QCPU

MITSUBISHI

用户手册

（功能解说/
程序基础篇）

三菱可编程控制器

MELSEC-Q

Q00JCPU	*Q12PHCPU*
Q00CPU	*Q25PHCPU*
Q01CPU	*Q12PRHCPU*
Q02CPU	*Q25PRHCPU*
Q02HCPU	*Q02UCPU*
Q06HCPU	*Q03UDCPU*
Q12HCPU	*Q04UDHCPU*
Q25HCPU	*Q06UDHCPU*

● 安全注意事项 ●

（在使用之前请务必阅读本说明）

在使用本产品之际，要仔细阅读本手册以及本手册介绍的相关手册，同时在充分注意安全的前提下，进行正确的操作。
在此安全注意事项中，将其划分为 “危险”、“注意”二个等级进行区分。

⚠ 危险	表示不正确的操作可能引起危险，导致伤亡或严重人身伤害。
⚠ 警告	表示不正确的操作可能引起危险，导致中等或轻微人身伤害或者财产损失。

此外，即使是⚠“注意”中的事项，在某些情况下有时也可能会导致严重的后果。
总之，都是很重要的内容，请务必遵守。

为保证本手册能在必要时能够得到查阅，请予以妥善保管，同时，请务必将本手册提供给最终的用户。

[设计上的注意事项]

⚠危险

● 外部电源出现异常和可编程控制器本体出现故障时，为了保证整个系统的安全，须在可编程控制器外部设置安全电路。否则误输出、误动作会有引起事故的危险。

(1) 请在可编程控制器的外部设置紧急停止电路、保护电路、正转 / 反转等相反操作的连锁电路以及定位上限 / 下限等防止机械破损的连锁电路等电路系统。

(2) 可编程控制器在检测出以下的异常状态时将会停止运算，并且：
在下表中 (a) 的情况下，所有输出均为 OFF。
在下表中 (b) 的情况下，根据参数设置所有输出均为保持或 OFF。
但是，对于 AnS 系列的模块来说，无论在 (a) 或 (b) 的任何一种情况下都会输出 OFF。

	Q系列的模块	AnS系列的模块
(a) 电源模块的过电流保护装置或者过电压保护装置动作时。	输出OFF	输出OFF
(b) 可编程控制器CPU的自我检测功能检测出看门狗时钟溢出等异常时。	根据参数设置所有输出保持或OFF。	输出OFF

另外，可编程控制器 CPU 在无法检测出的 I/O 控制部分等出现异常时，有可能发生所有的输出均为 ON 的情形。这时为了保证机械动作的安全，请在可编程控制器的外部设置失效安全电路或机构。关于失效安全电路的实例，请参照 QCPU 用户手册（硬件设计 / 维护点检篇）的安装和设置部分。

(3) 有时输出模块的继电器和晶体管等的故障，会使其输出保持 ON 状态或者保持 OFF 的状态。对于可能引起重大事故的输出信号，请在外部设置监视电路。

[设计方面的注意事项]

⚠危险

- 在输出模块方面，额定以上的负载电流或者负载短路等引起的过电流长时间持续通过的情况下，有发生冒烟、起火的危险，因此请在外部设置保险丝等安全电路。

- 请将电路设计成可编程控制器的主机的电源开启后，再投入外部供给电源。
  若先开启外部电源可能会有误输出、误动作，从而有引起事故的危险。

- 对于数据通讯发生异常时各站的动作状态，请参照各数据通讯的手册。
  误输出、误动作有引起事故的危险。

- 将外围设备与 CPU 模块连接，或者是将智能型功能模块 / 特殊功能模块与个人电脑连接，对运行中的可编程控制器进行控制（更改数据）的时候，为使系统整体保持安全运行，请在顺控程序上设置连锁电路。而且，对运行中的可编程控制器进行其它控制（更改程序，更改运行状态（状态控制））的时候，请熟读手册并充分确认安全后再进行操作。
  特别是利用外部设备对距离较远的可编程控制器进行上述控制时，会出现由于数据通讯异常而无法立即对可编程控制器的故障采取应对措施的情况。
  在顺控程序上设置连锁电路的同时，还应制定外部设备与可编程控制器之间发生数据通讯异常时的系统处置方法等。

⚠注意

- 控制线和通讯电缆请勿与主电路及动力电缆捆扎在一起，也勿使其太接近。
  大约隔开 100mm 以上。因为噪声会引起误动作。

- 用输出模块控制电灯负载、加热器、电磁阀等的时候，输出从 OFF 转至 ON 时有大量电流（为通常的十倍水平）通过的情形，因此应采取选用额定电流有富余量的输出模块的对策。

[安装上的注意事项]

⚠注意

● 可编程控制器应在 QCPU 用户手册（硬件设计 / 维护点检篇）中规定的一般技术规格的环境下使用。
如果在一般技术规格范围以外的环境下使用，会引起触电、火灾、误动作、产品损坏或者劣化现象的发生。

● 按住模块下部的安装卡子的同时，将模块固定用突起物可靠地插入基板的固定孔，并以固定孔为支点进行安装。
如果模块没有得到的正确安装，则会引起误动作、故障及脱落。
在振动频繁的环境下使用时，请用螺钉将模块拧紧。
紧固螺钉请在规定的扭矩范围内进行。
螺钉如果过松会引起脱落、短路、误动作。
螺钉如果过紧，会导致螺钉和模块的损坏从而引起脱落、短路以及误动作。

● 扩展电缆请可靠安装在基板用来扩展电缆的接头上。
安装之后请进行检查。
接触不良会引起误输入以及误输出。

● 存储卡请可靠的安装在存储卡插槽上。
安装之后请进行检查。
接触不良会引起误动作。

● 模块的拆装必须要在将系统中使用的外部电源全部切断之后进行。
如果不全部切断，就有损伤产品的危险。
对于可在线更换的 CPU 模块的系统以及 MELSECNET/H 远程 I/O 站，可以在在线中（通电中）进行模块的更换。
但是，在线中（通电中）进行模块更换是有限制的，每个模块都有规定的更换步骤。
详细内容请参照 QCPU 用户手册（硬件设计 / 维护点检篇）以及与在线模块更换相对应的手册中模块更换的事项。

● 请不要直接接触模块的导电部分。
否则会引起模块的误动作与故障的发生。

● 使用运动控制 CPU 模块、运动控制模块时，在投入电源之前必须确认模块的组合是否正确。
在错误的组合下使用时可能会引起产品损坏。
有关详细内容请参阅运动控制 CPU 模块的用户手册。

[连线时的注意事项]

⚠危险

● 连线作业等必须要在将系统中使用的外部电源全部切断之后进行。
不全部切断电源会有触电或者损伤产品的危险。

● 连线作业之后进行通电、运行时，必须在产品上安装附属的端子盖。
如果端子盖没有盖上的话，有触电的危险。

⚠注意

● FG 端子以及 LG 端子必须可靠接地，其接地等级为可编程控制器专用的 D 种接地（第三种接地）以上。
否则会有触电、误动作的危险。

● 应使用合适的压装端子，并按规定的扭矩拧紧。
如果使用 Y 端子，若端子螺栓松动将可能导致脱落、故障。

● 模块的连线必须在确认产品的额定电压以及端子排列之后地进行操作。
与额定电压相异的电源连接或者连线错误会导致火灾以及故障的发生。

● 用于外部连接的连接器必须用生产厂家指定的工具进行压装、压接或者正确地焊接。
如果接触不良则会引起短路、火灾以及误动作。

● 端子螺栓的紧固应在规定的扭矩范围内进行。
端子螺栓如果拧得过松可能会引起短路、火灾以及误动作。
端子螺栓如果拧得过紧，则可能由于螺栓和模块的损坏而引起脱落、短路以及误动作。

● 要注意模块内不要弄进切屑和连线碎块等异物。
否则会引起火灾、故障、误动作。

● 为了防止在连线时布线配件、碎块等异物进入模块内，在模块上部贴着防止杂物混入的贴纸。
在连线作业中不要揭下此贴纸。
在系统运行时，为了更好地散热，请务必揭下此贴纸。

● 三菱公司的可编程控制器应安装在控制盘内使用。
安装在控制盘内的可编程控制器的电源模块应通过中继端子排与主电源连线。
此外，在对电源模块进行更换及连线作业时，应由在触电保护方面受到过良好培训的维护人员进行操作。
关于连线方法请参阅 QSCPU 用户手册（硬件设计 / 维护点检篇）。

[启动、维护时的注意事项]

⚠危险

● 通电中不要接触端子。
否则会有触电的危险。

● 请正确地连接电池。
不要对电池进行充电、分解、加热、扔进火中、短路以及焊接等操作。
如果对电池处理不当，由于电池发热、破裂、起火的原因，会引发火灾以及造成人员损伤。

● 清理或对端子螺钉、模块固定螺钉的紧固必须在将系统中使用的外部电源全部切断之后进行。
不全部切断电源会有触电的危险。
端子螺钉过松则会引起短路以及误动作。
如果螺钉过紧，则可能由于螺钉和模块的破损从而引起脱落、短路以及误动作。

⚠注意

● 将外围设备连接到运行中的 CPU 模块上进行在线操作（特别是更改程序、强制输出、更改运行状态）时，必须要在熟读本手册，充分确认安全后进行。
如果操作不当会引起机器的破损以及事故的发生。

● 不要对各模块进行分解和改造。
否则会引起故障、误动作、人员受伤以及火灾的发生。

● 手机和 PHS 等无线通信设备要离可编程控制器 25cm 以上使用。
否则会引起误动作。

● 模块的装卸必须要在将系统使用的外部电源全部切断之后进行。不全部切断会引起模块发生故障以及产生误动作。
对于使用可在线更换的 CPU 模块的系统以及 MELSECNET/H 远程 I/O 站，可以在在线状态（通电状态）下进行模块的更换。
但是，可在在线状态（通电状态）下更换的模块是有限制的，各个模块都有规定的更换步骤。
详细内容请参阅 QCPU 用户手册（硬件设计 / 维护点检篇）以及与在线模块更换相对应的模块手册中的在线模块更换的事项。

● 模块、基板及端子排在投入使用后，其拆装次数应不超过 50 次。（根据 IEC61131-2 标准）
如果其拆装次数超过了 50 次，有可能导致误动作。

● 应防止安装在模块上的电池掉落以及受到撞击。
掉落以及受到撞击会使电池发生破损以及电池内部发生电池漏液。
掉落、受到撞击的电池请不要使用并应将其废弃。

● 在接触模块之前，必须要先触摸已接地的金属，释放掉人体上所带的静电。
如果不放掉静电则会引起模块发生故障以及产生误动作。

[废弃时的注意事项]

⚠注意
● 产品废弃的时候，请作为工业废品来处理。

[运输时的注意事项]

⚠注意
● 运输含有锂的电池时，有必要按照运输的规定进行处理。 （规定对象种类的详细内容请参照附录 6）

修订记录

使用说明书编号标记在本说明书封底的左下方

印刷日期	使用说明书编号	修订内容
2005 月 02 月	SH(NA)-080503CHN-A	初版
2007 月 02 月	SH(NA)-080503CHN-B	部分修订
2007 月 10 月	SH(NA)-080503CHN-C	新增了通用型 QCPU 机型。 进行了冗余 CPU 序列号 09012 相应的修订。 进行了高性能模式 QCPU 序列号 09012 相应的修订。 新增机型 Q65WRB、Q02UCPU、Q03UDCPU、Q04UDHCPU、Q06UDHCPU 新增 1.1.5 项、5.2.3 项、5.2.11 项、6.25.1 项、6.25.2 项、6.27 节、6.29 节、6.30 节、9.5.1 项、9.5.2 项、9.6.1 项、9.6.2 项、附录 5 更改 5.2.3 项→ 5.2.4 项、5.2.4 项→ 5.2.5 项、5.2.5 项→ 5.2.6 项、5.2.6 项→ 5.2.7 项、5.2.7 项→ 5.2.8 项、5.2.8 项→ 5.2.9 项、5.2.9 项→ 5.2.10 项、6.27 节→ 6.28 节、9.6.1 项→9.6.3 项、9.6.2 项→9.6.4 项、附录 5 →附录 6、附录 6 →附录 7 部分修订 安全注意事项、关于手册、手册的阅读方法、本手册所使用的总称与略称、第 1 章、1.1.2 项、1.1.3 项、1.1.4 项、1.2 节、1.3 节、1.4 节、第 2 章、第 3 章、3.1.1 项、3.1.2 项、3.1.3 项、3.3 节、3.3.1 项、3.3.3 项、3.3.4 项、3.3.5 项、3.3.6 项、3.4.1 项、3.4.2 项、3.4.4 项、3.5 节、3.6 节、3.7 节、3.8 节、3.8.1 项、3.8.2 项、3.9.1 项、3.9.2 项、3.9.4 项、4.1 节、4.2 节、4.3 节、4.4 节、4.6.1 项、4.6.2 项、4.7.2 项、5.2.1 项、5.2.2 项、5.2.4 项、5.2.5 项、5.2.6 项、5.2.7 项、5.2.8 项、5.2.9 项、5.2.10 项、5.4.1 项、5.4.2 项、5.4.3 项、5.4.4 项、6.1 节、6.2 节、6.3 节、6.4 节、6.5 节、6.6 节、6.6.1 项、6.6.2 项、6.6.3 项、6.6.4 项、6.7 节、6.10 节、6.11 节、6.11.1 项、6.11.2 项、6.11.3 项、6.12 节、6.12.1 项、6.12.2 项、6.12.3 项、6.13.1 项、6.13.3 项、6.14 节、6.15 节、6.15.1 项、6.15.2 项、6.17 节、6.17.1 项、6.18.2 项、6.19 节、6.19.1 项、6.19.2 项、6.20 节、6.21 节、6.21.1 项、6.21.2 项、6.22 节、6.22.1 项、6.22.2 项、6.22.3 项、6.22.4 项、6.24 节、6.25 节、6.26 节、6.28 节、第 7 章、7.1 节、7.1.1 项、7.1.2 项、7.1.4 项、7.1.5 项、7.2 节、第 8 章、8.1.1 项、8.1.2 项、8.2 节、8.3 节、9.1 节、9.2 节、9.2.1 项、9.2.3 项、9.2.4 项、9.2.5 项、9.2.7 项、9.2.8 项、9.2.9 项、9.2.10 项、9.2.11 项、9.2.13 项、9.2.14 项、9.3.1 项、9.3.2 项、9.3.3 项、9.4 节、9.5 节、9.6 节、9.6.3 项、9.6.4 项、9.7 节、9.7.1 项、9.7.2 项、9.7.3 项、9.7.6 项、9.9 节、9.10 节、9.10.1 项、9.11.1 项、9.11.2 项、9.11.3 项、9.12.2 项、9.12.3 项、9.13.1 项、10.1.1 项、10.1.2 项、10.1.3 项、10.1.4 项、10.2 节、11.2 节、11.2.1 项、11.2.2 项、11.2.3 项、11.2.4 项、11.2.5 项、附录 1、附录 2、附录 3、附录 4.2、附录 4.3、附录 4.4、附录 6、附录 7

英文手册原稿： SH-080484ENG-I

序言

此次，非常感谢您购买三菱通用可编程控制器 MELSEC-Q 系列。
在使用前请您熟读此书，并在充分理解 Q 系列可编程控制器的功能及性能的基础上正确地使用。

目录

（相关手册）..... QCPU 用户手册（硬件设计 / 维护点检篇）

目录

目录

目录

关于手册

与本产品相关的手册如下所述。
请根据需要参考与借助此表。

相关手册

(1) 各 CPU 模块公共部分
基本模式 QCPU、高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 共同相关的手册如下所示：

手册的名称	手册号
QCPU 用户手册（硬件设计 / 维护点检篇） 对 CPU 模块，电源模块，基板，扩展电缆及存储卡等的规格进行说明。 （另卖）	SH-080501CHN
QCPU(Q 模式)/QnACPU 编程手册（公共指令篇） 对顺控程序指令，基本指令及应用指令等的使用方法进行说明。 （另卖）	SH-080450CHN
QCPU(Q 模式)/QnACPU 编程手册（SFC 篇） 对 MELSAP3 的系统构成，性能规格，功能，编程，调试以及出错代码等进行说明。 （另卖）	SH-080283C
QCPU(Q 模式）编程手册（MELSAP-L 篇） 对 MELSAP-L 格式的 SFC 程序的编辑，规格，功能等进行说明。 （另卖）	SH-080076
QCPU(Q 模式)/ 编程手册（结构性文本篇） 对结构性文本语言的编程方法进行说明。 （另卖）	SH-080366E

(2) 基本模型 QCPU

除 “(1) 各 CPU 模块的公共部分”中显示的手册之外，基本模型 QCPU 的相关手册如下所示：

手册的名称	手册号
QCPU(Q 模式)/QnACPU 编程手册 (PID 控制指令篇) 对用于执行 PID 控制的专用指令进行说明。 (另卖)	SH-080240C
CPU 用户手册 (多 CPU 系统篇) 对多 CPU 系统的概要，系统构成，I/O 地址号，CPU 模块间的通讯，输入输出模块 / 智能性功能模块间通讯进行的说明。 (另卖)	SH-080505CHN
Q 系列 MELSEC 通信协议参考手册 对使用串行通信模块 /Ethernet 模块并通过 MC 协议从通信对象设备对 CPU 模块进行数据读取、写入的通信方法及控制步骤进行说明。 (另卖)	SH-080414C

(3) 高性能模型 QCPU

除 “(1)CPU 模块的公共部分”中显示的手册以外，高性能模型 QCPU 的相关手册如下所示：

手册的名称	手册号
QCPU(Q 模式)/QnACPU 编程手册 (PID 控制指令篇) 对用于执行 PID 控制的专用指令进行说明。 (另卖)	SH-080240C
QCPU 用户手册 (多 CPU 系统篇) 对多 CPU 系统的概要，系统构成，I/O 地址号，CPU 模块间的通讯，同输入输出模块 / 智能型功能模块间的通讯进行说明。 (另卖)	SH-080505CHN

(4) 过程 CPU

除 “(1)CPU 模块的公共部分”中显示的手册之外，过程 CPU 的相关手册如下所示：

手册的名称	手册号
QnPHCPU/QnPRHCPU 编程手册 (过程控制指令篇) 对用于执行过程控制的专用指令进行说明。 (另卖)	SH-080449CHN
QCPU 用户手册 (多 CPU 系统篇) 对多 CPU 系统的概要，系统构成，I/O 地址号，CPU 模块间的通讯，输入输出模块 / 智能型功能模块间的通讯进行说明。 (另卖)	SH-080505CHN

(5) 冗余 CPU

除 “(1) 各 CPU 模块的公共部分”中显示的手册之外，冗余 CPU 的相关手册如下所示：

手册的名称	手册号
QnPRHCPU 用户手册（冗余系统篇） 使用冗余 CPU 中，对作为冗余系统构建必要的冗余系统的构成，功能，与外围设备的通讯以及故障的排除进行说明。 （另卖）	SH-080499CHN
QCPU(Q 模式)/QnACPU 编程手册 (PID 控制指令篇） 对用于执行 PID 控制的专用指令进行说明。 （另卖）	SH-080240C
QnPHCPU/QnPRHCPU 编程手册（过程控制指令篇） 对用于执行过程控制的专用指令进行说明。 （另卖）	SH-080449CHN

(6) 通用型 QCPU

除 “(1) 各 CPU 模块的共用部分”中显示的手册之外，通用型 QCPU 的相关手册如下所示：

手册名称	手册号
QCPU 用户手册（多 CPU 系统篇） 对多 CPU 系统的概要，系统构成，I/O 地址号，CPU 模块间的通讯，同输入输出模块 / 智能型功能模块间的通讯进行说明。 （另售）	SH-080505CHN
QCPU(Q 模式)/QnACPU 编程手册 (PID 控制指令篇） 对用于执行 PID 控制的专用指令进行说明。 （另售）	SH-080240C

手册的阅读方法

注意事项的表示

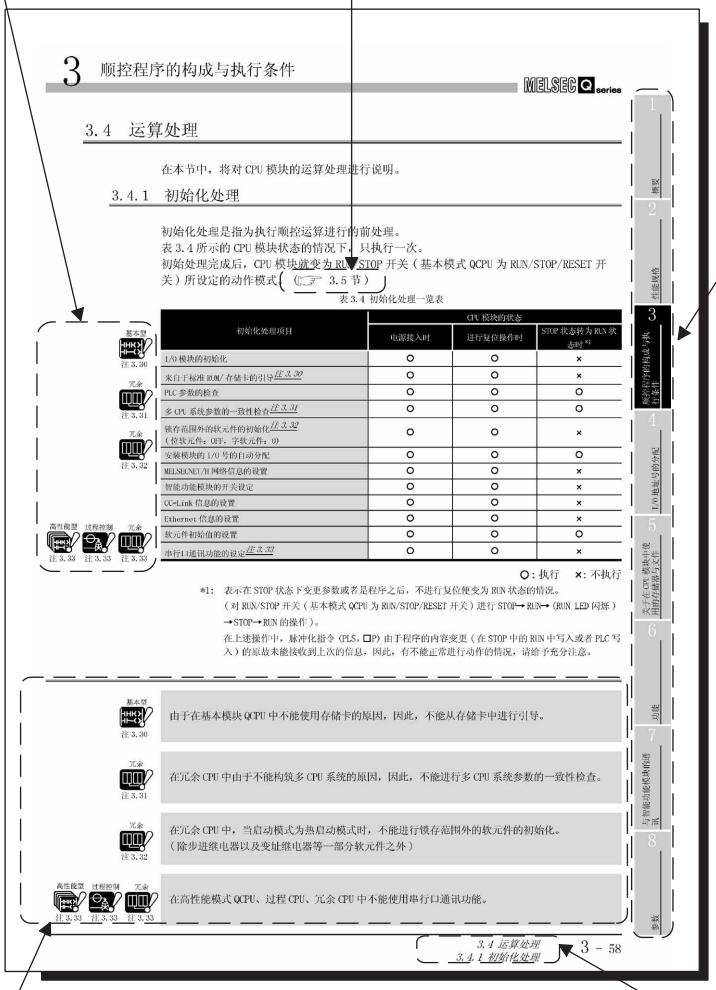
在每个CPU模块都有注意事项的情况下，会有与CPU模块相对应的图标。图标下方的“注●.▲”与正文中页下面的“注●.▲”相对应。但是，节，项的标题没有“注●.▲”。

参照对象的表示

参照对象和参照手册用 ☞ 标记来表示。

章节号的显示

由页右方的索引，翻开页所在的章节一目了然。



注意事项的说明

图标所对应的注意事项的说明。

关于节、项标题的表示

翻开页的节、项一目了然。

图标					内容
基本模型 QCPU	高性能模型 QCPU	过程 CPU	冗余 CPU	通用型 QCPU	
基本	高性能	过程	冗余	通用	有！标记的图标表示功能的一部分中有需要注意的地方。
基本	高性能	过程	冗余	通用	有ｘ标记的图标表示讲述的全部功能不能使用。

另外还有以下种类的说明。

要 点

用相应页的内容来说明特别需要注意的事项以及预先告知的功能。

备 注

说明与相应页内容相关的参照目标以及预先告知会带来方便的内容。

本手册的使用方法

本书是在您使用 Q 系列可编程控制器时，帮助您理解 CPU 模块中的存储升级、功能、程序、软元件等所必需的一本手册。

本手册的构成，大体可以分为以下几个部分：

1) 第 1 章	关于 CPU 概要的说明。
2) 第 2 章～第 5 章	关于 CPU 模块的性能规格、可执行程序、I/O 地址号、存储的说明。
3) 第 6 章	关于 CPU 模块功能的说明。
4) 第 7 章	关于智能功能模块和存取方式的说明。
5) 第 8 章～第 9 章	关于 CPU 模块的参数、软元件的说明。
6) 第 10 章	关于 CPU 模块处理时间的说明。
7) 第 11 章	关于通过 GX Developer 将参数、程序写入 CPU 模块的步骤的说明。

备 注

在本手册中，对于电源模块、基板、扩展电缆、存储卡、电池的规格等方面没有进行说明。
详细的情况，请参照下述手册：
☞ QCPU 用户手册（硬件设计 / 维护点检篇）

关于多 CPU 系统，请参照下述手册：
☞ QCPU 用户手册（多 CPU 系统篇）

关于冗余系统，请参照下述技术手册：
☞ QnPRHCPU 用户手册（冗余系统篇）

本手册所使用的总称与略称

本手册中，除了特别说明的情况以外，使用如下所示的总称与略称来阐述关于 Q 系列 CPU 模块的有关内容。

总称 / 略称	总称 / 略称的内容
基本模式 QCPU	Q00JCPU、Q00CPU、Q01CPU 的总称。
高性能模式 QCPU	Q02CPU、Q02HCPU、Q06HCPU、Q12HCPU、Q25HCPU 的总称。
过程 CPU	Q12PHCPU、Q25PHCPU 的总称。
冗余 CPU	Q12PRHCPU、Q25PRHCPU 的总称。
通用型 QCPU	Q02UCPU、Q03UDCPU、Q04UDHCPU、Q06UDHCPU 的总称。
QnHCPU	Q02HCPU、Q06HCPU、Q12HCPU、Q25HCPU 的总称。
Qn(H)CPU	Q02CPU、Q02HCPU、Q06HCPU、Q12HCPU、Q25HCPU 的总称。
QnPHCPU	Q12PHCPU、Q25PHCPU 的总称。
QnPRHCPU	Q12PRHCPU、Q25PRHCPU 的总称。
QnUCPU	Q02UCPU、Q03UDCPU、Q04UDHCPU、Q06UDHCPU 的总称。
CPU 模块	基本模式 QCPU、高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 的总称。
Q 系列	三菱通用可编程控制器 MELSEC-Q 系列的略称。
AnS 系列	三菱通用可编程控制器 MELSEC-A 系列的小型可编程控制器的略称。
A 系列	三菱通用可编程控制器 MELSEC-A 系列的大型可编程控制器的略称。
GX Developer	Q 系列对应的 SW□D5C-GPPW-E 型 GPP 功能软件包的产品名称。 □表示版本 关于各 CPU 模块使用过程中可能出现 GX Developer 版本的情况，请参阅 QCPU 用户手册（硬件设计 / 维护点检篇）的 “系统配置”。
Q3□B	可以安装 CPU 模块 (Q00JCPU 除外)、Q 系列电源模块、Q 系列 I/O 模块以及智能功能模块的 Q33B、Q35B、Q38B、Q312B 型主基板的总称。
Q3□SB	可以安装基本模式 QCPU(Q00JCPU 除外)、高性能模式 QCPU、超薄型电源模块、Q 系列 I/O 模块、智能功能模块的 Q32SB、Q33SB、Q35SB 超薄型主基板的总称。
Q3□RB	可以安装 CPU 模块 (Q00JCPU 除外)、冗余电源模块、Q 系列 I/O 模块、智能功能模块的 Q38RB 型电源冗余系统用主基板的别称。
Q3□DB	可以安装 CPU 模块 (Q00JCPU 除外)、Q 系列电源模块、Q 系列 I/O 模块、智能功能模块的 Q38DB、Q312DB 型多 CPU 高速主基板的总称。
Q5□B	可以安装 Q 系列 I/O 模块、智能功能模块的 Q52B、Q55B 型扩展基板的总称。
Q6□B	可以安装 Q 系列电源模块、Q 系列 I/O 模块、智能功能模块的 Q63B、Q65B、Q68B、Q612B 型扩展基板的总称。
Q6□RB	可以安装冗余电源模块、Q 系列 I/O 模块、智能功能模块的 Q68RB 型电源冗余系统用扩展基板的别称。
Q6□WRB	可以安装冗余电源模块、Q 系列 I/O 模块、智能功能模块的 Q65WRB 型冗余扩展基板的别称。
QA1S6□B	可以安装 AnS 系列电源模块、AnS 系列 I/O 模块、特殊功能模块的 QA1S65B、QA1S68B 型扩展基板的总称。

总称 / 略称	总称 / 略称的内容
QA6□B	可以安装 A 系列电源模块、A 系列 I/O 模块、特殊功能模块的 QA65B、QA68B 型扩展基板的总称。
主基板	Q3□B、Q3□SB Q3□RB 和 Q3□DB 的总称。
扩展基板	Q5□B、Q6□B、Q6□RB、Q6□WRB、QA1S6□B 和 QA6□B 的总称。
超薄型主基板	Q3□SB 的别称。
电源冗余主基板	Q3□RB 的别称。
电源冗余扩展基板	Q6□RB 的别称。
冗余扩展基板	Q6□WRB 的别称。
多 CPU 高速主基板	Q3□DB 的别称。
基板	主基板、扩展基板、超薄型主基板、电源冗余主基板、电源冗余扩展基板、冗余扩展基板、多 CPU 高速主基板的总称。
冗余基板	电源冗余主基板、电源冗余扩展基板、冗余扩展基板的总称。
扩展电缆	QC05B、QC06B、QC12B、QC30B、QC50B、QC100B 型扩展电缆的总称。
热备电缆	QC10TR、QC30TR 型冗余 CPU 所用的热备电缆的总称。
Q 系列电源模块	Q61P-A1、Q61P-A2、Q61P、Q62P、Q63P、Q64P 型电源模块的总称。
超薄型电源模块	Q61SP 超薄型电源模块的略称
冗余电源模块	Q63RP、Q64RP 型冗余系统所用电源模块的总称。
AnS 系列电源模块	A1S61PN、A1S62PN、A1S63PN 型电源模块的总称。
电源模块	Q 系列电源模块、AnS 系列电源模块、超薄型电源模块、冗余电源模块的总称。
电池	Q6BAT、Q7BAT、Q8BAT 型 CPU 模块所用电池、Q2MEM-BAT 型 SRAM 卡所用的电池的总称。
SRAM 卡	Q2MEM-1MBS、Q2MEM － 2MBS、Q3MEM-4MBS、Q3MEM-8MBS 型 SRAM 卡的总称。
Flash 卡	Q2MEM-2MBF、Q2MEM － 4MBF 型 Flash 卡的总称。
ATA 卡	Q2MEM-8MBA、Q2MEM-16MBA、Q2MEM-32MBA 型 ATA 卡的总称。
存储卡	SRAM 卡、Flash 卡、ATA 卡的总称。
个人计算机 CPU 模块	Contec（康泰克）公司制造的 MELSEC-Q 系列兼容的个人计算机 CPU 模块的略称。
控制系统	被指定为控制系统的冗余 CPU 的略称。
待机系统	被指定为待机系统的冗余 CPU 的略称。
A 系统	被指定为 A 系统的冗余 CPU 的略称。
B 系统	被指定为 B 系统的冗余 CPU 的略称。

第 1 章　概要

本手册将对 Q 系列 CPU 模块中的（☞如下面的 (1) 所示）程序、I/O 地址号分配方法、功能、软元件进行说明。

关于电源模块、基板、扩展电缆、存储卡、电池等问题，请参照下面的手册。
☞ QCPU 用户手册（ 硬件设计 / 维护点检篇 ）

(1) 与本手册相对应的 CPU 模块

与本手册相对应的 CPU 模块，如表 1.1 所示。

表 1.1 与本手册相对应的 CPU 模块一览表

对应的 CPU 模块	对应 CPU 模块的型号
基本模式 QCPU	Q00JCPU，Q00CPU，Q01CPU
高性能模式 QCPU	Q02CPU，Q02HCPU，Q06HCPU，Q12HCPU，Q25HCPU
过程 CPU	Q12PHCPU，Q25PHCPU
冗余 CPU	Q12PRHCPU，Q25PRHCPU
通用型 QCPU	Q02UCPU，Q03UDCPU，Q04UDHCPU，Q06UDHCPU

(2) Q 系列 CPU 模块相关手册一览表

与 Q 系列 CPU 模块相关的内容，有下述的手册。关于下述所示的手册号码的详细情况，请参照本手册的 “关于手册”。

(a) 基本模式 CPU

表 1.2 与基本模式 QCPU 相关的用户手册一览表

目的	QCPU(Q 模式)CPU 模块用户手册（硬件篇）	QCPU 用户手册（硬件设计 / 维护点检篇）	QCPU 用户手册（功能解说 / 程序基础篇）	QCPU 用户手册（多 CPU 系统篇）	QnPRHCPU 用户手册（冗余系统篇）
	硬件篇 (压缩)	维护和检查篇	程序基础篇	多CPU系统篇	冗余系统篇
CPU 模块各部件名称规格的确认	概要	详细	概要		
电源模块、基板和 I/O 模块连接方法的确认	概要	详细			
单 CPU 系统的构造（启动顺序和 I/O 地址号分配的方法确认）		详细			
多 CPU 系统的构造（启动顺序和 I/O 地址号分配的方法确认）				详细	
顺控程序组态和内存的确认			详细		
CPU 模块的功能、参数和软元件的确认			详细		
故障排除和出错代码的确认		详细			

表 1.3 基本模式 QCPU 编程手册列表

目的	QCPU(Q 模式)/QnACPU 编程手册（公共指篇）	QCPU(Q 模式)/QnACPU 编程手册（PID 控制指令篇）	QnPHCPU/QnPRHCPU 编程手册（过程控制指令篇）	QCPU(Q 模式)/QnACPU 编程手册（SFC 篇）	QCPU(Q 模式）编程手册（MELSAP-L 篇）	QCPU(Q 模式）编程手册（结构化文本篇）
顺控程序指令、基本指令和应用指令等使用方法的确认	详细					
PID 控制专用指令确认		详细				
MELSAP3 的系统组态、性能规格、功能、编程、调试和出错代码的确认				详细		
用于 MELSAP-L 类型的 SFC 编程需要的编程方法、规格和功能等的确认					详细	
结构化文本语言的编程方法的确认						详细

(b) 高性能型 QCPU

表 1.4 高性能模式 QCPU 的用户手册一览

目的	QCPU(Q 模式）类型 CPU 模块用户手册（硬件篇）	QCPU 用户手册（硬件设计／维护点检篇）	QCPU 用户手册（功能解说／程序基础篇）	QCPU 用户手册（多 CPU 系统篇）	QnPRHCPU 用户手册（冗余系统篇）
	硬件篇（压缩）	维护和检查篇	编程基础篇	多CPU系统篇	冗余系统篇
CPU 模块的部件名称和规格的确认	概要	详细	概要		
电源模块、基本单元和 I/O 模块的连接方法的确认	概要	详细			
单 CPU 系统的构造（启动顺序和 I/O 地址号分配方法的确认）		详细			
多 CPU 系统的构造（启动顺序和 I/O 地址号分配方法的确认）				详细	
顺控程序组态和内存的确认			详细		
CPU 模块的功能、参数和软元件的确认			详细		
故障排除和出错代码的确认		详细			

表 1.5 高性能模式 QCPU 的用户手册一览

目的	QCPU(Q 模式)/ QnACPU 编程手册（公共指令篇）	QCPU(Q 模式)/ QnACPU 编程手册（PID 控制指令篇）	QnPHCPU/QnPRHCPU 编程手册（过程控制指令篇）	QCPU(Q 模式)/ QnACPU 编程手册（SFC 篇）	QCPU(Q 模式）编程手册（MELSAP-L 篇）	QCPU(Q 模式）编程手册（结构化文本篇）
顺控程序指令、基本指令和应用指令等使用方法的确认	详细					
PID 控制专用指令确认		详细				
MELSAP3 的系统组态、性能规格、功能、编程、调试和出错代码的确认				详细		
用于 MELSAP-L 类型的 SFC 编程需要的编程方法、规格和功能等的确认					详细	
结构化文本语言的编程方法的确认						详细

(c) 过程控制 CPU

表 1.6 过程控制 CPU 的用户手册列表

目的	QCPU(Q 模式）类型 CPU 模块用户手册（硬件篇）	QCPU 用户手册（硬件设计／维护点检篇）	QCPU 用户手册（功能解说／程序基础篇）	QCPU 用户手册（多 CPU 系统篇）	QnPRHCPU 用户手册（冗余系统篇）
					
CPU 模块的部件名称和规格的确认	概要	详细	概要		
电源模块、基板和 I/O 模块的连接方法的确认	概要	详细			
单 CPU 系统的构造（启动顺序和 I/O 地址号分配方法的确认）		详细			
多 CPU 系统的构造（启动顺序和 I/O 地址号分配方法的确认）				详细	
顺控程序组态和内存的确认			详细		
CPU 模块的功能、参数和软元件的确认			详细		
故障排除和出错代码的确认		详细			

表 1.7 过程控制 CPU 的用户手册列表

目的	 QCPU(Q 模式)/ QnACPU 编程手册（公共指令篇）	 QCPU(Q 模式)/ QnACPU 编程手册（PID 控制指令篇）	 QnPHCPU/QnPRHCPU 编程手册（过程控制指令篇）	 QCPU(Q 模式)/ QnACPU 编程手册（SFC 篇）	 QCPU(Q 模式）编程手册（MELSAP-L 篇）	 QCPU(Q 模式）编程手册（结构化文本篇）
顺序指令、基本指令和应用指令等使用方法的确认	详细					
过程控制专用指令确认			详细			
MELSAP3 的系统组态、性能规格、功能、编程、调试和出错代码的确认				详细		
用于 MELSAP-L 类型的 SFC 编程需要的编程方法、规格和功能等的确认					详细	
结构化文本语言的编程方法的确认						详细

(d) 冗余 CPU

表 1.8 冗余 CPU 的手册一览表

目的	 QCPU(Q 模式）类型 CPU 模块用户手册（硬件篇）	 QCPU 用户手册（硬件设计／维护点检篇）	 QCPU 用户手册（功能解说／程序基础篇）	 QCPU 用户手册（多 CPU 系统篇）	 QnPRHCPU 用户手册（冗余系统篇）
CPU 模块规格及部件名称的确认	概要	详细	概要		
电源模块、基板和 I/O 模块的连接方法的确认	概要	详细			
冗余系统的构造（启动顺序和 I/O 地址号码分配方法的确认）					详细
顺控程序组态和内存的确认			详细		
CPU 模块的功能、参数和软元件等的确认			详细		
故障排除的确认					详细
出错代码的确认		详细			

表 1.9 冗余 CPU 的编程手册

目的	QCPU(Q 模式)/QnACPU 编程手册（公共指令篇）	QCPU(Q 模式)/QnACPU 编程手册（PID 控制指令篇）	QnPHCPU/QnPRHCPU 编程手册（过程控制指令篇）	QCPU(Q 模式)/QnACPU 编程手册（SFC 篇）	QCPU(Q 模式）编程手册（MELSAP-L 篇）	QCPU(Q 模式）编程手册（结构化文本篇）
顺控程序指令、基本指令和应用指令等使用方法的确认	详细					
PID 控制专用指令确认		详细				
过程控制专用指令确认			详细			
MELSAP3 的系统组态、性能规格、功能、编程、调试和出错代码的确认				详细		
用于 MELSAP-L 类型的 SFC 编程需要的编程方法、规格和功能等的确认					详细	
结构化文本语言的编程方法的确认						详细

(e) 通用型 QCPU

表 1.10 通用型 QCPU 的手册一览

目的	QCPU(Q 模式）类型 CPU 模块用户手册（硬件篇）	QCPU 用户手册（硬件设计／维护点检篇）	QCPU 用户手册（功能解说／程序基础篇）	QCPU 用户手册（多 CPU 系统篇）	QnPRHCPU 用户手册（冗余系统篇）
					
CPU 模块的部件名称和规格的确认	概要	详细	概要		
电源模块、基板和 I/O 模块的连接方法的确认	概要	详细			
单 CPU 系统的构筑（启动步骤和 I/O 地址号分配方法的确认）		详细			
多 CPU 系统的构筑（启动步骤和 I/O 地址号分配方法的确认）				详细	
顺控程序组态和内存的确认			详细		
CPU 模块的功能、参数和软元件的确认			详细		
故障排除和出错代码的确认		详细			

表 1.11 通用型 QCPU 的编程手册

目的	QCPU(Q 模式)/ QnACPU 编程手册（公共指令篇）	QCPU(Q 模式)/ QnACPU 编程手册（PID 控制指令篇）	QnPHCPU/QnPRHCPU 编程手册（过程控制指令篇）	QCPU(Q 模式)/ QnACPU 编程手册（SFC 篇）	QCPU(Q 模式）编程手册（MELSAP-L 篇）	QCPU(Q 模式）编程手册（结构化文本篇）
顺控程序命令、基本命令和应用命令等使用方法的确认	详细					
PID 控制专用命令确认		详细				
MELSAP3 的系统组态、性能规格、功能、编程、调试和出错代码的确认				详细		
用于 MELSAP-L 类型的 SFC 编程需要的编程方法、规格和功能等的确认					详细	
结构化文本语言的编程方法的确认						详细

1.1 特点

对各个 CPU 模块的特点进行说明。

1.1.1 基本模式 QCPU 的特点

基本模式 QCPU 特有的特点如下所示。

(1) 最适合小规模系统的成本性能

针对以小规模系统为对象的简单紧凑的系统的控制，基本模式 QCPU 是最适合的模块。
基本模式 QCPU 可以实现对小规模系统的最适合的成本性能。

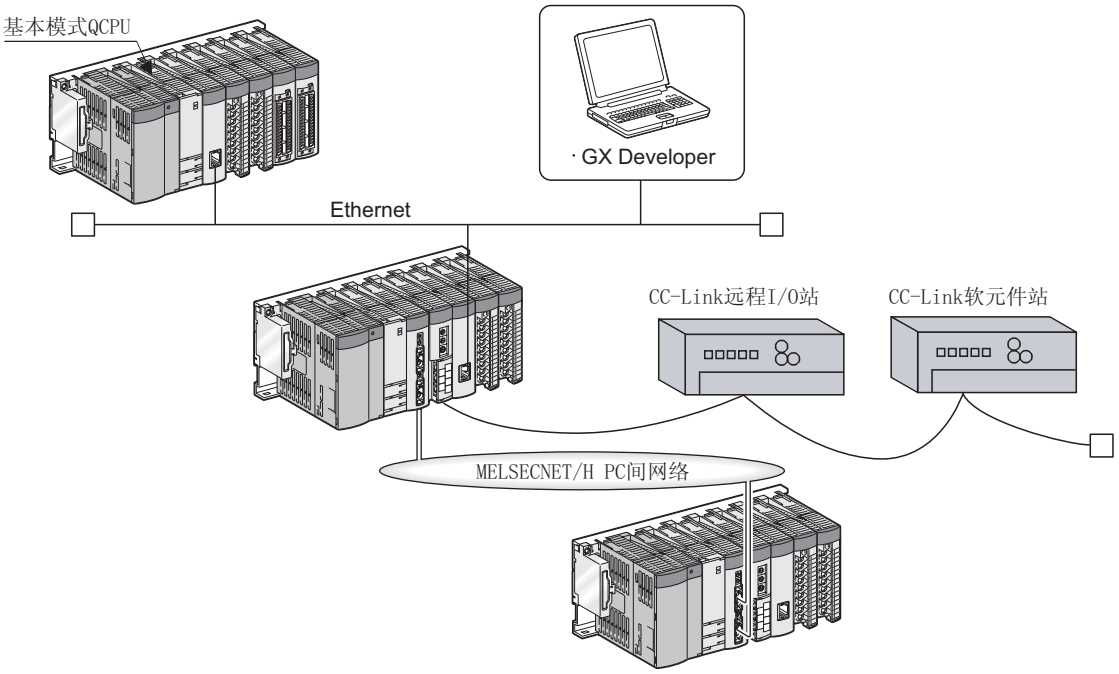


图 1.1 使用基本模式 QCPU 的系统示例

(2) 通过串行口通讯功能与个人计算机、显示器的通讯
(☞ 6.23 节)

Q00CPU，Q01CPU 可以通过 RS-232 接口与个人电脑 / 显示器等连接、通过 MELSEC 通讯协议（以下简称 MC 协议）进行通讯。

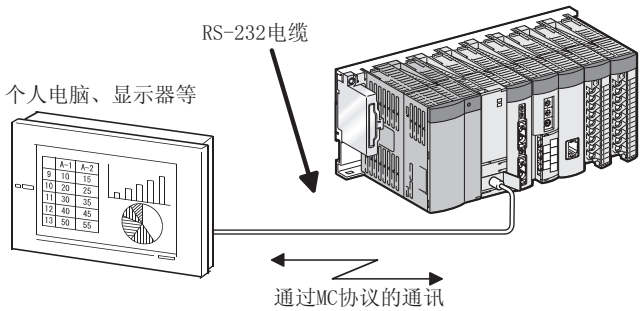


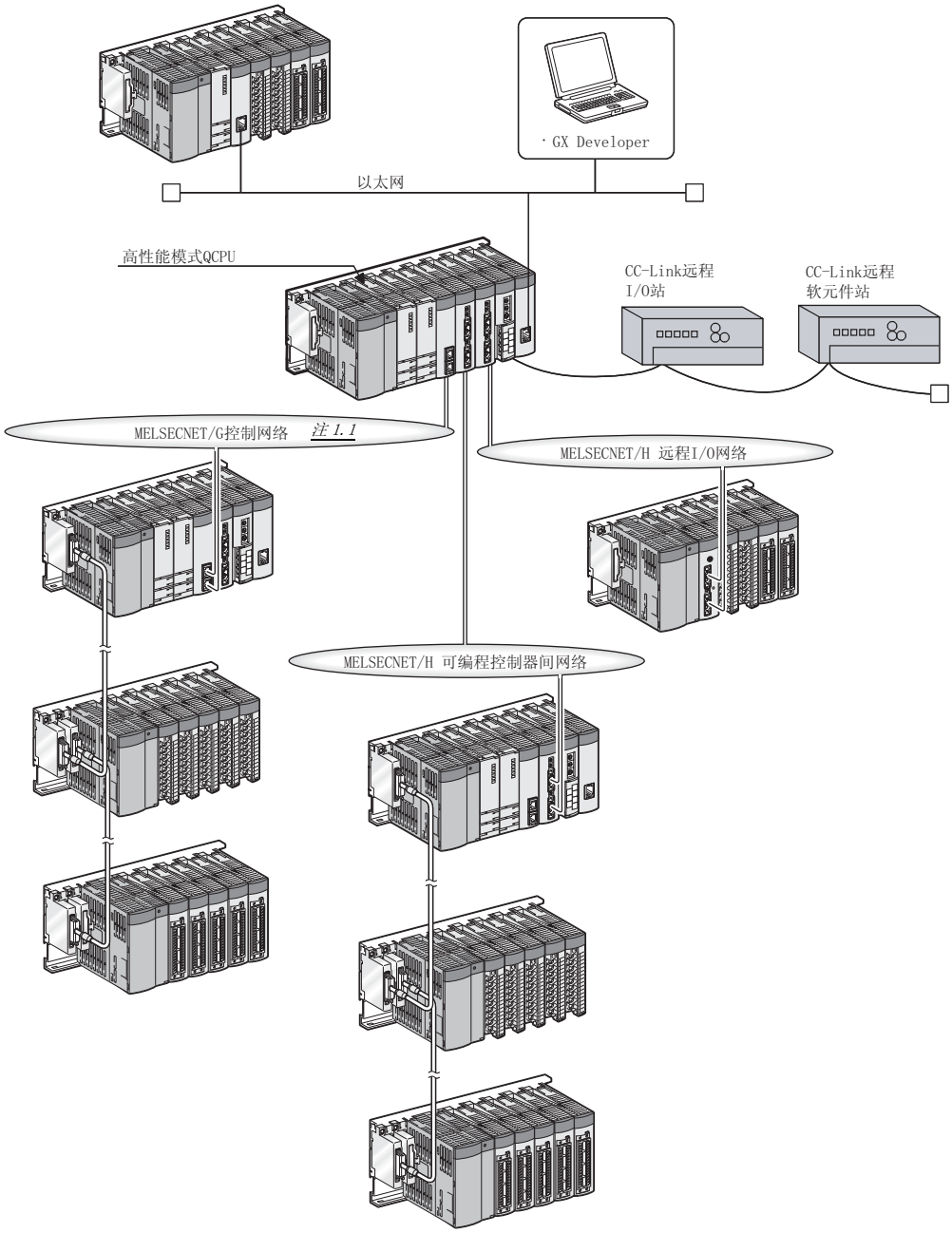
图 1.2 与个人电脑、显示器的通讯

1.1.2 高性能模式 QCPU 的特点

高性能模式 QCPU 特有的特点如下所示。

(1) 高性能的大容量

针对小规模系统到大规模系统，高性能模式 QCPU 是可以进行大容量高速处理的模块。
高性能模式 QCPU 可以构筑最适合的高性能的设备。



高性能
注 1.1

图 1.3 使用高性能模式 QCPU 的系统示例

高性能
注 1.1

在高性能模式 QCPU 中使用 MELSECNET/G 时，应确认 CPU 模块及 GX Developer 的版本。
(☞ 附录 4.2 节)

