul style="list-style-type: none"> 进行下一个运动控制步的启动准备, 通过指定位软元件OFF立即发出指令。 必须与运动控制步以1对1进行成对设置。
	移位Y/N	 (成立时)Y (不成立时)N	<ul style="list-style-type: none"> 之前为运动控制步的情况下, 不等待运动的动作完成, 通过转移条件Gn(G0~G4095)成立转移至下一个步, 条件不成立时转移至右侧连接的步。 之前为运算控制步的情况下, 执行运算后, 通过转移条件成立转移至下一个步, 条件不成立时转移至右侧连接的步。 之前为子程序调用/启动步的情况下, 不等待子程序的动作完成, 通过转移条件成立转移至下一个步, 条件不成立时转移至右侧连接的步。
	WAIT Y/N	 (成立时)Y (不成立时)N	<ul style="list-style-type: none"> 之前为运动控制步的情况下, 等待运动的动作完成, 转移条件Gn(G0~G4095)成立转移至下一个步, 条件不成立时转移至右侧连接的步。 之前为运算控制步的情况下, 执行运算后, 通过转移条件成立转移至下一个步, 条件不成立时转移至右侧连接的步。(变为与移位相同的动作。) 之前为子程序调用/启动步的情况下, 等待子程序的动作完成, 通过转移条件成立转移至下一个步, 条件不成立时转移至右侧连接的步。
跳转	跳转	 (14)	跳转至本程序内指定的指针Pn(P0~P16383)。
指针	指针	 (8)	<ul style="list-style-type: none"> 表示跳转目标指针(标签)。 可以设置为步、转换、分支点、合并点。 1个程序中可以设置P0~P16383。即使与其它程序的编号重复也没有关系。

3.3 分支·合并图一览

通过运动SFC图指定步、转换的流程的分支·合并的模式如下所示。

分类	名称 (代码容量(字节))	运动SFC图符号	功能
基本型	串联转移 (各符号容量)		<ul style="list-style-type: none"> 将串联连接的步、转换的各处理按从上至下的顺序执行。 即使步与转换交替排列也没有关系。 省略了转换的情况下，进行无条件的转移处理。
	选择分支 ($(\text{分支数}+2)\times 10$)		<ul style="list-style-type: none"> 执行分支之前的步或转换后，执行最先转移条件成立的路径。 选择分支的分支目标各起始必须为转换，限于全部移位或全部WAIT。(移位·WAIT同时存在的情况下，将变为并联分支。)
	选择合并 (8)		<ul style="list-style-type: none"> 执行通过选择分支进行了分支后的路径后，转移至合并点。 合并之前、之后可以是步或转换。
	并联分支 ($\text{分支数}\times 22+\text{合并点数}\times 2+12$)		<ul style="list-style-type: none"> 同时执行并联连接的多个路径(步)。 并联分支的分支目标各起始可以是步或转换。
	并联合并 (8)		<ul style="list-style-type: none"> 将通过并联分支进行了分支的各路径的执行完成通过合并点进行等待，通过全部路径的执行完成转移至下一个。 合并之前、之后可以是步或转换。 合并之前为FS步的情况下，等待中也执行扫描。等待完成后不执行扫描。
	跳转转移 (各符号容量)	<p>[普通跳转]</p> <p>[合并跳转]</p>	<p>[普通跳转]</p> <ul style="list-style-type: none"> 执行之前的步或转换后，将执行转移至本程序内指定的指针Pn。 跳转目标可以是步或转换。 从FS步跳转至转换的情况下，跳转目标的转移条件成立等待中也执行扫描。 <p>[合并跳转]</p> <ul style="list-style-type: none"> 并联分支后，跳转至并联分支内的其它路径的情况下，将变为“合并跳转”，通过跳转目标进行等待。

根据组合基本型的分支·合并，有下述应用型，定义按照基本型。

分类	名称	运动SFC图符号	功能
应用型	选择分支 并联分支		选择分支的后面可以设置并联分支。
	并联合并 选择合并		<ul style="list-style-type: none"> 选择分支→并联分支情况下的并联合并的合并点与选择合并点可能相同。但是，在运动SFC图中，如左图所示按并联合并→选择合并的顺序表示。 在此情况下，并联合并点 (PAEm) 与选择合并点 (IFEm) 之间不能设置指针 (Pn)。
	并联分支 选择分支		并联分支的后面可以设置选择分支。
	选择合并 并联合并		<ul style="list-style-type: none"> 并联分支→选择分支情况下的选择合并的合并点与并联合并点可能相同。但是，在运动SFC图中，如左图所示按选择合并→并联合并的顺序表示。 在此情况下，选择合并点 (IFEm) 与并联合并点 (PAEm) 之间不能设置指针 (Pn)。
	选择分支 选择分支		选择分支的后面可以设置选择分支。
	选择合并 选择合并		<ul style="list-style-type: none"> 选择分支→选择分支情况下的2个选择合并点可能相同。但是，在运动SFC图中，如左图所示按选择合并→选择合并的顺序表示。 在此情况下，选择合并点 (IFEm+1) 与选择合并点 (IFEm) 之间不能设置指针 (Pn)。
	并联分支 并联分支		<ul style="list-style-type: none"> 并联分支的后面可以设置并联分支。 并联分支的嵌套最多为4重。
	并联合并 并联合并		<ul style="list-style-type: none"> 并联分支→并联分支情况下的2个并联合并点可能相同。但是，在运动SFC图中，如左图所示按并联合并→并联合并的顺序表示。 在此情况下，并联合并点 (PAEm+1) 与并联合并点 (PAEm) 之间不能设置指针 (Pn)。

分类	名称	运动SFC图符号	功能
应用型	选择合并 并联分支		<ul style="list-style-type: none"> 选择合并点与并联分支点可以为同一点。但是，在运动SFC图中，如左图所示按选择合并→并联分支的顺序表示。 在此情况下，选择合并点(IFE_m)与并联分支点(PAB_m)之间可以设置指针(P_n)。
	并联合并 选择分支		<ul style="list-style-type: none"> 并联合并点与选择分支点可以为同一点。但是，在运动SFC图中，如左图所示按并联合并→选择分支的顺序表示。 通过并联合并点进行等待，转移至选择分支。 在此情况下，并联合并点(PAE_m)与选择分支点(IFB_m)之间可以设置指针(P_n)。
	选择合并 选择分支		<ul style="list-style-type: none"> 选择合并点与选择分支点可以为同一点。但是，在运动SFC图中，如左图所示按选择合并→选择分支的顺序表示。 在此情况下，选择合并点(IFE_m)与选择分支点(IFB_{m+1})之间可以设置指针(P_n)。
	并联合并 并联分支		<ul style="list-style-type: none"> 并联合并点与并联分支点可以为同一点。但是，在运动SFC图中，如左图所示按并联合并→并联分支的顺序表示。 通过并联合并点进行等待，转移至并联分支。 在此情况下，并联合并点(PAE_m)与并联分支点(PAB_{m+1})之间可以设置指针(P_n)。

3.4 运动SFC程序名

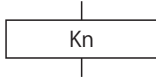
对运动SFC程序No. 0~No. 255分别设置“运动SFC程序名”。运动SFC程序名在半角16字符(全角8字符)以内设置。在“子程序调用/启动步(GSUB)”、“清除步(CLR)”中指定该运动SFC程序名。

要点

- 对运动SFC程序可在No. 0~No. 255内设置任意No.。没有具有特殊作用的特别No.。
- 运动SFC程序名的第1个字符不能设置“\$(半角)”。
- 运动SFC程序名中不能设置“\ / : ; , . * ? "<> |(半角)”。

3.5 步

运动控制步

名称	符号	设置范围
运动控制步		K0~K4095

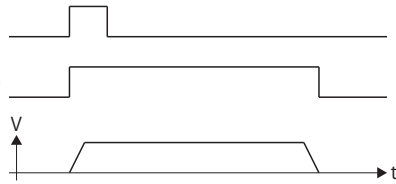
功能

- 指定的伺服程序Kn中指定轴的启动受理标志变为ON。
- 进行指定的伺服程序Kn的启动。

n 执行时机

转移条件成立

[St.1040]启动受理标志
(R: M30080+n/Q: M2001+n)




出错


指定的伺服程序Kn不存在时，将发生轻度出错(SFC) (出错代码: 31F0H)，在检测到出错的时刻运动SFC程序的执行将中止。

注意事项

- 在运动SFC程序中进行当前值更改的情况下，应通过伺服程序指定CHGA指令，通过运动控制步进行调用。
- 通过运动控制步指定的伺服程序的启动时/启动中，即使发生轻度出错，伺服程序由于出错而停止的情况下，运动SFC程序的执行也仍将继续。希望通过出错检测停止运动SFC程序的情况下，应在转换(转移条件)中置入出错检测的条件。
- 关于运动控制步中可记述的伺服程序，请参阅以下手册。

 MELSEC iQ-R运动控制器编程手册(定位控制篇)

运算控制步

名称	符号	设置范围
运算控制步		F0~F4095/FS0~FS4095

功能

n 1次执行型运算控制步Fn

Fn的情况下，执行1次指定的运算控制程序Fn。

n 扫描执行型运算控制步FSn

FSn的情况下，重复执行指定的运算控制程序FSn，直至下一个转移条件成立为止。


出错

指定的运算控制程序Fn/FSn不存在时，将发生轻度出错(SFC) (出错代码: 31F1H)，在检测到出错的时刻运动SFC程序的执行将中止。

注意事项

- 关于运算控制程序中可记述的运算公式，请参阅运算控制程序。(☞ 108页 运算控制程序)
- 运算控制程序执行中即使发生运算出错等，运动SFC程序的执行也仍将继续。

子程序调用/启动步

名称	符号	设置范围
子程序调用/启动步		登录的程序名

功能

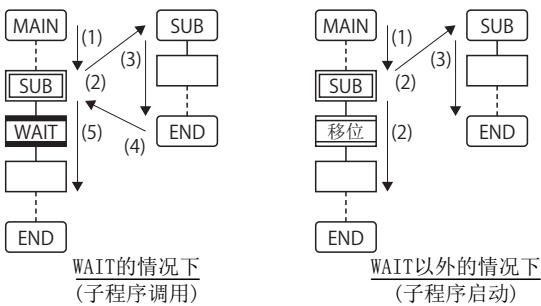
- 进行指定程序名的运动SFC程序的调用/启动。
- 根据子程序调用/启动步的下一个合并的转换的种类，控制有所不同。

n WAIT的情况下(子程序调用)

如果执行子程序调用步，如下图所示将控制转移至指定的程序，通过被调用的程序的END执行将控制返回至调用源程序。

n WAIT以外的情况下(子程序启动)

如果执行子程序启动步，如下图所示，启动指定程序后，转移至下一个。因此，启动源运动SFC程序与启动目标运动SFC程序将被并联执行。此外，启动的程序通过END执行而结束。



出错

- 子程序调用/启动时，指定的运动SFC程序不存在的情况下，将发生轻度出错(SFC) (出错代码： 32F5H)，在检测到出错的时刻调用源/启动源的运动SFC程序的执行将中止。
- 子程序调用/启动时，调用/启动的运动SFC程序已处于启动中的情况下，将发生轻度出错(SFC) (出错代码： 32F6H)，在检测到出错的时刻调用源/启动源的运动SFC程序的执行将中止。
- 子程序调用/启动时，本程序已调用/启动的情况下，将发生轻度出错(SFC) (出错代码： 33FAH)，在检测到出错的时刻调用源/启动源的运动SFC程序的执行将中止。
- 在通过运动SFC程序1调用/启动的运动SFC程序2中，子程序调用/启动时，调用/启动的子程序为运动SFC程序1(调用/启动程序)时，将发生轻度出错(SFC) (出错代码： 33FBH)，在检测到出错的时刻调用源/启动源的运动SFC程序2的执行将中止。

注意事项

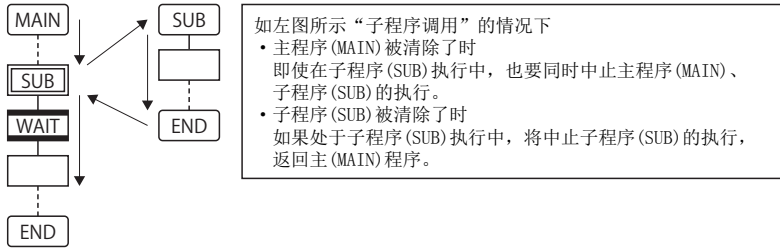
- 子程序调用/启动的嵌套深度无限制。
- 子程序启动的情况下，即使启动目标运动SFC程序因出错而停止，启动源运动SFC程序也将继续进行处理。
- 子程序调用的情况下，调用目标运动SFC程序因出错而停止时，在该时刻调用源运动SFC程序的执行也将中止。

清除步

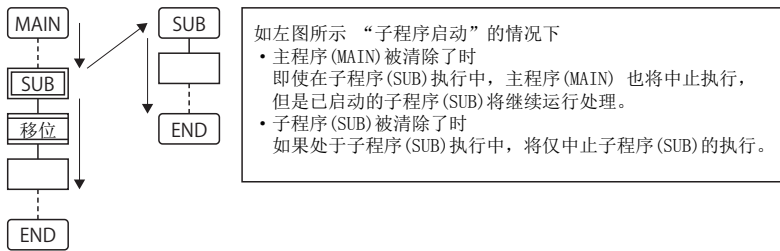
名称	符号	设置范围
清除步		登录的程序名

功能

- 执行中的指定运动SFC程序的执行将中止。
- 清除指定的运动SFC程序即使处于自动启动设置状态，中止后也不自动启动。
- 指定程序也可可是本程序。
- 指定程序处于子程序调用中的情况下，调用的子程序的执行也将中止。（下图）



- 指定程序处于子程序启动后的情况下，启动的子程序将继续进行处理。（下图）



- 通过指定程序启动的伺服程序启动中的情况下，伺服程序将继续进行处理。
- WAITON/WAITOFF+运动控制步中条件成立等待的情况下，等待条件成立，执行伺服程序。不执行伺服程序的情况下，应另行输入相应轴的停止指令。

出错

清除步中指定了不存在的运动SFC程序的情况下，如果通过MT Developer2进行运动SFC程序的转换将发生出错。

注意事项

- 清除步中指定的运动SFC程序未处于启动中时，不会出错但将被忽略。
- 即使通过清除步停止了运动SFC程序的执行，输出仍将保持。
- 为了配合清除步执行而停止动作中的轴的情况下，应另行输入相应轴的停止指令。

3.6 转换

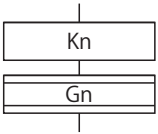
转换中可以记述条件表达式及运算公式。在此记述的运算公式与扫描执行型运算步一样，在转移条件成立之前将重复执行。为了配合清除步执行而停止动作中的轴的情况下，应另行输入相应轴的停止指令转换程序。（☞ 263页 转换程序）

与运动控制步的组合

功能

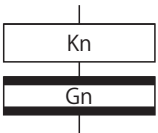
n 运动控制步+移位

- 不等待通过运动控制步启动的伺服程序Kn的动作完成，通过转移条件Gn的成立转移至下一个步。



n 运动控制步+WAIT

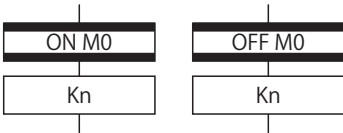
- 等待通过运动控制步启动的伺服程序Kn的动作完成，通过转移条件Gn的成立转移至下一个步。
- 转移条件Gn中，不需要伺服程序Kn的动作完成条件。
- 对于已启动的伺服程序Kn，即使启动时/启动中由于出错而停止的情况下，也将视为动作完成。



n WAITON/WAITOFF+运动控制步

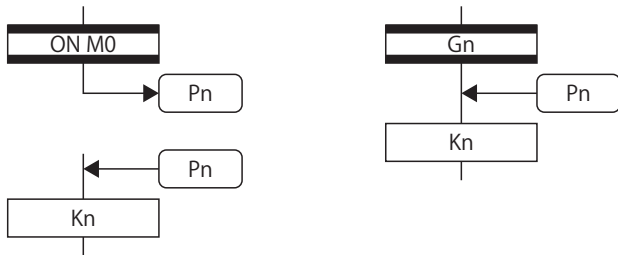
- 进行WAITON/WAITOFF的下一个运动控制步的启动准备，通过指定位软元件ON/OFF立即启动。不与WAITON/WAITOFF组合而执行运动控制步的情况下，运动控制步前面的转移条件成立后，进行启动准备，执行启动。因此，转移条件成立后至启动之前，将产生延迟时间/启动时间的偏差，但通过与WAITON/WAITOFF组合，可以消除延迟时间/启动时间的偏差。
- 关于WAITON/WAITOFF中可指定的位软元件的详细内容，请参阅以下手册。

☞ MELSEC iQ-R运动控制器编程手册(公共篇)



注意事项

- 必须与运动控制步以1对1进行成对设置。WAITON/WAITOFF的下一个步不是运动控制步的情况下，将发生轻度出错(SFC)(出错代码: 33F2H)，在检测到出错的时刻运动SFC程序的执行将中止。
- WAITON/WAITOFF之后的跳转目标为运动控制步时，不发生出错。(下图左)
- WAITON/WAITOFF之后可以存在指针。(下图右)

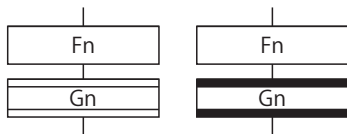


- 运动控制步中指定的伺服程序启动时，发生轻度出错而未能启动的情况下，与WAITON/WAITOFF位软元件的状态无关，运动SFC程序的执行将继续并转移到下一个步。希望通过检测到出错而停止运动SFC程序的情况下，应在下一个转换(转移条件)中置入出错检测的条件。
- 在与WAITON/WAITOFF组合使用的运动控制步中可以使用以下指令。(直线插补控制、圆弧插补控制、螺旋插补控制、位置跟踪控制、连续轨迹控制、高速振荡、固定位置停止速度控制)

与运算控制步的组合

功能



- 运算控制步的情况下，移位与WAIT的动作相同，执行运算控制程序Fn后，通过转移条件Gn成立而转移至下一个步。



与子程序调用/启动步的组合

关于与子程序调用/启动步的组合，请参阅子程序调用/启动步。(☞ 92页 子程序调用/启动步)

3.7 跳转·指针

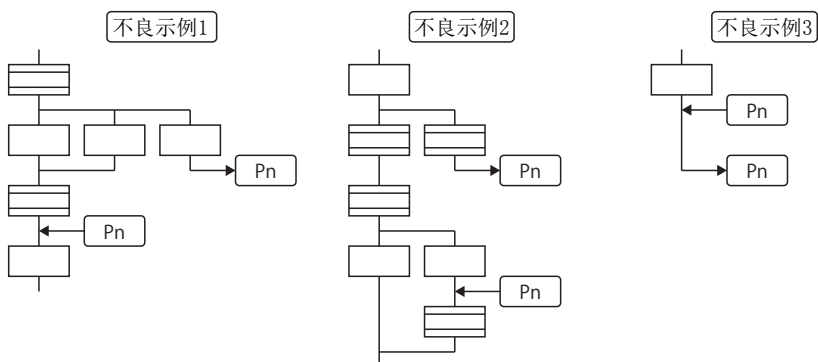
名称	符号	设置范围
跳转		P0~P16383
指针		P0~P16383

功能

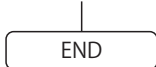
- 通过设置跳转，跳转至本程序内指定的指针Pn处。
- 步、转换、分支点、合并点可设置为指针。
- 1个程序中可设置P0~P16383的指针Pn。

注意事项

- 并不能进行从并联分支-并联合并内跳出的跳转设置。应进行直接连接。(以下为不良示例1)
- 不能进行从并联分支-并联合并外至并联分支-并联合并内的跳转设置。(以下为不良示例2)
- 不能进行标签与跳转连续的设置。(以下为不良示例3)



3.8 END

名称	符号	设置范围
END		—

功能

- 结束程序。(事件任务、NMI任务的情况下，根据程序参数的END动作设置其动作有所不同。(☞ 275页 程序参数)
- 进行了子程序调用的情况下，将返回至调用源运动SFC程序。

注意事项

- END在1个程序内可设置多个。
- 并联分支-合并之间不能设置END。
- 通过END结束运动SFC程序后，输出也仍将保持。

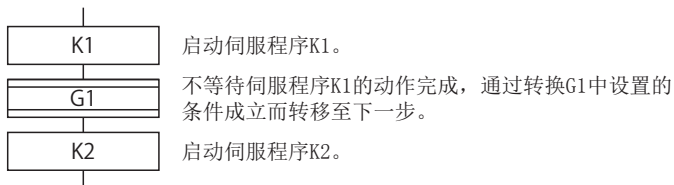
3.9 分支·合并

串联转移

将执行转移至串联连接的下一个步或转换处。伺服程序的启动或子程序的启动后，希望通过以下动作转移至下一步时，将转换设置为移位或WAIT。

希望不等待动作完成而转移至下一步的情况下

应将转换设置为移位。在此情况下，可以省略转换(移位)。省略了转换的情况下，将变为无条件移位转移。

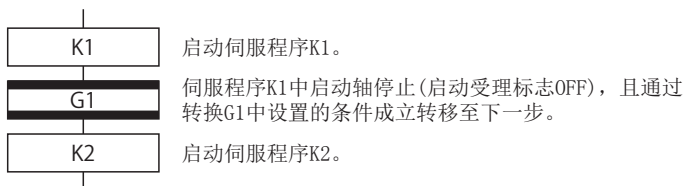


要点

子程序启动的情况下，对本程序与子程序进行并联处理。

希望通过动作完成进入下一个步的情况下

应将转换设置为WAIT。



要点

- 在上述中，对于下一个伺服程序K2中启动的轴的启动受理标志，不能设置为互锁条件。希望设置为互锁条件的情况下，应由用户设置到转移条件G1中。
- 希望通过动作完成进入下一个步的情况下，需要设置WAIT，但没有作为互锁条件而特别设置的条件的情况下，应在转换程序(Gn)中设置“NOP(无处理)”。

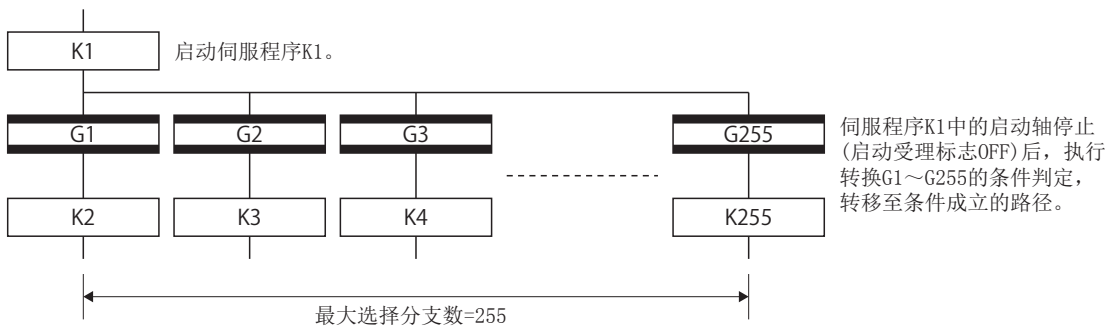
选择分支 · 选择合并

选择分支

进行并联连接的多个转换的条件判定，仅执行最先条件成立的路径。仅限于转换全部是移位或全部是WAIT。

例

WAIT的情况下

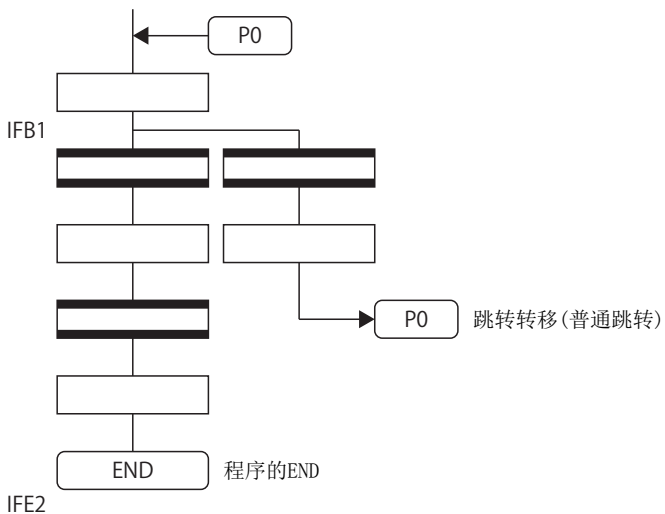


要点

- 转换的条件判定不限于按从左至右的顺序执行。
- 转换中同时存在移位及WAIT的情况下, 将变为并联分支。

选择合并

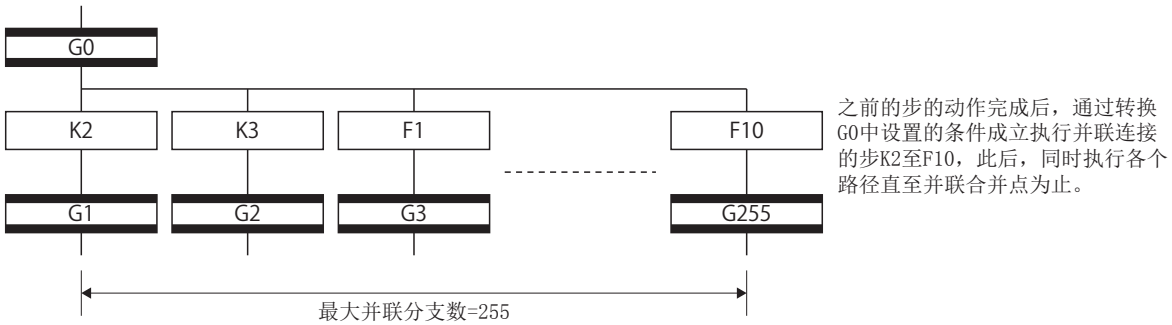
选择分支后, 各路径的处理完成后再次合并为1个路径的情况下, 设置为选择合并, 但也可按如下所示设置为不合并。



并联分支 · 并联合并

并联分支

同时执行并联连接的多个步。并联分支目标起始可以是步或转换。

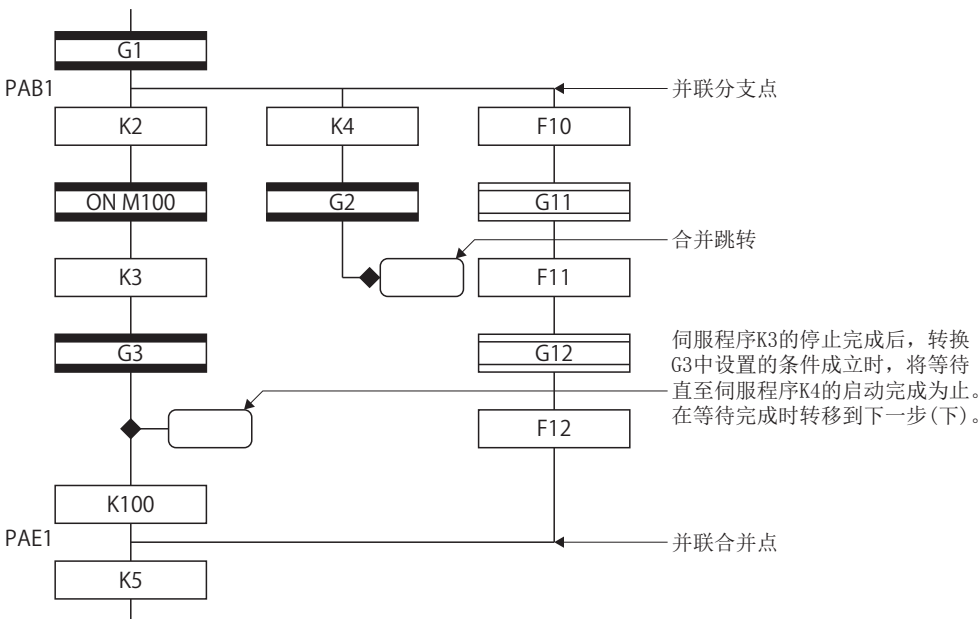


要点

并联分支之前的转换中可以设置“移位”或“WAIT”。不能设置“WAITON”、“WAITOFF”。

并联合并

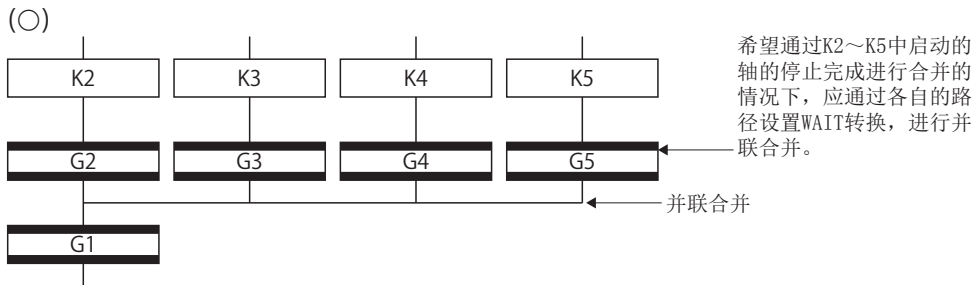
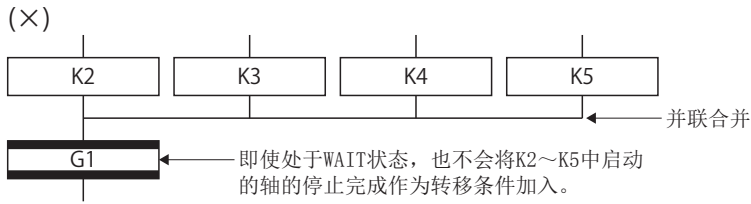
进行了并联分支的情况下，必须通过并联合并进行合并。在并联分支—并联合并之间，可以进行至其它分支路径的跳转设置。在此情况下，跳转目标将变为途中的并联合并点(合并跳转)。不能设置从并联分支—并联合并之间跳出的跳转。



要点

也可设置为并联分支数与并联合并点中的合并数不一致。(上图的示例中，并联分支数=3，合并数=2)

并联合并之后设置了WAIT转换的情况下，即使并联合并之前为运动控制步，该轴的停止完成也不作为等待条件。希望设置为通过停止完成进行并联合并的情况下，应将WAIT转换设置在并联合并之前。

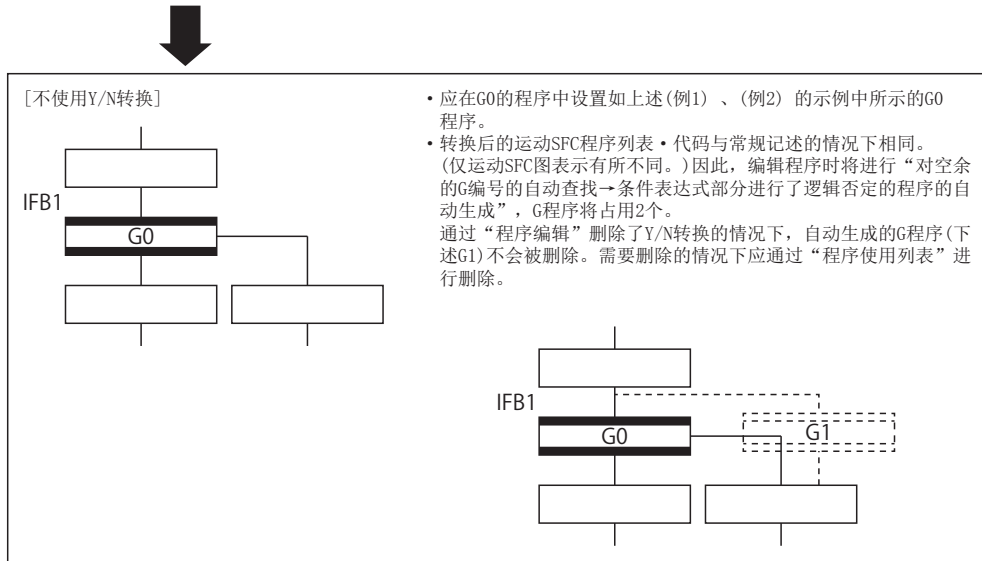
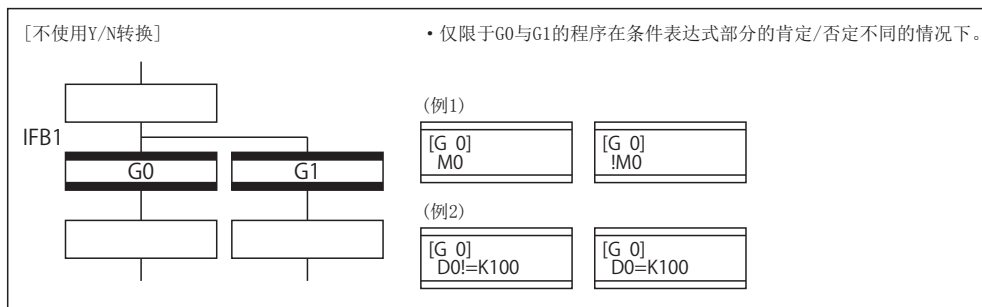


3.10 Y/N转换

转移条件成立时及不成立时进行路径分支的情况下，如果使用“移位Y/N转换”・“WAIT Y/N转换”将带来便利。

名称	符号	功能
移位Y/N转换		<ul style="list-style-type: none"> Gn中设置的转移条件成立时转移至下一步，不成立时转移至右侧连接的步。 “移位Y/N”与“WAIT Y/N”的区别和“移位”与“WAIT”的区别相同。
WAIT Y/N转换		

对于记述如下所示的2个路径的选择分支程序可带来方便。



空余G编号的自动查找规格

n 未自动编号设置时

以“移位Y/N”或“WAIT Y/N”符号从“设置的G编号+1”开始向后查找空余编号。查找至4095而没有空余的情况下，从0开始至“设置的G编号-1”为止进行查找。

n 自动编号设置时

以“移位Y/N”或“WAIT Y/N”符号从“自动编号的G编号+1(或-1)”开始向后(或向前)查找自动编号范围的空余编号。(查找方法按照自动编号设置。)

逻辑否定程序的自动生成规格

自动生成对“移位Y/N”或“WAIT Y/N”中设置的转换程序的条件表达式块(最终块)进行了逻辑否定的程序。基本如下所示。

[设置程序(条件表达式块)]

条件表达式 //(位条件表达式 · 比较条件表达式)



[逻辑否定的自动生成程序(条件表达式块)]

! 条件表达式 //(位条件表达式 · 比较条件表达式)

示例如下所示。

[设置程序(条件表达式块)]

(例1)

M0 //位软元件ON

[逻辑否定的自动生成程序(条件表达式块)]

!(M0) //位软元件OFF

(例2)

D0!=K100 //数据寄存器D0不是K100

!(D0!=K100) //数据寄存器D0是K100

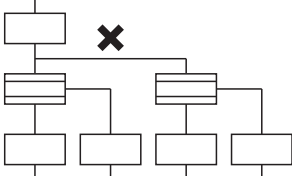
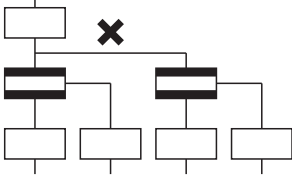
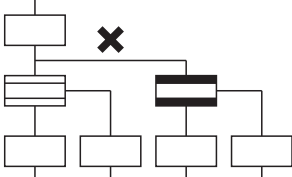
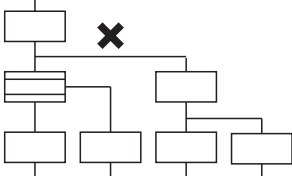
要点

- 关于可用于“移位Y/N”或“WAIT Y/N”转换程序的条件表达式的指令，请参阅运算控制·转换指令一览。
([☞](#) 22页 运算控制·转换指令一览)
- 设置程序中应只设置条件表达式块。

运动SFC图的注意事项

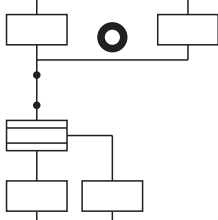
对于由于Y/N转换的定义而变得无含义或矛盾的运动SFC图，编辑(或运动SFC图转换后)时将发生出错。该模式及注意事项如下所示。

n “移位Y/N” 或 “WAIT Y/N” 作为选择分支或并联分支被连接的情况下
以下模式时将发生出错。

内容	运动SFC图的模式
“移位Y/N” 的选择分支	
“WAIT Y/N” 的选择分支	
“移位 Y/N” 与 “WAIT Y/N” 的并联分支	
“移位(或WAIT)Y/N” 与其它步·转换的并联分支或选择分支	

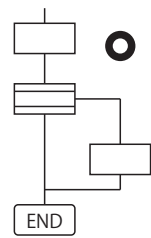
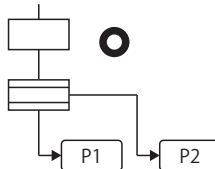
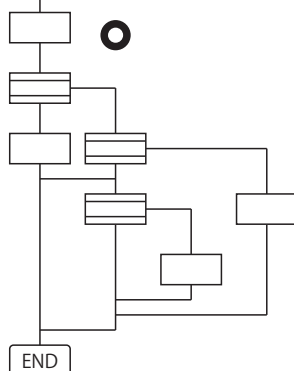
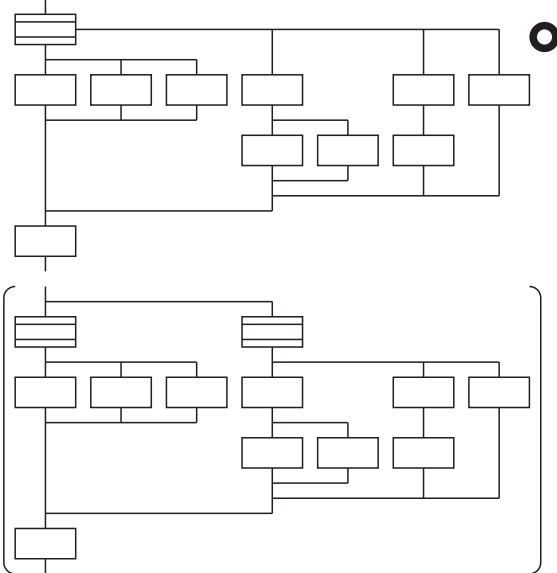
n 在 “移位Y/N” 或 “WAIT Y/N” 之前合并的情况下

应对 “合并-分支的连续” 置入间隔。

内容	运动SFC图的模式
至 “移位Y/N” 或 “WAIT Y/N” 不能直接合并。	
应对 “合并-分支的连续” 置入间隔。	

n可设置的模式

可以设置下述模式。

内容	运动SFC图的模式
从“移位Y/N”或“WAIT Y/N”的结束(END)	 <p>The diagram shows a vertical sequence of three states. The top state is a simple rectangle. The middle state is a rectangle with a horizontal line through its center. The bottom state is a rectangle labeled 'END'. A vertical line connects the top state to the middle state, and another vertical line connects the middle state to the 'END' state. A small circle with a dot inside is positioned to the right of the top state.</p>
从“移位Y/N”或“WAIT Y/N”的跳转	 <p>The diagram shows a vertical sequence of three states. The top state is a simple rectangle. The middle state is a rectangle with a horizontal line through its center. Below the middle state, two parallel paths lead to states labeled 'P1' and 'P2'. A small circle with a dot inside is positioned to the right of the top state.</p>
从“移位Y/N”或“WAIT Y/N”至“移位Y/N”或“WAIT Y/N”的连续(选择分支-选择分支)	 <p>The diagram shows a complex branching structure. It starts with a top state, followed by a state with a horizontal line. From this state, the path branches into two main paths. The left path goes through two more states before reaching an 'END' state. The right path goes through a state with a horizontal line, then branches into two more paths, each leading through a state to an 'END' state. A small circle with a dot inside is positioned to the right of the top state.</p>
从“移位Y/N”或“WAIT Y/N”的Y侧/N侧的连接线为2条以上的情况下将变为选择分支→选择分支或并联分支的连续。	 <p>The diagram shows a state with a horizontal line at the top. From this state, multiple lines branch out to a row of six states. From the second state from the left, two lines branch out to two more states. From the third state from the left, two lines branch out to two more states. From the fourth state from the left, two lines branch out to two more states. From the fifth state from the left, two lines branch out to two more states. From the sixth state from the left, two lines branch out to two more states. All these paths eventually lead to an 'END' state. A small circle with a dot inside is positioned to the right of the top state.</p>

3.11 运动SFC注释

对运动SFC图的步·转换等各符号可以设置注释。在运动SFC程序编辑画面中，通过将显示模式切换为“注释显示”，运动SFC图内将显示注释。

分类	名称	符号	注释设置
程序开始/结束	START		不能设置注释。
	END		
步	运动控制步		最大半角80(全角40)字符 20字符×4行显示
	1次执行型运算控制步		
	扫描执行型运算控制步		
	子程序调用/启动步		
	清除步		
转换	移位(预读转移)		最大半角64(全角32)字符 16字符×4行显示
	WAIT		
	WAITON		
	WAITOFF		
	移位Y/N		
	WAIT Y/N		
跳转	跳转		最大半角64(全角32)字符 16字符×4行显示
指针	指针		

- 运动SFC注释被存储在运动CPU本体的代码区域中。代码区域中存储有运动SFC图代码、运算控制(F/FS)程序代码、转换(G)程序代码以及运动SFC注释。应注意避免设置过多的注释，防止代码区域溢出。(关于代码区域的容量，请参阅运动SFC性能规格。(P.19页 运动SFC性能规格))
- 注释文中不能使用“,(半角)”。

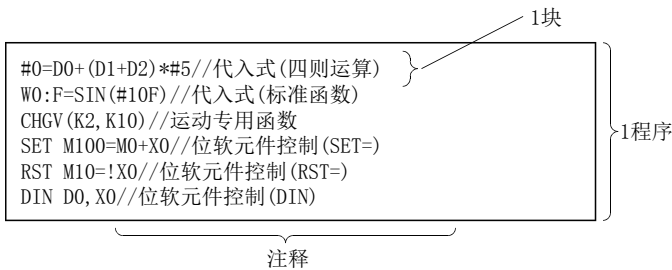
4 运算控制程序

4.1 运算控制程序

运算控制程序

- 在运算控制程序中，可以设置代入运算公式、运动专用函数、位软元件控制指令。
- 1个运算控制程序中可以设置多个块。
- 1个运算控制程序中可设置的块数无限制。但是，1个程序应在128k字节以内。
- 运算控制程序以半角字符进行设置。注释可以使用全角字符。
- 1个块的最大字符数为半角字符的1020字符。全角字符相当于半角字符的2个字符。
- 不能设置转移条件。转移条件只能通过转换程序进行设置。
- 对于运算控制程序中返回逻辑数据值(真假)的位条件表达式、比较条件表达式，只能作为软元件设置(SET=)/软元件复位(RST=)的源(S)进行设置。

运算控制程序示例如下所示。



运算符、函数的优先顺序

运算符、函数的优先顺序如下所示。通过使用括号，可以对运算的顺序进行任意指定。

优先顺序	项目(运算符、函数)
高 ↑ ↓ 低	1 括号内的计算((...))
	2 标准函数(SIN、COS等)、类型转换(USHORT、LONG等)
	3 位取反(^)、逻辑否定(!)、符号取反(-)
	4 乘法(*)、除法(/)、余数(%)
	5 加法(+)、减法(-)
	6 左移(<<)、右移(>>)
	7 比较运算符: 小于(<)、小于等于(<=)、大于(>)、大于等于(>=)
	8 比较运算符: 一致(==)、不一致(!=)
	9 位逻辑积(&)
	10 位异或(^)
	11 位逻辑和()
	12 逻辑积(*)
	13 逻辑和(+)
	14 代入(=)

指令的构成

运算控制程序中可使用的大部分指令可分为指令部及数据部。指令部及数据部的用途如下所示。

- 指令部：表示该指令的功能。
- 数据部：表示指令中使用的数据。

例

代入：=的情况下



n 源(S)

- 源是运算中使用的数据。
- 根据各指令中指定的软元件，其情况如下所示。

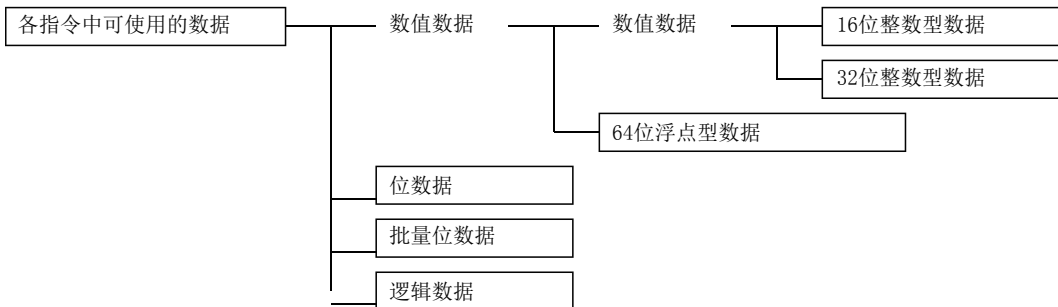
软元件	内容
位软元件、字软元件	指定存储运算中使用的数据的软元件。在执行运算之前，需要将数据预先存储到指定的软元件中。程序执行过程中，通过更改指定软元件中存储的数据，可以更改该指令中使用的数据。
常数	指定运算中使用的数值。由于是在创建程序时设置，因此在程序执行中不能更改。

n 目标(D)

- 目标数据中存储运算后的数据。
- 目标数据中必须设置用于存储数据的软元件。

数据的指定方法

各指令中可使用的数据有以下6种。



n 16位整数型数据

16位整数型数据是16位的整数值数据。字软元件以1点单位使用。数据范围如下所示。

	10进制表示	16进制表示
数据范围	K-32768~K32767	H0000~HFFFF

n 32位整数型数据

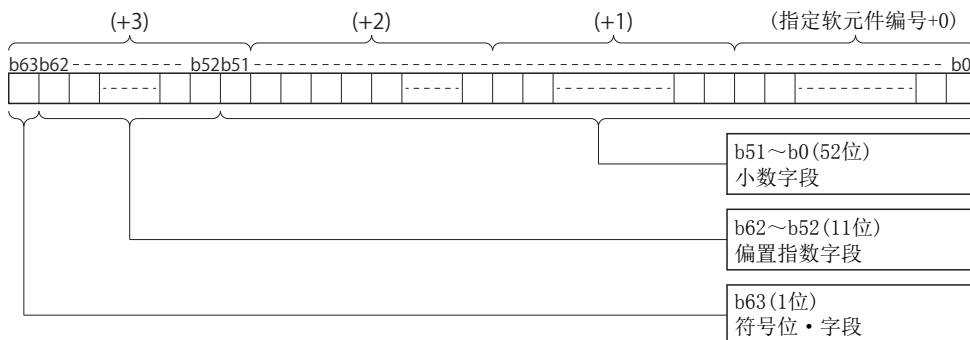
32位整数型数据是32位的整数值数据。字软元件以(指定软元件编号)、(指定软元件编号+1)的2点单位使用。数据范围如下所示。

	10进制表示	16进制表示
数据范围	K-2147483648L~K2147483647L	H00000000L~HFFFFFFFL

n 64位浮点型数据

64位浮点型数据是IEEE格式的64位浮点值数据。字软元件以(指定软元件编号)、(指定软元件编号+1)、(指定软元件编号+2)、(指定软元件编号+3)的4点单位使用。

- 内部的位构成如下所示。



- 显示的值如下所示。(偏置值为H3FF。)
 $(-1)^{[\text{符号位} \cdot \text{字段}]} * (1.0 + [\text{小数字段}]) * 2^{([\text{偏置指数字段}] - [\text{偏置值}])}$
- 数据范围如下所示。

	10进制表示	16进制表示
数据范围	K-1.79E+308~K-2.23E-308, K0.0, K2.23E-308~K1.79E+308	H0000000000000000, H0010000000000000~H7FE1CCF385EBC89F, H8000000000000000, H8010000000000000~HF FE1CCF385EBC89F

- 在64位浮点型数据的运算中，有可能产生化整误差。尤其是将64位浮点型数据用于比较运算的情况下，由于化整误差的影响，有可能无法按意图动作，应加以注意。

例

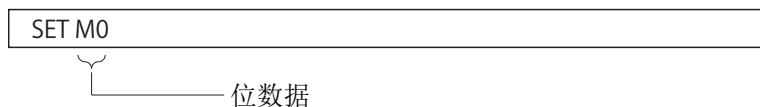
在下述转换程序中，由于化整误差的影响，根据#200F的值比较运算的结果有时不能变为真。

```
#100F=SQRT(#200F)
#300F=#100F*#100F
#200F==#300F
```

n 位数据

位数据是以触点·线圈等1位单位处理的数据。通过软元件设置(SET=)、软元件复位(RST=)使用。

例



n 批量位数据

批量位数据是将位数据以16点单位/32点单位处理的数据。通过软元件输入(DIN)、软元件输出(DOUT)使用。如下所示，变为16点单位还是32点单位取决于输入目标/输出源的字软元件的数据类型。

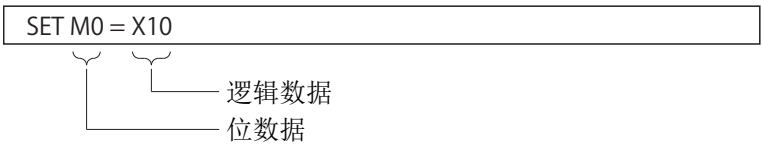
	16点单位	32点单位
程序示例	DIN #0, M0 DOUT M0, D0	DIN #0L, M0 DOUT M0, D0L
使用软元件	(指定软元件编号)~(指定软元件编号+15) 在上述程序示例中M0~M15	(指定软元件编号)~(指定软元件编号+31) 在上述程序示例中M0~M31

n 逻辑数据

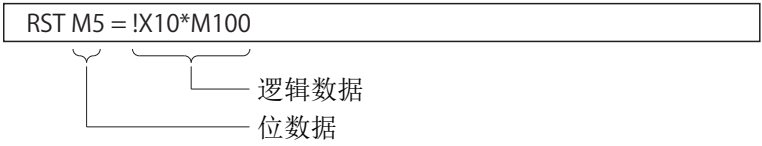
逻辑数据是位条件表达式、比较条件表达式返回的值，是表示真/假的数据。通常，在转换程序的条件表达式中使用，但在运算控制程序中，在软元件设置(SET=)、软元件复位(RST=)中设置的位条件表达式中使用。

例

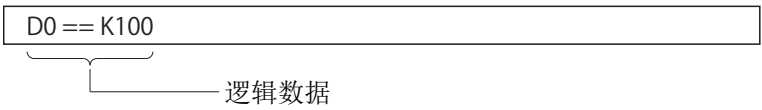
(例1)



(例2)



(例3) 转换程序



4.2 软元件记述

字软元件、位软元件的各记述如下所示。

字软元件记述

软元件名称	软元件记述*1		
	16位整数型	32位整数型 (n为偶数)	64位浮点型 (n为偶数)
数据寄存器	Dn	DnL	DnF
链接寄存器	Wn	WnL	Wn:F
运动寄存器	#n	#nL	#nF
特殊寄存器	SDn	SDnL	SDnF
CPU缓冲存储器访问软元件	U3E□\Gn*2	U3E□\GnL*2	U3E□\GnF*2
CPU缓冲存储器访问软元件 (恒定周期通信区域)	U3E□\HGn*2	U3E□\HGnL*2	U3E□\HGnF*2
模块访问软元件	U□\Gn*3	U□\GnL*3	U□\GnF*3

*1 关于可使用的软元件编号 (n) 的设置范围, 请参阅以下手册。

📖 MELSEC iQ-R运动控制器编程手册 (公共篇)

*2 □=CPU机号 (1号机: 0; 2号机: 1; 3号机: 2; 4号机: 3)。指定的CPU机号不能超过多CPU个数。

*3 □=00h~FFh (模块IO编号÷10h)。可访问的范围根据模块而有所不同。请参阅所使用的模块的手册。

- 32位整数型时附加L, 64位浮点型时附加F (链接寄存器的情况下:F) 进行区分。
- 32位整数型、64位浮点型的情况下以偶数指定软元件编号。(不能以奇数进行设置。)

位软元件记述

软元件名称	软元件记述*1
输入继电器	Xn
输出继电器	Yn
内部继电器	Mn
链接继电器	Bn
报警器	Fn
数据寄存器	Dn.m*2
链接寄存器	Wn.m*2
运动寄存器	#n.m*2
特殊继电器	SMn
特殊寄存器	SDn.m*2
CPU缓冲存储器访问软元件	U3E□\Gn.m*2*3
CPU缓冲存储器访问软元件 (恒定周期通信区域)	U3E□\HGn.m*2*3
模块访问软元件	U□\Gn.m*3*4

*1 关于可使用的软元件编号 (n) 的设置范围, 请参阅以下手册。

📖 MELSEC iQ-R运动控制器编程手册 (公共篇)

*2 “m”表示字软元件的位指定 (位编号: 0~F)。

*3 □=CPU机号 (1号机: 0; 2号机: 1; 3号机: 2; 4号机: 3)。
指定的CPU机号不能超过多CPU个数。

*4 □=00h~FFh (模块IO编号÷10h)。
可访问的范围根据模块而有所不同。请参阅所使用的模块的手册。

- 在DIN、DOUT中作为批量位数据使用的情况下n以16的倍数指定。
- 将以下软元件作为批量位数据使用的情况下，不进行位指定，作为字软元件进行指定。

软元件名称
数据寄存器(D)
链接寄存器(W)
运动寄存器(#)
特殊寄存器(SD)
CPU缓冲存储器访问软元件(U3E□\G)
CPU缓冲存储器访问软元件(恒定周期通信区域)(U3E□\HG)
模块访问软元件(U□\G)

软元件编号的间接指定

在字软元件记述/位软元件记述中，可以对软元件编号(n)进行间接指定。

n通过字软元件进行的软元件编号(n)的间接指定

- 不能使用间接指定的软元件编号的字软元件。
- 在16位整数型、32位整数型的字软元件中可以进行间接指定。不能使用64位浮点型。

例

记述示例

正确示例	错误示例
#(D10)	#(D(D5))
D(#10L)F	D(#4F)

n通过运算公式进行的软元件编号的间接指定

- 通过使用了以下数据、运算符的计算公式可以进行间接指定。

可用数据	16位整数型字软元件
	32位整数型字软元件
	16位整数型常数
	32位整数型常数
可用运算符	加法: +
	减法: -
	乘法: *
	除法: /
	余数: %
	符号取反: -

- 不能使用间接指定的软元件编号的字软元件。

例

记述示例

正确示例	错误示例
#(D10-K5)	#(D(D5)F+K20)
D(#10L%H6L)F	D(#4L<<K2)
D(K640+K2*K31)L	D(M0*#10.F)

- *1 使用上述以外的计算结果进行软元件编号的间接指定的情况下，应按以下方式分为2个块进行记述。

```
D0 = SHORT(ASIN(#0F))
W0 = #(D0)
```

4.3 常数记述

16位整数型、32位整数型、64位浮点型的各常数记述如下所示。

表示	16位整数型	32位整数型	64位浮点型
10进制数	K-32768~K32767	K-2147483648L~K2147483647L	K-1.79E+308~K-2.23E-308, K0.0, K2.23E-308~K1.79E+308
16进制数	H0000~HFFFF	H00000000L~HFFFFFFFL	—

- 对32位整数型附加L，对64位浮点型附加小数点以及指数部(E)以标明数据的类型。
- 省略了数据类型的情况下，将视为可处理的最小类型。
- 10进制数表示的情况下在起始处附加K，16进制数表示的情况下在起始处附加H。可以省略K。
- 64位浮点型不能以16进制数表示。

4.4 标签

标签是输入输出数据及内部处理中指定了任意字符串的变量。
如果在编程时使用标签，在创建程序时可以无需理会软元件。
因此，对于使用了标签的程序在模块配置不同的系统中也可方便地再利用。

种类

标签

是1个工程中相同数据的标签。可以在工程内的所有程序中使用。
在标签的设置中，进行标签名与数据类型的关联。
此外，通过对标签进行公开设置，可以通过GOT进行参照，可以进行监视及数据访问。对于输入及输出等中使用的软元件，希望作为标签进行编程的情况下，可以直接分配软元件。

要点

标签的种类如下所示。

- 系统标签

是支持iQ Works的所有工程中相同数据的标签。可以通过GOT及CPU模块进行参照，用于进行监视及数据访问。关于详细内容，请参阅以下手册。

 iQ Works入门指南

数据类型

对于标签，根据位长、处理方法、值的范围等进行数据类型分类。
数据类型有以下种类。

基本数据类型

数据类型有以下数据类型。

数据类型		内容	值的范围	位长
位	BOOL	是表示ON或OFF等二者选一的状态的类型。	0 (FALSE)、1 (TRUE)	1位
字[无符号]/位串[16位]	WORD	是表示16位的数组的类型。	0~65535	16位
双字[无符号]/位串[32位]	DWORD	是表示32位的数组的类型。	0~4294967295	32位
字[带符号]	INT	是处理正及负的整数值的类型。	-32768~32767	16位
双字[带符号]	DINT	是处理正及负的双精度整数值的类型。	-2147483648~2147483647	32位
双精度实数	LREAL	是处理小数点以下的数值(双精度实数值)的类型。	K-1.79E+308~K-2.23E-308, K0.0, K2.23E-308~ K1.79E+308	64位

要点

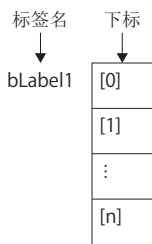
数据类型中使用双字[无符号]/位串[32位]、双字[带符号]、双精度实数的情况下，以偶数设置软元件编号。

数组

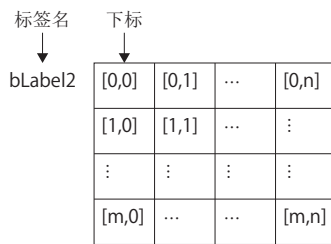
数组用于将相同数据类型的标签的连续的集合体以1个名称表示。

可以将基本数据类型及结构体定义为数组。根据数据类型，数组的最大数有所不同。

•1维数组的示意图



•2维数组的示意图



数组的定义

n 数组的要素

定义数组时，需要确定要素数(数组的长度)。

n 定义的格式

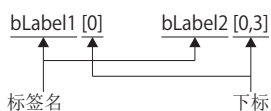
关于定义的格式，以3维数组为例进行说明。数组开始值~数组结束值的范围即为要素数。

数组的维数	格式	备注
1维数组	基本数据类型/结构体名的数组(数组开始值..数组结束值) [定义示例]位(0..2)	<ul style="list-style-type: none"> 关于基本数据类型的详细内容，请参阅基本数据类型(☞ 115页 基本数据类型) 关于结构体的详细内容，请参阅结构体(☞ 117页 结构体)
2维数组	基本数据类型/结构体名的数组(数组开始值..数组结束值, 数组开始值..数组结束值) [定义示例]位(0..2, 0..1)	
3维数组	基本数据类型/结构体名的数组(数组开始值..数组结束值, 数组开始值..数组结束值, 数组开始值..数组结束值) [定义示例]位(0..2, 0..1, 0..3)	

使用方法

使用数组时，为了识别各个标签，将标签名后面的下标用“[]”围住表示。

此外，2维以上数组的情况下，将“[]”内的下标用“逗号(,)”分开表示。



数组的下标中可以指定以下种类。

种类	指定示例	备注
常数	bLabel1[0]	可以指定整数。
软元件	bLabel1[D0]	软元件 可以指定字软元件、双字软元件、10进制常数、16进制常数。(软元件可以使用数据寄存器(D)、链接寄存器(W)、运动寄存器(#)。)

要点

- 通过在数组的下标中指定软元件，数据存储目标变为动态，因此可以用于进行重复处理的程序。“uLabel4”的数组中连续存储“1234”的程序如下所示。

```
FOR #0 = K0 TO K9 STEP K1
    uLabel4[#0] = K1234
NEXT
```

- 标签中分配的软元件的软元件编号超过32767的情况下，下标中指定的软元件应设置为32位整数型。(例)

标签数组“uLabel4”的起始软元件中分配了“D32000”，根据数组的下标“#0”的指定访问了D32768以后的情况下

```
uLabel4[#0L]
```

注意事项

n 使用事件任务的情况下

数组的下标中指定了软元件的情况下，组合多个指令进行运算。因此，数组中定义的标签的运算中发生了中断时，有可能会发生数据背离，变为意外的运算结果。

为了避免发生数据背离，应采取以下措施。

- 对于普通任务中作为下标使用的软元件在事件任务中不进行更改。
- 在使用数组中定义的标签前后使用事件任务禁止/允许(EI/DI指令)。

n 关于数组的要素

对于定义的数组的要素数，访问时应避免超出要素编号范围。

在常数中指定了超出数组中定义的范围的下标的情况下，如果进行运动SFC程序的转换将出错。此外，在常数以外指定了数组的下标的情况下，在运动SFC程序的转换中不会出错，在执行时进行其它软元件的访问处理。