(2) AnS/A 系列的 I/O 模块、特殊功能模块的使用可能性

高性能模式 QCPU 通过 AnS/A 系列对应的扩展基板 (QA1S6□B、QA6□B)，可以使用 AnS/A 系列的 I/O 模块及特殊功能模块。

备 注

关于高性能模式 QCPU 由于功能升级而新增的功能，请参阅附录 4.2 节。

1.1.3 过程 CPU 的特点

对过程 CPU 特有的特点进行说明。

(1) 作为过程控制指令追加 52 指令

在高性能模式 QCPU 的基础上追加了与高度过程控制相对应的 52 指令。
追加指令的详细情况，请参照下述手册。

☞ QnPHCPU/QnPRHCPU 编程手册（过程控制指令篇）

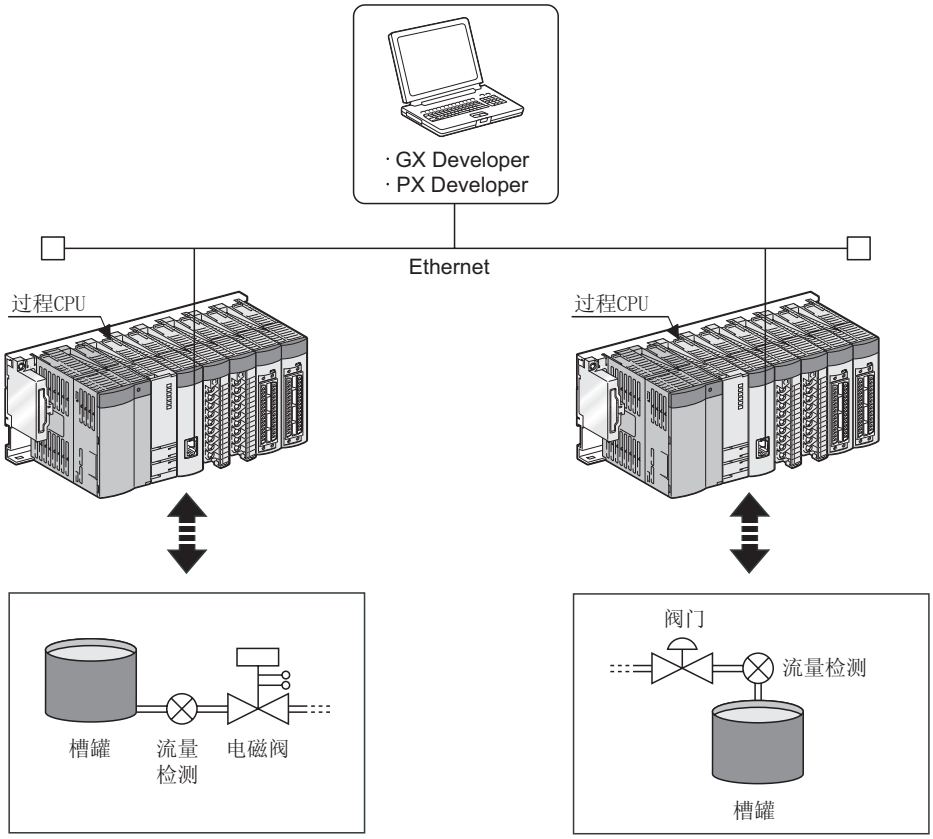


图 1.4 过程 CPU 的动作

(2) 采用 2 个自由度的 PID 控制方式

通过采用 2 个自由度的 PID 控制方式，可以达到目标值变动、外部干扰变动两方面的最适合的应答。

(3) 自动调谐功能 (PID 常量初始值的设定)

通过自动调谐功能，可以实现控制参数调整作业的自动化、调整作业的简单化、操作与控制工程的省力化、消除调整结果的个人误差。
关于自动调谐功能的详细情况，请参照下述手册。

☞ QnPHCPU/QnPRHCPU 编程手册（过程控制指令篇）

(4) 可以进行在线中的模块更换（在线模块更换）

在模块发生故障时，不停止系统可以更换模块。

Q 系列的 I/O 模块、功能版本 C 的 A/D 变换模块、D/A 变换模块、温度输入模块、温度调节模块、脉冲输入模块可以进行在线模块更换。

在线模块的更换的详细情况，请参照下述手册。

☞ QCPU 用户手册（ 硬件设计 / 维护点检篇 ）

☞ 与在线模块更换相对应的模块的手册。

(5) MELSECNET/H 多重远程 I/O 系统构筑的可能

安装 MELSECNET/H 远程主站，可以构筑 MELSECNET/H 多重远程 I/O 系统。

(6) 与计量装置专用软件包 (PX Developer) 对应

如果使用过程控制用软件包 (PX Developer)，用 FB（功能块）可以简单制作 PID 控制的程序。

通过组合使用过程控制专用软件包 (PX Developer)，可以提供优异的工程环境。

☒ 要 点

1. 使用过程 CPU 时，应使用 GX Developer Version 7.10L 以后的产品。
2. PX Developer 应与 GX Developer Version 7.20W 以后的产品组合使用。
   关于 PX Developer 的详细情况，请参阅 PX Developer 的手册。

1.1.4 冗余 CPU 的特点

在下面说明冗余 CPU 的特有特点。

(1) 与冗余系统相对应的、过程 CPU 的功能补充

(a) 使用冗余 CPU 的冗余系统

根据使用的冗余 CPU，包括基板、电源模块、CPU 模块（冗余 CPU）在内的系统都可以进行冗余化。

在控制系统中即使发生故障，通过向备机系统进行切换，可以实现系统的高可靠性。

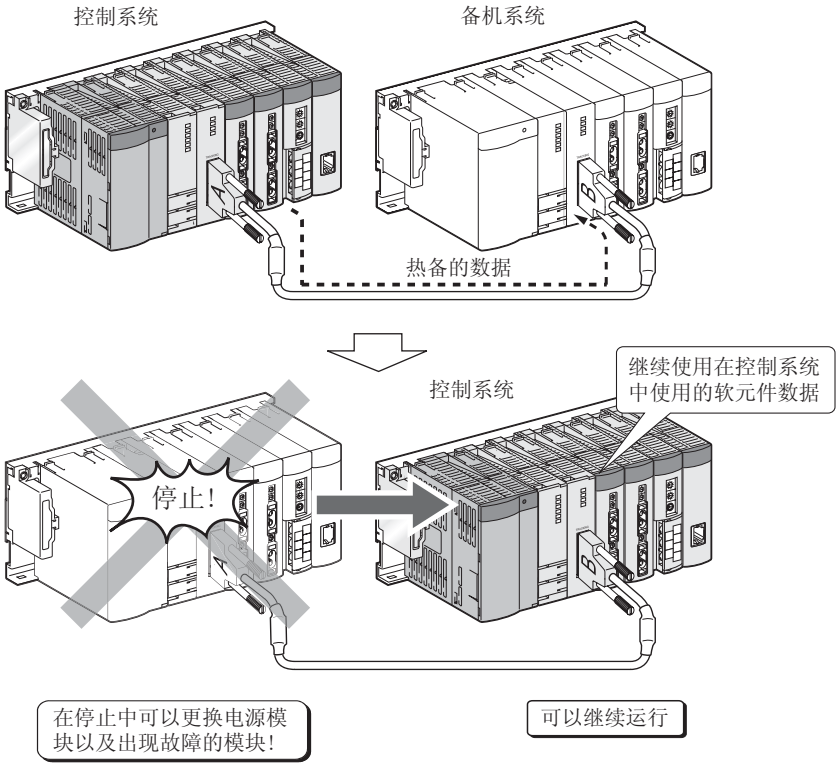


图 1.5 冗余 CPU 的动作

(b) 电源冗余系统

另外，通过在远程 I/O 站上使用电源冗余主基板 (Q3□RB) 以及冗余电源模块 (Q63RP、Q64RP)，可以实现远程 I/O 站的电源冗余化。

据此，即使远程 I/O 站的电源模块发生故障，也可以在不停运系统的状况下更换电源模块。

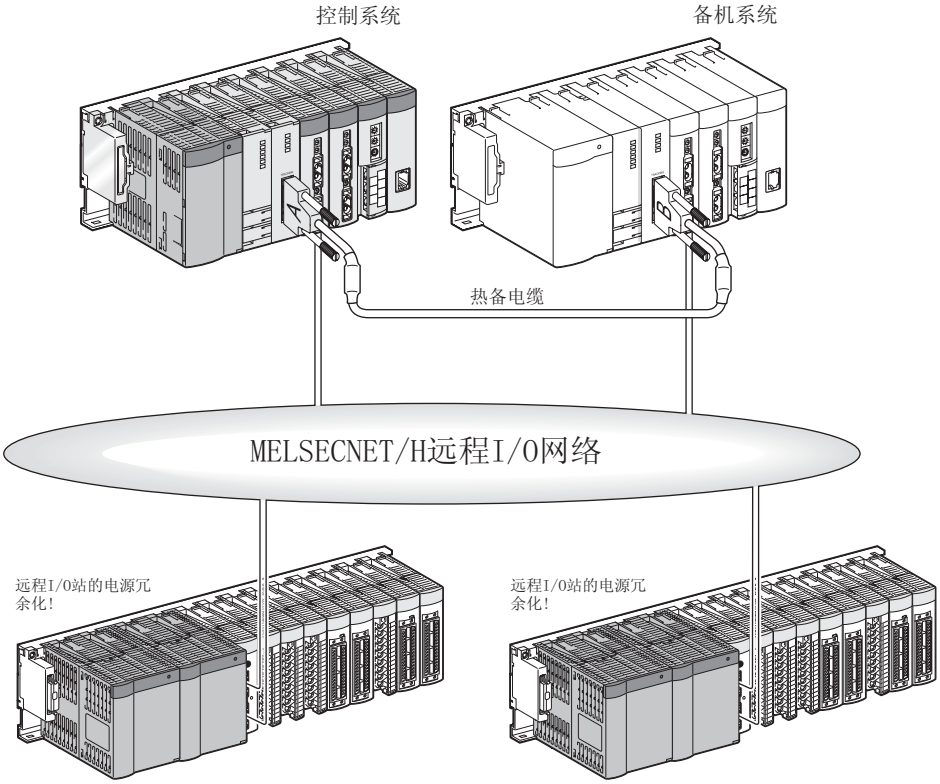


图 1.6 电源冗余系统

要 点

1. 使用冗余 CPU 时，应使用 GX Developer Version8.18U 以后的产品。
2. PX Developer 应与 GX Developer 组合使用。
   使用冗余 CPU 时，应使用 PX Developer Version 1.06G 以后的产品。
   关于 PX Developer 的详细情况，请参阅 PX Developer 的手册。

备 注

关于冗余 CPU 的详细特点、功能请参照下述手册。

☞ QnPRHCPU 用户手册（冗余系统篇）

1.1.5 通用型 QCPU 的特点

以下介绍通用型 QCPU 的特点。

(1) 更进一步实现高速化

与以前的产品相比，通用型 QCPU 的基本命令处理时间、浮点运算处理时间、文件寄存器的存取处理时间达到了更进一步的高速化。

此外，通过使用通用运算寄存器 (Z)*，也实现了寄存器运算（传送命令）处理的高速化。

*: 寄存器运算中使用的变址寄存器被称为通用运算寄存器。
（ 9.6.2 项）

(2) 大容量文件寄存器

CPU 模块内部最大可设置 384k 字 (Q06UDHCPU 时）的文件寄存器。

(3) 可以使用浮点双精度运算命令

可以在浮点单精度运算命令的基础上使用新的浮点双精度运算命令 (64 位命令）。
（ 3.9.4 项）
由此，可以实现高精度的模拟控制及定位控制。

(4) 可以在 32 位的范围内进行变址修饰

通过将变址修饰范围扩展为 32 位，可以对文件寄存器的整个区域进行变址修饰。
（ 9.6.1 项）

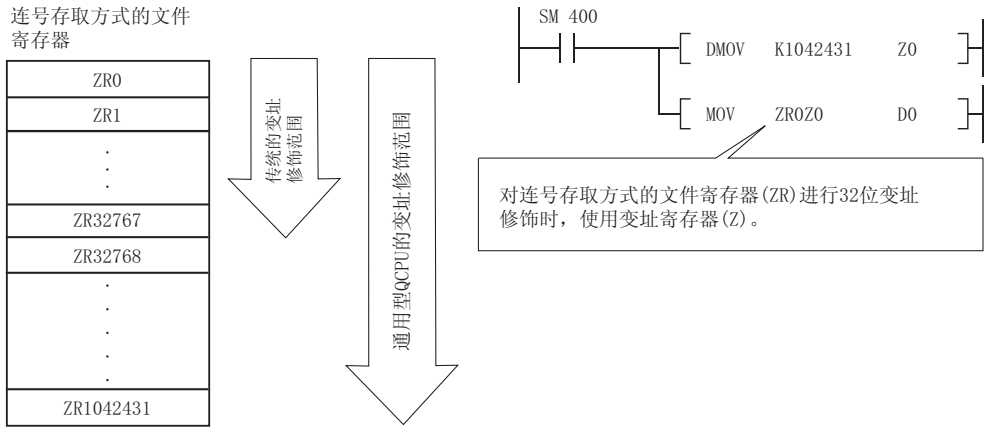


图 1.7 32 位变址修饰

1.2 程序的存储与运算

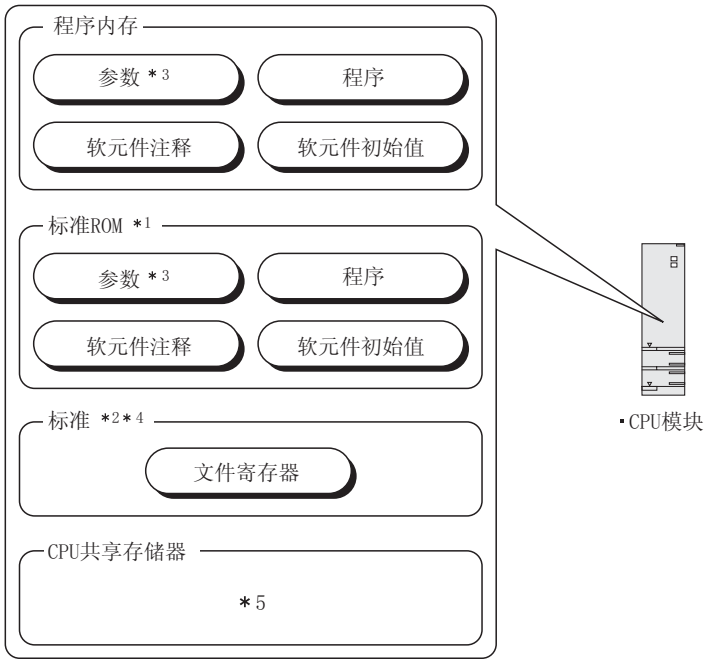
(1) 程序的存储

基本 注 1.2

(a) 通过 GX Developer 制作的程序的存储

通过 GX Developer 制作的程序可以存储在 CPU 模块的程序内存、标准 ROM 或者存储卡*注 1.2* 中。

1) 基本模式 QCPU



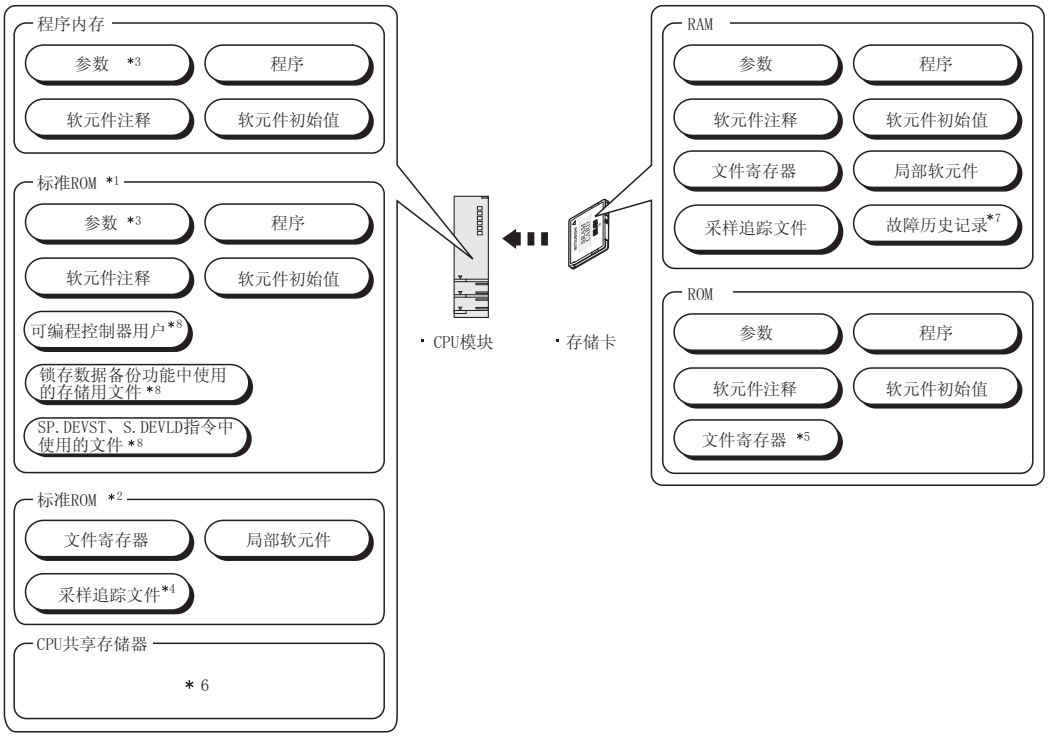
*1: 标准 ROM 在程序内存的 ROM 化时使用。
*2: 标准 RAM 为文件寄存器用的存储器。
*3: 包括通过 GX Configurator 设定的智能功能模块的参数。
*4: Q00JCPU 没有标准 RAM。
不能使用文件寄存器
*5: 关于 CPU 共享存储器请参照下述模块。
☞ QCPU 用户手册（多 CPU 系统篇）

图 1.8 基本模式 QCPU 的存储器构成与存储对象

基本 注 1.2

在基本模式 QCPU 中不能使用存储卡。

2）高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU



*1： 标准 ROM 在以下情况下使用。
- 高性能模式 QCPU、过程 CPU、冗余 CPU 的情况下
  程序内存的 ROM 化时使用。
- 通用型 QCPU 的情况下
  用于软元件注释、用户数据存储。

*2： 标准 RAM 是在未安装存储卡的情况下用于文件寄存器、局部软元件、采样追踪文件的存储器。
另外，标准 RAM 在文件寄存器存取高速化时使用。

*3： 包括通过 GX Configurator 中设置的智能功能模块的参数。

*4： 将采样追踪文件存储到标准 RAM 中时，应确认 CPU 模块及 GX Developer 的版本。
（ 附录 4）

*5： 只在 Flash 卡的情况下可以使用。
在顺控程序中只能进行读取。

*6： 关于 CPU 共享存储器请参阅下述手册。
QCPU 用户手册（多 CPU 系统篇）

*7： 在通用型 QCPU 中，不能将故障历史记录存储到存储卡中。

*8： 仅在使用通用型 QCPU 时才可以存储。

图 1.9 高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 的存储器构成与存储对象

(b) 程序的执行

CPU 模块对存储在程序内存中的程序进行运算。

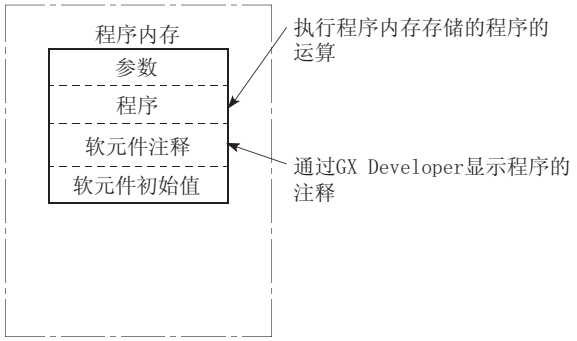


图 1.10 执行存储的程序

(c) 存储在标准 ROM/ 存储卡的程序的执行*注 1.3*

基本 注 1.3

程序等可以存储在标准 ROM/ 存储卡中。

存储在标准 ROM/ 存储卡上的程序在可编程控制器的电源处于 OFF → ON 或者 CPU 模块处于复位解除时，可以将程序从程序存储器中引导出（读出）并执行。

由于可将程序等存储在标准 ROM/ 存储卡上，在没有备用电池的情况下可以对程序等进行保存。

1) 基本模式 QCPU

从标准 ROM 向程序内存进行引导时，有必要通过 GX Developer 设定可编程控制器参数的引导文件。

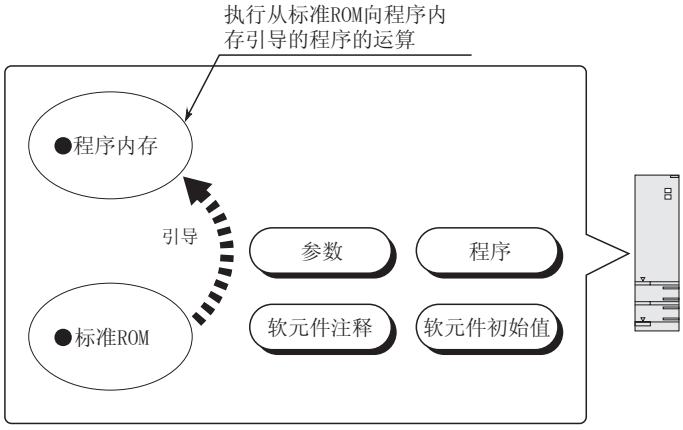


图 1.11 基本模式 QCPU 的引导运行

2) 高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU

通用 UD 注 1.4

向程序内存内进行引导*注 1.4*时，通过 GX Developer 进行可编程控制器参数的引导文件设定后，设置插杆开关的参数有效驱动。*注 1.5*

通用 UD 注 1.5

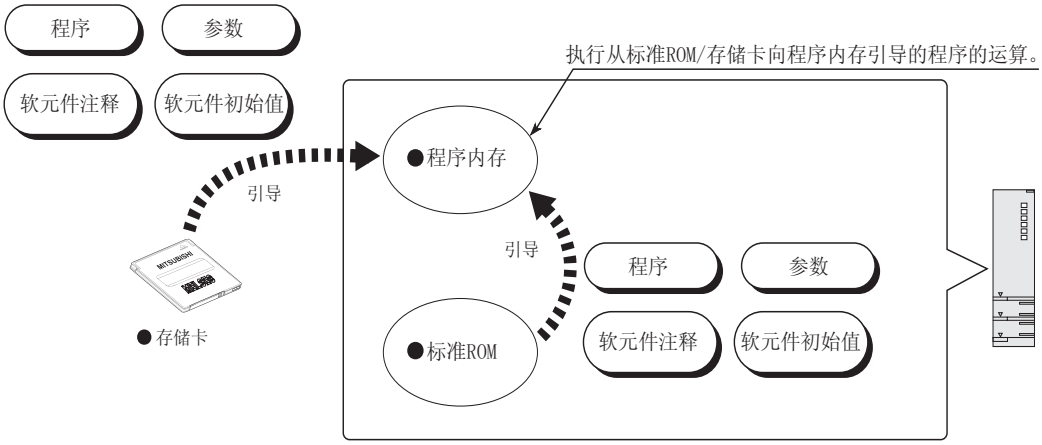


图 1.12 高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 的引导运行

基本 注 1.3

在基本模式 QCPU 中，由于不能使用存储卡，不能将程序存储在存储卡上。

通用 UD 注 1.4

在通用型 QCPU 中，不能执行从标准 ROM 向程序内存的引导。

（ 5.2.3 项）

通用 UD 注 1.5

在通用型 QCPU 中，不能通过插杆开关进行参数有效驱动设置。

（ 5.2.11 项）

(2) 构造化程序

CPU 模块的程序可以进行构造化。

通过程序的构造化，可以制成不同类别的工程、功能程序。

程序的构造化可以进行同一程序内的构造化（☞本节 (2)(a)）。以及分割文件的构造化（☞本节 (2)(b)）。

(a) 在同一程序内进行的构造化

通过制作副程序（☞ 3.1.2 项）或者中断程序（☞ 3.1.3 项）可以进行在同一程序内的构造化。

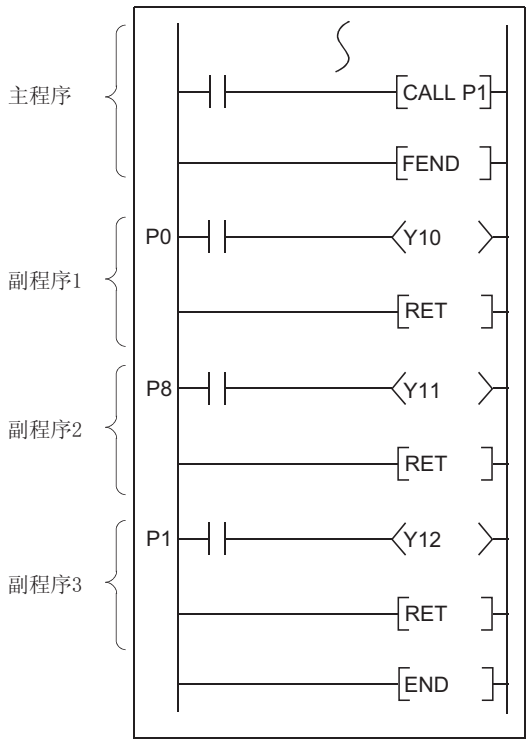


图 1.13 在同一程序内的构造化示例

基本
注 1.6

(b) 文件的分别制作*注 1.6*

程序以文件的形式存储。*1
在 CPU 模块中通过改变文件名，可以存储多个程序。

*1: 程序存储的可能对象，会因 CPU 模块的不同而不同。
(☞ 第 5 章)

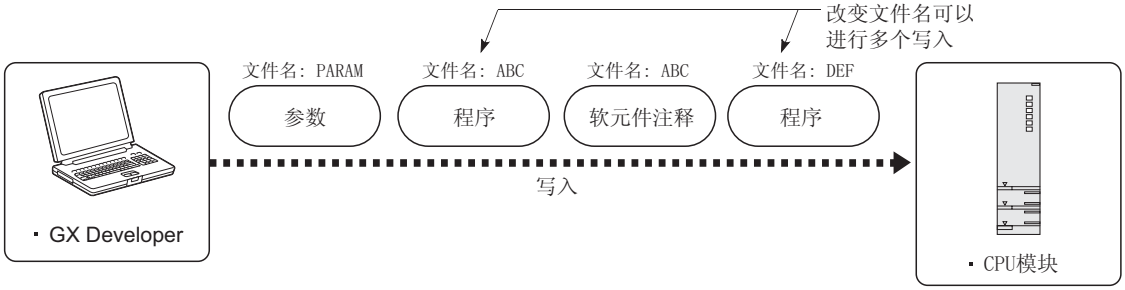
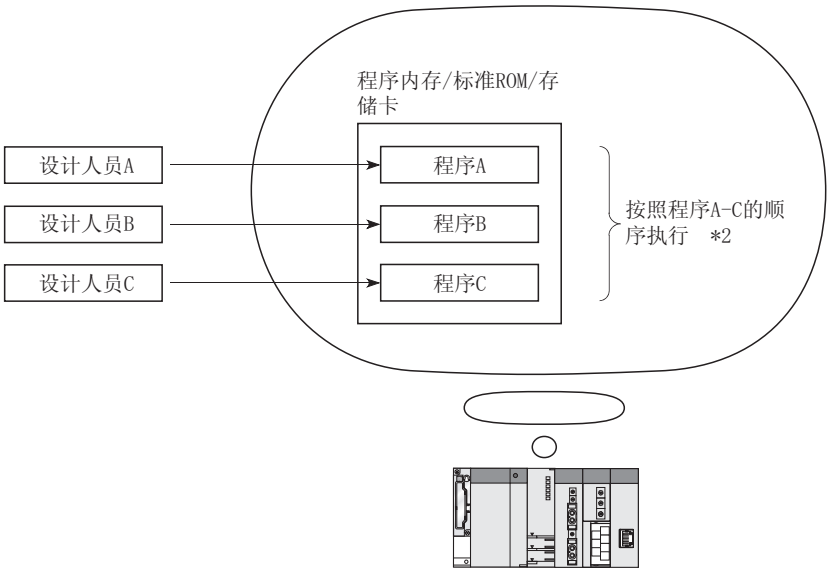


图 1.14 分割文件进行的构造化

由此，可以由多个设计人员分别制作程序，并对不同类别的工程、功能程序进行分别管理、保养。
另外，即使发生规格变更时，可以只相应的程序进行修正、调试等处理。

1) 由多个设计人员分别制作的程序的情况



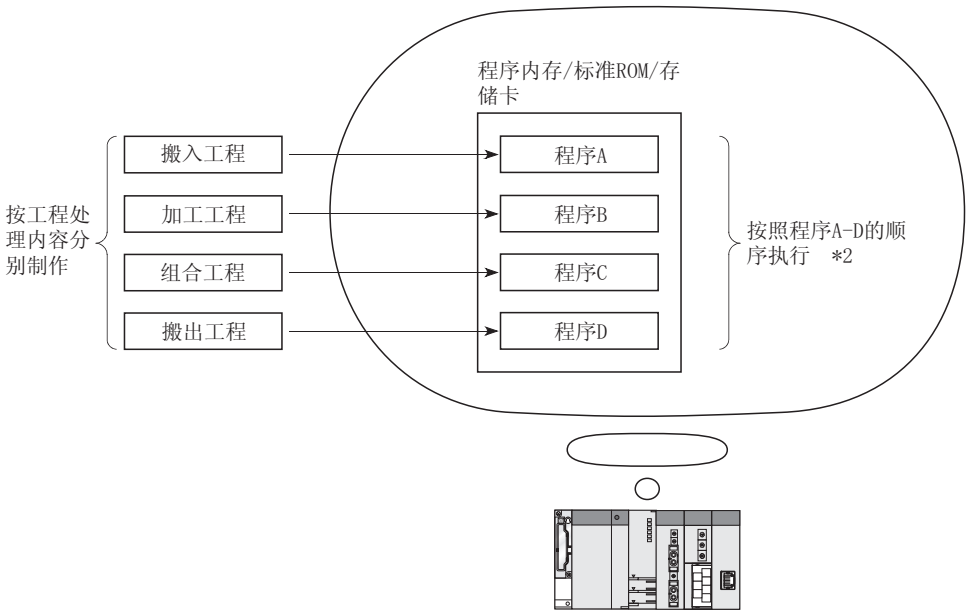
*2: 根据程序设定 (☞ 3.3.6 项 (1))，可以设定程序的执行顺序、执行条件。

图 1.15 程序的分割制作 (不同的设计人员)

基本
注 1.6

在基本模式 QCPU 中，不能通过变换文件名来进行多个程序的存储，因此，文件不能分别进行构造化。

2）按工程类别分别制作程序的情况 *1

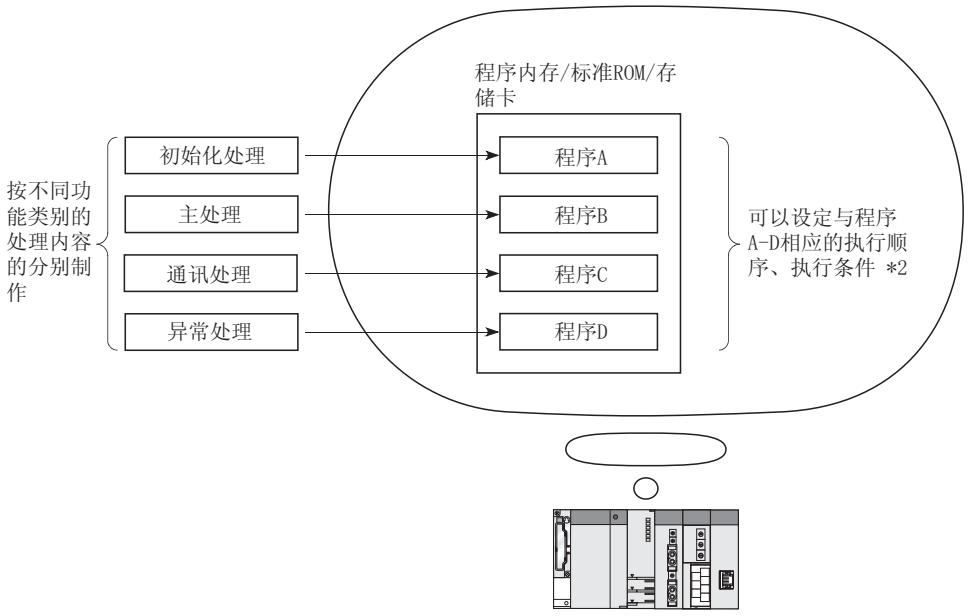


*1: 不同工程的处理内容可以进一步在不同的功能中分别管理。

*2: 根据程序设定（☞ 3.3.6 项 (1)），可以设定程序的执行顺序、执行条件。

图 1.16 程序的分别制作（按不同的工程）

3）按功能类别中进行分别制作的情况



*2: 根据程序设定（☞ 3.3.6 项 (1)），可以设定程序执行顺序、执行条件。

图 1.17 程序的分别制作（按不同的功能）

1.3 对于编程的便利的软元件、指令

在 CPU 模块中，对程序的制作有便利的软元件、指令。
主要的内容如下所示。

(1) 灵活的软元件指定

Q 系列 CPU 模块中，可以进行灵活的软元件指定。

(a) 字元件的各个位在触点、线圈操作中的使用

根据字元件的位指定，可以在触点、线圈操作中使用字元件的各个位。

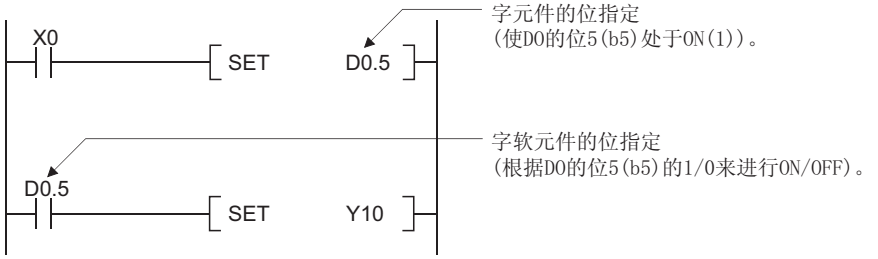


图 1.18 字元件的位指定

(b) 容易的 1 点单位的直接处理

通过直接存取输入 (DX□)，直接存取输出 (DY□)，在程序中可以容易地进行 1 点单位的直接处理。(☞ 3.8.2 项)

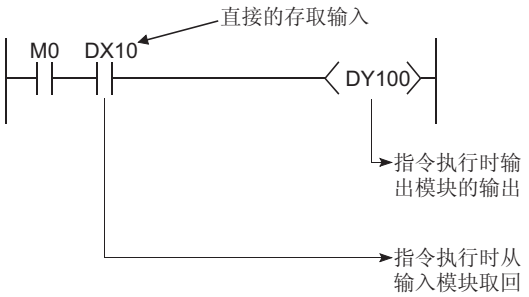


图 1.19 1 点单位的直接处理

(c) 通过使用微分触点，不需要进行输入的脉冲化处理

由于使用了微分触点（-|↑|-/-|↓|-）所以不需要进行输入的脉冲化处理。

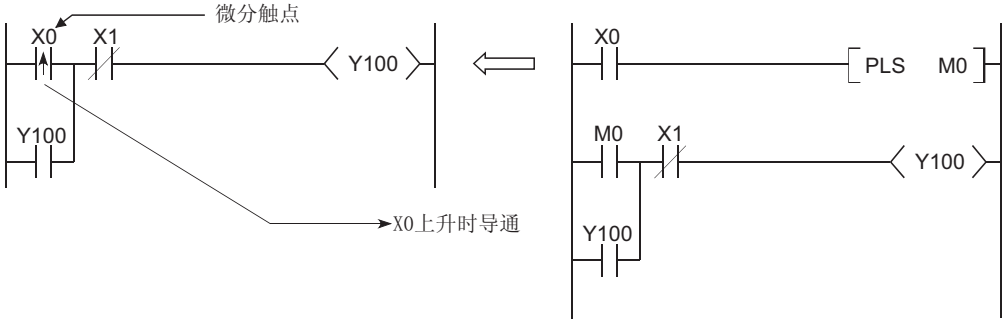


图 1.20 微分触点的使用

(d) 向智能功能模块的缓冲存储器进行直接存取

通过软元件的处理可以对智能功能模块的缓冲存储器进行编程。(☞ 9.5 节)

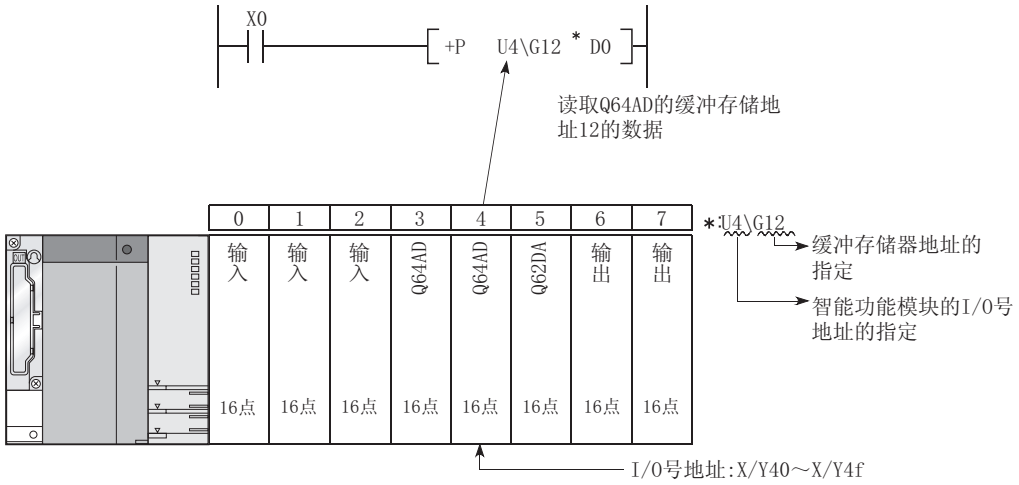


图 1.21 向智能功能模块的缓冲存储器进行的存取

高性能 注 1.7

(e) 向链接软元件进行的直接存取

即使不进行刷新设定也可以对 MELSECNET/G 网络模块 注 1.7 注 1.8 、MELSECNET/H 网络模块的链接软元件 (LX、LY、LB、LW、SB、SW) 进行直接的存取。(☞ 9.4 节)

基本 注 1.8 过程 注 1.8 冗余 注 1.8

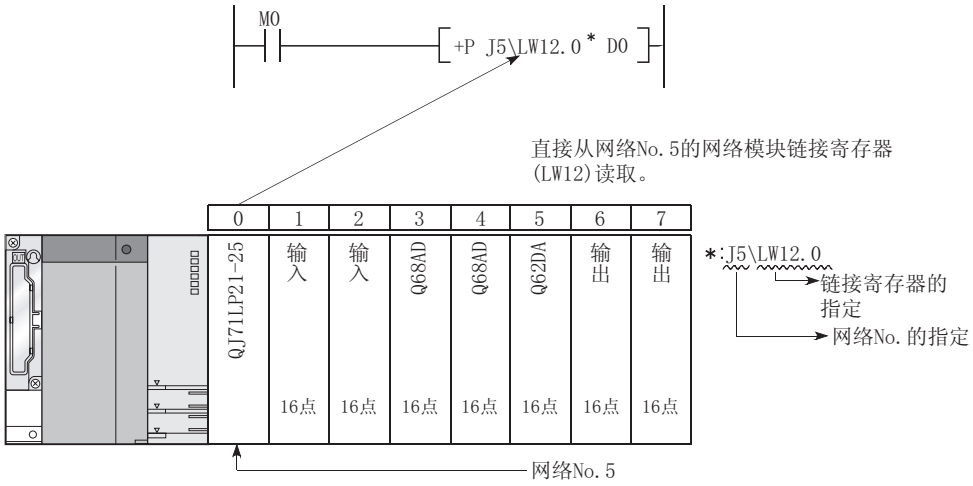


图 1.22 向链接软元件进行的直接存取

高性能 注 1.7

在高性能模式 QCPU 中使用 MELSECNET/G 时，请确认 CPU 模块以及 GX Developer 的版本。
(☞ 附录 4.2)

基本 注 1.8 过程 注 1.8 冗余 注 1.8

在基本模式 QCPU、过程 CPU、冗余 CPU 中，不能使用 MELSECNET/G。

(2) 对程序进行结构性的叙述

通过变址寄存器以及变址继电器，可以简单构造包括脉冲处理的程序（☞ 9.2.6 项）。

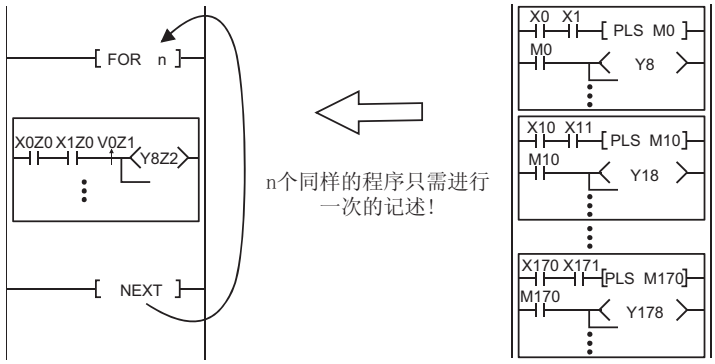


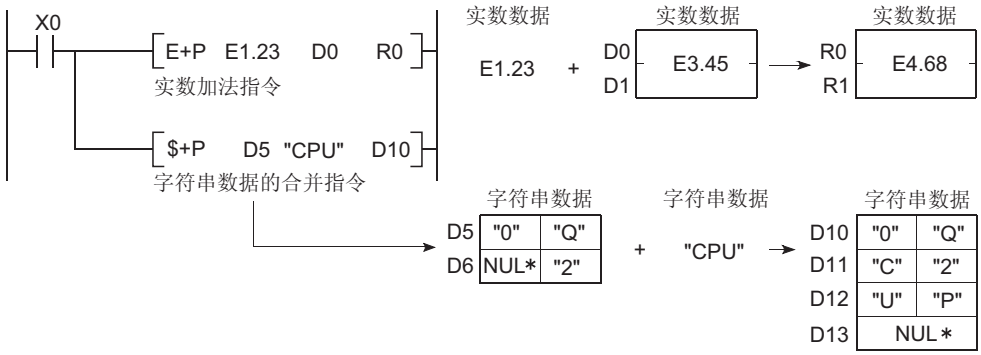
图 1.23 包括脉冲处理的程序的构造化

(3) 容易的数据处理

基本 注 1.9

(a) 实数以及字符串常数的直接使用

在编程中可以直接使用实数（浮动小数点数据）、字符串常数*注 1.9*



*:NUL 表示 “00H(字符串的最后)”。

图 1.24 实数以及字符串常数的使用

基本 注 1.9

在基本模式 QCPU 中使用实数（浮动小数点数据）时，请确认 CPU 模块的版本。
（☞ 附录 4.1(2)）
在基本模式 QCPU 中，只能使用 $MOV、STR、DSTR、VAL、DVAL、ESTR、EVAL 指令的字符串。