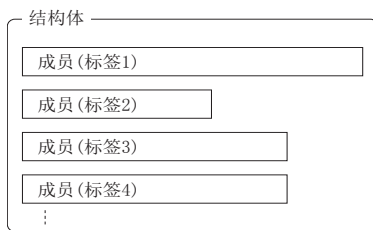
结构体

结构体是包含1个以上的标签的数据类型，可用于所有的程序。

对于结构体中包含的各个成员(标签)，即使数据类型不同也可定义。

结构体的创建

创建结构体时，首先创建结构体的定义，然后在创建的结构体中定义成员。



使用方法

使用结构体的情况下，将预先定义的结构体作为数据类型进行标签登录。

指定构成的各个成员时，在结构体标签名的后面以“点号(.)”分开附加成员名。

例

使用结构体的成员的情况下

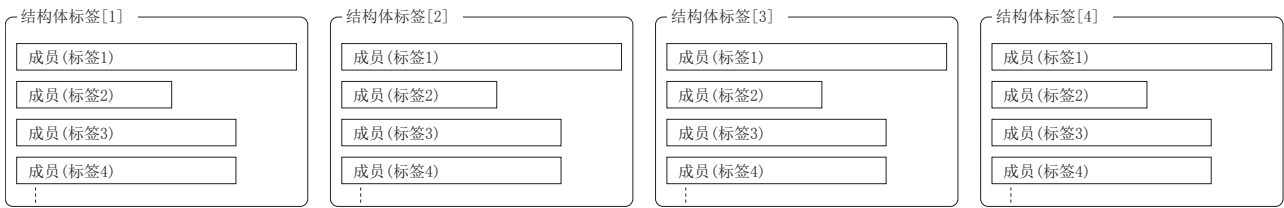


要点

在结构体的数据类型中使用双字[无符号]/位串[32位]、双字[带符号]、双精度实数的情况下，在使用了这些类型的所有成员中，应设置偶数的软元件编号。

结构体的数组

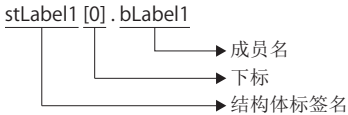
也可将结构体作为数组使用。



作为数组进行了宣言的情况下，在结构体标签名的后面将下标用“[]”围住表示。

例

使用设置为数组的结构体的要素的情况下



可指定的数据类型

基本数据类型可被指定为结构体的成员。

要点

在结构体的数据类型中使用双字[无符号]/位串[32位]、双字[带符号]、双精度实数的情况下，在使用了这些类型的所有成员中，应设置偶数的软元件编号。因此，根据成员的内容应在结构体中置入调整用的成员。(例)

标签名	数据类型
dLabel1	双字[带符号]
dLabel2	双字[带符号]
wLabel1	字[带符号]
wDummy1	字[带符号]
dLabel3	双字[带符号]
dLabel4	双字[带符号]

← 调整用的成员

注意事项

标签名的限制

对标签名有以下限制。

- 标签名以字符开始。不能定义以数字或下划线()开始的标签名。
- 不能将保留字定义到标签名中。

关于保留字的详细情况，请参阅以下内容。

📖 MT Developer2的帮助

4.5 二项运算

代入: =

格式	基本步数	可用步	
		F/FS	G
(D)=(S)	8	○	○

设置数据

n 可用数据

○: 可以设置

设置数据	可用数据								计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数						
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)				
(S)	—	○	○	○	○	○	○	○	—	—	
(D)	—	○	○	○	—	—	—	—	—	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	进行代入的字软元件/常数/计算式	(D)的数据类型
(D)	存储运算结果的软元件	

功能

- 将(S)中指定的数据的值代入(D)中指定的字软元件中。
- (S)与(D)的数据类型不相同的情况下,转换为(D)的数据类型后进行代入。(D)为16位整数型或32位整数型,(S)为64位浮点型的情况下,(S)的小数部将被舍去。)

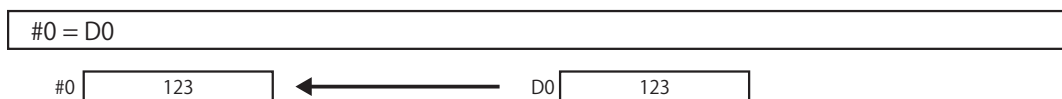
出错

以下情况下将发生运算出错。

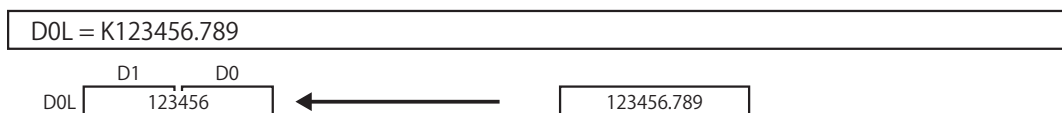
- (S)的数据超出(D)的数据类型的范围时。
- (D)或(S)为间接指定软元件,软元件编号超出范围时。

程序示例

n 将D0的值代入到#0中的程序

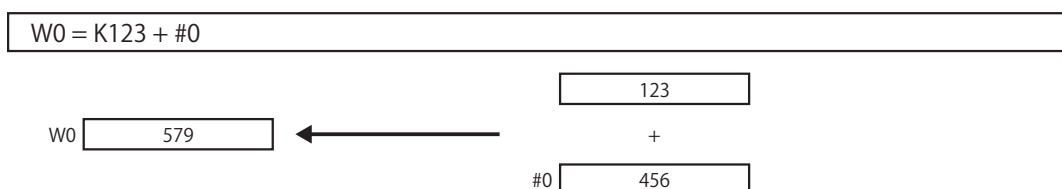


n 将K123456.789代入D0L中的程序



将64位浮点型转换为32位整数型后进行代入。

n 将K123与#0的加法运算值代入W0中的程序



加法: +

格式	基本步数	可用步	
		F/FS	G
(S1)+(S2)	7	○	○

设置数据

n 可用数据

○: 可以设置

设置数据	可用数据							计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数					
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S1)	—	○	○	○	○	○	○	—	—	
(S2)	—	○	○	○	○	○	○	—	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	被加数据	(S1)或(S2)的较大一方的数据类型
(S2)	加法数据	

功能

- 将(S1)中指定的数据与(S2)中指定的数据进行加法运算。
- (S1)与(S2)的数据类型不相同的情况下,转换为较大类型后进行运算。

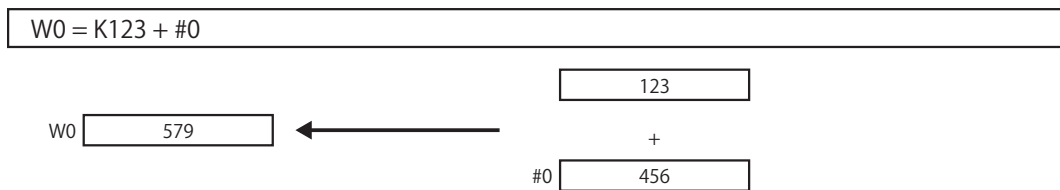
出错

以下情况下将发生运算出错。

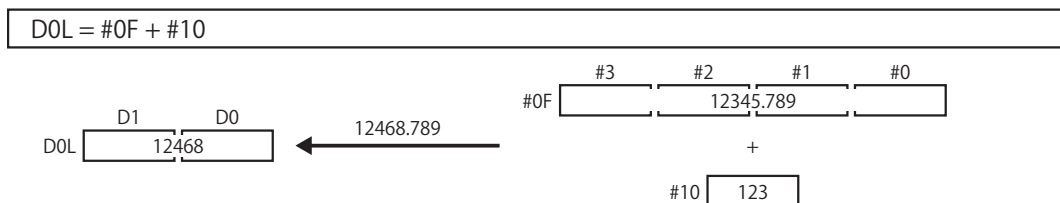
- (S1)或(S2)为间接指定软元件,软元件编号超出范围时。

程序示例

n 将K123与#0的加法运算结果代入到W0中的程序



n 将#0F与#10的加法运算结果代入到D0L中的程序



以64位浮点型进行加法运算,将结果转换为32位整数型后进行代入。

减法：-

格式	基本步数	可用步	
		F/FS	G
(S1)-(S2)	7	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据							计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数					
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S1)	—	○	○	○	○	○	○	—	—	
(S2)	—	○	○	○	○	○	○	—	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	被减数据	(S1)或(S2)的较大一方的数据类型
(S2)	减法数据	

功能

- 从(S1)中指定的数据减去(S2)中指定的数据。
- (S1)与(S2)的数据类型不相同的情况下，转换为较大类型后进行运算。

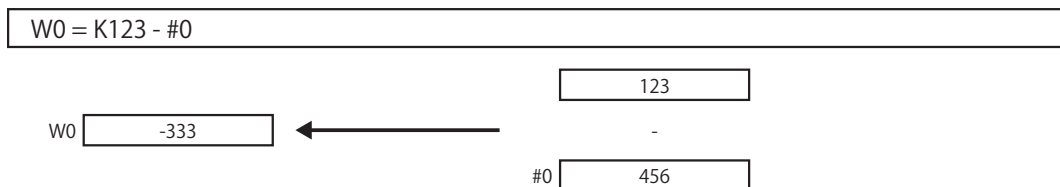
出错

以下情况下将发生运算出错。

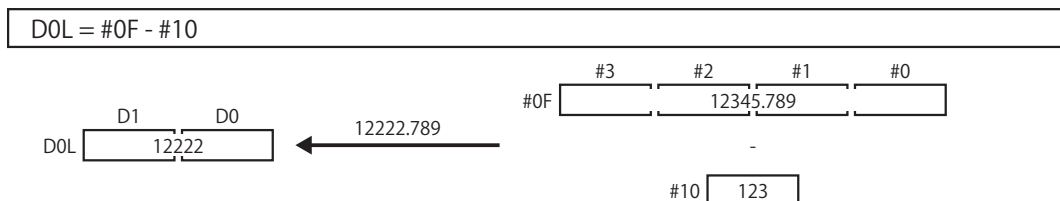
- (S1)或(S2)为间接指定软元件，软元件编号超出范围时。

程序示例

n 将从K123中减去#0后的结果代入到W0中的程序



n 将从#0F中减去#10后的结果代入到D0L中的程序



以64位浮点型进行减法运算，将结果转换为32位整数型后进行代入。

乘法：*

格式	基本步数	可用步	
		F/FS	G
(S1) * (S2)	7	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据							计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数					
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S1)	—	○	○	○	○	○	○	—	—	
(S2)	—	○	○	○	○	○	○	—	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	被乘数据	(S1)或(S2)的较大一方的数据类型
(S2)	乘法数据	

功能

- 将(S1)中指定的数据乘以(S2)中指定的数据。
- (S1)与(S2)的数据类型不相同的情况下，转换为较大类型后进行运算。
- 对于运动SFC程序，(S1)与(S2)的数据类型不相同的情况下，转换为较大类型后对乘法结果进行处理。乘法结果超出各类型中可处理的数值范围时，将发生溢出但不变为运算出错状态。通过使用类型转换指令转换设置数据，有可能防止溢出。有关详细内容，请参阅以下程序示例。
 - ☞ 123页 将#0乘以#10后的结果代入到W0L中的程序
 - ☞ 123页 将#0及#10分别转换为32位整数型之后相乘的结果代入到W0L中的程序

出错

以下情况下将发生运算出错。

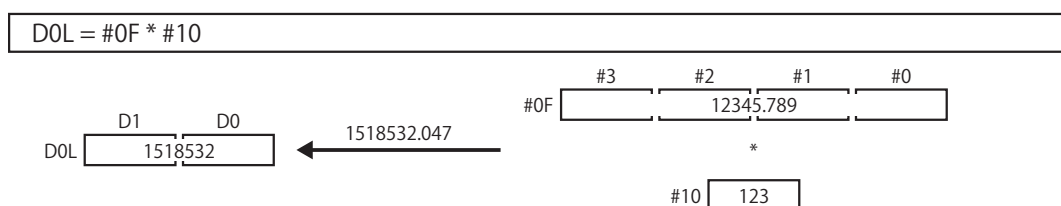
- (S1)或(S2)为间接指定软元件，软元件编号超出范围时。

程序示例

n 将K123乘以#0后的结果代入到W0中的程序



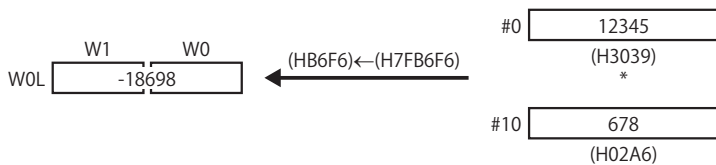
n 将#0F乘以#10后的结果代入到D0L中的程序



以64位浮点型进行乘法运算，将结果转换为32位整数型后进行代入。

n 将#0乘以#10后的结果代入到W0L中的程序

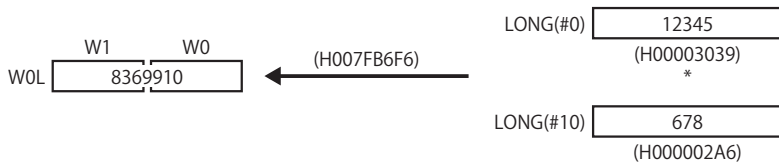
```
W0L = #0 * #10
```



设置数据均为16位整数型，因此乘法结果以16位整数型进行处理。发生溢出，乘法结果的低16位将成为运算结果。

n 将#0及#10分别转换为32位整数型之后相乘的结果代入到W0L中的程序

```
W0L = LONG(#0) * LONG(#10)
```



即使软件值与上述程序示例(将#0乘以#10后的结果代入到W0L中的程序)相同的情况下，通过类型转换指令乘法结果被处理为32位整数型，因此不发生溢出。

除法： /

格式	基本步数	可用步	
		F/FS	G
(S1) / (S2)	7	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据							计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数					
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S1)	—	○	○	○	○	○	○	—	—	
(S2)	—	○	○	○	○	○	○	—	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	被除数据	(S1) 或 (S2) 的较大一方的数据类型
(S2)	除法数据	

功能

- 将(S1)中指定的数据用(S2)中指定的数据相除，求出商。
- (S1)与(S2)的数据类型不相同的情况下，转换为较大类型后进行运算。

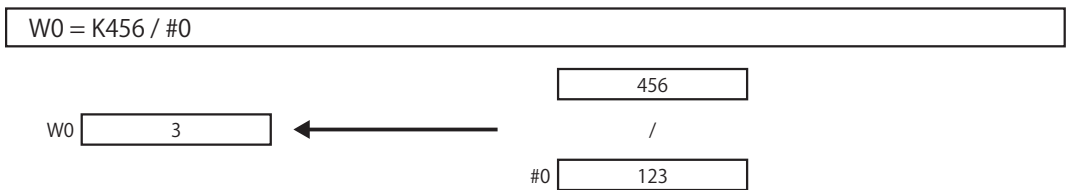
出错

以下情况下将发生运算出错。

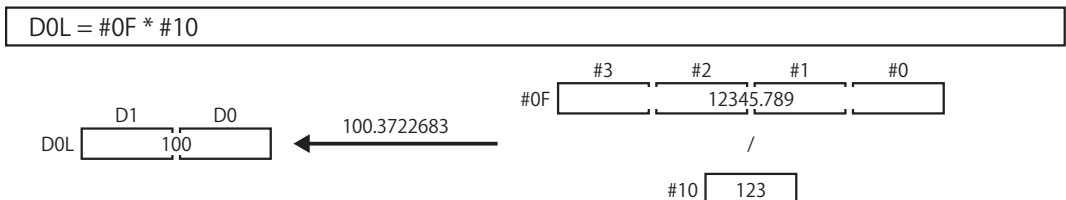
- (S2)为0时。
- (S1)或(S2)为间接指定软元件，软元件编号超出范围时。

程序示例

n 将K456用#0相除，将商代入W0中的程序



n 将#0F用#10相除，将商代入D0L中的程序



以64位浮点型进行除法运算，将商转换为32位整数型后进行代入。

余数：%

格式	基本步数	可用步	
		F/FS	G
(S1) % (S2)	7	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据								计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数						
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)				
(S1)	—	○	○	○	○	○	○	○	—	—	
(S2)	—	○	○	○	○	○	○	○	—	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	被除数据	(S1) 或 (S2) 中较大一方的数据类型 (整数型)
(S2)	除法数据	

功能

- 将(S1)中指定的数据用(S2)相除，求出余数。
- (S1)与(S2)的数据类型不相同的情况下，转换为较大类型后进行运算。

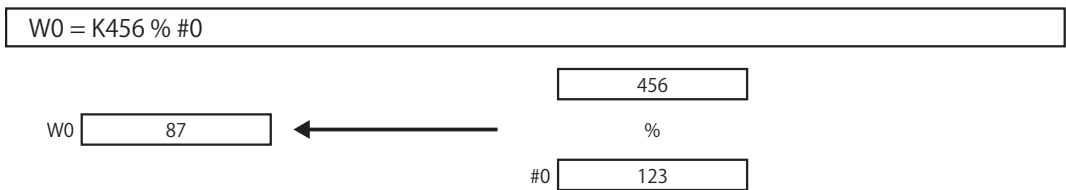
出错

以下情况下将发生运算出错。

- (S2)为0时。
- (S1)或(S2)为间接指定软元件，软元件编号超出范围时。

程序示例

n 将K456用#0相除，将余数代入W0中的程序



4.6 位运算

位取反(补数): ~

格式	基本步数	可用步	
		F/FS	G
~(S)	4	○	○

设置数据

n 可用数据

○: 可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	—	○	○	—	○	○	—	○	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	进行位取反的数据	(S)的数据类型(整数型)

功能

求出(S)中指定的数据的位取反值。

出错

以下情况下将发生运算出错。

- (S)为间接指定软元件，软元件编号超出范围时。

程序示例

n 求出#0的位取反值，代入到D0中的程序

D0 = ~#0																																		
b15 ----- b0 D0 <table border="1" style="display: inline-table; vertical-align: middle;"> <tr><td>1</td><td>1</td><td>0</td><td>1</td><td>1</td><td>0</td><td>1</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td><td>1</td></tr> </table>	1	1	0	1	1	0	1	0	1	1	0	0	1	0	1	1	←	b15 ----- b0 #0 <table border="1" style="display: inline-table; vertical-align: middle;"> <tr><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td></tr> </table>	0	0	1	0	0	1	0	1	0	0	1	1	0	1	0	0
1	1	0	1	1	0	1	0	1	1	0	0	1	0	1	1																			
0	0	1	0	0	1	0	1	0	0	1	1	0	1	0	0																			

位逻辑积：&

格式	基本步数	可用步	
		F/FS	G
(S1) & (S2)	7	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据								计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数						
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)				
(S1)	—	○	○	—	○	○	—	○	—	—	
(S2)	—	○	○	—	○	○	—	○	—	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	进行各位的逻辑积运算的数据	(S1)或(S2)中较大一方的数据类型(整数型)
(S2)		

功能

- 求出(S1)中指定的数据与(S2)中指定的数据的各位的逻辑积。
- (S1)与(S2)的数据类型不相同的情况下，转换为较大类型后进行运算。此时以带符号进行转换，应加以注意。

出错

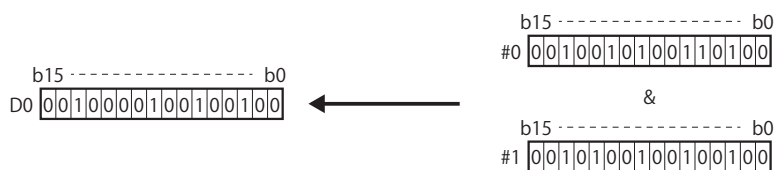
以下情况下将发生运算出错。

- (S1)或(S2)为间接指定软元件，软元件编号超出范围时。

程序示例

n 求出#0与#1的逻辑积，代入到D0中的程序

```
D0 = #0 & #1
```



位逻辑和： |

格式	基本步数	可用步	
		F/FS	G
(S1) (S2)	7	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据								计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数						
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)				
(S1)	—	○	○	—	○	○	—	○	—	—	
(S2)	—	○	○	—	○	○	—	○	—	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	进行各位的逻辑和运算的数据	(S1) 或 (S2) 中较大一方的数据类型 (整数型)
(S2)		

功能

- 求出 (S1) 中指定的数据与 (S2) 中指定的数据的各位的逻辑和。
- (S1) 与 (S2) 的数据类型不相同的情况下，转换为较大类型后进行运算。此时以带符号进行转换，应加以注意。

出错

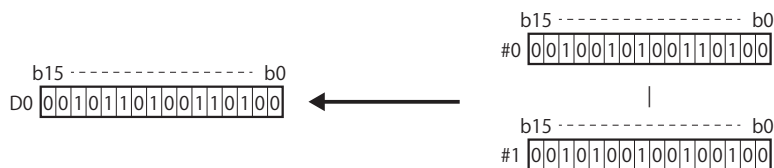
以下情况下将发生运算出错。

- (S1) 或 (S2) 为间接指定软元件，软元件编号超出范围时。

程序示例

n 求出#0与#1的逻辑和，代入D0中的程序

```
D0 = #0 | #1
```



位异或: ^

格式	基本步数	可用步	
		F/FS	G
(S1) ^ (S2)	7	○	○

设置数据

n 可用数据

○: 可以设置

设置数据	可用数据								计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数						
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)				
(S1)	—	○	○	—	○	○	—	○	—	—	
(S2)	—	○	○	—	○	○	—	○	—	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	进行各位的异或运算的数据	(S1) 或 (S2) 中较大一方的数据类型 (整数型)
(S2)		

功能

- 求出 (S1) 中指定的数据与 (S2) 中指定的数据的各位的异或。
- (S1) 与 (S2) 的数据类型不相同的情况下, 转换为较大类型后进行运算。此时以带符号进行转换, 应加以注意。

出错

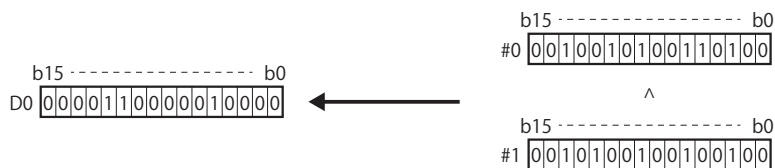
以下情况下将发生运算出错。

- (S1) 或 (S2) 为间接指定软元件, 软元件编号超出范围时。

程序示例

n 求出#0与#1的异或, 代入到D0中的程序

```
D0 = #0 ^ #1
```



位右移: >>

格式	基本步数	可用步	
		F/FS	G
(S1) >> (S2)	7	○	○

设置数据

n 可用数据

○: 可以设置

设置数据	可用数据								计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数						
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)				
(S1)	—	○	○	—	○	○	—	○	—	—	
(S2)	—	○	○	—	○	○	—	○	—	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	进行右移的数据	(S1) 的数据类型 (整数型)
(S2)	右移次数	

功能

- 将(S1)中指定的数据按(S2)中指定的数据的次数进行右移。
- (S1)的最高位为1的情况下,在右移结果的最高位放入1。(S1)的最高位为0的情况下,在右移结果的最高位放入0。
- (S1)为16位整数型,(S2)为负数或16以上时结果将变为0。
- (S1)为32位整数型,(S2)为负数或32以上时结果将变为0。

出错

以下情况下将发生运算出错。

- (S1)或(S2)为间接指定软元件,软元件编号超出范围时。

程序示例

n 将#0进行2位右移后,代入到D0中的程序

```
D0 = #0 >> K2
```



位左移: <<

格式	基本步数	可用步	
		F/FS	G
(S1) << (S2)	7	○	○

设置数据

n 可用数据

○: 可以设置

设置数据	可用数据								计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数						
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)				
(S1)	—	○	○	—	○	○	—	○	—	—	
(S2)	—	○	○	—	○	○	—	○	—	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	进行左移的数据	(S1) 的数据类型 (整数型)
(S2)	左移次数	

功能

- 将(S1)中指定的数据按(S2)中指定的数据的次数进行左移。
- 在左移结果的最低位放入0。
- (S1)为16位整数型, (S2)为负数或16以上时结果将变为0。
- (S1)为32位整数型, (S2)为负数或32以上时结果将变为0。

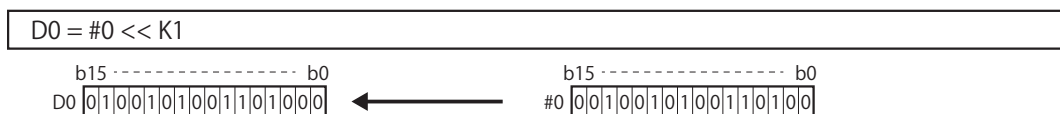
出错

以下情况下将发生运算出错。

- (S1)或(S2)为间接指定软元件, 软元件编号超出范围时。

程序示例

n 将#0进行1位左移后, 代入到D0中的程序



符号取反(2的补数): -

格式	基本步数	可用步	
		F/FS	G
-(S)	4	○	○

设置数据

n 可用数据

○: 可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	—	○	○	○	○	○	○	○	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	进行符号取反的数据	(S)的数据类型

功能

求出(S)中指定的数据的符号取反后的值。

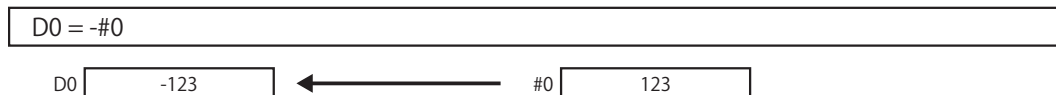
出错

以下情况下将发生运算出错。

- (S)为间接指定软元件，软元件编号超出范围时。

程序示例

n 将#0的符号取反后的值代入到D0中的程序



4.7 标准函数

正弦：SIN

格式	基本步数	可用步	
		F/FS	G
SIN(S)	4	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	—	○	○	○	○	○	○	○	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	进行SIN(正弦)运算的角度数据	浮点型

功能

- 进行(S)中指定的数据的SIN(正弦)运算。
- (S)中指定的数据的单位为角度[degree]。
- (S)为整数型的情况下，转换为浮点型后进行运算。

出错

以下情况下将发生运算出错。

- (S)为间接指定软元件，软元件编号超出范围时。

程序示例

n 进行D0的SIN运算后，代入到#0F中的程序

```
#0F = SIN(D0)
```



余弦：COS

格式	基本步数	可用步	
		F/FS	G
COS(S)	4	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	—	○	○	○	○	○	○	○	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	进行COS(余弦)运算的角度数据	浮点型

功能

- 进行(S)中指定的数据的COS(余弦)运算。
- (S)中指定的数据的单位为角度[degree]。
- (S)为整数型的情况下，转换为浮点型后进行运算。

出错

以下情况下将发生运算出错。

- (S)为间接指定软元件，软元件编号超出范围时。

程序示例

n 进行D0的COS运算后，代入到#0F中的程序



正切: TAN

格式	基本步数	可用步	
		F/FS	G
TAN(S)	4	○	○

设置数据

n 可用数据

○: 可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	—	○	○	○	○	○	○	○	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	进行TAN(正切)运算的角度数据	浮点型

功能

- 进行(S)中指定的数据的TAN(正切)运算。
- (S)中指定的数据的单位为角度[degree]。
- (S)为整数型的情况下,转换为浮点型后进行运算。

出错

以下情况下将发生运算出错。

- (S)为间接指定软元件,软元件编号超出范围时。
- (S)为 $90+(180*n)$ 的情况下。(n为整数)

程序示例

n 进行D0的TAN运算后,代入到#0F中的程序



反正弦：ASIN

格式	基本步数	可用步	
		F/FS	G
ASIN(S)	4	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	—	○	○	○	○	○	○	○	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	进行 SIN^{-1} (反正弦)运算的SIN值数据	浮点型

功能

- 进行(S)中指定的SIN值数据的 SIN^{-1} (反正弦)运算，求出角度。
- (S)中指定的SIN值必须在-1.0~1.0的范围内。
- 运算结果的单位为角度[degree]。
- (S)为整数型的情况下，转换为浮点型后进行运算。

出错

以下情况下将发生运算出错。

- (S)超出-1.0~1.0的范围时。
- (S)为间接指定软元件，软元件编号超出范围时。

程序示例

n 进行D0的 SIN^{-1} (反正弦)运算后，代入到#0F中的程序

```
#0F = ASIN(D0)
```



反余弦：ACOS

格式	基本步数	可用步	
		F/FS	G
ACOS(S)	4	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	—	○	○	○	○	○	○	○	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	进行 COS^{-1} (反余弦)运算的COS值数据	浮点型

功能

- 进行(S)中指定的COS值数据的 COS^{-1} (反余弦)运算，求出角度。
- (S)中指定的COS值必须在-1.0~1.0的范围内。
- 运算结果的单位为角度[degree]。
- (S)为整数型的情况下，转换为浮点型后进行运算。

出错

以下情况下将发生运算出错。

- (S)超出-1.0~1.0的范围时。
- (S)为间接指定软元件，软元件编号超出范围时。

程序示例

n 进行D0F的 COS^{-1} (反余弦)运算后，代入到#0F中的程序



反正切：ATAN

格式	基本步数	可用步	
		F/FS	G
ATAN(S)	4	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	—	○	○	○	○	○	○	○	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	进行 TAN^{-1} (反正切)运算的TAN值数据	浮点型

功能

- 进行(S)中指定的TAN值数据的 TAN^{-1} (反正切)运算，求出角度。
- 运算结果的单位为角度[degree]。
- (S)为整数型的情况下，转换为浮点型后进行运算。

出错

以下情况下将发生运算出错。

- (S)为间接指定软元件，软元件编号超出范围时。

程序示例

n 进行DOF的 TAN^{-1} (反正切)运算后，代入到#0F中的程序



平方根：SQRT

格式	基本步数	可用步	
		F/FS	G
SQRT (S)	4	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	—	○	○	○	○	○	○	○	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	进行平方根运算的数据	浮点型

功能

- 求出(S)中指定的数据的平方根。
- (S)中指定的值只能为正数。(不能以负数进行运算。)
- (S)为整数型的情况下，转换为浮点型后进行运算。

出错

以下情况下将发生运算出错。

- (S)为负数时。
- (S)为间接指定软元件，软元件编号超出范围时。

程序示例

n 求出D0F的平方根后，代入到#0F中的程序



自然对数：LN

格式	基本步数	可用步	
		F/FS	G
LN(S)	4	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	—	○	○	○	○	○	○	○	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	进行自然对数运算的数据	浮点型

功能

- 求出以(S)中指定的数据e为底的自然对数。
- (S)中指定的值只能为正数。(不能以负数进行运算。)
- (S)为整数型的情况下，转换为浮点型后进行运算。

出错

以下情况下将发生运算出错。

- (S)为0或负数时。
- (S)为间接指定软元件，软元件编号超出范围时。

程序示例

n 求出D0F的自然对数后，代入到#0F中的程序



指数运算：EXP

格式	基本步数	可用步	
		F/FS	G
EXP (S)	4	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	—	○	○	○	○	○	○	○	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	进行指数运算的数据	浮点型

功能

- 以e为底，将(S)中指定的数据作为指数进行幂运算。
- (S)为整数型的情况下，转换为浮点型后进行运算。

出错

以下情况下将发生运算出错。

- (S)为间接指定软元件，软元件编号超出范围时。

程序示例

n 进行DOF的指数运算后，代入到#0F中的程序



绝对值：ABS

格式	基本步数	可用步	
		F/FS	G
ABS (S)	4	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	—	○	○	○	○	○	○	○	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	进行绝对值转换的数据	(S)的数据类型

功能

求出(S)中指定的数据的绝对值。

出错

以下情况下将发生运算出错。

- (S)为16位整数型，超出-32767~32767的范围时。
- (S)为32位整数型，超出-2147483647~2147483647的范围时。
- (S)为间接指定软元件，软元件编号超出范围时。

程序示例

n 求出D0F的绝对值后，代入到#0F中的程序



四舍五入: RND

格式	基本步数	可用步	
		F/FS	G
RND (S)	4	○	○

设置数据

n 可用数据

○: 可以设置

设置数据	可用数据								计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数						
		16位整数型	32位整数型 (L)	64位浮点型 (F)	16位整数型 (K/H)	32位整数型 (K/H、L)	64位浮点型 (K)				
(S)	—	○	○	○	○	○	○	○	—	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	对小数点以下进行四舍五入的数据	(S) 的数据类型

功能

- 求出 (S) 中指定的数据的小数点以下四舍五入后的值。
- (S) 为负数的情况下, 求出 (S) 的绝对值, 将小数点以下进行四舍五入变为附加了符号的值。
- (S) 为整数型的情况下不进行转换处理, 将 (S) 的值原样不变地返回。

出错

以下情况下将发生运算出错。

- (S) 为间接指定软元件, 软元件编号超出范围时。

程序示例

n 求出对D0F的小数点以下进行了四舍五入后的值, 并将其代入到#0F中的程序



n 求出对D4F的小数点以下进行了四舍五入后的值, 并将其代入到#0F中的程序(负数的情况下)



舍去: FIX

格式	基本步数	可用步	
		F/FS	G
FIX(S)	4	○	○

设置数据

n 可用数据

○: 可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	—	○	○	○	○	○	○	○	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	进行小数点以下舍去的数据	(S)的数据类型

功能

- 求出不大于(S)中指定的数据的最大整数。
- (S)的值为正的情况下绝对值将变小,为负的情况下将变大。
- (S)为整数型的情况下不进行转换处理,将(S)的值原样不变地返回。

出错

以下情况下将发生运算出错。

- (S)为间接指定软元件,软元件编号超出范围时。

程序示例

n 求出对D0F的小数点以下进行了舍去后的值,并将其代入到#0F中的程序



n 求出对D4F的小数点以下进行了舍去的值后,代入到#0F中的程序(负数的情况下)



进位：FUP

格式	基本步数	可用步	
		F/FS	G
FUP(S)	4	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	—	○	○	○	○	○	○	○	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	进行小数点以下进位的数据	(S)的数据类型

功能

- 求出不小于(S)中指定的数据的最小的整数。
- (S)的值为正数的情况下绝对值将变大，为负数的情况下将变小。
- (S)为整数型的情况下不进行转换处理，将(S)的值原样不变地返回。

出错

以下情况下将发生运算出错。

- (S)为间接指定软元件，软元件编号超出范围时。

程序示例

n 求出对D0F的小数点以下进行了进位的值后，代入到#0F中的程序



n 求出对D4F的小数点以下进行了进位的值后，代入到#0F中的程序(负数的情况下)



BCD → BIN转换：BIN

格式	基本步数	可用步	
		F/FS	G
BIN(S)	4	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据								计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数						
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)				
(S)	—	○	○	—	○	○	—	○	—	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	转换为BIN数据的BCD数据	(S)的数据类型(整数型)

功能

- 将(S)中指定的BCD数据转换为BIN数据。
- (S)为16位整数型的情况下，数据范围为0~9999。
- (S)为32位整数型的情况下，数据范围为0~99999999。

出错

以下情况下将发生运算出错。

- (S)的各位中有0~9以外的值时。
- (S)为间接指定软元件，软元件编号超出范围时。

程序示例

n 将D0的BCD数据转换为BIN数据后，代入到#0中的程序

```
#0 = BIN(D0)
```



BIN → BCD转换：BCD

格式	基本步数	可用步	
		F/FS	G
BCD (S)	4	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型 (L)	64位浮点型 (F)	16位整数型 (K/H)	32位整数型 (K/H、L)	64位浮点型 (K)			
(S)	—	○	○	—	○	○	—	○	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	转换为BCD数据的BIN数据	(S)的数据类型(整数型)

功能

- 将(S)中指定的BIN数据转换为BCD数据。
- (S)为16位整数型的情况下，数据范围为0~9999。
- (S)为32位整数型的情况下，数据范围为0~99999999。

出错

以下情况下将发生运算出错。

- (S)为16位整数型的情况下数据超出0~9999的范围时。
- (S)为32位整数型的情况下数据超出0~99999999的范围时。
- (S)为间接指定软元件，软元件编号超出范围时。

程序示例

n 将D0的BIN数据转换为BCD数据后，代入到#0中的程序

```
#0 = BCD(D0)
```



4.8 数据控制

16位整数型标度：SCL

格式	基本步数	可用步	
		F/FS	G
SCL (S1), (S2), (S3), (D)	15	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S1)	—	○	—	—	○	—	—	—	—	—
(S2)	—	○	—	—	○	—	—	○	—	—
(S3)	—	○	—	—	—	—	—	—	—	—
(D)	—	○	—	—	—	—	—	—	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	指定搜索方法/转换方法的数据 0: 通过逐次搜索进行的正转换 1: 通过逐次搜索进行的逆转换 2: 通过二分搜索进行的正转换 3: 通过二分搜索进行的逆转换	—
(S2)	正转换/逆转换用的输入值	
(S3)	存储了标度用转换数据的软元件的起始编号	
(D)	存储转换结果的软元件的编号	

功能

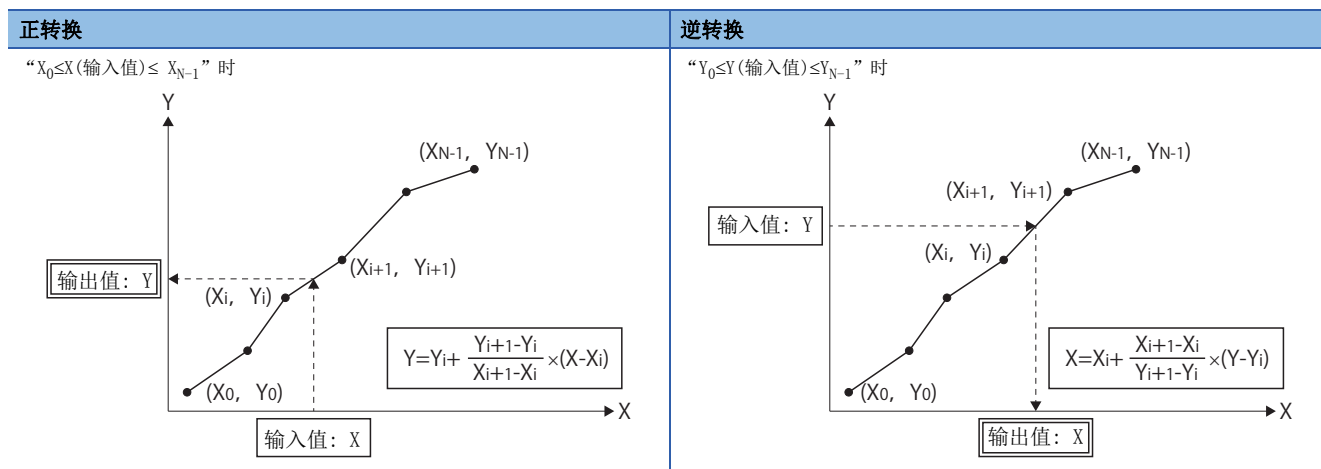
- 对于16位整数型标度，以最大4000点的点数 $((X_0, Y_0) \sim (X_{N-1}, Y_{N-1}))$ ，N: 点数)定义的标度用转换数据为基础，通过设置的输入值计算输出值。输入值对应的点数据必须以升序(正转换： $X_0 < X_1 < \dots < X_{N-1}$ ，逆转换： $Y_0 < Y_1 < \dots < Y_{N-1}$)进行设置。

- 输出值的计算方法中有正转换(输入值: 点X, 输出值: 点Y)及逆转换(输入值: 点Y, 输出值: 点X), 通过(S1)进行指定。各个计算方法如下所示。

n 输入值存在于标度用转换数据的任意2点之间的情况下

通过输入值前后的点计算输出值。

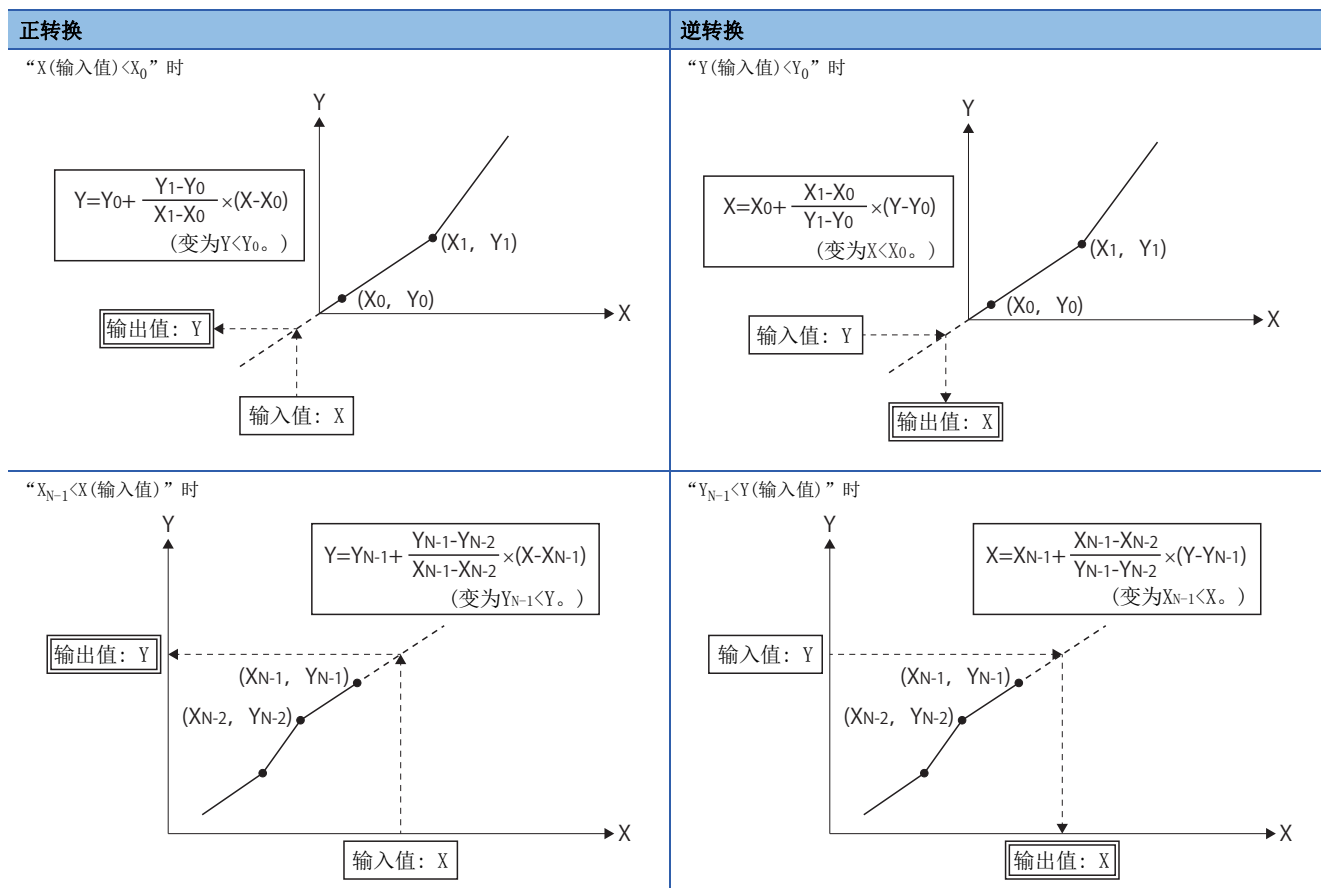
N: 点数



n 输入值不存在于标度用转换数据的任意2点之间的情况下

使用标度用转换数据的起始或终端的2点计算输出值。

N: 点数



要点

输入值超出标度用转换数据范围时, 输出值的计算结果超出一32768~32767的范围时, 将发生运算出错。

- 使用从(S3)开始的软元件以后的标度用转换数据，按照(S1)中指定的搜索方法/转换方法进行(S2)中指定的输入值的转换。转换结果存储在(D)中指定的软元件中。
- 在标度中，需要从输入值开始搜索输出值的计算中使用的点数据，通过(S1)指定搜索方法。搜索方法中有逐次搜索及二分搜索，其特点如下所示。应根据使用目的指定搜索方法。

搜索方法	点数4000时的搜索次数	处理时间	注意事项
逐次搜索	1~4000次	0.003~0.809[ms] 进行从起始数据开始按编号顺序查找的逐次查找，因此最大处理时间与点数成正比。	搜索处理中，可确认输入值对应的点数据处于升序。输入值对应的点数据不处于升序的情况下，将发生运算出错。
二分搜索	12次	0.006[ms] 由于使用二分搜索法，可在与点数不成比例的状况下以较短的时间进行搜索。	搜索处理中，仅参照二分搜索中所必要的点数据。由于无法确认输入值对应的全部点数据，因此该数据未以升序排列时，输出值将可能得出意外的计算结果。

- 应将(S3)中指定的软元件编号设置为偶数编号，在指定的软元件中按以下方式设置点数据。

偏置	名称	内容	范围
+0	点数(N)	设置标度用转换数据的点数。	2~4000
+1	用户不能使用	应设置为0。	0
+2	点0	应将 $(X_0, Y_0) \sim (X_{N-1}, Y_{N-1})$ 的点数据以连续的软元件编号进行设置。	-32768~32767
+3			
+4	点1	X_1	
+5		Y_1	
+6	点2	X_2	
+7		Y_2	
⋮	⋮		
+(2N)	点(N-1)	X_{N-1}	
+(2N+1)		Y_{N-1}	

要点

输入值对应的点数据必须以升序(正转换: $X_0 < X_1 < \dots < X_{N-1}$, 逆转换: $Y_0 < Y_1 < \dots < Y_{N-1}$)进行设置。

- (D)中指定的软元件中存储的转换结果不是整数值的的情况下，将小数点以下舍去。

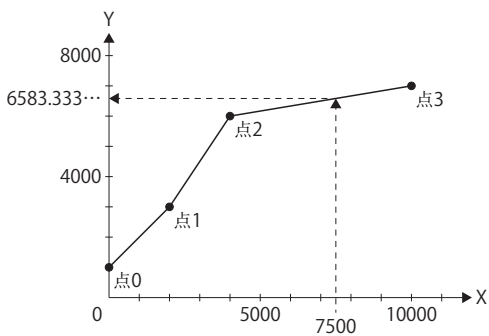
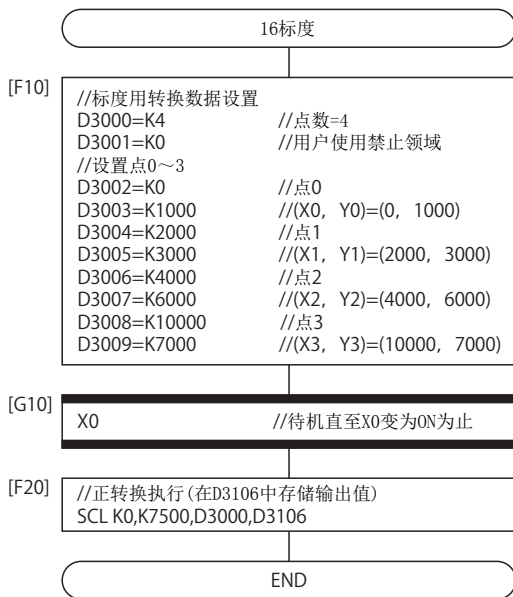
出错

以下情况下将发生运算出错，不进行输入值的转换。

- (S1)中设置了0~3以外时。
- (S3)不是偶数编号的软元件时。
- (S3)中指定的点设置一览表的点数超出了2~4000的范围时。
- (S3)中指定的点设置一览表超出软元件范围时。
- 逐次搜索((S1)为0或1)中，输入值对应的点(正转换: $X_0 \sim X_{N-1}$, 逆转换: $Y_0 \sim Y_{N-1}$)不是升序排列时。
- 转换结果超出-32768~32767的范围时。

程序示例

n 将4点的标度用转换数据设置到D3000~D3009中，将以输入值“7500”为基础进行了正转换的输出值代入到D3106中的程序



• 因为输入值“7500”存在与点2、3之间，所以通过点2、3的设置值计算输出值。

$$\text{输出值} = 6000 + \frac{7000 - 6000}{10000 - 4000} \times (7500 - 4000)$$

$$= 6583.333 \dots$$

↓ 小数点以下舍去

D3106 K6583

32位整数型标度：DSCL

格式	基本步数	可用步	
		F/FS	G
DSCL (S1), (S2), (S3), (D)	15	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S1)	—	○	—	—	○	—	—	—	—	—
(S2)	—	—	○	—	—	○	—	○	—	—
(S3)	—	○	—	—	—	—	—	—	—	—
(D)	—	—	○	—	—	—	—	—	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	指定搜索方法/转换方法的数据 0: 通过逐次搜索进行的正转换 1: 通过逐次搜索进行的逆转换 2: 通过二分搜索进行的正转换 3: 通过二分搜索进行的逆转换	—
(S2)	正转换/逆转换用的输入值	
(S3)	存储了标度用转换数据的软元件的起始编号	
(D)	存储转换结果的软元件的编号	

功能

- 对于32位整数型标度，以最大2000点的点数 $((X_0, Y_0) \sim (X_{N-1}, Y_{N-1}))$ ，N: 点数定义的标度用转换数据为基础，通过设置的输入值计算输出值。输入值对应的点数据必须以升序(正转换： $X_0 < X_1 < \dots < X_{N-1}$ ，逆转换： $Y_0 < Y_1 < \dots < Y_{N-1}$)进行设置。
- 输出值的计算方法与16位整数型标度的相同。(P149页 输入值存在于标度用转换数据的任意2点之间的情况下、P149页 输入值不存在于标度用转换数据的任意2点之间的情况下)

要点

输入值超出标度用转换数据范围时，输出值的计算结果超出-2147483648~2147483647的范围时，将发生运算出错。

- 使用从(S3)开始的软元件以后的标度用转换数据，按照(S1)中指定的搜索方法/转换方法进行(S2)中指定的输入值的转换。转换结果存储在(D)中指定的软元件中。
- (S1)的设置方法与16位整数型标度的相同。(☞ 148页 16位整数型标度：SCL)
- 应将(S3)中指定的软元件编号设置为偶数编号，在指定的软元件中按以下方式设置点数据。

偏置	名称	内容	范围
+0	点数(N)	设置标度用转换数据的点数。	2~2000
+1	用户不能使用	应设置为0。	0
+2	点0	应将 $(X_0, Y_0) \sim (X_{N-1}, Y_{N-1})$ 的点数据以连续的软元件编号进行设置。	-2147483648 ~ 2147483647
+3			
+4	Y_0		
+5		X_1	
+6	点1		
+7		Y_1	
+8	X_2		
+9		点2	
+10	Y_2		
+11		X_{N-1}	
+12	点(N-1)		
+13		Y_{N-1}	
⋮	⋮		
+(4N-2)	点(N-1)	X_{N-1}	
+(4N-1)			Y_{N-1}
+(4N)		X_{N-1}	
+(4N+1)			Y_{N-1}

要点 

输入值对应的点数据必须以升序(正转换： $X_0 < X_1 < \dots < X_{N-1}$ ，逆转换： $Y_0 < Y_1 < \dots < Y_{N-1}$)进行设置。

- (D)中指定的软元件中存储的转换结果不是整数值的情况下，将小数点以下舍去。

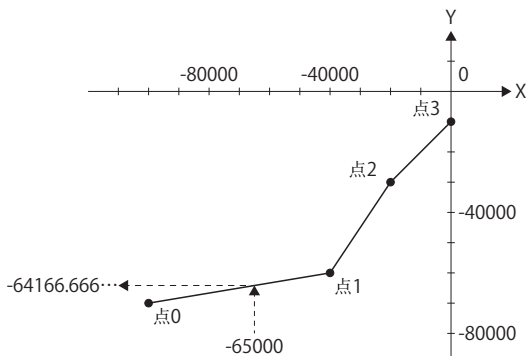
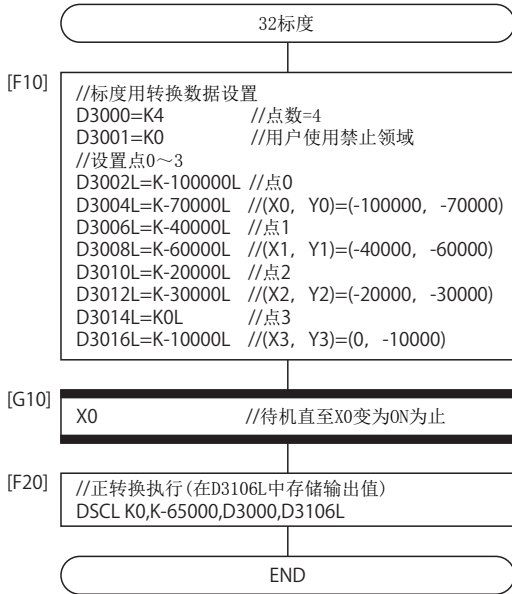
出错

以下情况下将发生运算出错，不进行输入值的转换。

- (S1)中设置了0~3以外时。
- (S2)、(S3)、(D)不是偶数编号的软元件时。
- (S3)中指定的点设置一览表的数据超出了2~2000的范围时。
- (S3)中指定的点设置一览表超出软元件范围时。
- 逐次搜索((S1)为0或1)中，输入值对应的点(正转换： $X_0 \sim X_{N-1}$ ，逆转换： $Y_0 \sim Y_{N-1}$)不是升序排列时。
- 转换结果超出-2147483648~2147483647的范围时。

程序示例

n 将4点的标度用转换数据设置到D3000~D3017中，将以输入值“-65000”为基础进行了正转换的输出值代入到D3106L中的程序



• 因为输入值“-65000”存在于点0、1之间，所以通过点0、1的设置值计算输出值。

$$\text{输出值} = -70000 + \frac{(-60000 - (-70000))}{(-40000 - (-100000))} \times (-65000 - (-100000))$$

$$= -64166.666\cdots$$

↓ 小数点以下舍去

D3106	D3107
K-64166	

4.9 类型转换

带符号16位整数值转换：SHORT

格式	基本步数	可用步	
		F/FS	G
SHORT (S)	4	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据								计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数						
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)				
(S)	—	○	○	○	○	○	○	○	—	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	进行带符号16位整数值转换的数据	16位整数型

功能

- 将(S)中指定的数据转换为带符号16位整数值。
- (S)的数据范围为-32768~32767。
- (S)为64位浮点型时，舍去小数点以下后，进行转换。
- (S)为16位整数型的情况下，不进行转换处理，将(S)的值原样不变地返回。

出错

以下情况下将发生运算出错。

- (S)的数据超出-32768~32767的范围时。
- (S)为间接指定软元件，软元件编号超出范围时。

程序示例

n 将D0L的数据转换为带符号16位整数值后，代入到#0中的程序

```
#0 = SHORT(D0L)
```



无符号16位整数值转换：USHORT

格式	基本步数	可用步	
		F/FS	G
USHORT (S)	4	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	—	○	○	○	○	○	○	○	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	进行无符号16位整数值转换的数据	16位整数型

功能

- 将(S)中指定的数据转换为无符号16位整数值。
- (S)的数据范围为0~65535。
- (S)为64位浮点型时，舍去小数点以下后，进行转换。
- (S)为16位整数型的情况下，不进行转换处理，将(S)的值原样不变地返回。

出错

以下情况下将发生运算出错。

- (S)的数据超出0~65535的范围时。
- (S)为间接指定软元件，软元件编号超出范围时。

程序示例

n 将D0L的数据转换为无符号16位整数值后，代入到#0中的程序

```
#0 = USHORT(D0L)
```



进行数据类型不同的二项运算的情况下，将转换为数据类型较大一方后进行运算，因此USHORT不生效。对象二项运算为以下8个。

- 加法(+)
- 减法(-)
- 乘法(*)
- 除法(/)
- 余数(%)
- 位逻辑积(&)
- 位逻辑和(|)
- 位异或(^)

(例)

W0:F=#0F+USHORT(D0L)

↑ ↑
64位浮点型 USHORT将不变为有效。

带符号32位整数值转换：LONG

格式	基本步数	可用步	
		F/FS	G
LONG (S)	4	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	—	○	○	○	○	○	○	○	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	进行带符号32位整数值转换的数据	32位整数型

功能

- 将(S)中指定的数据转换为带符号32位整数值。
- (S)的数据范围为-2147483648~2147483647。
- (S)为64位浮点型时，舍去小数点以下后，进行转换。
- (S)为32位整数型的情况下，不进行转换处理，将(S)的值原样不变地返回。

出错

以下情况下将发生运算出错。

- (S)的数据超出-2147483648~2147483647的范围时。
- (S)为间接指定软元件，软元件编号超出范围时。

程序示例

n 将D0的数据转换为带符号32位整数值后，代入到#0L中的程序

```
#0L = LONG(D0)
```



无符号32位整数值转换：ULONG

格式	基本步数	可用步	
		F/FS	G
ULONG (S)	4	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	—	○	○	○	○	○	○	○	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	进行无符号32位整数值转换的数据	32位整数型

功能

- 将(S)中指定的数据转换为无符号32位整数值。
- (S)的数据范围为0~4294967295。
- (S)为64位浮点型时，舍去小数点以下后，进行转换。
- (S)为32位整数型的情况下，不进行转换处理，将(S)的值原样不变地返回。

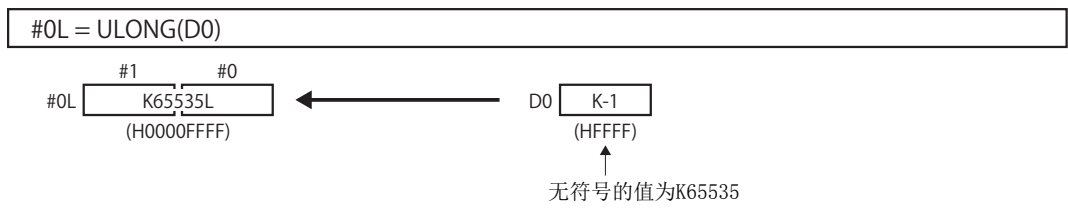
出错

以下情况下将发生运算出错。

- (S)的数据超出0~4294967295的范围时。
- (S)为间接指定软元件，软元件编号超出范围时。

程序示例

n 将D0的数据转换为无符号32位整数值后，代入到#0L中的程序



要点

进行数据类型不同的二项运算的情况下，将转换为数据类型较大一方后进行运算，因此ULONG不生效。对象二项运算为以下8个。

- 加法(+)
- 减法(-)
- 乘法(*)
- 除法(/)
- 余数(%)
- 位逻辑积(&)
- 位逻辑和(|)
- 位异或(^)

(例)

W0:F=#0F+ULONG(D0L)

↑ ↑
64位浮点型 ULONG将不变为有效。

带符号64位浮点值转换：FLOAT

格式	基本步数	可用步	
		F/FS	G
FLOAT(S)	4	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	—	○	○	○	○	○	○	○	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	进行带符号64位浮点值转换的数据	64位浮点型

功能

- 将(S)中指定的数据转换为带符号64位浮点值。
- (S)为64位浮点型的情况下，不进行转换处理，将(S)的值原样不变地返回。

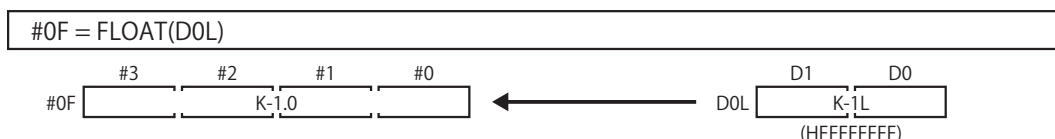
出错

以下情况下将发生运算出错。

- (S)为间接指定软元件，软元件编号超出范围时。

程序示例

n 将D0L的数据转换为带符号64位浮点值数据后，代入到#0F中的程序



无符号64位浮点值转换：UFLOAT

格式	基本步数	可用步	
		F/FS	G
UFLOAT(S)	4	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据								计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数						
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)				
(S)	—	○	○	○	○	○	○	○	—	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	进行无符号64位浮点值转换的数据	64位浮点型

功能

- 将(S)中指定的数据转换为无符号64位浮点值。
- (S)为64位浮点型的情况下，不进行转换处理，将(S)的值原样不变地返回。

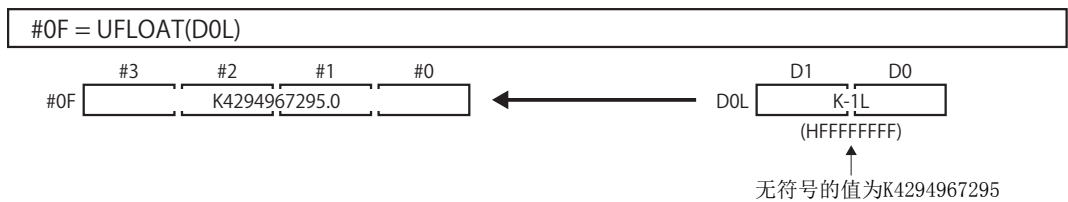
出错

以下情况下将发生运算出错。

- (S)为间接指定软元件，软元件编号超出范围时。

程序示例

n 将D0L的数据转换为无符号64位浮点值数据后，代入到#0F中的程序



浮点值的32位 → 64位转换：DFLT

格式	基本步数	可用步	
		F/FS	G
DFLT (S)	4	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	—	—	○*1	—	—	—	—	—	—	—

*1 程序上作为32位整数型处理，但在软元件中应存储32位浮点型的数据。

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	进行64位浮点值转换的数据	64位浮点型

功能

- 将(S)中指定的软元件中存储的32位浮点(单精度实数)值，转换为64位浮点(双精度实数)值。可转换的数据范围如下所示。
 $-3.40 \times 10^{38} \sim -1.18 \times 10^{-38}$, 0.0, $1.18 \times 10^{-38} \sim 3.40 \times 10^{38}$ (单精度实数)
- 在运动SFC程序中，作为浮点型的数据使用64位浮点型。从外部设备输入32位浮点型数据的情况下，应通过本指令进行转换。

出错

以下情况下将发生运算出错。

- (S)的数据不是有效的32位浮点型时。

程序示例

n 将D2000L的32位浮点值数据转换为64位浮点值数据后，代入到#0F中的程序



浮点值的64位 → 32位转换：SFLT

格式	基本步数	可用步	
		F/FS	G
SFLT (S)	4	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	—	—	—	○	—	—	—	○	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	进行32位浮点值转换的数据	32位浮点型

功能

- 将(S)中指定的软元件中存储的64位浮点(双精度实数)值,转换为32位浮点(单精度实数)值。可转换的数据范围如下所示。
 $-3.40 \times 10^{38} \sim -1.18 \times 10^{-38}$, 0.0, $1.18 \times 10^{-38} \sim 3.40 \times 10^{38}$ (单精度实数)
- 在运动SFC程序中,作为浮点型的数据使用64位浮点型。将数据输出到不能使用64位浮点型的外部设备中的情况下,应通过本指令进行转换。

要点

32位浮点值数据的有效位数约为7位,因此根据SFLT指令进行转换的结果的第7位以后有可能与(S)的数据不一致。

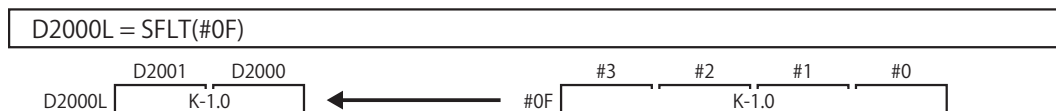
出错

以下情况下将发生运算出错。

- (S)的数据不是有效的64位浮点型时。
- 转换后的(S)的数据超出32位浮点型的范围时。

程序示例

n 将#0F的64位浮点值数据转换为32位浮点值数据后,代入到D2000L中的程序



4.10 位软元件状态

ON(常开触点): (无)

格式	基本步数	可用步	
		F/FS	G
(S)	4	○	○

设置数据

n 可用数据

○: 可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	○	—	—	—	—	—	—	—	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	位条件表达式中使用的位软元件	逻辑型(真/假)

功能

- 位条件表达式中(S)中指定的位软元件为ON(1)时返回真, 为OFF(0)时返回假。

出错

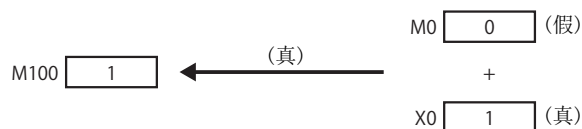
以下情况下将发生运算出错。

- (S)为间接指定软元件, 软元件编号超出范围时。

程序示例

n M0及X0之一为ON(1)时, 设置M100的程序

```
SET M100 = M0 + X0
```



OFF (常闭触点): !