(b) 大容量数据的高速处理

根据制表处理指令等的数据处理指令的充实，可以进行大容量数据的高速处理。

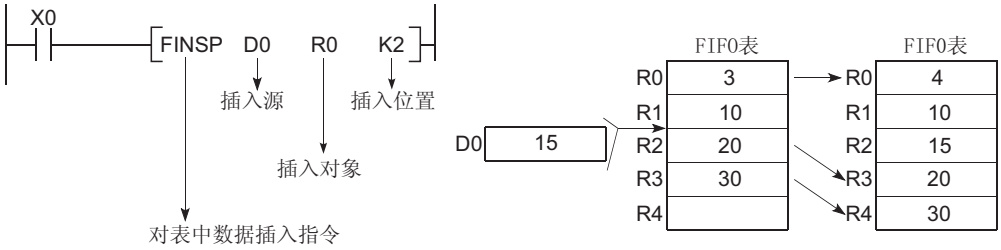


图 1.25 制表处理指令的数据处理

(4) 灵活的副程序的管理

(a) 副程序的共享化

由于副程序的共享，可以减少程序的步数。

另外，程序的制作与管理也变得容易。

基本

注 1.10

副程序可以在同一个程序内制作调用，根据使用的公共指针号在其它的程序内也可以进行程序的调用。*注 1.10*

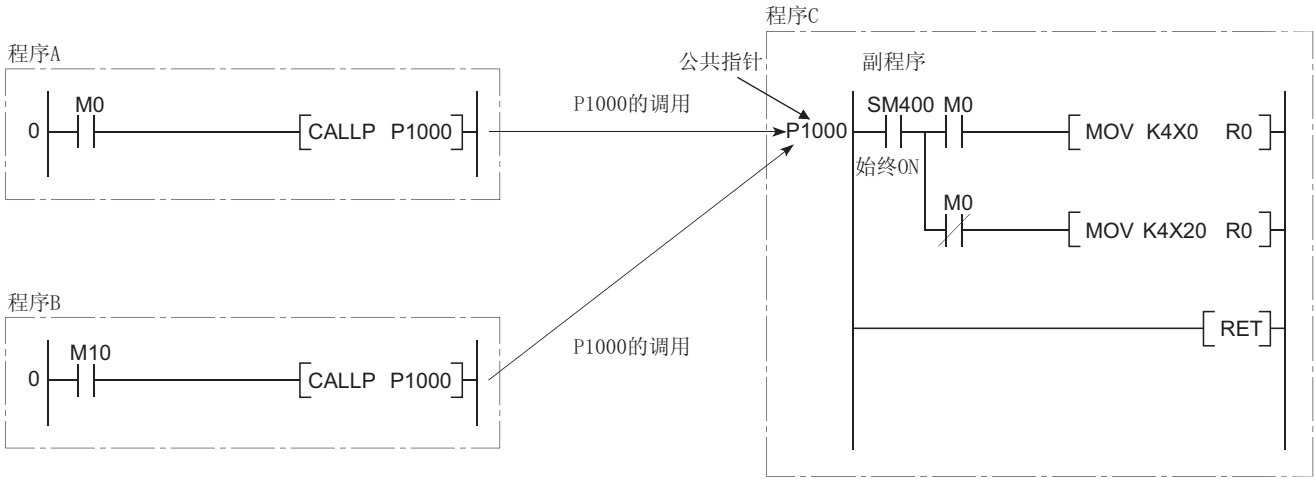


图 1.26 副程序的共享

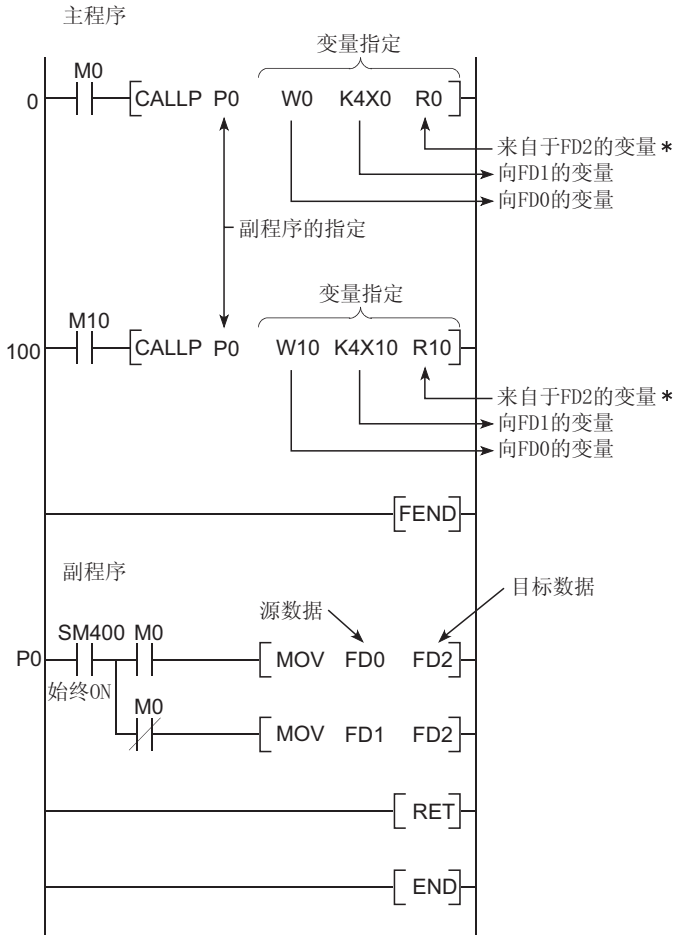
基本

注 1.10

基本模式 QCPU 中，由于不能执行多个程序，因此不能使用其它程序的副程序。

(b) 附带变量的副函数调用指令

根据附带变量的副函数调用指令，制作多次调用的副程序也将变得容易。



＊：关于变量的输入／输出的条件，请参照 9.3.1 项。

图 1.27 附带变量的副函数的调用

1.4 系列号与功能版本的确认方法

CPU 模块的系列号与功能版本可以通过额定值铭牌或者 GX Developer 的系统监视进行确认。

(1) 通过额定值铭牌进行确认

额定值铭牌在 CPU 模块的侧面。

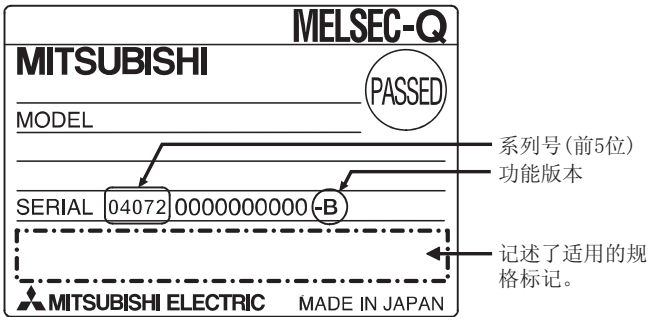


图 1.28 额定值铭牌

(2) 通过系统监视进行的确认（产品信息一览表）

系统监视的显示需要选择 GX Developer 的 [检测] → [系统监视]。

在系统监视中，也可以确认系列号与功能版本。

图 1.29 系统监视

(a) 生产编号的显示

1) 基本模式 QCPU、高性能模式 QCPU、过程 CPU、冗余 CPU 时
由于不兼容生产编号显示，因此显示为 “0000000000000000”。

2) 通用型 QCPU 时
生产编号栏中显示相应模块的序列号（生产编号）。

要 点

额定铭牌上记载的序列号与 GX Developer 的产品信息中显示的序列号有时会有所不同。

- 额定铭牌的序列号是产品的管理信息。
- GX Developer 的产品信息中显示的序列号表示产品的功能信息。
  产品的功能信息在新增功能时将被更新。

备忘录

第 2 章　性能规格

CPU 模块的性能规格如下所示。

表 2.1 性能规格

项目		基本模式 QCPU			备注
		Q00JCPU	Q00CPU	Q01CPU	
控制方式		存储程序循环运算			----
I/O 控制方式		刷新方式			通过指定直接访问 I/O（DX□、DY□）可直接访问 I/O。
程序语言	顺序控制语言	继电器符号语言、逻辑符号语言、MELSAP3(SFC)、MELSAP-L、功能块、结构化文本 (ST)			----
	过程控制语言	----			通过 PX Developer 进行编程。
处理速度（顺控指令）	LD X0	200ns	160ns	100ns	----
	MOV D0 D1	700ns	560ns	350ns	----
处理速度（冗余功能）	热备执行时间（扫描时间的延长时间）	----			☞ QnPRHCPU 用户手册（冗余系统篇）
恒定扫描（使扫描时间恒定的功能）		1 至 2000ms（可以以 1ms 为单位进行设置）			通过参数设置
程序容量 *1、*2		8k 步（32k 字节）		14k 步（56k 字节）	☞ 5.1.1 项、5.1.2 项
存储器容量 *1	程序内存（驱动器 0）	58k 字节	94k 字节		☞ 5.1.2 项、5.2.2 项
	存储卡 (RAM)（驱动器 1）	----			☞ 5.2.7 项
	存储卡 (ROM)（驱动器 2）	----			☞ 5.2.7 项
	标准 RAM（驱动器 3）	0	128k 字节 *3		☞ 5.1.4 项、5.2.6 项
	标准 ROM（驱动器 4）	58k 字节	94k 字节		☞ 5.1.3 项、5.2.5 项
	CPU 共享内存 *3 , *4	----	1k 字节		☞ QCPU 用户手册（多 CPU 系统篇）

*1　：根据 CPU 模块的不同，存储在存储区的文件大小单位会有不同。
☞ 5.4.4 项

*2　：可执行的最大顺控程序步数如下式。
（程序的容量）-（文件头大小（默认： 34 步））
关于程序的容量、文件的详细情况，请参照第 5 章。

*3　：容量可以通过 CPU 模块的功能升级得以扩大。（☞ 附录 4）

*4　：CPU 共享内存不能锁存。
如果进行可编程控制器的电源 ON 或者 CPU 模块的复位，CPU 共享内存将被清除。
☞ QCPU 用户手册（多 CPU 系统篇）

表 2.1 性能规格

项目			基本模式 QCPU			备注
			Q00JCPU	Q00CPU	Q01CPU	
最多存储文件个数	程序内存		6 *5			☞ 5.1.2 项、5.2.2 项
	存储卡 (RAM)		----			☞ 5.2.6 项
	存储卡 (ROM)	Flash 卡	----			☞ 5.2.6 项
		ATA 卡	----			☞ 5.2.6 项
	标准 RAM		----	1		☞ 5.1.4 项、5.2.5 项
	标准 ROM		6 *5			☞ 5.1.3 项、5.2.4 项
标准 ROM 的写入次数			最多 10 万次			----
I/O 软元件点数			2048 点 (X/Y0 至 7FF			程序可用点数
I/O 点数			256 点 (X/Y0 至 FF)	1024 点 (X/Y0 至 3FF)		物理 I/O 模块的可访问点数
软元件点数	内部继电器 [M]		默认 8192 点 (M0 至 8191) (可变更)			可在设置范围内变更使用点数 (☞ 9.2 节)
	锁存继电器 [L]		默认 2048 点 (L0 至 2047) (可变更)			
	链接继电器 [B]		默认 2048 点 (B0 至 7FF) (可变更)			
	定时器 [T]		默认 512 点 (T0 至 511) (低速定时器 / 高速定时器共用) (可变更) 低速定时器 / 高速定时器通过指令指定 低速定时器 / 高速定时器的计量单位在参数中设定 (低速定时器 : 1 至 1000ms ; 单位 1ms ; 默认 100ms) (高速定时器 : 0.1 至 100ms ; 单位 0.1ms ; 默认 10ms)			
	累计定时器 [ST]		默认 0 点 (低速定时器 / 高速定时器共用) (可变更) 低速累计定时器 / 高速累计定时器通过指令指定 低速累计定时器 / 高速累计定时器的计量单位在参数中设定 (低速累计定时器 : 1 至 1000ms ; 单位 1ms ; 默认 100ms) (高速累计定时器 : 0.1 至 100ms ; 单位 0.1ms ; 默认 10ms)			
	计数器 [C]		• 常规计数器 默认 512 点 (C0 至 511) (可变更) • 中断计数器 最多 128 点 (默认 0 点, 通过参数设定)			
	数据寄存器 [D]		默认 11136 点 (D0 至 11135) (可变更)			
	链接寄存器 [W]		默认 2048 点 (W0 至 7FF) (可变更)			
	报警器 [F]		默认 1024 点 (F0 至 1023) (可变更)			
	变址继电器 [V]		默认 1024 点 (V0 至 1023) (可变更)			
	文件寄存器	[R]	----	32768 点 (R0 至 32767) / 块		软元件点数固定
		[ZR]	----	65536 点 (ZR0 至 65535)		
	链接特殊继电器 [SB]		1024 点 (SB0 至 3FF)			
	链接特殊寄存器 [SW]		1024 点 (SW0 至 3FF)			
	步进继电器 [S] *6		2048 点 (S0 至 127/ 块)			
	变址寄存器 [Z]		10 点 (Z0 至 9)			
	指针 [P]		300 点 (P0 至 299)			

*5 : 参数、智能功能模块参数、顺控程序、SFC 程序、软元件注释、软元件初始值可以各存储为 1 个文件。

*6 : 步进继电器是用于 SFC 功能的软元件。

表 2.1 性能规格

项目		基本模式 QCPU			备注
		Q00JCPU	Q00CPU	Q01CPU	
软元件点数	中断指针 [I]	128 点 (I0 至 127) 可以通过参数设定系统中断指针 I28 到 31 的周期间隔。 (2 至 1000ms, 单位 1ms) 默认 I28: 100ms I29: 40ms I30: 20ms I31: 10ms			软元件点数固定
	特殊继电器 [SM]	1024 点 (SM0 至 1023)			
	特殊寄存器 [SD]	1024 点 (SD0 至 1023)			
	功能输入 [FX]	16 点 (FX0 至 F)			
	功能输出 [FY]	16 点 (FY0 至 F)			
	功能寄存器 [FD]	5 点 (FD0 至 4)			
软元件热备传送字数		----			☞ QnPRHCPU 用户手册 (冗余系统篇)
直接链接软元件		直接访问链接软元件的软元件。 MELSECNET/H 专用。 指定格式: J□□\X□□、J□□\Y□□、J□□\W□□、J□□\B□□、J□□\SW□□、J□□\SB□□			☞ 9.4 节
智能功能模块软元件		直接访问智能功能模块的缓冲存储器的软元件。 指定格式: U□□\G□□			☞ 9.5 节
锁存 (停电保持) 范围		L0 至 2047(默认) (可以为 B、F、V、T、ST、C、D、W 设置锁存范围。)			通过参数设置
RUN/PAUSE 触点		可以在 X0 到 7FF 中各设置一个 RUN/PAUSE 触点。			
定时器功能		年、月、日、时、分、秒、星期。 (自动识别闰年) 精度: -3.2 至 +5.27s(TYP.+1.98s)/d 在 0 ℃时 精度: -2.57 至 +5.27s(TYP.+2.22s)/d 在 25 ℃时 精度: -11.68 至 +3.65s(TYP.-2.64s)/d 在 55 ℃时			☞ 6.5 节
瞬间掉电允许时间		20ms 以内 (AC100V 以上)	根据电源模块		----
DC5V 内部消耗电流		0.26A *7	0.25A	0.27A	----
外形尺寸	H	98mm	98mm		----
	W	245mm *8	27.4mm		----
	D	98mm	89.3mm		----
重量		0.66kg *8	0.13kg		----

*7 : 是包括 CPU 模块、基板的值。

*8 : 是包括 CPU 模块、基板、电源模块的值。

备 注

关于一般规格，请参阅以下手册。

☞ QCPU 用户手册 (硬件设计 / 维护点检篇)

表 2.2 性能规格

项目		高性能模式 QCPU					备注
		Q02CPU	Q02HCPU	Q06HCPU	Q12HCPU	Q25HCPU	
控制方式		存储程序循环运算					----
I/O 控制方式		刷新方式					通过指定直接访问 I/O (DX□、DY□) 可直接访问 I/O。
程序语言	顺序控制语言	继电器符号语言、逻辑符号语言、MELSAP3(SFC)、MELSAP-L、功能块、结构化文本 (ST)					----
	过程控制语言	----					通过 PX Developer 进行编程。
处理速度 (顺控指令)	LD X0	79ns	34ns				----
	MOV D0 D1	237ns	102ns				----
处理速度 (冗余功能)	热备执行时间 (扫描时间的延长时间)	----					☞ QnPRHCPU 用户手册 (冗余系统篇)
恒定扫描 (使扫描时间恒定的功能)		0.5 至 2000ms (可以以 0.5ms 为单位进行设置)					通过参数设置
程序容量 *1、*2		28k 步 (112k 字节)		60k 步 (240k 字节)	124k 步 (496k 字节)	252k 步 (1008k 字节	☞ 5.1.1 项、5.1.2 项
存储器容量 *1	程序内存 (驱动器 0)	112k 字节		240k 字节	496k 字节	1008k 字节	☞ 5.1.1 项、5.1.2 项
	存储卡 (RAM) (驱动器 1)	安装的存储卡容量 (最大 2M 字节)					☞ 5.2.6 项
	存储卡 (ROM) (驱动器 2)	安装的存储卡容量 (Flash 卡： 最大 4M 字节； ATA 卡：最大 32M 字节)					☞ 5.2.6 项
	标准 RAM (驱动器 3)	64k 字节	128k 字节 *3		256k 字节 *3		☞ 5.1.4 项、5.2.5 项
	标准 ROM (驱动器 4)	112k 字节		240k 字节	496k 字节	1008k 字节	☞ 5.1.3 项、5.2.4 项
	CPU 共享内存 *3 、*4	8k 字节					☞ QCPU 用户手册 (多 CPU 系统篇)

*1 ：根据 CPU 模块的不同，存储在存储区的文件大小单位会有不同。
☞ 5.4.4 项

*2 ：可执行的最大顺控程序步数如下式。
(程序的容量)-(文件头大小 (默认： 34 步))
关于程序的容量、文件的详细情况，请参照第 5 章。

*3 ：容量可以通过 CPU 模块的功能升级得以扩大。 (☞ 附录 4)

*4 ：CPU 共享内存不能锁存。
如果进行可编程控制器的电源 ON 或者 CPU 模块的复位，CPU 共享内存将被清除。
☞ QCPU 用户手册 (多 CPU 系统篇)

表 2.2 性能规格

项目			高性能模式 QCPU					备注
			Q02CPU	Q02HCPU	Q06HCPU	Q12HCPU	Q25HCPU	
最多存储文件个数	程序内存		28		60	124	252 *6	☞ 5.1.2 项、5.2.2 项
	存储卡 (RAM)		287(当使用 Q2MEM-2MBS 时)					☞ 5.2.6 项
	存储卡 (ROM)	Flash 卡	288					☞ 5.2.6 项
		ATA 卡	512					☞ 5.2.6 项
	标准 RAM		3 *5					☞ 5.1.4 项、5.2.5 项
	标准 ROM		28		60	124	252	☞ 5.1.3 项、5.2.4 项
标准 ROM 的写入次数			最多 10 万次					----
I/O 软元件点数			8192 点 (X/Y0 至 1FFF)					程序可用点数
I/O 点数			4096 点 (X/Y0 至 FFF)					物理 I/O 模块的可访问点数
软元件点数	内部继电器 [M]		默认 8192 点 (M0 至 8191)(可变更)					可在设置范围内变更使用点数 (☞ 9.2 节)
	锁存继电器 [L]		默认 8192 点 (L0 至 8191)(可变更)					
	链接继电器 [B]		默认 8192 点 (B0 至 1FFF)(可变更)					
	定时器 [T]		默认 2048 点 (T0 至 2047) (低速定时器 / 高速定时器共用) (可变更) 低速定时器 / 高速定时器通过指令指定 低速定时器 / 高速定时器的计量单位在参数中设定 (低速定时器: 1 至 1000ms ; 单位 1ms ; 默认 100ms) (高速定时器: 0.1 至 100ms ; 单位 0.1ms ; 默认 10ms)					
	累计定时器 [ST]		默认 0 点 (低速定时器 / 高速定时器共用) (可变更) 低速累计定时器 / 高速累计定时器通过指令指定 低速累计定时器 / 高速累计定时器的计量单位在参数中设定 (低速累计定时器: 1 至 1000ms ; 单位 1ms ; 默认 100ms) (高速累计定时器: 0.1 至 100ms ; 单位 0.1ms ; 默认 10ms)					
	计数器 [C]		• 常规计数器 默认 1024 点 (C0 至 1023)(可变更) • 中断计数器 最多 256 点 (默认 0 点, 通过参数设定)					
	数据寄存器 [D]		默认 12288 点 (D0 至 12287)(可变更)					
	链接寄存器 [W]		默认 8192 点 (W0 至 1FFF)(可变更)					
	报警器 [F]		默认 2048 点 (F0 至 2047)(可变更)					
	变址继电器 [V]		默认 2048 点 (V0 至 2047)(可变更)					

*5 : 可以通过 CPU 模块的功能升级扩展。(☞ 附录 4)

*6 : CPU 模块中可执行的程序数最多为 124 个。125 及以上的程序不能执行。

表 2.2 性能规格

项目				高性能模式 QCPU					备注
				Q02CPU	Q02HCPU	Q06HCPU	Q12HCPU	Q25HCPU	
软元件点数	文件寄存器	[R]	标准 RAM	32768 点 (R0 至 32767)	通过 32768 点 (R0 至 32767) 单位进行块切换时，最多可以使用 65536 点。		通过 32768 点 (R0 至 32767) 单位进行块切换时，最多可以使用 131072 点。		• 只有使用 Flash 卡时才可以读取。 不能使用 ATA 卡。
			SRAM 卡 (1M 字节)	通过 32768 点 (R0 至 32767) 单位进行块切换时，最多可以使用 517120 点。					
			SRAM 卡 (2M 字节)	通过 32768 点 (R0 至 32767) 单位进行块切换时，最多可以使用 1041408 点。					
			Flash 卡 (2M 字节)	通过 32768 点 (R0 至 32767) 单位进行块切换时，最多可以使用 1041408 点。					
			Flash 卡 (4M 字节)	通过 32768 点 (R0 至 32767) 单位进行块切换时，最多可以使用 1042432 点。					
		[ZR]	标准 RAM	32768 点 (R0 至 32767)	65536 点 (ZR0 至 65535)，无需进行块切换。		131072 点 (ZR0 至 131071)，无需进行块切换。		
			SRAM 卡 (1M 字节)	517120 点 (ZR0 至 517119)， 无需进行块切换。					
			SRAM 卡 (2M 字节)	1041408 点 (ZR0 至 1041407)，无需进行块切换。					
			Flash 卡 (2M 字节)	1041408 点 (ZR0 至 1041407)，无需进行块切换。					
			Flash 卡 (4M 字节)	1042432 点 (ZR0 至 1042431)，无需进行块切换。					
	链接特殊继电器 [SB]			2048 点 (SB0 至 7FF)					软元件点数固定
	链接特殊寄存器 [SW]			2048 点 (SW0 至 7FF)					
	步进继电器 [S] *7			8192 点 (S0 至 8191)					
	变址寄存器 [Z]			16 点 (Z0 至 15)					
	指针 [P]			4096 点 (P0 至 4095)，可以通过参数设定局部指针 / 公共指针的使用范围。					
	中断指针 [I]			256 点 (I0 至 255) 可以通过参数设定系统中断指针 I28 到 31 的恒定周期间隔。 (0.5 至 1000ms，单位 0.5ms) 默认 I28：100ms I29：40ms I30：20ms I31：10ms					

*7 ：步进继电器是用于 SFC 功能的软元件。

表 2.2 性能规格

项目		高性能模式 QCPU					备注
		Q02CPU	Q02HCPU	Q06HCPU	Q12HCPU	Q25HCPU	
软元件点数	特殊继电器 [SM]	2048 点 (SM0 至 2047)					软元件点数固定
	特殊寄存器 [SD]	2048 点 (SM0 至 2047)					
	功能输入 [FX]	16 点 (FX0 至 F)					
	功能输出 [FY]	16 点 (FY0 至 F)					
	功能寄存器 [FD]	5 点 (FD0 至 4)					
软元件热备传送字数		----					☞ QnPRHCPU 用户手册（冗余系统篇）
直接链接软元件		直接访问链接软元件的软元件。 MELSECNET/G*8 、MELSECNET/H 专用。 指定格式： J□□\X□□, J□□\Y□□, J□□\W□□, J□□\B□□, J□□\SW□□, J□□\SB□□					☞ 9.4 节
智能功能模块软元件		直接访问智能功能模块的缓冲存储器的软元件。 指定格式： U□□\G□□					☞ 9.5 节
锁存（停电保持）范围		L0 至 8191（默认） （可以为 B、F、V、T、ST、C、D、W 设置锁存范围。）					通过参数设置
RUN/PAUSE 触点		可以在 X0 到 1FFF 中各设置一个 RUN/PAUSE 触点。					
时钟功能		年、月、日、时、分、秒、星期。（自动识别闰年） 精度： -3.18 至 +5.25s(TYP.+2.12s)/d 在 0 ℃时 精度： -3.93 至 +5.25s(TYP.+1.90s)/d 在 25 ℃时 精度： -14.69 至 +3.53s(TYP.-3.67s)/d 在 55 ℃时					☞ 6.5 节
瞬间掉电允许时间		根据电源模块					----
DC5V 内部消耗电流		0.60A	0.64A				----
外形尺寸	H	98mm					----
	W	27.4mm					----
	D	89.3mm					----
重量		0.20kg					----

*8 ：只能使用序列号的前 5 位数为 “09012” 及以后的高性能模式 QCPU。

备 注

关于一般规格，请参阅以下手册。

☞ QCPU 用户手册（硬件设计 / 维护点检篇）

表 2.3 性能规格

项目		过程 CPU		备注
		Q12PHCPU	Q25PHCPU	
控制方式		存储程序循环运算		----
I/O 控制方式		刷新方式		通过指定直接访问 I/O (DX□、DY□) 可直接访问 I/O。
程序语言	顺序控制语言	继电器符号语言、逻辑符号语言、MELSAP3(SFC)、MELSAP-L、功能块、结构化文本 (ST)		----
	过程控制语言	过程控制用 FBD		通过 PX Developer 进行编程。
处理速度（顺控指令）	LD X0	34ns		----
	MOV D0 D1	102ns		----
处理速度（冗余功能）	热备执行时间（扫描时间的延长时间）	----		☞ QnPRHCPU 用户手册（冗余系统篇）
恒定扫描（使扫描时间恒定的功能）		0.5 至 2000ms（可以以 0.5ms 为单位进行设置）		通过参数设置
程序容量 *1、*2		124k 步 (496k 字节)	252k 步 (1008k 字节)	☞ 5.1.1 项、5.1.2 项
存储器容量 *1	程序内存（驱动器 0）	496k 字节	1008k 字节	☞ 5.1.2 项、5.2.2 项
	存储卡 (RAM)（驱动器 1）	安装的存储卡容量（最大 2M 字节）		☞ 5.2.6 项
	存储卡 (ROM)（驱动器 2）	安装的存储卡容量 (Flash 卡： 最大 4M 字节； ATA 卡： 最大 32M 字节)		☞ 5.2.6 项
	标准 RAM（驱动器 3）	256k 字节 *3		☞ 5.1.4 项、5.2.5 项
	标准 ROM（驱动器 4）	496k 字节	1008k 字节	☞ 5.1.3 项、5.2.4 项
	CPU 共享内存 *3 、*4	8k 字节		☞ QCPU 用户手册（多 CPU 系统篇）

*1 ：根据 CPU 模块的不同，存储在存储区的文件大小单位会有不同。
☞ 5.4.4 项

*2 ：可执行的最大顺控程序步数如下式。
（程序的容量）-（文件头大小（默认： 34 步））
关于程序的容量、文件的详细情况，请参照第 5 章。

*3 ：容量可以通过 CPU 模块的功能升级得以扩大。（☞ 附录 4）

*4 ：CPU 共享内存不能锁存。
如果进行可编程控制器的电源 ON 或者 CPU 模块的复位，CPU 共享内存将被清除。
☞ QCPU 用户手册（多 CPU 系统篇）

表 2.3 性能规格

项目			过程 CPU		备注
			Q12PHCPU	Q25PHCPU	
最多存储文件个数	程序内存		124	252 *6	☞ 5.1.2 项、5.2.2 项
	存储卡 (RAM)		287(当使用 Q2MEM-2MBS 时)		☞ 5.2.6 项
	存储卡 (ROM)	Flash 卡	288		☞ 5.2.6 项
		ATA 卡	512		☞ 5.2.6 项
	标准 RAM		3 *5		☞ 5.1.4 项、5.2.5 项
	标准 ROM		124	252	☞ 5.1.3 项、5.2.4 项
标准 ROM 的写入次数			最多 10 万次		----
I/O 软元件点数			8192 点 (X/Y0 至 1FFF)		程序可用点数
I/O 点数			4096 点 (X/Y0 至 FFF)		物理 I/O 模块的可访问点数
软元件点数	软元件点数		默认 8192 点 (M0 至 8191)(可变更)		可在设置范围内变更使用点数 (☞ 9.2 节)
	锁存继电器 [L]		默认 8192 点 (L0 至 8191)(可变更)		
	链接继电器 [B]		默认 8192 点 (B0 至 1FFF)(可变更)		
	定时器 [T]		默认 2048 点 (T0 至 2047) (低速定时器 / 高速定时器共用) (可变更) 低速定时器 / 高速定时器通过指令指定 低速定时器 / 高速定时器的计量单位在参数中设定 (低速定时器: 1 至 1000ms ; 单位 1ms ; 默认 100ms) (高速定时器: 0.1 至 100ms ; 单位 0.1ms ; 默认 10ms)		
	累计定时器 [ST]		默认 0 点 (低速定时器 / 高速定时器共用) (可变更) 低速累计定时器 / 高速累计定时器通过指令指定 低速累计定时器 / 高速累计定时器的计量单位在参数中设定 (低速累计定时器: 1 至 1000ms ; 单位 1ms ; 默认 100ms) (高速累计定时器: 0.1 至 100ms ; 单位 0.1ms ; 默认 10ms)		
	计数器 [C]		• 常规计数器 默认 1024 点 (C0 至 1023)(可变更) • 中断计数器 最多 256 点 (默认 0 点, 通过参数设定)		
	数据寄存器 [D]		默认 12288 点 (D0 至 12287)(可变更)		
	链接寄存器 [W]		默认 8192 点 (W0 至 1FFF)(可变更)		
	报警器 [F]		默认 2048 点 (F0 至 2047)(可变更)		
	变址继电器 [V]		默认 2048 点 (V0 至 2047)(可变更)		

*5 : 可以通过 CPU 模块的功能升级扩展。(☞ 附录 4)

*6 : CPU 模块中可执行的程序数最多为 124 个。125 及以上的程序不能执行。

表 2.3 性能规格

项目				过程 CPU		备注
				Q12PHCPU	Q25PHCPU	
软元件点数	文件寄存器	[R]	标准 RAM	通过 32768 点 (R0 至 32767) 单位进行块切换时，最多可以使用 131072 点。		• 只有使用 Flash 卡时才可以读取。 不能使用 ATA 卡。
			SRAM 卡 (1M 字节)	通过 32768 点 (R0 至 32767) 单位进行块切换时，最多可以使用 517120 点。		
			SRAM 卡 (2M 字节)	通过 32768 点 (R0 至 32767) 单位进行块切换时，最多可以使用 1041408 点。		
			Flash 卡 (2M 字节)	通过 32768 点 (R0 至 32767) 单位进行块切换时，最多可以使用 1041408 点。		
			Flash 卡 (4M 字节)	通过 32768 点 (R0 至 32767) 单位进行块切换时，最多可以使用 1042432 点。		
		[ZR]	标准 RAM	131072 点 (ZR0 至 131071)，无需进行块切换。		
			SRAM 卡 (1M 字节)	517120 点 (ZR0 至 517119)， 无需进行块切换。		
			SRAM 卡 (2M 字节)	1041408 点 (ZR0 至 1041407)，无需进行块切换。		
			Flash 卡 (2M 字节)	1041408 点 (ZR0 至 1041407)，无需进行块切换。		
			Flash 卡 (4M 字节)	1042432 点 (ZR0 至 1042431)，无需进行块切换。		
	链接特殊继电器 [SB]			2048 点 (SB0 至 7FF)		软元件点数固定
	链接特殊寄存器 [SW]			2048 点 (SW0 至 7FF)		
	步进继电器 [S] *7			8192 点 (S0 至 8191)		
	变址寄存器 [Z]			16 点 (Z0 至 15)		
	指针 [P]			4096 点 (P0 至 4095)，可以通过参数设定局部指针 / 公共指针的使用范围。		
	中断指针 [I]			256 点 (I0 至 255) 可以通过参数设定系统中断指针 I28 到 31 的恒定周期间隔。 (0.5 至 1000ms，单位 0.5ms) 默认 I28: 100ms I29: 40ms I30: 20ms I31: 10ms		

*7 : 步进继电器是用于 SFC 功能的软元件。

表 2.3 性能规格

项目		过程 CPU		备注
		Q12PHCPU	Q25PHCPU	
软元件点数	特殊继电器 [SM]	2048 点 (SM0 至 2047)		软元件点数固定
	特殊寄存器 [SD]	2048 点 (SD0 至 2047)		
	功能输入 [FX]	16 点 (FX0 至 F)		
	功能输出 [FY]	16 点 (FY0 至 F)		
	功能寄存器 [FD]	5 点 (FD0 至 4)		
软元件热备传送字数		----		☞ QnPRHCPU 用户手册（冗余系统篇）
直接链接软元件		直接访问链接软元件的软元件。 MELSECNET/H 专用。 指定格式： J□□\X□□、J□□\Y□□、J□□\W□□、J□□\B□□、J□□\SW□□、J□□\SB□□		☞ 9.4 节
智能功能模块软元件		直接访问智能功能模块的缓冲存储器的软元件。 指定格式： U□□\G□□		☞ 9.5 节
锁存（停电保持）范围		L0 至 8191（默认） （可以为 B、F、V、T、ST、C、D、W 设置锁存范围。）		通过参数设置
RUN/PAUSE 触点		可以在 X0 到 1FFF 中各设置一个 RUN/PAUSE 触点。		
时钟功能		年、月、日、时、分、秒、星期。（自动识别闰年） 精度： -3.18 至 +5.25s(TYP.+2.12s)/d 在 0 ℃时 精度： -3.93 至 +5.25s(TYP.+1.90s)/d 在 25 ℃时 精度： -14.69 至 +3.53s(TYP.-3.67s)/d 在 55 ℃时		☞ 6.5 节
瞬间掉电允许时间		根据电源模块		----
DC5V 内部消耗电流		0.64A		----
外形尺寸	H	98mm		----
	W	27.4mm		----
	D	89.3mm		----
重量		0.20kg		----

备 注

关于一般规格，请参阅以下手册。

☞ QCPU 用户手册（硬件设计 / 维护点检篇）

表 2.4 性能规格

项目		冗余 CPU		备注
		Q12PRHCPU	Q25PRHCPU	
控制方式		存储程序循环运算		----
I/O 控制方式		刷新方式		通过指定直接访问 I/O (DX□、DY□) 可直接访问 I/O。
程序语言	顺序控制语言	继电器符号语言、逻辑符号语言、MELSAP3(SFC)、MELSAP-L、功能块、结构化文本 (ST)		----
	过程控制语言	过程控制用 FBD		通过 PX Developer 进行编程。
处理速度 (顺控指令)	LD X0	34ns		----
	MOV D0 D1	102ns		----
处理速度 (冗余功能)	热备执行时间 (扫描时间的延长时间)	软元件内存 48k 字点： 10ms 软元件内存 100k 字点： 15ms		☞ QnPRHCPU 用户手册 (冗余系统篇)
恒定扫描 (使扫描时间恒定的功能)		0.5 至 2000ms(可以以 0.5ms 为单位进行设置)		通过参数设置
程序容量 *1、*2		124k 步 (496k 字节)	252k 步 (1008k 字节)	☞ 5.1.1 项、5.1.2 项
存储器容量 *1	程序内存 (驱动器 0)	496k 字节	1008k 字节	☞ 5.1.2 项、5.2.2 项
	存储卡 (RAM) (驱动器 1)	安装的存储卡容量 (最大 2M 字节)		☞ 5.2.6 项
	存储卡 (ROM) (驱动器 2)	安装的存储卡容量 (Flash 卡： 最大 4M 字节； ATA 卡： 最大 32M 字节)		☞ 5.2.6 项
	标准 RAM (驱动器 3)	256k 字节 *3		☞ 5.1.4 项、5.2.5 项
	标准 ROM (驱动器 4)	496k 字节	1008k 字节	☞ 5.1.3 项、5.2.4 项
	CPU 共享内存	----		☞ QCPU 用户手册 (多 CPU 系统篇)

*1 ：根据 CPU 模块的不同，存储在存储区的文件大小单位会有不同。
☞ 5.4.4 项

*2 ：可执行的最大顺控程序步数如下式。
(程序的容量)-(文件头大小 (默认： 34 步))
关于程序的容量、文件的详细情况，请参照第 5 章。

*3 ：容量可以通过 CPU 模块的功能升级得以扩大。(☞ 附录 4)

表 2.4 性能规格

项目			冗余 CPU		备注
			Q12PRHCPU	Q25PRHCPU	
最多存储文件个数	程序内存		124	252 *5	☞ 5.1.2 项、5.2.2 项
	存储卡 (RAM)		287 (当使用 Q2MEM-2MBS 时)		☞ 5.2.6 项
	存储卡 (ROM)	Flash 卡	288		☞ 5.2.6 项
		ATA 卡	512		☞ 5.2.6 项
	标准 RAM		3 *4		☞ 5.1.4 项、5.2.5 项
	标准 ROM		124	252	☞ 5.1.3 项、5.2.4 项
标准 ROM 的写入次数			最多 10 万次		----
I/O 软元件点数			8192 点 (X/Y0 至 1FFF)		程序可用点数
I/O 点数			4096 点 (X/Y0 至 FFF)		物理 I/O 模块的可访问点数
软元件点数	内部继电器 [M]		默认 8192 点 (M0 至 8191) (可变更)		可在设置范围内变更使用点数 (☞ 9.2 节)
	锁存继电器 [L]		默认 8192 点 (L0 至 8191) (可变更)		
	链接继电器 [B]		默认 8192 点 (B0 至 1FFF) (可变更)		
	定时器 [T]		默认 2048 点 (T0 至 2047) (低速定时器 / 高速定时器共用) (可变更) 低速定时器 / 高速定时器通过指令指定 低速定时器 / 高速定时器的计量单位在参数中设定 (低速定时器 : 1 至 1000ms ; 单位 1ms ; 默认 100ms) (高速定时器 : 0.1 至 100ms ; 单位 0.1ms ; 默认 10ms)		
	累计定时器 [ST]		默认 0 点 (低速定时器 / 高速定时器共用) (可变更) 低速累计定时器 / 高速累计定时器通过指令指定 低速累计定时器 / 高速累计定时器的计量单位在参数中设定 (低速累计定时器 : 1 至 1000ms ; 单位 1ms ; 默认 100ms) (高速累计定时器 : 0.1 至 100ms ; 单位 0.1ms ; 默认 10ms)		
	计数器 [C]		• 常规计数器 默认 1024 点 (C0 至 1023) (可变更) • 中断计数器 最多 256 点 (默认 0 点，通过参数设定)		
	数据寄存器 [D]		默认 12288 点 (D0 至 12287) (可变更)		
	链接寄存器 [W]		默认 8192 点 (W0 至 1FFF) (可变更)		
	报警器 [F]		默认 2048 点 (F0 至 2047) (可变更)		
	变址继电器 [V]		默认 2048 点 (V0 至 2047) (可变更)		

*4 : 可以通过 CPU 模块的功能升级扩展。(☞ 附录 4)

*5 : CPU 模块中可执行的程序数最多为 124 个。125 及以上的程序不能执行。

表 2.4 性能规格

项目				冗余 CPU		备注
				Q12PRHCPU	Q25PRHCPU	
软元件点数	文件寄存器	[R]	标准 RAM	通过 32768 点 (R0 至 32767) 单位进行块切换时，最多可以使用 131072 点。		• 只有使用 Flash 卡时才可以读取。 不能使用 ATA 卡。
			SRAM 卡 (1M 字节)	通过 32768 点 (R0 至 32767) 单位进行块切换时，最多可以使用 517120 点。		
			SRAM 卡 (2M 字节)	通过 32768 点 (R0 至 32767) 单位进行块切换时，最多可以使用 1041408 点。		
			Flash 卡 (2M 字节)	通过 32768 点 (R0 至 32767) 单位进行块切换时，最多可以使用 1041408 点。		
			Flash 卡 (4M 字节)	通过 32768 点 (R0 至 32767) 单位进行块切换时，最多可以使用 1042432 点。		
		[ZR]	标准 RAM	131072 点 (ZR0 至 131071)，无需进行块切换。		
			SRAM 卡 (1M 字节)	517120 点 (ZR0 至 517119)， 无需进行块切换。		
			SRAM 卡 (2M 字节)	1041408 点 (ZR0 至 1041407)，无需进行块切换。		
			Flash 卡 (2M 字节)	1041408 点 (ZR0 至 1041407)，无需进行块切换。		
			Flash 卡 (4M 字节)	1042432 点 (ZR0 至 1042431)，无需进行块切换。		
	链接特殊继电器 [SB]			2048 点 (SB0 至 7FF)		软元件点数固定
	链接特殊寄存器 [SW]			2048 点 (SW0 至 7FF)		
	步进继电器 [S] *6			8192 点 (S0 至 8191)		
	变址寄存器 [Z]			16 点 (Z0 至 15)		
	指针 [P]			4096 点 (P0 至 4095)，可以通过参数设定局部指针 / 公共指针的使用范围。		
	中断指针 [I]			256 点 (I0 至 255) 可以通过参数设定系统中断指针 I28 到 31 的恒定周期间隔。 (0.5 至 1000ms，单位 0.5ms) 默认 I28：100ms I29：40ms I30：20ms I31：10ms		

*6 ：步进继电器是用于 SFC 功能的软元件。

表 2.4 性能规格

项目		冗余 CPU		备注
		Q12PRHCPU	Q25PRHCPU	
软元件点数	特殊继电器 [SM]	2048 点 (SM0 至 2047)		软元件点数固定
	特殊寄存器 [SD]	2048 点 (SD0 至 2047)		
	功能输入 [FX]	16 点 (FX0 至 F)		
	功能输出 [FY]	16 点 (FY0 至 F)		
	功能寄存器 [FD]	5 点 (FD0 至 4)		
软元件热备传送字数		最多 100k 字		☞ QnPRHCPU 用户手册（冗余系统篇）
直接链接软元件		直接访问链接软元件的软元件。 MELSECNET/H 专用。 指定格式： J□□\X□□、J□□\Y□□、J□□\W□□、J□□\B□□、J□□\SW□□、J□□\SB□□		☞ 9.4 节
智能功能模块软元件		直接访问智能功能模块的缓冲存储器的软元件。 指定格式： U□□\G□□		☞ 9.5 节
锁存（停电保持）范围		L0 至 8191（默认） （可以为 B、F、V、T、ST、C、D、W 设置锁存范围。）		通过参数设置
RUN/PAUSE 触点		可以在 X0 到 1FFF 中各设置一个 RUN/PAUSE 触点。		
时钟功能		年、月、日、时、分、秒、星期。（自动识别闰年） 精度： -3.2 至 +5.27s(TYP.+2.07s)/d 在 0 ℃时 精度： -2.77 至 +5.27s(TYP.+2.22s)/d 在 25 ℃时 精度： -12.14 至 +3.65s(TYP.-2.89s)/d 在 55 ℃时		☞ 6.5 节
瞬间掉电允许时间		根据电源模块		----
DC5V 内部消耗电流		0.89A		----
外形尺寸	H	98mm		----
	W	55.2mm		----
	D	89.3mm		----
重量		0.30kg		----

备 注

关于一般规格，请参阅以下手册。

☞ QCPU 用户手册（硬件设计 / 维护点检篇）

表 2.5 性能规格

项目			通用型 QCPU				备注
			Q02UCPU	Q03UDCPU	Q04UDHCPU	Q06UDHCPU	
控制方式			存储程序循环运算				----
I/O 控制方式			刷新方式				通过指定直接访问 I/O (DX□、DY□) 可直接访问 I/O。
程序语言	顺序控制语言		继电器符号语言、逻辑符号语言、MELSAP3(SFC)、MELSAP-L、功能块、结构化文本 (ST)				----
	过程控制语言		----				----
处理速度 (顺控指令)	LD X0		40ns	20ns	9.5ns		----
	MOV D0 D1		80ns	40ns	19ns		----
处理速度 (冗余功能)	热备执行时间 (扫描时间的延长时间)		----				----
恒定扫描 (使扫描时间恒定的功能)			0.5 至 2000ms (可以以 0.5ms 为单位进行设置)				通过参数设置
程序容量 *1、*2			20k 步 (80k 字节)	30k 步 (120k 字节)	40k 步 (160k 字节)	60k 步 (240k 字节)	----
存储器容量 *1	程序内存 (驱动器 0)		80k 字节	120k 字节	160k 字节	240k 字节	----
	存储卡 (RAM) (驱动器 1)		安装的存储卡容量 (最大 8M 字节)				QCPU 用户手册 (硬件设计 / 维护点检篇)
	存储卡 (ROM) (驱动器 2)		安装的存储卡容量 (Flash 卡 : 最大 4M 字节; ATA 卡 : 最大 32M 字节)				QCPU 用户手册 (硬件设计 / 维护点检篇)
	标准 RAM (驱动器 3)		128k 字节	192k 字节	256k 字节	768k 字节	----
	标准 ROM (驱动器 4)		512k 字节	1024k 字节			----
	CPU 共享内存 *3	QCPU 标准区	8k 字节	8k 字节			QCPU 用户手册 (多 CPU 系统篇)
		多 CPU 间高速通信区	----	32k 字节			

*1 : 根据 CPU 模块的不同，存储在存储区的文件大小单位会有不同。
5.4.4 项

*2 : 可执行的最大顺控程序步数如下式。
(程序的容量)-(文件头大小 (默认 : 34 步))
关于程序的容量、文件的详细情况，请参照第 5 章。

*3 : CPU 共享内存不能锁存。
如果进行可编程控制器的电源 ON 或者 CPU 模块的复位，CPU 共享内存将被清除。
QCPU 用户手册 (多 CPU 系统篇)

表 2.5 性能规格

项目			通用型 QCPU				备注
			Q02UCPU	Q03UDCPU	Q04UDHCPU	Q06UDHCPU	
最多存储文件个数	程序内存		64	124			----
	存储卡 (RAM)		319(当使用 Q3MEM-8MBS 时)				----
	存储卡 (ROM)	Flash 卡	288				----
		ATA 卡	512				----
	标准 RAM		3				只能文件寄存器、局部软元件、采样追踪文件个 1 个。
	标准 ROM		128	256			----
程序内存的写入次数			最多 10 万次 *4				----
标准 ROM 的写入次数			最多 10 万次 *5				----
I/O 软元件点数			8192 点 (X/Y0 至 1FFF)				程序可用点数
I/O 点数			2048 点 (X/Y0 至 7FF)	4096 点 (X/Y0 至 FFF)			物理 I/O 模块的可访问点数
软元件点数	内部继电器 [M]		默认 8192 点 (M0 至 8191)(可变更)				可在设置范围内变更使用点数。 (9.2 节)
	锁存继电器 [L]		默认 8192 点 (L0 至 8191)(可变更)				
	链接继电器 [B]		默认 8192 点 (B0 至 1FFF)(可变更)				
	定时器 [T]		默认 2048 点 (T0 至 2047)(低速定时器 / 高速定时器共用)(可变更) 低速定时器 / 高速定时器通过指令指定 低速定时器 / 高速定时器的计量单位在参数中设定 (低速定时器: 1 至 1000ms ; 单位 1ms ; 默认 100ms) (高速定时器: 0.1 至 100ms ; 单位 0.1ms ; 默认 10ms)				
	累计定时器 [ST]		默认 0 点 (低速定时器 / 高速定时器共用)(可变更) 低速累计定时器 / 高速累计定时器通过指令指定 低速累计定时器 / 高速累计定时器的计量单位在参数中设定 (低速累计定时器: 1 至 1000ms ; 单位 1ms ; 默认 100ms) (高速累计定时器: 0.1 至 100ms ; 单位 0.1ms ; 默认 10ms)				
	计数器 [C]		•常规计数器 默认 1024 点 (C0 至 1023)(可变更)				
	数据寄存器 [D]		默认 12288 点 (D0 至 12287)(可变更)				
	链接寄存器 [W]		默认 8192 点 (W0 至 1FFF)(可变更)				
	报警器 [F]		默认 2048 点 (F0 至 2047)(可变更)				
	变址继电器 [V]		默认 2048 点 (V0 至 2047)(可变更)				
	链接特殊继电器 [SB]		默认 2048 点 (SB0 至 7FF)(可变更)				
	链接特殊寄存器 [SW]		默认 2048 点 (SW0 至 7FF)(可变更)				

*4 : 有时会发生向程序内存的写入操作未被计数的情况。
程序内存的写入计数可以通过特殊寄存器 (SD682 至 SD683) 确认。

*5 : 有时会发生向标准 ROM 的写入操作未被计数的情况。
标准 ROM 的写入计数可以通过特殊寄存器 (SD687 至 SD688) 确认。

表 2.5 性能规格

项目				通用型 QCPU				备注
				Q02UCPU	Q03UDCPU	Q04UDHCPU	Q06UDHCPU	
软元件点数	文件寄存器	[R]	标准 RAM	通过 32768 点(R0 至 32767)单位进行块切换时，最多可以使用 65536 点。	通过 32768 点(R0 至 32767)单位进行块切换时，最多可以使用 98304 点。	通过 32768 点(R0 至 32767)单位进行块切换时，最多可以使用 131072 点。	通过 32768 点(R0 至 32767)单位进行块切换时，最多可以使用 393216 点。	• 只有使用 Flash 卡时才可以读取。 不能使用 ATA 卡。
			SRAM 卡(1M 字节)	通过 32768 点(R0 至 32767)单位进行块切换时，最多可以使用 517120 点。				
			SRAM 卡(2M 字节)	通过 32768 点(R0 至 32767)单位进行块切换时，最多可以使用 1041408 点。				
			SRAM 卡(4M 字节)	通过 32768 点(R0 至 32767)单位进行块切换时，最多可以使用 2087936 点。				
			SRAM 卡(8M 字节)	通过 32768 点(R0 至 32767)单位进行块切换时，最多可以使用 4184064 点。				
			Flash 卡(2M 字节)	通过 32768 点(R0 至 32767)单位进行块切换时，最多可以使用 1041408 点。				
			Flash 卡(4M 字节)	通过 32768 点(R0 至 32767)单位进行块切换时，最多可以使用 2087936 点。				
		[ZR]	标准 RAM	65536 点(ZR0 至 65535)，无需进行块切换。	98304 点(ZR0 至 98303)，无需进行块切换。	131072 点(ZR0 至 131071)，无需进行块切换。	393216 点(ZR0 至 393215)，无需进行块切换。	
			SRAM 卡(1M 字节)	517120 点(ZR0 至 517119)，无需进行块切换。				
			SRAM 卡(2M 字节)	1041408 点(ZR0 至 1041407)，无需进行块切换。				
			SRAM 卡(4M 字节)	2087936 点(ZR0 至 2087935)，无需进行块切换。				
			SRAM 卡(8M 字节)	4184064 点(ZR0 至 4182015)，无需进行块切换。				
			Flash 卡(2M 字节)	1041408 点(ZR0 至 1041407)，无需进行块切换。				
			Flash 卡(4M 字节)	2087936 点(ZR0 至 2087935)，无需进行块切换。				