格式	基本步数	可用步	
		F/FS	G
!(S)	4	○	○

设置数据

n 可用数据

○: 可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	○	—	—	—	—	—	—	—	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	位条件表达式中使用的位软元件	逻辑型(真/假)

功能

- 位条件表达式中(S)中指定的位软元件为OFF(0)时返回真, 为ON(1)时返回假。

出错

以下情况下将发生运算出错。

- (S)为间接指定软元件, 软元件编号超出范围时。

程序示例

n M0为OFF(0)时, 设置M100的程序

```
RST M100 = !M0
```

M100 0 ← IM0 0 (真)

4.11 位软元件控制

软元件设置：SET

格式	基本步数	可用步	
		F/FS	G
SET(D)=(S)	8	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(D)	○	—	—	—	—	—	—	—	—	—
(S)	○	—	—	—	—	—	—	—	○	○

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(D)	进行软元件设置的位数据	位逻辑型(真/假)
(S)	确定是否进行软元件设置的条件数据	

功能

- (S)中指定的数据为真时，对(D)中指定的位数据进行设置。
- (S)可以省略。此时格式将变为“SET(D)”，无条件进行软元件设置。
- 转换程序的最终块中作为转移条件进行了设置的情况下，将(S)中指定的数据的真/假作为逻辑型数据返回。在此情况下，不能省略(S)。

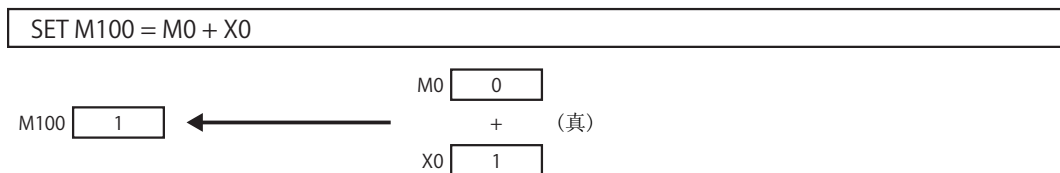
出错

以下情况下将发生运算出错。

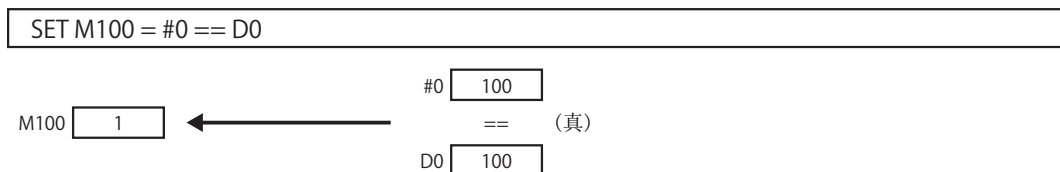
- (D)或(S)为间接指定软元件，软元件编号超出范围时。

程序示例

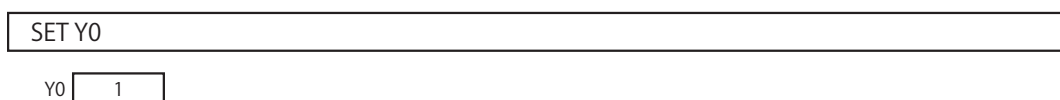
n M0及X0之一为1时，设置M100的程序



n #0与D0一致时，设置M100的程序



n 无条件设置Y0的程序



软元件复位：RST

格式	基本步数	可用步	
		F/FS	G
RST (D)=(S)	8	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(D)	○	—	—	—	—	—	—	—	—	—
(S)	○	—	—	—	—	—	—	—	○	○

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(D)	进行软元件复位的位数据	位逻辑型(真/假)
(S)	确定是否进行软元件复位的条件数据	

功能

- (S)中指定的数据为真时，对(D)中指定的位数据进行复位。
- (S)可以省略。此时格式将变为“RST (D)”，无条件进行软元件复位。
- 转换程序的最终块中作为转移条件进行了设置的情况下，将(S)中指定的数据的真/假作为逻辑型数据返回。在此情况下，不能省略(S)。

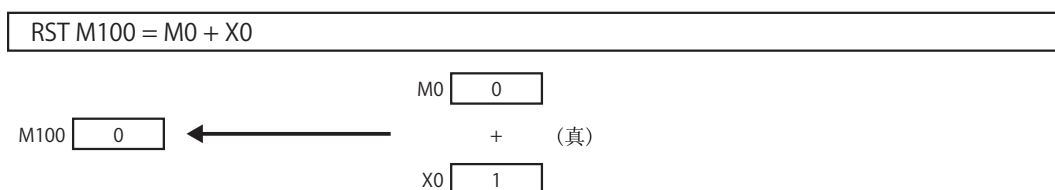
出错

以下情况下将发生运算出错。

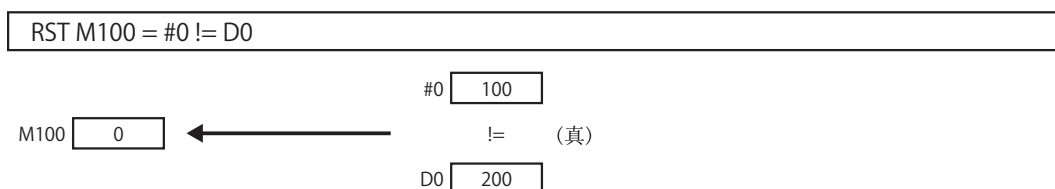
- (D)或(S)为间接指定软元件，软元件编号超出范围时。

程序示例

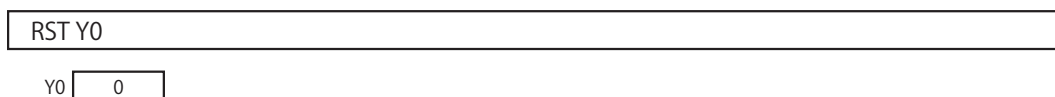
n M0及X0之一为1时，对M100进行复位的程序



n #0与D0不一致时，对M100进行复位的程序



n 无条件对Y0进行复位的程序



软元件输出：DOUT

格式	基本步数	可用步	
		F/FS	G
DOUT (D), (S)	8	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(D)	○	—	—	—	—	—	—	—	—	—
(S)	—	○	○	—	○	○	—	○	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(D)	输出目标位数据	批量位
(S)	输出源数据	

功能

- 将(S)中指定的数据，输出到(D)中指定的位数据中。
- (D)中指定的位数据的软元件编号以16的倍数进行指定。
- (S)的类型为16位整数型的情况下，将(D)中指定的位软元件作为起始，将(S)的数据从最低位开始依次进行16点输出。
- (S)的类型为32位整数型的情况下，将(D)中指定的位软元件作为起始，将(S)的数据从最低位开始依次进行32点输出。

出错

以下情况下将发生运算出错。

- (D)或(S)为间接指定软元件，软元件编号超出范围时。
- (D)为间接指定软元件，软元件编号不是16的倍数时。

程序示例

n 将D0的数据输出到Y0~YF中的程序

DOUT Y0,D0



软元件输入：DIN

格式	基本步数	可用步	
		F/FS	G
DIN (D), (S)	8	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据								计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数						
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)				
(D)	—	○	○	—	—	—	—	—	—	—	
(S)	○	—	—	—	—	—	—	—	—	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(D)	输入目标数据	(D)的数据类型(整数型)
(S)	输入源位数据	

功能

- 将(S)中指定的位数据，输入到(D)中指定的数据中。
- (S)中指定的位数据的软元件编号以16的倍数进行指定。
- (D)的类型为16位整数型的情况下，将(S)中指定的位软元件作为起始，从(D)的数据的最低位开始依次进行16点输入。
- (D)的类型为32位整数型的情况下，将(S)中指定的位软元件作为起始，从(D)的数据的最低位开始依次进行32点输入。

出错

以下情况下将发生运算出错。

- (D)或(S)为间接指定软元件，软元件编号超出范围时。
- (S)为间接指定软元件，软元件编号不是16的倍数时。

程序示例

n 将X0~XF的数据输入到D0中的程序

DIN D0,X0



位软元件输出：OUT

格式	基本步数	可用步	
		F/FS	G
OUT (D)=(S)	8	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(D)	○	—	—	—	—	—	—	—	—	—
(S)	○	—	—	—	—	—	—	—	○	○

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(D)	进行软元件输出的位软元件	位
(S)	软元件输出的条件数据	

功能

- (S)中指定的数据为真时，对(D)中指定的位软元件进行设置，为假时对(D)中指定的位软元件进行复位。
- 转换程序的最终块中作为转移条件进行了设置的情况下，将(S)中指定的数据的真/假作为逻辑型数据返回。
- 不能省略(S)。

出错

以下情况下将发生运算出错。

- (D)或(S)为间接指定软元件，软元件编号超出范围时。

程序示例

n M0为ON的情况下M100变为ON，M0为OFF的情况下M100变为OFF的程序

```
OUT M100 = M0
```

n M0及M1均为ON的情况下M100变为ON，除此以外M100变为OFF的程序

```
OUT M100 = M0 * M1
```

n D0与D2000的值一致时M100变为ON，不一致时M100变为OFF的程序

```
OUT M100 = (D0 == D2000)
```

4.12 逻辑运算

逻辑肯定：(无)

格式	基本步数	可用步	
		F/FS	G
(S)	—	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	○	—	—	—	—	—	—	—	○	○

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	进行逻辑肯定的数据	逻辑型(真/假)

功能

- 将(S)中指定的逻辑型数据的真/假原样不变地返回。(逻辑肯定)

出错

以下情况下将发生运算出错。

- (S)为间接指定软元件，软元件编号超出范围时。

程序示例

n M0及X0之一为ON(1)时，设置M100的程序

```
SET M100 = M0 + X0
```



逻辑否定：!

格式	基本步数	可用步	
		F/FS	G
!(S)	4	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	○	—	—	—	—	—	—	—	○	○

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	进行逻辑否定的数据	逻辑型(真/假)

功能

- 进行(S)中指定的数据的逻辑否定。

出错

以下情况下将发生运算出错。

- (S)为间接指定软元件，软元件编号超出范围时。

程序示例

n 不处于“M0及X0之一为ON(1)”时(M0及X0两方均为OFF(0)时)，对M100进行设置的程序

```
SET M100 = !(M0 + X0)
```



逻辑积：*

格式	基本步数	可用步	
		F/FS	G
(S1)*(S2)	7	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S1)	○	—	—	—	—	—	—	—	○	○
(S2)	○	—	—	—	—	—	—	—	○	○

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	进行逻辑积运算的数据	逻辑型(真/假)
(S2)		

功能

- 求出(S1)中指定的数据与(S2)中指定的数据的逻辑积。

出错

以下情况下将发生运算出错。

- (S)为间接指定软元件，软元件编号超出范围时。

程序示例

n M0及X0均为1时，对M100进行设置的程序

```
SET M100 = M0 * X0
```



逻辑和: +

格式	基本步数	可用步	
		F/FS	G
(S1)+(S2)	7	○	○

设置数据

n 可用数据

○: 可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S1)	○	—	—	—	—	—	—	—	○	○
(S2)	○	—	—	—	—	—	—	—	○	○

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	进行逻辑和运算的数据	逻辑型(真/假)
(S2)		

功能

- 求出(S1)中指定的数据与(S2)中指定的数据的逻辑和。

出错

以下情况下将发生运算出错。

- (S1)或(S2)为间接指定软元件，软元件编号超出范围时。

程序示例

n M0及X0之一为1时，设置M100的程序

```
SET M100 = M0 + X0
```



4.13 比较运算

一致： ==

格式	基本步数	可用步	
		F/FS	G
(S1)==(S2)	7	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据								计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数						
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)				
(S1)	—	○	○	○	○	○	○	○	—	—	
(S2)	—	○	○	○	○	○	○	○	—	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	进行比较的数据	逻辑型(真/假)
(S2)		

功能

- 对(S1)中指定的数据与(S2)中指定的数据进行比较，一致的情况下变为真。
- (S1)与(S2)的数据类型不相同的情况下，转换为类型较大的一方后进行比较。

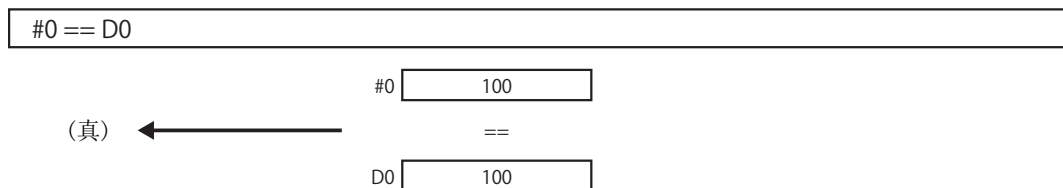
出错

以下情况下将发生运算出错。

- (S1)或(S2)为间接指定软元件，软元件编号超出范围时。

程序示例

n 对#0与D0是否一致进行比较的程序



不一致: !=

格式	基本步数	可用步	
		F/FS	G
(S1) != (S2)	7	○	○

设置数据

n 可用数据

○: 可以设置

设置数据	可用数据								计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数						
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)				
(S1)	—	○	○	○	○	○	○	○	—	—	
(S2)	—	○	○	○	○	○	○	○	—	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	进行比较的数据	逻辑型(真/假)
(S2)		

功能

- 对(S1)中指定的数据与(S2)中指定的数据进行比较，不一致的情况下变为真。
- (S1)与(S2)的数据类型不相同的情况下，转换为类型较大的一方后进行比较。

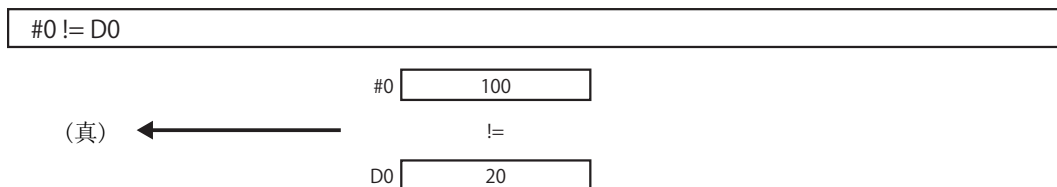
出错

以下情况下将发生运算出错。

- (S1)或(S2)为间接指定软元件，软元件编号超出范围时。

程序示例

n 对#0与D0是否不一致进行比较的程序



小于: <

格式	基本步数	可用步	
		F/FS	G
(S1) < (S2)	7	○	○

设置数据

n 可用数据

○: 可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S1)	—	○	○	○	○	○	○	○	—	—
(S2)	—	○	○	○	○	○	○	○	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	进行比较的数据	逻辑型(真/假)
(S2)		

功能

- (S1) 中指定的数据小于 (S2) 中指定的数据的情况下变为真。
- (S1) 与 (S2) 的数据类型不相同的情况下，转换为类型较大的一方后进行比较。

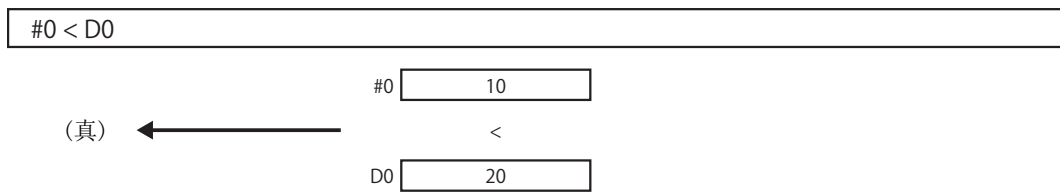
出错

以下情况下将发生运算出错。

- (S1) 或 (S2) 为间接指定软元件，软元件编号超出范围时。

程序示例

n 对#0是否小于D0进行比较的程序



小于等于: <=

格式	基本步数	可用步	
		F/FS	G
(S1) <= (S2)	7	○	○

设置数据

n 可用数据

○: 可以设置

设置数据	可用数据								计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数						
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)				
(S1)	—	○	○	○	○	○	○	○	—	—	
(S2)	—	○	○	○	○	○	○	○	—	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	进行比较的数据	逻辑型(真/假)
(S2)		

功能

- (S1)中指定的数据小于等于(S2)中指定的数据的情况下变为真。
- (S1)与(S2)的数据类型不相同的情况下,转换为类型较大的一方后进行比较。

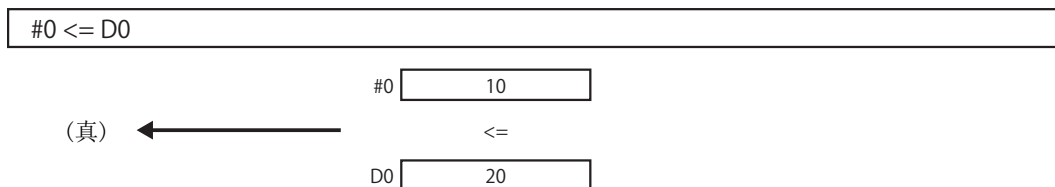
出错

以下情况下将发生运算出错。

- (S1)或(S2)为间接指定软元件,软元件编号超出范围时。

程序示例

n 对#0是否小于等于D0进行比较的程序



大于: >

格式	基本步数	可用步	
		F/FS	G
(S1)>(S2)	7	○	○

设置数据

n 可用数据

○: 可以设置

设置数据	可用数据								计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数						
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)				
(S1)	—	○	○	○	○	○	○	○	—	—	
(S2)	—	○	○	○	○	○	○	○	—	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	进行比较的数据	逻辑型(真/假)
(S2)		

功能

- (S1)中指定的数据大于(S2)中指定的数据的情况下变为真。
- (S1)与(S2)的数据类型不相同的情况下,转换为类型较大的一方后进行比较。

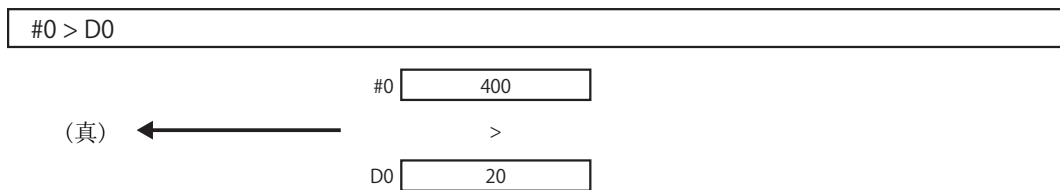
出错

以下情况下将发生运算出错。

- (S1)或(S2)为间接指定软元件,软元件编号超出范围时。

程序示例

n 对#0是否大于D0进行比较的程序



大于等于: >=

格式	基本步数	可用步	
		F/FS	G
(S1) >=(S2)	7	○	○

设置数据

n 可用数据

○: 可以设置

设置数据	可用数据								计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数						
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)				
(S1)	—	○	○	○	○	○	○	○	—	—	
(S2)	—	○	○	○	○	○	○	○	—	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	进行比较的数据	逻辑型(真/假)
(S2)		

功能

- (S1)中指定的数据大于等于(S2)中指定的数据的情况下变为真。
- (S1)与(S2)的数据类型不相同的情况下,转换为类型较大的一方后进行比较。

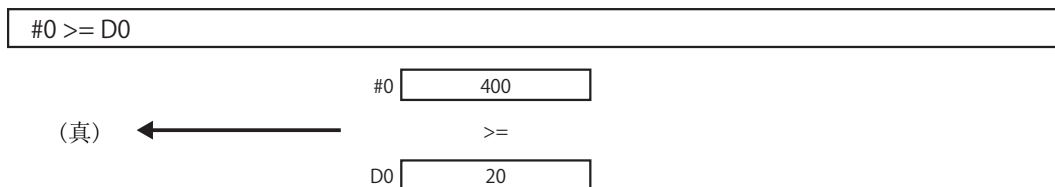
出错

以下情况下将发生运算出错。

- (S1)或(S2)为间接指定软元件,软元件编号超出范围时。

程序示例

n 对#0是否大于等于D0进行比较的程序



4.14 程序控制

条件分支控制：IF~ELSE~IEND

格式	基本步数	可用步	
		F/FS	G
IF(S)~ELSE~IEND	IF : 8 ELSE: 5 IEND: 1	○	○

设置数据

n 可用数据

○：可以设置

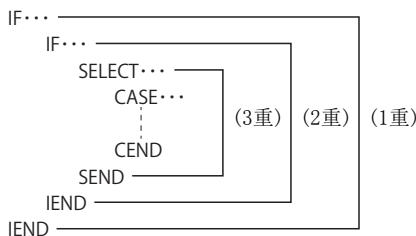
设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	○	—	—	—	—	—	—	—	○	○

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	对程序的流程进行控制的条件数据	—

功能

- (S)中指定的数据为真时，执行从IF至ELSE的块。
- (S)中指定的数据为假时，执行从ELSE至IEND的块。
- 可以省略ELSE。在此情况下，只有在(S)中指定的数据为真时执行从IF至IEND的块。
- 对于条件分支控制的多重度，与选择分支控制(SELECT~CASE~SEND)合计最多可为8重。



出错

以下情况下将发生运算出错，相应运动SFC程序No. 的执行将停止。此外，子程序调用的程序的情况下，调用源程序的执行也将停止。

- (S)为间接指定软元件，软元件编号超出范围时。

程序示例

n #0为K100时，使#100与K10相加，#0为K100以外时使#100与K20相加的程序

```
IF #0 == K100
  #100 = #100 + K10
ELSE
  #100 = #100 + K20
IEND
```

n M0或M1为ON时，通过CHGV指令进行轴2的速度更改的程序

```
IF M0 + M1
  CHGV(K2,K10)
IEND
```

选择分支控制：SELECT~CASE~SEND

格式	基本步数	可用步	
		F/FS	G
SELECT CASE (S1) ~CEND CASE (S2) ~CEND ⋮ CASE (Sn) ~CEND CELSE ~CEND SEND	SELECT: 1 CASE: 8 CEND: 5 CELSE: 1 SEND: 1	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S) ~ (Sn)	○	—	—	—	—	—	—	—	○	○

n 内容、结果的数据类型

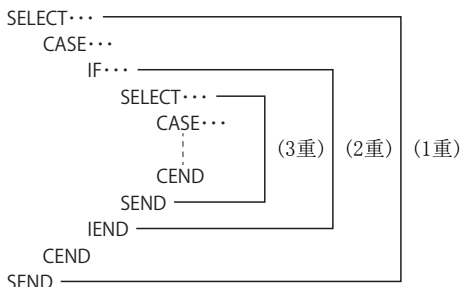
设置数据	内容	结果的数据类型
(S) ~ (Sn)	对程序的流程进行控制的条件数据	—

功能

- 根据 (S1) ~ (Sn) 中指定的数据的真/假，对从CASE至CEND之间记述的块进行选择性执行。
- 对 (S1) ~ (Sn) 的真/假评判按从上往下的顺序进行，最先变为真的从CASE至CEND之间记述的块将被执行。此后，在SEND之前不进行真/假评判，执行SEND的下一个块。
- (S1) ~ (Sn) 中指定的数据全部为假的情况下，执行从CELSE至CEND之间记述的块。
- 可以省略CELSE。此时，从 (S1) 至 (Sn) 中指定的数据全部为假的情况下，不执行从SELECT至SEND为止的块，执行SEND的下一个块。
- SELECT~SEND之间可记述以下个数的CASE (Sn) ~CEND。

CELSE	个数
不使用的情况下	64
使用的情况下	63

- 对于选择分支控制的多重度，与条件分支控制(IF~ELSE~IEND) 合计最多可为8重。



出错

以下情况下将发生运算出错，相应运动SFC程序No. 的执行将停止。此外，子程序调用的程序的情况下，调用源程序的执行也将停止。

- (S)为间接指定软元件，软元件编号超出范围时。

程序示例

n #0为K100时使#100与K10相加，#0为K200以上时使#100与K20相加，除此以外时使#100与K100相加的程序

```
SELECT
  CASE #0 == K100
    #100 = #100 + K10
  CEND
  CASE #0 >= K200
    #100 = #100 + K20
  CEND
  CELSE
    #100 = #100 + K100
  CEND
SEND
```

次数指定重复控制：FOR~NEXT

格式	基本步数	可用步	
		F/FS	G
FOR (D)=(S1) TO (S2) STEP (S3) ~NEXT	FOR: 18 NEXT: 15	○	○

设置数据

n 可用数据

○：可以设置

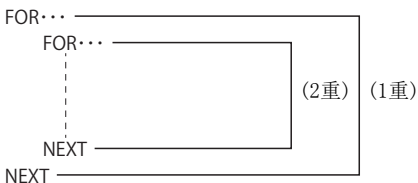
设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(D)	—	○	○	○	—	—	—	—	—	—
(S1)	—	○	○	○	○	○	○	—	—	—
(S2)	—	○	○	○	○	○	○	—	—	—
(S3)	—	—	—	—	○	○	○	—	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(D)	重复控制用计数器中使用的软元件	—
(S1)	重复控制用计数器初始值	
(S2)	重复控制用计数器的最终值	
(S3)	重复控制用计数器的增量值	

功能

- 将(S1)作为初始值代入到(D)中指定的软元件中，重复执行从FOR至NEXT为止的块。
- 每次执行NEXT时将(S3)中指定的增量值加到(D)中指定的软元件中。(D)中指定的软元件的值大于(S2)中指定的最终值时，结束从FOR至NEXT为止的块的重复控制，执行NEXT的下一个块。
- (S3)中指定的增量值为负数的情况下，(D)中指定的软元件的值小于(S2)中指定的最终值时，结束从FOR至NEXT为止的块的重复控制。
- 可以省略STEP。省略了STEP的情况下，作为“STEP 1”进行重复控制。
- 重复控制的多重度最多可为8重。



- (D)、(S1)、(S2)、(S3)的数据类型不相同的情况下，将进行类型转换处理，但有可能变为意外动作。数据类型应统一。

出错

以下情况下将发生运算出错，相应运动SFC程序No. 的执行将停止。此外，子程序调用的程序的情况下，调用源程序的执行也将停止。

- (S1)的数据超出(D)的数据类型的范围时。
- (D)、(S1)、(S2)为间接指定软元件，软元件编号超出范围时。
- 1个运算控制程序或转换程序中，FOR~NEXT指令的执行超出参数中设置的重复控制限制次数时。

程序示例

n #0为1~10之间(增量值1)，对将软元件编号通过“#0+100”进行了间接指定的运动寄存器(#)重复进行#0的数据的代入处理的程序

程序结束时，1~10将被代入到#101~#110中。

```
FOR #0 = K1 TO K10
  #(#0 + K100) = #0
NEXT
```

增量值为正数的情况下，FOR~NEXT的重复处理结束后，(D)中指定的软元件的值将变为大于(S2)中指定的最终值。在上述示例中，(D)中设置的#0将变为11。

n #0为100~10之间(增量值-10)，重复进行从#100中减去#0的处理的程序

```
FOR #0 = K100 TO K10 STEP K-10
  #100 = #100 - #0
NEXT
```

增量值为负数的情况下，FOR~NEXT的重复处理结束后，(D)中指定的软元件的值将变为小于(S2)中指定的最终值。在上述示例中，(D)中设置的#0将变为0。

要点

在(D)中指定的重复控制用计数器中，在超过最终值之前将被进行增量值的加法运算，因此应选择可处理该值的数据类型。重复控制用计数器超出可处理的数据范围的情况下，值将变为非法且被进行非意图的重复。在以下程序中，重复控制用计数器#0的数据类型为16位整数型，该数据范围为-32768~32767。

```
FOR #0 = K0 TO K30000 STEP K10000
  #1 = #1 + K1
NEXT
```

执行该程序时#0将发生如下所示的变化，途中超过16位整数型的数据范围，因此不按重复次数4次结束。

- 第1次重复：#0为0
- 第2次重复：#0为10000
- 第3次重复：#0为20000
- 第4次重复：#0为30000
- 第5次重复：#0为0 #0变为-25536(#0变为40000，但超出数据范围而发生溢出。)
- 第6次重复：#0为-15536
- ∴

重复控制的强制结束：BREAK

格式	基本步数	可用步	
		F/FS	G
BREAK	5	○	○

设置数据

没有设置数据。

功能

- 强制结束次数指定重复控制 (FOR~NEXT指令)，从NEXT的下一个块开始执行程序。
- BREAK只能记述在FOR~NEXT的重复控制处理块内。

出错

没有运算出错。

程序示例

n M0或M1变为ON时，强制结束通过FOR~NEXT进行的重复控制处理

```
FOR #0 = K1 TO K10
  #100 = #100 + K10
  IF M0 + M1
    BREAK
  IEND
NEXT
```

4.15 运动专用函数

速度更改请求：CHGV

格式	基本步数	可用步	
		F/FS	G
CHGV((S1), (S2))	7	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据								计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数						
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)				
(S1)	—	○	—	—	○	—	—	—	—	—	
(S2)	—	○	○	—	○	○	—	○	—	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	进行速度更改请求的轴No.	—
(S2)	指定速度	

功能

• 进行如下所示的速度更改。

1. (S1)中指定的轴对应的“[St. 1047]速度更改受理中标志(R: M30144+n/Q: M2061+n)”将ON。
2. 将(S1)中指定的轴的速度更改为(S2)中指定的速度。
3. 将速度更改受理中标志置为OFF。

• (S1)中可设置的轴编号的范围如下所示。

运动CPU	设置范围
R64MTCPU	1~64
R32MTCPU	1~32
R16MTCPU	1~16

• 插补控制时，在(S1)中设置插补轴的某个轴。进行直线插补控制的情况下，按照伺服程序中设置的定位速度的指定方法，进行如下所示的速度更改。

定位速度的指定方法	动作
合成速度指定	进行速度更改使合成速度变为(S2)中指定的速度。
长轴基准	进行速度更改使长轴速度变为(S2)中指定的速度。
基准轴速度指定	进行速度更改使基准轴速度变为(S2)中指定的速度。

• 根据(S2)中设置的指定速度的符号，按以下方式执行动作。

指定速度的符号	动作
正	速度的更改
0	暂时停止
负	逆转

• (S2)中可设置的指定速度的范围如下所示。

请求动作	设置范围			
	mm	inch	degree	pulse
速度更改请求	0~600000000 ($\times 10^{-2}$ [mm/min])	0~600000000 ($\times 10^{-3}$ [inch/min])	0~2147483647 ($\times 10^{-3}$ [degree/min])*1	0~2147483647 [pulse/s]
逆转请求	-1~-600000000 ($\times 10^{-2}$ [mm/min])	-1~-600000000 ($\times 10^{-3}$ [inch/min])	-1~-2147483647 ($\times 10^{-3}$ [degree/min])*1	-1~-2147483647 [pulse/s]

*1 通过固定参数，使“degree轴速度10倍指定”生效的情况下，变为($\times 10^{-2}$ [degree/min])。

- 通过CHGV指令更改的速度仅在启动中的伺服程序上才有效。
- (S1)中指定的轴的减速停止中时，不进行速度更改。
- (S1)中指定的轴的速度·转矩控制中时，不进行速度更改。
- (S1)中指定的轴为机器控制中(机器JOG运行、机器程序运行)时，不进行速度更改。
- 通过设置(S1)中指定的轴的加减速时间更改参数，可以更改速度更改时的加减速时间。关于加减速时间更改参数及加减速时间更改功能，请参阅以下手册。
- 启动中通过指定负的速度，执行速度更改请求，可以通过该时刻开始减速，通过减速完成返回至反方向。根据伺服指令，其动作如下所示。

控制模式	伺服指令	动作
直线控制	ABS-1 ABS-2 ABS-3 ABS-4 INC-1 INC-2 INC-3 INC-4	通过减速完成反转移动方向，以指定速度的绝对值返回至定位开始点后，停止(待机)。圆弧插补时，返回至圆弧的轨迹上。
圆弧插补控制	ABS圆弧 INC圆弧	
定距进给	FEED-1 FEED-2 FEED-3	
连续轨迹控制	CPSTART1 CPSTART2 CPSTART3 CPSTART4	通过减速完成反转移动方向，以指定速度的绝对值返回至前一个点后，停止(待机)。
速度控制(I)	VF VR	通过减速完成以指定速度的绝对值反转移动方向。 在停止指令被输入之前不停止。
速度控制(II)	VVF VVR	
速度·位置切换控制	VPF VPR VPSTART	不能逆转。 视为通常的速度更改请求。 发出警告(出错代码: 0991H)，通过速度限制值控制。
位置跟踪控制	PFSTART	
固定位置停止速度控制	PVF PVR	
JOG运行		
高速振荡	OSC	不能更改速度。发出警告(出错代码: 09EEH)。
原点复位	ZERO	不能更改速度。发出警告(出错代码: 09EDH)。

n 控制内容

- 进行至负速度的速度更改时，根据启动中的控制模式，进行上表中所示的控制。
- 返回时的指令速度将变为更改速度的绝对值。
- 在返回位置进行待机时，其情况如下所示。
 - 各信号状态

信号名称	状态
[St. 1040] 启动受理标志 (R: M30080+n/Q: M2001+n)	ON (与执行CHGV前无变化)
[St. 1060] 定位启动完成 (R: M32400+32n/Q: M2400+20n)	ON (与执行CHGV前无变化)
[St. 1061] 定位完成 (R: M32401+32n/Q: M2401+20n)	OFF
[St. 1062] 就位 (R: M32402+32n/Q: M2402+20n)	ON
[St. 1063] 指令就位 (R: M32403+32n/Q: M2403+20n)	OFF
[St. 1049] 速度更改“0”受理中标志 (R: M30272+n/Q: M2240+n)	ON

- 再次启动的情况下，应进行至正速度的速度更改。
- 结束定位的情况下，应将停止指令置为ON。
- 再次进行了负的速度更改的情况下，将被忽略。
- 速度控制模式下的逆转中，其情况如下所示。
 - 再次恢复移动方向的情况下，应进行至正速度的速度更改。
 - 停止的情况下，应将停止指令置为ON。
 - 再次进行了负速度更改的情况下，以逆转方向进行速度更改。
- 对于将行程限位设置为无效的轴，不进行至负速度的速度更改。

出错

以下情况下将发生运算出错，不进行速度更改。

- (S1) 的指定轴编号超出范围时。
- (S2) 为间接指定软元件，软元件编号超出范围时。

以下情况下将发出警告，不进行速度更改。

- (S1) 中指定的轴进行了原点复位时。(警告(出错代码: 09EDH))
- 对将行程限位设置为无效的轴进行了至负速度的速度更改时。(警告(出错代码: 09EFH))

要点

(S1) 中指定的轴处于减速中时即使进行速度更改，速度更改也将被忽略。此时不发生出错。

以下情况下将发出警告，通过速度限制值进行控制。

- (S2) 中指定的速度的绝对值大于速度限制值时。(警告(出错代码: 0991H))

要点

连续轨迹控制中，负的更改速度的绝对值超出伺服程序中指定的速度的情况下，将以程序中指定的速度进行逆转控制(连续轨迹控制状态下的速度更改中的速度钳制控制)。此时不发生出错。

程序示例

n 对轴2的定位速度进行更改的程序

```
CHGV(K2,K10)
```

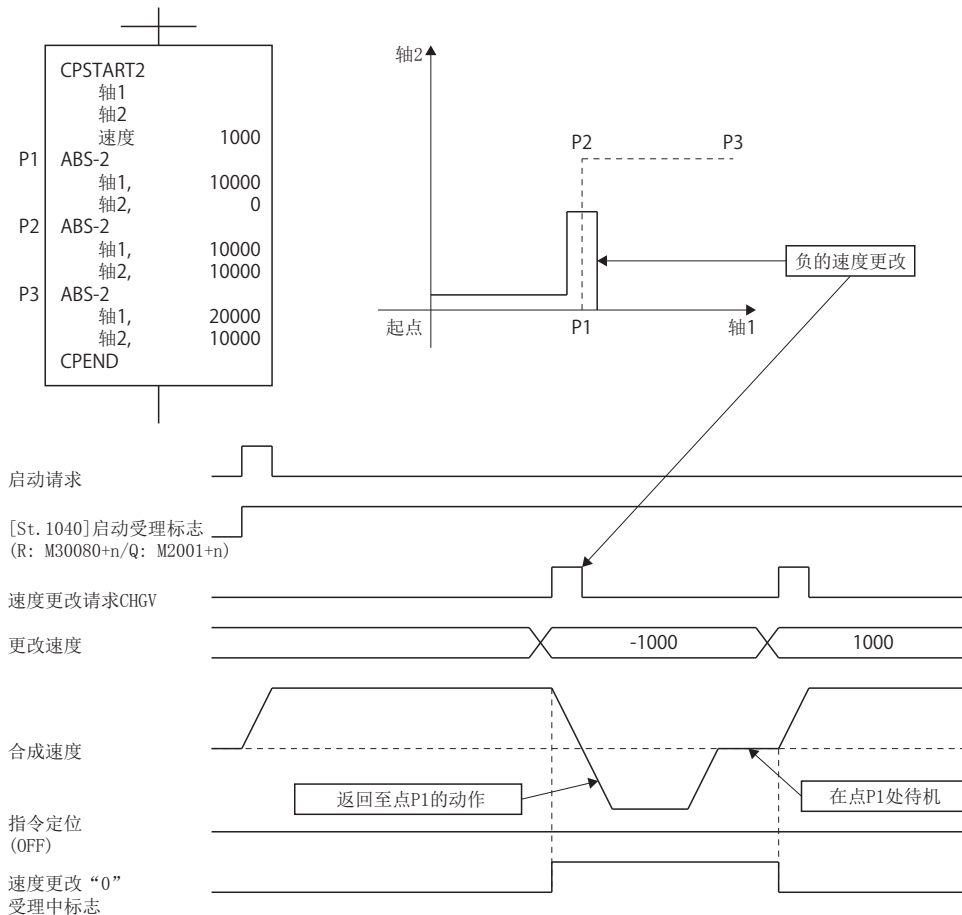
n 将轴1的定位速度更改为负值的逆转程序

CHGV(K1,K-1000)

连续轨迹控制中，进行了逆转请求时的动作如下所示。

[伺服程序]

[轨迹]

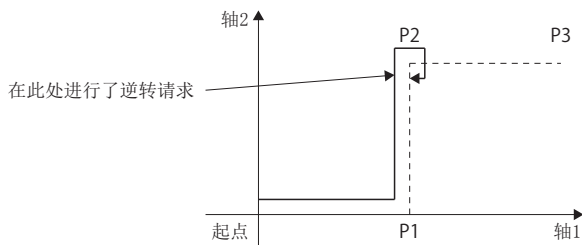


如上图所示，至P2的定位执行中，如果进行了至负速度的速度更改，将沿着程序中指定的轨迹返回至P1，在P1处进行待机。

要点

[速度更改时的注意事项]

- 伺服程序的启动请求时，“定位启动完成信号”状态变为ON之前执行了速度更改的情况下，速度更改有可能变为无效。与启动几乎相同时机进行速度更改的情况下，必须将程序创建为在“定位启动完成信号”变为ON后，再执行速度更改。
- 连续轨迹控制中使用M代码FIN信号等待功能，在FIN等待的停止中进行了逆转请求的情况下，将被忽略。
- 在上述示例中，在P2之前进行了逆转请求，在减速中通过了P2的情况下，将返回至P2。
- 执行CHGV指令之后，实际速度开始变化之前的响应时间最大将产生相当于运算周期的延迟。



指令生成轴速度更改请求：CHGVS

格式	基本步数	可用步	
		F/FS	G
CHGVS ((S1), (S2))	7	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S1)	—	○	—	—	○	—	—	—	—	—
(S2)	—	○	○	—	○	○	—	○	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	进行速度更改请求的轴No.	—
(S2)	指定速度	

功能

• 进行如下所示的速度更改。

1. (S1)中指定的轴对应的“[St. 346]指令生成轴速度更改受理中标志(R: M36571+32n/Q: M9811+20n)”变为ON。
2. 将(S1)中指定的轴的速度更改为(S2)中指定的速度。
3. 将速度更改受理中标志置为OFF。

• (S1)中可设置的轴编号的范围如下所示。

运动CPU	设置范围
R64MTCPU	1~64
R32MTCPU	1~32
R16MTCPU	1~16

• 插补控制时，在(S1)中设置插补轴的某个轴。进行直线插补控制的情况下，按照伺服程序中设置的定位速度的指定方法，进行如下所示的速度更改。

定位速度的指定方法	动作
合成速度指定	进行速度更改使合成速度变为(S2)中指定的速度。
长轴基准	进行速度更改使长轴速度变为(S2)中指定的速度。
基准轴速度指定	进行速度更改使基准轴速度变为(S2)中指定的速度。

• 根据(S2)中设置的指定速度的符号，按以下方式执行动作。

指定速度的符号	动作
正	速度的更改
0	暂时停止
负	逆转

- (S2)中可设置的指定速度的范围如下所示。

请求动作	设置范围			
	mm	inch	degree	pulse
速度更改请求	0~600000000 ($\times 10^{-2}$ [mm/min])	0~600000000 ($\times 10^{-3}$ [inch/min])	0~2147483647 ($\times 10^{-3}$ [degree/min])*1	0~2147483647 [pulse/s]
逆转请求	-1~-600000000 ($\times 10^{-2}$ [mm/min])	-1~-600000000 ($\times 10^{-3}$ [inch/min])	-1~-2147483647 ($\times 10^{-3}$ [degree/min])*1	-1~-2147483647 [pulse/s]

*1 通过指令生成轴参数，使“degree轴速度10倍指定”生效的情况下，变为($\times 10^{-2}$ [degree/min])。

- 通过CHGVS指令更改的速度仅在启动中的伺服程序上才有效。
- (S1)中指定的轴的减速停止中时，不进行速度更改。
- 通过设置(S1)中指定的轴的加减速时间更改参数，可以更改速度更改时的加减速时间。关于加减速时间更改参数，请参阅以下手册。

📖MELSEC iQ-R运动控制器编程手册(高级同步控制篇)

关于加减速时间更改功能，请参阅以下手册。

📖MELSEC iQ-R运动控制器编程手册(定位控制篇)

- 启动中通过指定负的速度，执行速度更改请求，可以通过该时刻开始减速，通过减速完成返回至反方向。根据伺服指令，其动作如下所示。

控制模式	伺服指令	动作
直线控制	ABS-1 ABS-2 ABS-3 ABS-4 INC-1 INC-2 INC-3 INC-4	通过减速完成反转移动方向，以指定速度的绝对值返回至定位开始点后，停止(待机)。圆弧插补时，返回至圆弧的轨迹上。
圆弧插补控制	ABS圆弧 INC圆弧	
定距进给	FEED-1 FEED-2 FEED-3	
连续轨迹控制	CPSTART1 CPSTART2 CPSTART3 CPSTART4	通过减速完成反转移动方向，以指定速度的绝对值返回至前一个点后，停止(待机)。
速度控制(I)	VF VR	通过减速完成以指定速度的绝对值反转移动方向。 在停止指令被输入之前不停止。
位置跟踪控制	PFSTART	不能逆转。 视为通常的速度更改请求。
固定位置停止速度控制	PVF PVR	发出警告(出错代码: 0991H)，通过速度限制值控制。
JOG运行		

n 控制内容

- 进行至负速度的速度更改时，根据启动中的控制模式，进行上表中所示的控制。
- 返回时的指令速度将变为更改速度的绝对值。
- 在返回位置进行待机时，其情况如下所示。
 - 各信号状态

信号名称	状态
[St. 345]指令生成轴启动受理标志(R: M36570+32n/Q: M9810+20n)	ON(与执行CHGVS前无变化)
[St. 340]指令生成轴定位启动完成(R: M36560+32n/Q: M9800+20n)	ON(与执行CHGVS前无变化)
[St. 341]指令生成轴定位完成(R: M36561+32n/Q: M9801+20n)	OFF
[St. 342]指令生成轴指令就位(R: M36563+32n/Q: M9803+20n)	OFF
[St. 347]指令生成轴速度更改“0”受理中标志(R: M36572+32n/Q: M9812+20n)	ON

- 再次启动的情况下，应进行至正速度的速度更改。
- 结束定位的情况下，应将停止指令置为ON。
- 再次进行了负的速度更改的情况下，将被忽略。
- 速度控制模式下的逆转中，其情况如下所示。
 - 再次恢复移动方向的情况下，应进行至正速度的速度更改。
 - 停止的情况下，应将停止指令置为ON。
 - 再次进行了负速度更改的情况下，以逆转方向进行速度更改。
- 对于将行程限位置为无效的轴，不进行至负速度的速度更改。

出错

以下情况下将发生运算出错，不进行速度更改。

- (S1)的指定轴编号超出范围时。
- (S2)为间接指定软元件，软元件编号超出范围时。

以下情况下将发出警告，不进行速度更改。

- 对将行程限位设置为无效的轴进行了至负速度的速度更改时。（警告(出错代码：09EFH)）

要点

(S1)中指定的轴处于减速中时即使进行速度更改，速度更改也将被忽略。此时不发生出错。

以下情况下将发出警告，通过速度限制值进行控制。

- (S2)中指定的速度的绝对值大于速度限制值时。（警告(出错代码：0991H)）

要点

连续轨迹控制中，负的更改速度的绝对值超出伺服程序中指定的速度的情况下，将以程序中指定的速度进行逆转控制(连续轨迹控制状态下的速度更改中的速度钳制控制)。此时不发生出错。

程序示例

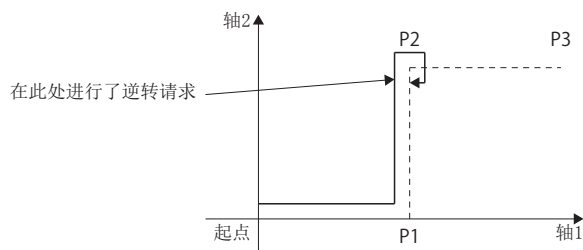
n 对轴2的定位速度进行更改的程序

```
CHGVS(K2,K10)
```

要点

[速度更改时的注意事项]

- 伺服程序的启动请求时，“定位启动完成信号”状态变为ON之前执行了速度更改的情况下，速度更改有可能变为无效。与启动几乎相同时机进行速度更改的情况下，必须将程序创建为在“定位启动完成信号”变为ON后，再执行速度更改。
- 连续轨迹控制中使用M代码FIN信号等待功能，在FIN等待的停止中进行了逆转请求的情况下，将被忽略。
- 在上述示例中，在P2之前进行了逆转请求，在减速中通过了P2的情况下，将返回至P2。
- 执行CHGVS指令之后，至实际速度开始变化之前的响应时间最大将产生相当于运算周期的延迟。



转矩限制值更改请求：CHGT

格式	基本步数	可用步	
		F/FS	G
CHGT ((S1), (S2), (S3))	10	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据								计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数						
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)				
(S1)	—	○	—	—	○	—	—	—	—	—	
(S2)	—	○	○	—	○	○	—	○	—	—	
(S3)	—	○	○	—	○	○	—	○	—	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	进行转矩限制值更改请求的轴No.	—
(S2)	正方向转矩限制值(×0.1[%])	
(S3)	负方向转矩限制值(×0.1[%])	

功能

- 将(S1)中指定的轴的转矩限制值，更改为(S2)中指定的正方向转矩限制值及(S3)中指定的负方向转矩限制值。正方向转矩限制值对伺服电机的正转(CCW)力行·反转(CW)回生转矩进行限制，负方向转矩限制值对伺服电机的反转(CW)力行·正转(CCW)回生转矩进行限制。
- 如果是伺服启动完成的轴，则与启动中·停止中·伺服ON中·伺服OFF中的状态无关，随时进行转矩限制值的更改。
- (S1)中可设置的轴编号的范围如下所示。

运动CPU	设置范围
R64MTCPU	1~64
R32MTCPU	1~32
R16MTCPU	1~16

- (S2)及(S3)不能省略。仅更改一方的转矩限制值的情况下，应在无需更改的设置数据中设置-1。
- (S2)及(S3)中可设置的转矩限制值范围为1~10000(×0.1[%])。
- 速度·转矩控制时，不能更改为大于伺服数据设置的速度·转矩控制数据中设置的速度·转矩控制时转矩限制值的转矩限制值。通过CHGT指令指定的(S2)及(S3)的值之一大于速度·转矩控制时转矩限制值的情况下将发出警告(出错代码：0A5EH)，不进行转矩限制值的个别更改。
- 在伺服数据设置的扩展参数中，通过设置正方向转矩限制值监视软元件及负方向转矩限制值监视软元件，可以对正方向转矩限制值及负方向转矩限制值进行监视。

出错

以下情况下将发生运算出错，不进行转矩限制值的更改。

- (S1)的指定轴编号超出范围时。
- (S2)或(S3)为间接指定软元件，软元件编号超出范围时。

以下情况下将发出警告，不进行转矩限制值的更改。

- (S2)或(S3)中指定的转矩限制值超出0.1~1000.0[%]的范围时。(警告(出错代码: 0A5BH))
- 对未启动轴执行了CHGT指令时。(警告(出错代码: 0A5CH))
- 对速度·转矩控制中的轴执行了CHGT指令的情况下，(S2)或(S3)的值大于速度·转矩控制时。(警告(出错代码: 0A5EH))

程序示例

n 将轴2的转矩限制值分别更改为正方向20.0[%]、负方向10.0[%]的程序

```
CHGT(K2,K200,K100)
```

要点

对于执行CHGT指令之后开始至实际转矩限制值传送至伺服放大器为止的时间，最大将产生相当于运算周期的延迟。

目标位置更改请求：CHGP

格式	基本步数	可用步	
		F/FS	G
CHGP ((S1), (S2), (S3))	11	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S1)	—	○	—	—	○	—	—	—	—	—
(S2)	—	○	—	—	○	—	—	—	—	—
(S3)	—	○	—	—	—	—	—	—	—	—

n 内容、结果的数据类型

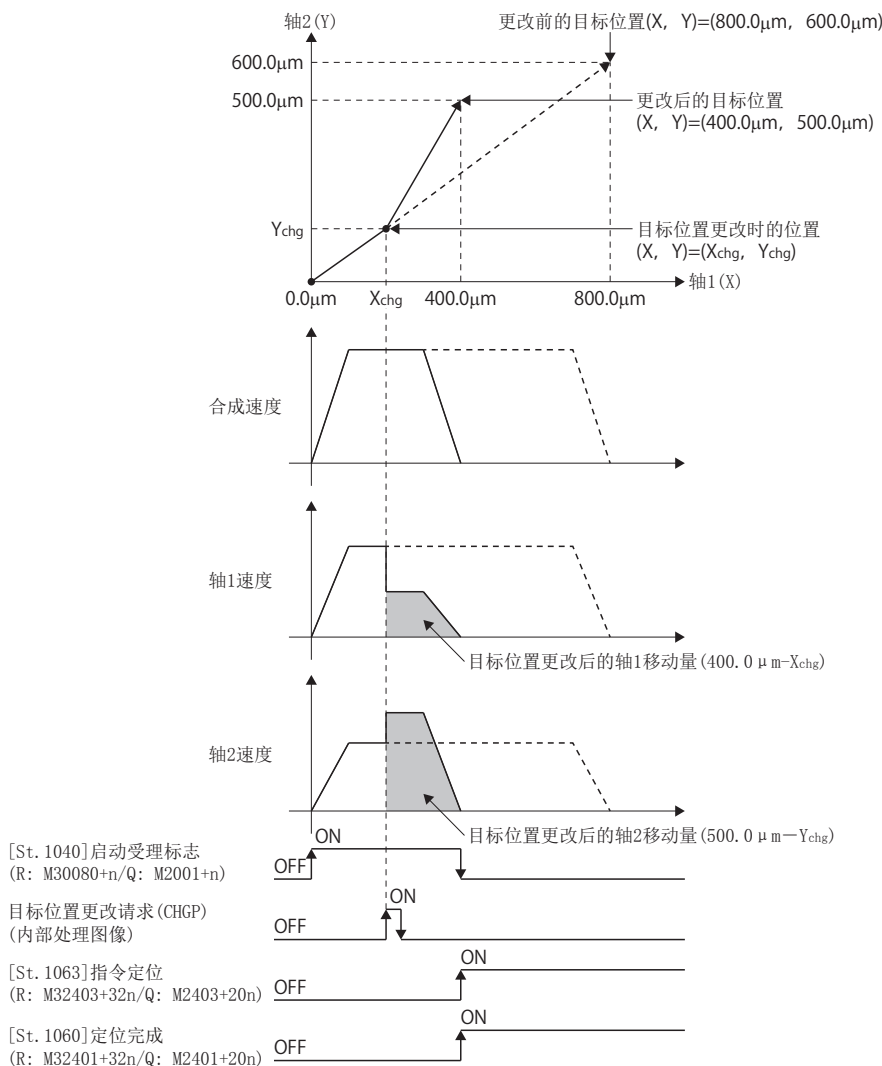
设置数据	内容	结果的数据类型
(S1)	进行目标位置更改请求的轴No.	—
(S2)	更改地址指定方式 0: 地址指定 1: 移动量指定	
(S3)	存储目标位置更改值的软元件的起始编号	

要点

CHGP指令不能用于高级同步控制的指令生成轴。

功能

- 目标位置更改请求时在定位指令执行中进行目标位置的更改。对于新的目标位置，可以通过从执行绝对地址或目标位置更改请求时的进给当前值的相对移动量进行指定。从定位开始位置 $(X, Y)=(0.0\mu\text{m}, 0.0\mu\text{m})$ 至 $(X, Y)=(800.0\mu\text{m}, 600.0\mu\text{m})$ 的直线插补控制中通过绝对地址指定执行了至 $(X, Y)=(400.0\mu\text{m}, 500.0\mu\text{m})$ 的目标位置更改请求情况下的动作如下所示。



- 进行(S1)中指定的轴的目标位置更改。根据(S2)中指定的方式，通过(S3)中指定的软件中存储的值计算出更改后的目标位置。

要点

- CHGP指令仅对启动中的轴有效。
- 指定的轴处于减速停止中时，不进行目标位置更改。
- 执行CHGP指令之后至实际目标位置被更改为止，最大将产生相当于运算周期的延迟。
- 伺服程序的启动请求时（“[St. 1060]定位启动完成(R: M32400+32n/Q: M2400+20n)”为OFF时）如果执行CHGP指令，目标位置更改将变为无效。与伺服程序的启动几乎相同的时机进行目标位置更改的情况下，应将程序创建为“定位启动完成信号”变为ON后再执行目标位置更改。

- (S1)中可设置的轴编号的范围如下所示。插补控制时，在(S1)中设置插补轴的某个轴。

运动CPU	设置范围
R64MTCPU	1~64
R32MTCPU	1~32
R16MTCPU	1~16

- 根据(S2)的设置，目标位置如下所示。
 - 将(S2)设置为0(地址指定方式)时，将(S3)中指定的软件元件中存储的目标位置更改值作为目标位置。
 - 将(S2)设置为1(移动量指定方式)时，将从执行CHGP指令时的进给当前值开始按(S3)中指定的软件元件中存储的目标位置更改值进行了移动后的位置作为目标位置。

要点

将(S2)设置为1(移动量指定方式)，在普通任务中执行CHGP指令时，由于指令受理时机的偏差，更改后的目标位置有可能产生偏差。通过在与运算周期相同的固恒定周期任务中执行，可以抑制偏差。

- 在(S3)中指定存储目标位置更改值的起始软件元件。应将起始软件元件设置为偶数编号，按以下方式设置目标位置更改值。

偏置	名称	设置范围				
		mm	inch	pulse	degree	
					地址指定	移动量指定
+0	目标位置更改值1	-2147483648~ 2147483647 ($\times 10^{-1}$ [μm])	-2147483648~ 2147483647 ($\times 10^{-5}$ [inch])	-2147483648~ 2147483647 ([pulse])	0~35999999 ($\times 10^{-9}$ [degree])	-2147483648~ 2147483647 ($\times 10^{-5}$ [degree])
+1						
+2	目标位置更改值2					
+3						
+4	目标位置更改值3					
+5	目标位置更改值4					
+6						
+7						

- 对于目标位置更改值，应根据(S2)的设置对定位地址或移动量进行设置。
- 应以插补轴内轴编号的升序设置目标位置更改值。

例

INC-3指令执行中进行目标位置更改请求的情况下

[K100]	INC-3		
	轴	3,	3000pulse
	轴	4,	4000pulse
	轴	1,	4000pulse
	速度		10000pulse/s

目标位置更改值1~4对应的轴No. 如下所示。

- 目标位置更改值1: 轴No. 1的设置
- 目标位置更改值2: 轴No. 3的设置
- 目标位置更改值3: 轴No. 4的设置
- 目标位置更改值4: 无需设置

• 执行CHGP指令时，根据执行中的伺服指令，其动作如下所示。

控制模式	伺服指令	动作
直线控制	ABS-1 ABS-2 ABS-3 ABS-4 INC-1 INC-2 INC-3 INC-4	在CHGP指令的执行中，通过直线插补控制进行从执行时的进给当前值至更改后的目标位置的定位。
定距进给	FEED-1 FEED-2 FEED-3	
圆弧插补控制	ABS圆弧 INC圆弧	目标位置更改将被忽略并发出警告(出错代码：099BH)。
螺旋插补控制	ABS螺旋 INC螺旋	
连续轨迹控制	CPSTART1 CPSTART2 CPSTART3 CPSTART4	在CHGP指令的执行中，通过直线插补控制进行从执行时的进给当前值至更改后的目标位置的定位。对剩余的点不执行定位。
速度控制(I)	VF VR	目标位置更改将被忽略并发出警告(出错代码：099BH)。
速度控制(II)	VVF VVR	
速度·位置切换控制	VPF VPR VPSTART	
位置跟踪控制	PFSTART	
固定位置停止速度控制	PVF PVR	
JOG运行		
速度·转矩控制		
压力控制		
高速振荡	OSC	
原点复位	ZERO	

• 执行CHGP指令后的动作如下所示。

- 对于“[St. 1048]自动减速中标志(R: M30208+n/Q: M2128+n)”，进行了至更改后的目标位置的自动减速时将变为0N。
- 对于“[St. 1063]指令就位(R: M32403+32n/Q: M2403+20n)”，更改后的目标位置与进给当前值的差的绝对值为“指令就位范围”以下时将变为0N。
- 对于“[St. 1061]定位完成(R: M32401+32n/Q: M2401+20n)”，通过至更改后的目标位置的指令输出完成将变为0N。