表 2.5 性能规格

项目			通用型 QCPU				备注
			Q02UCPU	Q03UDCPU	Q04UDHCPU	Q06UDHCPU	
软元件点数	步进继电器 [S] *6		8192 点 (S0 至 8191)				软元件点数固定
	变址寄存器 / 通用运算寄存器 [Z]		最多 20 点 (Z0 至 19)				----
	变址寄存器 [Z] (指定 ZR 软元件的 32 位修饰时)		最多 10 点 (Z0 至 18) (以双字使用变址寄存器 [Z])				----
	指针 [P]		4096 点 (P0 至 4095)，可以通过参数设定局部指针 / 公共指针的使用范围。				
	中断指针 [I]		256 点 (I0 至 255) 可以通过参数设定系统中断指针 I28 到 31 的恒定周期间隔。 (0.5 至 1000ms，单位 0.5ms) 默认 I28： 100ms I29： 40ms I30： 20ms I31： 10ms				----
	特殊继电器 [SM]		2048 点 (SM0 至 2047)				软元件点数固定
	特殊寄存器 [SD]		2048 点 (SD0 至 2047)				
	功能输入 [FX]		16 点 (FX0 至 F)				
	功能输出 [FY]		16 点 (FY0 至 F)				
	功能寄存器 [FD]		5 点 (FD0 至 4)				
软元件热备传送字数			----				----
直接链接软元件			直接访问链接软元件的软元件。 MELSECNET/G、MELSECNET/H 专用。 指定格式： J□□\X□□、J□□\Y□□、J□□\W□□、 J□□\B□□、J□□\SW□□、J□□\SB□□				----
智能功能模块软元件			直接访问智能功能模块的缓冲存储器的软元件。 指定格式： U□□\G□□				----
锁存 (停电保持) 范围			L0 至 8191(默认) (可以为 B、F、V、T、ST、C、D、W 设置锁存范围。)				通过参数设置
RUN/PAUSE 触点			可以在 X0 到 1FFF 中各设置一个 RUN/PAUSE 触点。				
时钟功能			年、月、日、时、分、秒、星期。(自动识别闰年) 精度： -2.96 至 +3.74s(TYP.+1.42s)/d 在 0 ℃时 精度： -3.18 至 +3.74s(TYP.+1.50s)/d 在 25 ℃时 精度： -13.20 至 +2.12s(TYP.-3.54s)/d 在 55 ℃时				----
瞬间掉电允许时间			根据电源模块				----
DC5V 内部消耗电流			0.23A	0.33A	0.39A		----
外形尺寸		H	98mm				----
		W	27.4mm				----
		D	89.3mm				----
重量			0.20kg				----

*6 ： 步进继电器是用于 SFC 功能的软元件。

备 注

关于一般规格，请参阅以下手册。

☞ QCPU 用户手册 (硬件设计 / 维护点检篇)

备忘录

第 3 章　顺控程序的构成与执行条件

在 CPU 模块中可以执行的程序有顺控程序、SFC 程序、ST 程序。
本手册对于 SFC 程序、ST 程序没有进行说明。
关于 SFC 程序、ST 程序请参照下述手册。
☞ QCPU(Q 模式)/QnACPU 编程手册 (SFC 篇)
☞ (Q 模式)/QnACPU 编程手册 (ST 文本篇)

(1) 基本模式 QCPU 的程序执行顺序
基本模式 QCPU 执行程序的顺序如图 3.1 所示。

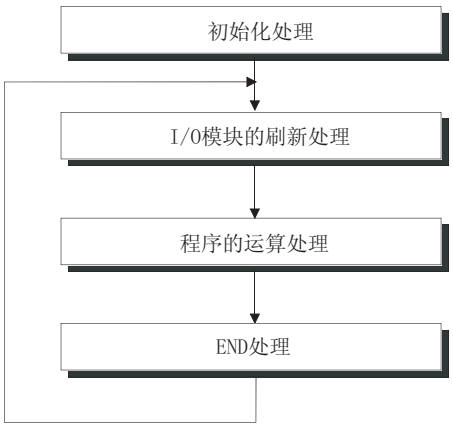


图 3.1 基本模式 QCPU 的程序的执行顺序

(2) 高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 的程序执行顺序

高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 的程序执行顺序如图 3.2 所示。

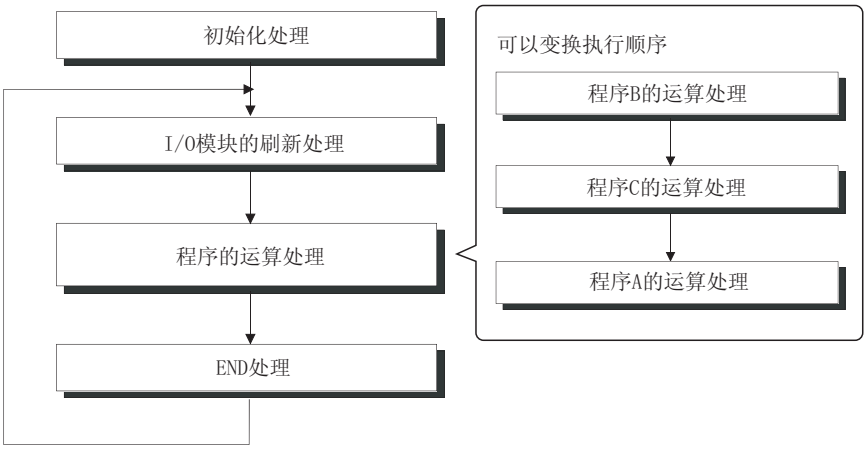


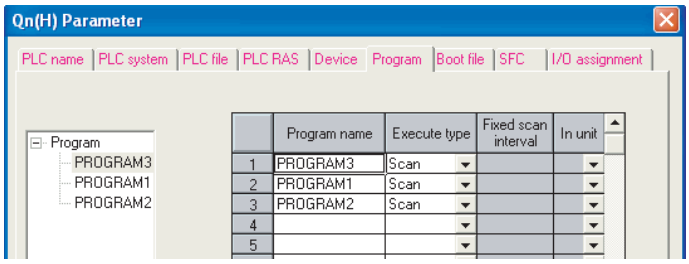
图 3.2 高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 的程序执行顺序

要 点

高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 可以存储多个程序。

程序的执行顺序在 GX Developer 的可编程控制器参数内的程序设定中进行。

(☞ 3.3.6 项)



3.1 顺控程序

顺控程序是使用顺控程序指令、基本指令、应用指令等制成的程序。

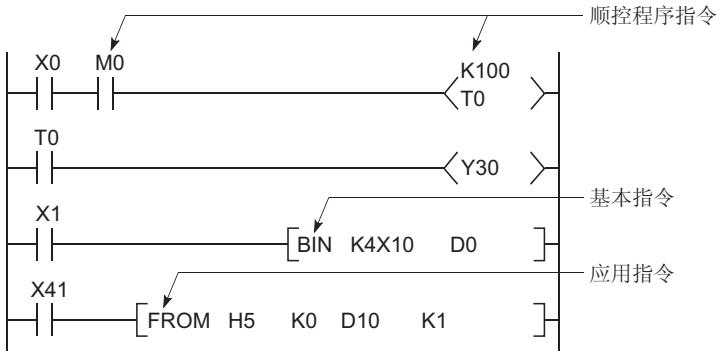


图 3.3 顺控程序

备 注

关于顺控程序指令、基本指令、应用指令，请参照如下手册。

☞ QCPU(Q 模式)/QnACPU 编程手册（公共指令篇）

(1) 顺控程序的记述方法

顺控程序的编程有梯形图模式与列表方式的两种方法。

(a) 梯形图模式

梯形图模式是指以继电器控制的顺控电路为基本着眼点的模式。
与顺控电路的编程相类似。
梯形图模式是以梯形图块为单位来进行编程。
梯形图块是进行顺控程序运算的最小单位。
梯形图块是指从左侧的纵线开始到右侧的纵线结束的电路。

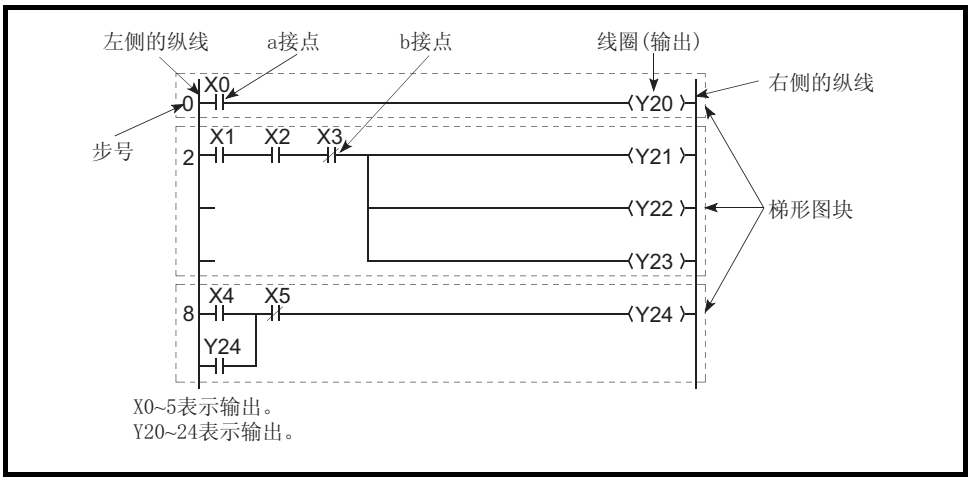


图 3.4 梯形图模式

(b) 列表方式

列表方式通过使用梯形图模式中标有记号的触点、线圈等的专用指令来进行编程。
A 触点、b 触点、线圈变为如下指令。

- a 触点 .... LD，AND，OR
- b 触点 .... LDI，ANI，ORI
- 线圈..... OUT

(2) 顺控程序的运算

顺控程序的运算按照从步 0 到 END/FEND 指令的顺序执行。

在梯形图模式的梯形图块中按照从左侧的纵线到右侧的纵线，从上到下的顺序进行运算。

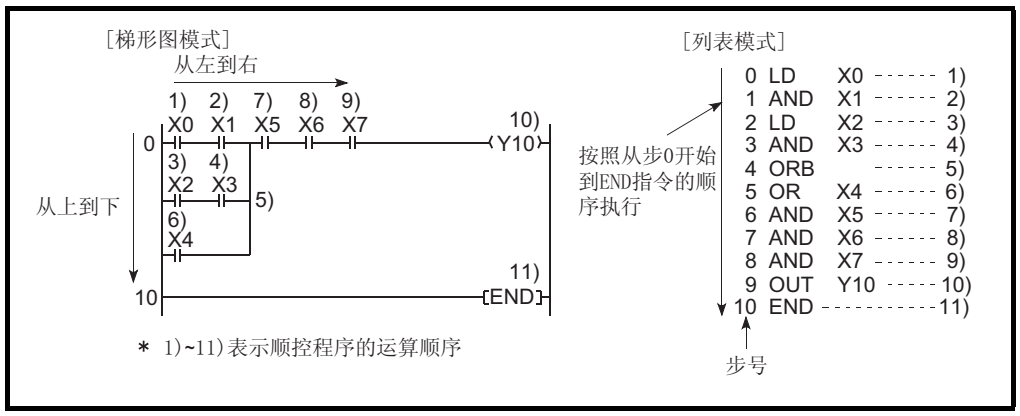


图 3.5 梯形图模式与列表模式的比较

(3) 顺控程序的分类

顺控程序如下被分为三类：

- 主程序 ☞ 3.1.1 项
- 子程序 ☞ 3.1.2 项
- 中断程序 ☞ 3.1.3 项

注 3.1

基本
注 3.1

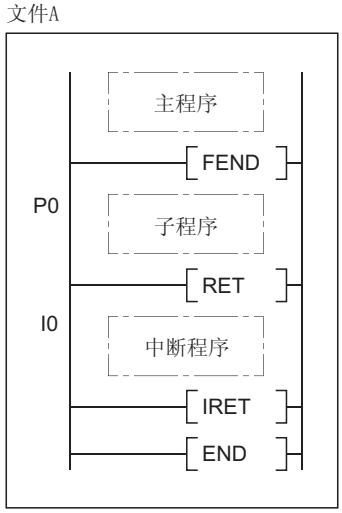
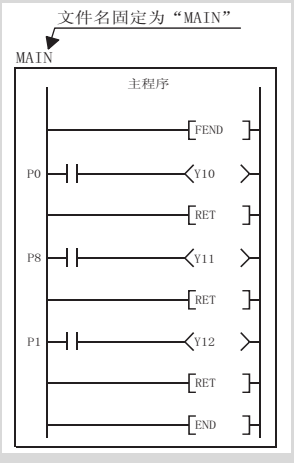


图 3.6 顺控程序的分类

基本
注 3.1

在基本模式 QCPU 中，由于不能执行多个程序的原因，将文件名固定为 “MAIN”。



3.1.1 主程序

(1) 关于主程序

主程序是指从步 0 到 END/FEND 指令的程序。

(2) 主程序的执行动作

主程序的执行动作如下所示。

(a) 只执行一个程序的情况

主程序从步 0 开始执行 END/FEND 指令，进行 END 处理。

在 END 处理过后再次进行从步 0 开始的运算。

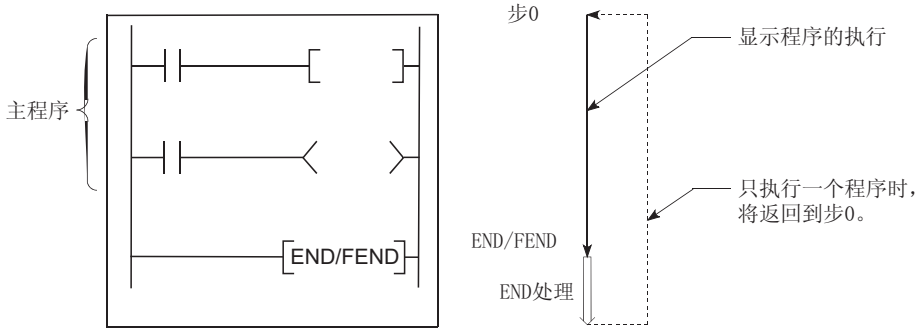


图 3.7 主程序

(b) 执行多个程序的情况

执行完主程序的 END/FEND 指令后的动作，根据设定的执行的条件而会有所不同。

基本 注 3.2

(3) 主程序设定的执行类型*注 3.2*

在执行多个程序的情况下，主程序与使用用途相对应，有如下五种类型的执行程序可以设定。

冗余 注 3.3　通用 UD 注 3.3

- 初始执行类型程序　☞ 3.3.1 项
- 扫描执行类型程序　☞ 3.3.2 项
- 低速度执行类型程序　☞ 3.3.3 项 *注 3.3*
- 备机类型程序　☞ 3.3.4 项
- 恒定周期执行类型程序　☞ 3.3.5 项

要 点

在只执行一个程序之际没有设定执行类型（☞ 3.3.6 项）的情况下，主程序以扫描执行类型程序进行动作。

备 注

关于 END/FEND 指令的详细情况，请参照下述手册。

☞ QCPU(Q 模式)/QnACPU 编程手册（公共指令篇）

基本 注 3.2

基本模式 QCPU 不能执行多个的程序。
因此没有必要设定程序的执行类型。

冗余 注 3.3　通用 UD 注 3.3

冗余 CPU、通用型 QCPU 中，不能使用低速执行类型程序。

3.1.2 关于子程序

(1) 关于子程序

子程序是指从指针 (P□) 开始到 RET 指令的程序。
子程序只有在接到从主程序中调用子程序的调用指令 (CALL(P)、FCALL(P) 等) 时才能被执行。

(2) 子程序的用途

通过如下所示的方法使用子程序，可以减少程序的步数。

- 在一个扫描周期中，通过将多次执行的程序制作成子程序，可以减少整体步数。
- 将只在某种条件成立时才执行的程序制作成子程序，可以减少正常执行的程序的步数。

(3) 子程序的管理

子程序在主程序之后（FEND 指令以后）制作。
子程序可以作为一个程序进行管理。

(a) 在主程序之后制作的情况

1) 子程序的制作地点

子程序在主程序的 FEND 指令 END 指令之间制作 *注 3.4*

基本 注 3.4

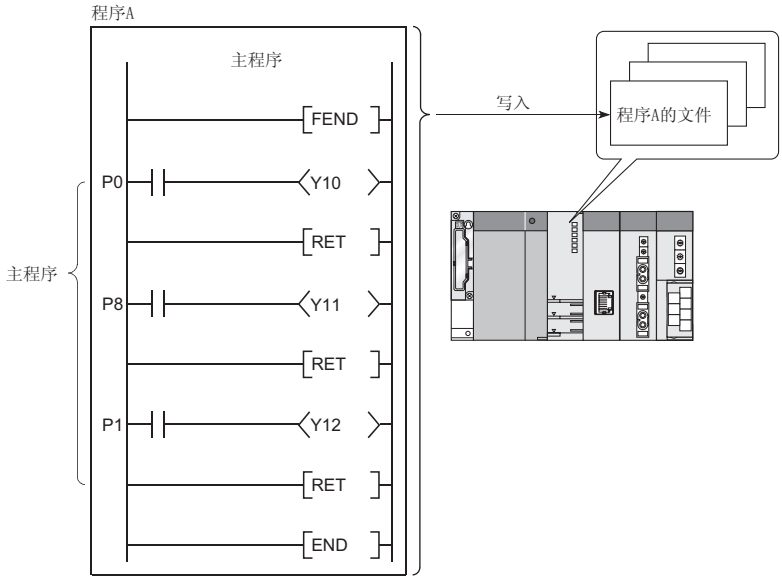
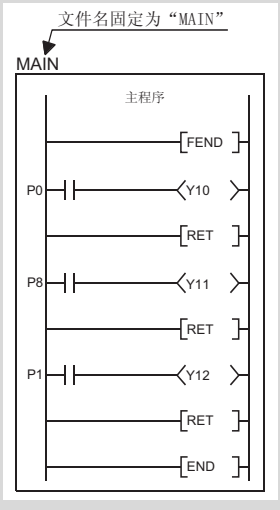


图 3.8 子程序

基本 注 3.4

在基本模式 QCPU 中，由于不能执行多个程序的原因，将文件名固定为 “MAIN”。



2) 对制作顺序的限制
在制作多个子程序时，没有必要将指针从小到大号顺序排列。

基本
注 3.5

3) 可以使用的指针
在子程序中，可以使用本地指针与公共指针。注 3.5
但是，在本地指针的情况下，从其它程序不能调出子程序。(☞ 9.9 节)

备 注

关于副程序的嵌套，请参照 9.8 节。

基本
注 3.6

(b) 作为其它程序进行管理时注 3.6
子程序可以归结为一个程序作为其它程序（备机类型程序）进行管理。
(☞ 3.3.4 项)

基本
注 3.5

基本模式 QCPU 由于不能执行多个程序的原因，因此，本地指针、公共指针没有区别。

基本
注 3.6

基本模式 QCPU 中，由于不能执行多个程序的因子，因此，不能将子程序作为其它的程序进行管理。

3.1.3 中断程序

(1) 关于中断程序

中断程序是指从中断指针 (I□) 开始到 IRET 指令的程序。

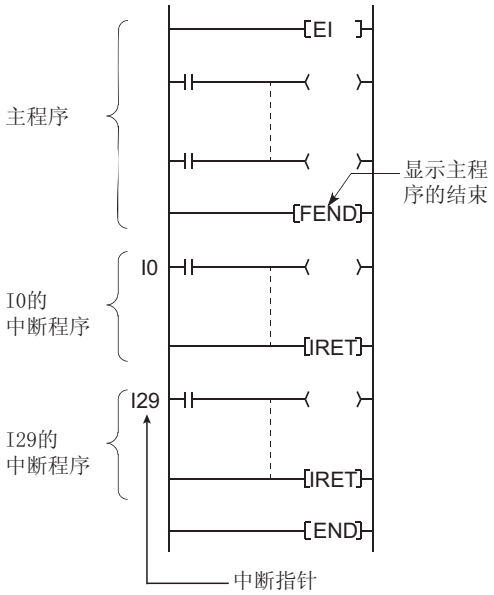


图 3.9 中断程序

根据中断指针号 (I□)，中断因子会有不同。(☞ 9.10 节)

如果出现中断因子，将执行与因子相对应的指针号的中断程序。(中断程序只有在出现中断因子的情况下才会被执行)

中断程序的中断时机

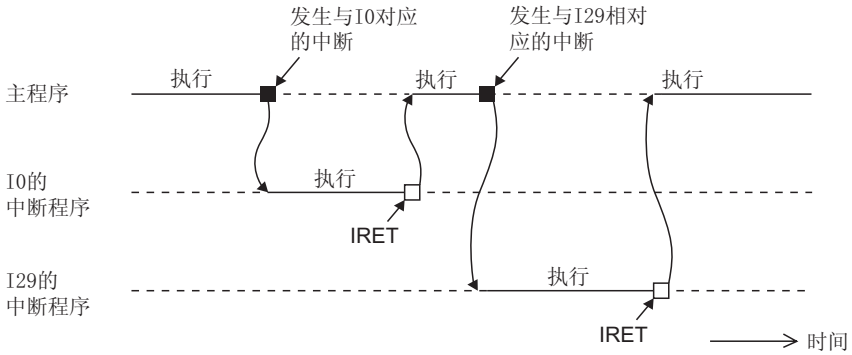


图 3.10 中断程序的执行时机

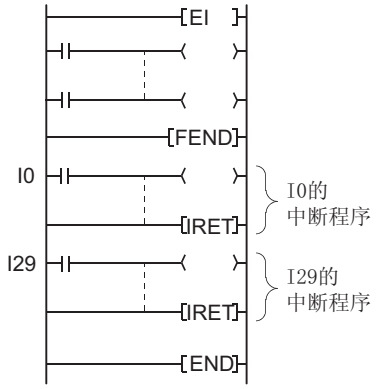
基本 过程 冗余 注 3.7 注 3.7 注 3.7
通用 UD 注 3.7 高性能 注 3.8

要 点

1. 在中断指针中，有高速中断功能专用的指针（I49）。
   注 3.7，注 3.8

   在使用 I49 的情况时，请不要执行下述所示的指针以及程序。
   - 其它的中断指针（I49 以外的中断指针）
   - 中断程序
   - 恒定周期执行类型程序

   如果执行上述显示的程序，I49 中断程序在设定的中断周期间隔内不能被执行。
2. 用一个中断指针号只可以制作一个中断程序。



备 注

关于中断因子、中断指针的详细情况，请参照 9.10 节。

基本 过程 冗余 注 3.7 注 3.7 注 3.7
通用 UD 注 3.7

基本模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 中不能使用高速中断功能专用指针（I49）。

高性能 注 3.8

对于高性能模式 QCPU，只有在 QnHCPU 中才可以使用高速中断功能专用指针（I49）。
在其它模块中不能使用。

(2) 中断程序的管理

中断程序在主程序之后 (FEND 指令之后) 制作。

中断程序可以作为一个程序进行管理。

(a) 在主程序之后制作时

1) 中断程序的制作场所

中断程序在主程序的 FEND 指令 ~END 指令之间制作。

基本 注 3.9 *注 3.9*

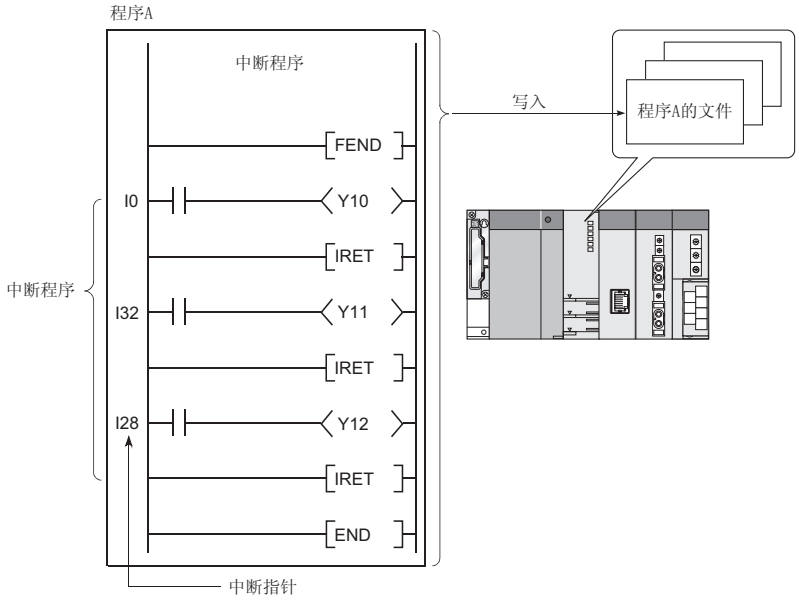


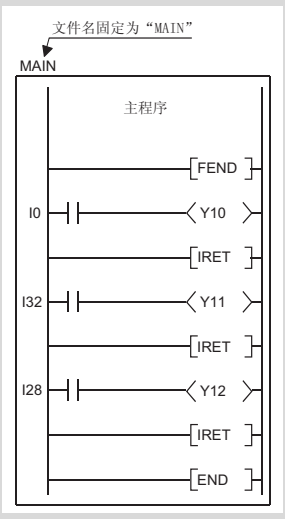
图 3.11 中断程序

2) 对制作顺序的限制

在制作多个中断程序时，没有必要对中断指针按从小到大顺序排列。

基本 注 3.9

在基本模式 QCPU 中，由于不能执行多个程序的原因，将文件名固定为 “MAIN”。



基本
注 3.10

(b) 作为其它程序进行管理的情况 注 3.10
中断程序可以归结为一个程序作为其它程序（备机类型程序）进行管理。
(☞ 3.3.4 项)

(3) 在执行中断程序之前
在执行中断程序时执行下述指令，设为允许中断的状态。

(a) 基本模式 QCPU 的情况
执行 EI 指令设为允许中断状态。

(b) 高性能模式 QCPU、过程 CPU、冗余 CPU 的情况
执行中断指针 I32~47 的中断程序时，通过 IMASK 指令以及 EI 指令设为允许中断状态。
在通过 EI 指令设为中断允许的状态下，中断指针 I0~31、I48~255 的中断程序可以被执行。

(c) 通用型 QCPU 的情况
在通过 EI 指令设为中断允许的状态下，中断指针 I0 ～ 15、I28 ～ 31、I45、I50 ～ 255 的中断程序可以被执行。

备 注

关于 IMASK 指令以及 EI 指令的详细情况，请参照下述手册。
☞ QCPU(Q 模式)/QnACPU 编程手册（公共指令篇）

基本
注 3.10

基本模式 QCPU 中由于不能执行多个程序的原因，因此，中断程序不能作为其它的程序进行管理。

(4) 发生中断因子时的动作

中断程序根据发生中断因子的时机会受到限制。

(a) 在允许中断状态之前发生中断因子的情况

CPU 模块会记忆发生的中断因子。

在变为允许中断状态的时点，与记忆的中断因子相对应的中断程序将被执行。

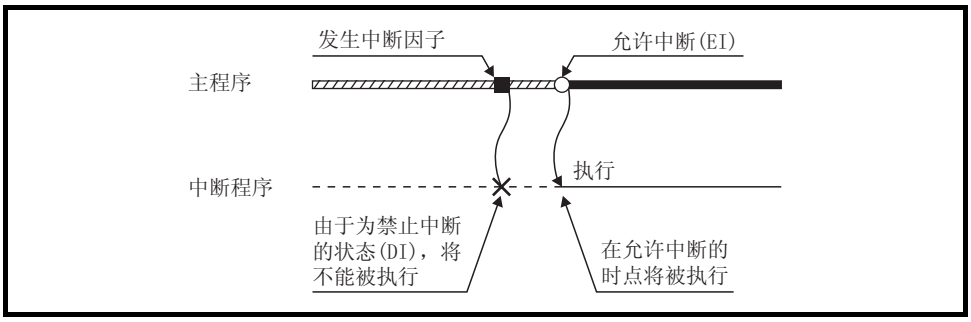


图 3.12 在允许中断状态之前发生中断因子的情况

在允许中断之前多次发生同一个中断因子的情况，如下述情形所示。

1) 基本模式 QCPU 的情况

I0~15，I28~31，I50~127 的中断因子只能被记忆一次。

2) 高性能模式 QCPU、过程 CPU、冗余 CPU 的情况

I0 ～ 27、I28 ～ 31、I50 ～ 255 以及恒定周期执行型程序的中断因子只能被记忆一次。

关于 I32 ～ 41、I49 的中断因子，在中断禁止期间发生时将被删除。

3) 通用型 QCPU 的情况

I0 ～ 15、I28 ～ 31、I45、I50 ～ 255 以及恒定周期执行型程序的中断因子只能被记忆一次。

但是，通过 IMASK 指令屏蔽时发生的中断因子将全部被删除。

(b) STOP/PAUSE 状态下发生中断因子的情况

STOP/PAUSE 状态下发生中断因子的情况下，CPU 模块在变为 RUN 状态后，在变为允许中断的时点执行与中断因子相对应的中断程序。

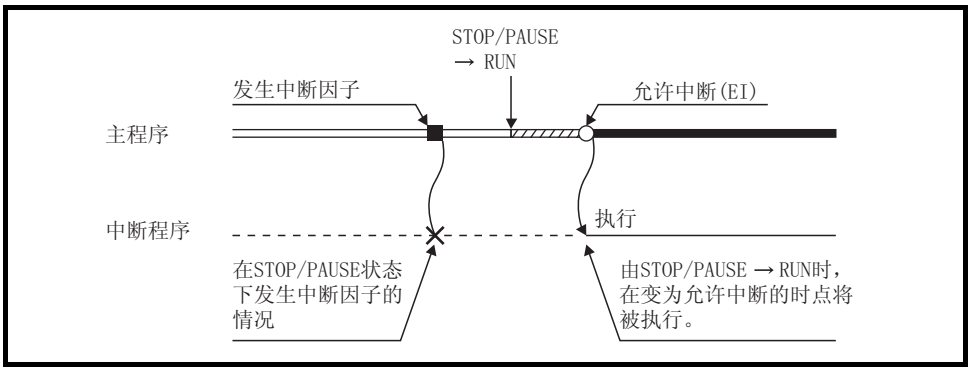


图 3.13 在 STOP/PAUSE 状态下发生中断因子的情况

(c) 在允许中断状态下，同时发生多个中断因子的情况。
在优先顺序中，从与高中断指针号（I□）相对应的中断程序开始执行。
（☞ 9.10 节）
其它的中断程序将一直等到执行中的中断程序处理完成。

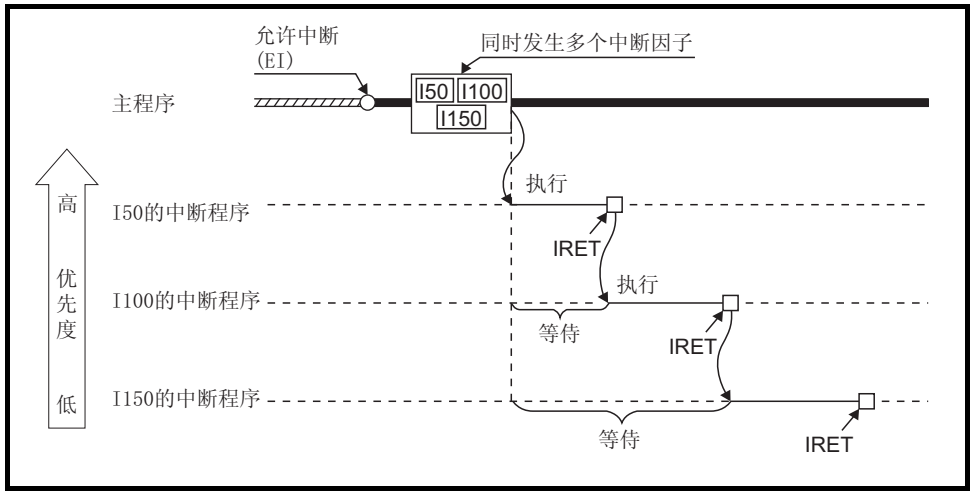
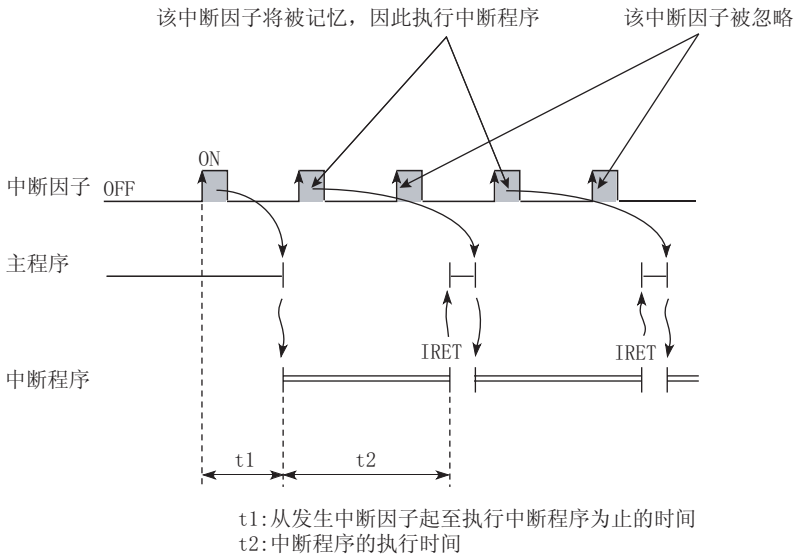


图 3.14 同时发生多个中断因子的情况

(d) 发生了与执行中的中断程序相同的中断因子时
在执行中的程序处理完成之前，发生了与执行中的中断程序相同的中断因子的情况如下所述：

1) 基本模式 QCPU 的情况
I0 ～ 15、I28 ～ 31、I50 ～ 127 的中断因子将只被记忆 1 次。中断程序执行结束后，执行记忆中断因子的中断程序。
即使发生了多次相同中断因子，可记忆的只有最初的 1 次。
从第 2 次以后的中断因子将被忽略。



t1:从发生中断因子起至执行中断程序为止的时间
t2:中断程序的执行时间

图 3.15 发生了与执行中的中断程序相同的中断因子时的动作

2) 高性能模式 QCPU、过程 CPU、冗余 CPU 的情况
- I0 ～ 27、I50 ～ 255 的中断因子只被记忆一次。中断程序执行结束后，执行所记忆的中断因子的中断程序。
  对于 I28 ～ 31 以及恒定周期执行类型程序，发生的次数的中断因子将全部被记忆，中断程序执行结束后，执行所记忆的中断因子的中断程序。
- I32 ～ 41、I49 的中断因子将被删除。

3) 通用型 QCPU 的情况
I0 ～ 15、I45、I50 ～ 255 的中断因子只被记忆一次。中断程序执行结束后，执行所记忆的中断因子的中断程序。
对于 I28 ～ 31 以及恒定周期执行类型程序，发生的次数的中断因子将全部被记忆，中断程序执行结束后，执行所记忆的中断因子的中断程序。

(e) 正在执行指令的情况
在主程序指令的执行过程中，有时会发生指令的执行处理被中断而执行中断程序的情况。
在主程序与中断程序中重复使用软元件的情况下，有时会发生与软元件数据相背离的情况。
在此时，有必要通过下述对策防止软元件数据的背离：

1) 将软元件数据转移到其它的软元件中
由中断程序写入的软元件不要通过主程序直接指定，应通过传送指令等将其转移到其它的软元件中使用。

2) 通过 DI 指令禁止中断
对于主程序中如果被中断将发生异常的指令，应通过 DI 指令执行中断禁止。但是，由于在对指令的各个变量的软元件进行存取的过程中没有中断程序进入，因此，不会发生各个变量单位数据背离的情况。注 3.11

基本
注 3.11

(f) 链接刷新过程中发生了中断的情况

如果在链接刷新过程中发生了中断，将中止链接刷新，执行中断程序。

在 MELSECNET/G 网络系统 注 3.12、注 3.13 或者 MELSECNET/H 网络系统中，即使执行了循环数据的站单位块保证，但如果在中断程序中使用了被设置为刷新对象的软元件，将无法进行循环数据的站单位块保证。

在中断程序中，请不要使用刷新对象的软元件。

高性能 注 3.12

基本 注 3.13　过程 注 3.13　冗余 注 3.13

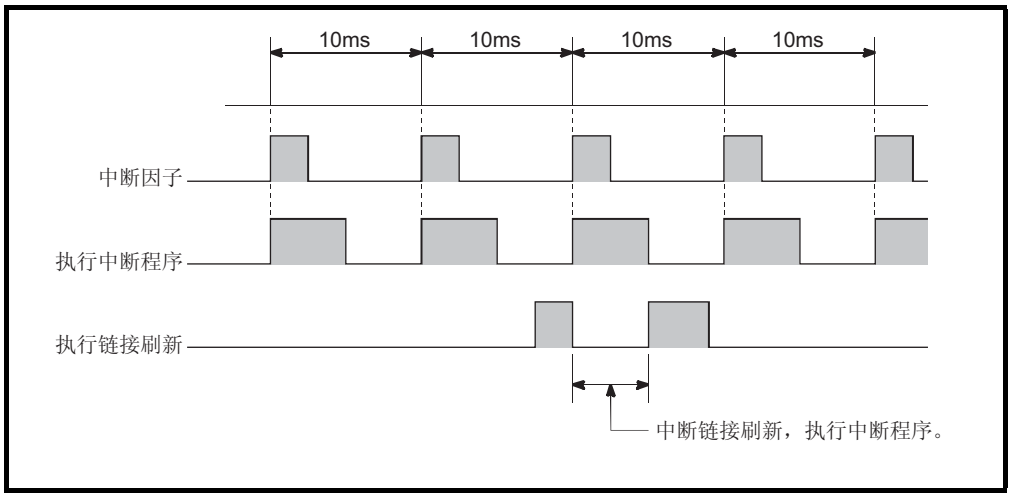


图 3.16 在网络刷新中发生中断的情况

备 注

关于循环数据的站单位块的保证，请参照下述手册。

☞ MELSECNET/G 网络系统参考手册（控制网络篇）

☞ Q 系列 MELSCNET/H 网络系统参考手册（可编程控制器网络篇）

☞ Q 系列 MELSCNET/H 网络系统参考手册（远程 I/O 网络篇）

(g) END 处理过程中的中断

在执行恒定扫描时，在等待 END 处理中发生中断因子时，执行与中断因子相对应的中断程序。

高性能 注 3.12

在高性能模式 QCPU 中使用 MELSCNET/G 时，请确认 CPU 模块以及 GX Developer 的版本。（☞ 附录 4.2）

基本 注 3.13　过程 注 3.13　冗余 注 3.13

在基本模式 QCPU、过程 CPU、冗余 CPU 中，不能使用 MELSCNET/G。

基本 注 3.14 冗余 注 3.14 通用 UD 注 3.14

(5) 变址寄存器以及文件寄存器号的退避与返回

通过 CPU 模块的默认值执行中断程序的情况下，在扫描执行类型程序 / 低速度执行类型程序注 3.14 与中断程序的切换时，将进行变址寄存器以及文件寄存器的块号的退避与返回。(☞ 9.6.4 项)

(6) 中断程序的高速执行的设定与总时间

可编程控制器参数的可编程控制器系统设定中，在选择了中断程序的 “高速执行”的情况下，从主程序向中断程序切换时，不进行变址寄存器的退避与返回。

可以缩短中断程序的总时间。

(☞ 第 10 章)

(7) 在程序制作上的限制

关于在中断程序制作上的限制的说明。

(a) 关于通过 PLS 等指令 ON/OFF 状态下的软元件

通过 PLS，PLF 之类的指令 ON/OFF 处理后的下一个扫描中，由 OFF/ON 处理指令接通 (ON) 的软元件，直到同一软元件指令再度执行时，都保持 ON/OFF 的原有状态。

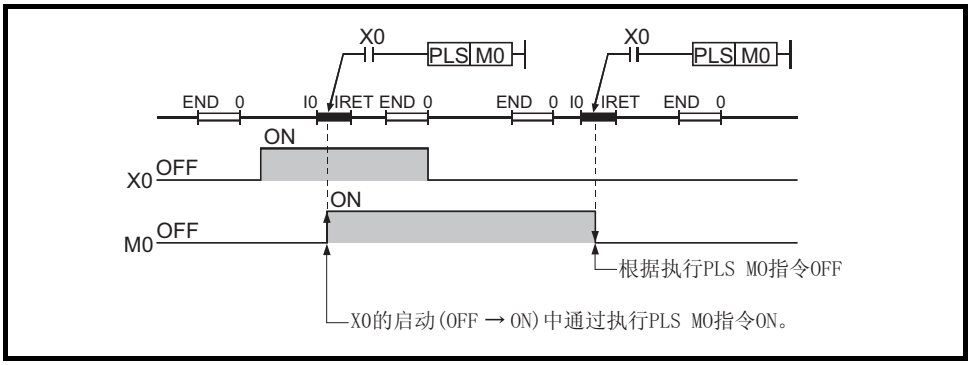


图 3.17 通过中断程序内的 PLS 开启 (ON) 的软元件

(b) 关于 EI/DI 指令

在执行中断程序过程中，为了不执行其它的中断处理，处于中断禁止 (DI) 状态。

在中断程序中，请不要执行 EI/DI 指令。

(c) 关于计时器 (T)、计数器 (C)

在中断程序中不要使用计时器 (T) 以及计数器 (C)。

在一个扫描周期内如果多次发生中断时，中断程序内的计时器 (T) 将不能进行正常的计测。

在一个扫描周期内如果多次发生中断或者由于执行中断程序时的 OUT C□指令的状态，中断程序内的计数器 (C) 将不能进行正常的计测。

基本 注 3.14 冗余 注 3.14 通用 UD 注 3.14

在基本模式 QCPU、冗余 CPU、通用型 QCPU 中，不能使用低速度执行类型程序。

(d) 关于在中断程序中不能使用的指令
关于在中断程序中不能使用的指令，请参照各个编程手册的各个指令部分。

基本
注 3.15

(e) 在执行时间测量时等执行中断 / 恒定周期执行类型程序的情况。
使用特殊寄存器测量扫描时间及执行时间时，如果执行了中断 / 恒定周期执行类型程序*注 3.15*，所测量的时间值将加上上述程序的执行时间。
下述显示的特殊寄存器的存储值以及 GX Developer 的监视值（测量时间）在上述程序执行的情况下将延长。

1) 基本模式 QCPU 的情况
- 特殊寄存器
  SD520， SD521 ：当前扫描时间
  SD524， SD525 ：最小扫描时间
  SD526， SD527 ：最大扫描时间
  SD540， SD541 ： END 处理时间
  SD542， SD543 ：恒定扫描等待时间
  SD548， SD549 ：扫描执行类型程序执行时间
- GX Developer 的监视值
  扫描时间的测量
  恒定扫描

基本
注 3.15

基本模式 QCPU 不使用恒定周期执行类型程序。

2) 高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 的情况

• 特殊寄存器

SD520、SD521 : 当前扫描时间
SD522、SD523 : 初始扫描时间
SD524、SD525 : 最小扫描时间
SD526、SD527 : 最大扫描时间
SD528、SD529 : 低速执行类型程序用当前扫描时间 *注 3.16*
SD532、SD533 : 低速执行类型程序用最小扫描时间 *注 3.16*
SD534、SD535 : 低速执行类型程序用最大扫描时间 *注 3.16*
SD540、SD541 : END 处理时间
SD542、SD543 : 恒定扫描等待时间
SD544、SD545 : 低速执行类型程序累计执行时间 *注 3.16*
SD546、SD547 : 低速执行类型程序执行时间 *注 3.16*
SD548、SD549 : 扫描执行类型程序执行时间
SD551、SD552 : 服务间隔时间 *注 3.17*

冗余 通用 注 3.16 注 3.16

• GX Developer 的监视值

执行时间测量
扫描时间的测量
恒定扫描1

通用 注 3.17

冗余 通用 注 3.16 注 3.16

在冗余 CPU、通用型 QCPU 中，由于不能使用低速执行类型程序，因此没有低速执行类型程序相关的特殊寄存器。

通用 注 3.17

在通用型 QCPU 中，由于不能使用服务间隔读取，因此没有服务间隔读取相关的特殊寄存器。

3.2 只执行一个顺控程序情况下的设定

顺控程序按照从步 0 到 END/FEND 指令的顺序进行运算。
如果执行 END/FEND 指令，将进行 END 处理。
如果进行 END 处理，将再度进行从步 0 开始的运算。
顺控程序就象这样从步 0 到 END/FEND 指令反复进行运算。

要 点

在本节中，对只制作一个顺控程序的情况进行说明。
在制作多个顺控程序的情况下，在启动时只进行一次的程序，与定期执行程序等一样，可以对每个程序指定执行类型。(☞ 3.3 节)

(1) 扫描时间

是指 CPU 模块对顺控程序的运算从步 0 开始执行，到再度执行同一顺控程序的步 0 的时间。
扫描时间由顺控程序的执行时间以及 END 处理时间构成。
在执行下述程序的情况下，下述程序的执行时间需计入扫描时间中。

基本 注 3.18

- 中断程序
- 恒定周期执行类型程序*注 3.18*

(a) 扫描时间的存储场所

在 CPU 模块中，测量扫描时间的当前值、最小值、最大值并存储在特殊寄存器(SD520，SD521，SD524~527)中。
根据对 SD520，SD521，SD524~527 的监视情况，可以确认扫描时间。

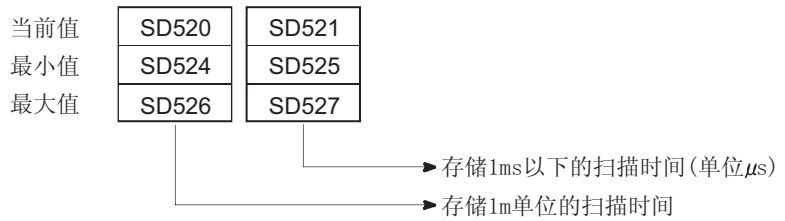


图 3.18 扫描时间的存储场所

SD520 为 3，SD521 为 400 的情况下，扫描时间为 3.4ms。

(b) 扫描时间的精确度与测量

存储在特殊寄存器里的各个扫描时间的精确度为±0.1ms。
另外，在顺控程序中即使执行看门狗定时器复位指令(WDT)，也将继续进行各个扫描时间的测量。

基本 注 3.18

基本模式 QCPU 中，不能使用恒定周期执行类型程序。

(c) 扫描时间的监视
在 CPU 模块中，有监视扫描时间的定时器（看门狗定时器）。(☞ 本节 (2))

(2) WDT（看门狗定时器）
看门狗定时器（以下简称 WDT）是监视扫描时间的定时器。
默认值设定为 200ms。

(a) WDT 的误差
WDT 的误差为 10ms。
WDT(t) 如果设定为 10ms，扫描时间在 10ms ＜ t ＜ 20ms 的范围内会出现 “WDT ERROR”。

(b) WDT 设定
WDT 在可编程控制器参数的可编程控制器 RAS 设定中可以在 10ms~2000ms 的范围内进行变更。（设定单位：10ms）

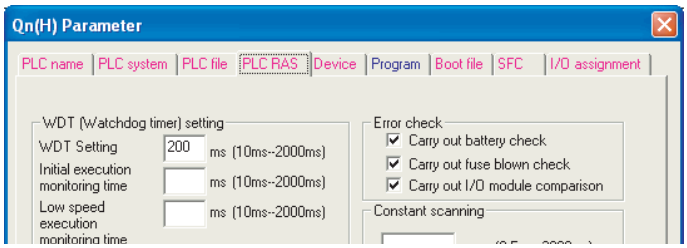


图 3.19 可编程控制器 RAS 设定（WDT 设定）

备 注

在 GX Developer 的程序监视一览表中，可以确认执行中的程序的执行时间。
(☞ 6.13.1 项)

(3) 一定间隔期中反复执行程序的功能。
如果使用恒定扫描功能 (☞ 6.2 节)，可以在一定间隔期内反复执行程序。
在恒定扫描设定时，根据设定的恒定扫描时间执行程序。

基本
注 3.19

3.3 制作、执行多个顺控程序情况下的设定

在制作多个顺控程序的情况下，在启动时只进行一次的程序，象在一定间隔期执行程序等一样，可以对每个程序指定执行类型。

(1) 制作多个顺控程序的用途

可以将程序按控制单位的个数分割成多个程序，并存储在 CPU 模块中。（也可以将程序归结为一个进行存储。）

据此，各个设计人员将可以按一个处理单位分别进行编程。

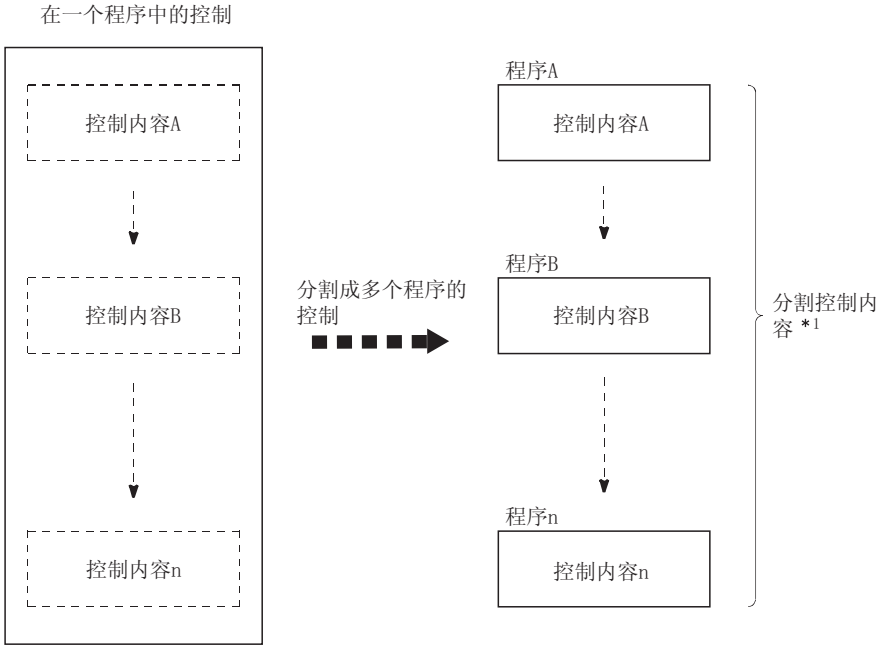


图 3.20 分割成多个程序进行控制

* 1: 关于程序的存储场所，请参照下述说明。

- 基本模式 QCPU : 5.1.1 项
- 高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU : 5.2.1 项

(2) 在执行多个顺控程序时必要的设定

在 CPU 模块中，执行多个顺控程序时，有必要设定执行程序的文件名（程序名）及执行条件。

设定在 GX Developer 的可编程控制器参数内的程序设定中进行。

(3.3.6 项)

基本
注 3.19

基本模式 QCPU 中，由于不能执行多个程序的原因，因此，不能使用制作、执行多个顺控程序时的设定。

(3) 程序的存储场所

在 CPU 模块中执行的程序，可以存储在下述的存储器中。

- 程序内存
- 标准 ROM
- 存储卡

(4) 可以设定的执行类型

在 CPU 模块中可以设定的程序执行类型如下所示。

- 初始执行类型 ☞ 3.3.1 项
- 扫描执行类型程序 ☞ 3.3.2 项
- 低速执行类型程序 ☞ 3.3.3 项 *注 3.20*
- 待机类型程序 ☞ 3.3.4 项
- 恒定周期类型程序 ☞ 3.3.5 项

在冗余 CPU、通用 QCPU 中，不能使用低速执行类型程序。

基本
注 3.21

3.3.1 初始执行类型程序

(1) 关于初始执行类型程序

初始执行类型程序是指在可编程控制器的电源由 OFF→ON 或者由 STOP→RUN 状态切换时只被执行一次的程序。

可以将初始执行类型程序象智能功能模块的初始化处理一样对待，只要执行一次后，从下次扫描开始时便没有必要执行了。

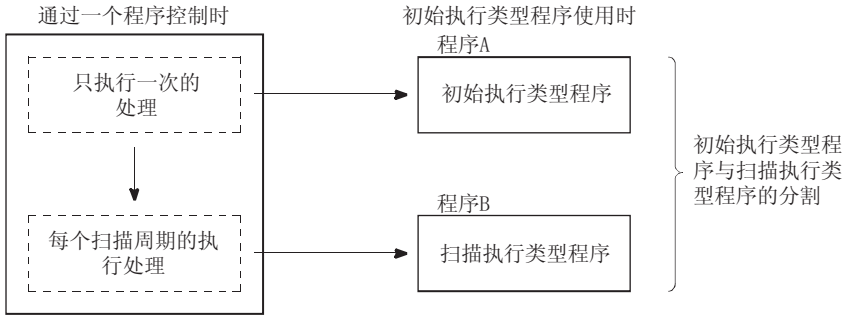


图 3.21 将只执行一次的处理作为执行类型程序进行分割的情况

基本
注 3.21

在基本模式 QCPU 中，不能使用初始执行类型程序。

(2) 初始执行类型程序的处理

(a) 执行顺序

在初始执行类型程序全部完成后进行 END 处理，在下一个扫描开始时执行扫描执行类型程序。

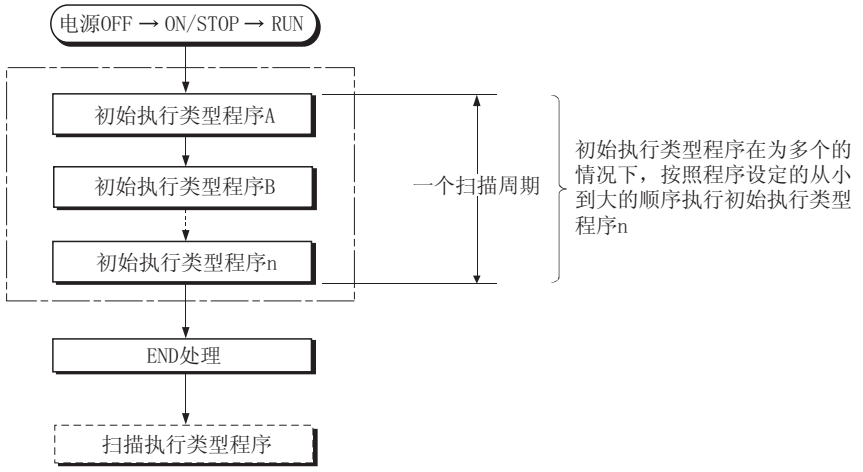


图 3.22 初始执行类型程序的执行顺序

(b) 初始扫描时间

初始扫描时间是指初始执行类型程序的执行时间。

在执行多个初始执行类型程序的情况下，初始扫描时间是指全部初始执行类型程序执行完毕的时间。

1) 初始扫描时间的存储场所

在 CPU 模块中，测量初始扫描时间并存储在特殊寄存器 (SD522，SD523) 中。

根据对 SD522，SD523 的监视情况，可以确认初始扫描时间。

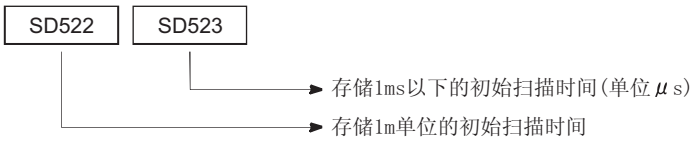


图 3.23 初始扫描时间的存储场所

示例：SD522 为 3、SD523 为 400 的情况下，初始扫描时间为 3.4ms。

2) 扫描时间的精确度与测量

存储在特殊寄存器里的初始扫描时间的精确度为±0.1ms。

另外，在顺控程序中即使执行看门狗计时器复位指令 (WDT)，也将继续进行初始扫描时间的测量。

3) 执行中断程序 / 恒定周期执行类型程序的情况

在初始执行类型程序执行完成之前，如果执行了中断程序 / 恒定周期执行类型程序，则需将 中断程序 / 恒定周期执行类型程序的执行时间加入到初始执行类型程序执行时间中。

(c) 初始执行监视时间

初始执行监视时间是指监视初始扫描的时间。

设定范围为 10ms~2000ms(设定单位：10ms)。

初始执行监视时间没有设定默认值(没有默认值)。

1) 在超出初始执行监视时间的情况

当初始扫描时间超出了设定的初始执行监视时间时，会出现 “WDT ERROR”。

CPU 模块的运算将停止。

要 点

冗余 通用 UD 注 3.22 注 3.22

1. 在执行初始执行类型程序与低速执行类型程序时注 3.22，初始执行类型程序完成后开始执行低速执行类型程序(☞ 3.3.3 项)。
请将初始执行监视时间设定为比初始扫描时间与低速执行类型程序的执行时间之和更长的时间。
2. 在设定初始执行监视时间时，测量值的误差为 10ms。
初始执行监视时间 (t) 如果设定为 10ms，初始扫描时间在 10ms < t < 20ms 的范围内会出现 “WDT ERROR”。

(3) 初始执行类型程序制作上的注意事项

在初始执行类型程序执行完毕之前，不能使用若干个扫描所必要的指令(结束软元件存在的指令)。

例如：SEND，RECV 指令等。

冗余 通用 UD 注 3.22 注 3.22

在冗余 CPU、通用型 QCPU 中，由于不能使用低速执行类型程序，因此在设定初始执行监视时间时，不必理会低速执行类型程序的执行时间。

(4) 为执行初始执行类型程序进行的设定

(a) 程序的设定

在可编程控制器的程序设定中，需将执行类型设为 “初始”。

使用多个初始执行类型程序时，请按照希望执行的顺序登陆程序。

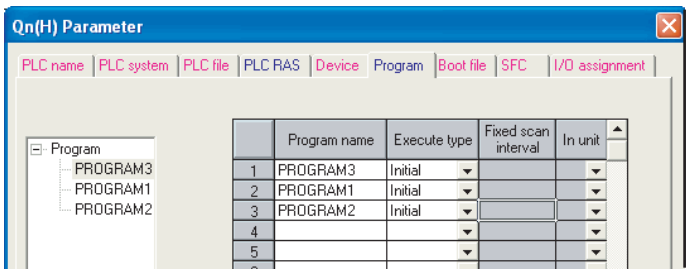


图 3.24 程序的设定

(b) 初始执行监视时间的设定

在监视初始执行类型程序的执行时间时，通过可编程控制器参数的可编程控制器 RAS 设定将初始执行监视时间设定在 10ms~2000ms 的范围内。（设定单位：10ms）

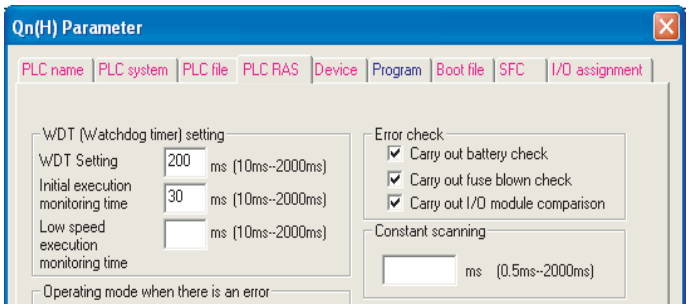


图 3.25 可编程控制器 RAS 设定（初始执行监视时间的设定）

基本
注 3.23

3.3.2 扫描执行类型程序

(1) 关于扫描执行类型程序

扫描执行类型程序是指在执行初始执行类型程序后的下一个扫描开始，在一次扫描周期内只被执行一次的程序。

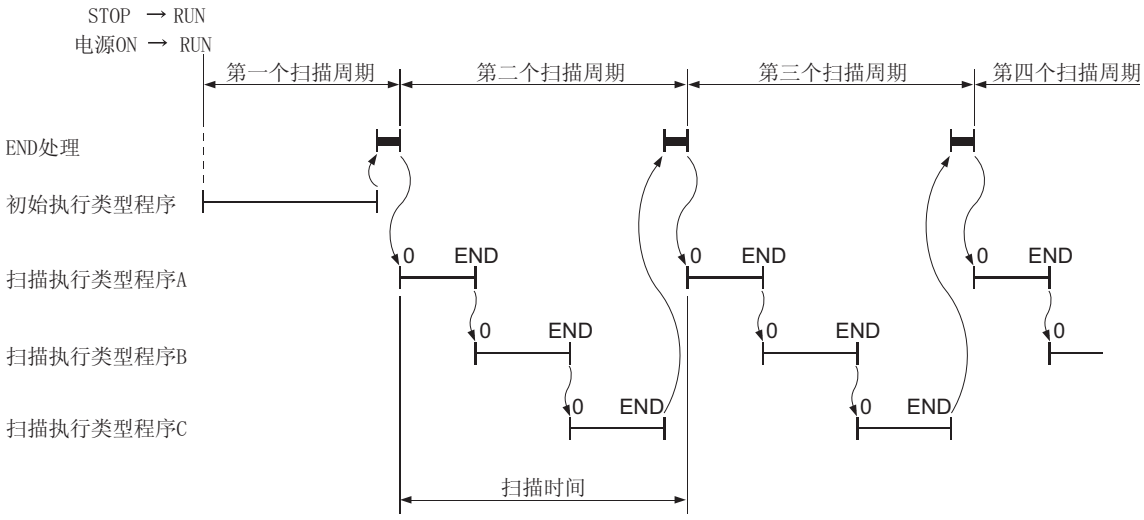


图 3.26 执行类型程序的执行顺序

(2) 扫描执行类型程序的处理

(a) 扫描时间

扫描时间是指扫描执行类型程序的执行时间与 END 处理时间的合计时间。

1) 扫描时间的存储场所

在 CPU 模块中，测量扫描时间的当前值、最小值、最大值并存储在特殊寄存器(SD520，SD521，SD524~527)中。

通过对 SD520，SD521，SD524~527 监视，可以确认扫描时间。

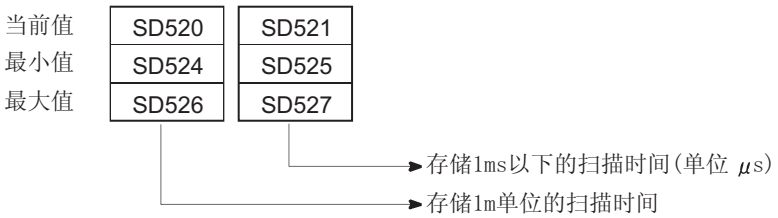


图 3.27 扫描时间的存储场所

例如：SD520 为 3、SD521 为 400 的情况下，扫描时间为 3.4ms。

基本
注 3.23

基本模式 QCPU 中，不能使用多个扫描执行类型程序。
关于在基本模式 QCPU 中执行程序的详细情况，请参照 3.2 节。

2) 扫描时间的精确度与测量
存储在特殊寄存器里的各个扫描时间的精确度为±0.1ms。
另外，在顺控程序中即使执行看门狗定时器复位指令 (WDT)，也将继续进行对各个扫描时间的测量。

3) 执行多个扫描时间执行类型程序的情况
执行多个扫描时间执行类型程序的情况下，所有的扫描执行类型程序执行结束的时间即为扫描执行类型程序的执行时间。
在执行中断程序 / 恒定周期执行类型程序的情况下，加上中断程序 / 恒定周期执行类型程序的执行时间的值即为扫描时间。

(b) END 处理
如果执行全部扫描执行类型程序，系统将进行 END 处理，再度执行起始的扫描执行类型程序。
在各个扫描执行类型程序的最后加上 COM 指令后，可以对每个程序进行 END 处理 (网络刷新)。

(c) WDT(看门狗定时器)
看门狗定时器 (以下简称 WDT) 是监视扫描时间的定时器。
默认值为 200ms。

1) WDT 的误差
WDT 的误差为 10ms。
WDT(t) 如果设定为 10ms，扫描时间在 10ms ＜ t ＜ 20ms 的范围内会出现“WDT ERROR”。

2) WDT 的设定
WDT 在可编程控制器参数的可编程控制器 RAS 设定中可以进行变更。
(☞ 本项 (3)(b))

备 注

在 GX Developer 的程序监视列表中，可以确认执行中的程序的执行时间。
(☞ 6.13.1 项)

(d) 一定间隔期中反复执行扫描执行类型程序的情况。
如果使用恒定扫描功能 (☞ 6.2 节)，可以在一定间隔期内反复执行扫描执行类型程序。
在恒定扫描设定时，根据设定的恒定扫描时间执行扫描执行类型程序。

(3) 为执行扫描执行类型程序进行的设定

(a) 程序的设定

在可编程控制器参数的程序设定中，将执行类型设定为 “扫描”。

在使用多个扫描执行类型程序时，请按照希望执行的顺序登陆程序。

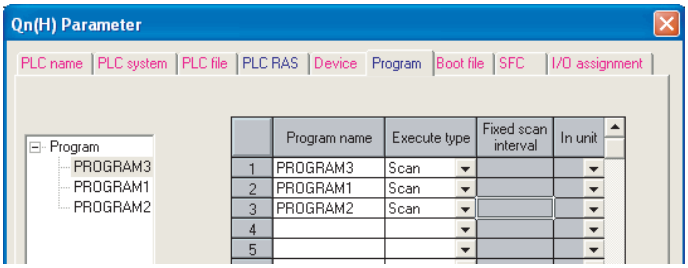


图 3.28 程序的设定

(b) WDT 设定

在变更 WDT 的默认值的情况下，通过可编程控制器参数的 WDT 设定，将 WDT 设定在 10ms~2000ms 的范围内。（设定单位：10ms）

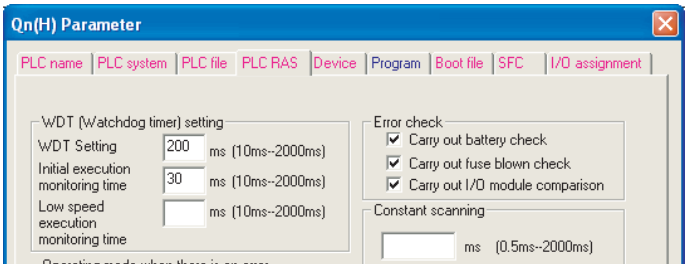


图 3.29 可编程控制器 RAS 设定（WDT 设定）

基本 注 3.24
冗余 注 3.24
通用 注 3.24

3.3.3 低速执行类型程序

(1) 关于低速执行类型程序

低速执行类型程序是指只在恒定扫描的剩余时间或者设定的低速程序执行时间内执行的程序。

低速执行程序是作为向打印机输出等没有必要在每个扫描内执行的程序。

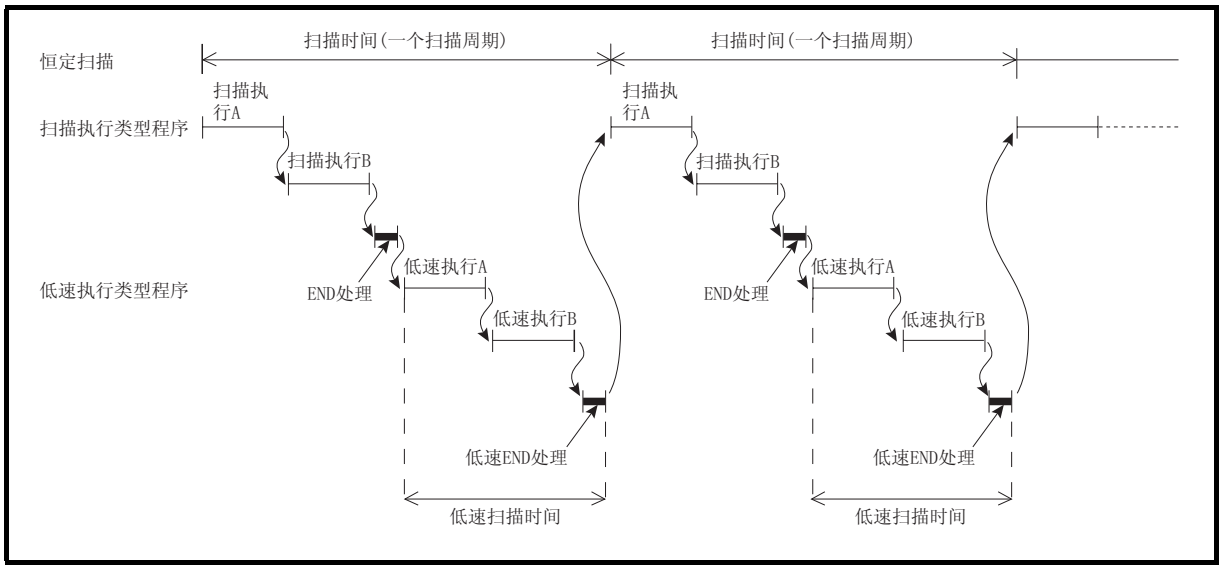


图 3.30 低速执行类型程序的执行（在恒定扫描的剩余时间里执行的情况）

(2) 低速执行类型程序的处理

(a) 执行动作

低速执行类型程序根据设定执行动作会有如下不同。

根据需要，请分别使用下述的操作。

1) 设定一定的扫描时间，控制的精度优先时的情况
设定恒定扫描。

2) 确保执行时间、切实地执行低速执行类型程序的情况
设定低速程序执行时间。

上述的设定可以通过可编程控制器参数的可编程控制器 RAS 设定（☞本项 (4)）进行。

要 点

在执行低速执行类型程序时，请设定恒定扫描或者低速程序执行时间的其中一个。

基本 注 3.24　冗余 注 3.24　通用 注 3.24

在基本模式 QCPU、冗余 CPU、通用型 QCPU 中，不能使用低速执行类型程序。

(b) 一个扫描周期内所有低速执行类型程序运算结束时还有剩余时间的情形
低速执行类型程序的运算完成后的处理，根据特殊继电器 SM330 的 ON/OFF 状态以及低速执行类型程序的执行条件会有不同。

1) 非同步方式 (SM330=OFF)
是指在剩余时间内继续进行低速执行类型程序运算的方式。

2) 同步方式 (SM330=ON)
即使有剩余时间也不继续低速执行类型程序的运算，而是从下一个扫描周期开始进行运算的方式。

表 3.1 动作方式与设定的搭配

低速执行类型程序的动作方式	SM330 的设定状态	低速执行类型程序的执行条件	
		设定恒定扫描时	设定低速程序执行时间时
非同步方式	OFF	再度执行低速执行类型程序 *1	再度执行低速执行类型程序 *2
同步方式	ON	发生恒定扫描的等待时间 *3	开始扫描执行程序的运算 *4

*1: 低速执行类型程序在恒定扫描的剩余时间内反复被执行。
因此，根据扫描周期的不同低速执行类型程序的执行时间会有不同。
(☞ 图 3.31)

*2: 低速执行类型程序在设定的低速程序执行时间内反复被执行。
因此，根据扫描的不同其扫描时间也会有不同。(☞ 图 3.33)

*3: 低速 END 处理结束后的剩余时间作为等待时间。
如果到达设定的恒定扫描时间，扫描执行类型程序将被执行。
因此，各个扫描的扫描时间是一定的。(☞ 图 3.32)

*4: 不考虑低速 END 处理结束后的剩余时间，开始扫描执行类型程序的运算。
因此，不同的扫描其扫描时间会有不同。(☞ 图 3.34)

设定恒定扫描时

在下述条件中执行低速执行类型程序时的动作如下所示。

- 恒定扫描时间 :8ms
- 扫描执行类型程序的合计执行时间 :4ms~5ms
- 低速执行类型程序 A 的执行时间 :1ms
- 低速执行类型程序 B 的执行时间 :3ms
- END 处理 / 低速 END 处理 :0ms(为了便于简单说明假定为 0ms)

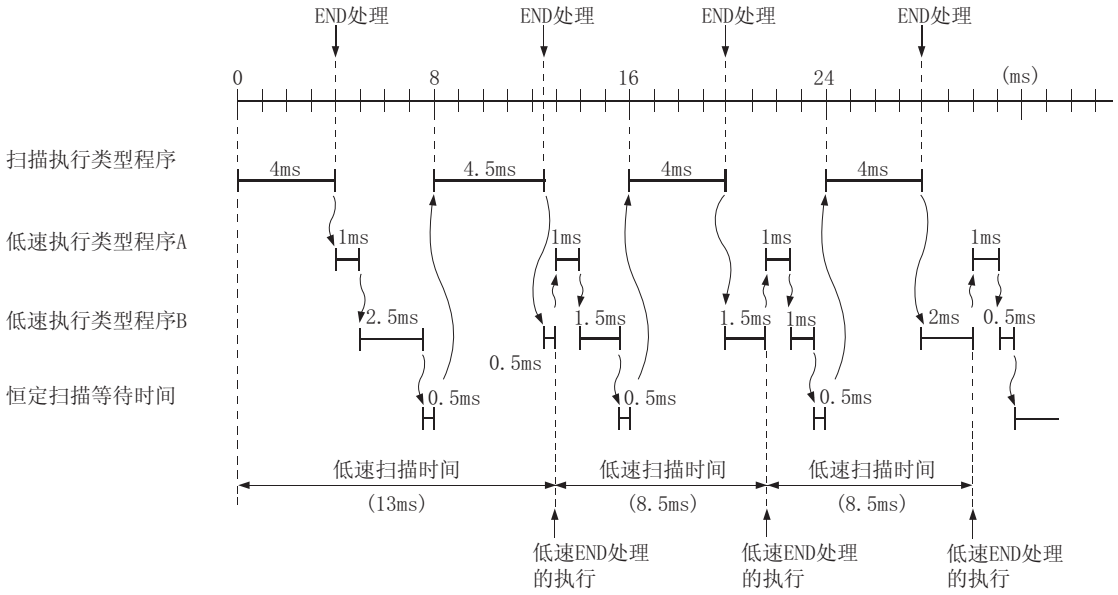


图 3.31 非同步方式 (SM330:OFF)

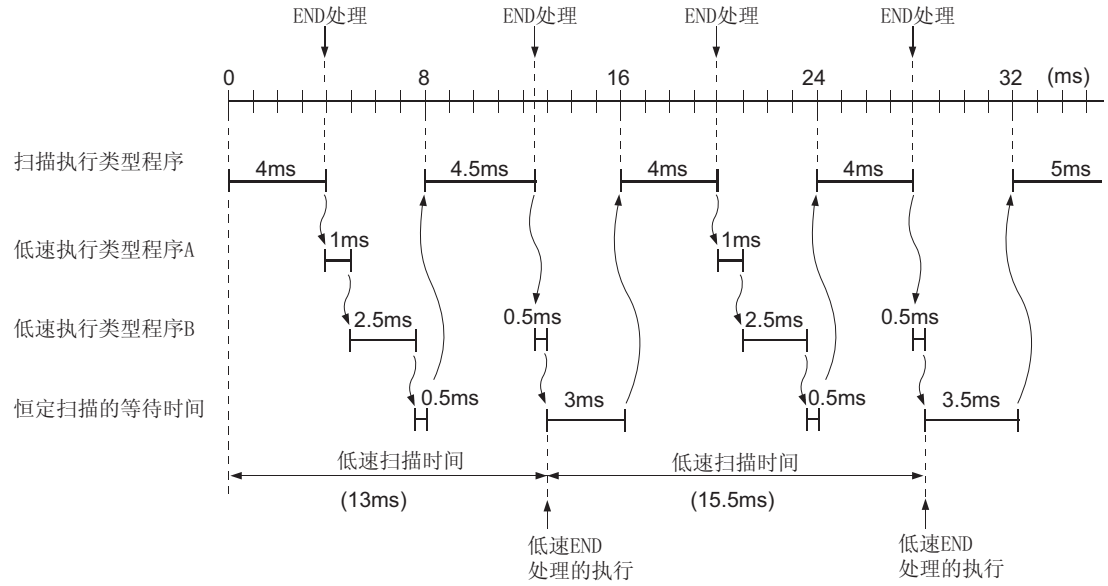


图 3.32 同步方式 (SM330:ON)

设定低速程序执行时间时

说明在下述条件中执行低速执行类型程序时的动作

- 低速程序执行时间 :3ms
- 扫描执行类型程序的合计 :4ms~5ms
- 低速执行类型程序 A 的执行时间：1ms
- 低速执行类型程序 B 的执行时间：3ms
- END 处理 / 低速 END 处理：0ms（为了便于简单说明假定为 0ms）

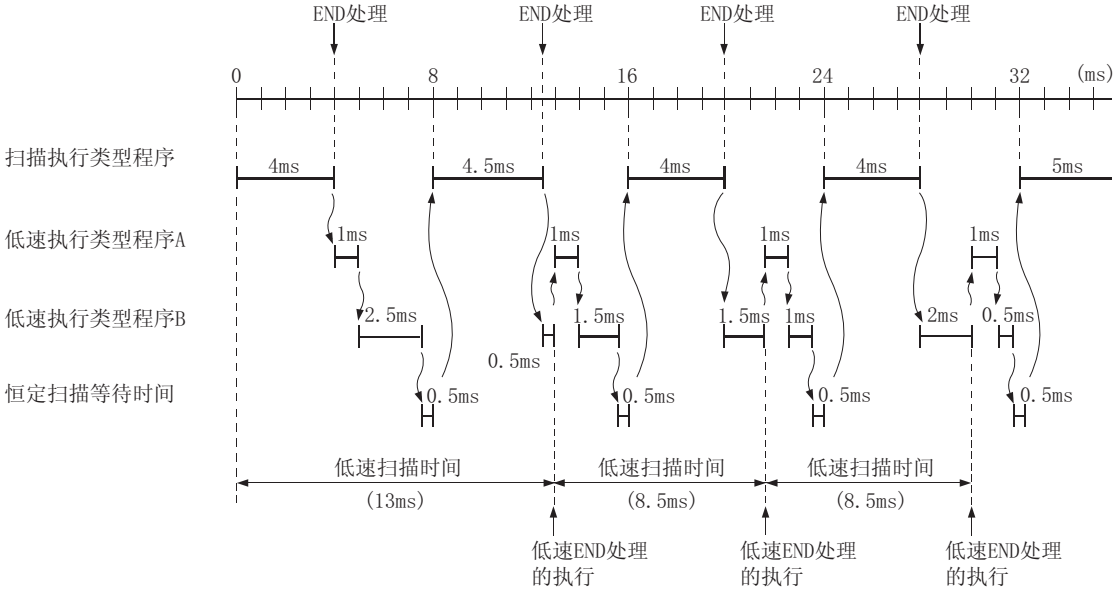


图 3.33 非同步方式 (SM330:OFF)

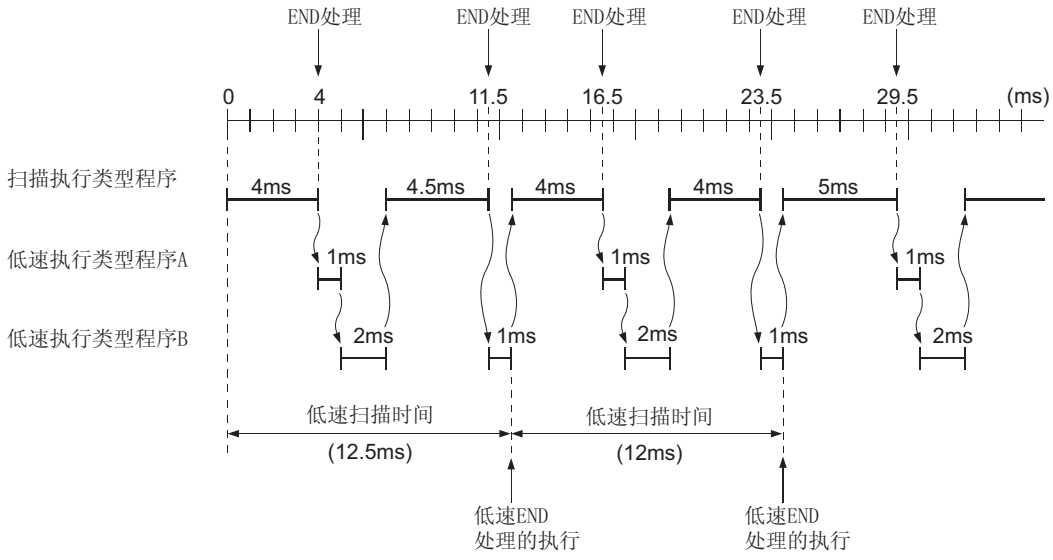


图 3.34 同步方式 (SM330：ON)

(c) 在恒定扫描的剩余时间或者低速执行程序执行时间内，不能进行低速执行类型程序的处理的情况
执行的程序一旦被中断，在下一个扫描周期内将继续执行剩余的程序。

(d) 低速 END 处理
如果执行所有的低速执行类型程序，将进行低速 END 处理。
在低速 END 处理中执行以下处理：

- 低速程序用特殊继电器 / 特殊寄存器的设置 *1
- 低速执行程序的 RUN 中写入
- 低速扫描时间的测量
- 低速执行类型程序的看门狗定时器的复位

低速 END 处理结束后，将再度执行起始的低速执行类型程序。

* 1: 低速程序用特殊继电器 / 特殊寄存器如下所示：
SM330，SM404，SM405，SM510
SD430，SD510，SD528~535，SD544~547

要 点

执行低速执行类型程序时，正在执行的指令的最大处理时间＋低速 END 处理时间，有与恒定扫描不一样的情况。

备 注

关于低速 END 处理与 END 处理的区别，请参照图 3.30。

(e) 低速扫描时间

低速扫描时间是指，所有的低速执行类型程序的执行结束的时间与低速 END 处理时间的合计时间。

关于低速扫描时间与扫描时间的区别，请参照图 3.30。

1) 扫描时间的存储场所

在 CPU 模块中，测量低速扫描时间并存储在特殊寄存器中 (SD528~535)。

根据对 SD528~535 的监视情况，可以确认低速扫描时间。

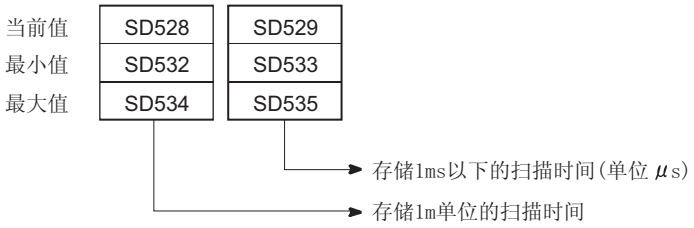


图 3.35 扫描时间的存储场所

例： SD528 为 50、SD529 为 400 的情况下，低速扫描时间为 50.4ms。

备 注

在 GX Developer 程序监视列表中，可以确认低速扫描时间、程序的执行时间等。(☞ 6.13.1 项)

2) 低速扫描时间的精确度与测量

存储在特殊寄存器里的各个扫描时间的精确度为±0.1ms。

另外，在顺控程序中即使执行看门狗定时器复位指令 (WDT)，也将继续进行各个扫描时间的测量。

3) 执行中断程序 / 恒定周期执行类型程序的情况

低速扫描时间为加上中断程序 / 恒定周期执行类型程序的执行时间的值。

(f) 低速执行监视时间

低速执行监视时间是指监视低速扫描时间的时间。

设定范围为 10ms~2000ms(设定单位：10ms)。

低速执行监视时间没有设定默认值 (没有默认值)。

1) 在超出低速执行监视时间的情况

低速扫描时间在超出设定的低速执行监视时间时，会出现 “PRG TIME OVER(出错代码：5010)”。

要 点

在执行低速执行类型程序与初始执行类型程序时，初始执行类型程序（☞ 3.3.1 项）完成后开始执行低速执行类型程序。
在执行低速执行类型程序与初始执行类型程序时，请将初始执行监视时间设定为比初始扫描时间与低速执行类型程序的执行时间之和更长的时间。

(3) 低速执行类型程序制作上的注意事项

(a) 关于低速程序执行时间的设定
如果设定低速程序执行时间，为了预留设定的低速执行类型程序的执行时间，因此，扫描时间将延长。
关于扫描时间，请将低速程序执行时间设定为小于 WDT，或者将 WDT 的值扩大。

(b) 关于不能使用的指令
在低速执行类型程序中不能使用 COM 指令。

(c) 关于执行时机
低速执行类型程序即使在执行初始执行类型程序的扫描周期内也可以执行。
如果不想在执行初始执行类型程序后，执行低速执行类型程序的情况下，请通过特殊继电器 (SM402，SM403) 解除互锁。

(d) 关于设定范围
请设定恒定扫描时间或低速程序执行时间的任何一方。
在设定恒定扫描时间或低速程序执行时间的情况下，如果（恒定扫描的剩余时间）<（低速程序执行时间）的话，将出现 “PRG TIME OVER（出错代码：5010）”

(e) 关于在程序切换之际的变址寄存器
关于从扫描执行类型程序向低速执行类型程序切换时的变址寄存器的处理情况，请参照 9.6.3 项。
关于在低速执行类型程序执行过程中执行中断程序 / 恒定周期执行类型程序情况下的变址寄存器的处理情况，请参照 9.6.4 项。

(4) 为执行低速执行类型程序进行的设定

(a) 程序的设定

在可编程控制器参数的程序设定中将执行类型设定为 “低速”。

在使用多个低速执行类型程序时，请按照希望执行的顺序登陆程序。

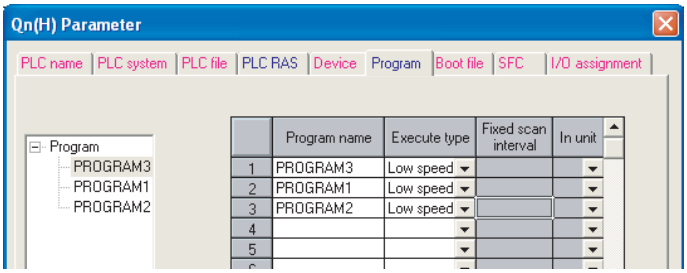


图 3.36 程序的设定

(b) 恒定扫描、低速程序执行时间、低速执行监视时间的设定

设定可编程控制器参数的恒定扫描或者低速程序执行时间。

恒定扫描的设定范围为：0.5ms~2000ms。

（设定单位 0.5ms）

低速程序执行时间的设定范围为：1ms~2000ms。

（设定单位：1ms）

此外，在监视低速执行类型程序的执行时间时，设定低速执行监视时间的范围为：10ms~2000ms。

（设定单位：10ms）

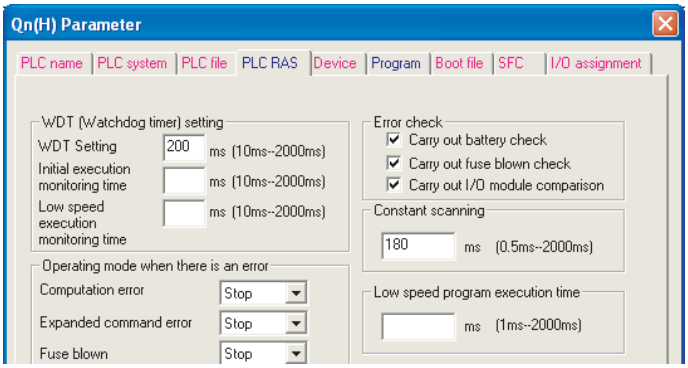


图 3.37 可编程控制器 RAS 设定（恒定扫描、低速程序的执行时间、低速执行监视时间）

要 点

在执行低速执行类型程序时，必须设定恒定扫描或者是低速程序执行时间的其中一方。

基本
注 3.25

3.3.4 待机类型程序

(1) 关于待机类型程序

待机类型程序是指只在有执行要求的情况下执行的程序。
另外，通过顺控程序的指令可以将待机类型切换为其它执行类型来使用。

(2) 待机类型程序的使用用途

待机类型的程序有如下所示的使用用途。

(a) 程序的库化

可将子程序及中断程设为为待机类型程序、与主程序分开管理。
一个待机类型程序中可以制成多个子程序、中断程序。

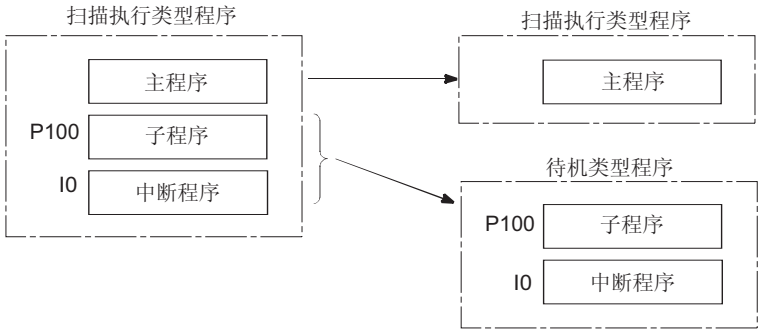


图 3.38 使用待机类型程序的程序库化

基本
注 3.25

在基本模式 QCPU 中不能使用待机类型程序。

(b) 程序的级替换

预先制成与所有系统相对应的程序，但只执行必要的程序的情况下使用该功能。
例如：预先通过可编程控制器参数设定的待机类型的程序，可以在顺控程序中将其变更为扫描执行类型程序，并加以执行。

(3) 待机类型程序的执行方法

待机类型程序通过下述显示的方法可以执行。

- 在待机类型程序内制作子程序、中断程序，在发生中断或者指针等调用时执行 (☞本项 (3)(a))
- 使用程序执行类型的切换指令，将待机类型程序切换为其它执行类型。 (☞本项 (3)(b))

(a) 在待机类型程序内，将子程序或者中断程序综合为一个程序的情况

将子程序或者是中断程序制为一个待机类型程序内时，需从步 0 开始制作。
主程序之后，在制作子程序、中断程序之际没有必要使用 FEND 指令。

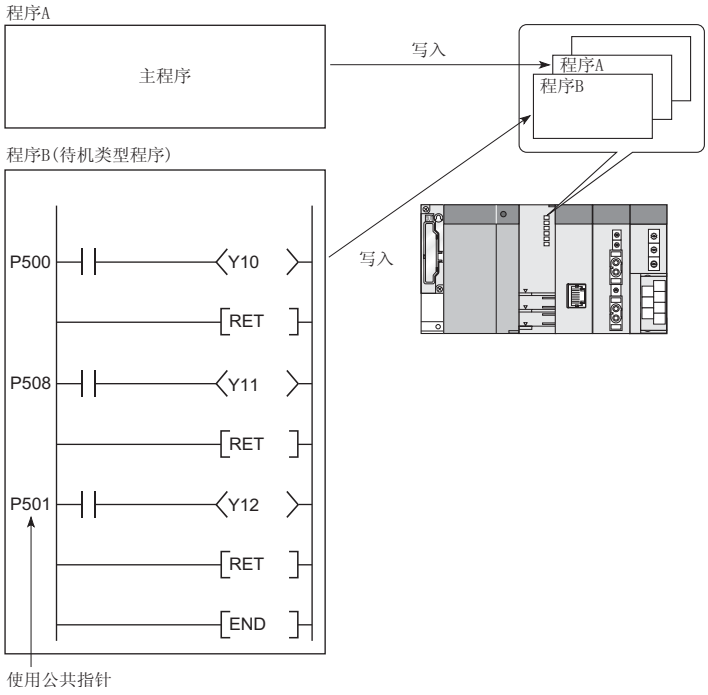


图 3.39 将子程序包含到待机类型程序内的情况

1) 执行待机类型程序内的子程序、中断程序时的动作
如果待机类型程序的执行结束，将重新执行从待机类型程序内的调出的程序。
待机类型程序内的子程序、中断程序执行时的动作如图 3.40 所示。

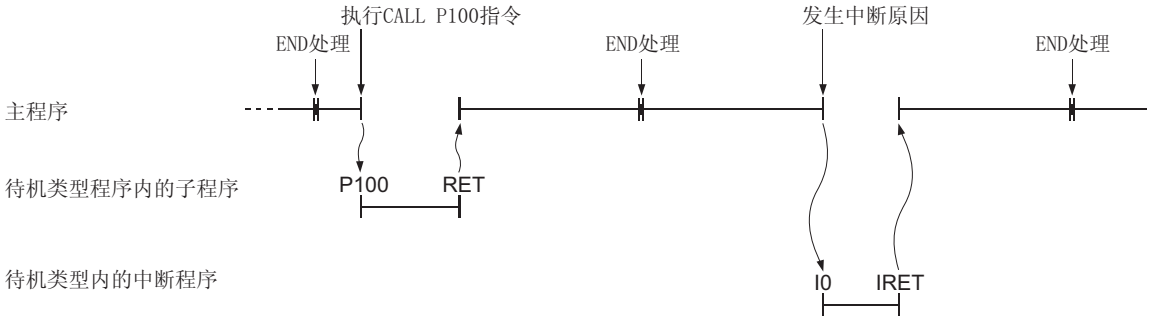


图 3.40 执行待机类型程序内的子程序、中断程序时的动作

要 点

1. 关于制作子程序、中断程序时的限制，请参照下述说明。
   - 子程序 :☞ 3.1.2 项
   - 中断程序 :☞ 3.1.3 项
2. 请使用公共指针。(☞ 9.9.2 项)
   在使用本地指针的情况下，不能从其它的程序执行待机类型程序内的子程序。

冗余 通用
注 3.26 注 3.26

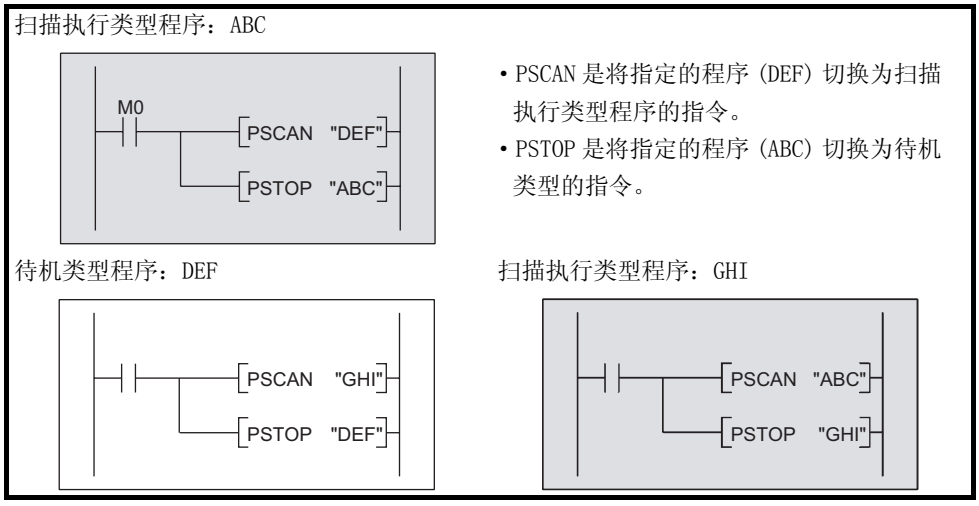
(b) 通过指令切换执行类型的情况

执行类型的切换可以通过 PSCAN 指令、PLOW 指令注 3.26、PSTOP 指令、POFF 指令来进行。

1) 在扫描执行类型程序中切换执行类型的示例。

- 将程序 “ABC”与”GHI”设定为扫描执行类型程序。
  将程序 “DEF” 设定为待机类型程序。
- 在条件成立（图 3.41 中内部继电器 (M0)ON) 时，将程序 “DEF”切换为扫描执行类型程序、“ABC”切换为待机执行类型程序。

[PSCAN、PSTOP 指令执行前]]



接通(ON)M0的情况

[PSCAN、PSTOP 指令执行后]