• 执行CHGP指令后，合成速度保持原样而各轴的速度根据更改后的目标位置而变化。因此，根据更改后的目标位置，各轴的速度有可能急剧变化，应加以注意。

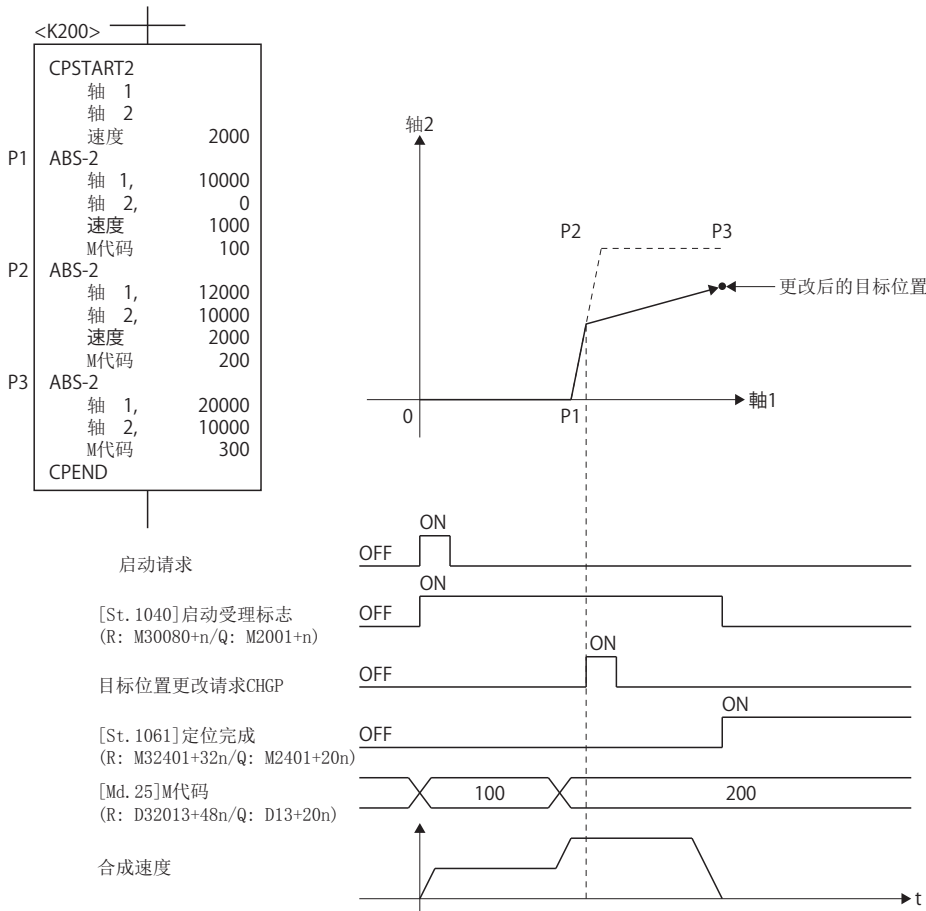
• 在直线插补控制中进行基准轴速度指定或长轴基准指定的情况下，其动作如下所示。

- 目标位置更改时不重新选定长轴。沿用目标位置更改前的长轴。
- 根据目标位置更改后的各轴的移动量定位速度将被重新计算。
- 由于目标位置更改基准轴或长轴的移动量变为0的情况下，将发生轻度出错(出错代码：1A5FH)而减速停止。

- 连续轨迹控制 (CPSTART) 中执行 CHGP 指令时, 将进行至更改后的目标位置的定位。目标位置更改请求时不对执行中的点以后的点执行定位。

[伺服程序]

[轨迹]

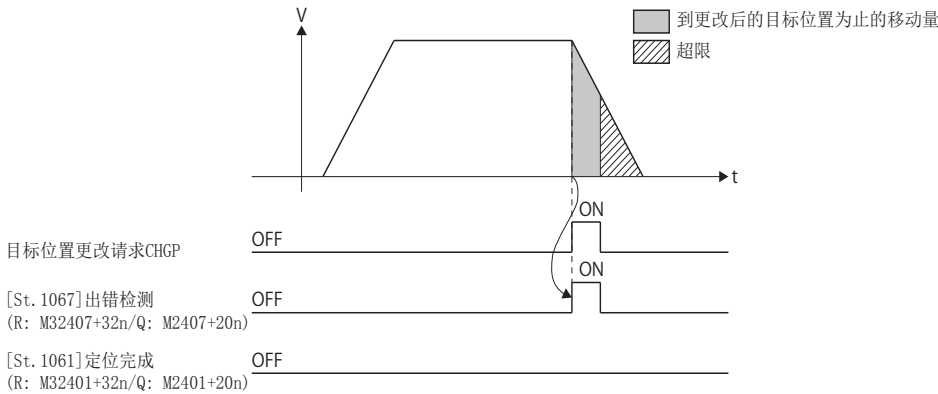


要点

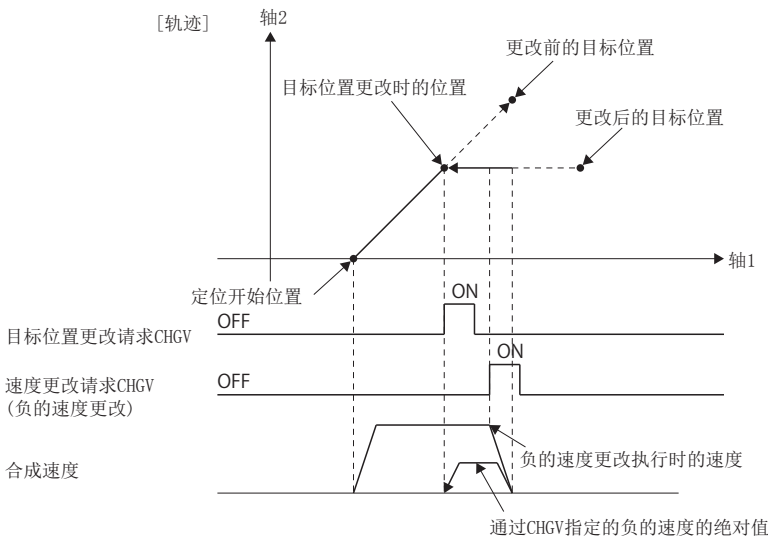
- 执行 CHGP 指令时, 沿用定位中点的设置项目进行定位。
- 执行 CHGP 指令时, 对 CPSTART 中指定的所有轴进行直线插补控制, 因此应对 CPSTART 中指定的所有轴的目标位置进行设置。
- 通过连续轨迹控制至圆弧插补、螺旋插补点的定位中执行了 CHGP 指令的情况下, 至圆弧插补、螺旋插补点的定位结束后, 与至直线插补点的定位开始的同时进行目标位置更改。

- 对控制单位为 [degree] 的轴执行了地址指定方式的目标位置更改请求的情况下, 其动作如下所示。
 - 保持当前的移动方向向更改后的地址进行定位。
 - 使用地址指定方式的情况下, 更改地址应设置为 $0 \sim 35999999 \times 10^{-5}$ [degree]。设置超出范围时, 将发生轻度出错 (出错代码: 1A5CH) 而减速停止。

- 由于CHGP指令的执行，至更改后的目标位置为止的移动量小于从控制中的速度至减速停止所需减速距离的情况下，其动作如下所示。
 - 在执行CHGP指令的时刻将发生轻度出错(出错代码：1A5DH)而减速停止。
 - 至减速停止为止的移动量与更改后的目标位置为止的移动量的差将超限。
 - “[St. 1061]定位完成(R: M32401+32n/Q: M2401+20n)”不变为ON。



- 执行CHGP指令后如果执行负速度更改，将减速至速度0为止，通过减速完成返回至直线插补中目标位置更改时(CHGP指令受理时)的位置为止后，停止(待机)。



出错

以下情况下将发生运算出错，不进行目标位置更改。

- (S1)的指定轴编号超出范围时。
- (S2)设置为0~1以外时。
- (S3)不是偶数编号的软件元件时。
- (S3)~(S3)+7的软件元件编号超出范围时。

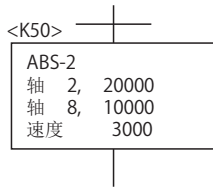
以下情况下将发出警告或发生轻度出错，不进行目标位置更改。

- 相应轴处于原点复位中时。(警告(出错代码: 099BH))
- 相应轴正在执行不支持目标位置更改的伺服程序时。(警告(出错代码: 099BH))
- 更改后的目标位置超出行程限位范围时。(轻度出错(出错代码: 1A5EH))
- 加减速方式设置为FIN加减速或高级S字加减速时。(轻度出错(出错代码: 19E6H))
- 直线插补控制中基准轴速度指定及长轴基准指定的情况下，由于目标位置更改基准轴或长轴的移动量变为0时。(轻度出错(出错代码: 1A5FH))
- 对控制单位为[degree]的轴执行了地址指定方式的目标位置更改请求的情况下，更改地址超出 $0 \sim 35999999 \times 10^{-5}$ [degree]的范围时。(轻度出错(出错代码: 1A5CH))
- 至目标位置更改后的目标位置为止的移动量小于从控制中的速度至减速停止所需的移动量时。(轻度出错(出错代码: 1A5DH))

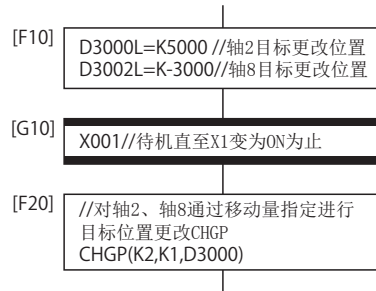
程序示例

n 对通过ABS-2定位中的轴2、轴8，通过移动量指定进行目标位置更改情况下的程序

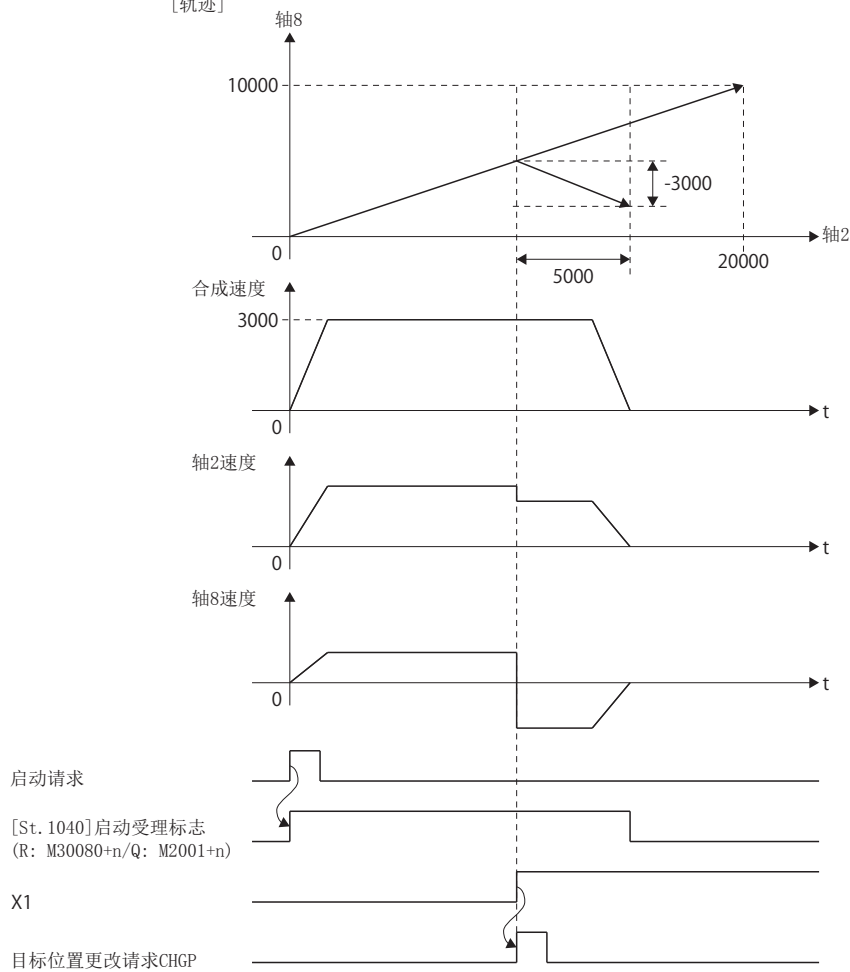
[伺服程序]



[运动SFC程序]



[轨迹]



机器程序运行启动请求：MCNST

格式	基本步数	可用步	
		F/FS	G
MCNST((S1), (S2))	7	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S1)	—	○	—	—	—	—	—	—	—	
(S2)	—	○	—	—	○	—	—	○	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	存储了机器定位数据的软元件的起始编号	—
(S2)	机器定位数据区域的容量(字)(78~5444)	

功能

- 从(S1)中指定的软元件的起始编号开始，将(S2)中指定容量的数据作为机器定位数据，进行机器程序运行启动请求。
- 关于(S1)中指定的软元件中存储的机器定位数据，请参阅以下手册。
 MELSEC iQ-R运动控制器编程手册(机器控制篇)
- (S2)中可设置的机器定位数据区域的范围为78~5444。

出错

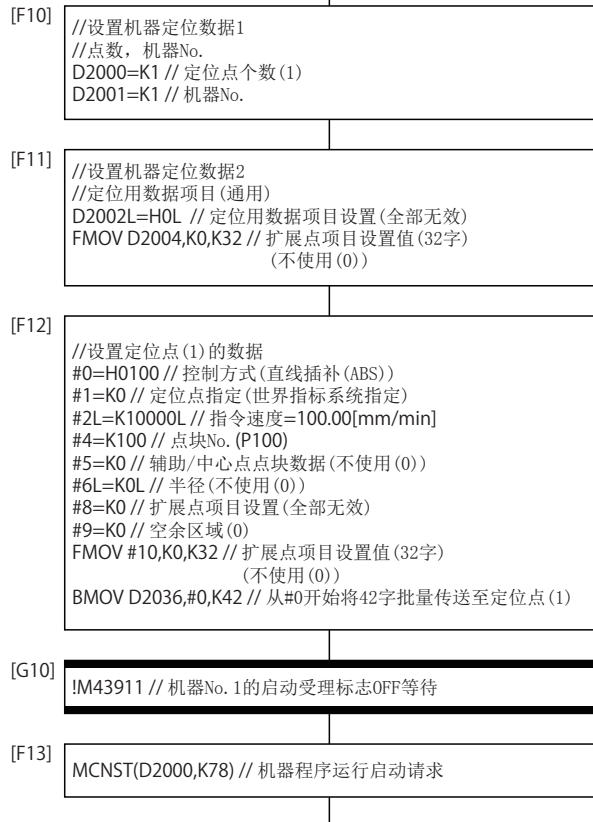
以下情况下将变为运算出错，不执行机器程序运行启动请求。

- (S1)为偶数编号的软元件以外时。(详细代码：1)
- (S1)+(S2)的软元件编号超出范围时。(详细代码：2)
- MCNST指令的指令区域不足时。(运动专用函数(MCNST)中动作中的指令超过了9个指令)(详细代码：3)
- 机器定位数据区域设置不正确时。(详细代码：4)
- (S2)中设置的数据区域点数范围超出78~5444时。(详细代码：4)
- 数据区域点数小于通过点数计算出的区域时。(详细代码：4)
- 定位点数超出1~128的范围时。(详细代码：4)
- 机器定位数据中指定的控制方式不正确时。(也包括各点的控制方式仅为NOP的情况下)(详细代码：5)
- 机器定位数据中指定的机器No.超出范围时。(详细代码：6)

程序示例

n 将由轴1与轴2构成的机器1以指令速度10000对定位点P100进行直线插补的机器程序运行启动请求的执行程序

MCNST(D2000,K78)



4.16 高级同步控制专用函数

凸轮数据读取：CAMRD

格式	基本步数	可用步	
		F/FS	G
CAMRD(S1), (S2), (n), (D), (S3)	16	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S1)	—	○	—	—	○	—	—	—	—	—
(S2)	—	○	○	—	○	○	—	○	—	—
(n)	—	○	○	—	○	○	—	○	—	—
(D)	—	○	—	—	—	—	—	—	—	—
(S3)	—	○	—	—	○	—	—	—	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	凸轮No. (1~1024)	—
(S2)	凸轮数据的起始位置 行程比数据形式：1~凸轮分辨率 坐标数据形式：0~(坐标数-1) 自动生成数据形式：0	—
(n)	凸轮数据的点数 行程比数据形式：1~32768 坐标数据形式：1~16384 自动生成数据形式：0	—
(D)	存储读取的凸轮数据的软元件的起始编号	—
(S3)	读取源 0：内置存储器的凸轮展开区域(初始值) 2：SD存储卡的凸轮文件 4：标准ROM的凸轮文件	—

功能

- 将(S1)中指定的凸轮No.的凸轮数据，从(S2)中指定的凸轮数据中的位置开始，以(n)中指定的点数读取(S3)的凸轮数据。读取的凸轮数据将被存储到(D)中指定的软元件以后。但是，读取自动生成数据形式的凸轮的情况下，(S2)及(n)将被忽略。
- 对于(S2)中指定的凸轮数据的起始位置，应在以下范围内进行设置。

数据形式	起始位置的范围
行程比数据	1~凸轮分辨率*1
坐标数据	0~(坐标数-1)

*1 第0点的凸轮数据的行程比固定为0%，不能进行读取。

- 在(n)中指定读取点数。指定读取凸轮数据的最终点的存储目标时请勿超出软元件编号范围。从起始位置开始的读取点数超出凸轮数据范围情况下的动作如下所示。

数据形式	动作内容
行程比数据	“(S2)+(n)-1”的值大于凸轮分辨率的情况下，读取从凸轮数据起始位置开始至凸轮分辨率为止的凸轮数据。
坐标数据	“(S2)+(n)”的值大于坐标数的情况下，读取从凸轮数据起始位置开始至最终坐标为止的凸轮数据。

- 将(D)中指定的软元件编号设置为偶数编号，根据凸轮数据形式将读取的凸轮数据按以下方式存储到指定的软元件中。

n 行程比数据形式

偏置	项目	范围
+0	凸轮数据形式(行程比数据形式)	1
+1	凸轮数据开始位置	0~(凸轮分辨率-1)
+2	凸轮分辨率	256/512/1024/2048/4096/8192/16384/32768
+3		
+4		
+4	第1点凸轮数据值的行程比	-2147483648~2147483647[$\times 10^{-7}\%$] (-214.7483648~214.7483647[%])
+5		
+6	第2点凸轮数据值的行程比	
+7		
⋮	⋮	
+(2N+2)	第N点凸轮数据值的行程比	
+(2N+3)		

n 坐标数据形式

偏置	项目	范围	
+0	凸轮数据形式(坐标数据形式)	2	
+1	用户不能使用	0	
+2	坐标数	2~65535	
+3			
+4	第1点凸轮数据值	输入值 X_1	0~2147483647 [凸轮轴循环单位]
+5		输出值 Y_1	-2147483648~2147483647 [输出轴位置单位]
+6			
+7	第2点凸轮数据值	输入值 X_2	0~2147483647 [凸轮轴循环单位]
+8			
+9		输出值 Y_2	-2147483648~2147483647 [输出轴位置单位]
+10			
+11			
⋮	⋮	⋮	
+(4N)	第N点凸轮数据值	输入值 X_N	0~2147483647 [凸轮轴循环单位]
+(4N+1)			
+(4N+2)		输出值 Y_N	-2147483648~2147483647 [输出轴位置单位]
+(4N+3)			

n 自动生成数据形式

对通过CAMMK指令创建的凸轮文件或使用MT Developer2通过凸轮曲线创建的凸轮文件执行CAMRD指令，作为自动生成数据形式进行读取。关于读取时的软元件配置，请参阅各自动生成数据的软元件配置。

- 旋转刀具用凸轮(☞ 217页 旋转刀具用凸轮自动生成数据的软元件配置)
- 简易行程比凸轮、详细行程比凸轮(☞ 219页 自动生成数据的软元件配置)

要点

- 对于使用MT Developer2通过凸轮曲线创建的凸轮文件，作为详细行程比凸轮自动生成数据进行读取。
- 将通过MT Developer2创建的自由曲线的凸轮数据通过CAMRD指令进行了读取的情况下，所有区间的凸轮曲线类型将变为“匀速”，作为详细行程比凸轮自动生成数据，读取到指定的软元件中。

- 在(S3)中指定读取源。“0”以外的情况下，从凸轮文件中读取凸轮数据。

设置值	读取源
0	内置存储器的凸轮展开区域
2	SD存储卡的凸轮文件
4	标准ROM的凸轮文件

- (S3)中指定的读取源为“0：内置存储器的凸轮展开区域”以外的情况下，从凸轮文件中读取凸轮数据的执行过程中，“凸轮数据操作中(SM505)”将变为ON，通过读取结束变为OFF。作为互锁条件应确认“凸轮数据操作中(SM505)”的OFF。“凸轮数据操作中(SM505)”为ON中将凸轮文件作为对象执行了CAMRD指令、CAMWR指令、CAMMK指令的情况下，将发生出错。
- (S3)可以省略。省略了(S3)情况下的读取源为“0：内置存储器的凸轮展开区域”。

出错

以下情况下将发生运算出错，不进行凸轮数据的读取。

- (S1)中指定的凸轮No. 超出1~1024的范围时。(详细代码：1)
- (S1)中指定的凸轮No. 的凸轮数据不存在于(S3)中指定的位置时。(详细代码：2)
- 行程比数据形式的凸轮中，(S2)中指定的凸轮数据起始位置超出1~凸轮分辨率的范围时。(详细代码：3)
- 坐标数据形式的凸轮中，(S2)中指定的凸轮数据起始位置超出0~(坐标数-1)的范围时。(详细代码：3)
- 行程比数据形式的凸轮中，凸轮数据点数超出1~32768的范围时。(详细代码：4)
- 坐标数据形式的凸轮中，凸轮数据点数超出1~16384的范围时。(详细代码：4)
- 存储(n)中指定的凸轮数据点数的软元件编号超出范围时。(详细代码：5)
- (D)不是偶数编号的软元件时。(详细代码：6)
- 对凸轮数据中设置了“禁止读取”的文件口令的凸轮文件进行了凸轮数据读取操作时。(详细代码：7)
- “凸轮数据操作中(SM505)”为ON中，在(S3)中指定了“0”以外时。(详细代码：8)
- 将通过自由曲线创建的凸轮文件作为对象执行了CAMRD指令。(详细代码：9)
- 凸轮文件访问中发生了异常时。(详细代码：19)

程序示例

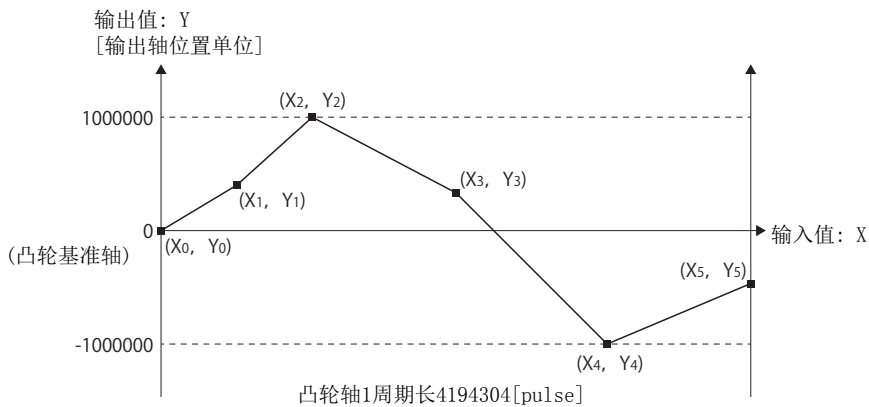
n将凸轮No. 2(行程比数据形式)的从第1点凸轮数据起的2048点的数据读取到#0至#4099中的程序

```
CAMRD K2,K1,K2048,#0
```

n 将凸轮No. 1 (坐标数据形式) 的从第0点凸轮数据起的6点的数据读取到#100至#127中的程序

```
CAMRD K1,K0,K6,#100
```

• 凸轮数据No. 1



• 读取的凸轮数据

- #100=K2 //坐标数据形式
- #102L=K6 //坐标数6
- #104L=K0 //(第1点的)输入值
- #106L=K0 //(第1点的)输出值
- #108L=K524288 //(第2点的)输入值
- #110L=K400000 //(第2点的)输出值
- #112L=K1048576 //(第3点的)输入值
- #114L=K1000000 //(第3点的)输出值
- #116L=K2097152 //(第4点的)输入值
- #118L=K300000 //(第4点的)输出值
- #120L=K3145728 //(第5点的)输入值
- #122L=K-1000000 //(第5点的)输出值
- #124L=K4194304 //(第6点的)输入值
- #126L=K-500000 //(第6点的)输出值

n 将位于SD存储卡中的凸轮No. 900 (坐标数据形式) 的第1点至2048点的凸轮数据读取到#0至#4099中的程序

```
CAMRD K900,K0,K2048,#0,K2
```

凸轮数据写入：CAMWR

格式	基本步数	可用步	
		F/FS	G
CAMWR (S1), (S2), (n), (S3), (D)	16	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S1)	—	○	—	—	○	—	—	—	—	—
(S2)	—	○	○	—	○	○	—	○	—	—
(n)	—	○	○	—	○	○	—	○	—	—
(S3)	—	○	—	—	—	—	—	—	—	—
(D)	—	○	—	—	○	—	—	—	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	凸轮No. (1~1024)	—
(S2)	凸轮数据的起始位置 行程比数据形式：1~凸轮分辨率 坐标数据形式：0~(坐标数-1)	—
(n)	凸轮数据的点数 行程比数据形式：1~32768 坐标数据形式：1~16384	
(S3)	存储写入凸轮数据的软元件的起始编号	
(D)	凸轮文件的写入目标(0000H~0401H)	

功能

- 将(S3)中指定的软元件以后存储的凸轮数据(行程比数据形式或坐标数据形式)，以从(S2)中指定的凸轮数据的位置开始的(n)中指定的点数，写入到凸轮展开区域及(D)中指定的凸轮文件的写入目标中。
- 对于(S2)中指定的凸轮数据的起始位置，应在以下范围内进行设置。

数据形式	起始位置的范围
行程比数据	1~凸轮分辨率*1
坐标数据	0~(坐标数-1)

*1 第0点的凸轮数据的行程比固定为0%，不能进行读取。

- 在(n)中指定从(S2)中指定的凸轮数据起始位置开始的写入点数。指定时应防止写入凸轮数据的最终点超出软元件编号的范围。从起始位置开始的写入点数超出凸轮数据范围的情况下将发生运算出错，不进行写入。
- 将(S3)中指定的软元件编号设置为偶数编号，根据凸轮数据形式将写入凸轮数据按以下方式存储到指定的软元件中。

n 行程比数据形式

偏置	项目	范围
+0	凸轮数据形式(行程比数据形式)	1
+1	凸轮数据开始位置	0~(凸轮分辨率-1)
+2	凸轮分辨率	256/512/1024/2048/4096/8192/16384/32768
+3		
+4	第1点凸轮数据值的行程比	-2147483648~2147483647[$\times 10^{-7}\%$] (-214.7483648~214.7483647[%])
+5		
+6	第2点凸轮数据值的行程比	
+7		
⋮	⋮	
+(2N+2)	第N点凸轮数据值的行程比	
+(2N+3)		

n 坐标数据形式

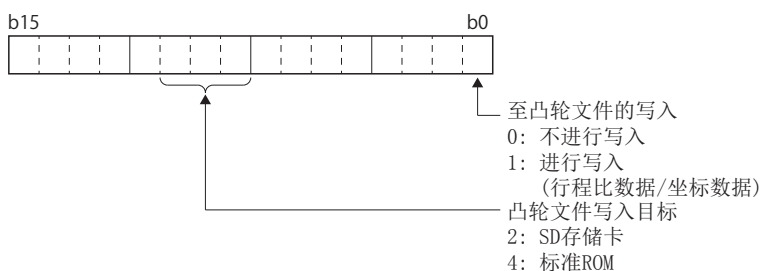
偏置	项目	范围	
+0	凸轮数据形式(坐标数据形式)	2	
+1	用户不能使用	0	
+2	坐标数	2~65535	
+3			
+4	第1点凸轮数据值	输入值 X_1	0~2147483647 [凸轮轴循环单位]
+5		输出值 Y_1	-2147483648~2147483647 [输出轴位置单位]
+6			
+7			
+8	第2点凸轮数据值	输入值 X_2	0~2147483647 [凸轮轴循环单位]
+9		输出值 Y_2	-2147483648~2147483647 [输出轴位置单位]
+10			
+11			
⋮	⋮	⋮	⋮
+(4N)	第N点凸轮数据值	输入值 X_N	0~2147483647 [凸轮轴循环单位]
+(4N+1)		输出值 Y_N	-2147483648~2147483647 [输出轴位置单位]
+(4N+2)			
+(4N+3)			

- 在CAMWR指令的执行中，不能对其它CAMWR指令、CAMMK指令进行处理。CAMWR指令执行中时，“凸轮数据操作中(SM505)”将变为ON，因此应作为互锁条件置入。“凸轮数据操作中(SM505)”为ON中执行了了CAMWR指令、CAMMK指令的情况下，将发生出错。

要点

CAMWR指令在同步控制的动作中也可执行。根据执行时机，执行中的凸轮数据将变化，应加以注意。

- 在(D)中指定至凸轮文件的写入、凸轮文件写入目标。



- (D)可以省略。省略(D)或指定超出范围的情况下,只能写入到凸轮展开区域中。进行至凸轮文件的写入的情况下,应指定(D)。

要点

在(D)中将凸轮文件的写入指定为“1: 写入”的情况下,至凸轮展开区域的写入完成后,进行至凸轮文件的写入。凸轮文件操作中发生了出错的情况下,凸轮展开区域的凸轮数据将被保持。

- (S1)中指定的凸轮No.已存在于凸轮展开区域中情况下的动作如下所示。
 - 凸轮数据形式及分辨率/坐标数相同的情况下,将覆盖现有的凸轮数据。
 - 凸轮数据形式或分辨率/坐标数不相同的情况下,现有的凸轮数据将被删除。(凸轮数据的点数小于分辨率/坐标数的情况下,未进行写入的区间的凸轮数据将变为不定值。)

出错

以下情况下将发生运算出错,不进行凸轮数据的写入。

- (S1)中指定的凸轮No.超出1~1024的范围时。(详细代码: 1)
- 行程比数据形式的凸轮中,(S2)中指定的凸轮数据起始位置超出1~凸轮分辨率的范围时。(详细代码: 2)
- 坐标数据形式的凸轮中,(S2)中指定的凸轮数据起始位置超出0~(坐标数-1)的范围时。(详细代码: 2)
- 行程比数据形式的凸轮中,凸轮数据点数超出1~32768的范围时。(详细代码: 3)
- 坐标数据形式的凸轮中,凸轮数据点数超出1~16384的范围时。(详细代码: 3)
- 设置了凸轮分辨率或超出坐标数的范围的起始位置及操作点数时。(详细代码: 3)
- 存储(n)中指定的凸轮数据点数的软元件编号超出范围时。(详细代码: 4)
- (S3)不是偶数编号的软元件时。(详细代码: 5)
- (S3)中指定的凸轮数据形式中设置了1、2以外的值时。(详细代码: 6)
- 在行程比数据形式的凸轮中,将凸轮分辨率设置为“256/512/1024/2048/4096/8192/16384/32768”以外的值时。(详细代码: 7)
- 坐标数据形式的凸轮中,将坐标数设置为“2~65535”以外的值时。(详细代码: 7)
- 行程比数据形式的凸轮中,凸轮数据开始位置超出0~(凸轮分辨率-1)的范围时。(详细代码: 8)
- 凸轮数据写入操作时,可写入的区域不足时。(详细代码: 10)
- 坐标数据的输入值为负值时。(详细代码: 11)
- 坐标数据的输入值为“ $X_n > X_{n+1}$ ”时。(详细代码: 11)
- “凸轮数据操作中(SM505) ON中,执行了CAMWR指令。(详细代码: 13)

以下情况下将发生运算出错,不进行至凸轮文件的写入。(对凸轮展开区域进行写入。)

- 凸轮文件访问中发生了异常时。(详细代码: 19)

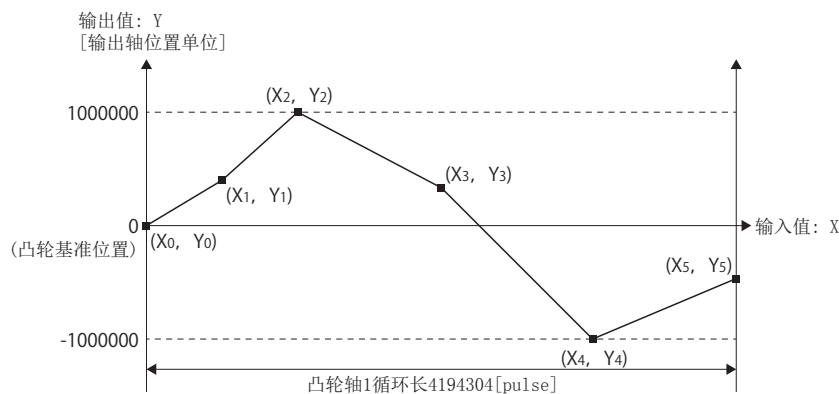
程序示例

n 将#0至#4099中存储的数据，写入到凸轮No. 256(行程比数据形式)的凸轮数据第1点至2048点的区域中的程序

```
CAMWR K256,K1,K2048,#0
```

n 将#0至#27中存储的数据，写入到凸轮No. 255(坐标数据形式)的第0点凸轮数据起的6点的区域中的程序(凸轮轴1循环长=4194304)

```
#0=K2           //坐标数据形式
#2L=K6         //坐标数6
#4L=K0         //(第1点的)输入值
#6L=K0         //(第1点的)输出值
#8L=K524288    //(第2点的)输入值
#10L=K400000   //(第2点的)输出值
#12L=K1048576 //(第3点的)输入值
#14L=K1000000 //(第3点的)输出值
#16L=K2097152 //(第4点的)输入值
#18L=K300000   //(第4点的)输出值
#20L=K3145728 //(第5点的)输入值
#22L=K-1000000 //(第5点的)输出值
#24L=K4194304 //(第6点的)输入值
#26L=K-500000  //(第6点的)输出值
CAMWR K255,K0,K6,#0 //拖轮数据写入
```



凸轮自动生成功能：CAMMK

格式	基本步数	可用步	
		F/FS	G
CAMMK (S1), (S2), (S3), (D)	13	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S1)	—	○	—	—	○	—	—	—	—	—
(S2)	—	○	—	—	○	—	—	—	—	—
(S3)	—	○	—	—	—	—	—	—	—	—
(D)	—	○	—	—	○	—	—	—	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	凸轮No. (1~1024)	—
(S2)	凸轮自动生成类型 旋转刀具用凸轮：1 简易行程比凸轮：2 详细行程比凸轮：3	
(S3)	存储自动生成用数据的软元件的起始编号	
(D)	凸轮文件的写入目标(0000H~0401H)	

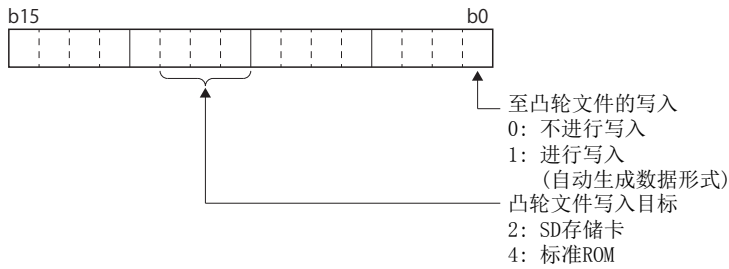
功能

- 以(S2)中指定的凸轮自动生成类型及(S3)中指定的软元件中存储的自动生成用数据为基础，将(S1)中指定的自动生成凸轮No.的凸轮数据创建到凸轮展开区域及(D)中指定的凸轮文件的写入目标中。通过在(D)中指定标准ROM，下次多CPU系统的电源投入时将自动执行凸轮自动生成。
- 在(S2)中指定以下凸轮自动生成类型。

凸轮自动生成类型	设置值
旋转刀具用凸轮	1
简易行程比凸轮	2
详细行程比凸轮	3

- 在(S3)中设置(S2)中指定的凸轮自动生成类型对应的自动生成用数据。指定软元件编号应指定为偶数编号。将自动生成用数据配置到指定软元件以后，指定时应避免最终点的软元件编号超出范围。
- CAMMK指令的执行中，不能处理其它CAMWR指令、CAMMK指令。CAMMK指令执行中时，“凸轮数据操作中(SM505)”将变为ON，因此应作为互锁条件置入。“凸轮数据操作中(SM505)”为ON中执行了CAMWR指令、CAMMK指令的情况下，将发生运算出错。

- 在(D)中指定至凸轮文件的写入、凸轮文件写入目标。至凸轮文件的写入为“0: 不写入”的情况下, 仅进行至凸轮展开区域的写入, 不对凸轮文件进行写入。



- (D)可以省略。省略(D)或指定超出范围的情况下, 只能写入到凸轮展开区域中。进行至凸轮文件的写入的情况下, 应指定(D)。

要点

在(D)中将凸轮文件的写入指定为“1: 写入”的情况下, 至凸轮展开区域的写入完成后, 进行至凸轮文件的写入。凸轮文件操作中发生了出错的情况下, 凸轮展开区域的凸轮数据将被保持。

出错

以下情况下将发生运算出错, 不进行凸轮自动生成。

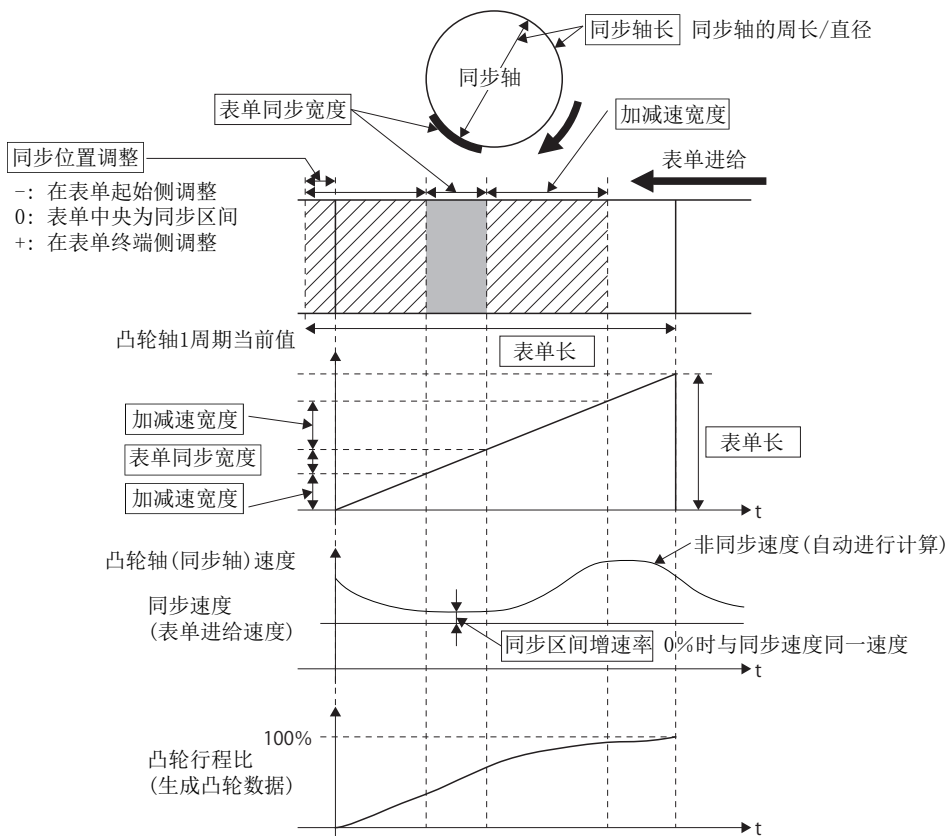
- (S1)中指定的凸轮No. 超出1~1024的范围时。(详细代码: 1)
- (S2)中指定的自动生成类型中, 设置了对应类型以外的值时。(详细代码: 2)
- 存储(S3)中指定的自动生成用数据的软元件编号超出范围时。(详细代码: 5)
- (S3)不是偶数编号的软元件时。(详细代码: 6)
- 凸轮数据写入操作时, 可写入的区域不足时。(详细代码: 4)
- 自动生成用数据中设置了超出范围的值时。(详细代码: 7)
- 在旋转刀具用凸轮中, 自动生成参数值被设置为“表单同步宽度 \geq 表单长”的值时。(详细代码: 8)
- 在旋转刀具用凸轮中, 自动生成用数据为“同步轴长(同步轴径 $\times \pi \times \pi$) $<$ 表单长”时, 非同步速度变为负值时。(详细代码: 9)
- 在旋转刀具用凸轮中, 根据自动生成用数据, 非同步速度大于同步速度的655.35倍时。(详细代码: 10)
- “凸轮数据操作中(SM505)”ON中, 执行了CAMMK指令。(详细代码: 12)
- 简易行程比凸轮或详细行程比凸轮中, 各区间中设置的结束点不是升序时。(详细代码: 20)
- 简易行程比凸轮或详细行程比凸轮中, 最终区间的结束点不足凸轮轴1循环长时。(详细代码: 21)
- 详细行程比凸轮中, 曲线应用范围(P1, P2)或加减速度范围补偿(L1, L2)的关系不正确时。(详细代码: 23)
- 详细行程比凸轮中, 各区间的行程与凸轮行程量的关系不正确时。(详细代码: 7)

以下情况下将发生运算出错, 不进行至凸轮文件的写入。(对凸轮展开区域进行写入。)

- 凸轮文件访问中发生了异常时。(详细代码: 19)

旋转刀具用凸轮

设置旋转刀具用凸轮数据的表单长及同步宽度等的自动生成用数据。



n 旋转刀具用凸轮自动生成用数据的软元件配置

同步位置调整为0的情况下，将生成表单中央变为同步区间的凸轮模式。

偏置	名称	内容	范围
+0	分辨率	设置生成的凸轮的分辨率。	256/512/1024/2048/4096/
+1			8192/16384/32768
+2	自动生成选项	<ul style="list-style-type: none"> 在位0中选择梯形/S字加减速方式。 在位1中选择是否将同步轴长以直径/周长进行设置。 应将位2~15设置为0。 	<ul style="list-style-type: none"> 位0: 加减速方式 0: 梯形加减速 1: S字加减速 位1: 同步轴长设置 0: 直径 1: 周长
+3	同步区间增速率	对同步区间的同步速度进行微调时进行此设置。变为“同步区间速度=同步速度×(100%+增速率)”。	-5000~5000[×0.01%]
+4	表单长	设置表单长。	1~2147483647
+5			[任意的同一单位]
+6	表单同步宽度	<ul style="list-style-type: none"> 设置表单同步宽度(封口宽度)。 表单同步宽度前后需要保存动作用的同步速度区间的情况下，应留出足够的保存宽度。 	1~2147483647
+7			[任意的同一单位]
+8	同步轴长	<ul style="list-style-type: none"> 设置旋转刀具轴长。 自动生成选项为直径设置的情况下，作为“周长=设置值×π”进行计算。 自动生成选项为周长设置的情况下，作为“周长=设置值”进行计算。 	<ul style="list-style-type: none"> 直径设置的情况下 1~680000000 周长设置的情况下 1~2147483647
+9			[任意的同一单位]
+10	同步位置调整	<ul style="list-style-type: none"> 设置同步区间的位置调整。 -: 将同步区间调整为表单起始侧 0: 表单中央为同步区间 +: 将同步区间调整为表单终端侧 应在表单长的1/2以内进行设置。 	-1073741823~1073741823
+11			[任意的同一单位]
+12	加减速宽度	<ul style="list-style-type: none"> 设置加减速区域的表单宽度(单侧)。 设置了负值的情况下，将以最大加减速宽度进行计算。 	0~2147483647
+13			[任意的同一单位]
			<ul style="list-style-type: none"> 除上述以外时按最大加减速宽度进行计算

偏置	名称	内容	范围
+14	刀具数	设置刀具数。	1~256
+15	非同步速度结果	凸轮自动生成正常生成时，非同步速度作为同步速度的比率被存储。	0~65535[0.01倍]

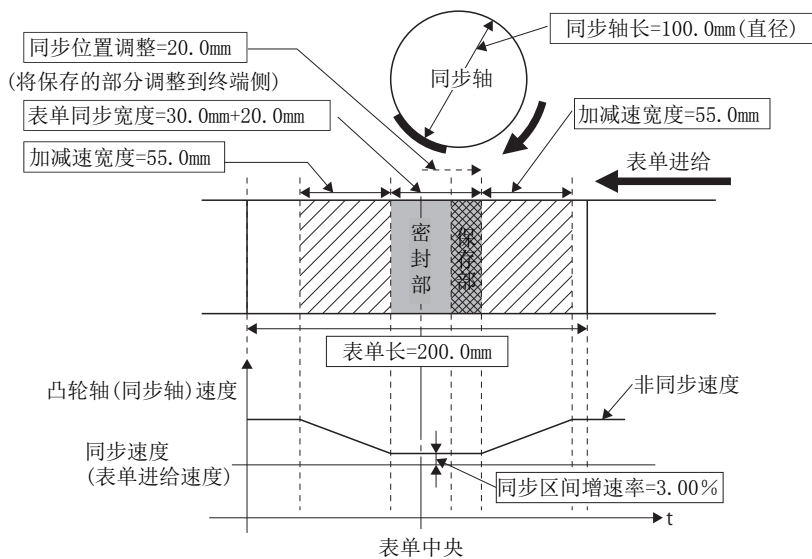
n 程序示例

在凸轮No. 5中创建旋转刀具用动作模式的凸轮数据(分辨率512)的程序。

```

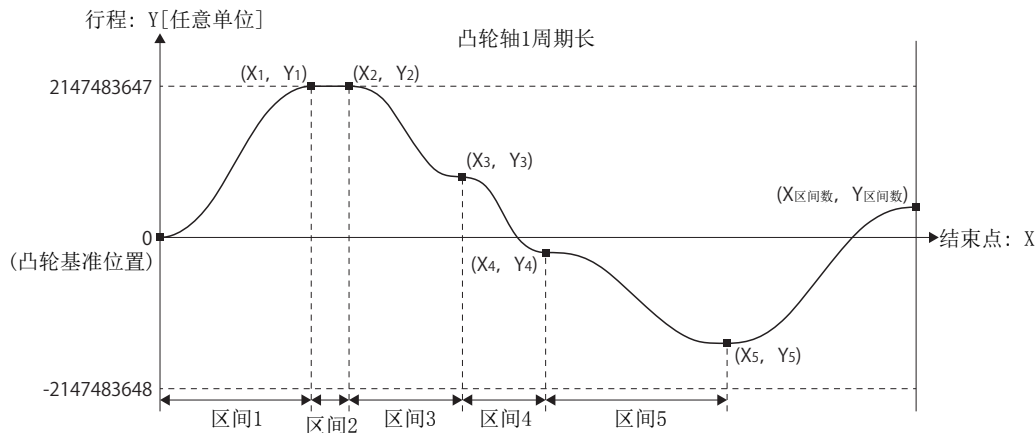
D5000L=K512 //分辨率=512
D5002=K0 //加减速方式=梯形，同步轴长设置=直径
D5003=K300 //同步区间增速率=3.00%
D5004L=K2000 //表单长=200.0mm
D5006L=K500 //表单同步宽度=30.0mm(密封部)+20.0mm(保存动作)
D5008L=K1000 //同步轴长=100.0mm(直径)
D5010L=K200 //同步位置调整=20.0mm
D5012L=K550 //加减速宽度=55.0mm
D5014=K1 //刀具数=1
CAMMK K5,K1,D5000 //凸轮自动生成(D5015中将存储非同步速度结果)

```



简易行程比凸轮/详细行程比凸轮

可在不使用MT Developer2的凸轮数据设置的情况下，通过设置区间及行程量自动创建凸轮数据。在简易行程比凸轮中，省略凸轮曲线的详细系数的设置，可以使用的曲线及区间数被限定。将1循环当前值0作为开始点，通过各区间中指定的结束点(凸轮轴1循环当前值)为止的行程及凸轮曲线类型自动生成凸轮数据。



n 自动生成用数据的软元件配置

• 简易行程比凸轮自动生成用数据

偏置	名称	内容	范围	
+0	分辨率	设置生成的凸轮的分辨率。	256/512/1024/2048/4096/ 8192/ 16384/32768	
+1				
+2	凸轮轴1循环长*1	设置凸轮的一系列动作的循环长。	1~2147483647 [凸轮轴1循环单位]	
+3				
+4	凸轮数据开始位置	设置“循环长=0”的位置对应的凸轮数据的开始位置。	0~(分辨率-1)	
+5				
+6	区间数	设置凸轮数据的区间数。 应设置相当于指定区间数的数据。	1~32	
+7	用户不能使用		应设置为0。	
+8	区间1	凸轮曲线类型*2	设置凸轮曲线。 0: 匀速 1: 加速度 2: 修正梯形 3: 修正正弦 4: 修正匀速 5: 摆线 6: 5次曲线	
+9		用户不能使用	应设置为0。	
+10		结束点 (X ₁)	设置对于凸轮轴1循环长的位置(凸轮轴1循环当前值)。结束点需设置为大于之前的结束点的值(X _n <X _{n+1})。此外, 最终结束点中, 应设置凸轮轴1循环长。	1~凸轮轴1循环长 [凸轮轴1循环单位]*3
+11				
+12				
+13	行程 (Y ₁)	设置凸轮轴1循环当前值位于指定的结束点时的, 从凸轮基准位置开始的行程位置。 设置了1000000000时, 将变为[Pr. 441]凸轮行程量(R: D42704+160n, D42705+160n/Q: D15064+150n, D15065+150n)中设置的位置。	-2147483648~2147483647 [×0.0000001%]	
+14	区间2	凸轮曲线类型*2	“区间数”中设置的数据将生效。 设置的区间数后面的数据无需设置。 0: 匀速 1: 恒定加速度 2: 修正梯形 3: 修正正弦 4: 修正匀速 5: 摆线 6: 5次曲线	
+15		用户不能使用	0	
+16		结束点 (X ₂)	1~凸轮轴1循环长 [凸轮轴1循环单位]*3	
+17				
+18				
+19	行程 (Y ₂)	-2147483648~ 2147483647[×0.0000001%]		
:	:	:	:	
+194	区间32	凸轮曲线类型*2	0: 匀速 1: 恒定加速度 2: 修正梯形 3: 修正正弦 4: 修正匀速 5: 摆线 6: 5次曲线	
+195		用户不能使用	0	
+196		结束点 (X ₃₂)	1~凸轮轴1循环长[凸轮轴1循环单位]*3	
+197				
+198				
+199	行程 (Y ₃₂)	-2147483648~ 2147483647[×0.0000001%]		

*1 设置的值仅用于生成凸轮数据时。控制时通过输出轴参数“[Pr. 439]凸轮轴1循环长(R: D42700+160n, D42701+160n/Q: D15060+150n, D15061+150n)”进行控制。

*2 关于凸轮曲线类型的形状, 请参阅凸轮曲线一览(☞ 222页 凸轮曲线一览)。但是, P1=0、P2=1.00、L1、L2使用默认值。

*3 设置超出范围的情况下, 将被视为区间设置的最终结束点被设置为凸轮轴1循环长。

• 详细行程比凸轮自动生成用数据

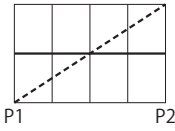
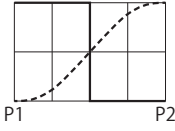
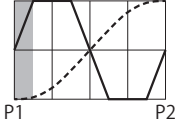
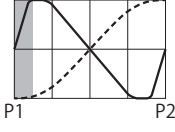
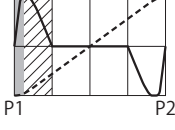
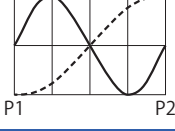
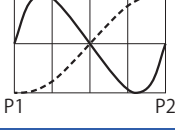
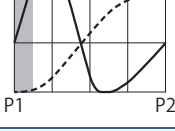
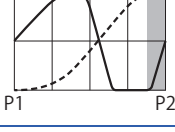
偏置	名称	内容	范围
+0	分辨率	设置生成的凸轮的分辨率。	256/512/1024/2048/4096/ 8192/ 16384/32768
+1			
+2	凸轮轴1循环长*1	设置凸轮的一系列动作的循环长。	1~2147483647 [凸轮轴1循环单位]
+3			
+4	凸轮行程量*1	设置行程(Yn)中指定的行程量的基准值。 凸轮行程量单位为“%”时忽略。	1~2147483647 [凸轮行程量单位]
+5			
+6	单位设置	以16进制数设置凸轮轴1循环长的显示单位及凸轮行程量单位。设置了超出设置范围的值的情况下，生成默认值(H7952)的凸轮数据。	 <p> H□□□□ □ 凸轮轴1周期长单位 (显示用) 0: mm 1: inch 2: degree 3: pulse 小数点位数 0~9 凸轮行程量单位 0: mm 1: inch 2: degree 3: pulse 9: % 小数点位数 0~9 </p>
+7	用户不能使用	应设置为0。	0
+8	凸轮数据开始位置	设置“循环长=0”的位置对应的凸轮数据的开始位置。	0~(分辨率-1)
+9			
+10	区间数	设置凸轮数据的区间数。应设置相当于指定区间数的数据。	1~360
+11	用户不能使用	应设置为0。	0
+12	区间1	凸轮曲线类型*2	设置凸轮曲线。 0: 匀速 1: 恒定加速度 2: 修正梯形 3: 修正正弦 4: 修正匀速 5: 摆线 6: 5次曲线 7: 单停留摆线 8: 单停留逆摆线 9: 多弦 10: 逆多弦 11: 单弦
+13	用户不能使用	应设置为0。	0
+14	结束点 (X ₁)	设置对于凸轮轴1循环长的位置(凸轮轴1循环当前值)。 结束点需设置为大于之前的结束点的值(Xn<Xn+1)。此外，最终结束点中，应设置凸轮轴1循环长。	1~凸轮轴1循环长 [凸轮轴1循环单位]*3
+15			
+16	行程 (Y ₁)*4	设置凸轮轴1循环当前值位于指定的结束点时的，从凸轮基准位置开始的行程位置。 设置了1000000000时，将变为“[Pr. 441]凸轮行程量(R: D42704+160n, D42705+160n/Q: D15064+150n, D15065+150n)”中设置的位置。	-2147483648~2147483647 [凸轮行程单位]
+17			
+18	曲线应用范围(P1)	设置凸轮曲线的曲线应用范围(起点: P1, 终点: P2)。设置时应满足“P1<P2”的条件。但是，“P1=P2=0”时使用“P1=0”、“P2=100”。	0~100[×0.01]
+19	曲线应用范围(P2)		
+20	加减速范围补偿(范围L1)*2	设置凸轮曲线中加减速的范围(L1、L2)。根据凸轮曲线可设置的范围有所不同。“L1=L2=0”时，使用各凸轮曲线的默认值。	1~9999[×0.0001]
+21	加减速范围补偿(范围L2)*2		

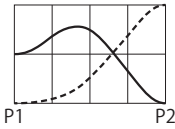
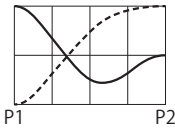
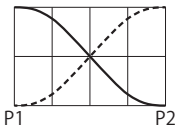
偏置	名称	内容	范围
+22	区间2 凸轮曲线类型*2	“区间数”中设置的数据将生效。 设置的区间数后面的数据无需设置。	0: 匀速 1: 恒定加速度 2: 修正梯形 3: 修正正弦 4: 修正匀速 5: 摆线 6: 5次曲线 7: 单停留摆线 8: 单停留逆摆线 9: 多弦 10: 逆多弦 11: 单弦
+23	用户不能使用		0
+24	结束点 (X ₂)		1~凸轮轴1循环长 [凸轮轴1循环单位]*3
+25			
+26	行程 (Y ₂)*4		-2147483648~2147483647[凸轮行程单位]
+27			
+28	曲线应用范围(P1)		0~100[×0.01]
+29	曲线应用范围(P2)		
+30	加减速范围补偿 (范围L1)*2		1~9999[×0.0001]
+31	加减速范围补偿 (范围L2)*2		
⋮	⋮		⋮
+3602	区间 360 凸轮曲线类型*2		0: 匀速 1: 恒定加速度 2: 修正梯形 3: 修正正弦 4: 修正匀速 5: 摆线 6: 5次曲线 7: 单停留摆线 8: 单停留逆摆线 9: 多弦 10: 逆多弦 11: 单弦
+3603	用户不能使用		0
+3604	结束点 (X ₃₂)		1~凸轮轴1循环长[凸轮轴1循环单位]*3
+3605			
+3606	行程 (Y ₃₂)*4		-2147483648~2147483647 [凸轮行程单位]
+3607			
+3608	曲线应用范围(P1)		0~100[×0.01]
+3609	曲线应用范围(P2)		
+3610	加减速范围补偿 (范围L1)*2		1~9999[×0.0001]
+3611	加减速范围补偿 (范围L2)*2		

- *1 设置的值仅用于生成凸轮数据时。控制时通过输出轴参数“[Pr. 439]凸轮轴1循环长(R: D42700+160n, D42701+160n/Q: D15060+150n, D15061+150n)”、“[Pr. 441]凸轮行程量(R: D42704+160n, D42705+160n/Q: D15064+150n, D15065+150n)”进行控制。
- *2 关于凸轮曲线类型的形状以及L1与L2的范围, 请参阅凸轮曲线一览(☞ 222页 凸轮曲线一览)。在不使用L1及L2的凸轮曲线类型L1及L2中, 应设置为0。
- *3 设置超出范围的情况下, 将被视为区间设置的最终结束点被设置为凸轮轴1循环长。
- *4 凸轮行程量单位为“%”以外时, 行程量的设置应满足“行程(Yn)÷凸轮行程量”的范围在“-214.7483648~214.7483647”以内。

- 应设置相当于设置的区间数的数据。设置的区间数以后的数据无需设置。
- 结束点的数据应按升序进行设置。
- 对于凸轮数据，根据各区间的数据及行程设置，将创建各式各样的凸轮模式。行程的变化量较大等情况下，放大器中有可能发生过速度、数据异常等伺服出错。创建凸轮时，应通过无放大器运行确认凸轮的动作。
- 结束点将被视为在自动生成用数据中设置的凸轮轴1循环长以上的区间中结束凸轮数据。

n 凸轮曲线一览

凸轮曲线类型			加速度曲线的形状*1	曲线应用范围 (P1~P2)	加减速范围补偿 ()为默认值	
设置值	凸轮曲线名	范围L1			范围L2	
0	匀速	不连续		0.00~1.00	—	—
1	恒定加速度			0.00~1.00	—	—
2	修正梯形	双停留对称		0.00~1.00	0.0001~0.2499 (0.1250)	—
3	修正正弦			0.00~1.00	0.0001~0.4999 (0.1250)	—
4	修正匀速			0.00~1.00	0.0001~0.1249 (0.0625)	0.0001~0.4999 (0.2500)
5	摆线			0.00~1.00	—	—
6	5次曲线			0.00~1.00	—	—
7	单停留摆线	双停留非对称		0.00~1.00	0.0001~0.2499 (0.1250)	—
8	单停留逆摆线			0.00~1.00	0.0001~0.2499 (0.1250)	—

凸轮曲线类型		加速度曲线的形状*1	曲线应用范围 (P1~P2)	加减速范围补偿 ()为默认值	
设置值	凸轮曲线名			范围L1	范围L2
9	多弦 单侧停留		0.00~1.00	—	—
10	逆多弦		0.00~1.00	—	—
11	单弦 无停留		0.00~1.00	—	—

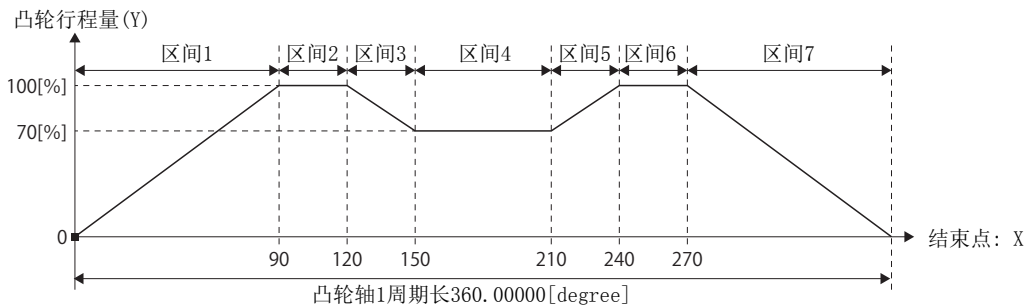
*1 —: 行程比 ----: 加速度 ■: 范围1 ▨: 范围2

n 创建简易行程比凸轮的凸轮数据的程序

- 在凸轮No. 5中创建凸轮数据(分辨率512)的程序

```

D5000L=K512           //分辨率=512
D5002L=K36000000      //凸轮轴1循环长=360.0[degree]
D5004L=K0             //凸轮数据开始位置=第0点
D5006=K7              //区间数=7区间
D5007=K0              //用户不能使用
D5008=K0              //(区间1的)凸轮曲线类型=匀速
D5009=K0              //用户不能使用
D5010L=K9000000       //(区间1的)结束点(X1)=90.0[degree]
D5012L=K100000000     //(区间1的)行程(Y1)=100.0[%]
D5014=K0              //(区间2的)凸轮曲线类型=匀速
D5015=K0              //用户不能使用
D5016L=K12000000      //(区间2的)结束点(X2)=120.0[degree]
D5018L=K100000000     //(区间2的)行程(Y2)=100.0[%]
D5020=K0              //(区间3的)凸轮曲线类型=匀速
D5021=K0              //用户不能使用
D5022L=K15000000      //(区间3的)结束点(X3)=150.0[degree]
D5024L=K700000000     //(区间3的)行程(Y3)=70.0[%]
D5026=K0              //(区间4的)凸轮曲线类型=匀速
D5027=K0              //用户不能使用
D5028L=K21000000      //(区间4的)结束点(X4)=210.0[degree]
D5030L=K700000000     //(区间4的)行程(Y4)=70.0[%]
D5032=K0              //(区间5的)凸轮曲线类型=匀速
D5033=K0              //用户不能使用
D5034L=K24000000      //(区间5的)结束点(X5)=240.0[degree]
D5036L=K100000000     //(区间5的)行程(Y5)=100.0[%]
D5038=K0              //(区间6的)凸轮曲线类型=匀速
D5039=K0              //用户不能使用
D5040L=K27000000      //(区间6的)结束点(X6)=270.0[degree]
D5042L=K100000000     //(区间6的)行程(Y6)=100.0[%]
D5044=K0              //(区间7的)凸轮曲线类型=匀速
D5045=K0              //用户不能使用
D5046L=K36000000      //(区间7的)结束点(X7)=360.0[degree]
D5048L=K0              //(区间7的)行程(Y7)=0[%]
CAMMK K5,K2,D5000     //凸轮自动生成
    
```

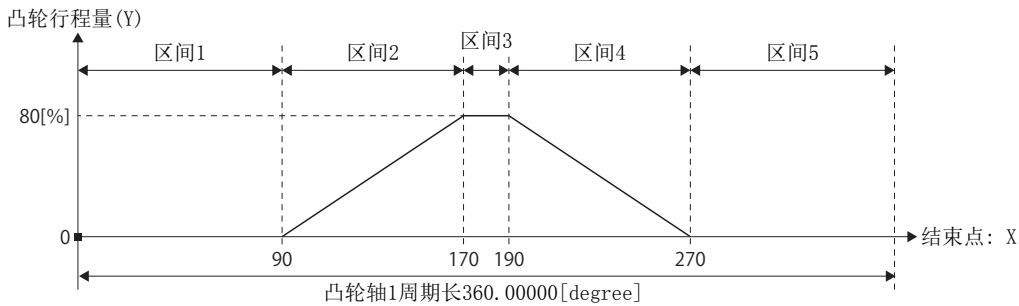


- 在凸轮No. 6中创建凸轮数据(分辨率512)的程序

```

D6000L=K512 //分辨率=512
D6002L=K3600000 //凸轮轴1循环长=360.0[degree]
D6004L=K0 //凸轮数据开始位置=第0点
D6006=K5 //区间数=5区间
D6007=K0 //用户不能使用
D6008=K0 //(区间1的)凸轮曲线类型=匀速
D6009=K0 //用户不能使用
D6010L=K9000000 //(区间1的)结束点(X1)=90.0[degree]
D6012L=K0 //(区间1的)行程(Y1)=0[%]
D6014=K0 //(区间2的)凸轮曲线类型=匀速
D6015=K0 //用户不能使用
D6016L=K17000000 //(区间2的)结束点(X2)=170.0[degree]
D6018L=K80000000 //(区间2的)行程(Y2)=80[%]
D6020=K0 //(区间3的)凸轮曲线类型=匀速
D6021=K0 //用户不能使用
D6022L=K19000000 //(区间3的)结束点(X3)=190.0[degree]
D6024L=K80000000 //(区间3的)行程(Y3)=80[%]
D6026=K0 //(区间4的)凸轮曲线类型=匀速
D6027=K0 //用户不能使用
D6028L=K27000000 //(区间4的)结束点(X4)=270.0[degree]
D6030L=K0 //(区间4的)行程(Y4)=0[%]
D6032=K0 //(区间5的)凸轮曲线类型=匀速
D6033=K0 //用户不能使用
D6034L=K36000000 //(区间5的)结束点(X5)=360.0[degree]
D6036L=K0 //(区间5的)行程(Y5)=0[%]
CAMMK K6,K2,D6000 //凸轮自动生成

```



凸轮位置计算：CAMPSCL

格式	基本步数	可用步	
		F/FS	G
CAMPSCL (S1), (S2), (D)	12	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据								计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数						
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)				
(S1)	—	○	—	—	○	—	—	—	—	—	
(S2)	—	○	—	—	—	—	—	—	—	—	
(D)	—	—	○	—	—	—	—	—	—	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	凸轮位置计算凸轮No. (0~1024)	—
(S2)	存储凸轮位置计算控制数据的软元件的起始编号	
(D)	存储凸轮位置计算结果的软元件编号	

功能

- 对(S1)中指定的凸轮No.的凸轮，以(S2)中指定的凸轮位置计算控制数据为基础计算凸轮轴进给当前值或凸轮轴1循环当前值后，输出到(D)中指定的软元件中。
- 在(S1)中指定进行凸轮位置计算的凸轮No.。指定了凸轮No.0的情况下，以直线凸轮计算凸轮位置。对于凸轮No.1~1024，以凸轮展开区域上的凸轮数据计算凸轮位置。
- 将(S2)中指定的软元件编号设置为偶数编号，指定的软元件中应按以下方式设置凸轮位置计算控制数据。

n 凸轮位置计算控制数据的软元件配置

偏置	名称	内容	范围
+0	凸轮位置计算类型	指定凸轮轴进给当前值计算/凸轮轴1循环当前值计算。	0: 凸轮轴进给当前值计算 1: 凸轮轴1循环当前值计算
+1	用户不能使用	应设置为0。	
+2	凸轮行程量	设置凸轮位置计算中使用的凸轮行程量。	-2147483648~2147483647 [输出轴位置单位]
+3			
+4	凸轮轴1循环长	设置凸轮位置计算中使用的凸轮轴1循环长。	1~2147483647 [凸轮轴循环单位]
+5			
+6	凸轮基准位置	设置凸轮位置计算中使用的凸轮基准位置。	-2147483648~2147483647 [输出轴位置单位]
+7			
+8	凸轮轴1循环当前值	<ul style="list-style-type: none"> 凸轮轴进给当前值计算时，设置凸轮位置计算中使用的凸轮轴1循环当前值。 凸轮轴1循环当前值计算时，设置凸轮位置计算时开始搜索的凸轮轴1循环当前值。 	0~(凸轮轴1循环长) [凸轮轴循环单位]
+9			
+10	凸轮轴进给当前值	凸轮轴1循环当前值计算时，设置凸轮位置计算中使用的凸轮轴进给当前值。(在凸轮位置计算类型为凸轮轴进给当前值计算中不使用。)	-2147483648~2147483647 [输出轴位置单位]
+11			

- (D)中指定的软元件编号应指定为偶数编号。计算完成时如下所示的凸轮位置计算的结果将被存储到指定软元件中。

凸轮位置计算	内容
凸轮轴进给当前值计算时	按以下范围存储计算出的凸轮轴进给当前值的值。 -2147483648~2147483647 [输出轴位置单位]
凸轮轴1循环当前值计算时	按以下范围存储计算出的凸轮轴1循环当前值的值。 0~(凸轮轴1循环长-1) [凸轮轴循环单位]

- 在凸轮位置计算功能中，不自动更新凸轮基准位置。

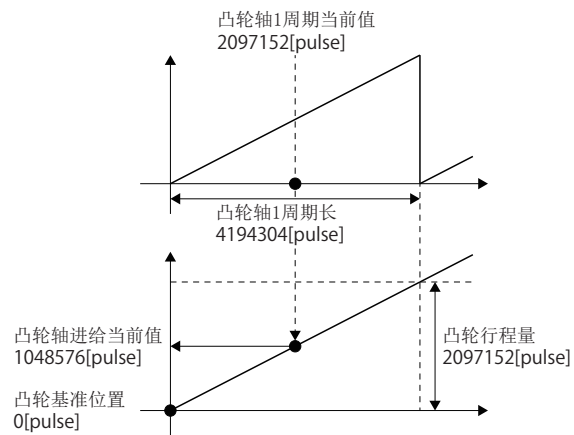
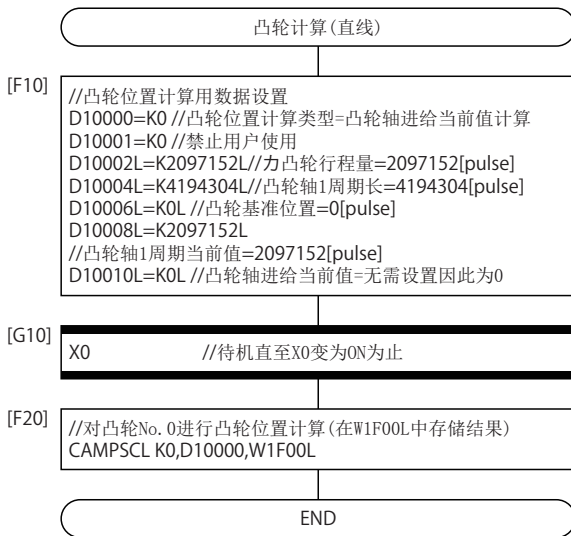
出错

以下情况下将发生运算出错，不进行凸轮位置计算。

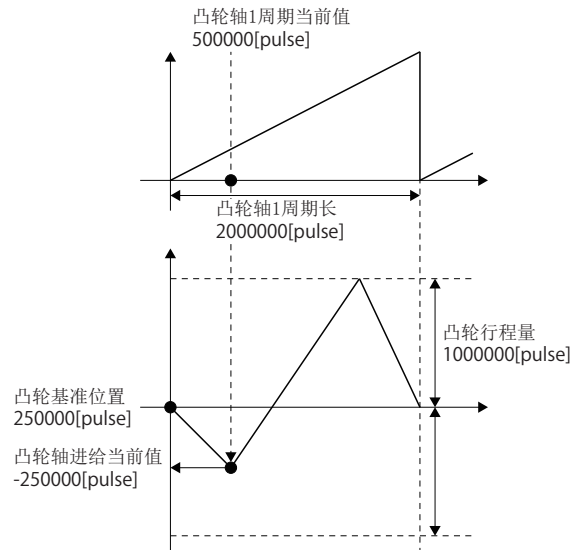
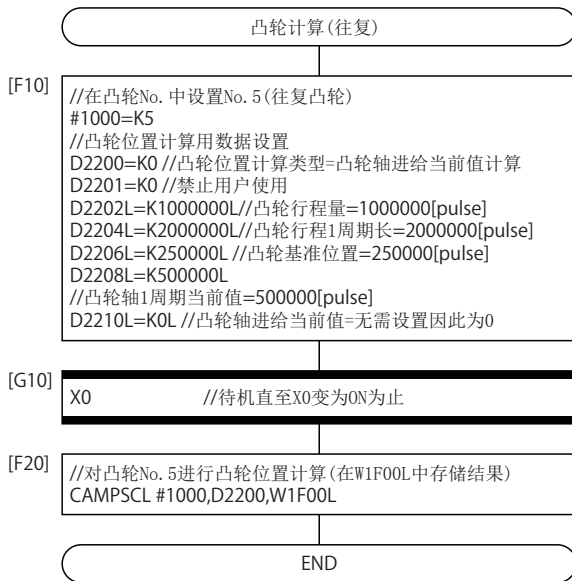
- (S1)中指定的凸轮No. 超出0~1024的范围时。(详细代码: 1)
- (S1)中指定的凸轮No. 的凸轮数据在凸轮展开区域中不存在时。(详细代码: 2)
- 存储(S2)中指定的凸轮位置计算控制数据的软元件编号超出范围时。(详细代码: 3)
- (S2)为偶数编号的软元件以外时。(详细代码: 4)
- 凸轮位置计算控制数据中指定的凸轮位置计算类型中设置了0、1以外的值时。(详细代码: 5)
- 凸轮轴1循环长超出1~2147483647的范围时。(详细代码: 6)
- 凸轮轴1循环当前值超出0~(凸轮轴1循环长)的范围时。(详细代码: 7)
- 存储(D)中指定的凸轮位置计算结果的软元件编号超出范围时。(详细代码: 8)
- (D)为偶数编号的软元件以外时。(详细代码: 9)
- 凸轮轴1循环当前值计算时，无法计算相应凸轮轴1循环当前值时。(详细代码: 10)

程序示例

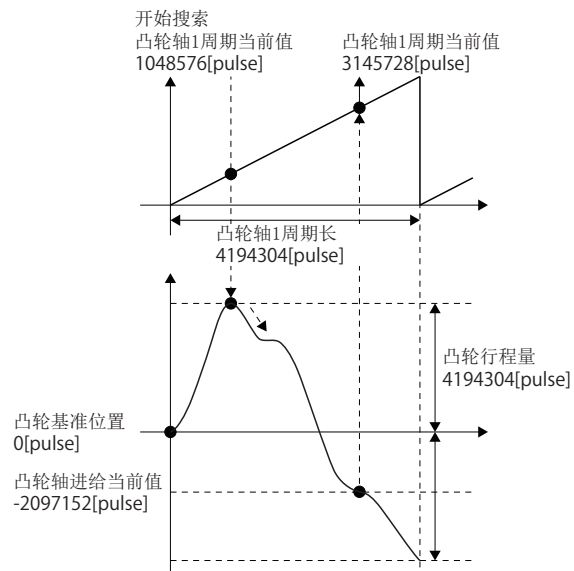
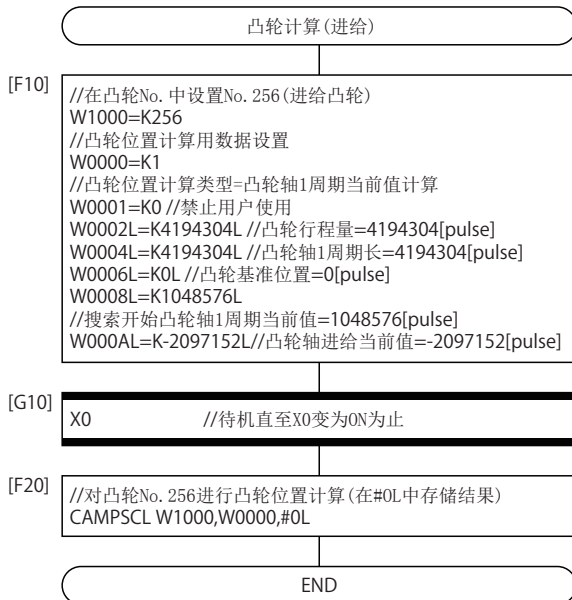
n 在直线的凸轮模式(凸轮No. 0)中进行凸轮轴进给当前值计算的程序



n 在往复动作的凸轮模式中进行凸轮轴进给当前值计算的程序



n 在进给动作的凸轮模式中进行凸轮轴1循环当前值计算的程序



4.17 视觉系统专用函数

线路打开: MVOPEN

格式	基本步数	可用步	
		F/FS	G
MVOPEN(S1), (S2)	8	○	○

设置数据

n 可用数据

○: 可以设置

设置数据	可用数据								计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数						
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)				
(S1)	—	○	—	—	○	—	—	—	—	—	
(S2)	—	○	—	—	○	—	—	—	—	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	登录的视觉系统(照相机)编号(1~32)	—
(S2)	至视觉系统中登录完成为止的超时时间(1~32767)[×10ms]	

功能

- 登录(连接)到(S1)中指定的视觉系统中。
- 运动SFC程序的执行不等待视觉系统的登录完成而转移到下一个块中。处理完成时,以太网通信线路设置参数中设置的状态存储软元件值将变为20(可受理)。
- (S2)以10ms单位进行设置。省略了设置的情况下,超时时间将变为10秒(与设置了1000的情况下相同)。

出错

以下情况下将发生运算出错。

- (S1)的数据超出1~8的范围时。
- (S2)的数据超出1~32767的范围时。
- 对已登录的视觉系统再次执行了MVOPEN时。
- 视觉系统参数的设置与视觉系统的设置不相同。
- 经过了指定的超时时间后仍然无法登录时。

程序示例

n 视觉系统(照相机)3的视觉系统中登录的程序

```
MVOPEN K3
```

视觉程序装载：MVLOAD

格式	基本步数	可用步	
		F/FS	G
MVLOAD (S1), (S2)	8	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S1)	—	○	—	—	○	—	—	—	—	—
(S2)	—	○	—	—	○	—	—	—	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	装载的视觉程序编号(1~128)	—
(S2)	至作业装载完成为止的超时时间(1~32767) [×10ms]	

功能

- 将(S1)中指定的视觉程序编号的作业装载到视觉系统中(将视觉系统中存储的作业文件展开到视觉系统上的存储器中，置为可执行(激活)作业)后，将视觉系统置为在线状态。
- 运动SFC程序的执行不等待上述处理的结束而转移到下一个块中。处理完成时，以太网通信线路设置参数中设置的状态存储软元件值将变为20(可受理)。作业的装载正常进行，视觉系统变为在线状态时，(S1)中指定的视觉程序动作设置参数中设置的状态存储软元件值将变为1(作业装载完成处于在线状态)。
- (S1)中指定的视觉程序编号的作业已被装载到视觉系统中的情况下，将强制进行作业的再装载。通过In-Sight[®] Explorer等更改了作业内容的情况下，更改内容将丢失，应事先进行作业保存。
- (S2)以10ms单位进行设置。省略了设置的情况下，超时时间将变为10秒(与设置了1000的情况下相同)。根据视觉系统中作业的内容处理时间将变化。对超时时间应根据视觉系统及作业内容进行设置。

出错

以下情况下将发生运算出错。

- (S1)的数据超出1~128的范围时。
- (S2)的数据超出1~32767的范围时。
- 未在(S1)中指定的视觉程序所使用的视觉系统中进行登录时。
- 视觉系统参数的设置与视觉系统及作业的设置不相同。
- 经过了指定的超时时间后作业的装载仍未完成时。

程序示例

n 对视觉程序编号2的作业进行装载的程序

```
MVLOAD K2
```


触发发行：MVTRG

格式	基本步数	可用步	
		F/FS	G
MVTRG (S1), (S2)	8	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S1)	—	○	—	—	○	—	—	—	—	
(S2)	—	○	—	—	○	—	—	—	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	进行触发发行的视觉系统(照相机)编号(1~32)	—
(S2)	从视觉系统接收执行结果为止的超时时间(1~32767) [×10ms]	

功能

- 对(S1)中指定的视觉系统发行触发，在视觉系统内执行作业，进行将该结果存储到视觉程序动作设置参数中设置的图形数据存储软元件中的一系列处理。
- 运动SFC程序的执行不等待上述处理的结束而转移到下一个块中。视觉系统内作业结束后，通过TCP/IP协议进行的图形数据(通过图形处理创建的各种数据)发送完成时，以太网通信线路设置参数中设置的状态存储软元件值将变为40(图形数据接收完成)。视觉系统参数中有读取数值的设置的情况下，数据将被接着存储到读取数值存储软元件中，以太网通信线路设置参数中设置的状态存储软元件值将变为50(数值单元接收完成)。
- (S2)以10ms单位进行设置。省略了设置的情况下，超时时间将变为10秒(与设置了1000的情况下相同)。

出错

以下情况下将发生运算出错。

- (S1)的数据超出1~32的范围时。
- (S2)的数据超出1~32767的范围时。
- 未在(S1)中指定的视觉系统中登录时。
- 视觉系统参数的设置与视觉系统的设置不相同。
- 读取数值单元中指定的标签或电子表单的数据不是整数时。
- 经过了指定的超时时间，一系列的处理仍未完成时。

程序示例

n 对视觉系统(照相机)1发行触发的程序

MVTRG K1

视觉程序启动：MVPST

格式	基本步数	可用步	
		F/FS	G
MVPST (S1), (S2)	8	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S1)	—	○	—	—	○	—	—	—	—	
(S2)	—	○	—	—	○	—	—	—	—	

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	启动的视觉程序编号(1~128)	—
(S2)	从视觉系统接收执行结果为止的超时时间(1~32767)[×10ms]	

功能

- 将(S1)中指定的视觉程序编号的作业装载到视觉系统中(将视觉系统内存的作业文件展开到视觉系统上的存储器中,置为可执行的(激活的)作业)并置为在线状态后,发行触发并执行作业,进行将该结果存储到视觉程序动作设置参数中设置的图形数据存储软元件中等一系列的处理。
- 运动SFC程序的执行不等待上述处理的结束而转移到下一个块中。视觉系统内作业结束,通过TCP/IP协议进行的图形数据(通过图形处理创建的各种数据)发送完成时,以太网通信线路设置参数中设置的状态存储软元件值将变为40(图形数据接收完成)。视觉系统参数中有读取数值的设置的情况下,数据将被接着存储到读取数值存储软元件中,以太网通信线路设置参数中设置的状态存储软元件值将变为50(数值单元接收完成)。
- (S1)中指定的视觉程序编号的作业已被安装在视觉系统中的情况下,不进行作业的再装载,只对视觉系统发行触发并执行作业,进行将该结果存储到视觉程序动作设置参数中设置的图形数据存储软元件中等一系列的处理。
- (S2)以10ms单位进行设置。省略了设置的情况下,超时时间将变为10秒(与设置了1000的情况下相同)。根据视觉系统中作业的内容处理时间将变化。对超时时间应根据视觉系统及作业内容进行设置。

出错

以下情况下将发生运算出错。

- (S1)的数据超出1~128的范围时。
- (S2)的数据超出1~32767的范围时。
- 未在(S1)中指定的视觉程序所使用的视觉系统中进行登录时。
- 视觉系统参数的设置与视觉系统及作业的设置不相同。
- 读取数值单元中指定的标签或电子表单的数据不是整数时。
- 经过了指定的超时时间,一系列的处理仍未完成时。

程序示例

n 执行视觉程序编号20的作业的程序

MVPST K20

数据导入：MVIN

格式	基本步数	可用步	
		F/FS	G
MVIN(S1), (S2), (D), (S3)	15以上	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据							计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数					
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S1)	—	○	—	—	○	—	—	—	—	—
(S2)	—	○*1	—	—	—	—	—	—	—	—
(D)		—	○*2	○*2	—	—	—	—	—	—
(S3)		○	—	—	○	—	—	—	—	—

*1 指定存储字符串数据的起始软元件。也可直接指定字符串。

*2 数据形式应与视觉系统中设置的作业形式一致。(形式不不同时，数据将被转换为(D)中指定的类型。)

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	读取数据的视觉系统(照相机)编号(1~32)	—
(S2)	读取数据的电子表单的单元格或标签	
(D)	存储读取的数据的软元件	
(S3)	从视觉系统中读取数据为止的超时时间(1~32767)[×10ms]	

功能

- 从(S1)中指定的视觉系统中，将(S2)中指定的电子表单的单元格或标签的数值存储到(D)中指定的软元件中。

要点

(S2)中指定的电子表单的单元格或标签的数据不是数值(字符串等)的情况下，将发生轻度出错(SFC)(出错代码：38F2H)，因此应使用MVCOM指令(☞ 239页 任意原生模式指令发送：MVCOM)。

- 运动SFC程序的执行不等待上述处理的结束而转移到下一个块中。处理完成时，以太网通信线路设置参数中设置的状态存储软元件值将变为20(可受理)。
- 在(S2)中将电子表单的单元格或标签作为半角32字符以内的字符串用“”围住后进行直接记述，或指定存储了半角32字符以内的字符串的软元件的起始。字符串的指定方法如下所示。

指定方法	内容
通过单元格指定	对电子表单的列(A~Z)及行(0~399)进行排列记述。 (例) 单元格为A5时设置“A5”
通过标签指定	对符号标签名原样不变地进行记述。 (例) 标签为Job.Pass_count时设置“Job.Pass_count”

- 从视觉系统中读取的数值按以下形式被存储。

电子表单的单元格，或标签的数值数据形式	(D)中存储的数据的形式	使用点数
整数	32位整数型	连续2点
浮点	64位浮点型	连续4点

- (S3)以10ms单位进行设置。省略了设置的情况下，超时时间将变为10秒(与设置了1000的情况下相同)。

出错

以下情况下将发生运算出错。

- (S1)的数据超出1~32的范围时。
- (S2)中指定的电子表单的单元格或标签的字符串长超出1~32字节的范围时
- (S2)中指定的电子表单的单元格或标签不存在时。
- (S2)中指定的电子表单的单元格或标签的数据不是数值时。
- (S3)的数据超出1~32767的范围时。
- 未在(S1)中指定的视觉系统中登录时。
- 视觉系统参数的设置与视觉系统的设置不相同。
- 经过了指定的超时时间，数据的读取仍未完成时。

程序示例

n将视觉系统(照相机)1的标签“模式_1. 夹具. 得分”中存储的数值存储到D3000以后的程序

```
MVIN K1,"模式_1. 夹具. 得分",D3000F
```

n对视觉系统(照相机)3，将D100以后存储的字符串的标签中存储的数值存储到D2000以后的程序

```
MVIN K3,D100,D2000L
```

数据导出: MVOUT

格式	基本步数	可用步	
		F/FS	G
MVOUT (S1), (S2), (S3), (S4)	15以上	○	○

设置数据

n 可用数据

○: 可以设置

设置数据	可用数据								计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数						
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)				
(S1)	—	○	—	—	○	—	—	—	—	—	
(S2)	—	○*1	—	—	—	—	—	—	—	—	
(S3)	—	○*2	○*2	○*2	○*2	○*2	○*2	—	—	—	
(S4)	—	○	—	—	○	—	—	—	—	—	

*1 指定存储字符串数据的起始软元件。也可直接指定字符串。

*2 数据形式应与传送数据的形式一致。也可直接指定字符串。

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	进行数据传送的视觉系统(照相机)编号(1~32)	—
(S2)	进行数据传送的电子表单的单元格或标签	
(S3)	传送的数据	
(S4)	将数据传送至视觉系统中为止的超时时间(1~32767)[×10ms]	

功能

- 将(S3)中指定的数据传送到(S1)中指定的视觉系统的(S2)中指定的电子表单的单元格或标签中。
- 运动SFC程序的执行不等待上述处理的结束而转移到下一个块中。处理完成时,以太网通信线路设置参数中设置的状态存储软元件值将变为20(可受理)。
- 在(S2)中将电子表单的单元格或标签作为半角32字符以内的字符串用“”围住后进行直接记述,或指定存储了半角32字符以内的字符串的软元件的起始。字符串的指定方法入下所示。

指定方法	内容
通过单元格指定	对电子表单的列(A~Z)及行(0~399)进行排列记述。 (例) 单元格为A5时设置“A5”
通过标签指定	对符号标签名原样不变地进行记述。 (例) 标签为Job.Pass_count时设置“Job.Pass_count”

- (S3)中指定存储了传送到电子表单的单元格或标签中的数据的软元件的起始。此外,也可直接指定常数或半角99字符以内的字符串(应使用“”围住进行记述)。

[间接指定]

(S3)中指定的数据的形式	使用点数	指定示例
16位整数型	1点	D1000
32位整数型	连续2点	D2000L
64位浮点型	连续4点	D3000F

[直接指定]

(S3)中指定的常数的形式	指定示例
16位整数型	K12345
32位整数型	K12345678L
64位浮点型	K1234.5
字符串	“MITSUBISHI”

要点

将浮点数据传送至视觉系统中时，将被作为32位浮点数据处理。有效位数约为7位，因此数据的第7位以后有可能与(S3)的数据不一致。

- (S4)以10ms单位进行设置。省略了设置的情况下，超时时间将变为10秒(与设置了1000的情况下相同)。

出错

以下情况下将发生运算出错。

- (S1)的数据超出1~32的范围时。
- (S2)中指定的电子表单的单元格或标签的字符串长超出1~32字节的范围时。
- (S2)中指定的电子表单的单元格或标签不存在时。
- (S2)中指定的电子表单的单元格或标签的数据类型与(S3)中指定的数据形式不相同。
- (S3)中指定的数据超出范围时。
- (S4)的数据超出1~32767的范围时。
- 未在(S1)中指定的视觉系统中登录时。
- 视觉系统参数的设置与视觉系统的设置不同时。
- 经过了指定的超时时间，数据的读取仍未完成时。

程序示例

n 将D3000F中存储的浮点值传送至视觉系统(照相机)1的标签“校准_1.世界点0.X”中的程序

```
MVOUT K1,"校准_1.世界点0.X",D3000F
```

状态存储软元件复位：MVFIN

格式	基本步数	可用步	
		F/FS	G
MVFIN(S)	6	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	—	○	—	—	○	—	—	—	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	对状态存储软元件进行复位的视觉系统(照相机)编号(1~32)	—

功能

- 将(S)中指定的以太网通信线路设置参数中设置的状态存储软元件值设置为20(可受理)。
- 对视觉系统发行触发的情况下，需要预先通过MVFIN指令复位状态存储软元件，以便能检测出触发的处理的完成。

出错

以下情况下将发生运算出错。

- (S)的数据超出1~32的范围时。
- 未在(S)中指定的视觉系统中进行登录时。
- 视觉系统参数的设置与视觉系统的设置不相同。
- 以太网通信线路设置参数中设置的状态存储软元件值不是20(可受理)、40(图形数据接收完成)、50(数值单元接收完成)时。

程序示例

n 对视觉系统(照相机)1的状态存储软元件进行复位的程序

```
MVFIN K1
```

线路关闭：MVCLOSE

格式	基本步数	可用步	
		F/FS	G
MVCLOSE (S)	6	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	—	○	—	—	○	—	—	—	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	进行注销的视觉系统(照相机)编号(1~32)	—

功能

- 从(S)中指定的视觉系统中注销(断开)。以太网通信线路设置参数中设置的状态存储软元件值将变为0(未连接)。
- 对未登录(连接)的视觉系统执行MVCLOSE指令时，将变为无处理。

出错

以下情况下将发生运算出错。

- (S)的数据超出1~32的范围时。
- 视觉系统参数的设置与视觉系统的设置不相同。

程序示例

n 从视觉系统(照相机)1的视觉系统中注销的程序

```
MVCLOSE K1
```


任意原生模式指令发送：MVCOM

格式	基本步数	可用步	
		F/FS	G
MVCOM(S1), (S2), (D), (S3), (S4)	19以上	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S1)	—	○	—	—	○	—	—	—	—	—
(S2)	—	○*1	—	—	—	—	—	—	—	—
(D)	—	○	—	—	—	—	—	—	—	—
(S3)	—	○	—	—	○	—	—	—	—	—
(S4)	—	○	—	—	○	—	—	—	—	—

*1 指定存储字符串数据的起始软元件。也可直接指定字符串。

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	进行原生模式指令发送的视觉系统(照相机)编号(1~32)	—
(S2)	原生模式指令字符串	
(D)	返回值存储起始软元件	
(S3)	返回值转换模式设置	
(S4)	从视觉系统中读取数据为止的超时时间(1~32767)[×10ms]	

功能

- 将(S2)中指定的原生模式指令发送到(S1)中指定的视觉系统中，将该返回值以(S3)中指定的形式存储到(D)中指定的软元件中。
- 运动SFC程序的执行不等待原生模式指令的结束，将转移到下一个块中。处理完成时，以太网通信线路设置参数中设置的状态存储软元件值将变为20(可受理)。
- 关于(S2)中指定的原生模式指令的详细内容，请参阅康耐视(COGNEX)的手册·帮助等。对于原生模式指令，在(S2)中以半角99字符以内的字符串用“”围住进行直接记述，或者指定存储了半角191字符以内的字符串的软元件的起始。

- 对于原生模式指令的返回值，根据(S3)的指定按以下方式被存储到(D)中指定的软元件中。返回值的数据为以下的情况下 ([CR]表示复位代码，[LF]表示换行代码。)



n 指定0 (ASCII模式) 时

(S3)中指定了0 (ASCII模式) 时，从(D)中指定的软元件开始按以下顺序存储数据。

1. 状态代码 (16位整数形式)

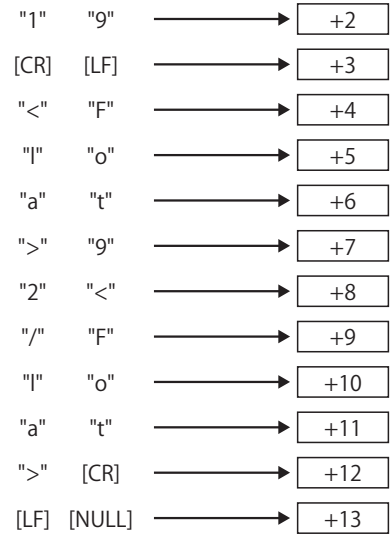


2. 下述3. 数据的字节数 (16位整数形式)



3. 数据部分的字符串 (ASCII代码)

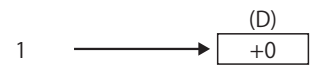
(数据的最后将存储结束[NULL]代码。)



n 指定1 (二进制模式) 时

(S3)中指定了1 (二进制模式) 时，从(D)中指定的软元件开始按以下顺序存储数据。

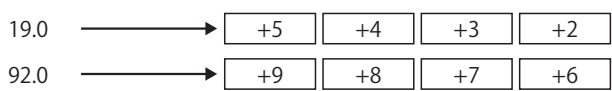
1. 状态代码 (16位整数形式)



2. 下述3. 数据的字节数 (16位整数形式)



3. 转换为64位浮点型的值的数据



- (S4)以10ms单位进行设置。省略了设置的情况下，超时时间将变为10秒(与设置了1000的情况下相同)。

出错

以下情况下将发生运算出错。

- (S1)的数据超出1~32的范围时。
- (S2)中指定的原生模式指令有错误时。
- (S3)的数据超出0~1的范围时。
- (S4)的数据超出1~32767的范围时。
- 未在(S1)中指定的视觉系统中登录时。
- (S2)中指定的原生模式的字符串超出字符数范围时。
- (S3)中指定了1(二进制模式)时返回值数据不是数值时。
- 视觉系统参数的设置与视觉系统的设置不不同时。
- 经过了指定的超时时间，原生模式指令返回值的软元件存储仍未完成时。
- 原生模式指令的返回值超出(D)~软元件末尾的范围时。(在存储了至软元件末尾为止的数据的时刻将发生运算出错。)

程序示例

n 向视觉系统(照相机)1发送原生模式指令“EV GetCellValue(“距离_1.最大”)”后，将该返回值以二进制模式存储到#0以后的程序

```
MVCOM K1,"EV GetCellValue("距离_1.最大"),#0,K1
```

4.18 附加专用函数

附加组件调用：MCFUN

格式	基本步数	可用步	
		F/FS	G
MCFUN(S1), (S2), (D1), (D2)	11以上	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S1)	—	○*1	—	—	—	—	—	—	—	
(S2)	—	○	—	—	—	—	—	—	—	
(D1)	—	○	—	—	—	—	—	—	—	
(D2)	○	○	—	—	—	—	—	—	—	

*1 指定存储字符串数据的起始软元件。也可直接指定字符串。

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S1)	附加组件名	—
(S2)	存储附加组件的输入值的起始软元件	
(D1)	存储附加组件的输出值的起始软元件	
(D2)	附加组件的执行完成标志	

功能

- 调用(S1)中指定的附加组件，通过(S2)中指定的软元件传递输入值，通过(D1)中指定的软元件接受输出值。通过附加组件的执行完成，(D2)中指定的执行完成标志将变为ON。
- 执行完成标志的复位应通过用户程序进行。
- 关于(S1)中指定的附加组件名，请参阅所安装的附加库的使用说明书。对于附加组件名，在(S1)中作为半角31字符以内的字符串用“”围住后进行直接记述，或指定存储了半角31字符以内的字符串的软元件的起始。
- 对于(S2)、(D1)中指定的软元件编号应指定偶数编号。
- (S2)、(D1)、(D2)可以省略。省略情况下的记述示例如下所示。

内容	记述示例
不省略的情况下	MCFUN “AddonFunc1”, D5000, D5100, M0
仅省略(D2)的情况下	MCFUN “AddonFunc1”, D5000, D5100,
省略(S2)、(D1)的情况下	MCFUN “AddonFunc1” , , M0
省略(S2)、(D1)、(D2)的情况下	MCFUN “AddonFunc1” , , ,

要点

- 对于省略了(S2)、(D1)情况下的动作，根据调用的附加组件而有所不同。
- 指定了(D2)的情况下，运动SFC程序的执行不等待附加组件的结束而转移到下一个块。附加组件的处理是在后台执行。可后台执行的附加组件为1个。在不等待(D2)的ON而连续执行了后台执行的附加组件的情况下将出错。
- 省略了(D2)的情况下，运动SFC程序的执行等待附加组件结束之后，转移至下一个块。

- 关于(S2)、(D1)、(D2)中指定的软元件的详细内容，请参阅所安装的附加库的使用说明书。

出错

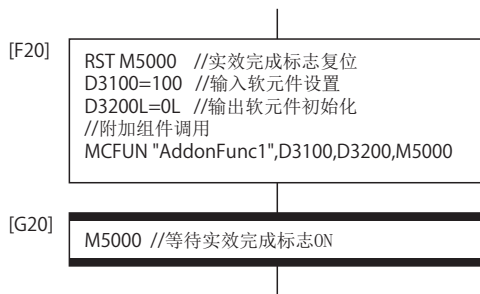
以下情况下将发生运算出错。

- (S1)中指定的附加组件未被登录时。
- (S1)中指定的字符串超过31字符时。
- (S2)、(D1)不是偶数编号的软元件时。
- (S1)、(S2)、(D1)、(D2)为间接指定软元件，软元件编号超出范围时。
- 在附加组件执行完成之前，再次执行了附加组件时。

程序示例

n 使用D3100的数据调用附加组件“AddonFunc1”，将处理结果存储到D3200中的程序

```
MCFUN "AddonFunc1",D3100,D3200,M5000
```



4.19 其它指令

事件任务允许：EI

格式	基本步数	可用步	
		F/FS	G
EI	1	○	○

设置数据

没有设置数据。

功能

- 允许事件任务的执行。
- 只能在普通任务中使用。

出错

以下情况下将发生运算出错。

- 在普通任务以外使用时。

程序示例

n 允许事件任务的执行。

```
EI
```

事件任务禁止：DI

格式	基本步数	可用步	
		F/FS	G
DI	1	○	○

设置数据

没有设置数据。

功能

- 禁止事件任务的执行。
- 执行DI指令后，发生了外部中断或可编程控制器中断的情况下，在执行了EI指令的时刻对应的事件任务将被执行1次。（即使DI中发生了多次外部中断或可编程控制器中断的情况下，在执行EI指令的时刻对应的事件任务也仅执行1次。）
- DI中不能执行恒定周期事件任务。
- 不能禁止NMI任务的执行。
- 多CPU系统电源投入时或进行了复位的情况下，将变为DI状态。根据“[Rq. 1120]可编程控制器就绪标志(R: M30000/Q: M2000)”的ON/OFF，EI/DI的状态不变化。

出错

以下情况下将发生运算出错。

- 在普通任务以外使用时。

程序示例

n 禁止事件任务的执行的程序

```
DI
```

无处理：NOP

格式	基本步数	可用步	
		F/FS	G
NOP	1	○	○

设置数据

没有设置数据。

功能

在无处理的指令中，对至目前为止的运算不带来任何影响。

出错

没有运算出错。

块传送：BMOV

格式	基本步数	可用步	
		F/FS	G
BMOV (D), (S), (n)	12	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据							计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数					
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(D)	○	○	—	—	—	○	—	—	—	—
(S)	○	○	—	—	—	○	—	—	—	—
(n)	—	○	○	—	○	○	—	—	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(D)	传送目标软元件的起始编号	—
(S)	传送源软元件的起始编号	
(n)	传送字数(1~1000000000)	

功能

- 将(S)中指定的软元件开始的n字的内容，进行(D)中指定的软元件开始的n字批量传送。
- 传送源与传送目标软元件重复的情况下也可进行传送。向软元件编号小的一方传送的情况下从(S)开始进行传送，向软元件编号大的一方传送的情况下从(S)+(n-1)开始进行传送。
- 该指令的处理时间与传送字数(n)成正比，因此为了避免妨碍运动运算的执行，应参考运算处理时间，对执行任务、传送字数(n)进行调整。
- (D)及(S)中设置的传送目标软元件与传送源软元件的组合如下所示。

○：可以组合；×：不能组合

传送源软元件(S)		传送目标软元件(D)						
		用户软元件/系统软元件	CPU缓冲存储器访问软元件		CPU缓冲存储器访问软元件(恒定周期区域)		模块访问软元件	
			本机	其它机号	本机	其它机号	本机管理	其它机号管理
用户软元件/系统软元件		○	○	×	○	×*1	○	×
CPU缓冲存储器访问软元件	本机	○	○	×	○	×*1	○	×
	其它机号	○	○	×	○	×*1	×	×
CPU缓冲存储器访问软元件(恒定周期区域)	本机	○	○	×	○	×*1	○	×
	其它机号	○	○	×	○	×*1	○	×
模块访问软元件	本机管理	○	○	×	○	×*1	×	×
	其它机号管理	○	○	×	○	×*1	×	×

*1 不能反映写入数据，但也不发生出错。

出错

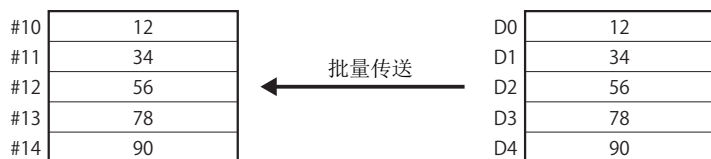
以下情况下将发生运算出错。

- (S)至(S)+(n-1)超出软元件范围时。
- (D)至(D)+(n-1)超出软元件范围时。
- (n)为0或负数时。
- (S)与(D)的组合不正确时。
- (S)及(D)不是本机的CPU缓冲存储器访问软元件或不是本机管理的模块的模块访问软元件时。
- (S)为位软元件，软元件编号不是16的倍数时。
- (D)为位软元件，软元件编号不是16的倍数时。

程序示例

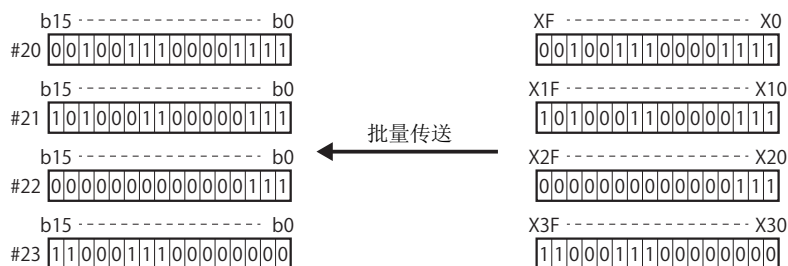
n 将D0开始的5字的内容批量传送到#10开始的5字的程序

BMOV #10,D0,K5



n 将X0开始的4字的内容批量传送到#20开始的4字的程序

BMOV #20,X0,K4



同一数据块传送：FMOV

格式	基本步数	可用步	
		F/FS	G
FMOV (D), (S), (n)	12	○	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(D)	○	○	—	—	—	○	—	—	—	—
(S)	○	○	—	—	○	—	—	—	—	—
(n)	—	○	○	—	○	○	—	—	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(D)	传送目标软元件的起始编号	—
(S)	传送数据或存储了传送数据的软元件的编号	
(n)	传送字数(1~1000000000)	

功能

- 将(S)中指定的数据或软元件内容向(D)中指定的软元件进行(n)字传送。
- 该指令的处理时间与传送字数(n)成正比，因此为了避免妨碍运动运算的执行，应参考运算处理时间，对执行任务、传送字数(n)进行调整。

出错

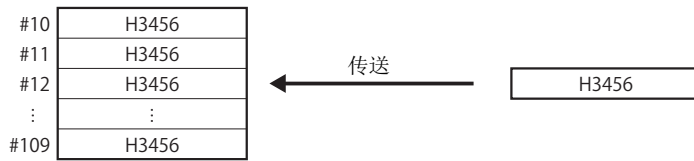
以下情况下将发生运算出错。

- (D)至(D)+(n-1)超出软元件范围时。
- (n)为0或负数时。
- (D)不是本机的CPU缓冲存储器访问软元件或不是本机管理的模块的模块访问软元件时。
- (S)为位软元件，软元件编号不是16的倍数时。
- (D)为位软元件，软元件编号不是16的倍数时。

程序示例

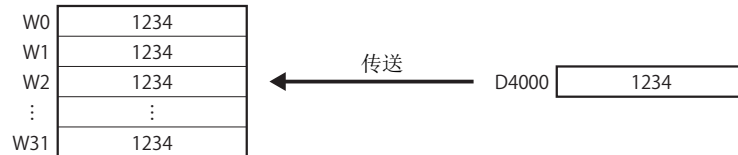
n 将从#10开始的100字全部设置为3456H的程序

```
FMOV #10,H3456,K100
```



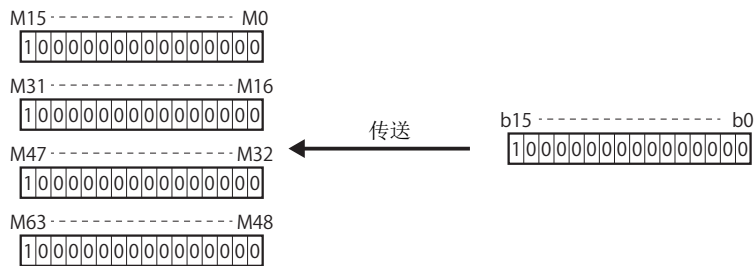
n 将D4000的内容设置到从W0开始的50字的程序

```
FMOV W0,D4000,K50
```



n 将从M0开始的4字全部设置为8000H的程序

```
FMOV M0,H8000,K4
```



至缓冲存储器的字数据写入：T0

格式	基本步数	可用步	
		F/FS	G
T0 (D1), (D2), (S), (n)	14	○	○

设置数据

n 可用数据

○：可以设置

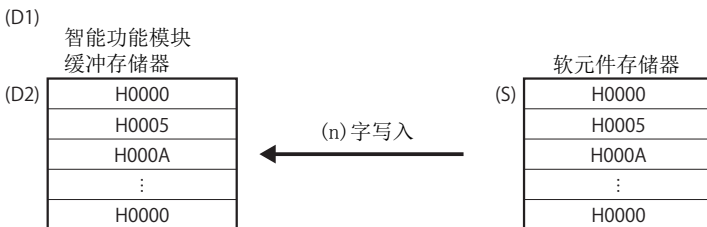
设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(D1)	—	○	—	—	○	—	—	—	—	—
(D2)	—	○	○	—	○	○	—	—	—	—
(S)	○	○	—	—	—	—	—	—	—	—
(n)	—	○	○	—	○	○	—	—	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(D1)	模块的起始输入输出编号(000H~FF0H, 3E00H~3E30H)	—
(D2)	写入数据的缓冲存储器的起始地址(0~对象模块的缓冲存储器末尾)	
(S)	存储了写入数据的起始软元件编号	
(n)	写入字数(1~对象模块的缓冲存储器范围内)	

功能

- 将(S)中指定的软元件开始的(n)字的数据，写入到(D1)中指定的模块内的缓冲存储器的(D2)中指定的地址以后。



要点

通过模块访问软元件(U□\G)，可以进行至缓冲存储器的字数据写入。

(例)

将从#0开始的2字写入到智能功能模块(起始输入输出编号：010H)的缓冲存储器地址0H中。

```
U1\G0L = #0L
```

关于模块访问软元件，请参阅以下手册。

📖 MELSEC iQ-R运动控制器编程手册(公共篇)

- 对于(D1)，指定在GX Works3的[系统参数]⇒[I/O分配设置]中设置的模块的起始输入输出编号。在下述系统配置中，对D/A转换模块(R64DA)执行了T0指令的情况下，(D1)将变为20H。

电源模块	R04CPU 起始输入输出编号 No.: 3E00H	R32MTCPU 起始输入输出编号 No.: 3E10H	RX40C7 起始输入输出编号 No.: 00H	R64AD 起始输入输出编号 No.: 10H	R64DA 起始输入输出编号 No.: 20H	
------	----------------------------------	------------------------------------	--------------------------------	-------------------------------	-------------------------------	--

- 关于缓冲存储器的起始地址(D2)及写入字数(n)中可指定的值的上限，根据对象模块而有所不同。关于详细内容，请参阅所使用的模块的手册。
- 该指令的处理时间与写入字数(n)成正比，因此为了避免妨碍运动运算的执行，应参考运算处理时间，对执行任务、写入字数(n)进行调整。
- 关于可作为运动CPU管理模块使用的智能功能模块，请参阅以下手册。

📖 MELSEC iQ-R运动控制器编程手册(公共篇)

- (S)及(D1)中设置的写入目标与写入源的组合如下所示。

○：可以组合；×：不能组合

写入源(S)		写入目标(D1)			
		CPU起始输入输出编号(3E00H~3E30H)		模块起始输入输出编号(000H~FF0H)	
		本机	其它机号	本机管理	其它机号管理
用户软元件/系统软元件		○	×	○	×
CPU缓冲存储器访问软元件	本机	○	×	○	×
	其它机号	○	×	×	×
CPU缓冲存储器访问软元件(恒定周期区域)	本机	○	×	○	×
	其它机号	○	×	○	×
模块访问软元件	本机管理	○	×	×	×
	其它机号管理	○	×	×	×

出错

以下情况下将发生运算出错。

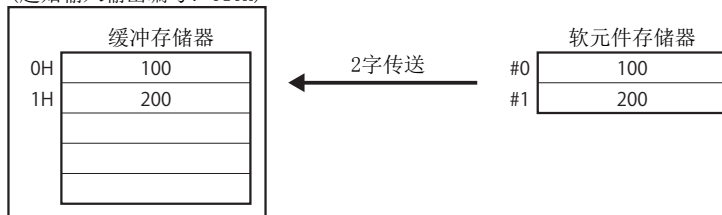
- 写入字数(n)超出范围时。
- 执行指令时无法与对象模块进行通信时。
- 执行指令时检测出对象模块异常时。
- (D2)中指定的地址超出缓冲存储器范围时。
- 存储了写入数据的起始软元件编号(S)+写入字数(n)超出软元件范围时。
- (S)为位软元件，软元件编号不是16的倍数时。
- (S)与(D1)的组合不正确时。

程序示例

n将从#0开始的2字写入到智能功能模块(起始输入输出编号：010H)的缓冲存储器地址0H中。

TO H010,H0,#0,K2

智能功能模块
(起始输入输出编号：010H)



从缓冲存储器中的字数据读取：FROM

格式	基本步数	可用步	
		F/FS	G
FROM(D), (S1), (S2), (n)	14	○	○

设置数据

n 可用数据

○：可以设置

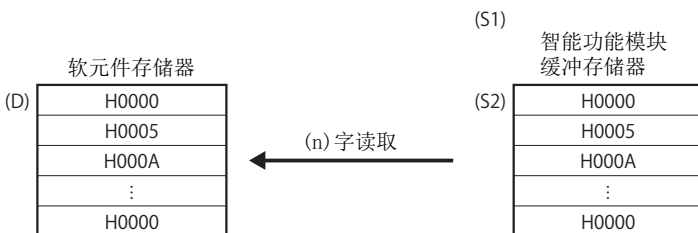
设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(D)	○	○	—	—	—	—	—	—	—	—
(S1)	—	○	—	—	○	—	—	—	—	—
(S2)	—	○	○	—	○	○	—	—	—	—
(n)	—	○	○	—	○	○	—	—	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(D)	存储读取的数据的起始软元件编号	—
(S1)	模块的起始输入输出编号(000H~FF0H, 3E00H~3E30H)	
(S2)	读取缓冲存储器的起始地址(0~对象模块的缓冲存储器末尾)	
(n)	读取字数(1~对象模块的缓冲存储器范围内)	

功能

- 从(S1)中指定的模块内的缓冲存储器的(S2)中指定的地址开始，读取(n)字的数据，写入到(D)中指定的软元件以后。



要点

通过模块访问软元件(U□\G)，可以进行至缓冲存储器的字数据写入。

(例)

从智能功能模块(起始输入输出编号：020H)的缓冲存储器地址10H中读取1字的数据后，存储到W0中。

```
W0 = U2\G16
```

关于模块访问软元件，请参阅以下手册。

📖 MELSEC iQ-R运动控制器编程手册(公共篇)

- 对于(S1)，指定在GX Works3的[系统参数]⇒[I/O分配设置]中设置的模块的起始输入输出编号。在下述系统配置中，对A/D转换模块(R64AD)执行FROM指令的情况下，(S1)将变为10H。

电源模块	R04CPU 起始输入输出编号 No.: 3E00H	R32MTCPU 起始输入输出编号 No.: 3E10H	RX40C7 起始输入输出编号 No.: 00H	R64AD 起始输入输出编号 No.: 10H	R64DA 起始输入输出编号 No.: 20H	
------	----------------------------------	------------------------------------	--------------------------------	-------------------------------	-------------------------------	--

- 关于缓冲存储器的起始地址(D2)及读取字数(n)中可指定的值的上限，根据对象模块而有所不同。关于详细内容，请参阅所使用的模块的手册。
- 该指令的处理时间与读取字数(n)成正比，因此为了避免妨碍运动运算的执行，应参考运算处理时间，对执行任务、读取字数(n)进行调整。
- 关于可作为运动CPU管理模块使用的智能功能模块，请参阅以下手册。
- (S1)与(D)中设置的读取源与读取目标的组合如下所示。

○：可以组合；×：不能组合

读取源(S1)	读取目标(D)						
	用户软元件/系统软元件	CPU缓冲存储器访问软元件		CPU缓冲存储器访问软元件(恒定周期区域)		模块访问软元件	
		本机	其它机号	本机	其它机号	本机管理	其它机号管理
用户软元件/系统软元件	○	○	×	○	×*1	○	×
CPU起始输入输出编号(3E00H~3E30H)	本机	○	×	○	×*1	○	×
	其它机号	○	×	○	×*1	×	×
模块起始输入输出编号(000H~FF0H)	本机管理	○	×	○	×*1	×	×
	其它机号管理	○	×	○	×*1	×	×

*1 不能反映读取数据，但也不发生出错。

出错

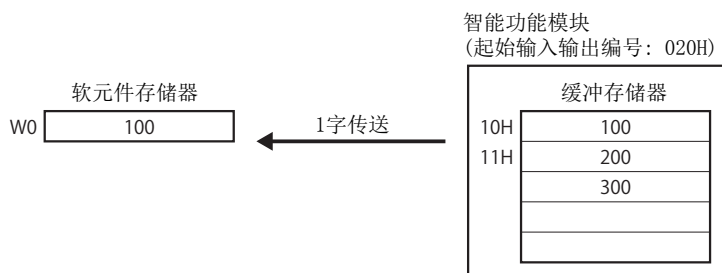
以下情况下将发生运算出错。

- 读取字数(n)超出范围时。
- 执行指令时无法与对象模块进行通信时。
- 执行指令时检测出对象模块异常时。
- (S2)中指定的地址超出缓冲存储器范围时。
- 存储读取数据的起始软元件编号(D)+读取字数(n)超出软元件范围时。
- (D)为位软元件，软元件编号不是16的倍数时。
- (S1)与(D)的组合不正确时。

程序示例

n从智能功能模块(起始输入输出编号：020H)的缓冲存储器地址10H中读取1字的数据后，存储到W0中。

```
FROM W0,H020,H10,K1
```



至对象站缓冲存储器的字数据写入：RTO

格式	基本步数	可用步	
		F/FS	G
RTO (D1), (D2), (D3), (S), (n), (D4)	21	○	○

设置数据

n 可用数据

○：可以设置

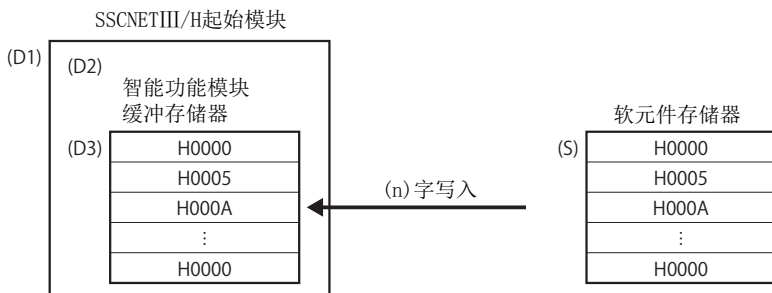
设置数据	可用数据							计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数					
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(D1)	—	○	—	—	○	—	—	—	—	—
(D2)	—	○	—	—	○	—	—	—	—	—
(D3)	—	○	—	—	○	—	—	—	—	—
(S)	○	○	—	—	—	—	—	—	—	—
(n)	—	○	—	—	○	—	—	—	—	—
(D4)	○	—	—	—	—	—	—	—	—	—

n 内容、结果的数据类型

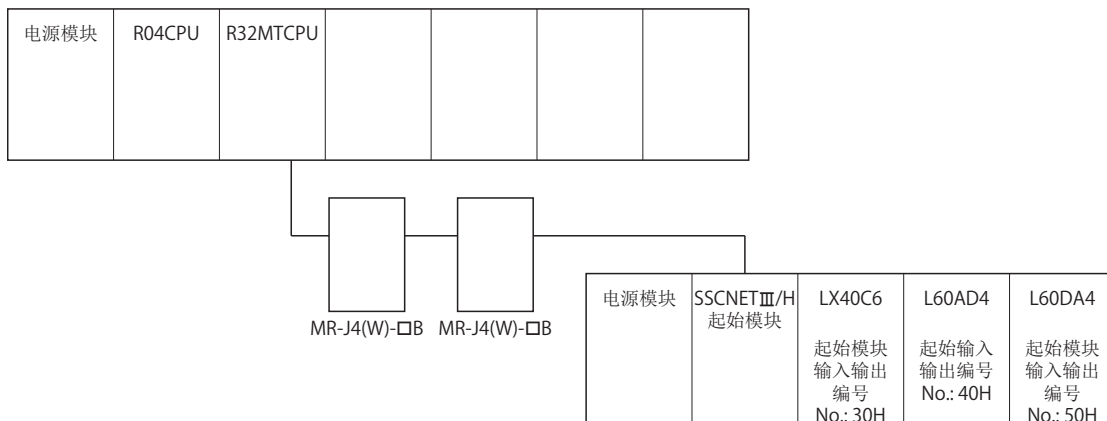
设置数据	内容	结果的数据类型
(D1)	对象SSCNETⅢ/H起始模块的RIO轴No. (601~608)	—
(D2)	写入数据的智能功能模块的起始输入输出编号 (00~FEH: 将输入输出编号以3位数表示时的高2位)	—
(D3)	写入数据的智能功能模块的缓冲存储器的起始地址	—
(S)	存储了写入数据的起始软元件编号	—
(n)	写入字数(1~32767)	—
(D4)	完成软元件 (D4+0): 通过写入完成使其ON的本机软元件 (D4+1): 通过写入异常完成使其ON的本机软元件(异常完成时, D4+0也变为ON)	—

功能

- 将(S)中指定的软元件开始的(n)字的数据，写入到(D1)中指定的对象SSCNETⅢ/H起始模块上安装的，(D2)中指定的智能功能模块内的缓冲存储器的(D3)中指定的地址以后。写入完成时，(D4)中指定的完成位软元件将变为ON。



- 对于(D2)，指定在[运动CPU通用参数]⇒[起始模块]中设置的SSCNETⅢ/H起始模块中安装的模块的起始输入输出编号。在下述系统配置中，对D/A转换模块(L60DA4)执行RT0指令的情况下，(D2)将变为05H。



- 该指令的处理时间与写入字数(n)成正比，因此为了避免妨碍运动运算的执行，应参考运算处理时间，对执行任务、写入字数(n)进行调整。
- 可使用的模块仅为下述模块。

模块名称	型号
模拟输入	L60AD4、L60AD4-2GH
模拟输出	L60DA4
高速计数器	LD62、LD62D

- 完成位软元件的复位应通过用户程序进行。
- 在执行RT0指令后完成位软元件变为ON之前，不能处理其它的RT0指令。在执行RT0指令后完成位软元件变为ON之前的期间，再次执行了RT0指令的情况下，后执行的RT0指令将发生出错。
- 对SSCNETⅢ/H起始模块写入3字以上的数据时，重复进行1次2字的数据写入。因此对于指定3字以上进行了写入的数据，不能在同一时机写入。对于需要以同一时机进行写入的数据，应使用SSCNETⅢ/H起始模块的软元件的循环传送(SSCNETⅢ/H起始模块及智能功能模块的缓冲存储器的刷新。关于详细内容，请参阅以下手册。

📖 MELSEC-L SSCNETⅢ/H起始模块用户手册

出错

以下情况下将发生运算出错。

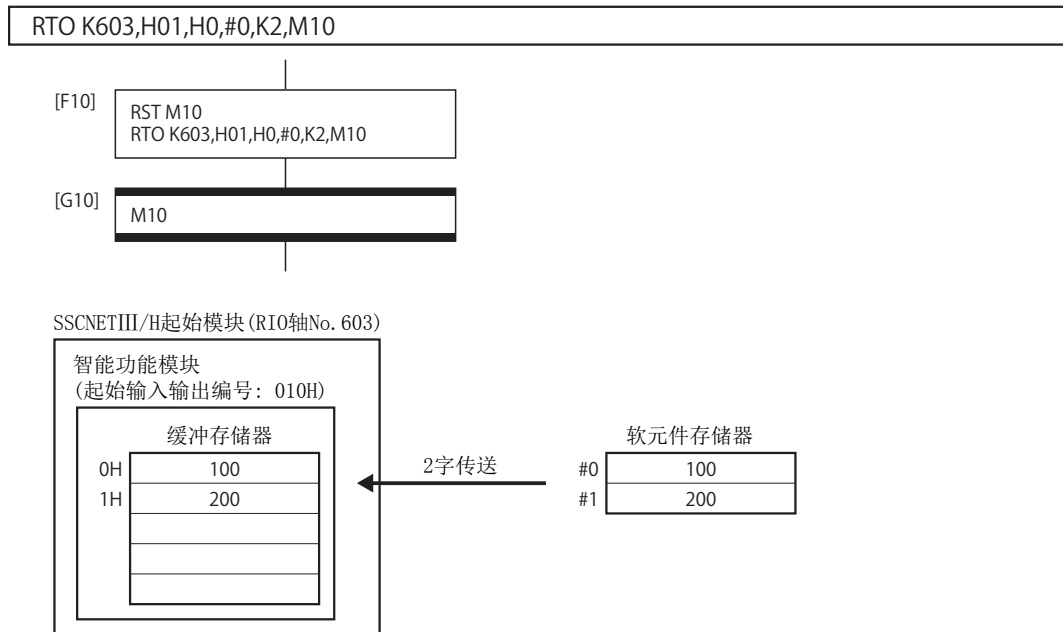
- 写入字数(n)超出范围时。
- (D1)中指定的对象SSCNETⅢ/H起始模块RIO轴编号超出601~608的范围时。
- 执行指令时未连接对象SSCNETⅢ/H起始模块时。
- 存储了写入数据的起始软元件编号(S)+写入字数(n)超出软元件范围时。
- (S)为位软元件，软元件编号不是16的倍数时。
- 在执行RT0指令后至完成位软元件变为ON为止期间，再次执行了RT0指令时。