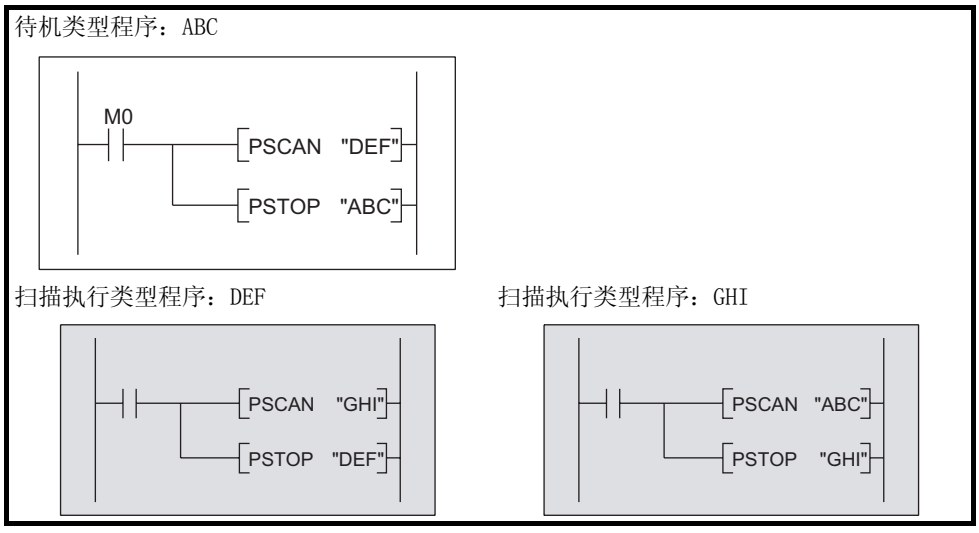


图 3.41 在扫描执行类型程序中切换执行类型的示例

冗余 通用
注 3.26 注 3.26

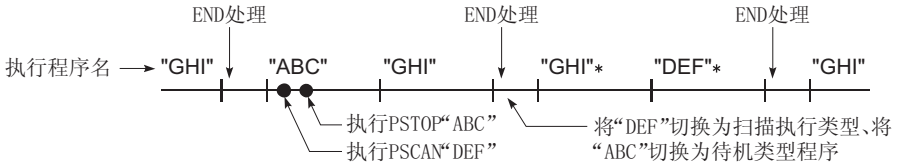
在冗余 CPU、通用型 QCPU 中由于不能使用低速执行类型程序，因此，不能使用 PLOW 指令进行执行类型的切换。

2) 执行类型的切换时机

程序的执行类型的切换由 END 处理执行。

因此，在程序执行过程中，不能切换程序的执行类型。

另外，若将同一个扫描周期的同一程序设定为不同类型时，此后执行的类型，将变为执行类型切换指令所指定的执行类型。



*: "GHI" 与 "DEF" 的程序的执行顺序为通过可编程控制器参数的程序设定中设定的顺序。

图 3.42 执行类型的切换时机

(4) 待机类型程序制作上的注意事项

(a) 关于不能使用的软元件

不能使用的软元件，依据程序的种类（子程序、中断程序）或者根据指令可切换的执行类型的不同而不同。

(b) 关于使用局部软元件的子程序的执行

关于使用局部软元件的子程序的执行情况，请参 9.13.1 项。

(5) 为执行待机类型程序进行的设定

(a) 程序的设定

将制成的程序预先设定为待机类型程序时，可以通过可编程控制器参数的程序设定，将执行类型变为 "待机"。

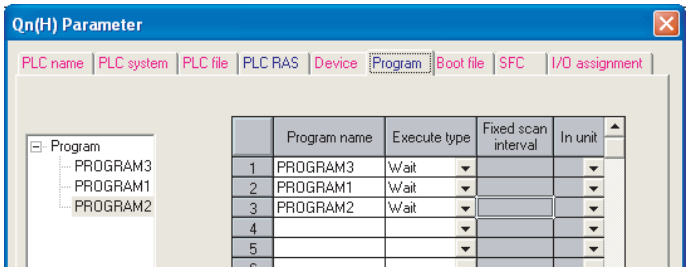


图 3.43 程序的设定

基本
注 3.27

3.3.5 恒定周期执行类型程序

(1) 关于恒定周期执行类型程序

是指根据指定时间执行的程序。

在没有中断指针、IRET 指令的文件单位中可以执行恒定周期程序。

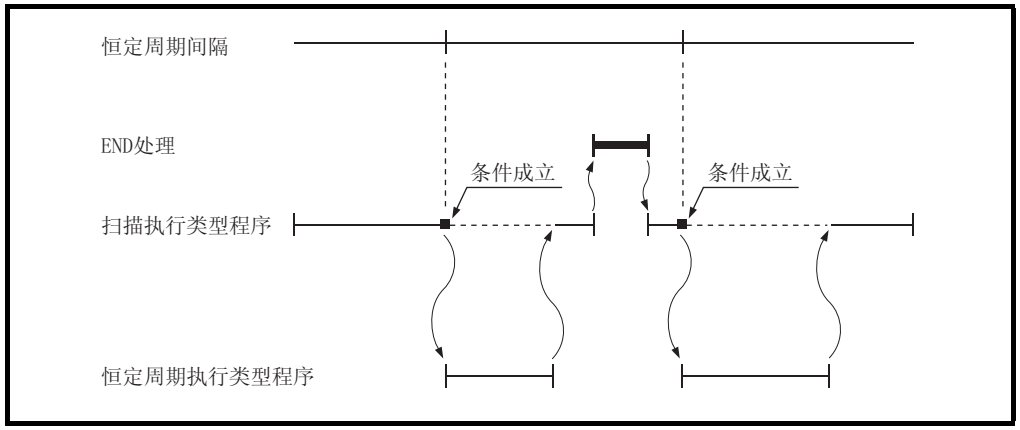


图 3.44 恒定周期执行类型程序的执行

要 点

执行恒定周期执行类型程序时，应在初始执行类型程序 / 扫描执行类型程序中，通过 EI 指令置于中断允许状态。

(2) 恒定周期执行类型程序的处理

下面说明恒定周期执行类型程序的处理。

(a) 在有多个恒定周期执行类型程序的情况

恒定周期执行类型程序根据指定时间而执行。

多个恒定周期执行类型程序通过同一计时达到指定时间的情况下，按照可编程控制器参数的程序设定 (☞本项 (4)(a)) 的由小到大的号码顺序执行。

(b) 有恒定周期执行类型程序与中断程序的情况

恒定周期执行类型程序与中断程序 (I28~31) 通过同一计时达到指定时间的情况下，优先执行中断程序。

基本
注 3.27

在基本模式 QCPU 中不能使用恒定周期执行类型程序。

(c) 在网络刷新过程中的执行

高性能 注 3.28

在网络刷新中恒定周期执行类型程序的执行条件如果成立，则中断网络刷新，执行恒定周期执行类型程序。

基本 过程 冗余 注 3.29 注 3.29 注 3.29

在 MELSECNET/G 网络系统 注 3.28、注 3.29 或 MELSECNET/H 网络系统中，即使进行循环数据的站单位块保证，但如果在恒定周期执行类型程序中使用了设定为刷新对象的软元件，也无法进行循环数据的站单位块保证。

在恒定周期执行类型程序中，请不要使用网络刷新对象的软元件。

备 注

关于循环数据的站单位保证，请参照下述手册

☞ MELSECNET/G 网络系统参考手册（控制网络篇）

☞ Q 系列 MELSCNET/H 网络系统参考手册（可编程控制器网络篇）

☞ Q 系列 MELSCNET/H 网络系统参考手册（远程 I/O 网络篇）

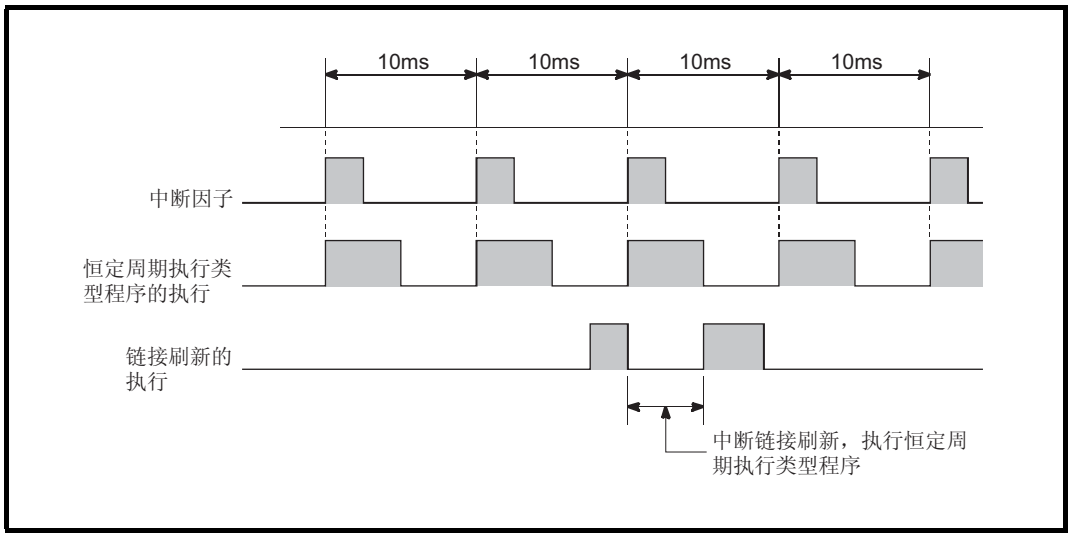


图 3.45 网络刷新中恒定周期执行类型程序执行的情况

高性能 注 3.28

在高性能 QCPU 中使用 MELSECNET/G 时，请确认 CPU 模块以及 GX Developer 的版本。

（☞ 附录 4.2）

基本 过程 冗余 注 3.29 注 3.29 注 3.29

在基本模式 QCPU、过程 CPU、冗余 CPU 中，不能使用 MELSECNET/G。

(d) 在 END 处理中的执行

在执行恒定扫描时以及在 END 指令的等待时间中，如果恒定周期执行类型程序的执行条件成立，将执行恒定周期执行类型程序。

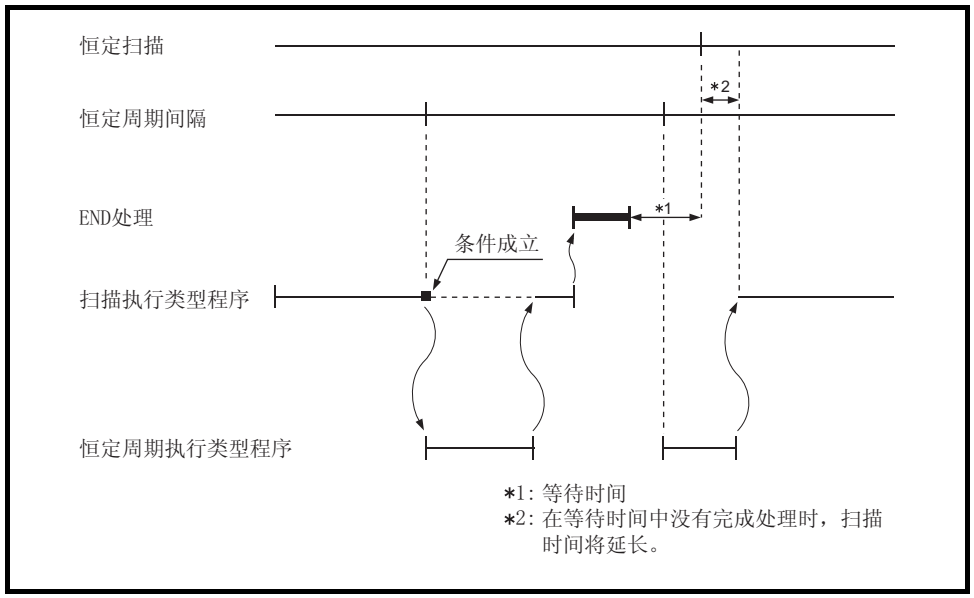


图 3.46 45 在等待时间中恒定周期执行类型程序的执行

冗余 注 3.30　通用 UD 注 3.30

(e) 变址寄存器的处理

关于从扫描执行类型程序 / 低速执行类型程序[注 3.30]向恒定周期执行类型程序切换时的变址寄存器的有关处理情况，请参照 9.6.4 项。

(f) 恒定周期执行类型程序的高速执行的设定与总时间

在执行恒定周期执行类型程序时，进行以下的处理。

- 变址寄存器的退避与返回
- 使用中的文件寄存器的文件名的退避与返回

在可编程控制器参数的可编程控制器系统设定中，如果选择中断程序 / 恒定周期执行类型程序的 “高速执行”，不进行上述处理。

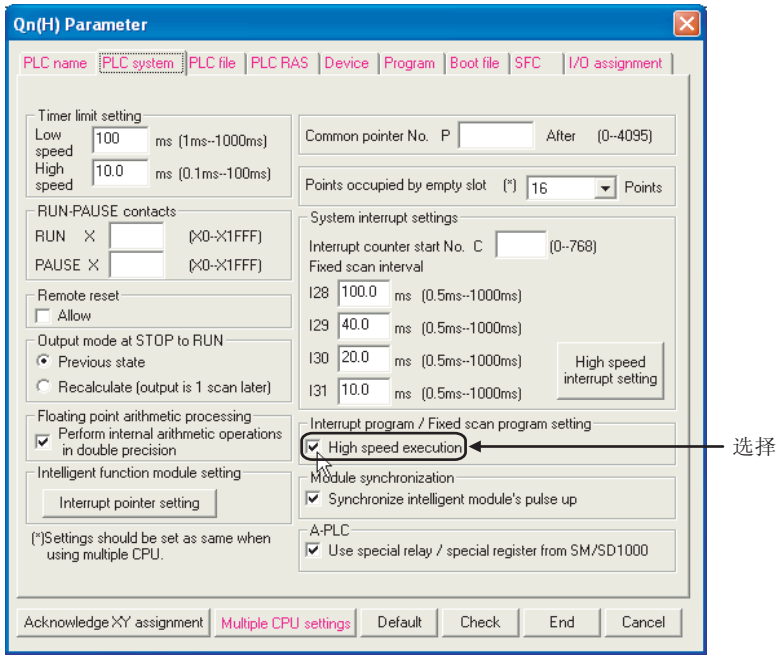


图 3.47 可编程控制器系统的设定画面

可以缩短恒定周期执行类型程序的总时间 (第 10 章)。

冗余 注 3.30　通用 UD 注 3.30

冗余 CPU、通用型 QCPU 不能使用低速执行类型程序。

(3) 程序制作上的注意事项

(a) 关于通过 PLS 等指令 ON/OFF 的软元件

被 PLS PLF 之类的指令 ON/OFF 处理后，在下一个扫描周期里被 OFF/ON 处理指令接通 (ON) 的软元件一直到同一软元件指令再度执行时，都将保持原有的 ON/OFF 状态不变。

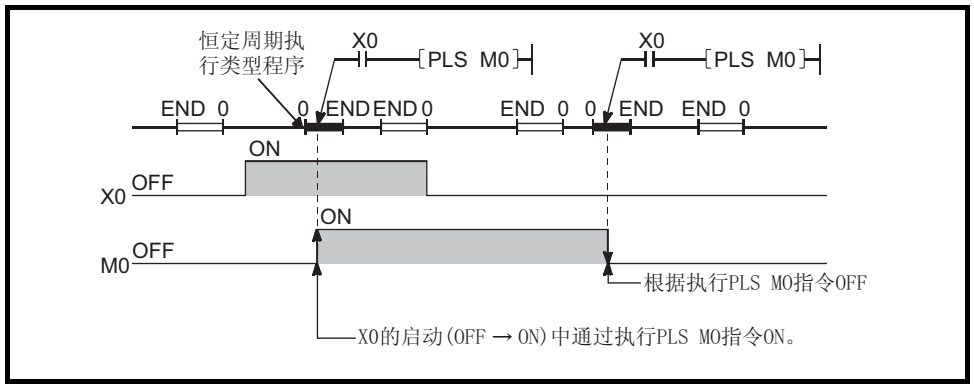


图 3.48 通过恒定周期执行类型程序内的 PLS 指令 ON 的软元件

(b) 关于 EI/DI 指令

在执行恒定周期执行类型程序过程中，为防止其它的中断而变为中断禁止 (DI) 状态。

在恒定周期执行类型程序中，请不要执行 EI/DI 指令。

(c) 关于计时器 (T)、计数器 (C)

在恒定周期执行类型程序中不要使用计时器 (T) 以及计数器 (C)。

在一个扫描周期内如果多次执行恒定周期执行类型程序时，恒定周期执行类型程序内的计时器 (T) 将不能进行正常的测量。

在一个扫描周期内如果多次执行恒定周期执行类型程序时，或者根据执行时的 OUT C□指令的状态，恒定周期执行类型程序内的计数器 (C) 将不能进行正常的测量。

(d) 关于在恒定周期执行类型程序中不能使用的指令

关于在恒定扫描周期执行类型程序中不能使用的指令，请参照各个编程手册的各个指令部分。

(e) 在执行时间测量等时执行中断 / 恒定周期执行类型程序的情况。

使用特殊寄存器测量扫描时间及执行时间等时，如果了执行中断 / 恒定周期执行类型程序，则测量时间值将变为加上上述程序时间的值。

下述显示的对特殊寄存器的存储值以及 GX Developer 的监视值（测量时间）在上述程序执行的情况下将延长。

- 特殊寄存器

  SD520， SD521: 当前扫描时间
  SD522， SD523: 初始扫描时间
  SD524， SD525: 最小扫描时间
  SD526， SD527: 最大扫描时间

冗余 通用 注 3.31 注 3.31
  SD528， SD529: 低速执行类型程序用当前扫描时间注 3.31
  SD532， SD533: 低速执行类型程序用最小扫描时间注 3.31
  SD534， SD535: 低速执行类型程序用最大扫描时间注 3.31
  SD540， SD541: END 处理时间
  SD542， SD543: 恒定扫描等待时间
  SD544， SD545: 低速执行类型程序累计执行时间注 3.31
  SD546， SD547: 低速执行类型程序执行时间注 3.31
  SD548， SD549: 扫描执行类型程序执行时间

通用 注 3.32
  SD551， SD552: 服务间隔时间注 3.32

- GX Developer 的监视值

  测量执行时间
  扫描时间的测量
  恒定扫描

冗余 通用 注 3.31 注 3.31 在冗余 CPU、通用型 QCPU 中，由于不能使用低速执行类型程序，因此没有与低速执行类型程序相关的特殊寄存器。

通用 注 3.32 在通用型 QCPU 中，由于不能使用模块服务间隔读取，因此没有与模块服务间隔读取相关的特殊寄存器。

(f) 关于恒定周期执行程序的执行间隔

根据 DI 指令导致的中断禁止中的时间（中断禁止时间），恒定周期执行程序的执行间隔有时会长于设置时间，请加以注意。

由于 DI 指令导致中断禁止时间变长时，不要使用恒定周期执行类型，应通过恒定周期中断（I28 至 131）使用中断程序。

恒定周期执行间隔的最大公约数＊＜ 中断禁止时间 ... 式 1)

＊恒定周期执行间隔的最大公约数是指，多个恒定周期执行程序中设置的执行间隔设定值的最大公约数。

如果式 1) 成立，与恒定周期执行间隔中设置的间隔相比，实际的恒定周期执行类型程序的执行间隔有时会发生如以下公式所示的时间延迟。

$$\frac{\text{中断禁止时间}}{\text{恒定周期执行间隔的最大公约数}} \times \text{相应程序的恒定周期执行间隔设定值}$$

恒定周期执行类型程序的执行时间的延迟时间如下例所示。

例

- 恒定周期执行间隔 ... 10ms、5ms、1ms、0.5ms
- 恒定周期执行间隔的最大公约数 ... 0.5ms
- 中断禁止时间 (DI).. 5ms
  （中断允许时间 (EI)... 0.5ms 以内）

在上述设置的情况下，式 1) 将为 0.5ms ＜ 5ms。

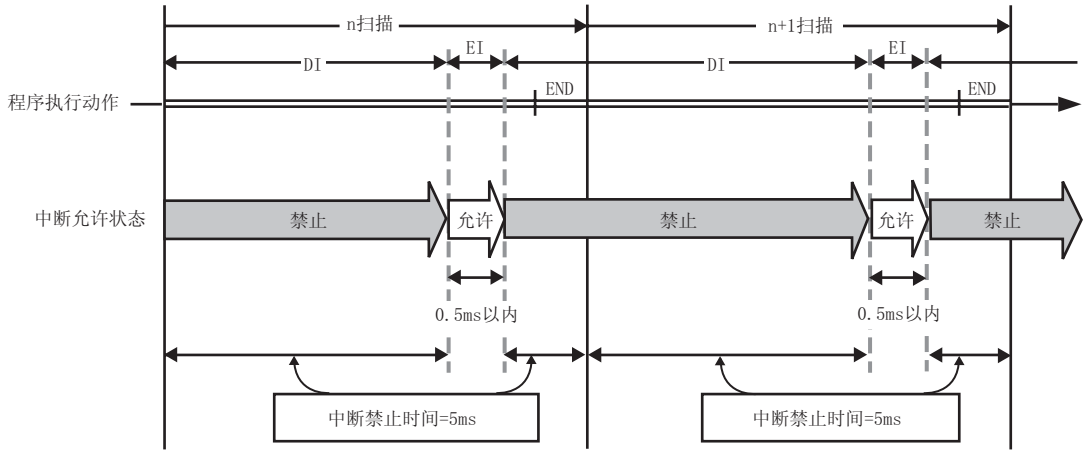


图 3.49 程序的执行动作及中断允许状态

由此，10ms 设置的恒定周期执行类型程序的执行时间为 (5 ÷ 0.5 × 10 = 100)，即最多延迟 100ms。

(4) 为执行恒定周期执行类型程序进行的设定

(a) 程序的设定

在可编程控制器的程序设定中，需将执行类型设定为 “恒定周期”。
使用多个恒定周期执行类型程序时，请按照希望执行的顺序登陆程序。

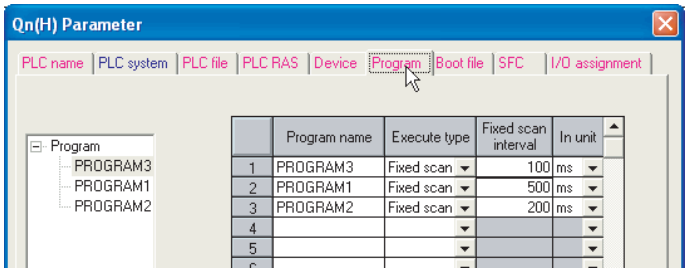


图 3.50 程序的设定

在将执行类型设定为 “恒定周期后”，设定恒定周期间隔与单位。
恒定周期间隔与单位的设定范围如下所示。

- 单位为 ms 的情况 :0.5~999.5ms (0.5ms)
- 单位为 s 的情况 :1~60s (1s 单位)

基本 注 3.33

3.3.6 执行类型的设定以及切换示例

(1) 执行类型的设定

下面对在执行多个程序情况下必要的程序设定进行说明。
程序的执行类型在 GX Developer 的可编程控制器参数内的程序设定中进行。
在 CPU 模块中，设定了执行类型的程序按照设定顺序执行。

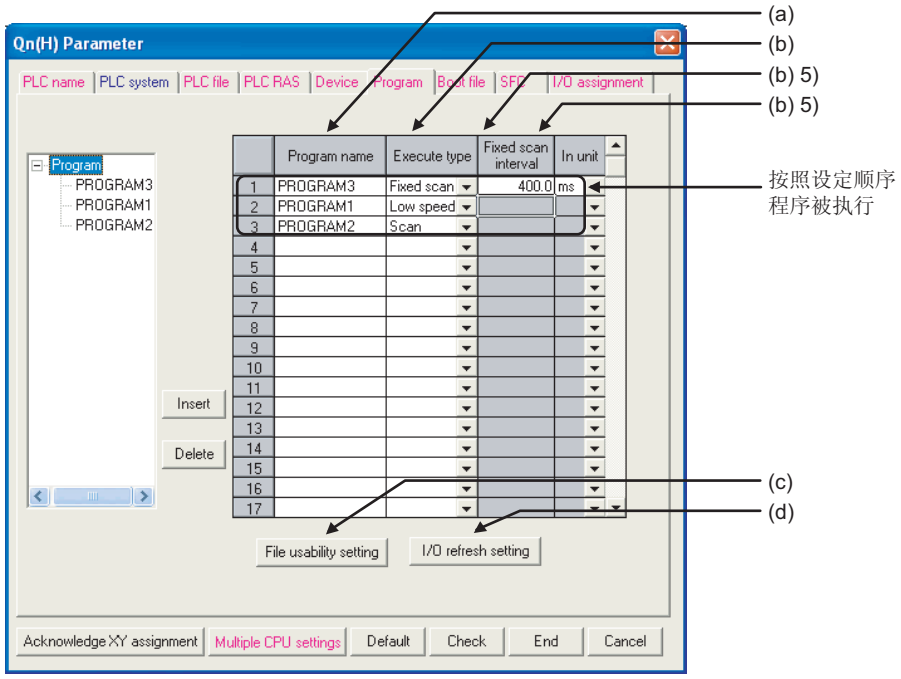


图 3.51 程序的设定

(a) 程序名

在 CPU 模块中设定执行程序的程序名（文件名）。

(b) 执行类型

选择已设定了程序名的文件的执行类型。

1) 初始执行类型（初始）

是指在电源处于由 OFF→ON 或者由 STOP 向 RUN 状态切换时，只被执行一次的程序。(☞ 3.3.1 项)

2) 扫描执行类型（扫描）

是指从执行初始执行类型程序后的下一个扫描周期开始，在一个扫描周期内执行一次的程序。(☞ 3.3.2 项)

冗余 注 3.34　通用 UD 注 3.34

3) 低速执行类型（低速）注 3.34

只在设定恒定扫描时或者设定低速类型程序执行时间时被执行的程序。
(☞ 3.3.3 项)

基本 注 3.33

在基本模式 QCPU 中，由于不能使用多个程序的原因，因此，不能使用执行类型的设定以及指令切换。

冗余 注 3.34　通用 UD 注 3.34

在冗余 CPU、通用型 QCPU 中，由于不能使用低速执行类型程序，因此不能将执行类型选择为“低速”。

4) 待机类型（待机）
是指只在有执行要求的情况才被执行的程序。
(☞ 3.3.4 项)

5) 恒定执行类型（恒定周期）
是指根据设定的 “恒定周期间隔” 与 “单位” 的时间执行的程序。
(☞ 3.3.5 项)

- 恒定周期间隔
  设定恒定周期执行类型程序的执行间隔。
  根据恒定周期间隔中设定的单位不同，其设定范围会有不同。
  - 单位为 ms 的情况：0.5~999.5ms(0.5ms 单位)
  - 单位为 s 的情况 ：1~60s(1s 单位)
- 单位
  选择恒定周期间隔的单位 (ms 或者 s)

通用 UD 注 3.35

6) 文件使用方法的设定*注 3.35*
可以在程序中设定，是否使用在可编程控制器参数的可编程控制器文件设定中所设定的文件寄存器、软元件初始值、软元件注释、局部软元件的文件。

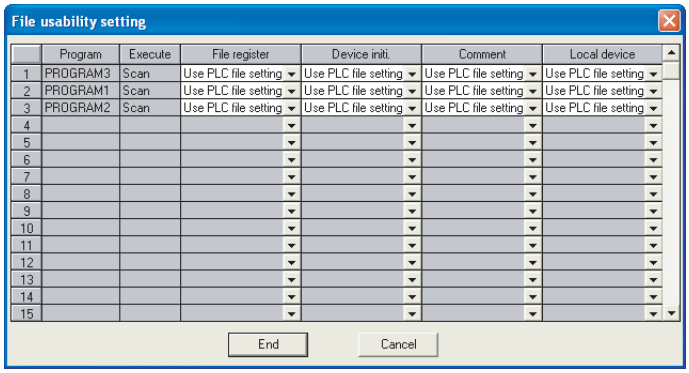


图 3.52 文件的使用方法的设定

默认值被设定为：“服从可编程控制器文件设定（使用与否遵从可编程控制器文件的设定）”。
如果将文件的使用方法设定为 “不使用” 的话，其结果如表 3.2 所示。

表 3.2 文件使用方法设定的项目

设定项目	选择 “不使用” 时的处理
文件寄存器	在程序中不能使用文件寄存器
软元件初始值	文件的软元件初始值与程序中的相同的情况下，不设置软元件初始值。
注释	在程序中，不能使用软元件注释
局部软元件	程序切换时，不能进行局部软元件的退避与返回。

通用 UD 注 3.35

在通用型 QCPU 中，不能使用文件时间方法设置。

(c) I/O 刷新设定

CPU 模块在批量 I/O 刷新中进行 I/O 模块以及智能功能模块的输入输出的更新。(☞ 3.8.1 项)

如果进行 I/O 刷新设定，在每个设定的程序内可以进行指定范围的 I/O 刷新。

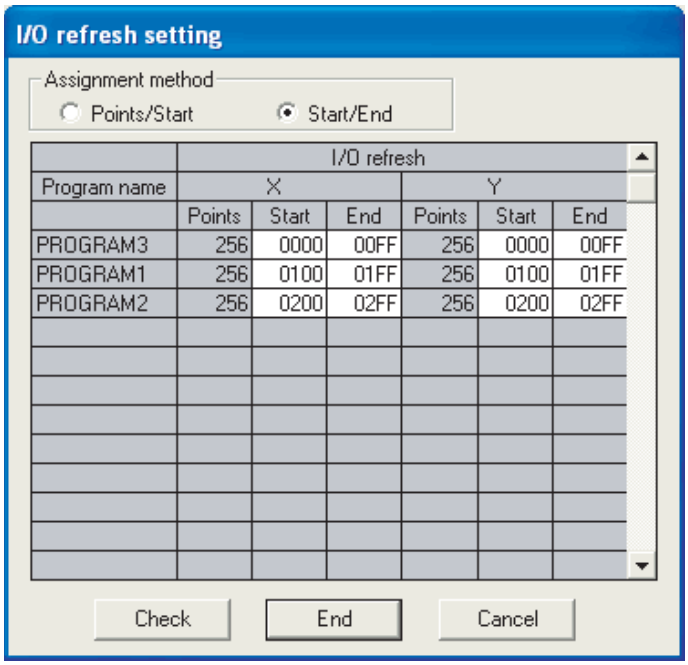


图 3.53 I/O 刷新设定

1) I/O 刷新设定的用途

读取恒定周期执行类型程序执行前所使用的输入 (X)，输出恒定周期执行类型程序中 ON/OFF 后的输出 (Y)。

要 点

执行程序（除恒定周期执行类型程序以外）的扫描时间在程序监视列表中可以进行确认。(☞ 6.13.1 项)

(2) CPU 模块的各程序的流程

可编程控制器的电源由 OFF→向 ON 或者 CPU 模块由 STOP→RUN 时的各个程序的流程如图 3.54 所示。

冗余 注 3.36　通用 注 3.36

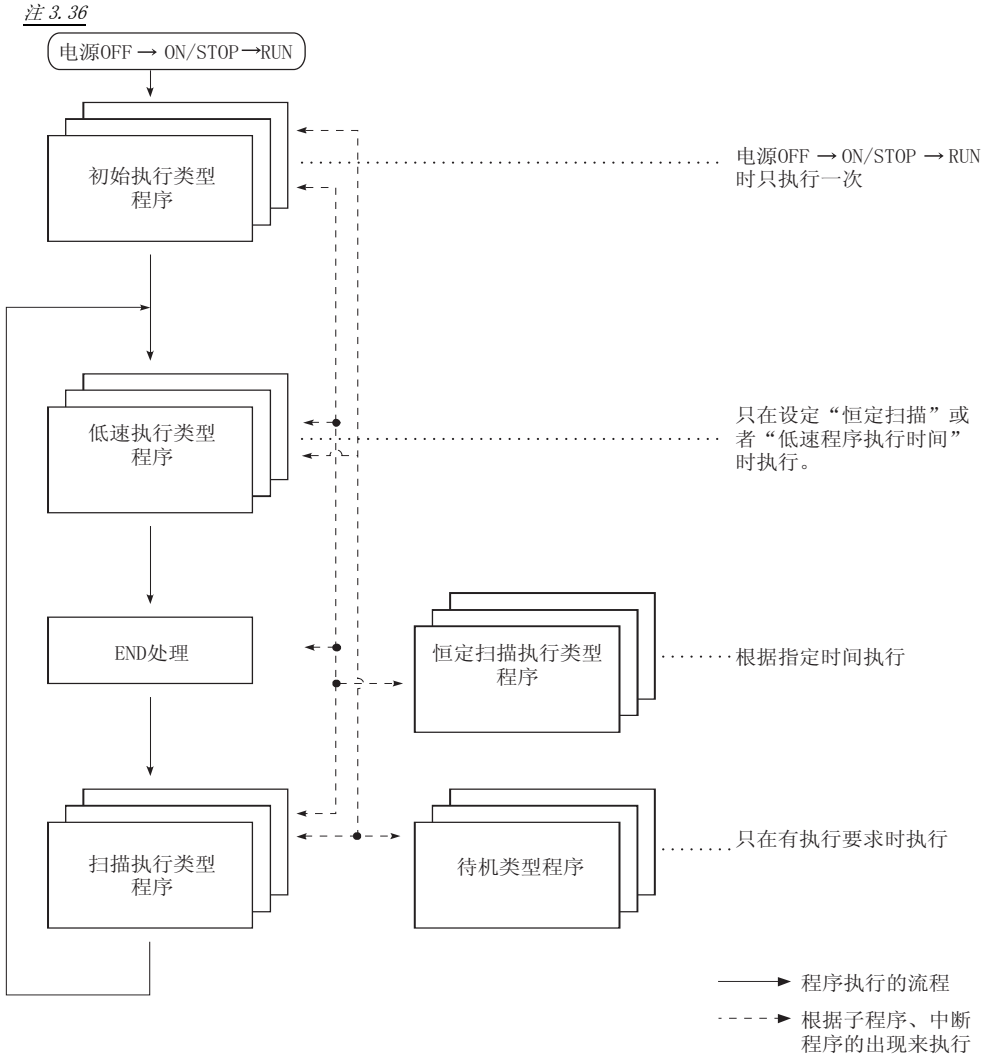


图 3.54 各程序的流程

冗余 注 3.36　通用 注 3.36

要 点

请根据需要使用初始执行类型程序、低速执行类型程序注 3.36，待机类型程序、恒定周期执行类型程序。

冗余 注 3.36　通用 注 3.36

在冗余 CPU、通用型 QCPU 中不能使用低速执行类型程序。

(3) 通过指令来切换执行类型的示例

(a) 执行类型切换的指令

执行类型根据使用的指令即使在顺控程序执行过程中也可以进行变更。

执行类型的变更通过 PSCAN 指令、PLOW 指令注 3.37、PSTOP 指令、POFF 指令进行。

冗余 注 3.37　通用 注 3.37

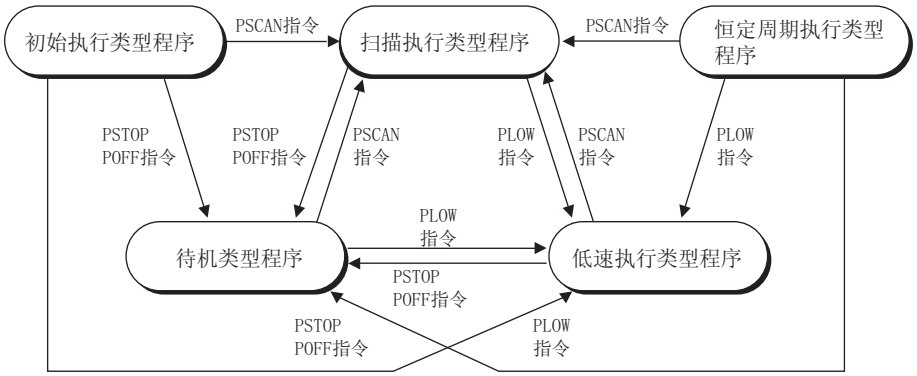


图 3.55 通过指令进行执行类型切换的模式

表 3.3 根据指令进行执行类型切换的时机

变更前的执行类型	执行指令			
	PSCAN	PSTOP	POFF	PLOW注 3.37
扫描执行类型	维持扫描执行类型状态不变。	变为待机类型	在下一个扫描周期输出 OFF。 再下一个扫描周期以后变为待机类型。	变为低速执行类型
初始执行类型	变为扫描执行类型			
待机类型		待机类型状态，不变化。	不进行处理	
低速执行类型注 3.37	中断低速执行类型的执行，从下一个扫描周期开始变为扫描类型。 （从步 0 开始执行）	中断低速执行类型的执行，在下一个扫描周期以后变为待机类型。	中断低速执行类型的执行，在下一个扫描周期输出 OFF。再下一个扫描周期之后变为待机类型。	低速执行类型状态，不变化
恒定周期执行类型	变为扫描执行类型	变为待机类型	在下一个扫描周期 OFF 输出。 再下一个扫描周期以后变为待机类型。	变为低速执行类型。

冗余 注 3.37　通用 注 3.37

在冗余 CPU、通用型 QCPU 中，由于不能使用低速执行类型程序，因此不能使用 PLOW 指令进行执行类型的切换。

要 点

如果将恒定周期执行类型程序变更为其它的执行类型后，将不能返回到恒定周期执行类型。

(b) 执行类型的切换示例

在管理程序中，将与设定的条件相符的待机类型程序变更为扫描执行类型程序后执行。

不使用的扫描执行类型程序也可以变更为待机类型程序。

在管理程序中切换 “ABC”、“DEF”、“GHI”、“JKL” 的待机类型程序的执行类型时的例子如图 3.56 所示。

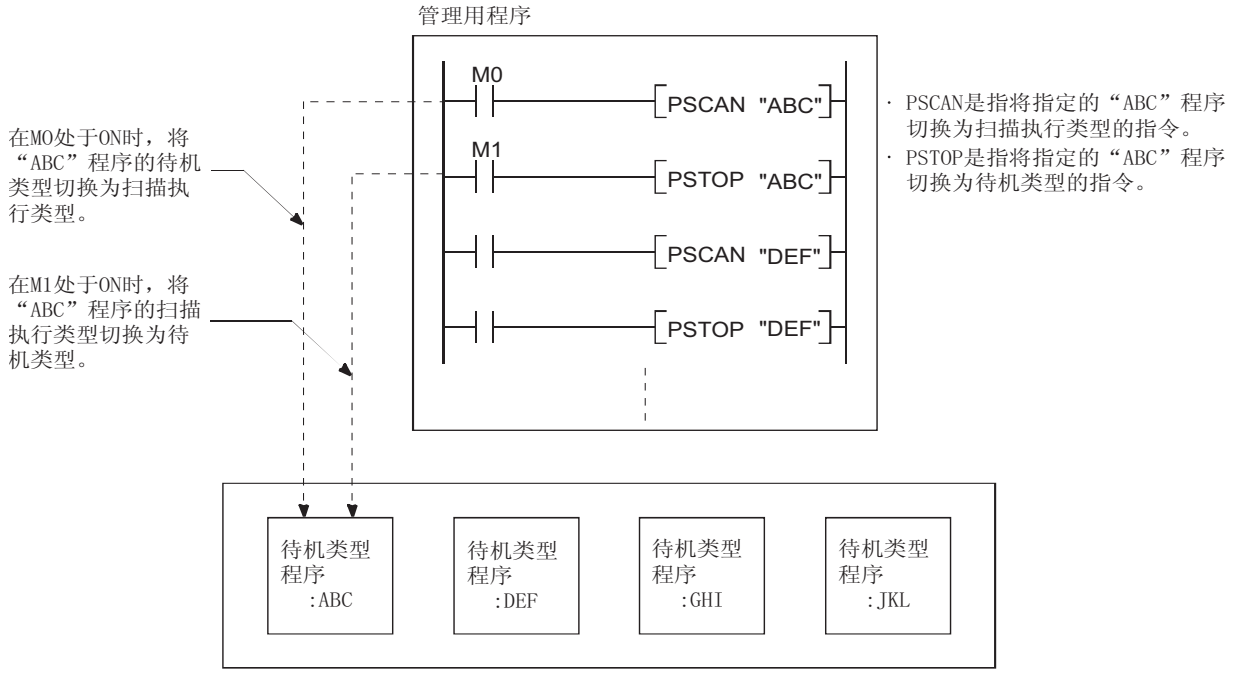


图 3.56 根据指令进行执行类型切换的示例

(4) 根据程序监视列表进行执行类型的切换

执行类型可以通过 GX Developer 的程序监视列表进行切换。

关于程序监视列表的切换，请参照 6.13.1 项 (3)。

3.4 运算处理

在 3.4 节中，将对 CPU 模块的运算处理进行说明。

3.4.1 初始化处理

初始化处理是指为执行顺控运算进行的前处理。
表 3.4 所示的 CPU 模块状态的情况下，只执行一次。
初始化处理完成后，CPU 模块将变为 RUN/STOP 开关（基本模式 QCPU、通用型 QCPU 时为 RUN/STOP/RESET 开关）中所设定的动作状态。（☞ 3.5 节）

表 3.4 初始化处理一览表

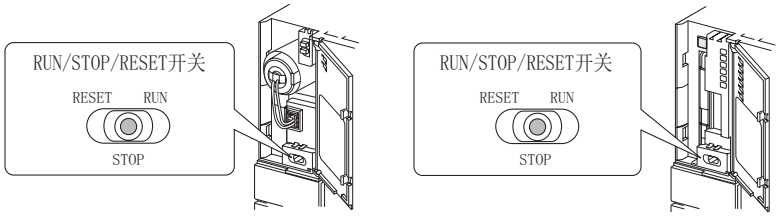
初始化处理项目	CPU 模块的状态		
	电源接入时	进行复位操作时	STOP 状态转为 RUN 状态时 *1
I/O 模块的初始化	○	○	×
来自于标准 ROM/ 存储卡的引导 *2*8	○	○	×
可编程控制器参数的检查	○	○	○
多 CPU 系统参数的一致性检查 *3	○	○	○
锁存范围外的软元件的初始化 *4 （位软元件：OFF；字软元件：0）	○	○	×
安装模块的 I/O 号的自动分配	○	○	○
MELSECNET/G*5*6 网络信息的设置	○	○	×
MELSECNET/H 网络信息的设置	○	○	×
智能功能模块的开关设定	○	○	×
CC-Link 信息的设置	○	○	×
Ethernet 信息的设置	○	○	×
软元件初始值的设置	○	○	○
串行口通讯功能的设定 *7	○	○	×

○：执行　×：不执行

*1: 表示在 STOP 状态下变更参数或者程序之后，未进行复位便变为 RUN 状态的情况。
（对 RUN/STOP 开关（基本模式 QCPU、通用型 QCPU 时为 RUN/STOP/RESET 开关）进行 STOP → RUN → (RUN LED 闪烁) → STOP → RUN 的操作）。
在上述操作中，脉冲化指令 (PLS、□P) 由于程序的内容变更（在 STOP 中的 RUN 中写入或者可编程控制器写入）而不能接续上次的信息，因此有时会发生不能正常动作的现象，请给予充分注意。
*2: 由于在基本模式 QCPU 中不能使用存储卡，因此不能进行由存储卡的引导。
*3: 在冗余 CPU 中由于不能构筑多 CPU 系统，因此不能进行多 CPU 系统参数的一致性检查。
*4: 在冗余 CPU 中，当启动模式为热启动模式时，不能进行锁存范围外的软元件的初始化。
（除步进继电器以及变址继电器等部分软元件以外。）
*5: 在高性能模式 QCPU 中使用 MELSECNET/G 时，应确认 CPU 模块及 GX Developer 的版本。
（☞ 附录 4.2)
*6: 在基本模式 QCPU、过程 CPU、冗余 CPU 中不能使用 MELSECNET/G。
*7: 在高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 中不能使用串行通讯功能。
*8: 在通用型 QCPU 中，不能进行由标准 ROM 的引导。（☞ 5.2.3 项）

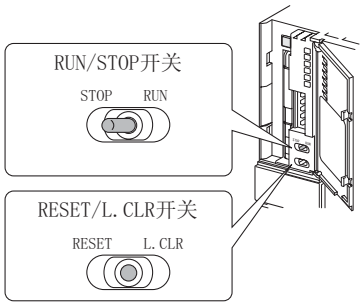
要 点

1. CPU 模块中的 STOP 开关、RUN 开关、RESET 开关在各个 CPU 模块中有所不同。
   • 基本模式 QCPU、通用型 QCPU 的开关



基本模式QCPU　　通用型QCPU

   • 高性能模式 QCPU、过程 CPU、冗余 CPU 的开关



2. 在 STOP 状态中变更参数或者程序的情况下，请通过 RESET/L.CLR 开关（基本模式 QCPU、通用型 QCPU 时为 RUN/STOP/RESET）进行复位。

3.4.2 I/O 刷新（I/O 模块的刷新处理）

在 I/O 刷新中，如果从 I/O 模块 / 智能功能模块向 CPU 模块进行 ON/OFF 数据输入，那么也进行从 CPU 模块向 I/O 模块 / 智能功能模块的 ON/OFF 数据输出。

(1) I/O 刷新的时机

I/O 刷新在顺控程序的运算开始之前被执行。
在执行恒定扫描时，在恒定扫描的等待时间结束后，进行 I/O 刷新。
（I/O 刷新在各个恒定扫描时间内被执行）

3.4.3 智能功能模块的自动刷新

在进行智能功能模块的自动刷新设定时，进行向设定的智能功能模块的数据的存取。
关于智能功能模块的自动刷新设定的情况，请参照所使用的智能功能模块的手册。

3.4.4 END 处理

END 处理是指结束一个扫描周期的顺控程序运算处理，使顺控程序的执行返回到步 0 所进行的后处理。

(1) END 处理的内容

END 处理有如下的处理方法。

表 3.5 END 处理一览表

END 处理项目	进行 END 处理的 CPU 模块					参考
	基本模式 QCPU	高性能模式 QCPU	过程 CPU	冗余 CPU	通用型 QCPU	
MELSECNET/G 的刷新	×	○*1	×	×	○	--
MELSECNET/H、CC-Link 的刷新	○	○	○	○	○	--
智能功能模块的自动刷新	○	○	○	○	○	7.1.1 项
自诊断处理	○	○	○	○	○	6.17 节
与 GX Developer 等外部设备进行的通讯处理	○	○	○	○	○	--
智能功能模块专用指令的处理	○	○	○	○	○	--
通过采样追踪功能进行软元件内容的采集（在各个扫描 (END 指令执行后）中设定了追踪点时）	×	○	○	○	○	6.14 节
CPU 共享内存的自动刷新	○	○	○	×	○	QCPU 用户手册（多 CPU 系统篇）
看门狗定时器的复位	○	○	○	○	○	6.16 节
对特殊继电器 / 特殊寄存器值的设置（设置的时机为 END 处理时的时机）	○	○	○	○	○	附录 1 附录 2

○：执行　×：不执行

*1: 在高性能模式 QCPU 中使用 MELSECNET/G 时，应确认 CPU 模块及 GX Developer 的版本。
（☞ 附录 4.2）

要 点

1. 在设定恒定扫描功能（☞6.2 节）时，在 END 处理后或者到下一个扫描周期开始的时间，保持 END 处理时间的结果。
2. 在执行低速执行类型程序的情况下，在结束所有的低速执行类型程序后执行低速 END 处理。注 3.38（☞3.3.3 项）

基本 注 3.38　冗余 注 3.38　通用 UD 注 3.38

基本 注 3.38　冗余 注 3.38　通用 UD 注 3.38

在基本模式 QCPU、冗余 CPU、通用型 QCPU 中不能使用低速执行类型程序，因此不进行低速 END 处理。

3.5 RUN 状态、STOP 状态、PAUSE 状态的运算处理

在 CPU 模块的动作状态中有 RUN 状态、STOP 状态、PAUSE 状态 3 种。
对在各个动作状态中 CPU 模块的运算处理进行说明。

(1) RUN 状态的运算处理

RUN 状态是指顺控程序的运算执行按照从步 0 → END(FEND) 指令 → 步 0 的顺序反复运算的状态。

(a) 进入 RUN 状态时的输出状态

进入 RUN 状态时，根据参数的 STOP → RUN 时的输出模式设定，输出 STOP 状态时退避的输出 (Y) 状态或输出一个扫描周期后的运算结果。(☞ 6.4 节)