以下情况下完成软元件的异常完成(D4+1)将变为ON。

- 执行指令时检测出对象SSCNETⅢ/H起始模块的异常时。
- (D2)中指定的智能功能模块的起始输入输出编号不是智能功能模块时。
- (D3)中指定的智能功能模块的缓冲存储器的起始地址超出缓冲存储器范围时。

程序示例

n 将#0开始的2字写入到SSCNETⅢ/H起始模块RIO轴603的智能功能模块(起始输入输出编号: 010H)的缓冲存储器地址0H中。



从对象站缓冲存储器中的字数据读取：RFROM

格式	基本步数	可用步	
		F/FS	G
RFROM (D), (S1), (S2), (S3), (n), (D1)	21	○	○

设置数据

n 可用数据

○：可以设置

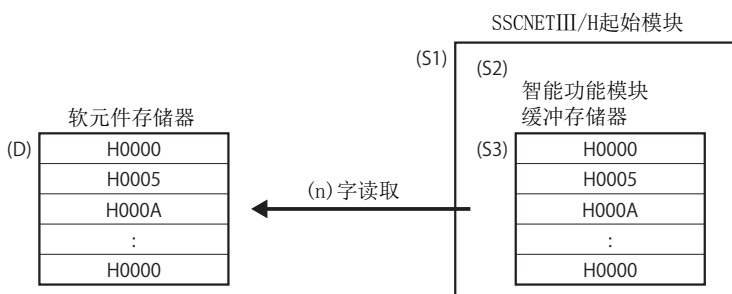
设置数据	可用数据								计算公式	位条件表达式	比较条件表达式
	位软元件	字软元件			常数						
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)				
(D)	○	○	—	—	—	—	—	—	—	—	
(S1)	—	○	—	—	○	—	—	—	—	—	
(S2)	—	○	—	—	○	—	—	—	—	—	
(S3)	—	○	—	—	○	—	—	—	—	—	
(n)	—	○	—	—	○	—	—	—	—	—	
(D1)	○	—	—	—	—	—	—	—	—	—	

n 内容、结果的数据类型

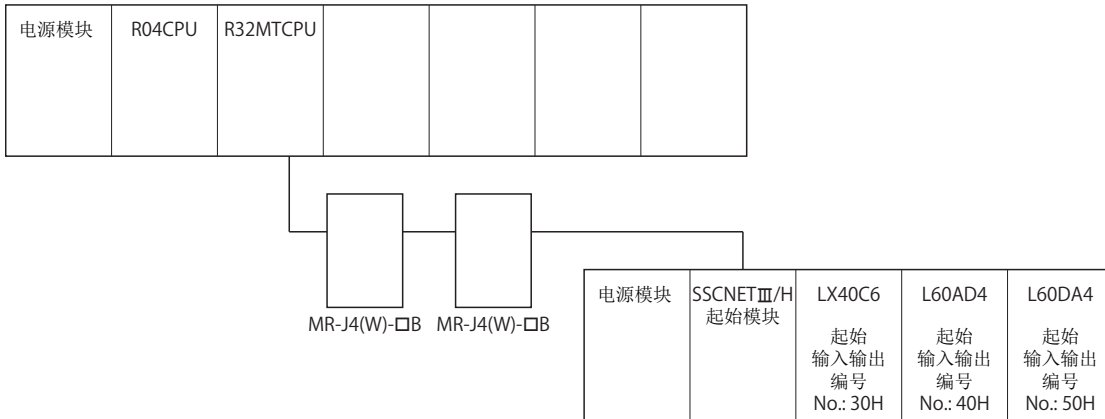
设置数据	内容	结果的数据类型
(D)	存储读取的数据的起始软元件编号	—
(S1)	对象SSCNETⅢ/H起始模块的RIO轴No. (601~608)	
(S2)	存储了读取数据的智能功能模块的起始输入输出编号 (00~FEH: 将输入输出编号以3位数表示时的高2位)	
(S3)	存储了读取数据的智能功能模块的缓冲存储器的起始地址	
(n)	读取字数 (1~32767)	
(D1)	完成软元件 (D1+0): 通过读取完成使其ON的本机软元件 (D1+1): 通过读取异常完成使其ON的本机软元件 (异常完成时, D1+0也变为ON)	

功能

- 从(S1)中指定的对象SSCNETⅢ/H起始模块中安装的, (S2)中指定的智能功能模块内的缓冲存储器的(S3)中指定的地址开始, 读取(n)字的数据, 写入到(D)中指定的软元件以后。



- 对于(S2)，指定在[运动CPU通用参数]⇒[起始模块]中设置的SSCNETⅢ/H起始模块中安装的模块的起始输入输出编号。在下述系统配置中，对A/D转换模块(L60AD4)执行RFROM指令的情况下，(S2)将变为04H。



- 该指令的处理时间与读取字数(n)成正比，因此为了避免妨碍运动运算的执行，应参考运算处理时间，对执行任务、读取字数(n)进行调整。
- 可使用的模块仅为下述模块。

模块名称	型号
模拟输入	L60AD4、L60AD4-2GH
模拟输出	L60DA4
高速计数器	LD62、LD62D

- 完成位软元件的复位应通过用户程序进行。
- 在执行RFROM指令后至完成位软元件变为ON之前，不能处理其它的RFROM指令。在执行RFROM指令后至完成位软元件变为ON为止期间，再次执行了RFROM指令的情况下，后执行的RFROM指令将发生出错。
- 从SSCNETⅢ/H起始模块中读取5字以上的数据时，重复进行1次4字的数据读取。因此，指定5字以上读取的数据不是同一时刻的数据。对于需要以同一时刻进行读取的数据，应使用SSCNETⅢ/H起始模块的字软元件的循环传送(SSCNETⅢ/H起始模块及智能功能模块的缓冲存储器)的刷新。关于详细内容，请参阅以下手册。

📖 MELSEC-L SSCNETⅢ/H起始模块用户手册

出错

以下情况下将发生运算出错。

- 读取字数(n)超出范围时。
- (S1)中指定的对象SSCNETⅢ/H起始模块RIO轴编号超出601~608的范围时。
- 执行指令时未连接对象SSCNETⅢ/H起始模块时。
- 存储读取数据的起始软元件编号(D)+读取字数(n)超出软元件范围时。
- (D)为位软元件，软元件编号不是16的倍数时。
- 在执行RFROM指令后至完成位软元件变为ON为止的期间，再次执行了RFROM指令时。

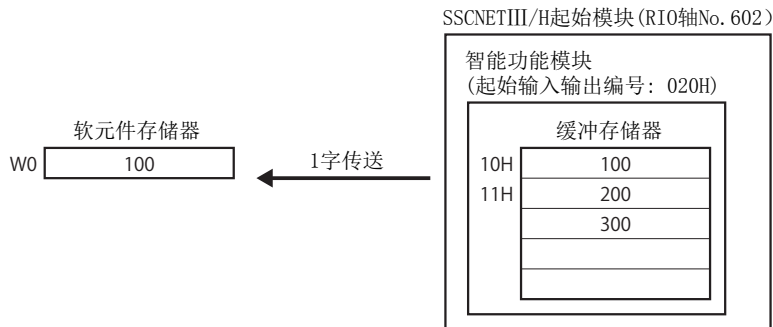
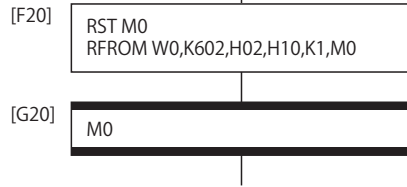
以下情况下完成软元件的异常完成(D1+1)将变为ON。

- 执行指令时检测出对象SSCNETⅢ/H起始模块的异常时。
- (S2)中指定的智能功能模块的起始输入输出编号不是智能功能模块时。
- (S3)中指定的智能功能模块的缓冲存储器的起始地址超出缓冲存储器范围时。

程序示例

n 从SSCNETⅢ/H起始模块RIO轴602的智能功能模块(起始输入输出编号: 020H)的缓冲存储器地址10H中读取1字的数据后, 存储到W0中。

RFROM W0,K602,H02,H10,K1,M0



时间等待：TIME

格式	基本步数	可用步	
		F/FS	G
TIME (S)	8	—	○

设置数据

n 可用数据

○：可以设置

设置数据	可用数据									
	位软元件	字软元件			常数			计算公式	位条件表达式	比较条件表达式
		16位整数型	32位整数型(L)	64位浮点型(F)	16位整数型(K/H)	32位整数型(K/H、L)	64位浮点型(K)			
(S)	—	○	○	—	○	○	—	—	—	—

n 内容、结果的数据类型

设置数据	内容	结果的数据类型
(S)	等待时间(0~2147483647) [ms]	逻辑型(真/假)

功能

- 变为(S)中指定的时间等待状态。经过时间小于设置时间时，结果将变为假，经过设置时间以上将变为真而进行转移。
- 将(S)以16位整数型字软元件指定后，有可能指定32768~65535[ms]的情况下，应通过ULONG进行无符号32位整数值转换。

出错

以下情况下将发生运算出错。

- (S)为间接指定软元件，软元件编号超出范围时。
- (S)中指定的数据(间接指定时为软元件数据)超出0~2147483647的范围时。

程序示例

n 60秒等待的程序(指定常数时)

```
TIME K60000
```

n 在16位的整数型间接指定(#0)中，有等待32768~65535[ms]情况下的程序

```
TIME ULONG(#0)
```

n 经过了指定时间以上，对位软元件进行SET (RST)后转移的程序

```
SET M100 = TIME K60000
```

要点

- 通过字软元件间接指定了等待时间设置的情况下，将以首次获取的软元件值进行控制。等待时间状态中即使更改软元件值，也无法更改设置时间。
- TIME指令相当于条件表达式，因此只能设置到转换(G)程序的最终行。
- 将设置了TIME指令的同一编号的转换程序(Gn)用于多个运动SFC程序的情况下，应避免同时执行(如果同时执行，先执行的一方的等待时间将变为不正确)。
- 如果是其它编号的转换程序(Gn)，可以在多个运动SFC程序中同时执行TIME指令(同时激活步最多为256以内)。
- 在通过TIME指令进行的时间等待中，不能对时间等待进行中断。

4.20 注释文: //

格式	基本步数	可用步	
		F/FS	G
//	—	○	○

设置数据

没有设置数据。

功能

//以后，块结束为止的字符串作为注释。

出错

没有运算出错。

程序示例

n 对代入程序附加了注释的示例

```
D0 = D1 // 将D0的值(16位整数数据)代入到D1中。
```

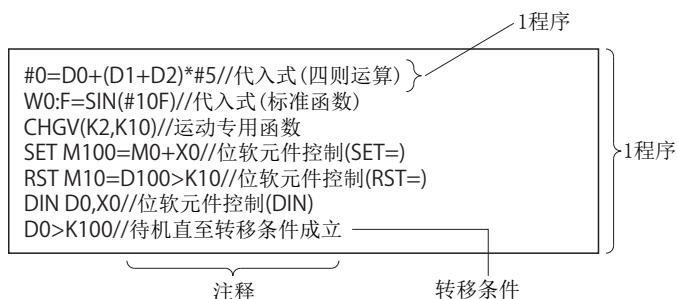

5 转换程序

5.1 转换程序

转换程序

- 在转换程序中可以设置代入运算公式、运动专用函数、位软元件控制指令、转移条件。
- 1个转换程序中，可以设置多个块。
- 1个转换程序中可设置的块数无限制。但是，1个程序应在128k字节以内。
- 转换程序以半角字符进行设置。注释中全角字符也可使用。
- 1块最大字符数为半角字符1020字符。1个全角字符相当于半角字符2字符。
- 转换程序的最终块中必须设置转移条件。转换程序在转移条件成立之前重复执行程序，通过转移条件成立转移至下一个步。转移条件只能设置在最终块中。
- 作为转换程序的特殊情况，可以创建仅将无处理(NOP)设置为1个块的程序。该程序用于通过伺服程序的动作完成进入下一个步的情况下，不作为互锁条件设置时使用。(☞ 98页 希望通过动作完成进入下一个步的情况下)

转换程序示例如下所示。



可作为转移条件设置到最终块中的为返回逻辑数据值(真/假)的位条件表达式、比较条件表达式以及软元件设置(SET=)/软元件复位(RST=)。软元件设置(SET=)/软元件复位(RST=)的情况下，(S)中指定的位条件表达式、比较条件表达式的真/假将成为转移条件，根据转移条件成立执行软元件设置/复位，转移至下一个步。

转移条件的记述示例如下所示。

分类	记述示例
位条件表达式	M0
	!M0+X10*M100
条件表达式	(D0>K100)+(D100L!=K20L)
软元件设置(SET=)	SET Y0=M100
软元件复位(RST=)	RST M10=D0=K100

-
- 转换程序与运算控制程序的区别在于在最终块中设置转移条件。除此以外的设置内容与运算控制程序的相同。
 - 将软元件设置 (SET=) / 软元件复位 (RST=) 作为转移条件设置到最终块中的情况下，不能省略 (S) 中指定的位条件表达式、比较条件表达式。
 - 位条件表达式、比较条件表达式只能设置到最终块中。软元件设置 (SET=) / 软元件复位 (RST=) 可以设置到最终块以外。
-

6 运动SFC的动作及参数

6.1 任务的类型

通过程序参数对各程序设置以哪个时机执行运动SFC程序的处理。按照处理时机进行大致分类，有如下表所示的3种类型。

任务类型	内容
普通任务	通过运动CPU的主周期(空余时间)执行。
事件任务	<ul style="list-style-type: none">通过恒定周期(0.222ms、0.444ms、0.888ms、1.777ms、3.555ms、7.111ms、14.222ms)执行。在外部中断(中断指针(I0~I15)的16点)中，通过事件任务原因中设置的输入的ON执行。通过来自于可编程控制器的中断执行。
NMI任务	在外部中断(中断指针(I0~I15)的16点)中，通过NMI任务原因中设置的输入的ON执行。

要点

- 恒定周期事件任务的动作与运算周期设置无关。
(例)
运算周期设置中设置了0.888ms的情况下，仍然执行0.222ms的恒定周期事件任务。
- 对于运动CPU管理的输入模块，在通过[R系列通用参数]⇒[模块配置一览]⇒“设置项目”⇒“详细”按钮显示的模块详细设置中设置中断指针(I0~I15)。

6.2 连续转移数及任务动作

连续转移数

运动SFC程序的执行控制时，在各个任务的执行周期中，将“激活步的执行→下一个转换条件的判定→条件成立时的转移处理(激活步的转移)”作为基本1动作，按照相应任务的激活步数执行该动作后，结束1次处理。然后，在下一个周期中继续进行处。在此情况下，变为在转移条件成立的下一个周期中，执行转移目标步的方式。

连续转移控制的含义是，在1次执行周期中，转移条件成立的情况下，同一周期中转移目标步也连续执行的(连续执行基本1动作)控制。在此情况下，可以设置连续转移数。对于普通任务中执行的运动SFC程序，进行通用控制。

要点

对于通过事件任务、NMI任务执行的运动SFC程序，对各程序设置连续转移数。

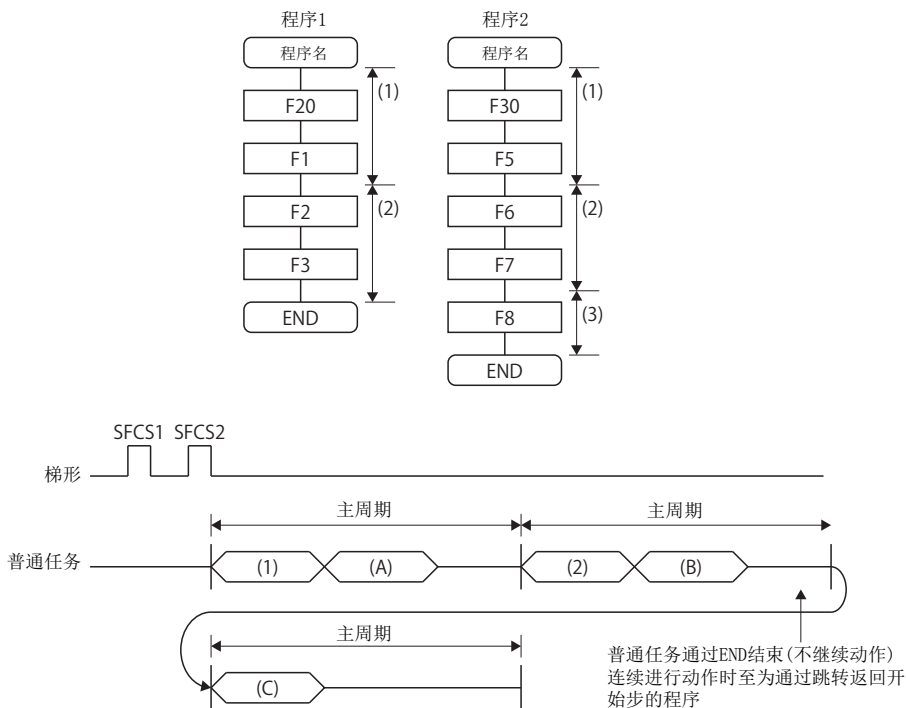
任务动作

普通任务时的动作

n 动作说明

通过运动CPU的主周期(空余时间)，执行运动SFC程序。处理概要如下所示。

- 连续转移数设置为“2”的情况下



n 注意事项

- 对于包含运动控制步的运动SFC程序，应设置到普通任务中。
- 事件任务·NMI任务的执行过程中，普通任务的执行将中断。但是，在普通任务的运算控制步中，可以记述事件任务禁止指令(DI)，因此在事件任务禁止指令(DI)及事件任务允许指令(EI)所围住的部分中，可以禁止事件任务的中断。对于事件任务允许·禁止状态，可以通过“EI标志(SM752)”进行确认。

事件任务时的动作

n 动作说明

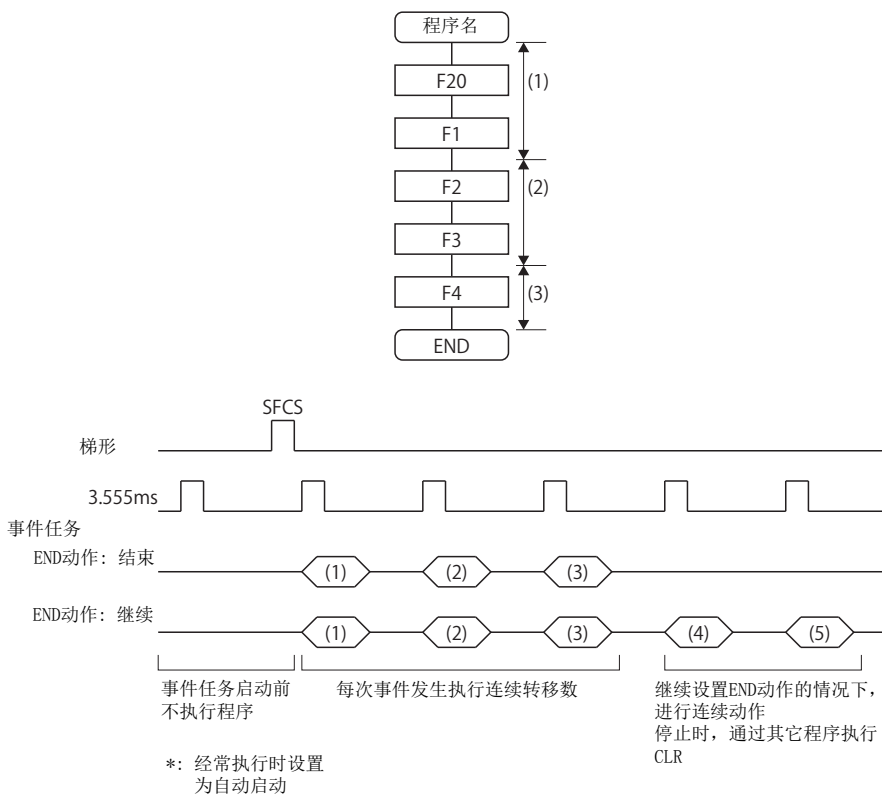
事件任务在发生事件时执行运动SFC程序。事件有以下几种。

事件	内容
恒定周期	以0.222ms/0.444ms/0.888ms/1.777ms/3.555ms/7.111ms/14.222ms中之一为1个周期，定期执行运动SFC程序。
外部中断(中断指针(I0~I15)的16点)	在分配至运动CPU管理的输入模块的16点的中断指针(I0~I15)中，用于事件任务而设置的输入变为ON时，执行运动SFC程序。
可编程控制器中断	通过顺控程序执行了M(P).GINT/D(P).GINT指令时，执行运动SFC程序。

例

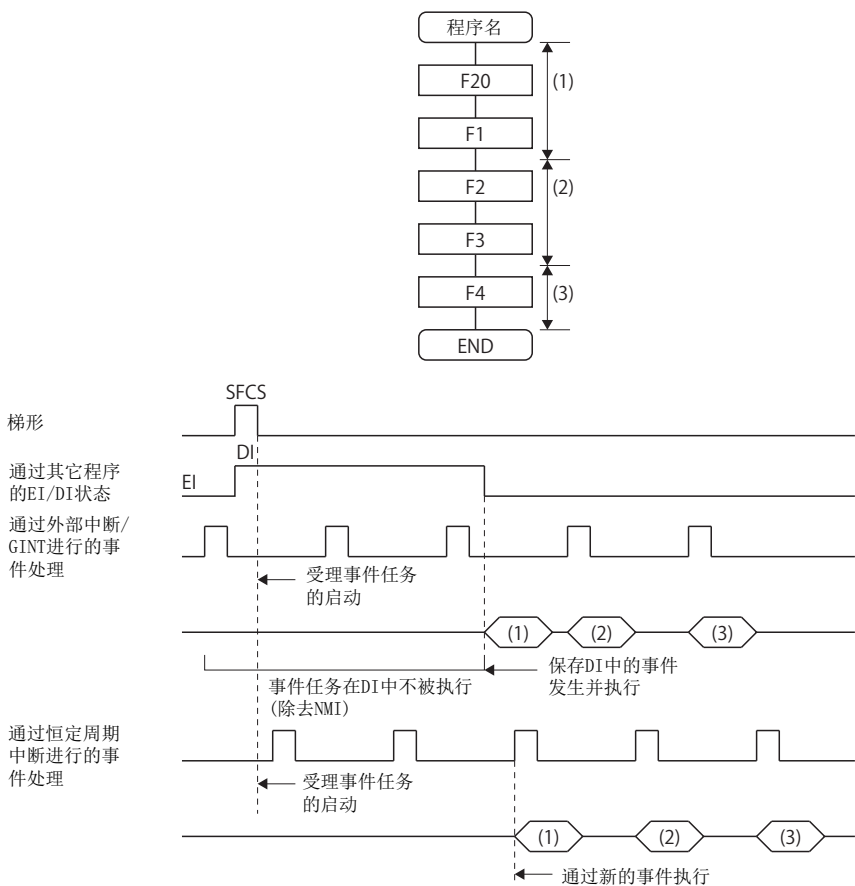
(例1)

3. 55ms的恒定周期任务的动作(连续转移数设置为“2”的情况下)



(例2)

通过D(P).GINT/M(P).GINT进行的可编程控制器中断的动作(连续转移数设置为“2”的情况下)



n 注意事项

- 对1个运动SFC程序可以设置多个事件。但是，不能设置多个恒定周期。
- 1个事件中，也可执行多个运动SFC程序。
- 在事件任务中，不能执行运动控制步。
- 通过普通任务设置为事件任务禁止的情况下，不能执行事件任务。事件任务禁止中发生了事件的情况下，将在事件任务被允许的時刻执行。

n 出错

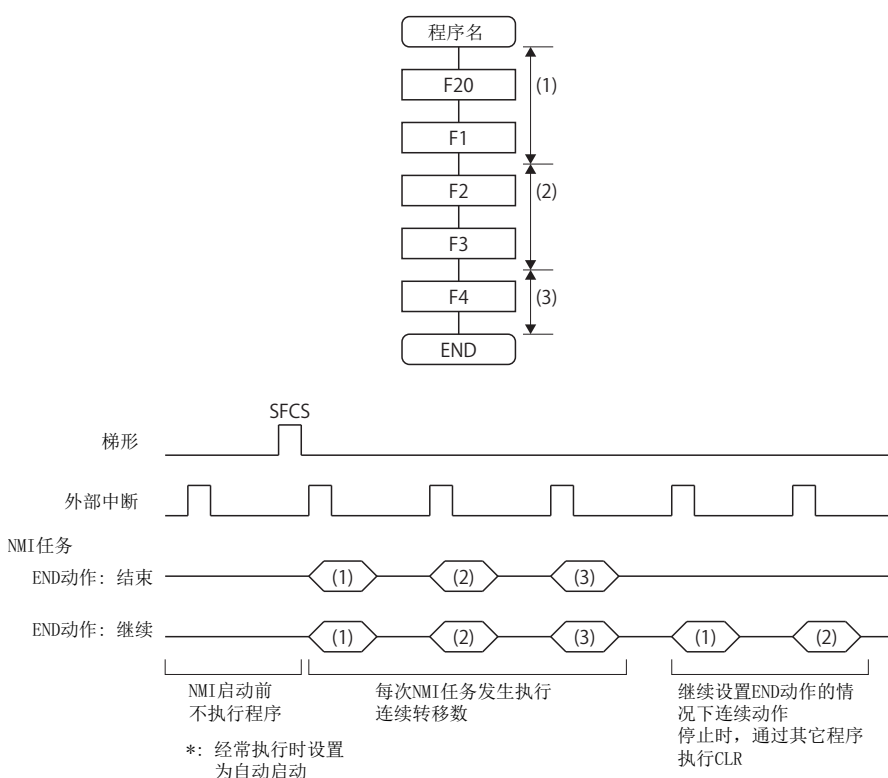
在事件任务中设置的运动SFC程序内，执行运动控制步时，将发生轻度出错(SFC) (出错代码： 33FDH)，运动SFC程序的执行将中止。

NMI任务时的动作

n 动作说明

在外部中断(运动CPU管理的输入模块中分配的16点的中断指针(I0~I15))中，在NMI任务原因中设置的输入变为ON的時刻，执行运动SFC程序。

- 连续转移数设置为“2”的情况下



n 注意事项

- 在普通任务/事件任务/NMI任务中，NMI任务的优先度最高。
- 即使通过普通任务设置为事件任务禁止(DI)，NMI任务的中断也将被执行而不会被屏蔽。
- NMI任务执行中进行了并联分支的情况下，由于分支而增加的路径在发生下一个中断时将开始执行。

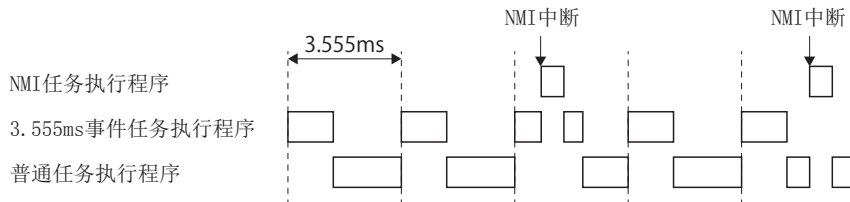
n 出错

在NMI任务中，不能执行运动控制步。在NMI任务中执行运动控制步时，将发生轻度出错(SFC) (出错代码： 33FDH)，运动SFC程序的执行将中止。

6.3 多个任务的执行状态

在多个任务中执行运动SFC程序的情况下

在多个任务中正在执行运动SFC程序的情况下，各运动SFC程序的执行状态示例如下所示。



存在有通过NMI任务执行的程序、通过3.555ms恒定周期事件任务执行的程序以及通过普通任务执行的程序的情况下，其动作如上图所示。

- (1) 3.555ms恒定周期事件任务每隔3.555ms被执行
- (2) 输入了NMI中断时，优先执行NMI任务
- (3) 普通任务在空余时间内执行

关于包含了运动运算及主周期处理等的整个运动CPU的处理时机，请参阅以下手册。

📖 MELSEC iQ-R运动控制器编程手册(公共篇)

n 注意事项

在1个运动SFC程序中，希望通过其它任务执行部分程序的情况下，将希望通过其它任务执行的部分设置为子程序，通过将子程序的执行任务设置为其它任务类型，可以实现上述目的。

例

No. 0 主运动SFC程序：普通任务

No. 1 子程序：事件任务（3.555ms周期）

⚠ 注意

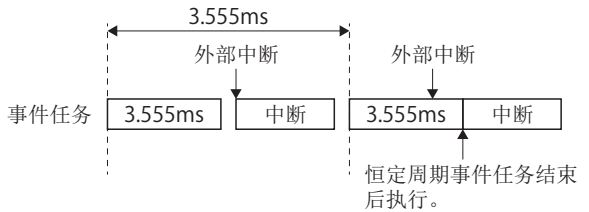
- 执行多个NMI任务、事件任务时，有可能发生运算周期溢出，或几乎不执行普通任务而发生WDT出错。

同一任务类型中发生多个启动原因的情况下

- 在事件任务程序执行过程中，发生了其它事件的情况下，在执行中程序的连续转移数的执行完成之后，执行下一个事件对应的程序。

例

恒定周期事件任务执行中发生了外部中断的情况下，在恒定周期事件任务的执行(连续转移数)完成之后，执行外部中断事件任务。



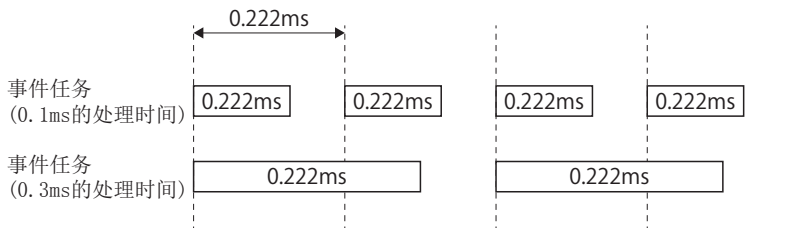
- 再次发生与执行中的事件任务程序相同的事件的情况下，或NMI任务执行中再次发生NMI中断的情况下，其动作如下所示。

任务类型		发生同一事件时的动作
事件任务	恒定周期(0.222ms、0.444ms、0.888ms、1.777ms、3.555ms、7.111ms、14.222ms)	后面的事件将被忽略
	外部中断	依次执行
	可编程控制器中断	
NMI	外部中断	

例

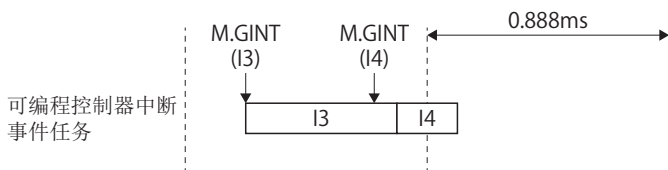
(例1)

将以下处理时间的处理通过0.222ms恒定周期事件任务执行的情况下，0.3ms的处理仅在每隔0.444ms时启动。

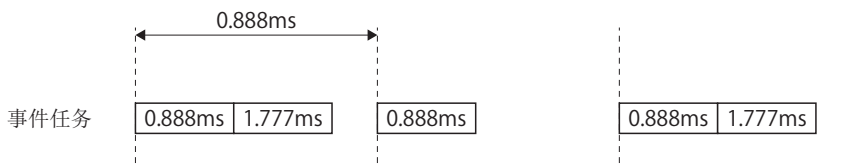


(例2)

多次发行了可编程控制器中断的情况下，按发行次数依次启动事件任务。



- 恒定周期的事件任务按照从周期较短任务开始的顺序执行。



6.4 运动SFC程序的启动方法

在“[Rq. 1120]可编程控制器就绪标志(R: M30000/Q: M2000)”为ON的状态下运动SFC程序将动作。运动SFC程序的启动方法有以下3种，在各运动SFC程序中，通过程序参数进行启动方法的设置。(☞ 275页 程序参数)

- 自动启动
- 通过运动SFC程序的启动
- 通过其它机号的运动专用顺控程序指令(D(P). SFCS/M(P). SFCS)的启动

自动启动

通过将“[Rq. 1120]可编程控制器就绪标志(R: M30000/Q: M2000)”置为ON，自动启动。

通过运动SFC程序的启动

在运动SFC程序中，通过执行子程序调用/启动步进行启动。(☞ 92页 子程序调用/启动步)

通过其它机号的运动专用顺控程序指令进行启动(顺控程序指令： M(P). SFCS/D(P). SFCS)

通过使用顺控程序执行M(P). SFCS/D(P). SFCS指令，启动运动SFC程序。

(☞ 29页 至运动CPU的运动SFC启动请求: M(P). SFCS/D(P). SFCS)

6.5 运动SFC程序的结束方法

- 通过执行运动SFC程序中设置的END结束程序。
- 通过将“[Rq. 1120]可编程控制器就绪标志(R: M30000/Q: M2000)”置为OFF, 停止运动SFC程序。
- 可以通过清除步结束程序。(☞ 93页 清除步)

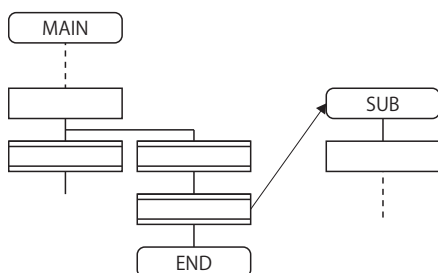
要点

在1个运动SFC程序中可以设置多个END。

6.6 运动SFC程序的切换方法

为了中止执行中的运动SFC程序, 切换至其它运动SFC程序, 应使用子程序启动。

- 通过子程序启动进行的运动SFC程序切换



6.7 多CPU系统电源断开时、复位时的动作

本节介绍多CPU系统电源断开时或复位时的运动SFC程序的动作有关内容。

- 进行多CPU系统电源断开或复位操作时, 运动SFC程序的执行将中止。
- 接通多CPU系统电源时或复位时, 锁存范围中设置的软元件的内容将被保持, 应根据需要在运动SFC程序中进行初始化。
- 多CPU系统电源断开时或复位处理后的运动SFC程序操作如下所示。
 - 对于设置为自动启动的运动SFC程序, 通过将“[Rq. 1120]可编程控制器就绪标志(R: M30000/Q: M2000)”置为ON, 从程序起始开始执行。
 - 其它的运动SFC程序在启动时也从程序起始开始执行。
- 多CPU系统电源发生瞬间掉电时, 运动SFC程序的执行仍将继续。

6.8 任务参数

项目		设置范围	初始值	备注
连续转移数	普通任务 (普通任务通用)	1~30	3	在“[Rq. 1120]可编程控制器就绪标志(R: M30000/Q: M2000)”的OFF→ON的上升沿时获取该参数, 进行此后的控制。 对该参数进行设置・更改的情况下, 应将“[Rq. 1120]可编程控制器就绪标志(R: M30000/Q: M2000)”置为OFF。
中断设置		对于外部中断输入(I0~I15), 设置事件任务或NMI任务。	事件任务	
重复控制限制 次数	普通任务	1~100000	1000	
	事件任务	1~10000	100	
	NMI任务	1~10000	100	

连续转移数

n 内容

运动SFC程序的执行控制时, 在各个任务的执行周期中, 将“激活步的执行→下一个转换条件的判定→条件成立时的转移处理(激活步的转移)”作为基本1动作, 按照相应任务的激活步数执行该动作后, 结束1次处理。然后, 在下一个周期中继续进行。在此情况下, 变为在转移条件成立的下一个周期中, 执行转移目标步的方式。连续转移控制的含义是, 在1次执行周期中, 转移条件成立的情况下, 同一周期中转移目标步也连续执行的(连续执行基本1动作)控制。在此情况下, 可以设置连续转移数。对于普通任务中执行的运动SFC程序, 进行通用控制。

要点

对于通过事件任务、NMI任务执行的运动SFC程序, 在各程序中进行设置。

n 出错

“[Rq. 1120]可编程控制器就绪标志(R: M30000/Q: M2000)”的上升沿时, 获取该参数, 进行检查。设置值超出设置范围的情况下, 将发生中度出错(出错代码: 2220H)。

中断设置

n 内容

对于运动CPU管理的输入模块中分配的中断指针(I0~I15), 设置是作为NMI任务的输入使用还是作为事件任务的输入使用。可对每个电进行任意设置。初始值为全部点事件任务。

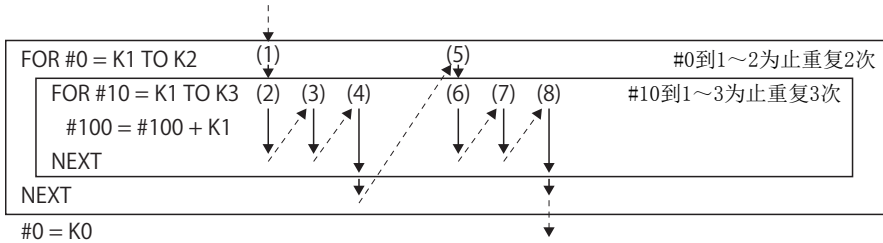
n 出错

无

重复控制限制次数

n 内容

在运算控制程序或转换程序中，如果重复控制指令(FOR~NEXT)的重复次数过多，运算控制程序的处理时间将变长。因此，这可能导致主周期的增大及事件任务/NMI任务中运算周期溢出，但通过设置“重复控制限制次数”可以防止。设置“重复控制限制次数”时，是对普通任务/事件任务/NMI任务分别进行设置。在1个运算控制程序或转换程序中，重复控制指令(FOR~NEXT)的执行超出“重复控制限制次数”时，将发生轻度出错(SFC)(出错代码：31F8H)，相应运动SFC程序No.的执行将停止。此外，子程序调用的程序的情况下，调用源程序的执行也将停止。对于重复控制指令的执行次数，执行FOR指令时判定条件为重复条件继续的情况下(判定条件为真时)将变为1次。在以下程序中，各块按箭头路径被执行，重复控制指令的执行次数变为8。



n 出错

无

“[Rq. 1120]可编程控制器就绪标志(R: M30000/Q: M2000)”的上升沿时获取参数，进行检查。超出设置范围的情况下，将发生中度出错(出错代码：2220H)。

6.9 程序参数

在各运动SFC程序中，应进行以下参数设置。

项目	设置范围	初始值	备注
启动设置	自动启动进行/不进行	不进行	在“[Rq. 1120]可编程控制器就绪标志(R: M30000/Q: M2000)”的上升沿时获取该参数，进行此后的控制。对该参数进行设置・更改的情况下，应将“[Rq. 1120]可编程控制器就绪标志(R: M30000/Q: M2000)”置为OFF。
执行任务	仅普通任务/事件任务/NMI任务中之一。	普通任务	
	设置了事件任务的情况下，还应设置生效的事件。必须进行以下设置之一。 <ul style="list-style-type: none"> 恒定周期：0.222ms/0.444ms/0.888ms/1.777ms/3.555ms/7.111ms/14.222ms中之一或无设置。 外部中断(根据事件任务中的设置选择)：可以从 I0・I1・I2・I3・I4・I5・I6・I7・I8・I9・I10・I11・I12・I13・I14・I15中设置多个。 可编程控制器中断：可以从 I0・I1・I2・I3・I4・I5・I6・I7・I8・I9・I10・I11・I12・I13・I14・I15中设置多个。 可以进行上述多个设置。 同一事件也可在多个运动SFC程序中共用。	无	
连续转移数	设置了NMI任务的情况下，还应设置生效的中断输入。 <ul style="list-style-type: none"> 外部中断(根据NMI任务中的设置选择)：可以从 I0・I1・I2・I3・I4・I5・I6・I7・I8・I9・I10・I11・I12・I13・I14・I15中设置多个。 		
	1~10 对于事件任务或NMI任务中设置的程序，应设置连续转移数。	1	
END动作	结束/继续 对于事件任务或NMI任务中设置的程序，应设置END步的动作模式。	结束	
执行中标志	无/位软元件 应设置运动SFC程序执行中变为ON的位软元件。 ^{*1}	无	

*1 关于可设置的位软元件的范围，请参阅以下手册。

📖 MELSEC iQ-R运动控制器编程手册(公共篇)

要点 🔍

“END动作”的设置对于子程序调用的程序无效。“END动作”被作为“结束”进行控制。

启动设置

n 内容

以下控制将根据“自动启动的进行/不进行”的设置而变化。

- 通过普通任务执行的程序的情况下

项目	“进行自动启动”的情况下	“不进行自动启动”的情况下
启动控制	在“[Rq. 1120]可编程控制器就绪标志(R: M30000/Q: M2000)”的上升沿后的主周期中，通过初始化(起始)步按照普通任务的连续转移数执行。	通过其它机号的运动SFC启动请求(M(P).SFCS/D(P).SFCS)启动，或通过运动SFC程序中子程序调用/启动(GSUB)启动。 <ul style="list-style-type: none"> 通过运动SFC启动请求(M(P).SFCS/D(P).SFCS)启动的情况下：在运动SFC启动请求(M(P).SFCS/D(P).SFCS)执行后的主周期中，通过初始化(起始)步按照普通任务的连续转移数执行。 子程序启动的情况下：在执行GSUB后(下次)主周期中，通过起始步按照普通任务的连续转移数执行。 子程序调用的情况下：在同一周期中通过起始步执行。
	然后，在运动CPU的主周期中，以普通任务的连续转移数继续执行。 (子程序调用的程序的“执行任务”・“连续转移数”的设置无效。作为普通任务进行控制。)	
结束控制 [END]	结束本程序。	

- 通过事件任务执行的程序的情况下

项目	“进行自动启动”的情况下	“不进行自动启动”的情况下
启动控制	通过“[Rq. 1120]可编程控制器就绪标志(R: M30000/Q: M2000)”的上升沿后的事件发生，通过初始化(起始)步按照相应程序的连续转移数执行。	通过其它机号的运动SFC启动请求(M(P). SFCS/D(P). SFCS)启动，或通过运动SFC程序中子程序调用/启动(GSUB)启动。 <ul style="list-style-type: none"> • 通过运动SFC启动请求(M(P). SFCS/D(P). SFCS)启动的情况下：通过运动SFC启动请求(M(P). SFCS/D(P). SFCS)执行后的事件发生，通过初始化(起始)步按照相应程序的连续转移数执行。 • 子程序启动的情况下：通过执行GSUB后的事件发生，通过起始步按照相应程序的连续转移数执行。 • 子程序调用的情况下：立即通过起始步执行。
	每次发生事件时，按照相应程序的连续转移数继续执行。 (对于子程序调用的程序，按照调用源程序的“执行任务”、“连续转移数”进行控制。)	
结束控制 [END]	按照END动作的指定。	

- 通过NMI任务执行的程序的情况下

项目	“进行自动启动”的情况下	“不进行自动启动”的情况下
启动控制	通过“[Rq. 1120]可编程控制器就绪标志(R: M30000/Q: M2000)”的上升沿后的事件发生，通过初始化(起始)步按照相应程序的连续转移数执行。	通过其它机号的运动SFC启动请求(M(P). SFCS/D(P). SFCS)启动，或通过运动SFC程序中子程序调用/启动(GSUB)启动。 <ul style="list-style-type: none"> • 通过运动SFC启动请求(M(P). SFCS/D(P). SFCS)启动的情况下：通过运动SFC启动请求(M(P). SFCS/D(P). SFCS)执行后的有效事件发生，通过初始化(起始)步按照相应程序的连续转移数执行。 • 子程序启动的情况下：通过执行GSUB后的有效事件发生，通过起始步按照相应程序的连续转移数执行。 • 子程序调用的情况下：立即通过起始步执行。
	每次发生事件时，按照相应程序的连续转移数继续执行。	
结束控制 [END]	按照END动作的指定。	

n 出错

无

要点

通过普通任务执行的程序的情况下，希望通过1个循环动作结束，再次通过初始化自动启动的情况下，应设置为不通过[END]结束，通过跳转返回至开始步的程序。

执行任务

n 内容

设置程序的执行时机(任务)。在“普通任务(主周期)/事件任务(恒定周期·外部中断·可编程控制器中断)/NMI任务(外部中断)”中，设置通过哪个任务执行。

设置了事件任务的情况下，在“恒定周期·外部中断(事件任务用)·可编程控制器中断”中，可以设置多个事件。但是，不能对1个运动SFC程序设置多个恒定周期。

例

中断设置：事件任务用输入I6, I7, I8, I9, I10, I11, I12, I13, I14, I15

运动SFC程序No. 10 —— 事件：恒定周期(3.555ms)

运动SFC程序No. 20 —— 事件：恒定周期(1.777ms)+外部中断(I6)

运动SFC程序No. 30 —— 事件：外部中断(I7, I15)+可编程控制器中断

设置了NMI任务的情况下，外部中断(NMI任务用)中可以设置多个中断输入。

例

中断设置：NMI任务用输入I0, I1, I2, I3, I4, I5

运动SFC程序No. 10 —— NMI：I0

运动SFC程序No. 20 —— NMI：I1+I2

运动SFC程序No. 30 —— NMI：I5

n 出错

“[Rq. 1120]可编程控制器就绪标志(R: M30000/Q: M2000)”的上升沿时，获取该程序参数，运动SFC程序启动时(自动启动或通过可编程控制器的启动或子程序调用/启动)进行检查。值不正确的情况下，将发生中度出错(出错代码: 2220H)，以初始值进行控制。

要点

- 可以对各运动SFC程序No. 进行执行任务的设置，因此对于1个控制(机械动作)，不需要设置在各执行时机分开处理的多个程序。在普通任务执行的运动SFC程序中，通过将部分程序以恒定周期执行，将希望作为外部中断执行的部分设置为子程序启动，可以方便地实现预期目的。
- 对于恒定周期的事件任务的周期，应确认“运动运算周期(SD522)”，设置大于运动运算周期的恒定周期。

连续转移数

n 内容

对于通过事件任务或NMI任务执行的程序，对各程序设置连续转移数。

关于“连续转移数”，请参阅连续转移数。(☞ 273页 连续转移数)

n 出错

“[Rq. 1120]可编程控制器就绪标志(R: M30000/Q: M2000)”的上升沿时，获取该程序参数，运动SFC程序启动时(自动启动或通过可编程控制器的启动或子程序启动)进行检查。值不正确的情况下，将发生中度出错(出错代码: 2220H)，以初始值进行控制。

END动作

n 内容

对于通过事件任务或NMI任务执行的程序，设置执行END步时的动作。由此，对于下述项目的规格有所不同。

项目	“结束”的情况下	“继续”的情况下
执行[END]时的控制	结束本程序。	结束本次的事件/中断中的本程序的执行。
执行[END]后的重启	再次从其它机号通过运动SFC启动请求(D(P). SFCS/M(P). SFCS)启动，或在运动SFC程序中通过子程序调用/启动(GSUB)进行启动。	通过下次的事件/中断发生进行重启，从初始化(起始)步开始按照相应程序的连续转移数执行。此后，通过事件/中断发生按照相应程序的连续转移数进行控制。
通过清除步CLR结束后的重启	再次从其它机号通过运动SFC启动请求(D(P). SFCS/M(P). SFCS)启动，或在运动SFC程序中通过子程序调用/启动(GSUB)进行启动。	

要点

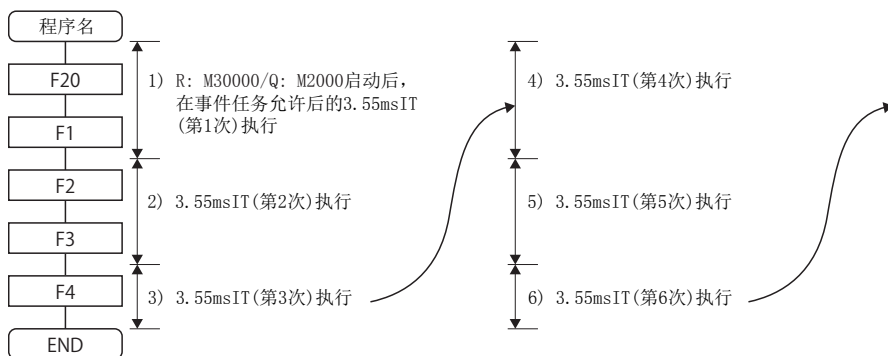
子程序调用的程序的END动作作为“结束”进行控制。

- 将END动作设置为“继续”情况下的动作示例如下所示。

例

程序参数

项目	设置内容
启动设置	进行自动启动
执行任务	事件任务(恒定周期: 3.555ms)
连续转移数	2
END动作	继续



执行中标志

运动SFC程序启动时，设置的位软元件将变为ON，结束时将变为OFF。

7 运动SFC的功能

7.1 运动SFC程序的运行中写入

是定位控制中([Rq. 1120]可编程控制器就绪标志(R: M30000/Q: M2000)ON中)将运动SFC程序写入到运动CPU的内置存储器的功能。重复进行程序的修改及动作确认时,可以在不停止运动CPU的动作的状况下更改程序。此外,由于只需写入运动SFC程序的更改位置,写入时间较短,可以缩短调试作业所需时间。可进行运行中写入的数据如下所示。

○: 可以; ×: 不能

对象数据	运行中写入可否	备注	
R系列通用参数	×		
运动CPU通用参数	×		
运动控制参数	×		
运动SFC程序	运动SFC参数	×	
	运动SFC图	○	仅停止中的程序可以进行运行中写入
	运算控制步(F/FS)	○	
	转换(G)	○	
	伺服程序(K)	○	不能进行模式分配设置的运行中写入
凸轮数据	×		

要点

- 运行中写入时,在定位控制中进行程序写入,因此应在充分注意安全的基础上执行作业。
- 使用了SD存储卡的引导时文件传送功能中指定了运动SFC程序文件或伺服程序文件的情况下,根据运行中写入用文件的有无,有可能发生意外的程序动作。关于详细内容,请参阅以下手册。
📖 MELSEC iQ-R运动控制器编程手册(公共篇)
- 从多个个人计算机对1个运动CPU同时进行运行中写入时,有可能无法进行程序写入,因此请勿进行此操作。
- 在通过MT Developer2进行的运动SFC程序的监视模式、运动SFC程序调试模式或测试模式的操作中,如果从其它个人计算机进行运行中写入,有可能发生监视值不正确及误动作,因此请勿进行此操作。
- 对新增的运动SFC图进行运行中写入时,不能进行运动SFC参数的运行中写入,因此作为普通任务(初始值)执行动作。
- 在伺服程序编辑画面的“程序分配设置”中即使更改指令生成轴程序的范围后进行运行中写入,更改内容也不被反映,运行中写入将中断。
- 运行中写入过程中,如果拔下个人计算机与可编程控制器CPU模块之间的电缆,进行多CPU系统的电源OFF或复位,程序将损坏。应通过MT Developer2进行数据的写入操作,再次写入程序。
- 运行中写入时,只有在通过运动CPU动作中的程序与MT Developer2的工程数据(更改前)一致时才进行写入。写入前进行检查,数据不一致的情况下,中断运行中写入。

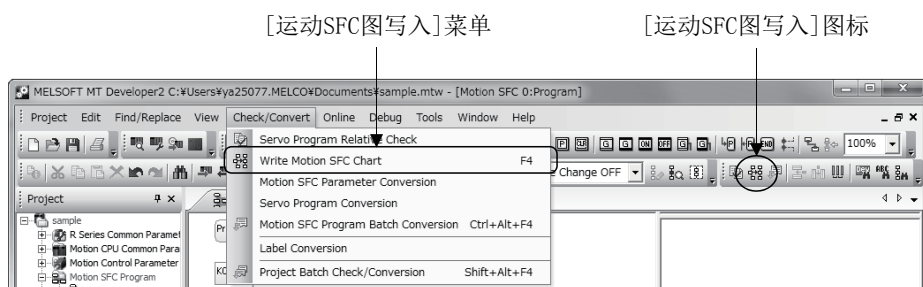
运行中写入操作方法

在通过MT Developer2的菜单栏[工具]⇒[运行中写入设置]显示的运行中写入设置画面中选择“运行中写入的不进行/进行”。运动SFC程序的运行中写入方法如下所示。

运行中写入对象数据	运行中写入操作
运动SFC图	<ul style="list-style-type: none">通过菜单栏选择[检查/转换]⇒[运动SFC图写入]。从工具栏点击[运动SFC图写入]图标。
运算控制程序(F/FS)	点击运算控制程序·转换程序编辑画面的[转换]按钮。
转换程序(G)	
伺服程序(K)	点击伺服程序编辑画面的[转换]按钮。

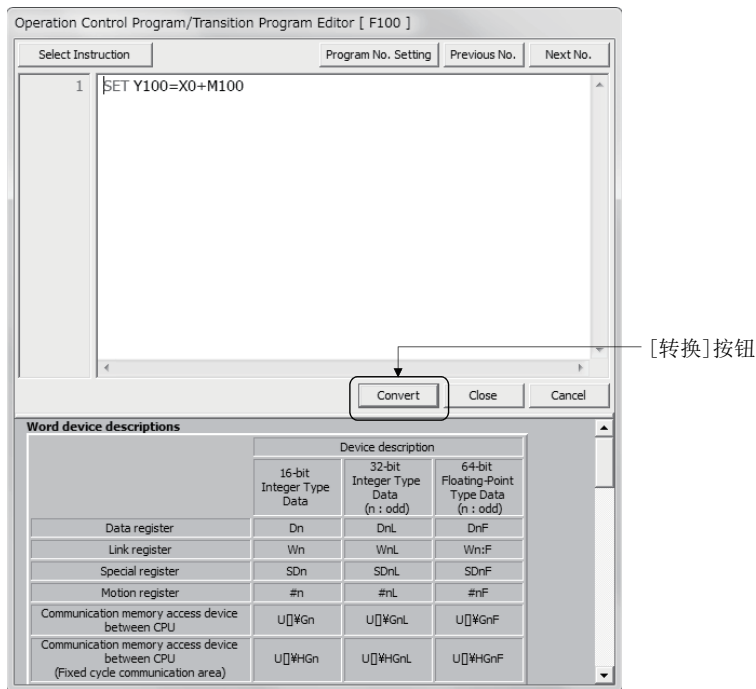
运动SFC图的运行中写入

通过工具栏的图标按钮或菜单选择，对编辑中的运动SFC图进行运行中写入。对停止中的运动SFC程序可以进行运行中写入。对执行中的运动SFC程序进行了运行中写入的情况下，将显示报警信息。(对于运动SFC程序的执行/停止状态，可以通过监视模式的程序批量监视进行确认。)对运行中写入过程中的运动SFC程序有启动请求的情况下，将发生中度出错(出错代码：32F7H)，运动SFC程序不启动。



运算控制/转换程序的运行中写入

通过[转换]按钮选择，对编辑中的运算控制/转换程序进行运行中写入。可以对执行中的运算控制/转换程序进行运行中写入。从下一个扫描时开始运行中写入的程序将被执行。

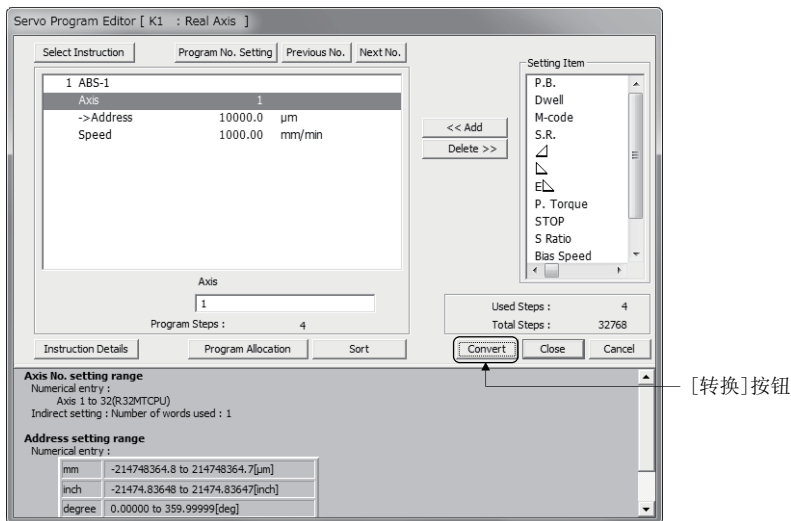


以下述条件，对执行中的运算控制/转换程序进行了运行中写入时的动作内容如下所示。以下述条件进行运行中写入的情况下，应加以注意。

程序	条件	动作
	Gn的条件成立等待中，FSn执行中对FSn的运算控制程序进行运行中写入。	运行中写入完成后，FSn重复执行运行中写入的运算控制程序，直至Gn的条件成立为止。
	Gn的条件成立等待中，对Gn的程序进行运行中写入(写入程序的条件语句为TIME指令以外)。	运行中写入完成后，Gn在运行中写入的条件的条件成立之前不转移至下一个步。
	Gn的条件成立等待中，对Gn的程序中包含TIME指令的程序进行运行中写入。	运行中写入完成后，与TIME指令的等待时间无关，结束Gn的执行，执行下一个步。
	Kn的伺服程序执行中，对Gn的程序进行运行中写入。	伺服程序执行后，执行更改后的Gn的程序。

伺服程序的运行中写入

通过[转换]按钮选择，对编辑中的伺服程序进行运行中写入。对执行中的伺服程序可以进行运行中写入。启动以下伺服程序时运行中写入的程序将被执行。



以下述条件，对执行中的伺服程序进行了运行中写入时的动作内容如下所示。以下述条件进行运行中写入的情况下，应加以注意。

程序	条件	动作
	WAITON/WAITOFF的条件成立等待中，对WAIT ON或WAIT OFF后的伺服程序Kn进行运行中写入。	<ul style="list-style-type: none"> • WAITON/WAITOFF的条件成立后，进行运行中写入以前的伺服程序将启动。 • 启动下一个伺服程序时，将执行运行中写入后的伺服程序。
	Gn的条件成立等待中，对Gn后的伺服程序Kn进行运行中写入。	Gn的条件成立后，执行运行中写入后的伺服程序。

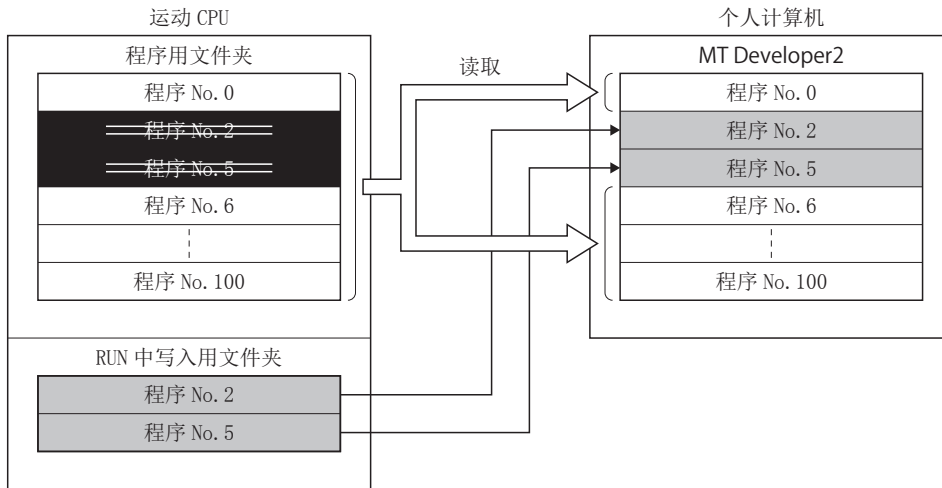
程序的读取/写入

以下介绍通过MT Developer2将程序写入到运动CPU的程序存储器时的大致动作。

通过MT Developer2的操作

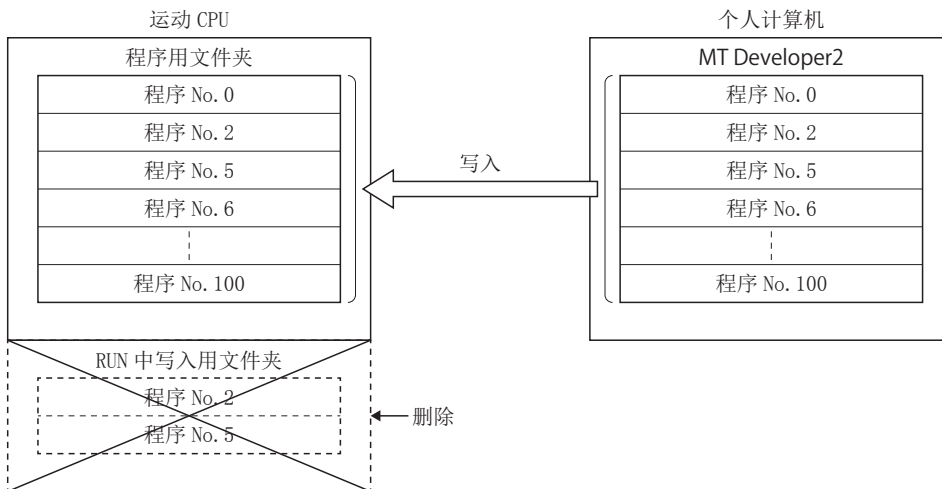
n 通过MT Developer2的读取操作进行程序的读取

进行程序的读取时，对程序用文件夹及运行中写入用文件夹中存储的所有文件进行读取，合并后的程序将被反映到工程中。（运动CPU内处于执行状态的程序构成将被恢复。）



n 通过MT Developer2的写入操作进行程序的写入

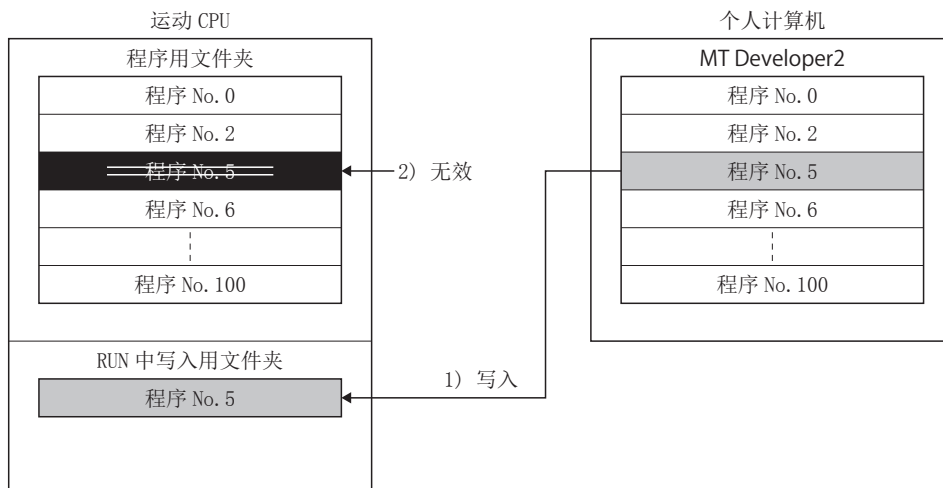
进行程序的写入时，将程序文件更新到程序用文件夹中并存储。此外，存在有运行中写入用文件夹的情况下，删除运行中写入用文件夹。



n 通过MT Developer2的运行中写入操作进行程序的写入

进行运行中写入时，运行中写入用文件夹中将存储新的程序。(参阅(1))

然后，将先写入的程序设置为无效，使本次写入的程序生效。(参阅(2))



要点

- 对于程序用文件夹及运行中写入用文件夹中存储的文件以成套方式进行处理。引导时文件传送功能中程序用文件夹内的程序文件为对象的情况下，运行中写入用文件的内容也将自动被传送。
- 重复进行运行中写入时，运动CPU内部的作业用空余区域有可能消失，无法进行运行中写入。在此情况下，应通过MT Developer2的写入操作进行程序的写入(通过MT Developer2的写入操作进行程序的写入)。

使用引导时文件传送功能时的运行中写入

使用了SD存储卡的引导时文件传送功能中指定了运动SFC程序文件或伺服程序文件的情况下，根据运行中写入用文件的有无，有可能发生意外的程序动作。关于详细内容，请参阅以下手册。

📖 MELSEC iQ-R运动控制器编程手册(公共篇)

7.2 运动SFC程序的监视及调试模式

运动SFC程序的监视

通过MT Developer2的运动SFC程序监视可以监视运动SFC程序的执行状态。可监视的项目如下所示。关于运动SFC程序监视的详细情况，请参阅以下内容。

📖 MT Developer2的帮助

- 执行状态一览(执行中/停止中/中断中)
- 各程序的执行状态(执行中/停止中/中断中)
- 激活状态(激活中/并联合并等待中)

此外，通过特殊继电器(SM)、特殊寄存器(SD)，可以确认程序的执行信息。关于详细内容，请参阅以下手册。

📖 MELSEC iQ-R运动控制器编程手册(公共篇)

- 程序执行时间
- EI/DI状态

调试模式

在调试模式中将创建的运动SFC程序在任意位置中断(break)，以单步方式执行，可以确认动作。调试模式中可进行的操作如下所示。关于调试模式的详细情况，请参阅以下内容。

📖 MT Developer2的帮助

- 中断点的设置(有效/无效、次数)
- 强制中断
- 单步执行
- 转换的强制转移
- 程序的强制结束

调试模式中，“[St. 1041]运动SFC调试模式中标志(R: M30038/Q: M2038)”将变为ON。

附录

附1 处理时间

运算控制・转换指令处理时间

各指令的运算处理时间如下所示。运算处理时间根据源、目标的内容而有所不同，因此对表中的值应作为处理时间的大致参考进行参照。

运算指令

分类	符号	指令	运算公式	单位[μs]			
二项运算	=	代入	#0=#1	1.5			
			D800=D801	1.5			
			U3E1\G10000=U3E1\G10001	2.0			
			U3E1\HG10000=U3E1\HG10001	1.8			
			#0L=#2L	1.5			
			D800L=D802L	1.5			
			U3E1\G10000L=U3E1\G10002L	2.0			
			U3E1\HG10000L=U3E1\HG10002L	1.8			
			#0F=#4F	1.5			
			D800F=D804F	1.5			
			U3E1\G10000F=U3E1\G10004F	2.0			
			U3E1\HG10000F=U3E1\HG10004F	1.8			
			+	+	加法	#0=#1+#2	2.2
						D800=D801+D802	2.2
						U3E1\G10000=U3E1\G10001+U3E1\G10002	4.6
U3E1\HG10000=U3E1\HG10001+U3E1\HG10002	2.8						
#0L=#2L+#4L	2.2						
D800L=D802L+D804L	2.2						
U3E1\G10000L=U3E1\G10002L+U3E1\G10004L	3.2						
U3E1\HG10000L=U3E1\HG10002L+U3E1\HG10004L	2.9						
#0F=#4F+#8F	2.2						
D800F=D804F+D808F	2.2						
U3E1\G10000F=U3E1\G10004F+U3E1\G10008F	3.2						
U3E1\HG10000F=U3E1\HG10004F+U3E1\HG10008F	2.9						
-	-	减法				#0=#1-#2	2.2
						D800=D801-D802	2.2
						U3E1\G10000=U3E1\G10001-U3E1\G10002	3.1
			U3E1\HG10000=U3E1\HG10001-U3E1\HG10002	2.8			
			#0L=#2L-#4L	2.2			
			D800L=D802L-D804L	2.2			
			U3E1\G10000L=U3E1\G10002L-U3E1\G10004L	3.2			
			U3E1\HG10000L=U3E1\HG10002L-U3E1\HG10004L	2.9			
			#0F=#4F-#8F	2.2			
			D800F=D804F-D808F	2.2			
			U3E1\G10000F=U3E1\G10004F-U3E1\G10008F	3.2			
			U3E1\HG10000F=U3E1\HG10004F-U3E1\HG10008F	2.9			

分类	符号	指令	运算公式	单位[μs]		
二项运算	*	乘法	#0=#1*#2	2.2		
			D800=D801*D802	2.2		
			U3E1\G10000=U3E1\G10001*U3E1\G10002	3.1		
			U3E1\HG10000=U3E1\HG10001*U3E1\HG10002	2.8		
			#0L=#2L*#4L	2.2		
			D800L=D802L*D804L	2.2		
			U3E1\G10000L=U3E1\G10002L*U3E1\G10004L	3.1		
			U3E1\HG10000L=U3E1\HG10002L*U3E1\HG10004L	2.9		
			#0F=#4F*#8F	2.2		
			D800F=D804F*D808F	2.2		
			U3E1\G10000F=U3E1\G10004F*U3E1\G10008F	3.2		
			U3E1\HG10000F=U3E1\HG10004F*U3E1\HG10008F	2.9		
			/	除法	#0=#1/#2	2.2
					D800=D801/D802	2.2
	U3E1\G10000=U3E1\G10001/U3E1\G10002	3.2				
	U3E1\HG10000=U3E1\HG10001/U3E1\HG10002	2.9				
	#0L=#2L/#4L	2.2				
	D800L=D802L/D804L	2.2				
	U3E1\G10000L=U3E1\G10002L/U3E1\G10004L	3.2				
	U3E1\HG10000L=U3E1\HG10002L/U3E1\HG10004L	2.9				
	#0F=#4F/#8F	3.4				
	D800F=D804F/D808F	3.4				
	U3E1\G10000F=U3E1\G10004F/U3E1\G10008F	4.8				
	U3E1\HG10000F=U3E1\HG10004F/U3E1\HG10008F	4.5				
	%	余数	#0=#1%#2	2.2		
			D800=D801%D802	2.2		
			U3E1\G10000=U3E1\G10001%U3E1\G10002	3.2		
			U3E1\HG10000=U3E1\HG10001%U3E1\HG10002	2.9		
			#0L=#2L%#4L	2.2		
			D800L=D802L%D804L	2.2		
			U3E1\G10000L=U3E1\G10002L%U3E1\G10004L	3.2		
			U3E1\HG10000L=U3E1\HG10002L%U3E1\HG10004L	2.9		
	位运算	~	位取反(补数)	#0=~#1	1.6	
D800=~D801				1.6		
U3E1\G10000=~U3E1\G10001				2.0		
U3E1\HG10000=~U3E1\HG10001				1.9		
#0L=~#2L				1.6		
D800L=~D802L				1.6		
U3E1\G10000L=~U3E1\G10002L				2.1		
U3E1\HG10000L=~U3E1\HG10002L				1.9		
&		位逻辑积	#0=#1	2.1		
			D800=D801&D802	2.1		
			U3E1\G10000=U3E1\G10001&U3E1\G10002	3.1		
			U3E1\HG10000=U3E1\HG10001&U3E1\HG10002	2.8		
			#0L=#2LL	2.1		
			D800L=D802L&D804L	2.2		
			U3E1\G10000L=U3E1\G10002L&U3E1\G10004L	3.1		
U3E1\HG10000L=U3E1\HG10002L&U3E1\HG10004L	2.8					

分类	符号	指令	运算公式	单位[μs]	
位运算		位逻辑和	#0=#1 #2	2.1	
			D800=D801 D802	2.1	
			U3E1\G10000=U3E1\G10001 U3E1\G10002	3.1	
			U3E1\HG10000=U3E1\HG10001 U3E1\HG10002	2.8	
			#0L=#2L #4L	2.2	
			D800L=D802L D804L	2.2	
			U3E1\G10000L=U3E1\G10002L U3E1\G10004L	3.1	
			U3E1\HG10000L=U3E1\HG10002L U3E1\HG10004L	2.8	
	^	位异或	#0=#1^#2	2.1	
			D800=D801^D802	2.1	
			U3E1\G10000=U3E1\G10001^U3E1\G10002	3.1	
			U3E1\HG10000=U3E1\HG10001^U3E1\HG10002	2.8	
			#0L=#2L^#4L	2.2	
			D800L=D802L^D804L	2.2	
			U3E1\G10000L=U3E1\G10002L^U3E1\G10004L	3.1	
			U3E1\HG10000L=U3E1\HG10002L^U3E1\HG10004L	2.8	
	>>	位右移	#0=#1>>#2	2.2	
			D800=D801>>D802	2.1	
			U3E1\G10000=U3E1\G10001>>U3E1\G10002	3.1	
			U3E1\HG10000=U3E1\HG10001>>U3E1\HG10002	2.8	
			#0L=#2L>>#4L	2.2	
			D800L=D802L>>D804L	2.2	
			U3E1\G10000L=U3E1\G10002L>>U3E1\G10004L	3.1	
			U3E1\HG10000L=U3E1\HG10002L>>U3E1\HG10004L	2.8	
	<<	位左移	#0=#1<<#2	2.1	
			D800=D801<<D802	2.1	
			U3E1\G10000=U3E1\G10001<<U3E1\G10002	3.1	
			U3E1\HG10000=U3E1\HG10001<<U3E1\HG10002	2.8	
			#0L=#2L<<#4L	2.2	
			D800L=D802L<<D804L	2.2	
			U3E1\G10000L=U3E1\G10002L<<U3E1\G10004L	3.2	
			U3E1\HG10000L=U3E1\HG10002L<<U3E1\HG10004L	2.8	
	符号	-	符号取反(2的补数)	#0=#1	1.6
				D800=-D812	1.6
				U3E1\G10000=-U3E1\G10001	2.1
				U3E1\HG10000=-U3E1\HG10001	1.9
#0L=-#2L				1.6	
D800L=-D802L				1.6	
U3E1\G10000L=-U3E1\G10002L				2.1	
U3E1\HG10000L=-U3E1\HG10002L				1.9	
#0F=-#4F				2.5	
D800F=-D804F				2.4	
U3E1\G10000F=-U3E1\G10004F				2.9	
U3E1\HG10000F=-U3E1\HG10004F				2.8	

分类	符号	指令	运算公式	单位[μ s]
标准函数	SIN	正弦	#OF=SIN(#4F)	4.7
			D800F=SIN(D804F)	4.6
			U3E1\G10000F=SIN(U3E1\G10004F)	5.8
			U3E1\HG10000F=SIN(U3E1\HG10004F)	5.6
COS	余弦	#OF=COS(#4F)	4.6	
		D800F=COS(D804F)	4.6	
		U3E1\G10000F=COS(U3E1\G10004F)	5.8	
		U3E1\HG10000F=COS(U3E1\HG10004F)	5.6	
TAN	正切	#OF=TAN(#4F)	4.7	
		D800F=TAN(D804F)	4.7	
		U3E1\G10000F=TAN(U3E1\G10004F)	6.0	
		U3E1\HG10000F=TAN(U3E1\HG10004F)	5.8	
ASIN	反正弦	#OF=ASIN(#4F)	7.6	
		D800F=ASIN(D804F)	7.5	
		U3E1\G10000F=ASIN(U3E1\G10004F)	8.4	
		U3E1\HG10000F=ASIN(U3E1\HG10004F)	8.2	
ACOS	反余弦	#OF=ACOS(#4F)	5.3	
		D800F=ACOS(D804F)	5.3	
		U3E1\G10000F=ACOS(U3E1\G10004F)	6.1	
		U3E1\HG10000F=ACOS(U3E1\HG10004F)	6.1	
ATAN	反正切	#OF=ATAN(#4F)	6.0	
		D800F=ATAN(D804F)	6.0	
		U3E1\G10000F=ATAN(U3E1\G10004F)	6.8	
		U3E1\HG10000F=ATAN(U3E1\HG10004F)	6.8	
SQRT	平方根	#OF=SQRT(#4F)	3.9	
		D800F=SQRT(D804F)	3.9	
		U3E1\G10000F=SQRT(U3E1\G10004F)	4.8	
		U3E1\HG10000F=SQRT(U3E1\HG10004F)	4.7	
LN	自然对数	#OF=LN(#4F)	4.1	
		D800F=LN(D804F)	4.1	
		U3E1\G10000F=LN(U3E1\G10004F)	5.4	
		U3E1\HG10000F=LN(U3E1\HG10004F)	5.2	
EXP	指数运算	#OF=EXP(#4F)	14.2	
		D800F=EXP(D804F)	14.2	
		U3E1\G10000F=EXP(U3E1\G10004F)	15.6	
		U3E1\HG10000F=EXP(U3E1\HG10004F)	15.4	
ABS	绝对值	#OF=ABS(#4F)	1.6	
		D800F=ABS(D804F)	1.6	
		U3E1\G10000F=ABS(U3E1\G10004F)	2.1	
		U3E1\HG10000F=ABS(U3E1\HG10004F)	1.9	
RND	四舍五入	#OF=RND(#4F)	3.5	
		D800F=RND(D804F)	3.5	
		U3E1\G10000F=RND(U3E1\G10004F)	4.0	
		U3E1\HG10000F=RND(U3E1\HG10004F)	3.8	
FIX	舍去	#OF=FIX(#4F)	2.5	
		D800F=FIX(D804F)	2.5	
		U3E1\G10000F=FIX(U3E1\G10004F)	3.0	
		U3E1\HG10000F=FIX(U3E1\HG10004F)	2.8	

分类	符号	指令	运算公式	单位[μs]	
标准函数	FUP	进位	#0F=FUP (#4F)	2.5	
			D800F=FUP (D804F)	2.5	
			U3E1\G10000F=FUP (U3E1\G10004F)	3.0	
			U3E1\HG10000F=FUP (U3E1\HG10004F)	2.8	
	BIN	BCD→BIN转换	#0=BIN (#1)	1.6	
			D800=BIN (D801)	1.6	
			U3E1\G10000=BIN (U3E1\G10001)	2.1	
			U3E1\HG10000=BIN (U3E1\HG10001)	1.9	
			#0L=BIN (#2L)	1.7	
			D800L=BIN (D802L)	1.7	
			U3E1\G10000L=BIN (U3E1\G10002L)	2.2	
			U3E1\HG10000L=BIN (U3E1\HG10002L)	2.0	
	BCD	BIN→BCD转换	#0=BCD (#1)	1.7	
			D800=BCD (D801)	1.7	
			U3E1\G10000=BCD (U3E1\G10001)	2.2	
			U3E1\HG10000=BCD (U3E1\HG10001)	2.0	
			#0L=BCD (#2L)	1.8	
			D800L=BCD (D802L)	1.8	
			U3E1\G10000L=BCD (U3E1\G10002L)	2.3	
			U3E1\HG10000L=BCD (U3E1\HG10002L)	2.1	
	数据控制	SCL	16位整数型标度	SCL K0, K2000, #0, #2002* ¹	6.9
				SCL K0, K2000, D2000, D4002* ¹	6.9
				SCL K0, K2000, U3E1\G10000, U3E1\G12002* ¹	17.4
				SCL K0, K2000, U3E1\HG0, U3E1\HG12002* ¹	14.8
				SCL K0, K2000, #0, #2002* ²	43.1
				SCL K0, K2000, D2000, D4002* ²	43.1
				SCL K0, K2000, U3E1\G10000, U3E1\G12002* ²	133.6
SCL K0, K2000, U3E1\HG0, U3E1\HG12002* ²				110.9	
SCL K0, K2000, #0, #2002* ³				405.4	
SCL K0, K2000, D2000, D4002* ³				405.3	
SCL K0, K2000, U3E1\G10000, U3E1\G12002* ³				1298.1	
SCL K0, K2000, U3E1\HG0, U3E1\HG12002* ³				1093.9	
SCL K2, K1, #0, #2002* ¹				4.5	
SCL K2, K1, D2000, D4002* ¹				4.6	
SCL K2, K1, U3E1\G10000, U3E1\G12002* ¹				9.6	
SCL K2, K1, U3E1\HG0, U3E1\HG12002* ¹				8.3	
DSCL		32位整数型标度	DSCL K0, K2000L, #0, #4002L* ¹	6.9	
			DSCL K0, K2000L, D2000, D6002L* ¹	6.9	
			DSCL K0, K2000L, U3E1\G10000, U3E1\G14002L* ¹	17.4	
			DSCL K0, K2000L, U3E1\HG0, U3E1\HG12002L* ¹	14.6	
			DSCL K0, K2000L, #0, #4002L* ²	42.8	
			DSCL K0, K2000L, D2000, D6002L* ²	43.5	
			DSCL K0, K2000L, U3E1\G10000, U3E1\G14002L* ²	135.9	
			DSCL K0, K2000L, U3E1\HG0, U3E1\HG12002L* ²	111.0	
			DSCL K0, K2000L, #0, #4002L* ³	415.4	
			DSCL K0, K2000L, D2000, D6002L* ³	415.3	
			DSCL K0, K2000L, U3E1\G10000, U3E1\G14002L* ³	1306.0	
DSCL K0, K2000L, U3E1\HG0, U3E1\HG12002L* ³	1074.1				
DSCL K2, K1L, #0, #4002L* ¹	4.6				
DSCL K2, K1L, D2000, D6002L* ¹	4.6				
DSCL K2, K1L, U3E1\G10000, U3E1\G14002L* ¹	9.8				
DSCL K2, K1L, U3E1\HG0, U3E1\HG12002L* ¹	8.3				

分类	符号	指令	运算公式	单位[μ s]
类型转换	SHORT	转换为16位整数型(带符号)	#0=SHORT (#2L)	1.6
			D800=SHORT (D802L)	1.6
			U3E1\G10000=SHORT (U3E1\G10002L)	2.1
			U3E1\HG10000=SHORT (U3E1\HG10002L)	1.9
			#0=SHORT (#4F)	1.7
			D800=SHORT (D804F)	1.7
			U3E1\G10000=SHORT (U3E1\G10004F)	2.2
			U3E1\HG10000=SHORT (U3E1\HG10004F)	2.0
	USHORT	转换为16位整数型(无符号)	#0=USHORT (#2L)	1.6
			D800=USHORT (D802L)	1.6
			U3E1\G10000=USHORT (U3E1\G10002L)	2.1
			U3E1\HG10000=USHORT (U3E1\HG10002L)	1.9
			#0=USHORT (#4F)	1.7
			D800=USHORT (D804F)	1.7
			U3E1\G10000=USHORT (U3E1\G10004F)	2.2
			U3E1\HG10000=USHORT (U3E1\HG10004F)	2.0
	LONG	转换为32位整数型(带符号)	#0L=LONG (#2)	1.6
			D800L=LONG (D802)	1.6
			U3E1\G10000L=LONG (U3E1\G10002)	2.1
			U3E1\HG10000L=LONG (U3E1\HG10002)	1.9
			#0L=LONG (#4F)	1.7
			D800L=LONG (D804F)	1.7
			U3E1\G10000L=LONG (U3E1\G10004F)	2.2
			U3E1\HG10000L=LONG (U3E1\HG10004F)	2.0
ULONG	转换为32位整数型(无符号)	#0L=ULONG (#2)	1.6	
		D800L=ULONG (D802)	1.6	
		U3E1\G10000L=ULONG (U3E1\G10002)	2.1	
		U3E1\HG10000L=ULONG (U3E1\HG10002)	1.9	
		#0L=ULONG (#4F)	1.7	
		D800L=ULONG (D804F)	1.7	
		U3E1\G10000L=ULONG (U3E1\G10004F)	2.2	
		U3E1\HG10000L=ULONG (U3E1\HG10004F)	2.0	
FLOAT	转换为64位浮点型(带符号)	#0F=FLOAT (#4)	1.6	
		D800F=FLOAT (D804)	1.6	
		U3E1\G10000F=FLOAT (U3E1\G10004)	2.1	
		U3E1\HG10000F=FLOAT (U3E1\HG10004)	1.9	
		#0F=FLOAT (#4L)	1.6	
		D800F=FLOAT (D804L)	1.6	
		U3E1\G10000F=FLOAT (U3E1\G10004L)	2.1	
		U3E1\HG10000F=FLOAT (U3E1\HG10004L)	1.9	
UFLOAT	转换为64位浮点型(无符号)	#0F=UFLOAT (#4)	1.6	
		D800F=UFLOAT (D804)	1.6	
		U3E1\G10000F=UFLOAT (U3E1\G10004)	2.1	
		U3E1\HG10000F=UFLOAT (U3E1\HG10004)	1.9	
		#0F=UFLOAT (#4L)	1.6	
		D800F=UFLOAT (D804L)	1.6	
		U3E1\G10000F=UFLOAT (U3E1\G10004L)	2.1	
		U3E1\HG10000F=UFLOAT (U3E1\HG10004L)	1.9	
DFLT	浮点值的32位→64位转换	#0F=DFLT (#4L)	1.6	
		D2000F=DFLT (D2004L)	1.6	
		U3E1\G10000F=DFLT (U3E1\G10004L)	2.1	
		U3E1\HG10000F=DFLT (U3E1\HG10004L)	2.0	

分类	符号	指令	运算公式	单位[μs]
类型转换	SFLT	浮点值的64位→32位转换	#0L=SFLT(#2F)	1.7
			D2000L=SFLT(D2002F)	1.7
			U3E1\G10000L=SFLT(U3E1\G10002F)	2.6
			U3E1\HG10000L=SFLT(U3E1\HG10002F)	2.4
位软元件状态	无	ON(常开触点) (条件成立时)	SET M1000=M0	1.4
			SET M1000=X100	1.5
			SET M1000=X20*4	4.6
			SET M1000=U3E1\G10000.0	2.0
			SET M1000=U3E1\HG10000.0	1.8
	!	OFF(常闭触点) (条件成立时)	SET M1000=!M0	1.5
			SET M1000=!X100	1.6
			SET M1000=!X20*4	4.7
			SET M1000=!U3E1\G10000.0	2.0
			SET M1000=!U3E1\HG10000.0	1.9
位软元件控制	SET	软元件的设置	SET M1000	0.9
			SET Y100	0.9
			SET Y60*4	3.2
			SET U3E1\G11000.0	1.5
			SET U3E1\HG11000.0	1.4
	RST	软元件的复位	RST M1000	0.9
			RST Y100	0.9
			RST Y60*4	3.2
			RST U3E1\G11000.0	1.5
			RST U3E1\HG11000.0	1.4
	DOUT	软元件的输出	DOUT M0, #0	1.5
			DOUT Y100, #0	1.5
			DOUT Y60, #0*4	4.0
			DOUT M0, #0L	1.5
			DOUT Y100, #0L	1.5
			DOUT Y60, #0L*4	4.1
	DIN	软元件的输入	DIN #0, M0	1.5
			DIN #0, X0	1.5
			DIN #0, X20*4	6.2
			DIN #0L, M0	1.5
			DIN #0L, X0	1.6
			DIN #0L, X20*4	6.5
	OUT	位软元件的输出	OUT M100=M0	1.4
			OUT Y0=M0	1.5
OUT Y60=M0*4			3.8	
OUT U3E1\G10000.0=M0			2.1	
OUT U3E1\HG10000.0=M0			1.9	
逻辑运算	*	逻辑积	SET M1000=M0*M1	2.0
			SET M1000=X100*X101	1.7
			SET M1000=X20*X21*4	9.5
			SET M1000=U3E1\G10000.0*U3E1\G10000.1	2.6
			SET M1000=U3E1\HG10000.0*U3E1\HG10000.1	2.3
	+	逻辑和	SET M1000=M0+M1	2.0
			SET M1000=X100+X101	1.7
			SET M1000=X20+X21*4	9.4
			SET M1000=U3E1\G10000.0+U3E1\G10000.1	3.1
			SET M1000=U3E1\HG10000.0+U3E1\HG10000.1	2.9

分类	符号	指令	运算公式	单位[μs]
比较运算	==	一致(条件成立时)	SET M1000=#0==#1	2.1
			SET M1000=D800==D801	2.1
			SET M1000=U3E1\G10000==U3E1\G10001	3.1
			SET M1000=U3E1\HG10000==U3E1\HG10001	2.8
			SET M1000=#0L==#2L	2.2
			SET M1000=D800L==D802L	2.2
			SET M1000=U3E1\G10000L==U3E1\G10002L	3.2
			SET M1000=U3E1\HG10000L==U3E1\HG10002L	2.9
			SET M1000=#0F==#4F	2.2
			SET M1000=D800F==D804F	2.2
			SET M1000=U3E1\G10000F==U3E1\G10004F	3.2
			SET M1000=U3E1\HG10000F==U3E1\HG10004F	2.9
	!=	不一致(条件成立时)	SET M1000=#0!=#1	2.1
			SET M1000=D800!=D801	2.1
			SET M1000=U3E1\G10000!=U3E1\G10001	3.1
			SET M1000=U3E1\HG10000!=U3E1\HG10001	2.9
			SET M1000=#0L!=#2L	2.1
			SET M1000=D800L!=D802L	2.2
			SET M1000=U3E1\G10000L!=U3E1\G10002L	3.1
			SET M1000=U3E1\HG10000L!=U3E1\HG10002L	2.9
			SET M1000=#0F!=#4F	2.2
			SET M1000=D800F!=D804F	2.2
			SET M1000=U3E1\G10000F!=U3E1\G10004F	3.2
			SET M1000=U3E1\HG10000F!=U3E1\HG10004F	2.9
	<	小于(条件成立时)	SET M1000=#0<#1	2.1
			SET M1000=D800<D801	2.1
			SET M1000=U3E1\G10000<U3E1\G10001	3.1
			SET M1000=U3E1\HG10000<U3E1\HG10001	2.8
			SET M1000=#0L<#2L	2.2
			SET M1000=D800L<D802L	2.2
			SET M1000=U3E1\G10000L<U3E1\G10002L	3.1
			SET M1000=U3E1\HG10000L<U3E1\HG10002L	2.9
			SET M1000=#0F<#4F	2.2
			SET M1000=D800F<D804F	2.2
			SET M1000=U3E1\G10000F<U3E1\G10004F	3.2
			SET M1000=U3E1\HG10000F<U3E1\HG10004F	2.9
<=	小于等于(条件成立时)	SET M1000=#0<=#1	2.1	
		SET M1000=D800<=D801	2.2	
		SET M1000=U3E1\G10000<=U3E1\G10001	3.1	
		SET M1000=U3E1\HG10000<=U3E1\HG10001	2.9	
		SET M1000=#0L<=#2L	2.2	
		SET M1000=D800L<=D802L	2.2	
		SET M1000=U3E1\G10000L<=U3E1\G10002L	3.2	
		SET M1000=U3E1\HG10000L<=U3E1\HG10002L	2.9	
		SET M1000=#0F<=#4F	2.2	
		SET M1000=D800F<=D804F	2.2	
		SET M1000=U3E1\G10000F<=U3E1\G10004F	3.2	
		SET M1000=U3E1\HG10000F<=U3E1\HG10004F	2.9	

分类	符号	指令	运算公式	单位[μs]
比较运算	>	大于(条件成立时)	SET M1000=#0>#1	2.1
			SET M1000=D800>D801	2.1
			SET M1000=U3E1\G10000>U3E1\G10001	3.1
			SET M1000=U3E1\HG10000>U3E1\HG10001	2.9
			SET M1000=#0L>#2L	2.1
			SET M1000=D800L>D802L	2.1
			SET M1000=U3E1\G10000L>U3E1\G10002L	3.1
			SET M1000=U3E1\HG10000L>U3E1\HG10002L	2.9
			SET M1000=#0F>#4F	2.2
			SET M1000=D800F>D804F	2.2
	SET M1000=U3E1\G10000F>U3E1\G10004F	3.2		
	SET M1000=U3E1\HG10000F>U3E1\HG10004F	2.9		
	>=	大于等于(条件成立时)	SET M1000=#0>=#1	2.1
			SET M1000=D800>=D801	2.1
			SET M1000=U3E1\G10000>=U3E1\G10001	3.1
			SET M1000=U3E1\HG10000>=U3E1\HG10001	2.8
			SET M1000=#0L>=#2L	2.1
			SET M1000=D800L>=D802L	2.1
			SET M1000=U3E1\G10000L>=U3E1\G10002L	3.2
			SET M1000=U3E1\HG10000L>=U3E1\HG10002L	2.9
SET M1000=#0F>=#4F			2.2	
SET M1000=D800F>=D804F			2.2	
SET M1000=U3E1\G10000F>=U3E1\G10004F	3.2			
SET M1000=U3E1\HG10000F>=U3E1\HG10004F	2.9			
程序控制	IF~ ELSE ~IEND	条件分支控制	IF #0==#1*5 #2=#3 ELSE #4=#5 IEND	3.3
			IF D800==D801*5 #2=#3 ELSE #4=#5 IEND	1.6
			IF U3E1\G10000==U3E1\G10001*5 #2=#3 ELSE #4=#5 IEND	2.1
			IF U3E1\HG10000==U3E1\HG10001*5 #2=#3 ELSE #4=#5 IEND	2.0
			IF #0==#1*6 #2=#3 ELSE #4=#5 IEND	1.6
			IF D800==D801*6 #2=#3 ELSE #4=#5 IEND	1.6
			IF U3E1\G10000==U3E1\G10001*6 #2=#3 ELSE #4=#5 IEND	2.1

分类	符号	指令	运算公式	单位[μs]		
程序控制	IF~ ELSE ~IEND	条件分支控制	IF U3E1\HG10000==U3E1\HG10001*6 #2=#3 ELSE #4=#5 IEND	2.0		
			SELECT~ CASE~ SEND	选择分支控制	SELECT*7 CASE #0==K1 #2=#3 CEND CASE #1==K1 #4=#5 CEND CELSE #6=#7 CEND SEND	3.1
					SELECT*7 CASE D800==K1 #2=#3 CEND CASE D801==K1 #4=#5 CEND CELSE #6=#7 CEND SEND	0.8
					SELECT*7 CASE U3E1\G10000==K1 #2=#3 CEND CASE U3E1\G10001==K1 #4=#5 CEND CELSE #6=#7 CEND SEND	0.9
					SELECT*7 CASE U3E1\HG10000==K1 #2=#3 CEND CASE U3E1\HG10001==K1 #4=#5 CEND CELSE #6=#7 CEND SEND	0.9
					SELECT*8 CASE #0==K1 #2=#3 CEND CASE #1==K1 #4=#5 CEND CELSE #6=#7 CEND SEND	1.0

附

分类	符号	指令	运算公式	单位[μs]
程序控制	SELECT~ CASE~ SEND	选择分支控制	SELECT*8 CASE D800==K1 #2=#3 CEND CASE D801==K1 #4=#5 CEND CELSE #6=#7 CEND SEND	1.0
			SELECT*8 CASE U3E1\G10000==K1 #2=#3 CEND CASE U3E1\G10001==K1 #4=#5 CEND CELSE #6=#7 CEND SEND	1.3
			SELECT*8 CASE U3E1\HG10000==K1 #2=#3 CEND CASE U3E1\HG10001==K1 #4=#5 CEND CELSE #6=#7 CEND SEND	1.2
			SELECT*9 CASE #0==K1 #2=#3 CEND CASE #1==K1 #4=#5 CEND CELSE #6=#7 CEND SEND	1.1
			SELECT*9 CASE D800==K1 #2=#3 CEND CASE D801==K1 #4=#5 CEND CELSE #6=#7 CEND SEND	1.1
			SELECT*9 CASE U3E1\G10000==K1 #2=#3 CEND CASE U3E1\G10001==K1 #4=#5 CEND CELSE #6=#7 CEND SEND	1.4
			SELECT*9 CASE U3E1\G10000==K1 #2=#3 CEND CASE U3E1\G10001==K1 #4=#5 CEND CELSE #6=#7 CEND SEND	1.4

分类	符号	指令	运算公式	单位[μs]		
程序控制	SELECT~ CASE~ SEND	选择分支控制	SELECT* ⁹ CASE U3E1\HG10000==K1 #2=#3 CEND CASE U3E1\HG10001==K1 #4=#5 CEND CELSE #6=#7 CEND SEND	1.3		
			FOR~NEXT	次数指定重复控制	FOR #0=K1 TO 10 #1=#1+1 NEXT	47.6
					FOR D800=K1 TO 10 #1=#1+1 NEXT	23.8
					FOR U3E1\G10000=K1 TO 10 #1=#1+1 NEXT	29.1
FOR U3E1\HG10000=K1 TO 10 #1=#1+1 NEXT	27.6					
运动专用函数	CHGV	速度更改请求	CHGV (K1, #0)	1.3		
			CHGV (K1, D800)	1.3		
			CHGV (K1, U3E1\G10000)	1.8		
			CHGV (K1, U3E1\HG10000)	1.7		
			CHGV (K1, #0L)	1.3		
			CHGV (K1, D800L)	1.3		
			CHGV (K1, U3E1\G10000L)	1.8		
			CHGV (K1, U3E1\HG10000L)	1.7		
	CHGVS	指令生成轴速度更改请求	CHGVS (K1, #0)	1.3		
			CHGVS (K1, D800)	1.3		
			CHGVS (K1, U3E1\G10000)	1.8		
			CHGVS (K1, U3E1\HG10000)	1.7		
			CHGVS (K1, #0L)	1.3		
			CHGVS (K1, D800L)	1.3		
			CHGVS (K1, U3E1\G10000L)	1.8		
			CHGVS (K1, U3E1\HG10000L)	1.7		
	CHGT	转矩限制值更改请求	CHGT (K1, #0, #1)	1.6		
			CHGT (K1, D800, D801)	1.6		
			CHGT (K1, U3E1\G10000, U3E1\G10001)	2.6		
			CHGT (K1, U3E1\HG10000, U3E1\HG10001)	2.3		
			CHGT (K1, #0L, #2L)	1.7		
			CHGT (K1, D800L, D802L)	1.7		
			CHGT (K1, U3E1\G10000L, U3E1\G10002L)	2.7		
			CHGT (K1, U3E1\HG10000L, U3E1\HG10002L)	2.4		
	CHGP	目标位置更改请求	CHGP (K1, K1, #0) * ¹⁰	3.0		
			CHGP (K1, K1, D800) * ¹⁰	3.0		
			CHGP (K1, K1, U3E1\G10000) * ¹⁰	4.2		
			CHGP (K1, K1, U3E1\HG10000) * ¹⁰	3.8		
CHGP (K1, K1, #0) * ¹¹			2.0			
CHGP (K1, K1, D800) * ¹¹			1.9			
CHGP (K1, K1, U3E1\G10000) * ¹¹			4.2			
CHGP (K1, K1, U3E1\HG10000) * ¹¹			3.8			

分类	符号	指令	运算公式	单位[μs]
运动专用函数	MCNST	机器程序运行启动请求	MCNST (#13000, #12998)*12	153.2
			MCNST (D58000, D57998)*12	144.0
			MCNST (U3E1\G10002, U3E1\G10000)*12	165.2
			MCNST (U3E1\HG5002, U3E1\HG5000)*12	160.6
			MCNST (#13000, #12998)*13	1187.7
			MCNST (D58000, D57998)*13	1195.1
			MCNST (U3E1\G10002, U3E1\G10000)*13	2221.8
			MCNST (U3E1\HG5002, U3E1\HG5000)*13	2050.2
高级同步控制专用函数	CAMRD	凸轮数据读取	CAMRD #0, #2L, K256, #4*14	52.0
			CAMRD D2000, D2002L, K256, D2004*14	46.1
			CAMRD U3E1\G10000, U3E1\G10002L, K256, U3E1\G10004*14	19.5
			CAMRD U3E1\HG10000, U3E1\HG10002L, K256, U3E1\HG0*14	19.2
			CAMRD #0, #2L, K1024, #4*14	190.5
			CAMRD D2000, D2002L, K1024, D2004*14	191.0
			CAMRD U3E1\G10000, U3E1\G10002L, K1024, U3E1\G10004*14	58.6
			CAMRD U3E1\HG10000, U3E1\HG10002L, K1024, U3E1\HG0*14	58.2
			CAMRD #0, #2L, K2048, #4*14	382.5
			CAMRD D2000, D2002L, K2048, D2004*14	383.1
			CAMRD U3E1\G10000, U3E1\G10002L, K2048, U3E1\G10004*14	113.0
			CAMRD U3E1\HG10000, U3E1\HG10002L, K2048, U3E1\HG0*14	112.9
			CAMRD #0, #2L, K256, #4*15	95.0
			CAMRD D2000, D2002L, K256, D2004*15	94.2
			CAMRD U3E1\G10000, U3E1\G10002L, K256, U3E1\G10004*15	31.2
			CAMRD U3E1\HG10000, U3E1\HG10002L, K256, U3E1\HG0*15	31.0
			CAMRD #0, #2L, K512, #4*15	190.6
			CAMRD D2000, D2002L, K512, D2004*15	191.0
			CAMRD U3E1\G10000, U3E1\G10002L, K512, U3E1\G10004*15	58.6
			CAMRD U3E1\HG10000, U3E1\HG10002L, K512, U3E1\HG0*15	58.3
			CAMRD #0, #2L, K1024, #4*15	382.4
			CAMRD D2000, D2002L, K1024, D2004*15	383.1
			CAMRD U3E1\G10000, U3E1\G10002L, K1024, U3E1\G10004*15	113.3
			CAMRD U3E1\HG10000, U3E1\HG10002L, K1024, U3E1\HG0*15	113.0

分类	符号	指令	运算公式	单位[μs]		
高级同步控制 专用函数	CAMWR	凸轮数据写入	CAMWR #0, #2L, K256, #4, H401* ¹⁴	27.3		
			CAMWR D2000, D2002L, K256, D2004, H401* ¹⁴	27.1		
			CAMWR U3E1\G10000, U3E1\G10002L, K256, U3E1\G10004, H401* ¹⁴	107.2		
			CAMWR U3E1\HG10000, U3E1\HG10002L, K256, U3E1\HG0, H401* ¹⁴	93.1		
			CAMWR #0, #2L, K1024, #4, H401* ¹⁴	74.5		
			CAMWR D2000, D2002L, K1024, D2004, H401* ¹⁴	72.0		
			CAMWR U3E1\G10000, U3E1\G10002L, K1024, U3E1\G10004, H401* ¹⁴	362.1		
			CAMWR U3E1\HG10000, U3E1\HG10002L, K1024, U3E1\HG0, H401* ¹⁴	308.5		
			CAMWR #0, #2L, K2048, #4, H401* ¹⁴	137.6		
			CAMWR D2000, D2002L, K2048, D2004, H401* ¹⁴	135.8		
			CAMWR U3E1\G10000, U3E1\G10002L, K2048, U3E1\G10004, H401* ¹⁴	702.1		
			CAMWR U3E1\HG10000, U3E1\HG10002L, K2048, U3E1\HG0, H401* ¹⁴	596.6		
			CAMWR #0, #2L, K256, #4, H401* ¹⁵	45.1		
			CAMWR D2000, D2002L, K256, D2004, H401* ¹⁵	46.0		
			CAMWR U3E1\G10000, U3E1\G10002L, K256, U3E1\G10004, H401* ¹⁵	196.9		
			CAMWR U3E1\HG10000, U3E1\HG10002L, K256, U3E1\HG0, H401* ¹⁵	169.7		
			CAMWR #0, #2L, K512, #4, H401* ¹⁵	81.8		
			CAMWR D2000, D2002L, K512, D2004, H401* ¹⁵	81.0		
			CAMWR U3E1\G10000, U3E1\G10002L, K512, U3E1\G10004, H401* ¹⁵	370.6		
			CAMWR U3E1\HG10000, U3E1\HG10002L, K512, U3E1\HG0, H401* ¹⁵	317.5		
			CAMWR #0, #2L, K1024, #4, H401* ¹⁵	153.4		
			CAMWR D2000, D2002L, K1024, D2004, H401* ¹⁵	152.6		
			CAMWR U3E1\G10000, U3E1\G10002L, K1024, U3E1\G10004, H401* ¹⁵	719.9		
			CAMWR U3E1\HG10000, U3E1\HG10002L, K1024, U3E1\HG0, H401* ¹⁵	614.3		
			CAMWR	凸轮数据写入 (凸轮展开区域)	CAMWR #0, #2L, K256, #4, H0* ¹⁴	23.5
					CAMWR D2000, D2002L, K256, D2004, H0* ¹⁴	20.4
					CAMWR U3E1\G10000, U3E1\G10002L, K256, U3E1\G10004, H0* ¹⁴	100.6
					CAMWR U3E1\HG10000, U3E1\HG10002L, K256, U3E1\HG0, H0* ¹⁴	86.3
					CAMWR #0, #2L, K1024, #4, H0* ¹⁴	67.4
					CAMWR D2000, D2002L, K1024, D2004, H0* ¹⁴	63.1
					CAMWR U3E1\G10000, U3E1\G10002L, K1024, U3E1\G10004, H0* ¹⁴	348.7
					CAMWR U3E1\HG10000, U3E1\HG10002L, K1024, U3E1\HG0, H0* ¹⁴	294.9
					CAMWR #0, #2L, K2048, #4, H0* ¹⁴	128.1
					CAMWR D2000, D2002L, K2048, D2004, H0* ¹⁴	123.5
					CAMWR U3E1\G10000, U3E1\G10002L, K2048, U3E1\G10004, H0* ¹⁴	684.8
					CAMWR U3E1\HG10000, U3E1\HG10002L, K2048, U3E1\HG0, H0* ¹⁴	578.6
CAMWR #0, #2L, K256, #4, H0* ¹⁵	39.2					
CAMWR D2000, D2002L, K256, D2004, H0* ¹⁵	38.4					
CAMWR U3E1\G10000, U3E1\G10002L, K256, U3E1\G10004, H0* ¹⁵	184.9					
CAMWR U3E1\HG10000, U3E1\HG10002L, K256, U3E1\HG0, H0* ¹⁵	157.4					
CAMWR #0, #2L, K512, #4, H0* ¹⁵	72.6					
CAMWR D2000, D2002L, K512, D2004, H0* ¹⁵	71.3					
CAMWR U3E1\G10000, U3E1\G10002L, K512, U3E1\G10004, H0* ¹⁵	356.1					
CAMWR U3E1\HG10000, U3E1\HG10002L, K512, U3E1\HG0, H0* ¹⁵	302.7					
CAMWR #0, #2L, K1024, #4, H0* ¹⁵	139.9					
CAMWR D2000, D2002L, K1024, D2004, H0* ¹⁵	138.7					
CAMWR U3E1\G10000, U3E1\G10002L, K1024, U3E1\G10004, H0* ¹⁵	700.1					
CAMWR U3E1\HG10000, U3E1\HG10002L, K1024, U3E1\HG0, H0* ¹⁵	594.0					

分类	符号	指令	运算公式	单位[μs]
高级同步控制专用函数	CAMMK	凸轮自动生成功能	CAMMK #0, #1, #2*16	275.6
			CAMMK D2000, D2001, D2002*16	258.3
			CAMMK U3E1\G10000, U3E1\G10001, U3E1\G10002*16	263.0
			CAMMK U3E1\HG10000, U3E1\HG10001, U3E1\HG10002*16	262.0
			CAMMK #0, #1, #2*17	7981.1
			CAMMK D2000, D2001, D2002*17	7938.2
			CAMMK U3E1\G10000, U3E1\G10001, U3E1\G10002*17	7928.1
			CAMMK U3E1\HG10000, U3E1\HG10001, U3E1\HG10002*17	7927.5
			CAMMK #0, #1, #2*18	31856.3
			CAMMK D2000, D2001, D2002*18	31717.5
			CAMMK U3E1\G10000, U3E1\G10001, U3E1\G10002*18	31664.8
			CAMMK U3E1\HG10000, U3E1\HG10001, U3E1\HG10002*18	31667.4
			CAMMK #0, #1, #2*19	1249.2
			CAMMK D2000, D2001, D2002*19	1130.7
			CAMMK U3E1\G10000, U3E1\G10001, U3E1\G10002*19	1144.3
			CAMMK U3E1\HG10000, U3E1\HG10001, U3E1\HG10002*19	1141.0
			CAMMK #0, #1, #2*20	29793.0
			CAMMK D2000, D2001, D2002*20	29754.1
			CAMMK U3E1\G10000, U3E1\G10001, U3E1\G10002*20	29777.7
			CAMMK U3E1\HG10000, U3E1\HG10001, U3E1\HG10002*20	29770.4
			CAMMK #0, #1, #2*21	118553.4
			CAMMK D2000, D2001, D2002*21	118480.0
			CAMMK U3E1\G10000, U3E1\G10001, U3E1\G10002*21	118557.6
			CAMMK U3E1\HG10000, U3E1\HG10001, U3E1\HG10002*21	118515.2
			CAMMK #0, #1, #2*22	1804.5
			CAMMK D2000, D2001, D2002*22	1772.6
			CAMMK U3E1\G10000, U3E1\G10001, U3E1\G10002*22	1811.0
			CAMMK U3E1\HG10000, U3E1\HG10001, U3E1\HG10002*22	1801.2
			CAMMK #0, #1, #2*23	41531.6
			CAMMK D2000, D2001, D2002*23	41506.9
			CAMMK U3E1\G10000, U3E1\G10001, U3E1\G10002*23	41549.0
			CAMMK U3E1\HG10000, U3E1\HG10001, U3E1\HG10002*23	41540.1
CAMMK #0, #1, #2*24	164697.7			
CAMMK D2000, D2001, D2002*24	164645.2			
CAMMK U3E1\G10000, U3E1\G10001, U3E1\G10002*24	164697.9			
CAMMK U3E1\HG10000, U3E1\HG10001, U3E1\HG10002*24	164674.1			

分类	符号	指令	运算公式	单位[μ s]
高级同步控制专用函数	CAMPSCL	凸轮位置计算	CAMPSCL #0, #2, #14L*25*27	8.6
			CAMPSCL D2000, D2002, D2014L*25*27	5.6
			CAMPSCL U3E1\G10000, U3E1\G10002, U3E1\G10014L*25*27	10.6
			CAMPSCL U3E1\HG10000, U3E1\HG10002, U3E1\HG10014L*25*27	10.0
			CAMPSCL #0, #2, #14L*25*28	5.8
			CAMPSCL D2000, D2002, D2014L*25*28	5.5
			CAMPSCL U3E1\G10000, U3E1\G10002, U3E1\G10014L*25*28	9.5
			CAMPSCL U3E1\HG10000, U3E1\HG10002, U3E1\HG10014L*25*28	8.7
			CAMPSCL #0, #2, #14L*25*29	7.1
			CAMPSCL D2000, D2002, D2014L*25*29	5.7
			CAMPSCL U3E1\G10000, U3E1\G10002, U3E1\G10014L*25*29	9.8
			CAMPSCL U3E1\HG10000, U3E1\HG10002, U3E1\HG10014L*25*29	8.9
			CAMPSCL #0, #2, #14L*25*30	7.9
			CAMPSCL D2000, D2002, D2014L*25*30	6.2
			CAMPSCL U3E1\G10000, U3E1\G10002, U3E1\G10014L*25*30	10.1
			CAMPSCL U3E1\HG10000, U3E1\HG10002, U3E1\HG10014L*25*30	9.3
			CAMPSCL #0, #2, #14L*26*27	31.6
			CAMPSCL D2000, D2002, D2014L*26*27	29.0
			CAMPSCL U3E1\G10000, U3E1\G10002, U3E1\G10014L*26*27	32.7
			CAMPSCL U3E1\HG10000, U3E1\HG10002, U3E1\HG10014L*26*27	32.0
			CAMPSCL #0, #2, #14L*26*28	731.4
			CAMPSCL D2000, D2002, D2014L*26*28	708.1
			CAMPSCL U3E1\G10000, U3E1\G10002, U3E1\G10014L*26*28	711.0
			CAMPSCL U3E1\HG10000, U3E1\HG10002, U3E1\HG10014L*26*28	710.2
			CAMPSCL #0, #2, #14L*26*29	15.3
			CAMPSCL D2000, D2002, D2014L*26*29	12.5
			CAMPSCL U3E1\G10000, U3E1\G10002, U3E1\G10014L*26*29	16.1
			CAMPSCL U3E1\HG10000, U3E1\HG10002, U3E1\HG10014L*26*29	15.4
			CAMPSCL #0, #2, #14L*26*30	221.3
			CAMPSCL D2000, D2002, D2014L*26*30	176.2
CAMPSCL U3E1\G10000, U3E1\G10002, U3E1\G10014L*26*30	180.3			
CAMPSCL U3E1\HG10000, U3E1\HG10002, U3E1\HG10014L*26*30	179.6			
视觉系统专用函数	MVOPEN	线路打开	MVOPEN K1, K1000	152.8
			MVOPEN #0, #1	173.5
			MVOPEN D2000, D2001	173.5
			MVOPEN U3E1\G10000, U3E1\G10001	173.9
			MVOPEN U3E1\HG10000, U3E1\HG10001	173.8
	MVLOAD	视觉程序装载	MVLOAD K1, K1000	49.6
			MVLOAD #0, #1	50.5
			MVLOAD D2000, D2001	64.8
			MVLOAD U3E1\G10000, U3E1\G10001	50.6
			MVLOAD U3E1\HG10000, U3E1\HG10001	49.7
	MVTRG	触发发行	MVTRG K1, K1000	59.0
			MVTRG #0, #1	62.3
			MVTRG D2000, D2001	60.1
			MVTRG U3E1\G10000, U3E1\G10001	69.7
			MVTRG U3E1\HG10000, U3E1\HG10001	61.7
	MVPST	视觉程序启动	MVPST K1, K1000	53.4
			MVPST #0, #1	49.6
			MVPST D2000, D2001	50.6
			MVPST U3E1\G10000, U3E1\G10001	62.4
			MVPST U3E1\HG10000, U3E1\HG10001	61.9

分类	符号	指令	运算公式	单位[μs]
视觉系统专用函数	MVIN	数据导入	MVIN K1, "A1", #0L, K1000	76.6
			MVIN D2000, D2001, #0L, K1000* ³¹	81.6
			MVIN D2000, D2001, #0L, K1000* ³²	60.7
			MVIN U3E1\G10000, U3E1\G10001, U3E1\G10020L, K1000* ³²	65.9
			MVIN U3E1\HG10000, U3E1\HG10001, U3E1\HG10020L, K1000* ³²	65.3
	MVOUT	数据导出	MVOUT K1, "A1", #0L, K1000	60.5
			MVOUT D2000, D2001, #0L, K1000* ³³	63.1
			MVOUT D2000, D2001, #0L, K1000* ³⁴	67.0
			MVOUT U3E1\G10000, U3E1\G10001, U3E1\G10020L, K1000* ³⁴	79.4
			MVOUT U3E1\HG10000, U3E1\HG10001, U3E1\HG10020L, K1000* ³⁴	78.2
	MVFIN	状态存储软元件复位	MVFIN K1	2.0
			MVFIN #0	2.0
			MVFIN D2000	2.0
			MVFIN U3E1\G10000	2.0
			MVFIN U3E1\HG10000	2.0
	MVCLOSE	线路关闭	MVCLOSE K1	90.2
			MVCLOSE #0	92.2
			MVCLOSE D2000	92.0
			MVCLOSE U3E1\G10000	91.5
			MVCLOSE U3E1\HG10000	88.8
	MVCOM	任意原生模式指令发送	MVCOM K1, "GO", #0, KO, K1000	114.9
			MVCOM D2000, D2001, #0, D2100, K1000* ³⁵	121.3
			MVCOM D2000, D2001, #0, D2100, K1000* ³⁶	132.2
			MVCOM U3E1\G10000, U3E1\G10002, U3E1\G11000, U3E1\G10001, K1000* ³⁶	138.9
			MVCOM U3E1\HG10000, U3E1\HG10002, U3E1\HG11000, U3E1\HG10001, K1000* ³⁶	126.9
其它	EI	事件任务允许	EI	0.6
	DI	事件任务禁止	DI	0.6
	NOP	无处理	NOP	0.2
	BMOV	块传送	BMOV #0, #100, K10	2.8
			BMOV D800, D100, K10	2.8
			BMOV U3E1\G10000, U3E1\G10100, K10	5.1
			BMOV U3E1\HG10000, U3E1\HG10100, K10	4.5
			BMOV #0, #100, K100	9.4
			BMOV D800, D100, K100	9.4
			BMOV U3E1\G10000, U3E1\G10100, K100	15.6
			BMOV U3E1\HG10000, U3E1\HG10100, K100	13.0
	FMOV	同一数据块传送	FMOV #0, #100, K10	2.3
			FMOV D800, D100, K10	2.4
FMOV U3E1\G10000, U3E1\G10100, K10			3.0	
FMOV U3E1\HG10000, U3E1\HG10100, K10			2.8	
FMOV #0, #100, K100			9.4	
FMOV D800, D100, K100			9.4	
FMOV U3E1\G10000, U3E1\G10100, K100			5.0	
FMOV U3E1\HG10000, U3E1\HG10100, K100			4.8	

分类	符号	指令	运算公式	单位[μ s]		
其它	TO	缓冲存储器的字数据写入	TO H0020, H0, #0, K1	4.5		
			TO H0020, H0, D800, K1	4.5		
			TO H0020, H0, U3E1\G10000, K1	5.0		
			TO H0020, H0, U3E1\HG10000, K1	4.9		
			TO H0020, H0, #0, K10	5.4		
			TO H0020, H0, D800, K10	5.4		
			TO H0020, H0, U3E1\G10000, K10	6.8		
			TO H0020, H0, U3E1\HG10000, K10	6.4		
			TO H0020, H0, #0, K100	6.7		
			TO H0020, H0, D800, K100	6.8		
			TO H0020, H0, U3E1\G10000, K100	17.9		
			TO H0020, H0, U3E1\HG10000, K100	15.3		
			TO H0020, H0, #0, K256	9.2		
			TO H0020, H0, D800, K256	9.2		
			TO H0020, H0, U3E1\G10000, K256	37.5		
			TO H0020, H0, U3E1\HG10000, K256	30.8		
			FROM	缓冲存储器的字数据读取	FROM #0, H0060, H0, K1	6.6
					FROM D800, H0060, H0, K1	6.6
	FROM U3E1\G10000, H0060, H0, K1	6.7				
	FROM U3E1\HG10000, H0060, H0, K1	6.7				
	FROM #0, H0060, #0, K10	8.6				
	FROM D800, H0060, H0, K10	8.6				
	FROM U3E1\G10000, H0060, H0, K10	8.7				
	FROM U3E1\HG10000, H0060, H0, K10	8.6				
	FROM #0, H0060, #0, K100	26.0				
	FROM D800, H0060, H0, K100	26.0				
	FROM U3E1\G10000, H0060, H0, K100	23.6				
	FROM U3E1\HG10000, H0060, H0, K100	23.4				
	FROM #0, H0060, H0, K256	55.3				
	FROM D800, H0060, H0, K256	55.3				
	FROM U3E1\G10000, H0060, H0, K256	46.3				
	FROM U3E1\HG10000, H0060, H0, K256	46.3				

分类	符号	指令	运算公式	单位[μs]
其它	RTO	对象站缓冲存储器的字数据写入*37	RTO #4000, #4001, #4002, #0, K1, M0	5.9
			RTO D2000, D2001, D2002, D800, K1, M0	5.2
			RTO U3E1\G12000, U3E1\G12001, U3E1\G12002, U3E1\G10000, K1, M0	7.1
			RTO U3E1\HG12000, U3E1\HG12001, U3E1\HG12002, U3E1\HG10000, K1, M0	6.7
			RTO #4000, #4001, #4002, #0, K10, M0	5.0
			RTO D2000, D2001, D2002, D800, K10, M0	5.0
			RTO U3E1\G12000, U3E1\G12001, U3E1\G12002, U3E1\G10000, K10, M0	7.1
			RTO U3E1\HG12000, U3E1\HG12001, U3E1\HG12002, U3E1\HG10000, K10, M0	6.7
			RTO #4000, #4001, #4002, #0, K100, M0	5.1
			RTO D2000, D2001, D2002, D800, K100, M0	5.4
			RTO U3E1\G12000, U3E1\G12001, U3E1\G12002, U3E1\G10000, K100, M0	7.1
			RTO U3E1\HG12000, U3E1\HG12001, U3E1\HG12002, U3E1\HG10000, K100, M0	6.8
			RTO #4000, #4001, #4002, #0, K240, M0	5.2
			RTO D2000, D2001, D2002, D800, K240, M0	5.3
			RTO U3E1\G12000, U3E1\G12001, U3E1\G12002, U3E1\G10000, K240, M0	7.4
			RTO U3E1\HG12000, U3E1\HG12001, U3E1\HG12002, U3E1\HG10000, K240, M0	7.1
	RFROM	至对象站缓冲存储器的字数据读取*37	RFROM #0, #4000, #4001, #4002, K1, M0	5.3
			RFROM D800, D2000, D2001, D2002, K1, M0	5.0
			RFROM U3E1\G10000, U3E1\G12000, U3E1\G12001, U3E1\G12002, K1, M0	7.1
			RFROM U3E1\HG10000, U3E1\HG12000, U3E1\HG12001, U3E1\HG12002, K1, M0	6.8
			RFROM #0, #4000, #4001, #4002, K10, M0	4.9
			RFROM D800, D2000, D2001, D2002, K10, M0	4.9
			RFROM U3E1\G10000, U3E1\G12000, U3E1\G12001, U3E1\G12002, K10, M0	7.0
			RFROM U3E1\HG10000, U3E1\HG12000, U3E1\HG12001, U3E1\HG12002, K10, M0	6.6
			RFROM #0, #4000, #4001, #4002, K100, M0	5.0
			RFROM D800, D2000, D2001, D2002, K100, M0	5.2
			RFROM U3E1\G10000, U3E1\G12000, U3E1\G12001, U3E1\G12002, K100, M0	7.1
			RFROM U3E1\HG10000, U3E1\HG12000, U3E1\HG12001, U3E1\HG12002, K100, M0	6.8
			RFROM #0, #4000, #4001, #4002, K240, M0	5.2
			RFROM D800, D2000, D2001, D2002, K240, M0	5.3
			RFROM U3E1\G10000, U3E1\G12000, U3E1\G12001, U3E1\G12002, K240, M0	7.4
			RFROM U3E1\HG10000, U3E1\HG12000, U3E1\HG12001, U3E1\HG12002, K240, M0	7.1
	TIME	时间等待	TIME K1	1.9
			TIME #0	2.2
			TIME D800	2.2
			TIME U3E1\G10000	3.0
			TIME U3E1\HG10000	2.9