(b) 到运算开始时的处理时间

由 STOP → RUN 开始一直到顺控程序运算开始的处理时间根据系统构成与参数设定内容会有变动。(通常状况下为 1~3 秒)

但是，根据条件的不同有延长的情况。

(2) STOP 状态的运算处理

STOP 状态是指，通过 RUN/STOP 开关 (基本模式 QCPU、通用型 QCPU 时为 RUN/STOP/RESET) 或者远程 STOP 功能 (☞ 6.6.1 项) 中止顺控程序运算的状态。
另外，在发生停止出错时也将变成 STOP 状态。

(a) 进入 STOP 状态时的输出状态

进入 STOP 状态时，退避输出 (Y) 状态，输出的所有点 OFF。输出 (Y) 以外的软元件内存将被保持。

(3) PAUSE 状态的运算处理

PAUSE 状态是指根据远程 PAUSE 功能 (☞ 6.6.2 项) 保持输出以及软元件内存的状态不变，终止顺控程序的运算的状态。

(4) 根据开关操作进行的 CPU 模块的运算处理

表 3.6 根据开关操作进行的运算处理

RUN/STOP 状态	CPU 模块的运算处理			
	顺控程序的运算处理	外部输出	软元件内存	
			M，L，S，T，C，D	Y
RUN→STOP	执行到 END 指令后停止	在变为 STOP 状态前，退避输出 (Y) 状态，OFF 所有点。	在变为 STOP 状态前，保持软元件内存的状态	在变为 STOP 状态前，退避输出 (Y) 状态，OFF 所有点。
STOP→RUN	从步 0 开始	根据可编程控制器参数的“STOP→RUN 时的输出模式”来决定。	在变为 STOP 状态前，保持软元件内存的状态。 但是，在设定了软元件初始值的情况下，需设置软元件初始值的值。 另外，清除本地软元件注 3.39	根据可编程控制器参数的“STOP→RUN 时的输出模式”来决定。(☞ 6.4 节)

基本 注 3.39

要 点

CPU 模块在 RUN 状态、STOP 状态、PAUSE 状态中的任何一个状态下均进行下述处理。

- I/O 模块的刷新处理
- 网络模块的刷新处理
- 智能功能模块的自动刷新处理
- 自诊断处理
- 与 GX Developer 等外部设备的通讯处理
- 智能功能模块专用指令的处理 (仅结束处理)
- 多 CPU 系统的多 CPU 间高速通信进行的运算处理 (仅通用型 QCPU)

因此，即使是在 STOP 状态、PAUSE 状态，也可执行以下动作。

- 通过 GX Developer 进行 I/O 监视及测试操作
- 通过使用了 MC 协议的外部设备进行读出 / 写入
- 通过 MELSECNET/G 注 3.40、注 3.41 、MELSECNET/H 与其它站进行通讯
- 与 CC-Link 的远程站进行通讯

高性能 注 3.40

基本 注 3.41　过程 注 3.41　冗余 注 3.41

基本 注 3.39

在基本模块 QCPU 中，由于不能执行多个程序的原因，因此，局部软元件与全局软元件没有区别。

高性能 注 3.40

在高性能模式 QCPU 中使用 MELSECNET/G 时，应确认 CPU 模块及 GX Developer 的版本。
(☞ 附录 4.2)

基本 注 3.41　过程 注 3.41　冗余 注 3.41

在基本模式 QCPU、过程 CPU、冗余 CPU 中不能使用 MELSECNET/G。

3.6 瞬间掉电时的运算处理

CPU 模块在供给电源模块的输入电源电压低于规定范围时将检测出瞬间掉电，并进行下述的运算处理。

(1) 发生瞬间掉电允许的时间以内的瞬间掉电的情况
在发生瞬间掉电时，保持输出状态，在登陆出错历史记录后中断运算处理。
（继续进行定时器软元件的测量）

(a) 在有 SFC 程序的继续运行开始指定的情况
在有 SFC 程序的继续运行开始指定的情况下，进行系统的退避处理。

(b) 瞬间掉电被解除的情况
在瞬间掉电被解除的情况下，继续进行运算处理。

(c) 发生瞬间掉电时的看门狗定时器 (WDT) 的测量
发生瞬间掉电时，即使中断运算，也将继续进行看门时钟 (WDT) 的测量。
例如：可编程控制器参数的 WDT 设定为 200ms、扫描时间设定为 190ms 时，如果发生 15ms 的瞬间掉电的话，将出现看门狗定时器溢出。

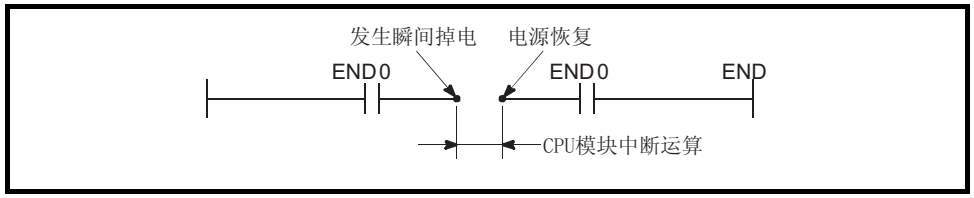


图 3.57 发生瞬间掉电时的运算处理

(2) 发生超出瞬间掉电允许时间的停电的情况
CPU 模块会出现初始化开始。
在进行下述操作时会出现同样的运算处理。
- 可编程控制器的电源接入。
- 通过 RESET/L.CLR 开关（基本模式 QCPU、通用型 QCPU 时为 RUN/STOP/RESET）进行复位操作。
- 通过 GX Developer 进行远程复位操作。

要 点

1. 在对电源模块进行冗余化时，虽然在一方电源模块发生瞬间掉电时也不中断运算，但若只对一方的电源模块供给电源的状态下发生瞬间掉电时，将中止运算。
2. 电源冗余系统中瞬间掉电的信息被存储在 SM1782 ～ 1783 以及 SD1782 ～ 1783 中。
在电源不是冗余状态时的瞬间掉电的信息被存储在 SM53 以及 SD53 中。
(☞ 附录 1、附录 2)

3.7 数据的清除处理

对 CPU 模块的数据清除处理方法、锁存数据清除相关的设定进行说明。

(1) 用数据的清除方法与不能清除的数据

CPU 模块通过 RESET/L.CLR 开关（使用基本模式 QCPU、通用型 QCPU 时为 RUN/STOP/RESET 开关）进行复位操作，通过电源的 ON→OFF→ON 进行电源复位等时，数据将被清除。

但是，下述 (a) 显示的数据在上述操作中不能被清除。

(a) 用复位操作不能清除的数据

- 程序内存的数据
- 标准 ROM 的数据
- 存储卡内的数据 基本 注 3.42 注 3.42
- 锁存指定的软元件的数据（☞本节 (2)）
- 文件寄存器的数据

(b) 用复位操作不能清除的数据的清除方法

说明上述 (a) 中所示的数据的清除方法

1) 程序内存的数据

用下述的任何一种方法进行清除。

- 在可编程控制器参数的引导文件设定中设定 “清除程序内存”。 基本 注 3.43 注 3.43
- 在 GX Developer 的可编程控制器数据清除画面中，清除程序内存。

2) 标准 ROM 的数据

在向标准 ROM 写入数据时，自动清除。

3) 存储卡内的数据 基本 注 3.42 注 3.42

在 GX Developer 的可编程控制器数据清除画面中，清除存储卡内的数据。

4) 锁存指定的软元件的数据

请参照（本节 (2)）

5) 文件寄存器的数据

通过 RST 指令复位或者 MOV/FMOV 指令传送 K0。

另外，在 GX Developer 的 “可编程控制器写入”→“可编程控制器存储清除” 中执行对所有文件寄存器的清除。

基本 注 3.42 在基本模式 QCPU 中不能使用存储卡。

基本 注 3.43 在基本模式 QCPU 中，在引导时不能清除程序内存。

(2) 软元件的锁存指定

软元件的锁存指定（锁存范围设定）在 GX Developer 的可编程控制器参数的软元件设定中进行。(☞ 6.3 节 (5))

对每个软元件进行设定。

(a) 锁存范围的设定

通过 GX Developer 进行锁存范围的设定有如下的 2 种类型。

1) 有效的锁存清除操作（锁存 (1) 起始 / 最终）

通过 RESET/L.CLR 开关*注 3.44* 或者通过远程锁存清除，在锁存清除操作中设定可以清除的锁存范围。

2) 无效的锁存清除操作（锁存 (2) 起始 / 最终）

通过 RESET/L.CLR 开关*注 3.44* 或者通过远程锁存清除，在锁存清除操作中设定不能清除的锁存范围。

(b) 设定为锁存清除操作有效的软元件数据的清除

- 通过 RESET/L.CLR 开关进行锁存清除操作*注 3.44*
- 通过来自 GX Developer 的远程锁存清除进行的清除操作。(☞ 6.6.4 项)

(c) 设定为锁存清除操作无效的软元件数据的清除

设定为锁存清除操作无效的软元件数据只能根据指令或者 GX Developer 来进行清除操作。

1) 通过指令的清除方法

通过 RST 指令进行复位或者通过 MOV/FMOV 指令传送 K0。

☞ QCPU(Q 模式)/QnACPU 编程手册。（公共指令篇）

2) 通过 GX Developer 进行的清除方法

在 “在线”→“可编程控制器储存清除”中，进行所有软元件内存（包括锁存）的清除。

在基本模式 QCPU、通用型 QCPU 中，不能通过开关操作进行锁存清除。

要 点

1. 冗余 CPU 的启动模式为热启动模式时，没有设定在锁存范围内的数据也将被保存。
（变址寄存器以及步进继电器等一部分软元件除外）
2. 在通用型 QCPU 中，如果对软元件进行锁存，扫描时间将延长。对软元件进行锁存时，应考虑扫描时间的延长时间。（☞ 10.1.2 项 (11)）

备 注

关于 GX Developer 的操作方法，请参照下述手册。
☞ GX Developer 操作手册。

3.8 I/O 处理与响应延迟

CPU 模块的 I/O 处理为刷新方式。
但是，在顺控程序中由于使用的是直接存取输出输入，因此在执行各个指令时可能进行直接方式的 I/O 处理。
下面对 CPU 模块的 I/O 处理方式与响应延迟进行说明。

(a) 刷新方式 (☞ 3.8.1 项)
刷新方式是指在顺控程序运算开始之前进行 I/O 模块的批量存取的方式。

(b) 直接方式 (☞ 3.8.2 项)
直接方式是指在顺控程序的各个指令执行时进行 I/O 模块的存取的方式。
通过直接方式进行 I/O 模块的存取时，顺控程序使用直接存取输入、或者直接存取输出。

(1) 刷新方式与直接存取方式的不同点。
直接存取方式在指令执行时，由于直接进行 I/O 模块的存取，因此，与刷新方式相比，输入的读取要快些。
但是，与刷新方式相比，直接存取方式的指令处理时间要长一些。
另外，只有安装在主基板以及扩展基板上的 I/O 模块、智能功能模块才能以直接方式进行输入输出的存取。
刷新方式与直接存取方式的不同点如表 3.7 所示。

表 3.7 刷新方式与直接存取方式的不同点一览表

项目	刷新方式	直接存取方式
安装在基板上的 I/O 模块	可以使用	可以使用
安装在基板上的智能功能模块的输入输出		
基本 注 3.45 安装在扩展基板上的 I/O 链接的输入输出 *注 3.45*		
MELSECNET/G 网络系统 *注 3.46* *注 3.47*	可以使用	可以使用
高性能 注 3.46 MELSECNET/H 网络系统 CC-Link 系统中使用的输入输出		

基本 注 3.47　过程 注 3.47　冗余 注 3.47

基本 注 3.45　在基本模式 QCPU 中，不能使用安装在扩展基板上的 I/O 链接的输入输出。

高性能 注 3.46　在高性能模式 QCPU 中使用 MELSCNET/G 时，请确认 CPU 模块以及 GX Developer 的版本。
(☞ 附录 4.2)

基本 注 3.47　过程 注 3.47　冗余 注 3.47　在基本模式 QCPU、过程 CPU、冗余 CPU 中，不能使用 MELSCNET/G。

3.8.1 刷新方式

(1) 关于刷新方式

刷新方式是指在顺控程序运算开始之前进行 I/O 模块的批量存取的方式。

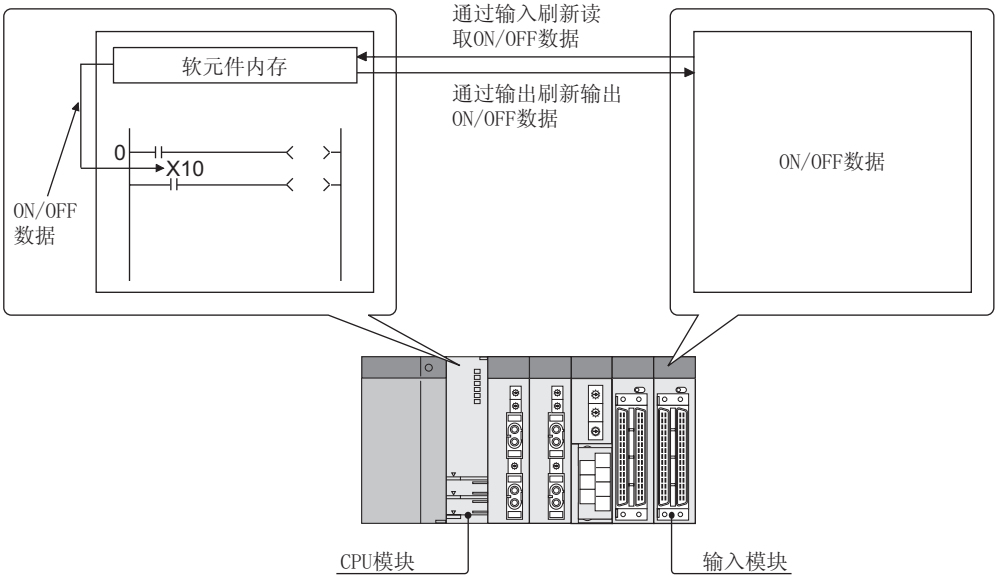


图 3.58 刷新方式

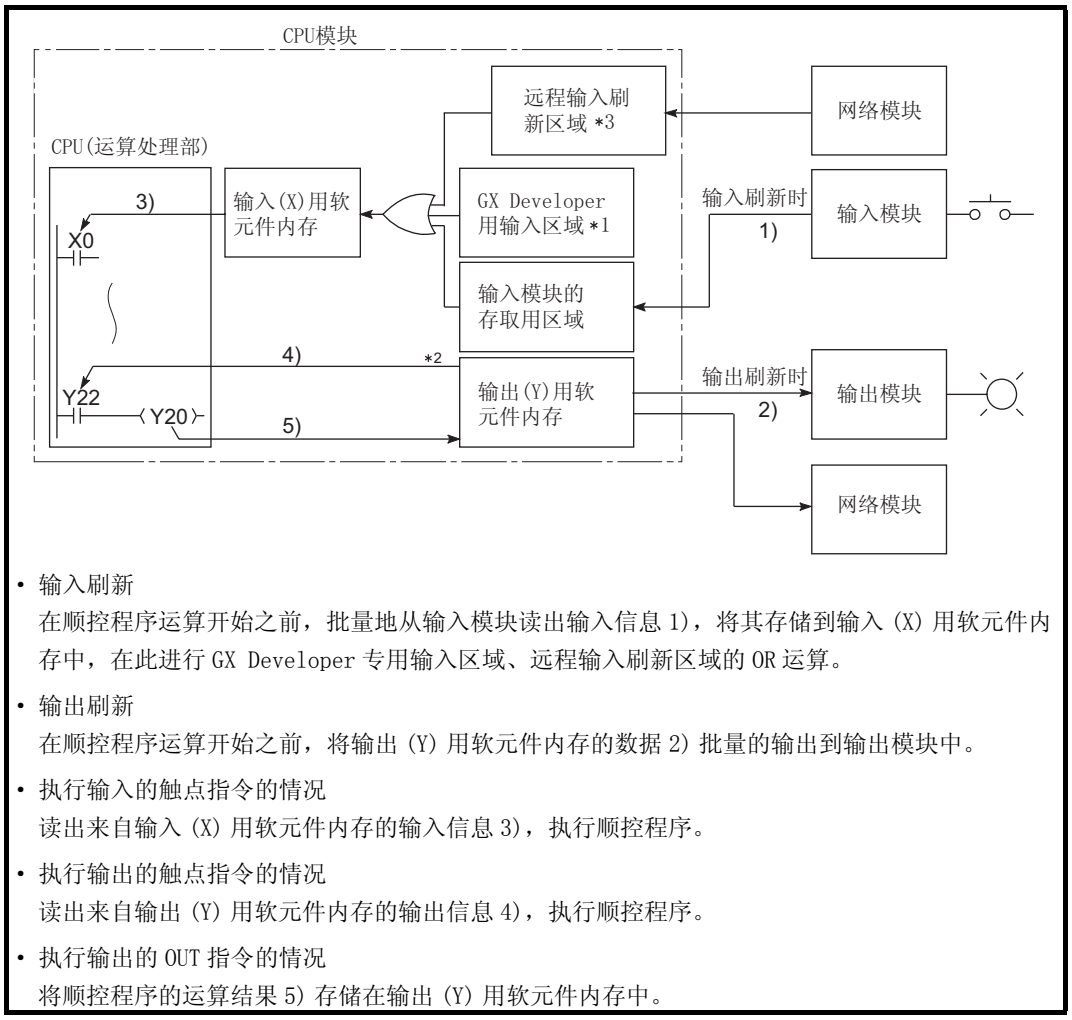
(2) 关于输入

输入模块的 ON/OFF 信息，在顺控程序的运算开始之前，批量地读取到 CPU 模块内部的输入模块的通讯用区域。

在执行顺控程序时，使用输入 (X) 用软元件内存的 ON/OFF 数据进行运算。

(3) 关于输出

输出 (Y) 的顺控程序的运算结果每次都输出到 CPU 模块内的输出 (Y) 用软元件内存中，在顺控程序的运算开始之前，与输出 (Y) 用软元件内存的 ON/OFF 数据汇总后输出到输出模块中。



- 输入刷新

  在顺控程序运算开始之前，批量地从输入模块读出输入信息 1)，将其存储到输入 (X) 用软元件内存中，在此进行 GX Developer 专用输入区域、远程输入刷新区域的 OR 运算。

- 输出刷新

  在顺控程序运算开始之前，将输出 (Y) 用软元件内存的数据 2) 批量的输出到输出模块中。

- 执行输入的触点指令的情况

  读出来自输入 (X) 用软元件内存的输入信息 3)，执行顺控程序。

- 执行输出的触点指令的情况

  读出来自输出 (Y) 用软元件内存的输出信息 4)，执行顺控程序。

- 执行输出的 OUT 指令的情况

  将顺控程序的运算结果 5) 存储在输出 (Y) 用软元件内存中。

图 3.59 刷新方式的输入 / 输出信息的流程

*1: 如下情况可以 ON/OFF GX Developer 专用输入区域:
- 通过 GX Developer 进行的测试操作
- 来自网络模块的写入

等等

*2: 如下情况可以 ON/OFF 软元件内存输出 (Y):
- 通过 GX Developer 进行的测试操作
- MELSECNET/G 网络系统的链接刷新 *注 3.48*、*注 3.49*
- MELSECNET/H 网络系统的链接刷新
- 使用 MC 协议从外部设备写入
- CC-Link 的自动刷新

等等

高性能 注 3.48

基本 注 3.49 过程 注 3.49 冗余 注 3.49

*3: 远程输入刷新区域表示在 MELSECNET/G *注 3.48*、*注 3.49*、MELSECNET/H、CC-Link 中对输入 (X) 进行了自动刷新设定时的区域。
远程输入刷新区域的自动刷新在 END 处理时进行。

高性能 注 3.48

在高性能模式 QCPU 中使用 MELSCNET/G 时，请确认 CPU 模块以及 GX Developer 的版本。
(☞ 附录 4.2)

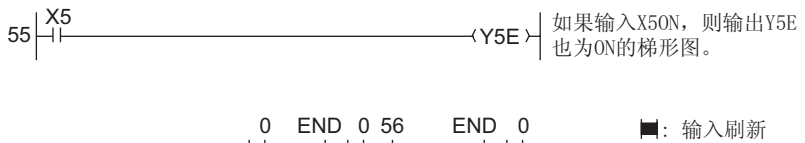
基本 注 3.49 过程 注 3.49 冗余 注 3.49

在基本模式 QCPU、过程 CPU、冗余 CPU 中，不能使用 MELSCNET/G。

(4) 响应延迟

与输入模块的变化相对应的输出变化，根据外部触点 ON 的时机，最大延后 2 个扫描周期。

梯形图示例



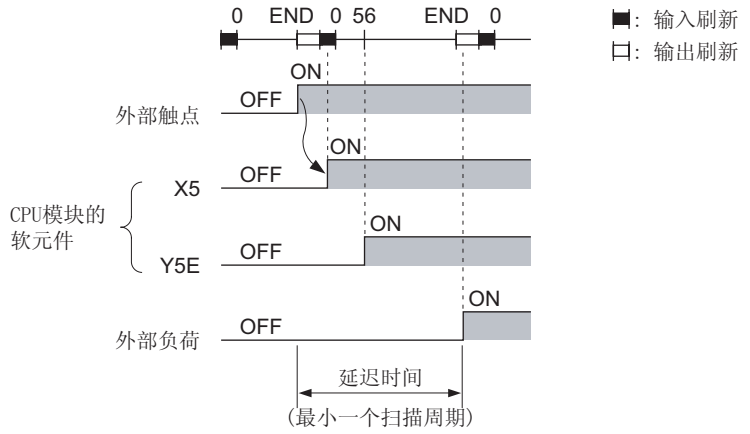


图 3.60 Y5E 最快 ON 时的情况

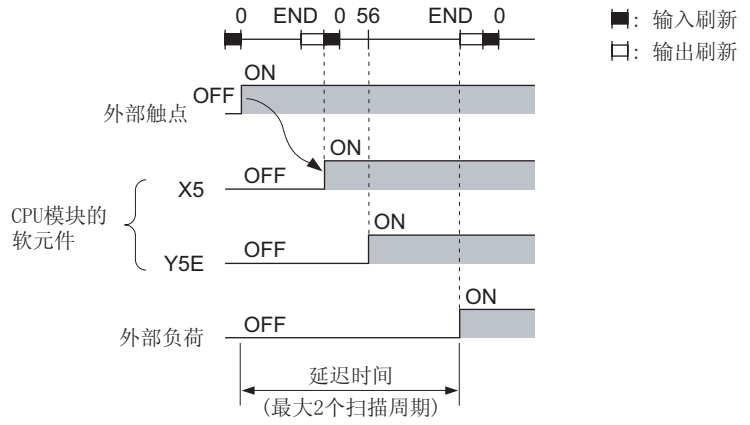


图 3.61 59Y5E 最迟 ON 时的情况

3.8.2 直接方式

(1) 关于直接方式

直接方式是指在执行顺控程序的各个指令时进行 I/O 模块存取的方式。

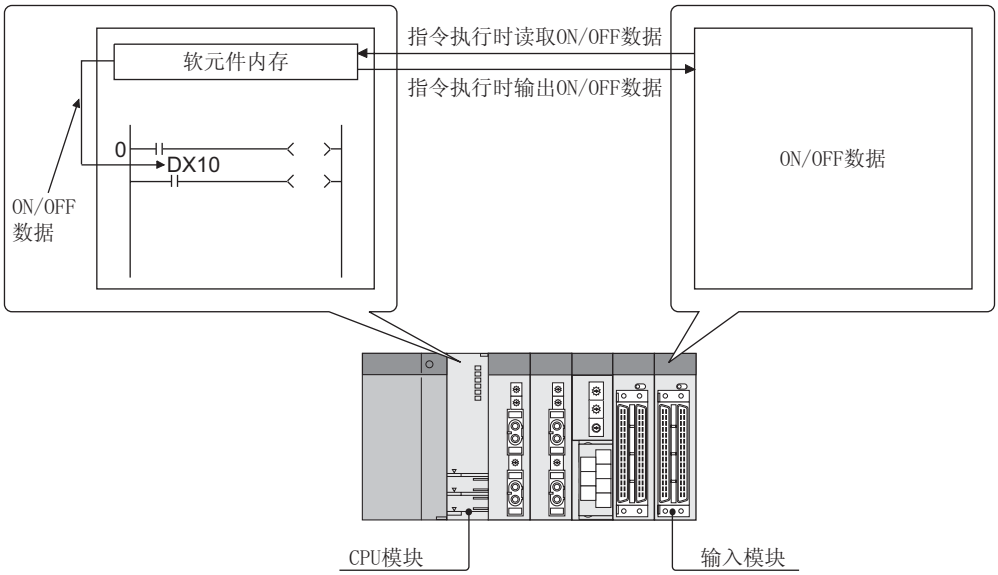
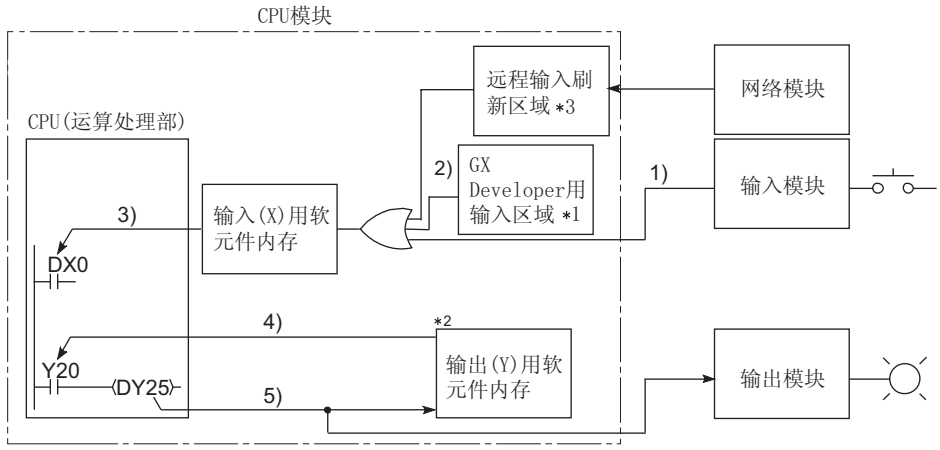


图 3.62 直接方式

在 CPU 模块中，使用直接存取输入 (DX)、直接存取输出 (DY) 进行直接方式的 I/O 处理。



- 执行输入的触点指令的情况
  进行 I/O 模块的输入信息 1)、GX Developer 用输入区域的输入信息 2)、远程输入刷新区域的 OR 运算。
  将其结果存储在输入 (X) 用软元件内存中，并作为输入信息 3) 执行顺控程序。
- 执行输出的触点指令的情况
  读出来自输出 (Y) 用软元件内存的输出信息 4)，执行顺控程序。
- 执行输出的 OUT 指令的情况
  将顺控程序的运算结果 5) 输出到输出模块的同时，存储到输出 (Y) 用软元件内存中。

图 3.63 直接方式的输入 / 输出信息的流程

*1: 如下情况可以 ON/OFF GX Developer 用输入区域：
- 通过 GX Developer 进行的测试操作
- 来自网络模块的写入

等等

*2: 以下情况下可以对输出 (Y) 用软元件内存进行 ON/OFF：
- 通过 GX Developer 进行的测试操作
- MELSECNET/G 网络系统的链接刷新 *注 3.50*、*注 3.51*
- MELSECNET/H 网络系统的链接刷新
- 通过使用了 MC 协议的外部设备进行写入
- CC-Link 的自动刷新

等等

高性能 注 3.50

基本 过程 冗余 注 3.51 注 3.51 注 3.51

*3: 远程输入刷新区域表示在通过 MELSECNET/G *注 3.50*、*注 3.51*、MELSECNET/H、CC-Link 对输入 (X) 进行了自动刷新设定时的区域。
远程输入刷新区域的自动刷新在 END 处理时进行。

高性能 注 3.50

在高性能模式 QCPU 中使用 MELSCNET/G 时，请确认 CPU 模块以及 GX Developer 的版本。
(☞ 附录 4.2)

基本 过程 冗余 注 3.51 注 3.51 注 3.51

在基本模式 QCPU、过程 CPU、冗余 CPU 中，不能使用 MELSCNET/G。

(2) 响应延迟

与 I/O 模块的的变化相对应的输出变化根据外部触点 ON 时的时机，最大延迟时间为一个扫描周期。

梯形图示例

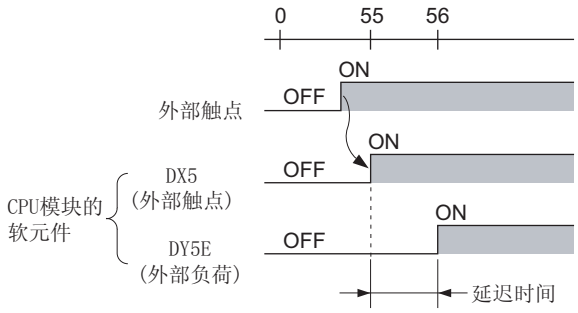


图 3.64 DY5E 最快 ON 时的情况

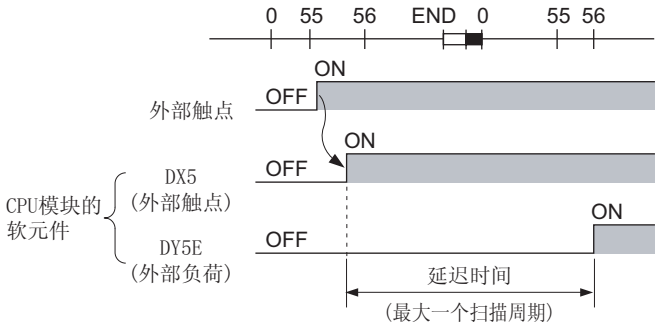


图 3.65 DY5E 最迟 ON 时的情况

3.9 在顺控程序中可以使用的数值

在 CPU 模块中，数值、字母等数据表示为 0(OFF) 与 1(ON) 两种状态。
用 0 与 1 表示的数据称作 BIN(2 进制数)。
在 CPU 模块中，可以使用将 BIN 数据按每 4 位为一组表示的 HEX(16 进制数)、BCD(2 阶 10 进制数)
另外，可以使用实数。(☞ 3.9.4 项)
BIN，HEX，BCD，DEC(10 进制数) 的数值表示如表 3.8 所示。

表 3.8 BIN，HEX，BCD，DEC 的数值表示

DEC(10 进制数)	HEX(16 进制数)	BIN(2 进制数)				BCD(2 阶 10 进制数)			
0	0				0				0
1	1				1				1
2	2				10				10
3	3				11				11
•	•				•				•
•	•				•				•
•	•				•				•
9	9				1001				1001
10	A				1010			1	0000
11	B				1011			1	0001
12	C				1100			1	0010
13	D				1101			1	0011
14	E				1110			1	0100
15	F				1111			1	0101
16	10			1	0000			1	0110
17	11			1	0001			1	0111
•	•				•				•
•	•				•				•
•	•				•				•
47	2F			10	1111			100	0111
•	•								
•	•								
•	•								
32766	7FFE	0111	1111	1111	1110			–	
32767	7FFF	0111	1111	1111	1111			–	
-32768	8000	1000	0000	0000	0000	1000	0000	0000	0000
-32767	8001	1000	0000	0000	0001	1000	0000	0000	0001
•	•								
•	•								
•	•								
-2	FFFE	1111	1111	1111	1110			–	
-1	FFFF	1111	1111	1111	1111			–	

(1) 从外部设备对 CPU 模块进行数据输入

从外部通过数字开关等设定 CPU 模块数值的情况下，通过下述 (b) 表示的方法，可以对 BCD(2 阶 10 进制数) 与 DEC(10 进制数) 进行同样的设定。

(a) 在 CPU 模块内部中使用的数据

在 CPU 模块中，用 BIN(2 进制数) 进行运算。

如果直接使用 BCD 设定的值，CPU 模块将把设定的值作为 BIN 进行运算。

因此，将会导致以与设定值不同的值进行运算。(☞如下 (b))

(b) 对 BIN 无干扰的数值输入的方法

为了使 BCD 中设定的数据在 CPU 模块中变更为所使用的 BIN，需使用 BIN 指令。

如果使用 BIN 指令，来自外部的数值数据设定将在不干扰 BIN 的情况下进行。

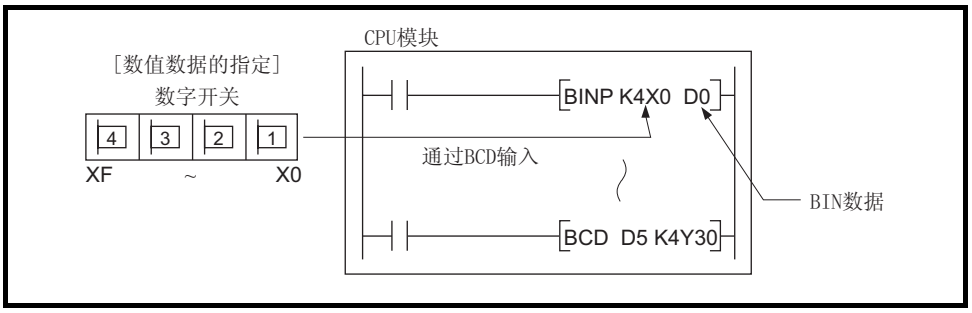


图 3.66 从数字开关向 CPU 模块的数据导入

备 注

关于 BIN 指令的详细情况，请参照下述手册。

☞ QCPU(Q 模式)/QnACPU 编程手册 (公共指令篇)

(2) 从 CPU 模块向外部进行的数值输出

将 CPU 模块中运算的数值向外部显示时，可以使用数字显示器等。

(a) 数值的输出方法

CPU 模块通过 BIN 进行运算。

CPU 模块即使将使用的 BIN 直接由数字显示器输出，也不能进行正常的显示。

经过 BIN 运算的数据为了变更为在外部显示器等中使用的 BCD，需使用 BCD 指令。

如果使用 BCD 指令，在外部显示器等中可以进行与 DCE(10 进制数) 同样的显示。

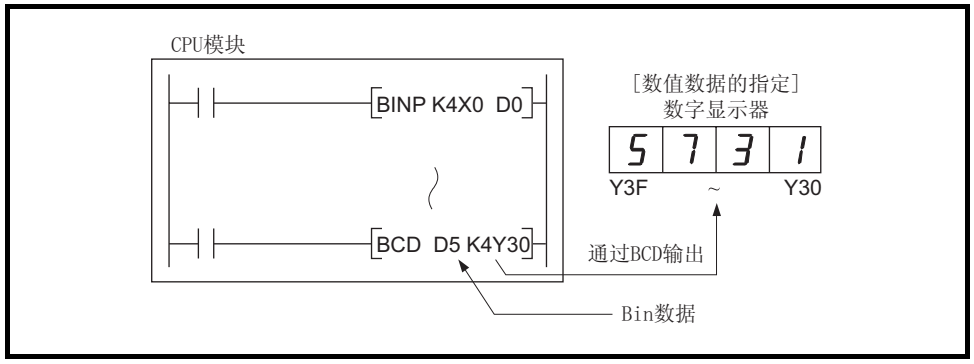


图 3.67 通过数字显示器显示的 CPU 模块的运算数据

备 注

关于 BCD 指令的详细情况，请参照下述手册。

☞ QCPU(Q 模式)/QnACPU 编程手册 (公共指令篇)

3.9.1 BIN(2 进制数：Binary Code)

(1) 2 进制数

BIN(2 进制数) 是指通过 0(OFF) 与 1(ON) 表示的数值。
DEC(10 进制数) 从 0 增到 9，再增加将进位为 10。
在 BIN 中，在 0、1 之后进位，将变为 10(10 进制数的 2)。
BIN 与 DEC 数值表示如表 3.9 所示。

表 3.9 BIN 与 DEC 数值表示的区别

DEC(10 进制数)	BIN(2 进制数)	
0	0000	
1	0001	
2	0010	进位
3	0011	
4	0100	进位
5	0101	
6	0110	
7	0111	
8	1000	进位
9	1001	
10	1010	
11	1011	

(2) BIN 的数值表示

(a) 在 CPU 模块中使用的 BIN 的位构成
CPU 模块的各个寄存器 (数据寄存器、链接寄存器等) 由 16 位构成。

(b) CPU 模块中可以使用的数值数据
在 CPU 模块中的各个寄存器可以存储 -32768~32767 的数值。
CPU 模块中的各个寄存器的数值表示如图 3.68 所示。

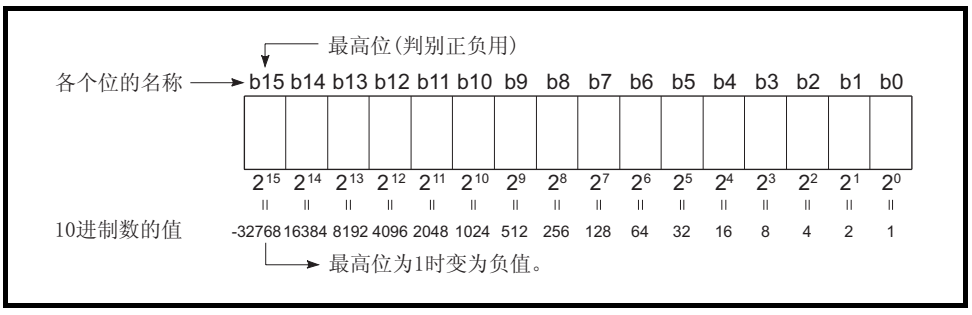


图 3.68 CPU 模块的各个寄存器的数值表示

要 点

各个寄存器的各个位被分配成 2^n 的数值。
但是，由于最高位的位是用来判别正负的，因此不能使用没有符号（无符号）的BIN(0 至 65535)。
1) 最高位为 0 时 .... 正
2) 最高位为 1 时 .... 负

3.9.2 HEX(16 进制数：Hexadeciamal)

(1) HEX

HEX(16 进制数）以 BIN 的 4 位作为一个位数来表示的。
BIN(2 进制数）如果使用 4 位，可以进行 0~15 的 16 种表示。
在 HEX 中，为了将 0~15 表示为一个位数，在 9 后面的 10 用 AH，11 用 BH 等字母表示，在 FH(15) 之后进位。
BIN、HEX、DEC(10 进制数）的数值表示如表 3.10 所示。

表 3.10 BIN，HEX，DEC 的数值表示

DEC(10 进制数）	HEX(16 进制数）	BIN(2 进制数）	
0	0		0
1	1		1
2	2		10
3	3		11
•	•		•
•	•		•
•	•		•
9	9		1001
10	A		1010
11	B		1011
12	C		1100
13	D		1101
14	E		1110
15	F		1111
16	10	1	0000
17	11	1	0001
•	•		•
•	•		•
•	•		•
47	2F	10	1111

进位

(2) HEX 的数值表示

CPU 模块的各个寄存器（数据寄存器、链接寄存器等）由 16 位构成。
在 16 位的情况下，在 16 位进制数中可以指定 0~FFFFH。

3.9.3 BCD(2 阶 10 进制数：Binary Coded Decimal)

(1) BCD

BCD(2 阶 10 进制数）是指将 DEC(10 进制数）的 1 位数通过 BIN(2 进制数）表示。
虽然与 HEX(16 进制数）同样可以表示为 4 位，但是不使用 HEX 的 A~F。
BIN、BCD、DEC 数值表示如表 3.11 所示。

表 3.11 BIN，BCD，DEC 的数值表示

10 进制数	BIN(2 进制数）	BCD(2 进位 10 进数）	
0	0000		0
1	0001		1
2	0010		10
3	0011		11
4	0100		100
5	0101		101
6	0110		110
7	0111		111
8	1000		1000
9	1001		1001
10	1010	1	0000
11	1011	1	0001
12	1100	1	0010

进位

(2) BCD 的数值表示

CPU 模块的各个寄存器（数据寄存器、链接寄存器等）由 16 位构成。
因此，可以存储在各个寄存器中的数值在 BCD 中可以在 0~9999 的范围内表示。

3.9.4 实数（浮动小数点数据）

实数数据中，有单精度浮动小数点数据及双精度浮动小数点数据。
在通用型 QCPU 中，可以使用双精度浮动小数点数据。

(1) 单精度浮动小数点数据

(a) 实数数据的内部表示

下面对在 CPU 模块中使用的实数数据的内部表示进行说明。
实数数据使用 2 个字元件，其表示如下。

[符号] 1.[虚数部分]×2[指数部分]

将数据在内部表示时的位构成与含义如下所示。

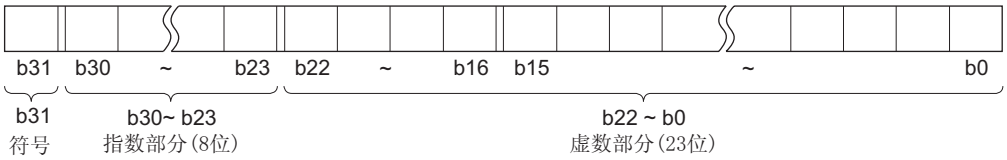


图 3.69 实数数据的位构成

1) 符号

b31 表示符号
0: 正
1: 负

2) 指数部分

通过 b23~b30 表示 2^n 中的 n。
b23~b30 的 BIN 值，n 成为如下所示的。

b23 ~ b30	FFH	FEH	FDH		81H	80H	7FH	7EH		02H	01H	00H
n	未使用	127	126		2	1	0	-1		-125	-126	未使用

图 3.70 指数部分存储值与指数的关系

3) 虚数部分

通过 b0~b22 的 23 位表示 2 进制数中 1.XXXXXX 中的 XXXXXX 的值。

(b) 计算示例

计算示例说明。(说明 (nnnnnn)X 在 X 进制数中表示的某些数据)

1) 存储了 10 时

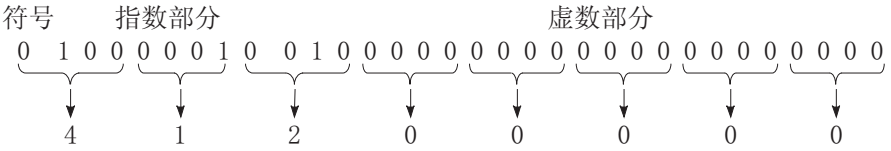
$$(10)_{10} \rightarrow (1010)_2 \rightarrow (1.010000..... \times 2^3)_2$$

符号　　正→0

指数部分　　3→82H→(10000010)2

虚数部分　　(010 00000 00000 00000 00000)2

由上可得



数据变为 41200000H。

2) 存储了 0.75 时

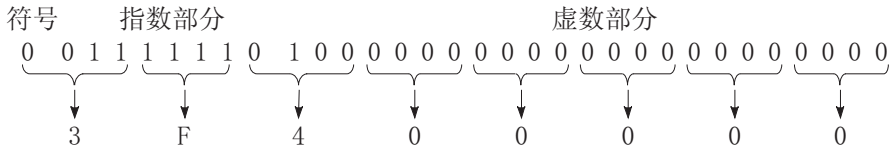
$$(0.75)_{10} \rightarrow (0.11)_2 \rightarrow (1.100..... \times 2^{-1})_2$$

符号　　正→0

指数部分　　-1→7EH →(01111110)2

虚数部分　　(100 00000 00000 00000 00000)2

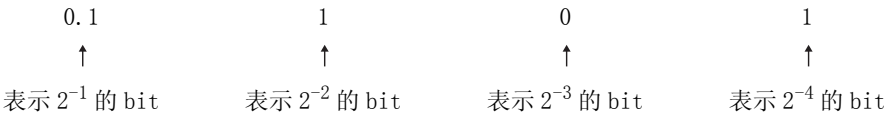
由上可得



数据变为 3F400000H。

要 点

在 2 进制数中小数点以下的值如下进行计算。



$(0.1101)_2=2^{-1}+2^{-2}+2^{-4}=0.5+0.25+0.0625=(0.8125)_{10}$

基本 过程 冗余
注 3.52 注 3.52 注 3.52

通用
注 3.52

(c) 进行双精度内部运算处理的方法与注意事项*注 3.52*

在高性能模块 QCPU 中，可以选择 “进行双精度内部运算处理”或者 “不进行双精度内部运算处理”。

1) 为双精度内部运算处理而进行的设定

设定在可编程控制器参数的可编程控制器系统设定的浮动小数点运算处理中进行。

（默认值：进行双精度内部运算处理）

不以双精度进行内部运算处理时，取消该勾选。

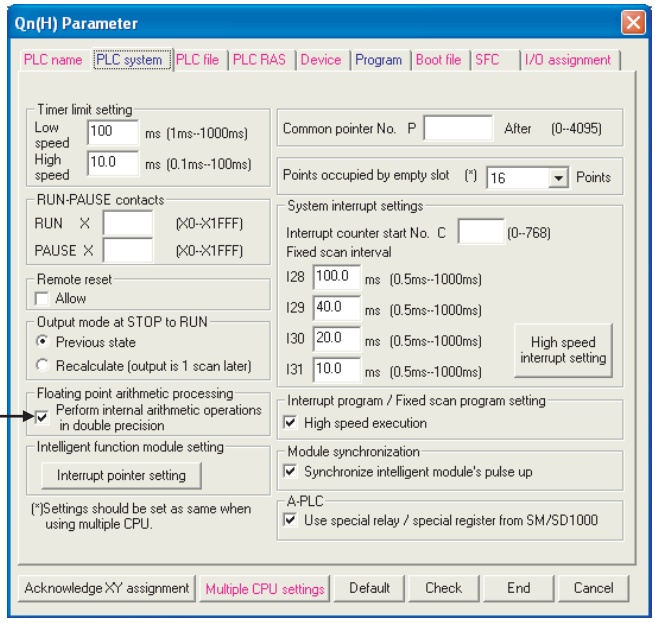


图 3.71 可编程控制器系统的设定画面

2) 关于运算结果

运算结果是与浮动小数点运算处理设定没有关系的单精度。

双倍精度的内部运算处理只在内部运算时进行双倍精度（64 位）。

基本 过程 冗余
注 3.52 注 3.52 注 3.52

通用
注 3.52

在基本模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 中，不能以双精度进行内部运算处理。

3) 用途与注意事项

对内部运算处理进行与不进行双倍精度时的用途以及注意事项进行说明。

- 对内部运算进行双倍精度的情况
  为与以前的机种具有兼容性而要求精度时使用。
  另外，象 SIN 指令、COS 指令那样在内部运算中使用多用指令进行实数运算时，通过对内部运算处理进行双倍精度，可以提高精度。

- 不对内部运算进行双倍精度的情况
  在有必要进行实数运算的高速化的情况下使用。
  内部运算单精度（32 位）虽然可以提高实数运算速度，但是，损失了一部分精度。

要 点

1. 通过 GX Developer 的监视功能可以对 CPU 模块的实数数据进行监视。
   但是，象 “$FFFF_H$”这样的数据虽然想对其进行监视，但在其不能表示为实数的情况下，将被表示为 “—”。
2. 表示 0 时，将 b0～b31 全部作为 0。

基本 高性能 过程
注 3.53 注 3.53 注 3.53
冗余
注 3.53

(2) 双精度浮动小数点数据注 3.53

(a) 实数数据的内部表示

在通用型 QCPU 中处理实数数据的内部表示如下所示。
实数数据使用 4 个字软元件按以下格式表示。

[符号]1.[虚数部分] × $2^{[\text{指数部分}]}$

对实数数据进行内部表示时的位构成及其含义如下所示。

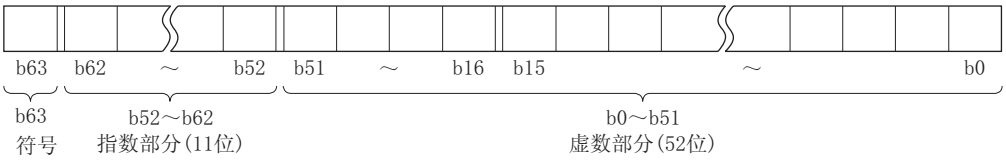


图 3.72 实数数据的位构成

1) 符号
b63 表示符号。
0：正
1：负

2) 指数部分
通过 b52 ～ b62 表示 2^n 中的 n。
根据 b52 ～ b62 的 BIN 值，n 变为如下所示的值。

b52～b62	7FFH	7FEH	7FDH			400H	3FFH	3FEH	3FDH	3FCH			02H	01H	00H
n	未使用	1023	1022			2	1	0	-1	-2			-1021	-1022	未使用