- *1 标度用转换数据的查找次数为10次
- *2 标度用转换数据的查找次数为100次
- *3 标度用转换数据的查找次数为1000次
- *4 设置实际输入/实际输出
- *5 在IF(S)~ELSE~IEND的(S)中设置真数据

- *6 在IF(S)~ELSE~IEND的(S)中设置假数据
- *7 在SELECT CASE(S1)~CEND CASE(S2)~CEND CELSE~CEND SEND的(S1)中指定真数据
- *8 在SELECT CASE(S1)~CEND CASE(S2)~CEND CELSE~CEND SEND的(S1)中设置假数据，在(S2)中设置真数据
- *9 在SELECT CASE(S1)~CEND CASE(S2)~CEND CELSE~CEND SEND的(S1)及(S2)中设置假数据
- *10 执行1轴的直线定位控制时
- *11 执行4轴的直线插补控制时
- *12 执行1点的机器程序运行
- *13 执行128点的机器程序运行
- *14 凸轮数据为行程比数据形式
- *15 凸轮数据为坐标数据形式
- *16 凸轮自动生成类型为旋转刀具用凸轮，凸轮分辨率为256，自动生成选项为加减速方式S字加减速
- *17 凸轮自动生成类型为旋转刀具用凸轮，凸轮分辨率为8192，自动生成选项为加减速方式S字加减速
- *18 凸轮自动生成类型为旋转刀具用凸轮，凸轮分辨率为32768，自动生成选项为加减速方式S字加减速
- *19 凸轮自动生成类型为简易行程比凸轮，区函数8，凸轮分辨率为256，凸轮曲线为修正正弦
- *20 凸轮自动生成类型为简易行程比凸轮，区函数8，凸轮分辨率为8192，凸轮曲线为修正正弦
- *21 凸轮自动生成类型为简易行程比凸轮，区函数8，凸轮分辨率为32768，凸轮曲线为修正正弦
- *22 凸轮自动生成类型为简易行程比凸轮，区函数32，凸轮分辨率为256，凸轮曲线为修正正弦
- *23 凸轮自动生成类型为简易行程比凸轮，区函数32，凸轮分辨率为8192，凸轮曲线为修正正弦
- *24 凸轮自动生成类型为简易行程比凸轮，区函数32，凸轮分辨率为32768，凸轮曲线为修正正弦
- *25 凸轮位置计算类型为凸轮轴进给当前值计算
- *26 凸轮位置计算类型为凸轮轴1循环当前值计算
- *27 凸轮数据为行程比数据形式，分辨率为256，计算中间点(128)
- *28 凸轮数据为行程比数据形式，分辨率为8192，计算中间点(4096)
- *29 凸轮数据为坐标数据形式，坐标数为256，计算中间点(128)
- *30 凸轮数据为坐标数据形式，坐标数为8192，计算中间点(4096)
- *31 在MVIN (S1), (S2), (D), (S3)的(S2)中设置2字节的字符串
- *32 在MVIN (S1), (S2), (D), (S3)的(S2)中设置32字节的字符串
- *33 在MVOUT (S1), (S2), (S3), (S4)的(S2)中设置2字节的字符串
- *34 在MVOUT (S1), (S2), (S3), (S4)的(S2)中设置32字节的字符串
- *35 在MVCOM (S1), (S2), (D), (S3), (S4)的(S2)中设置2字节的字符串
- *36 在MVCOM (S1), (S2), (D), (S3), (S4)的(S2)中设置191字节的字符串
- *37 是运动CPU中的处理时间，不是数据传送完成为止的时间。

转移条件表达式

分类	符号	指令	运算公式	单位 [μ s]
位软元件状态	无	ON(常开触点) (条件成立时)	M0	0.6
			X100	1.3
			X20* ¹	6.7
			U3E1\G10000.0	1.8
			U3E1\HG10000.0	1.6
	!	OFF(常闭触点) (条件成立时)	!M0	0.8
			!X100	1.4
			!X20* ¹	6.9
			!U3E1\G10000.0	1.9
			!U3E1\HG10000.0	1.8
逻辑运算	*	逻辑积	M0*M1	1.4
			X100*X101	2.1
			X20*X20* ¹	11.1
			U3E1\G10000.0*U3E1\G10000.1	2.9
			U3E1\HG10000.0*U3E1\HG10000.1	2.7
	+	逻辑和	M0+M1	1.3
			X100+X101	2.1
			X20+X20* ¹	11.1
			U3E1\G10000.0+U3E1\G10000.1	2.9
			U3E1\HG10000.0+U3E1\HG10000.1	2.7
比较运算	==	一致(条件成立时)	#0==#1	1.6
			D800==D801	1.5
			U3E1\G10000==U3E1\G10001	2.9
			U3E1\HG10000==U3E1\HG10001	2.7
			#0L==#2L	2.0
			D800L==D802L	1.7
			U3E1\G10000L==U3E1\G10002L	3.1
			U3E1\HG10000L==U3E1\HG10002L	2.8
			#0F==#4F	1.8
			D800F==D804F	1.8
	U3E1\G10000F==U3E1\G10004F	3.1		
	U3E1\HG10000F==U3E1\HG10004F	2.9		
	!=	不一致(条件成立时)	#0!=#1	1.6
			D800!=D801	1.6
			U3E1\G10000!=U3E1\G10001	2.9
			U3E1\HG10000!=U3E1\HG10001	2.7
			#0L!=#2L	1.8
			D800L!=D802L	1.7
			U3E1\G10000L!=U3E1\G10002L	3.1
			U3E1\HG10000L!=U3E1\HG10002L	2.8
#0F!=#4F			3.5	
D800F!=D804F			1.8	
U3E1\G10000F!=U3E1\G10004F	3.2			
U3E1\HG10000F!=U3E1\HG10004F	2.9			

分类	符号	指令	运算公式	单位[μs]		
比较运算	<	小于(条件成立时)	#0<#1	1.6		
			D800<D801	1.5		
			U3E1\G10000<U3E1\G10001	2.9		
			U3E1\HG10000<U3E1\HG10001	2.6		
			#0L<#2L	1.8		
			D800L<D802L	1.7		
			U3E1\G10000L<U3E1\G10002L	3.2		
			U3E1\HG10000L<U3E1\HG10002L	2.9		
			#0F<#4F	1.8		
			D800F<D804F	1.8		
			U3E1\G10000F<U3E1\G10004F	3.1		
			U3E1\HG10000F<U3E1\HG10004F	2.9		
			<=	小于等于(条件成立时)	#0<=#1	1.6
					D800<=D801	1.6
	U3E1\G10000<=U3E1\G10001	3.0				
	U3E1\HG10000<=U3E1\HG10001	2.7				
	#0L<=#2L	1.8				
	D800L<=D802L	1.8				
	U3E1\G10000L<=U3E1\G10002L	3.2				
	U3E1\HG10000L<=U3E1\HG10002L	2.9				
	#0F<=#4F	1.9				
	D800F<=D804F	1.8				
	U3E1\G10000F<=U3E1\G10004F	3.2				
	U3E1\HG10000F<=U3E1\HG10004F	3.0				
	>	大于(条件成立时)			#0>#1	1.6
					D800>D801	1.5
			U3E1\G10000>U3E1\G10001	2.9		
			U3E1\HG10000>U3E1\HG10001	2.6		
			#0L>#2L	1.7		
			D800L>D802L	1.7		
			U3E1\G10000L>U3E1\G10002L	3.1		
			U3E1\HG10000L>U3E1\HG10002L	2.9		
			#0F>#4F	1.8		
			D800F>D804F	1.7		
			U3E1\G10000F>U3E1\G10004F	3.1		
			U3E1\HG10000F>U3E1\HG10004F	2.9		
			>=	大于等于(条件成立时)	#0>=#1	1.6
					D800>=D801	1.5
	U3E1\G10000>=U3E1\G10001	2.9				
	U3E1\HG10000>=U3E1\HG10001	2.6				
#0L>=#2L	1.7					
D800L>=D802L	1.7					
U3E1\G10000L>=U3E1\G10002L	3.1					
U3E1\HG10000L>=U3E1\HG10002L	2.8					
#0F>=#4F	1.8					
D800F>=D804F	1.7					
U3E1\G10000F>=U3E1\G10004F	3.1					
U3E1\HG10000F>=U3E1\HG10004F	2.9					

*1 设置实际输入/实际输出

根据F/G的组合的处理时间 (F/G中记述的程序为NOP)

名称	运动SFC图		单位 [μs]	
F单独			8.0	
G单独			7.5	
F+G			9.0	
GSUB			14.0	
CLR			10.0	
JMP・合并			7.0	
并联分支	2个	分支时		13.0
		合并时		10.5
	5个	分支时		28.0
		合并时		10.5
选择分支	2个			28.5
		5个		

*1 根据启动・清除的程序而有较大不同。

要点 🔍

处理时间变长时，有可能发生运动CPU WDT出错或伺服异常，尤其是对于通过事件任务/NMI任务动作的运动SFC程序，应注意避免处理时间变得过长(避免运算周期溢出)。

高级同步控制专用函数处理时间

各高级同步控制专用函数指令的运算处理时间如下所示。处理时间根据凸轮个数及保存目标而有所不同，因此对表中的值应作为处理时间的大致参考进行参照。

CAMRD指令处理时间

n 数据形式(行程比凸轮)

分辨率	处理时间[ms]		
	不保存(展开区域)	标准ROM	SD存储卡
256	0.02	54	82
4096	0.58	181	406
32768	3.76	2170	411

n 数据形式(坐标)

分辨率	处理时间[ms]		
	不保存(展开区域)	标准ROM	SD存储卡
256	0.04	53	120
4096	1.14	591	139
32768	4.11	2410	621

CAMWR指令处理时间

n 数据形式(行程比凸轮)

分辨率	处理时间[ms]		
	不保存(展开区域)	标准ROM	SD存储卡
256	0.02	42	76
4096	0.04	194	109
32768	3.81	2497	423

n 数据形式(坐标)

分辨率	处理时间[ms]		
	不保存(展开区域)	标准ROM	SD存储卡
256	0.04	55	106
4096	0.96	291	152
32768	4.18	3641	641

CAMMK指令处理时间

n 旋转刀具用凸轮

分辨率	处理时间[ms]		
	不保存(展开区域)	标准ROM	SD存储卡
256	0.26	34	74
1024	0.51	31	70
4096	2.7	32	71
32768	23	53	87

n 简易行程比凸轮

分辨率	区间	处理时间[ms]		
		不保存(展开区域)	标准ROM	SD存储卡
256	1	0.76	34	74
	16	0.88	31	65
	32	0.92	32	65
4096	1	8.9	39	74
	16	5.9	37	71
	32	5.9	39	71
32768	1	69	100	133
	16	47	78	112
	32	47	80	112

n 详细行程比凸轮

• 曲线类型(匀速的情况下)

分辨率	区间	处理时间[ms]		
		不保存(展开区域)	标准ROM	SD存储卡
256	1	1.2	42	73
	32	0.65	33	68
	180	1.1	52	71
	360	1.7	77	91
4096	1	8.9	42	77
	32	6.1	39	73
	180	6.6	58	76
	360	7.1	81	102
32768	1	70	104	134
	32	47	81	113
	180	47	99	114
	360	48	121	139

• 各曲线处理时间(凸轮分辨率256, 区间数1的情况下)

曲线类型	处理时间[ms]		
	不保存(展开区域)	标准ROM	SD存储卡
匀速	1.1	36	61
恒定加速度	0.91	31	53
修正梯形	2.5	32	64
修正正弦	2.5	34	55
修正匀速	3.1	33	60
摆线	1.4	30	61
5次曲线	1	32	60
单停留摆线	3.3	33	60
单停留逆摆线	3.5	33	56
多弦	1.9	32	60
逆多弦	1.9	32	60
单弦	1.5	30	57

CAMPSCL指令处理时间

凸轮分辨率	凸轮位置计算类型	处理时间[ms]
256	凸轮轴进给当前值计算	0.01
	凸轮轴1循环当前值计算	0.003
32768	凸轮轴进给当前值计算	0.011
	凸轮轴1循环当前值计算	0.008

凸轮数据展开时间

将标准ROM中存储的1个凸轮文件展开到展开区域的时间如下所示。根据存储的凸轮文件及其它文件数，查找时间有所不同，因此应将表中的值作为处理时间的大致参考进行参照。

行程比数据形式

凸轮分辨率	凸轮登录数	处理时间 [ms]
256	16	8.5
	1024	20
32768	2	118
	128	125

坐标数据形式

凸轮分辨率	凸轮登录数	处理时间 [ms]
256	16	12
	1024	20
32768	2	612
	128	614

凸轮自动生成

n 旋转刀具用凸轮

凸轮分辨率	凸轮登录数	处理时间 [ms]
1024	256	33
	1024	94

n 简易行程比凸轮

凸轮分辨率	凸轮登录数	区间数	处理时间 [ms]
256	256	1	36
	1024		94

n 详细行程比凸轮

凸轮分辨率	凸轮登录数	区间数	处理时间 [ms]
256	1024	1	103
		180	93
		360	105
4096	1024	1	107
		180	112
		360	115
32768	128	1	103
		180	78
		360	83

运动专用顺控程序指令处理时间

分类	指令(条件)	符号	处理时间[μs]	
			R04CPU/R08CPU/R16CPU/R32CPU/R120CPU/ R08PCPU/R16PCPU/R32PCPU/R120PCPU	
			最小	最大
运动专用顺控程序指令	指定的运动SFC程序的启动请求	D. SFCS	38.0	77.0
		M. SFCS	29.0	68.0
	指定的伺服程序的启动请求	D. SVST	48.0	86.0
		M. SVST	38.0	76.0
	直接定位启动请求(数据点数=14点)	D. SVSTD	62.0	99.0
		M. SVSTD	57.0	91.0
	指定轴的当前值更改请求	D. CHGA	48.0	86.0
		M. CHGA	39.0	74.0
	指定的指令生成轴的当前值更改请求	D. CHGAS	48.0	86.0
		M. CHGAS	38.0	75.0
	指定轴的速度更改请求	D. CHGV	48.0	86.0
		M. CHGV	38.0	75.0
	指定的指令生成轴的速度更改请求	D. CHGVS	48.0	86.0
		M. CHGVS	38.0	75.0
	指定轴的转矩限制值更改请求	D. CHGT	48.0	86.0
		M. CHGT	39.0	76.0
	将位操作写入到其它机号运动CPU的位软元件	D. BITWR	47.0	84.0
		M. BITWR	38.0	74.0
	其它机号运动CPU的事件任务执行请求	D. GINT	40.0	77.0
		M. GINT	31.0	68.0

附2 样本程序

使用了R32MTCPU的样本程序如下所示。

对于本节的样本程序以软件配置方式为“Q兼容配置方式”为例进行说明。

通过运动SFC程序进行的运动控制示例

进行运动控制的运动SFC程序的构成示例

本样本程序示例在以下各功能中有记述。

n 样本程序的功能

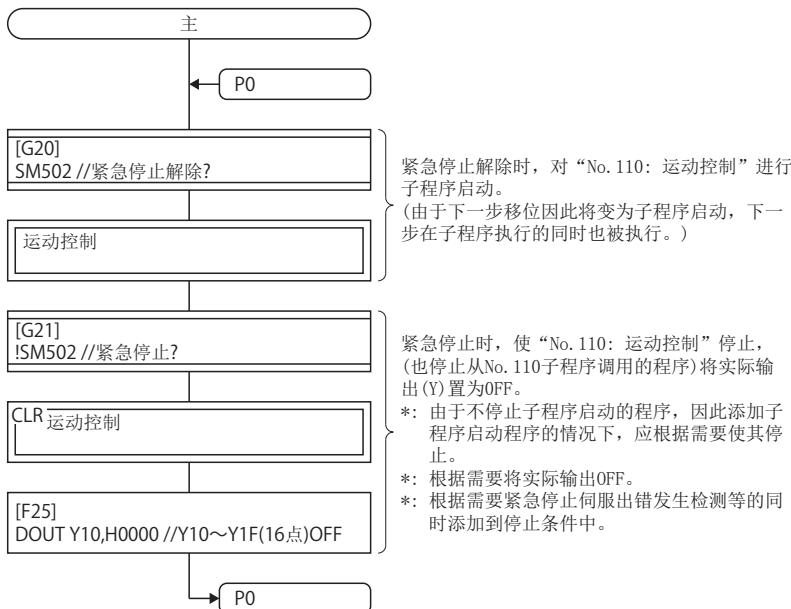
No.	项目	内容
1	紧急停止	X0中分配的紧急停止输入变为ON(紧急停止解除)时,全部轴伺服ON,执行运动控制。紧急停止输入变为OFF时,紧急停止伺服放大器,在停止运动控制的同时将实际输出(Y)置为OFF。
2	运动控制	根据X1、X2的状态通过下述各模式进行运动控制。 <ul style="list-style-type: none">• X2: OFF X1: OFF “JOG模式”• X2: OFF X1: ON “手动脉冲器模式”• X2: ON X1: OFF “原点复位模式”• X2: ON X1: ON “程序运行模式”
3	JOG模式	X3~X6的各信号变为ON时,进行下述JOG运行。 <ul style="list-style-type: none">• X3: 1轴正转JOG• X4: 1轴反转JOG• X5: 2轴正转JOG• X6: 2轴反转JOG
4	手动脉冲器模式	进行下述手动脉冲器运行。 <ul style="list-style-type: none">• 通过手动脉冲器P1进行1轴的手动脉冲器运行。• 通过手动脉冲器P2进行2轴的手动脉冲器运行。
5	原点复位模式	进行下述原点复位。 <ul style="list-style-type: none">• X3为ON时,进行1轴的原点复位。• X4为ON时,进行2轴的原点复位。
6	程序运行模式	进行下述程序运行。 <ul style="list-style-type: none">• 检测出X3的OFF→ON时,1轴的定位后,待机1000[ms],进行2轴的定位。• X4为ON时,进行1、2轴的直线插补定位后,进行就位检查,向反方向以1倍的速度进行1、2轴的直线插补定位,在X4变为OFF之前待机。

运动SFC程序的处理内容

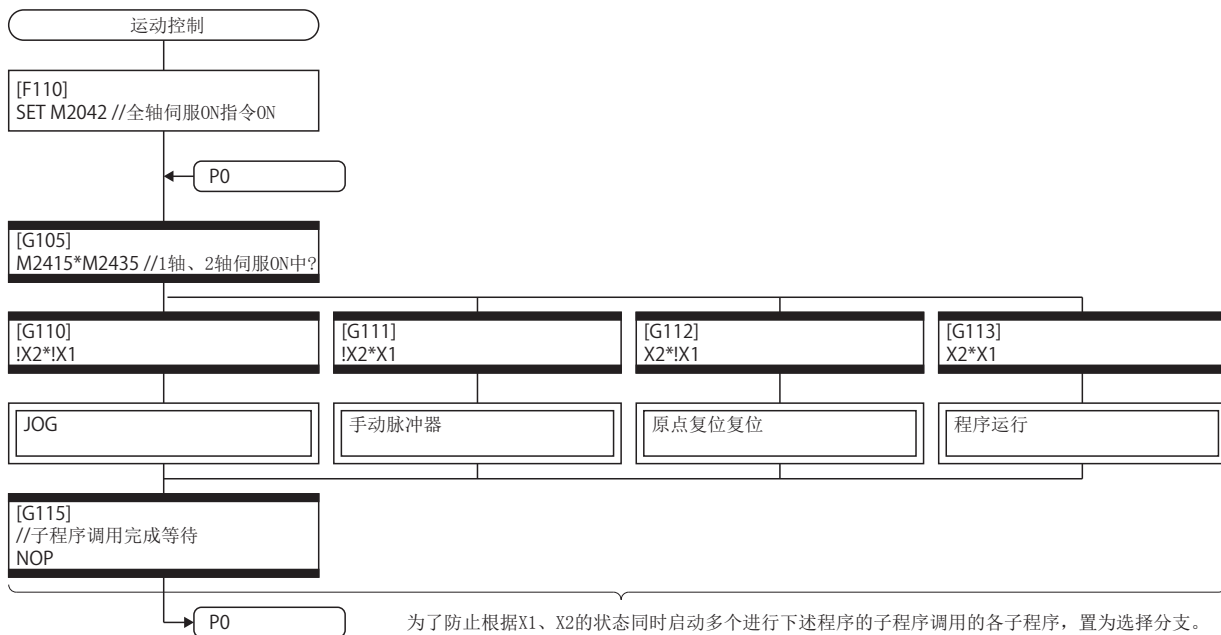
n 运动SFC程序一览

No.	程序名称	任务	自动启动	连续转移数设置	处理内容
20	主	普通	进行	3	<ul style="list-style-type: none"> 运动CPU的RUN时自动启动，常时执行。 紧急停止解除时对“ No. 110: 运动控制”进行子程序启动。 紧急停止时停止“ No. 110: 运动控制”，将实际输出(Y)置为OFF。
110	运动控制	普通	不进行	3	<ul style="list-style-type: none"> 全部轴伺服ON。 根据X1、X2的状态，进行下述程序的子程序调用。 (1) X2: OFF X1: OFF “No. 120: JOG” (2) X2: OFF X1: ON “No. 130: 手动脉冲器” (3) X2: ON X1: OFF “No. 140: 原点复位” (4) X2: ON X1: ON “No. 150: 程序运行”
120	JOG	普通	不进行	3	<ul style="list-style-type: none"> 设置1轴、2轴的JOG运行速度。 X3为ON时，将1轴JOG正转指令置为ON，X4为ON时将反转指令置为ON。 X5为ON时，将2轴JOG正转指令置为ON，X6为ON时将反转指令置为ON。 X2: OFF, X1: OFF (JOG模式)期间重复上述，除此以外的情况下，将1轴、2轴的JOG正转指令、反转指令置为OFF，结束程序。
130	手动脉冲器	普通	不进行	3	<ul style="list-style-type: none"> 设置1轴、2轴的1脉冲输入倍率。 设置为通过P1进行1轴控制，通过P2进行2轴控制后，将P1、P2的手动脉冲器允许标志置为ON。 X2: OFF, X1: ON(手动脉冲器模式)以外的情况下，将P1、P2的手动脉冲器允许标志置为OFF，结束程序。
140	原点复位	普通	不进行	3	<ul style="list-style-type: none"> X3为ON时启动“K140: 1轴的原点复位程序”，X4为ON时启动“K141: 2轴的原点复位程序”。 X2: ON, X1: OFF(原点复位模式)以外时结束程序。
150	程序运行	普通	不进行	3	<ul style="list-style-type: none"> 检测出X3的OFF→ON时，1轴的定位后，待机1000[ms]，进行2轴的定位。 X4为ON时，进行1、2轴的直线插补定位后，进行就位检查，向反方向以1倍的速度进行1、2轴的直线插补定位，在X4变为OFF之前待机。 X2: ON, X1: ON(程序运行模式)以外时，结束程序。

nNo. 20: 主



nNo. 110: 运动控制



为了防止根据X1、X2的状态同时启动多个进行下述程序的子程序调用的各子程序，置为选择分支。此外，通过本程序“No. 20: 主”的清除步进行了停止时，由于各子程序也被停止，因此应将下一步置为WAIT，以确保子程序调用。

X1、X2的状态		子程序调用程序	
X2	X1	No.	程序名称
OFF	OFF	120	JOG
OFF	ON	130	手动脉冲器
ON	OFF	140	原点复位
ON	ON	150	程序运行

nNo. 120: JOG



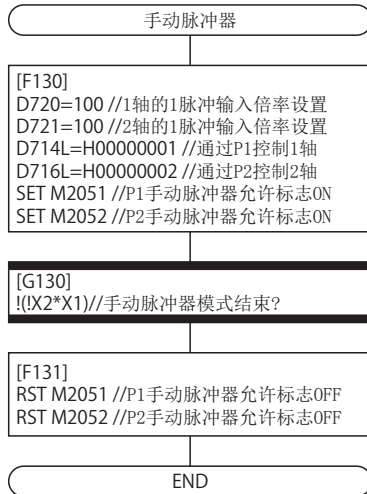
X3~X6的各信号变为了ON/OFF时，将对应的JOG指令软元件置为SET/RST。此时，请勿使同一轴的正转JOG指令与逆转JOG指令同时ON。

信号名称	对应JOG指令软元件
X3	M3202(1轴正转 JOG)
X4	M3203(1轴逆转 JOG)
X5	M3222(2轴正转 JOG)
X6	M3223(2轴逆转 JOG)

*: 通过Y/N转换也可以记述各信号的ON/OFF判别，但是仅通过SET=/RST可记述处理的情况下，通过左边方式记述时，可以缩小步数，缩短处理时间。

转移至其它模式后，为了保证安全，不继续JOG动作，在JOG模式结束时，将1轴、2轴的正转/逆转JOG指令置为OFF。

nNo. 130: 手动脉冲器

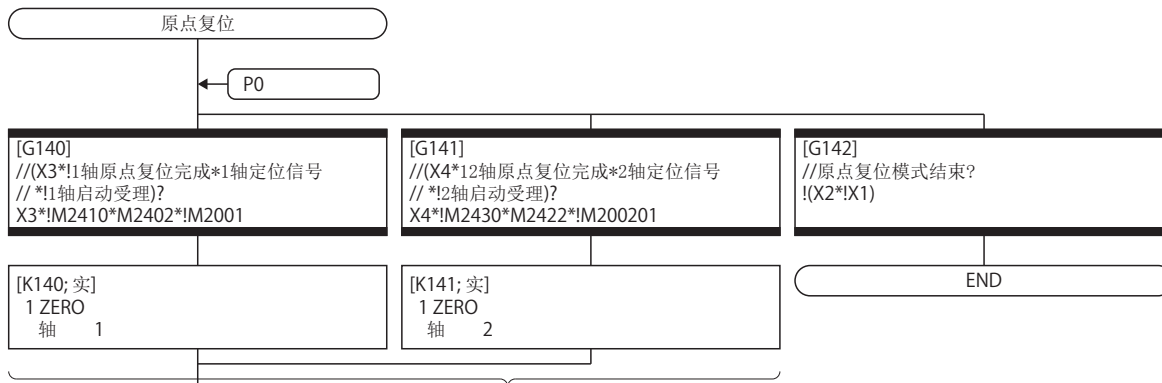


为了通过手动脉冲器P1，在1轴、P2中进行2轴的手动脉冲器运行，进行下述设置。

- 1轴、2轴的1脉冲输入倍率设置。
- 设置手动脉冲器轴设置寄存器以确保通过P1控制1轴，通过P2控制2轴。
- 将P1、P2的手动脉冲器允许标志置为ON。

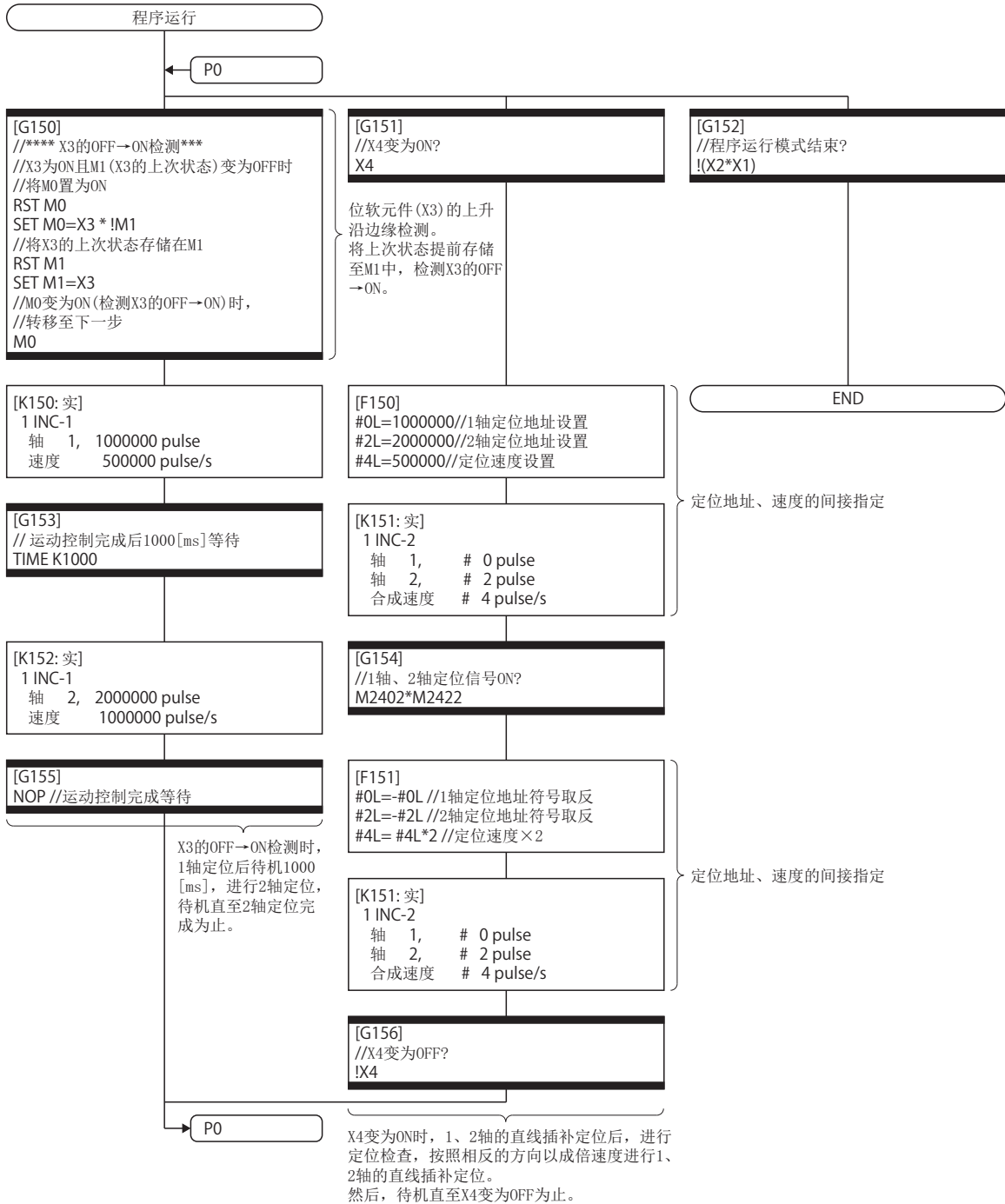
转移至其它模式后，为了保证安全，不继续手动脉冲器动作，在手动脉冲器结束时，将P1、P2的手动脉冲器允许标志置为OFF。

nNo. 140: 原点复位



X3变为ON时，进行1轴的原点复位，X4变为ON时，进行2轴的原点复位。
 此时，确认定位信号处于ON，启动受理处于OFF后，进行原点复位程序的启动。
 *：由于在本程序中可以启动“K140”执行中“K141”，因此变为在运动控制步的下一步无原点复位完成等待WAIT的结构。
 (为了防止K140以及K141的重复启动，需要将各轴的启动受理加入互锁条件中)

nNo. 150: 程序运行

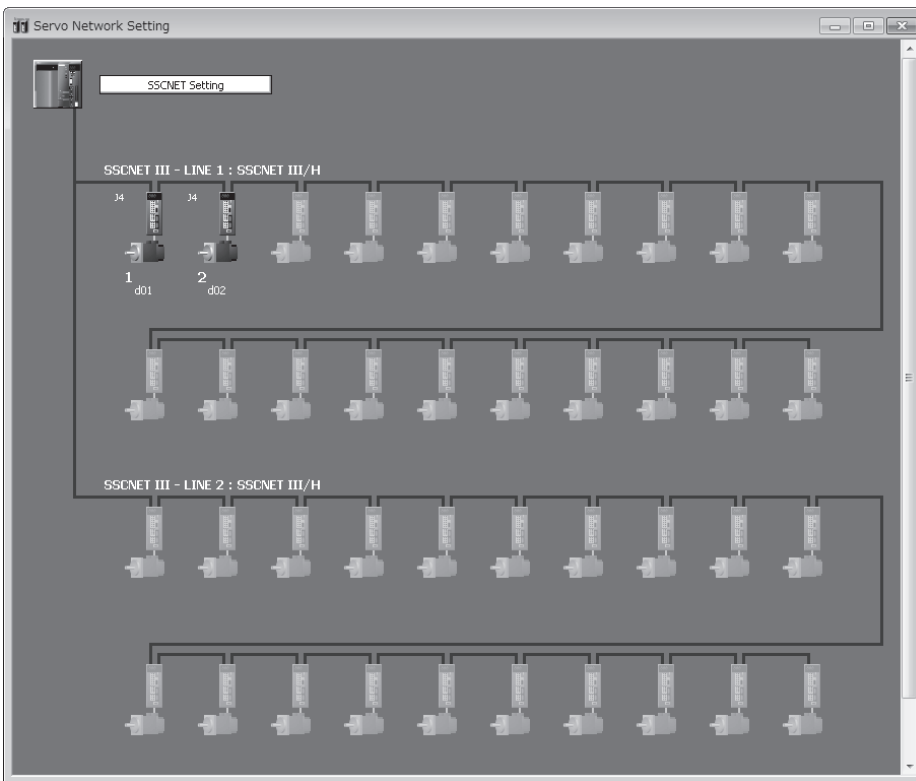


运动CPU的模块构成数据

模块构成如下所示。

	Start I/O No.	Series	Operation Type	Operation Point	Control CPU	Synchronization Setting within the Modules	Setting Item
Main - Power	-	-	Power	-	-	-	-
Main - CPU	3E00	IQ-R	CPU	-	-	-	-
Main - CPU	3E10	IQ-R	CPU (Host Station)	-	-	-	-
Main - I/O 1	0000	IQ-R	Input	16 Point	CPU No.2	-	Detailed
Main - I/O 2	0010	IQ-R	Output	16 Point	CPU No.2	-	Detailed
Main - I/O 3	0020	IQ-R	Intelligent	16 Point	CPU No.2	-	Detailed
Main - I/O 4	-	-	-	-	-	-	-
Main - I/O 5	-	-	-	-	-	-	-
Main - I/O 6	-	-	-	-	-	-	-
Main - I/O 7	-	-	-	-	-	-	-

Points Occupied by Empty Slot | 16 | Point



通过运动SFC程序进行子程序重启时的继续执行示例

动作说明

将运动控制执行中的子程序程序通过清除步停止后，进行重启时，从中途停止的运动控制步开始继续执行的程序示例如下所示。在本程序中，通过紧急停止解除进行伺服ON，X4为ON时进行2轴直线插补的定位控制。定位动作完成后，确认X4变为OFF，完成1个循环动作。定位动作中紧急停止的情况下，中断路定位动作，停止伺服电机。此后紧急停止被解除时，从中断的定位动作开始进行重启。在本程序示例中，通过下述处理进行子程序重启时的继续执行。

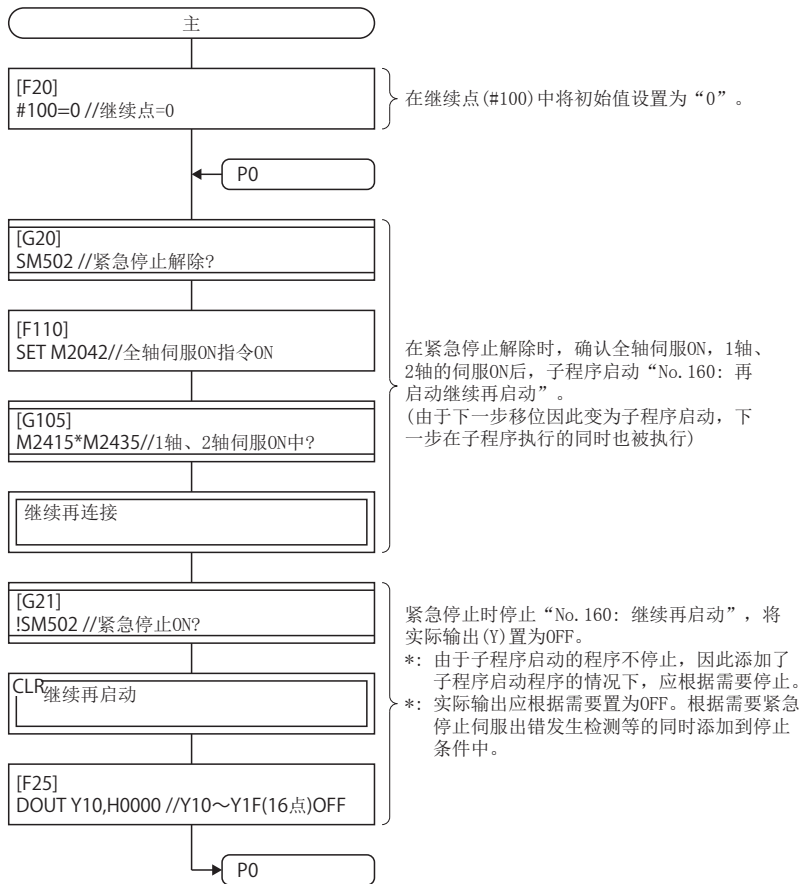
- 子程序的运动控制执行中，用户软元件中记忆哪个运动控制步的定位已完成。
- 子程序重启时以上述中记忆的信息为基础，从中途停止的运动控制步开始进行重启。
- 为了支持运动控制步定位途中停止后的重启，进行绝对定位。
- 定位中是否中途停止的判定使用“[St. 1060]定位完成(M2401+20n)”。

运动SFC程序的处理内容

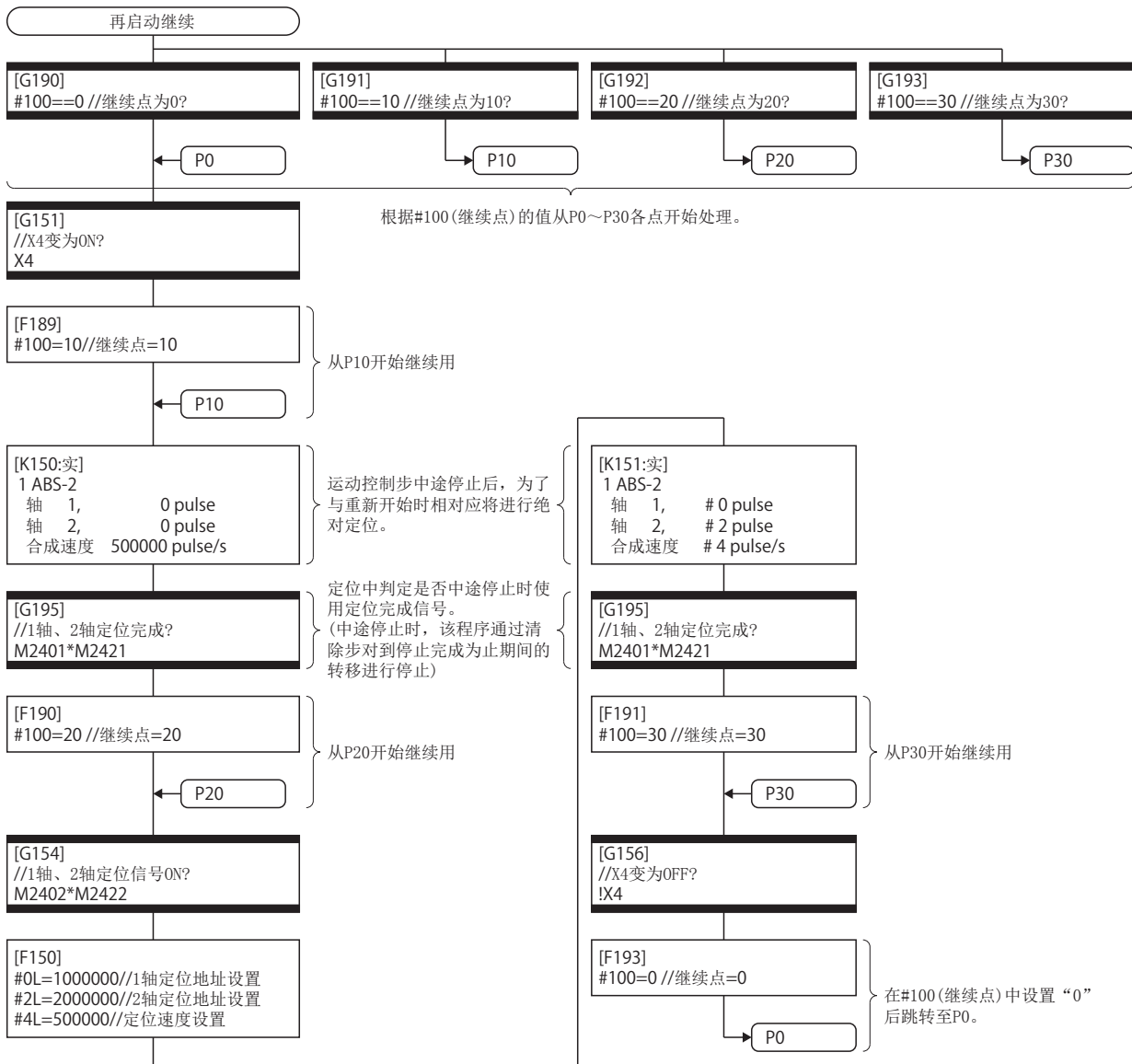
n 运动SFC程序一览

No.	程序名称	任务	自动启动	连续转移数设置	处理内容										
20	主	普通	进行	3	<ul style="list-style-type: none"> • 运动CPU的RUN时自动启动，常时执行。 • 在继续点(#100: 用户软元件)中作为初始值设置“0”。 • 紧急停止解除时全部轴伺服ON，进行1轴与2轴的伺服ON确认后，对“No. 160: 重启继续”进行子程序启动。 • 紧急停止时停止“No. 160: 重启继续”，将实际输出(Y)置为OFF。 										
160	重启继续	普通	不进行	3	<ul style="list-style-type: none"> • 根据继续点(#100)的值跳转至下述(1)~(9)。 <table border="1" data-bbox="766 974 1145 1160"> <thead> <tr> <th>#100</th> <th>跳转目标</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>下述(1)</td> </tr> <tr> <td>10</td> <td>下述(3)</td> </tr> <tr> <td>20</td> <td>下述(5)</td> </tr> <tr> <td>30</td> <td>下述(8)</td> </tr> </tbody> </table> <ul style="list-style-type: none"> • 进行下述运动控制。 <ol style="list-style-type: none"> (1) X4变为ON之前待机。 (2) 在继续点(#100)中设置“10”。 (3) 将1轴、2轴通过直线插补(绝对2轴定位)定位到(0, 0)。 (4) 确认1轴、2轴的定位完成信号ON后，在继续点(#100)中设置“20”。 (5) 确认1轴、2轴的就位ON。 (6) 将1轴、2轴通过直线插补(绝对2轴定位)定位到(1000000, 2000000)。 (7) 确认1轴、2轴的定位完成信号ON后，在继续点(#100)中设置“30”。 (8) X4变为OFF之前待机。 (9) 在继续点(#100)中设置“0”。 	#100	跳转目标	0	下述(1)	10	下述(3)	20	下述(5)	30	下述(8)
#100	跳转目标														
0	下述(1)														
10	下述(3)														
20	下述(5)														
30	下述(8)														

nNo. 20: 主



nNo. 160: 重启运行



通过运动SFC程序暂停后的继续执行示例

动作说明

通过来自于输入模块的暂时停止用外部输入信号ON暂停运动SFC程序的执行后，通过暂时停止用外部信号OFF继续执行的程序示例如下所示。在本程序中，通过紧急停止解除进行伺服ON，在X4变为ON时进行2轴直线插补的定位控制。定位动作完成后，确认X4变为OFF后完成1个循环动作。定位动作中X5变为ON时，通过停止指令停止定位后，通过X5变为OFF从中断的定位动作开始进行重启。在WAIT转换中，X5为ON中不进行至下一步的转移。此外，定位动作中紧急停止的情况下，中断定位动作，停止伺服电机。此后紧急停止被解除时，从中断的定位动作开始进行重启。在本程序示例中，通过下述处理进行暂停及暂停的继续执行。

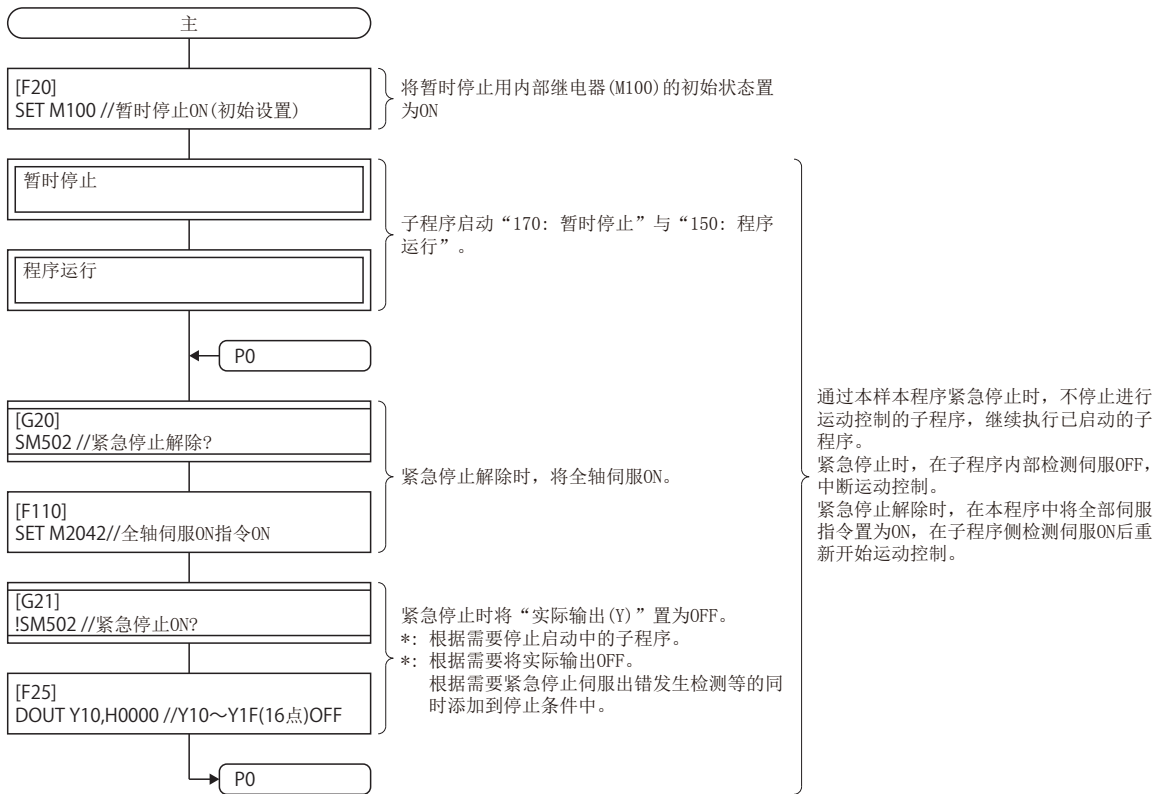
- X5为ON中将“[Rq. 1140]停止指令(M3200+20n)”及暂时停止用内部继电器(M100)置为ON。
- X5为OFF中将“[Rq. 1140]停止指令(M3200+20n)”及暂时停止用内部继电器(M100)置为OFF。
- 为了支持运动控制步定位途中停止后的重启，进行绝对定位。
- 定位中是否中途停止的判定使用“[St. 1060]定位完成(M2401+20n)”。
- 定位中途中停止的情况下，待暂时停止用内部继电器(M100)变为OFF后，重启中断的运动控制步。
- 在需要暂停的WAIT转换转移条件中将“暂时停止用内部继电器(M100)的OFF”置入到AND条件中。

运动SFC程序的处理内容

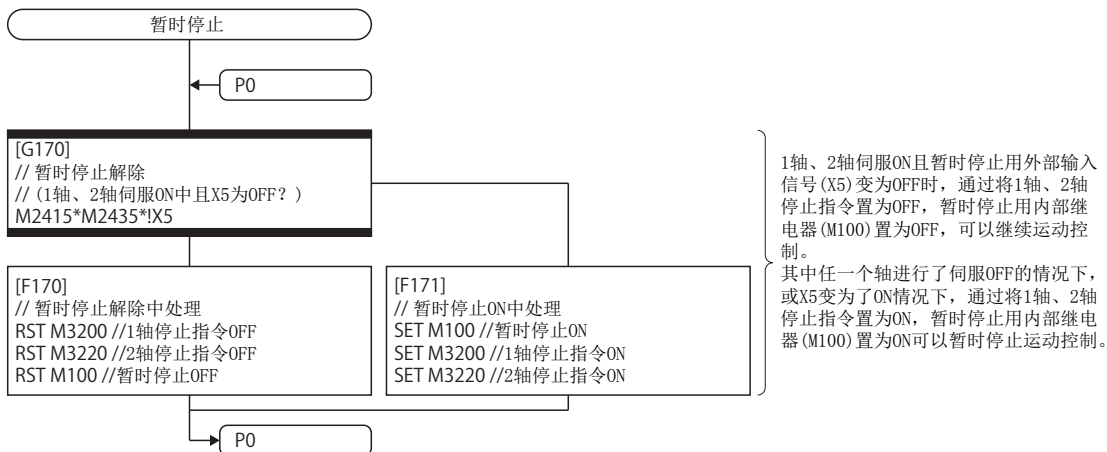
n 运动SFC程序一览

No.	程序名称	任务	自动启动	连续转移数设置	处理内容
20	主	普通	进行	3	<ul style="list-style-type: none"> • 运动CPU的RUN时自动启动，常时执行。 • 将暂时停止用内部继电器(M100)的初始化状态置为ON。 • 将“No. 170: 暂时停止”进行子程序启动。 • 将“No. 150: 程序运行”进行子程序启动。 • 紧急停止解除时进行全部轴伺服ON。 • 紧急停止时将实际输出(Y)置为OFF。
170	暂时停止	普通	不进行	3	<ol style="list-style-type: none"> (1) 1轴、2轴为伺服ON中来自于输入模块的暂时停止输入信号(X5)为OFF的情况下，执行下述(2)的处理，除此以外的情况下执行下述(3)。 (2) 将1轴、2轴停止指令置为OFF后，将暂时停止用内部继电器(M100)置为OFF。 (3) 将1轴、2轴停止指令置为ON后，将暂时停止用内部继电器(M100)置为ON。
150	程序运行	普通	不进行	3	<ul style="list-style-type: none"> • 进行下述运动控制。 (1) X4变为ON之前待机。 (2) 对1轴、2轴通过直线插补(绝对2轴定位)定位至(0, 0)。 (3) 确认1轴、2轴的定位完成信号ON。 (4) 确认1轴、2轴的就位ON。 (5) 对1轴、2轴通过直线插补(绝对2轴定位)定位至(1000000, 2000000)。 (6) 确认1轴、2轴的定位完成信号ON。 (7) 在X4变为OFF之前待机。 • 上述(3)及(6)中定位完成信号变为OFF的情况下(定位途中停止的情况下)，等待暂时停止用内部继电器(M100)的OFF，再次执行之前的运动控制步(2)或(5)。 • 在上述(1)及(7)中，在暂时停止用内部继电器(M100)变为ON之前不转移至下一步。

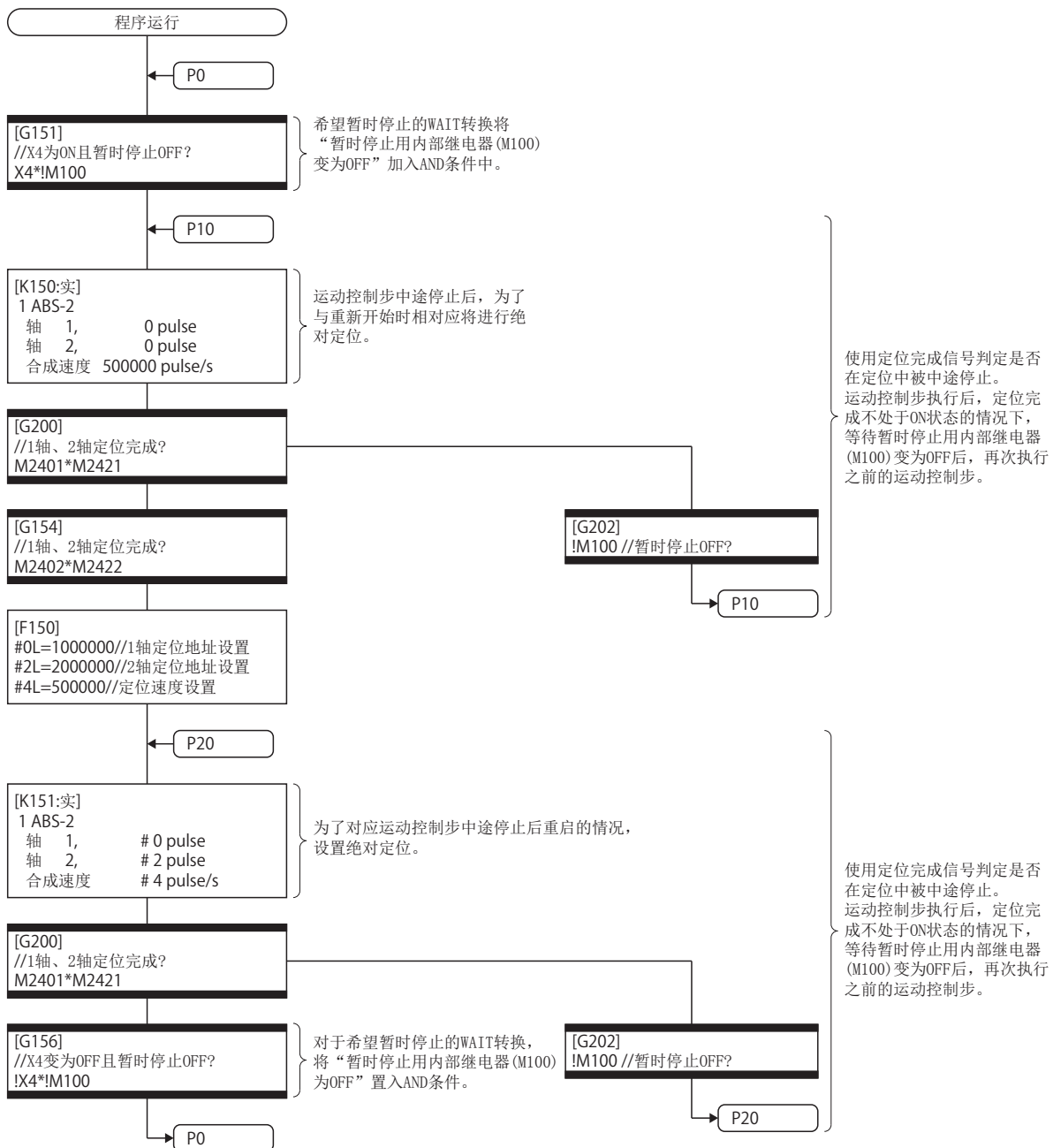
nNo. 20: 主



nNo. 170: 暂时停止



nNo. 150: 程序运行



附3 机器出错详细代码

详细信息1

n 机器控制设置数据报警(报警(出错代码: 0EE0H)、机器控制设置数据不正确(轻度出错(出错代码: 1FE0H))

检测出机器控制设置数据报警(报警(出错代码: 0EE0H)、机器控制设置数据不正确(轻度出错(出错代码: 1FE0H))时的详细代码如下所示。

详细代码	出错名称
0001H	指令设置不正确
0003H	参数块No.
0004H	差补控制单位
0005H	超出速度限制值设置范围
0006H	加速时间
0007H	减速时间
0008H	急停止减速时间
0009H	启动时转矩限制值
000AH	STOP输入时减速处理
000BH	圆弧插补误差允许范围
000DH	S字比率
0031H	控制方式
0032H	坐标系指定
0033H	指令速度
0034H	点块No.
0035H	辅助点/中心点点块No.
0038H	M代码
0039H	停顿时间
003AH	运行中转矩限制值
003BH	近傍通过方式
003CH	近傍量
0061H	内部处理异常
0062H	坐标系指定·定位类别组合不正确
0063H	急停止减速时间>减速时间
0064H	辅助点·终点设置出错(辅助点圆弧)
0065H	中心点·终点设置出错(中心点圆弧)
0066H	辅助点·中心点溢出
0067H	超程
00E0H	机器JOG运行方式
00E1H	机器JOG速度
00E2H	机器JOG坐标系指定超出设置范围
00F0H	伺服电机最大旋转速度超出设置范围
00F1H	振动抑制指令平滑滤波器未设置报警

n 机器控制机器库出错(轻度出错(出错代码: 1FE1H))、机器配置出错(重度出错(出错代码: 30FAH))

检测出机器控制机器库出错(轻度出错(出错代码: 1FE1H))、机器配置出错(重度出错(出错代码: 30FAH))时的详细代码如下所示。

详细代码	出错名称
0001H	机器参数不正确
0101H	机器库不存在
0102H	机器类型设置不正确
0103H	参数块指定不正确
0104H	机器JOG速度限制值不正确
0105H	动作范围类型不正确
0106H	关节轴配置不正确
0107H	臂长设置不正确
0108H	正交行程限位设置不正确
0109H	选项设置A不正确
010AH	选项设置B不正确
010BH	单位设置不正确
010CH	行程限位上限值/下限值不正确
0201H	基本转换值超出设置范围
0301H	工具转换值超出设置范围
0501H	特点
0502H	超出动作范围
0601H	姿势标志不正确
0602H	坐标系指定不正确
0603H	终点溢出

修订记录

*本手册号在封底的左下角。

修改日期	*手册编号	修改内容
2014年10月	IB(NA)-0300275CHN-A	第一版
2015年09月	IB(NA)-0300275CHN-B	第二版 部分改版
2015年12月	IB(NA)-0300275CHN-C	第三版 部分改版

日文原稿手册：IB-0300238-D

本手册不授予工业产权或任何其它类型的权利，也不授予任何专利许可。三菱电机对于使用了本手册中的内容而引起的涉及工业产权的任何问题不承担责任。

©2014 MITSUBISHI ELECTRIC CORPORATION

质保

使用之前请确认以下产品质保的详细说明。

1. 免费质保期限和免费质保范围

在免费质保期内使用本产品时如果出现任何属于三菱电机责任的故障或缺陷（以下称“故障”），则经销商或三菱电机服务公司将负责免费维修。

但是如果需要在国内现场或海外维修时，则要收取派遣工程师的费用。对于涉及到更换故障模块后的任何再试运转、维护或现场测试，三菱电机将不负任何责任。

[免费质保期限]

免费质保期限为自购买日或交货的一年内。

注意产品从三菱电机生产并出货之后，最长分销时间为 6 个月，生产后最长的免费质保期为 18 个月。维修零部件的免费质保期不得超过修理前的免费质保期。

[免费质保范围]

(1) 范围局限于按照使用手册、用户手册及产品上的警示标签规定的使用状态、使用方法和使用环境正常使用的情况下。

(2) 以下情况下，即使在免费质保期内，也要收取维修费用。

1. 因不当存储或搬运、用户过失或疏忽而引起的故障。因用户的硬件或软件设计而导致的故障。
2. 因用户未经批准对产品进行改造而导致的故障等。
3. 对于装有三菱电机产品的用户设备，如果根据现有的法定安全措施或工业标准要求配备必需的功能或结构后本可以避免的故障。
4. 如果正确维护或更换了使用手册中指定的耗材（电池、背光灯、保险丝等）后本可以避免的故障。
5. 因火灾或异常电压等外部因素以及因地震、雷电、大风和水灾等不可抗力而导致的故障。
6. 根据从三菱电机出货时的科技标准还无法预知的原因而导致的故障。
7. 任何非三菱电机或用户责任而导致的故障。

2. 产品停产后的有偿维修期限

(1) 三菱电机在本产品停产后的 7 年内受理该产品的有偿维修。

停产的消息将以三菱电机技术公告等方式予以通告。

(2) 产品停产，将不再提供产品（包括维修零件）。

3. 海外服务

在海外，维修由三菱电机在当地的海外 FA 中心受理。注意各个 FA 中心的维修条件可能会不同。

4. 意外损失和间接损失不在质保责任范围内

无论是否在免费质保期内，对于任何非三菱电机责任的原因而导致的损失、机会损失、因三菱电机产品故障而引起的用户利润损失、无论能否预测的特殊损失和间接损失、事故赔偿、除三菱电机以外产品的损失赔偿、用户更换设备、现场机械设备的再调试、运行测试及其它作业等，三菱电机将不承担责任。

5. 产品规格的改变

目录、手册或技术文档中的规格如有改变，恕不另行通知。

商标

Microsoft、Windows、Windows Vista、Windows NT、Windows XP、Windows Server、Visio、Excel、PowerPoint、Visual Basic、Visual C++、Access是美国Microsoft Corporation在美国、日本及其它国家的注册商标或商标。

Intel、Pentium、Celeron是美国及其它国家Intel Corporation的注册商标或商标。

以太网、Ethernet是富士施乐公司的注册商标。

SD标志、SDHC标志是SD-3C、LLC的注册商标或商标。

本手册中使用的其它公司名和产品名是相应公司的商标或注册商标。



IB (NA) -0300275CHN-C (1512) MEACH

MODEL: RMT-P-PRG-C

 **三菱电机自动化(中国)有限公司**

地址：上海市虹桥路1386号三菱电机自动化中心

邮编：200336

电话：021-23223030 传真：021-23223000

网址：<http://cn.MitsubishiElectric.com/fa/zh/>

技术支持热线 **400-821-3030**



扫描二维码,关注官方微博



扫描二维码,关注官方微信

内容如有更改 恕不另行通知