图 3.73 指数部分存储的值与指数的关系

3) 虚数部分
通过 b0 ～ b51 的 52 位表示 2 进制数的 1.XXXXXX ..... 中的 XXXXXX.... 的值。

基本 高性能 过程
注 3.53 注 3.53 注 3.53
冗余
注 3.53

在基本模式 QCPU、高性能模式 QCPU、过程 CPU、冗余 CPU 中，不能使用双精度浮动小数点数据。

(b) 计算示例

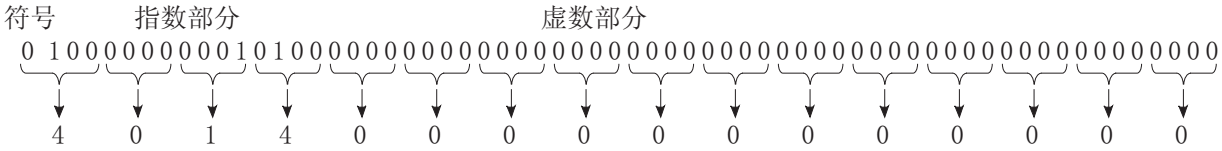
计算示例如下所示。((nnnnnn)X 表示以 X 进制数表示的数据。)

1) 存储了 10 时

$$(10)_{10} \rightarrow (1010)_2 \rightarrow (1.010000..... \times 2^3)_2$$

符号　　正→ 0
指数部分　3 → 401H → (100 0000 0001)2
虚数部分　(0100 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000)2

由上可得



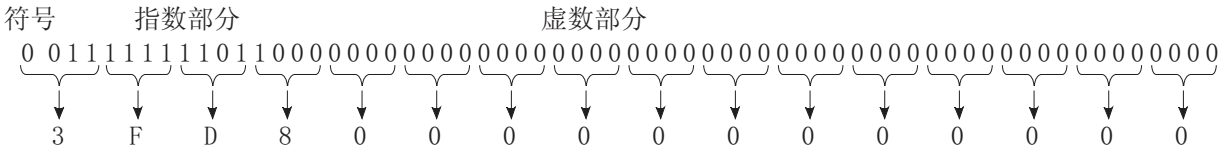
数据变为 4014000000000000H。

2) 存储了 0.75 时

$$(0.75)_{10} \rightarrow (0.11)_2 \rightarrow (1.100..... \times 2^{-1})_2$$

符号　　正→ 0
指数部分　-1 → 3FDH → (11 1111 1101)2
虚数部分　(1000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000)2

由上可得



数据变为 3FD8000000000000H。

要 点

对 2 进制数中小数点以下的值按以下方式计算。

0.1	1	0	1
↑	↑	↑	↑
表示 2^{-1} 的 bit	表示 2^{-2} 的 bit	表示 2^{-3} 的 bit	表示 2^{-4} 的 bit

$(0.1101)_2 = 2^{-1} + 2^{-2} + 2^{-4} = 0.5 + 0.25 + 0.0625 = (0.8125)_{10}$

3.10 字符串数据

基本 注 3.54

(1) 字符串数据*注 3.54*

在 CPU 模块中使用的字符串为 JIS58 代码的字符串。

(2) JIS58 代码的字符串

JIS58 代码的字符串如表 3.12 所示。

在表 3.12 中，00_H(NUL 代码）在字符串的结束时使用。

表 3.12 JIS8 代码的字符串

b8	b7	b6	b5	b4	b3	b2	b1	行 \ 栏	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
				0	0	0	0	0	NUL		(SP)	0	@	P	`	p								
				0	0	0	1	1			!	1	A	Q	a	q								
				0	0	1	0	2			"	2	B	R	b	r								
				0	0	1	1	3			#	3	C	S	c	s								
				0	1	0	0	4			$	4	D	T	d	t								
				0	1	0	0	5			%	5	E	U	e	u								
				0	1	1	0	6			&	6	F	V	f	v								
				0	1	1	1	7			'	7	G	W	g	w								
				1	0	0	0	8			(	8	H	X	h	x								
				1	0	0	1	9			)	9	I	Y	i	y								
				1	0	1	0	A			*	:	J	Z	j	z								
				1	0	1	1	B			+	;	K	[	k	{								
				1	1	0	0	C			(小数点) .	<	L	\	l	|							
				1	1	0	1	D			(减) −	=	M	]	m	}								
				1	1	1	0	E			(句号) 。	>	N	^	n	¯								
				1	1	1	1	F			/	?	O	(下划线) _	o									

基本 注 3.54

在基本模式 QCPU 中，只能通过 $MOV，STR，DSTR，VAL，DVAL，ESTR，EVAL 指令才可能使用字符串。

第 4 章　I/O 地址号的分配

在本章中，对 CPU 模块与 I/O 模块、智能功能模块进行数据收发需要的 I/O 地址号的分配进行说明。

4.1　主基板与插槽数的关系

(1) 可以使用的主基板

表 4.1 中显示了在每个 CPU 模块中可以使用的主基板 / 小型主基板。

表 4.1 CPU 模块与可能使用的主基板

CPU 模块	可使用的主基板
Q00JCPU	不需要 *1
Q00CPU、Q01CPU	Q3□B、Q3□SB、Q3□RB、Q3□DB*3
Q02CPU、Q02HCPU、Q06HCPU、Q12HCPU、Q25HCPU	
Q12PHCPU、Q25PHCPU	Q3□B、Q3□RB*2、Q3□DB*3
Q12PRHCPU、Q25PRHCPU	
Q02UCPU、Q03UDCPU、Q04UDHCPU、Q06UDHCPU	Q3□B、Q3□SB、Q3□RB、Q3□DB

*1:　Q00JCPU 为电源模块与主基板一体的 CPU 模块。
　　不需要电源模块与主基板。
*2:　Q12PHCPU、Q25PHCPU、Q12PRHCPU、Q25PRHCPU 不能使用小型主基板 (Q3□SB)。
*3:　不能使用多 CPU 间的高速存取功能。

(2) 可以使用的插槽数

可以使用的插槽数（模块数），包括空插槽数。

（图 4.1 所示的即使将插槽 2 设定为 “空插槽 0 个”，也占用一个插槽。

可以使用的插槽数（模块数）根据各主基板会有不同。

☞ QCPU 用户手册（ 硬件设计 / 维护点检篇 ）

另外，关于基板的分配、I/O 地址号的分配思路，请参照 4.4 节与 4.6 节。

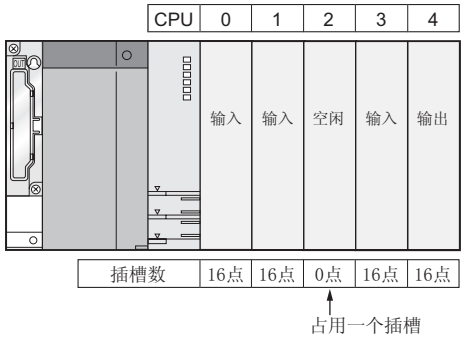


图 4.1 包含空插槽的模块数

4.2 扩展基板级数与插槽数的关系

CPU 模块通过下述显示的构成可以构筑系统。

- 由一个主基板与扩展基板构成
- 由一个小型主基板构成

(1) 可以使用的基板级数与插槽数

表 4.2 中说明在各个 CPU 模块中可以使用的基板级数与插槽数的关系。

表 4.2 扩展基板扩展级数与插槽数

CPU 模块	扩展基板（扩展级数）	小型主基板 *1	可以使用的插槽数 *2（可以使用的模块数）
Q00JCPU	○(2 级)	×	16 插槽（模块）
Q00CPU，Q01CPU	○(4 级)	○	24 插槽（模块）
Q02CPU，Q02HCPU，Q06HCPU，Q12HCPU，Q25HCPU	○(7 级)	○	64 插槽（模块）
Q12PHCPU，Q25PHCPU	○(7 级)	×	64 插槽（模块）
Q12PRHCPU*3，Q25PRHCPU*3	○(7 级)	×	63 插槽（模块）
Q02UCPU	○(4 级)	○	36 插槽（模块）
Q03UDCPU，Q04UDHCPU，Q06UDHCPU	○(7 级)	○	64 插槽（模块）

○: 可以使用　×: 不能使用

*1: 小型主基板不能连接扩展基板。

*2: 可以使用的插槽数（模块数）包括空插槽数。
（图 4.2 所示的即使将插槽 2 设定为 “空 0 点”，也会占用一个插槽。）

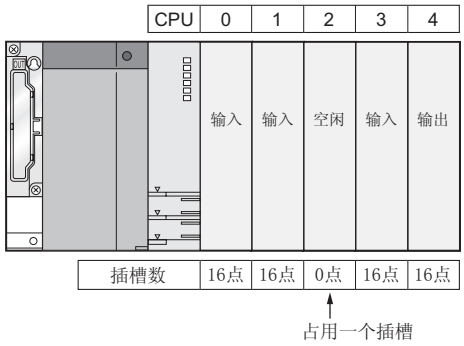


图 4.2 包含空插槽的模块数

*3: 是以序列号的前 5 位数为 “09012”以后的 Q12PRHCPU、Q25PRHCPU 为对象。序列号的前 5 位数为 “09011”以前的 Q12PRHCPU、Q25PRHCPU 不能连接扩展基板。
因此可使用的插槽数为 11 个插槽。

(2) 与安装模块数相关的注意事项

请在可以使用的插槽数的范围内进行模块的安装。

主基板与扩展基板的合计即使比可以使用的插槽数多（例如：使用 12 个插槽基板中的 6 个），被安装在可以使用的插槽数之内的模块也不会出现错误。

在可以使用的插槽数范围外安装的模块会出现（“SP. UNIT LAY ERR) 的错误。

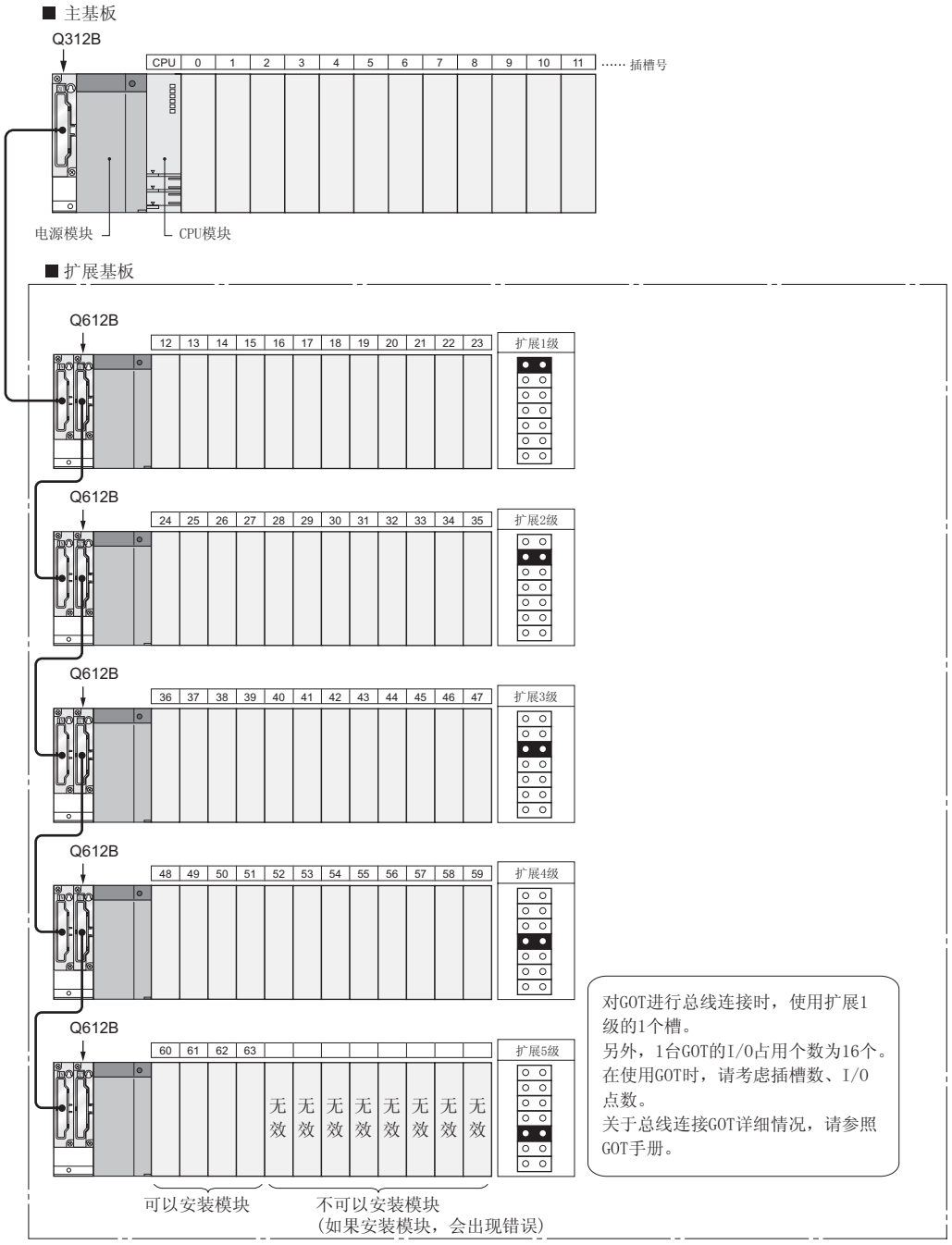


图 4.3 高性能模式 QCPU 情况的构成示例

4.3 关于扩展基板的安装与级数设定

可以使用扩展基板的类型如表 4.3 所示。

表 4.3 可以使用的扩展基板

CPU 模块	可使用的扩展基板
Q00JCPU	Q5□B、Q6□B、Q6□RB
Q00CPU、Q01CPU	
Q02CPU、Q02HCPU、Q06HCPU、Q12HCPU、Q25HCPU	Q5□B、Q6□B、Q6□RB、QA1S6□B、QA6□B
Q12PHCPU、Q25PHCPU	Q5□B、Q6□B、Q6□RB
Q12PRHCPU、Q25PRHCPU	Q6□WRB(固定为扩展第 1 级)、Q6□RB(扩展第 2 级以后)
Q02UCPU、Q03UDCPU、Q04UDHCPU、Q06UDHCPU	Q5□B、Q6□B、Q6□RB

要 点

小型主基板不能连接扩展基板。

(1) 扩展级数的设定及设定步骤

使用扩展基板扩展时，请通过扩展基板上的级数设定连接器设定扩展级数。
对扩展级数请从与主基板连接的扩展基板开始设定连接顺序。

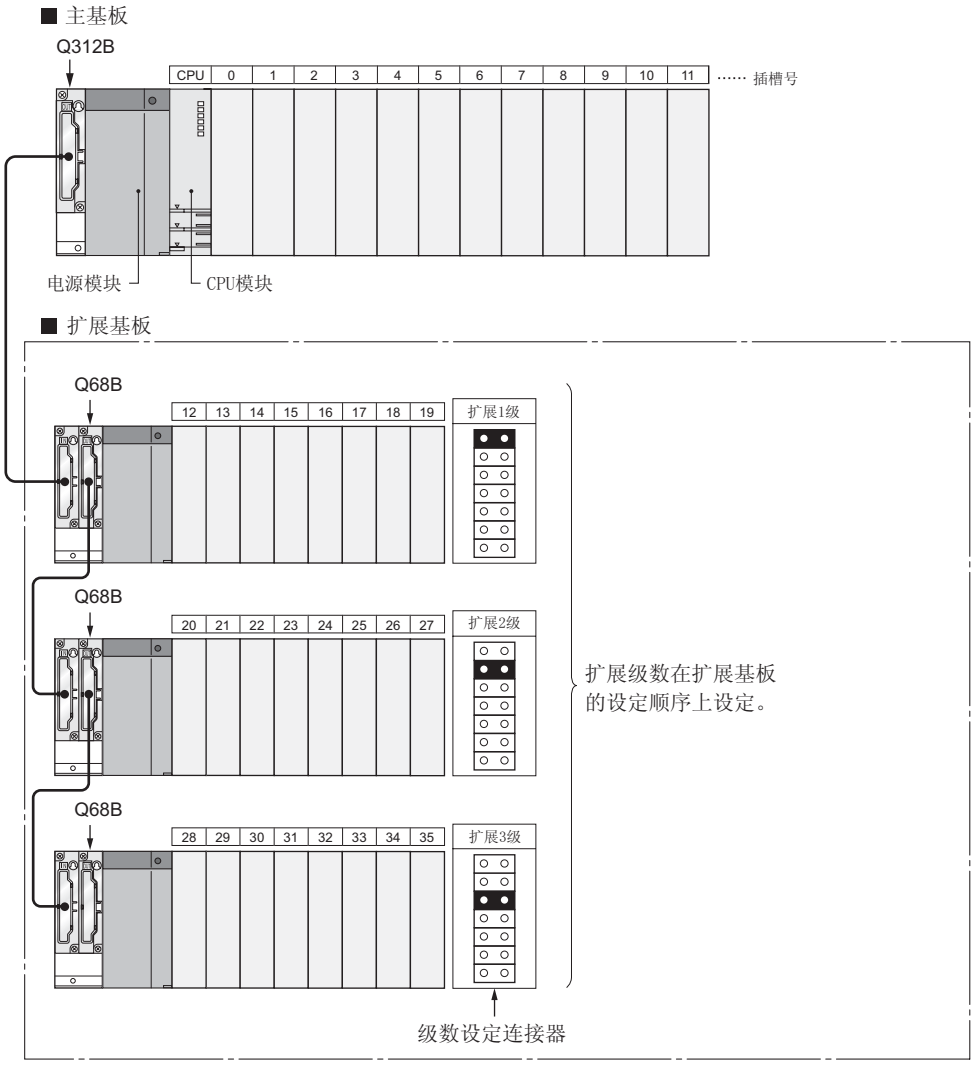


图 4.4 扩展级数的设定与设定顺序

(2) 扩展级数设定时的注意事项

(a) 关于扩展级数的设定顺序

请连续设定扩展级数。

在基板的分配为自动模式（☞ 4.4 节 (1)）的情况下，即使跳跃式地设定扩展级数，跳过的级数会变成插槽数 0，不增加空插槽数。

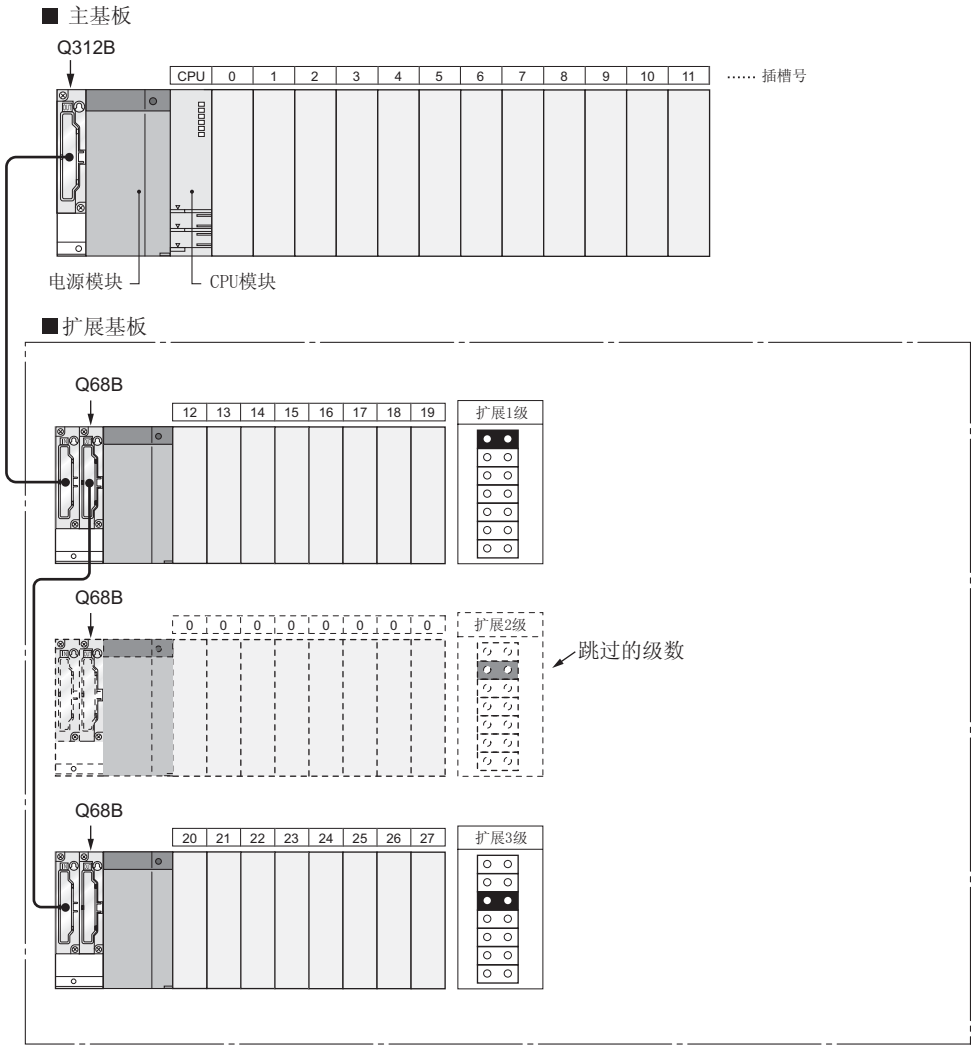


图 4.5 跳跃式地设定扩展级数时插槽数

(b) 设定同样的扩展级数的情况

在多个扩展基板中，设定同样的扩展级数将不能使用。

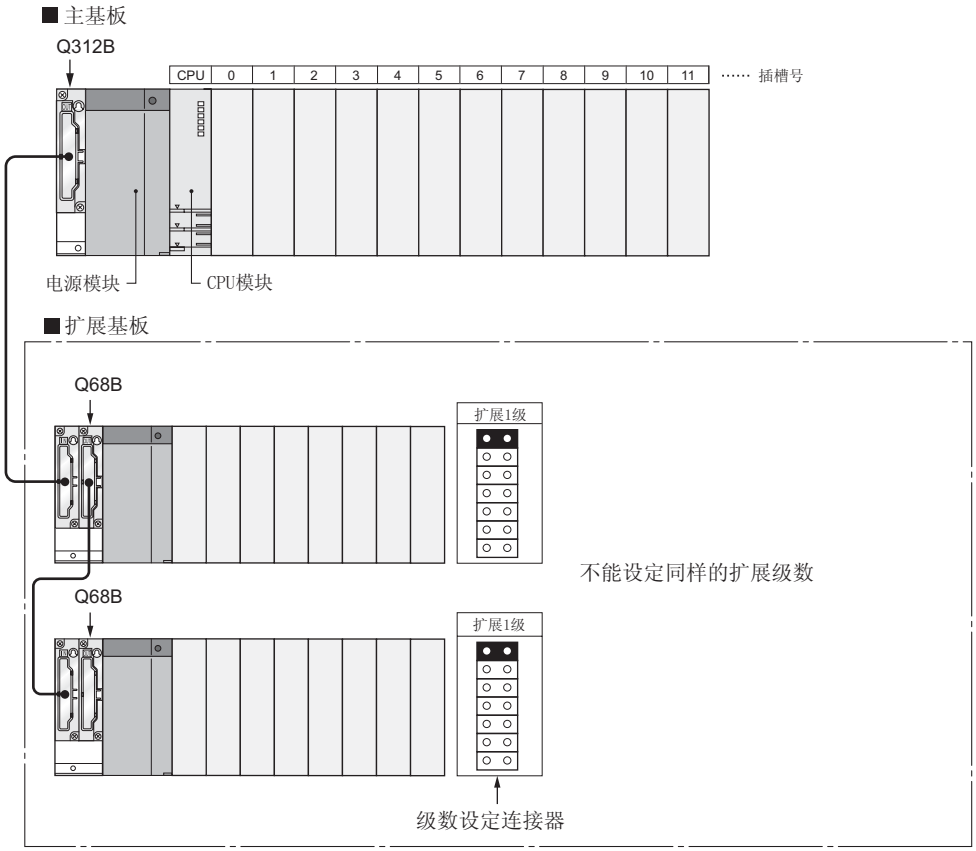


图 4.6 设定为相同扩展级数的情况

(c) 连接器针插入两个以上的地方、或者不插入的情况

级数设定的连接器针插入两个以上的地方的状态下将不能使用。

另外，扩展级数设定的连接器针在不插入的情况下也不能使用。

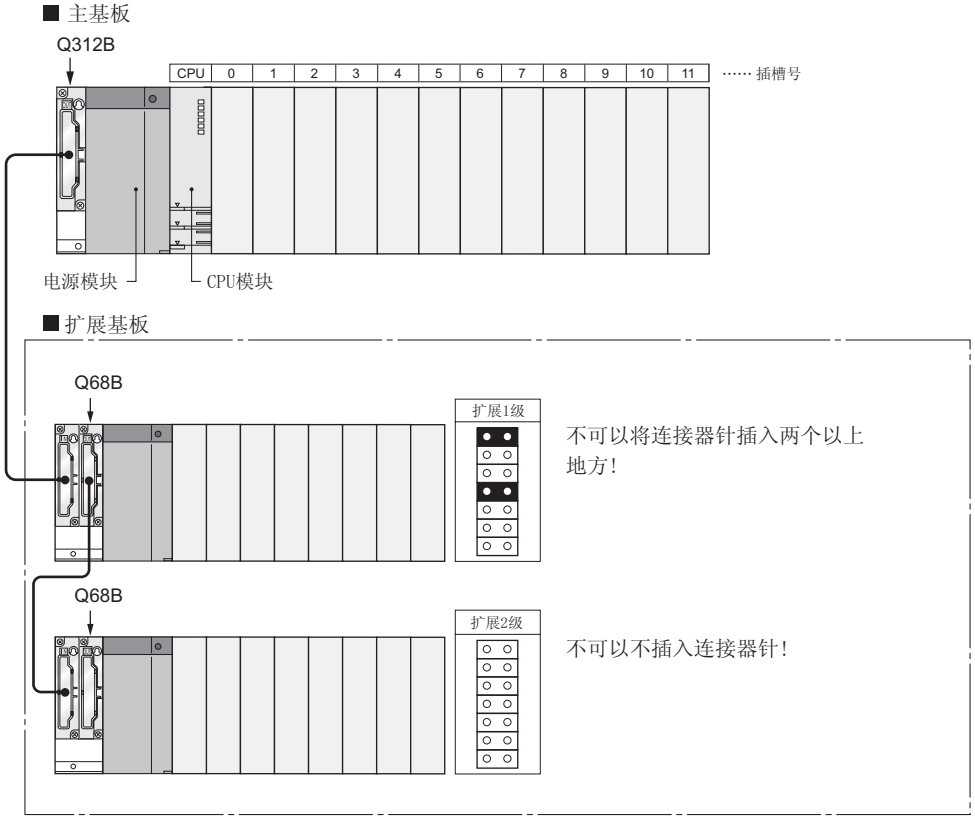


图 4.7 连接器针插入两个以上的地方、或者不插入的情况

基本 过程 冗余
注 4.1 注 4.1 注 4.1

通用
注 4.1

(d) 关于使用 AnS/A 系列对应的扩展基板 (QA1S□B、QA6□B) 时的扩展位置 *注 4.1*

Q5□B/Q6□B 与 QA1S6□B/QA6□B 混和使用时，靠近主基板地方的首先集中连接 Q5□B/Q6□B，然后再连接 QA1S6□B、QA6□B。

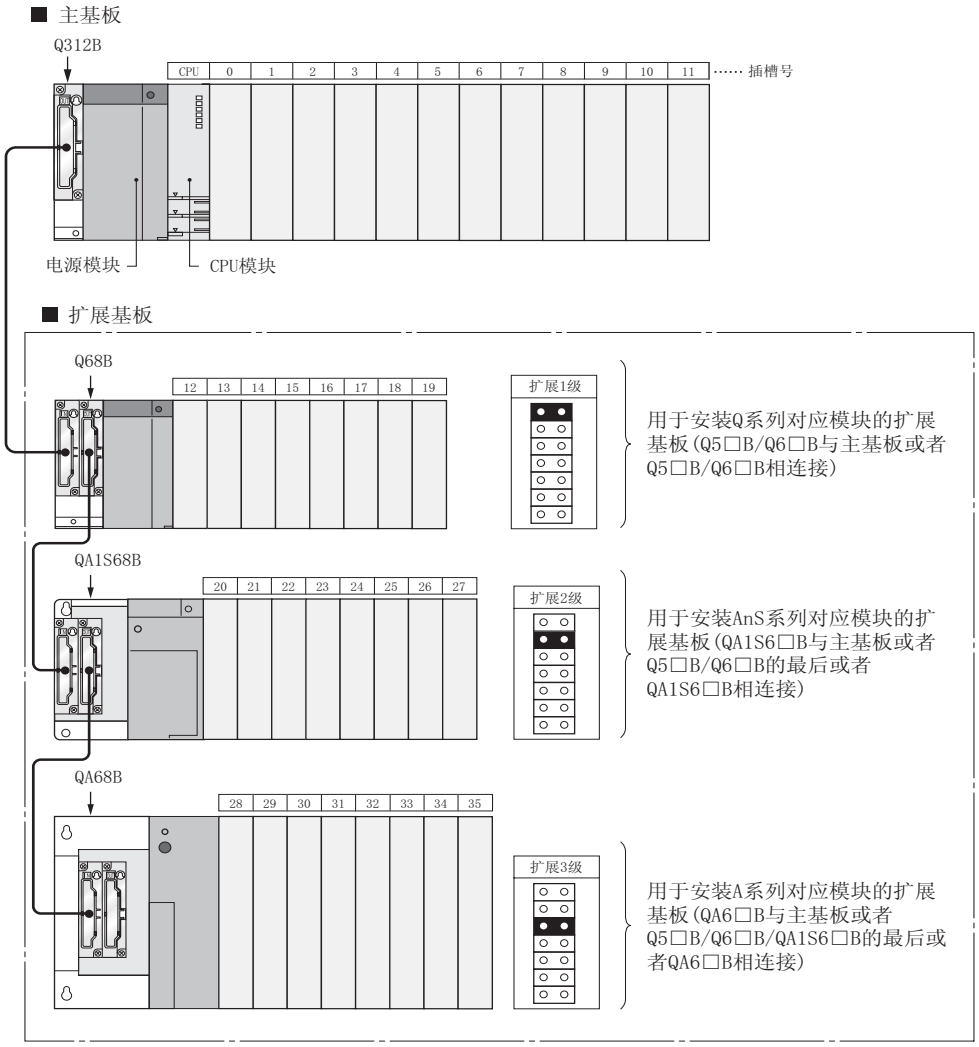


图 4.8 使用 AnS/A 对应扩展基板时的扩展位置

基本 过程 冗余
注 4.1 注 4.1 注 4.1

通用
注 4.1

在基本模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 中，不能使用 AnS/A 系列对应的扩展基板 (QA1S□B、QA6□B)。

4.4 基板的分配（基本模式）

基本 过程 冗余
注 4.2 注 4.2 注 4.2

CPU 模块的主基板、小型主基板注 4.2、扩展基板的模块数的分配有自动模式与详细模式两种。
基本模式的设定在可编程控制器参数的 I/O 分配设定中进行。(☞ 本项 (3))

(1) 关于自动模式

自动模式是指根据主基板、小型主基板、扩展基板的允许安装插槽数进行基板分配的模式。
I/O 地址号根据正在使用的基板上安装的模块数来进行分配。

AnS/A 系列中，由于主基板、扩展基板为固定的 8 个插槽，所以，即使使用 3 个插槽 / 5 个插槽的基板，也将占用 8 个插槽。
Q 系列 CPU 模块由于只占用可以实际安装的插槽数，在使用 3 个插槽的基板时，只占用 3 个插槽。

基本 过程 冗余
注 4.2 注 4.2 注 4.2

Q00JCPU、过程 CPU、冗余 CPU 不能使用小型主基板。

(a) 3 个插槽基板的情况：占用 3 个插槽

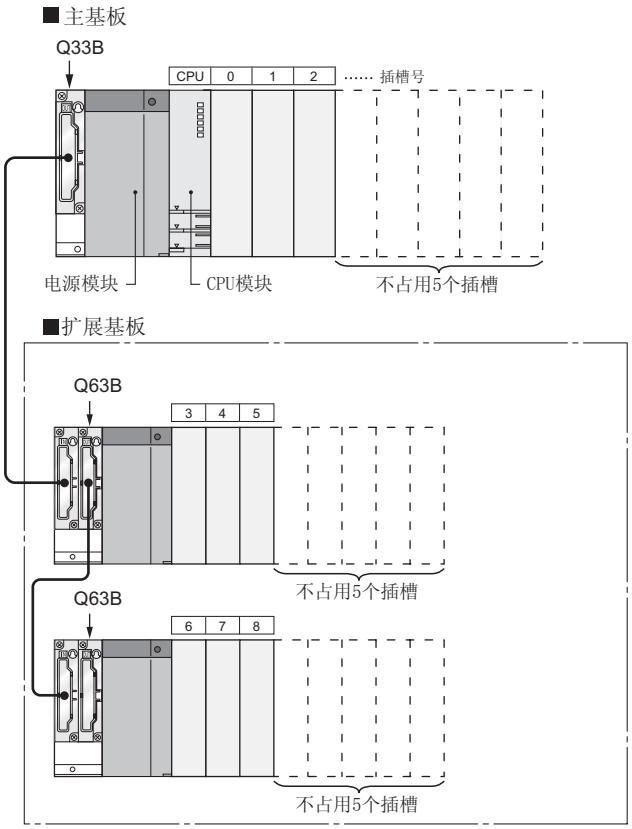


图 4.9 3 个插槽基板的情况

(b) 5 个插槽基板 /Q00JCPU 的情况：占用 5 个插槽

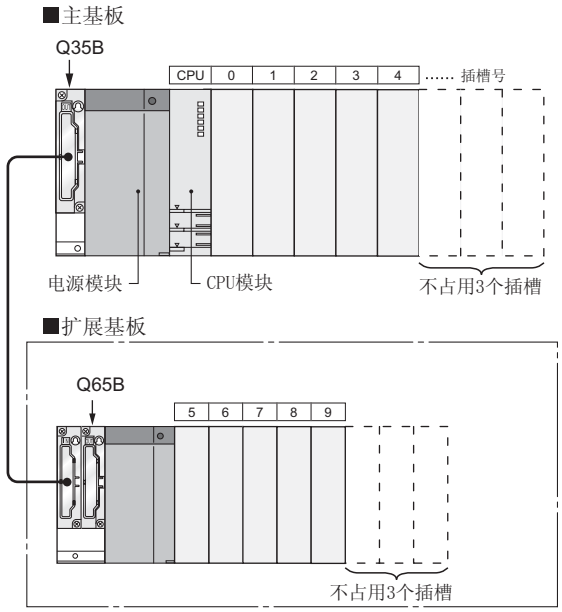


图 4.10 5 个插槽基板的情况

(c) 8 个插槽基板的情况：占用 8 个插槽

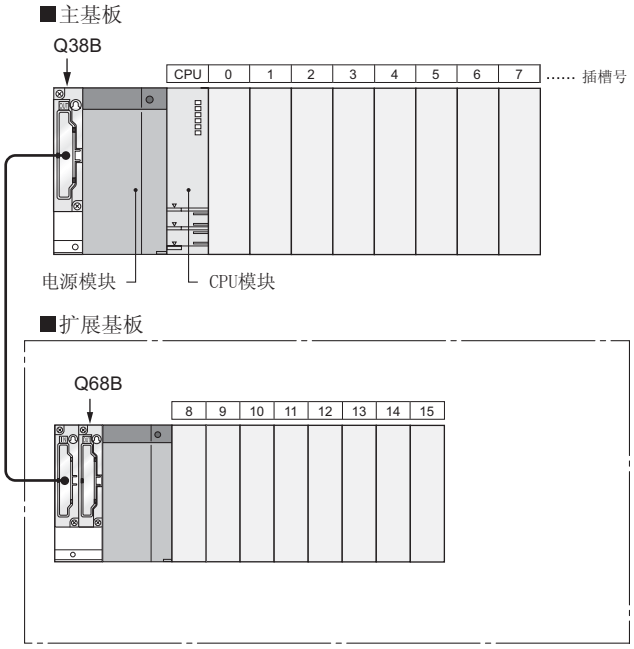


图 4.11 8 个插槽基板的情况

(d) 12 个插槽基板的情况：占用 12 个插槽

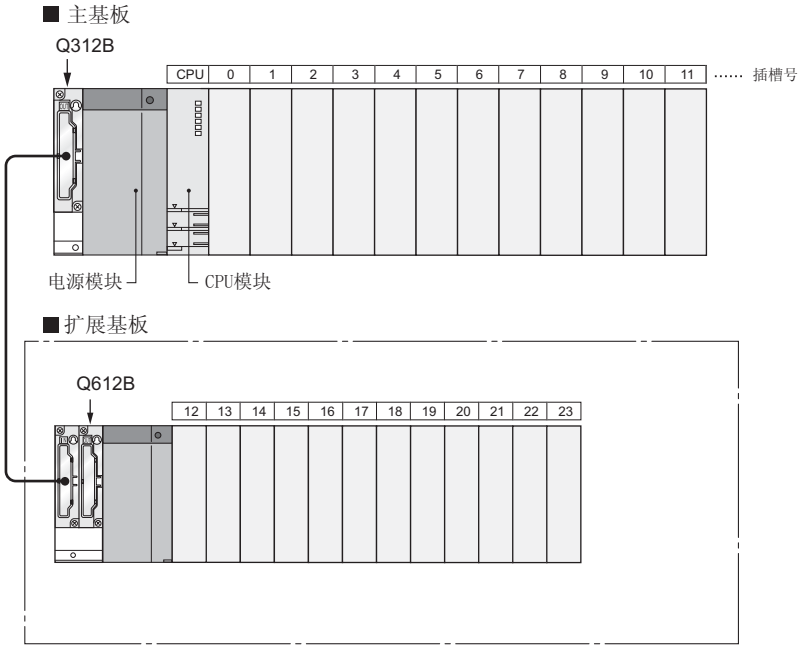


图 4.12 12 个插槽基板的情况

(2) 关于详细模式

详细模式是指在可编程控制器参数的 I/O 分配设定中对各基板设定可以安装模块数的模式。

(a) 使用用途

与 AnS/A 系列的基板的占用数（固定的 8 个插槽）相吻合等情况下可以使用。
此外还可以用于如下情况：由于对空插槽进行 I/O 分配时，即使设定为 0 点也会占用 1 个插槽，此时系统无法识别安装模块空槽后面的插槽。

(b) 插槽数的设定与注意事项

插槽数的设定可以与使用的基板的插槽数无关。

要 点

对于使用中的所有基板，请进行插槽数的设定。
如果没有对所有基板进行插槽数设定，则 I/O 分配有可能不能正常动作。

设定的插槽数与使用中的基板的插槽数出现不一致情况时，将变为如下状况。

1) 设定的插槽数比使用中的基板的插槽数多的情况
占用设定的插槽数。
比使用中的基板的插槽数多的部分成为空插槽。
例如：使用 5 个插槽的基板，如果设定的是 8 个插槽的话，将有 3 个空插槽。

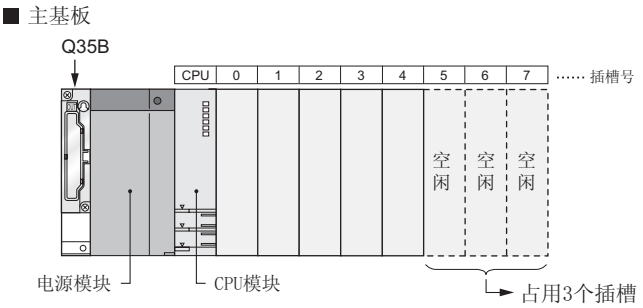


图 4.13 设定的插槽数比使用的基板的插槽数多的情况

空插槽的点数将成为可编程控制器参数的可编程控制器系统设定中设定的空插槽点数，或者可编程控制器参数的 I/O 分配中设定的点数（默认值为 16 点）。

2) 设定的插槽数比使用中的基板的插槽数少的情况
设定的插槽数之后的插槽将不能使用。
例如：使用 12 个插槽的基板，如果设定的是 8 个插槽的话，基板右面的 4 个插槽将禁止使用。
（禁止使用的插槽如果安装模块的话将会出错（SP.UNIT LAY ERR”）。

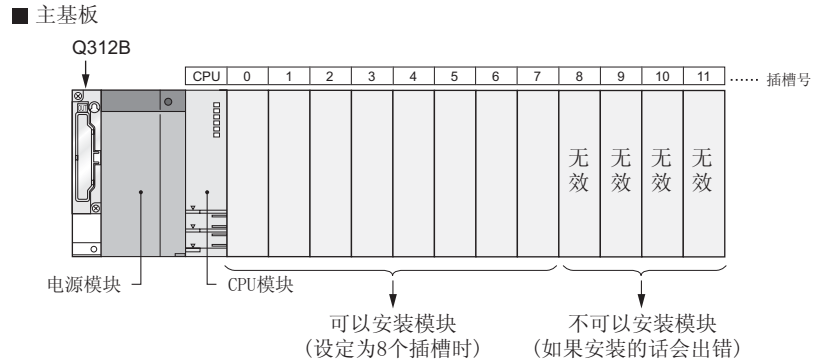
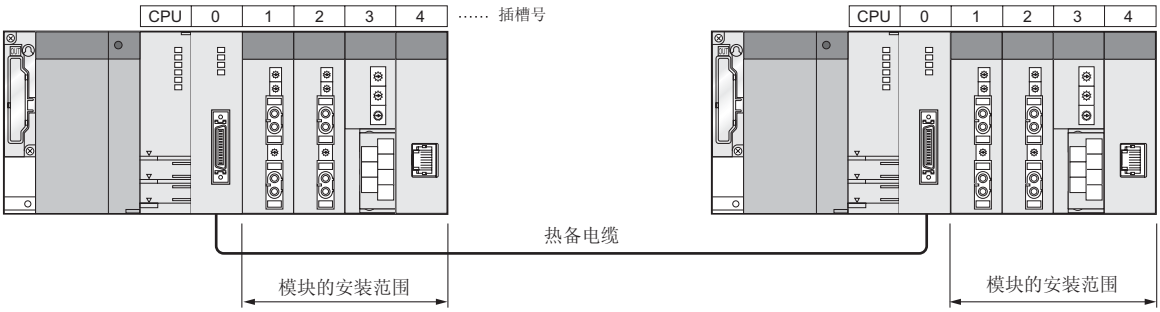


图 4.14 设定的插槽数比基板的插槽数少的情况

要 点

对于冗余 CPU 而言，一个冗余 CPU 占用 2 个插槽。
因此，可以安装的模块数为：设定的插槽数减 1。
例如：使用 5 个插槽的主基板时，实际上可以安装的插槽数为 4 个。



(3) GX Developer 的基本模式的设定画面与设定内容

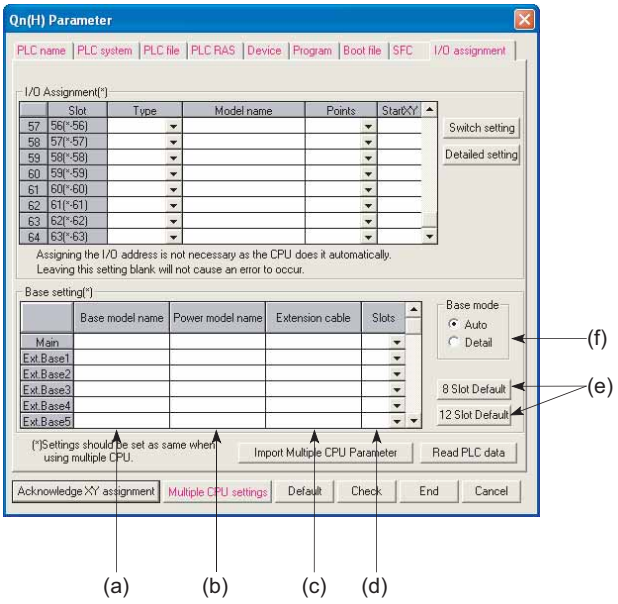


图 4.15 I/O 分配设定

(a) 基板的型号名称
安装的基板的型号名称设定为半角 16 号字以下。
CPU 模块不使用设定的型号名称。(请作为用户的记录或者参数的打印使用。)

(b) 电源模块的型号名称
安装的电源模块的型号名称设定为半角 16 号字以下。
CPU 模块不使用设定的型号名称。(请作为用户的记录或者参数的打印使用。)

(c) 扩展电缆的型号名称
使用的扩展电缆的型号名称设定为半角 16 号字以下。
CPU 模块不能使用设定的型号名称。(请作为用户的记录或者参数的打印使用。)

(d) 插槽数 (CPU 模块中使用)
使用的基板的插槽数的点数如何设置，有下述插槽数可供选择。
- 2(2 个插槽)
- 3(3 个插槽)
- 5(5 个插槽)
- 8(8 个插槽)
- 10(10 个插槽)
- 12(12 个插槽)

(e) 8 个固定 /12 个固定 (CPU 模块中使用)
在对基板批量设定指定插槽数时选择。

(f) 自动 / 详细
选择进行基板分配的自动模式还是详细模式。

4.5 关于 I/O 地址号

I/O 地址号表示在顺控程序中，CPU 模块接收 ON/OFF 数据以及从 CPU 模块向外部输出 ON/OFF 数据时所使用的地址。

(1) ON/OFF 数据的读取与输出

CPU 模块接收 ON/OFF 数据由输入 (X)，CPU 模块输出 ON/OFF 数据由输出 (Y) 进行。

(2) I/O 地址号的表示

I/O 地址号通过 16 进制数表示。

使用 16 个输出模块的情况下，I/O 地址号如图 4.16 所示，1 个插槽有□□ 0～□□F 的 16 个点，且为连续号。

安装在基板上的模块：

- 输入模块的情况下在 I/O 地址号起始上加上 “X”
- 输出模块的情况下在 I/O 地址号起始上加上 “Y”

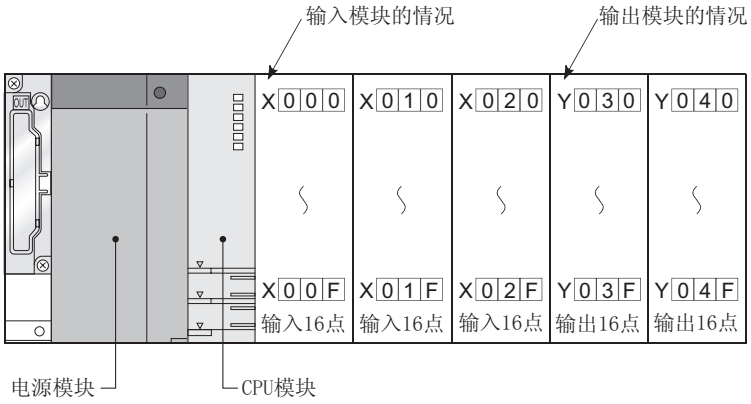


图 4.16 输入地址号与输出地址号

4.6 I/O 地址号的分配思路

4.6.1 基板的 I/O 地址号

CPU 模块在电源投入或者复位解除时，进行 I/O 地址号的分配。
基板的设定为自动模式时，在不进行 I/O 分配情况下的 I/O 地址号的分配示例如图 4.17 所示。

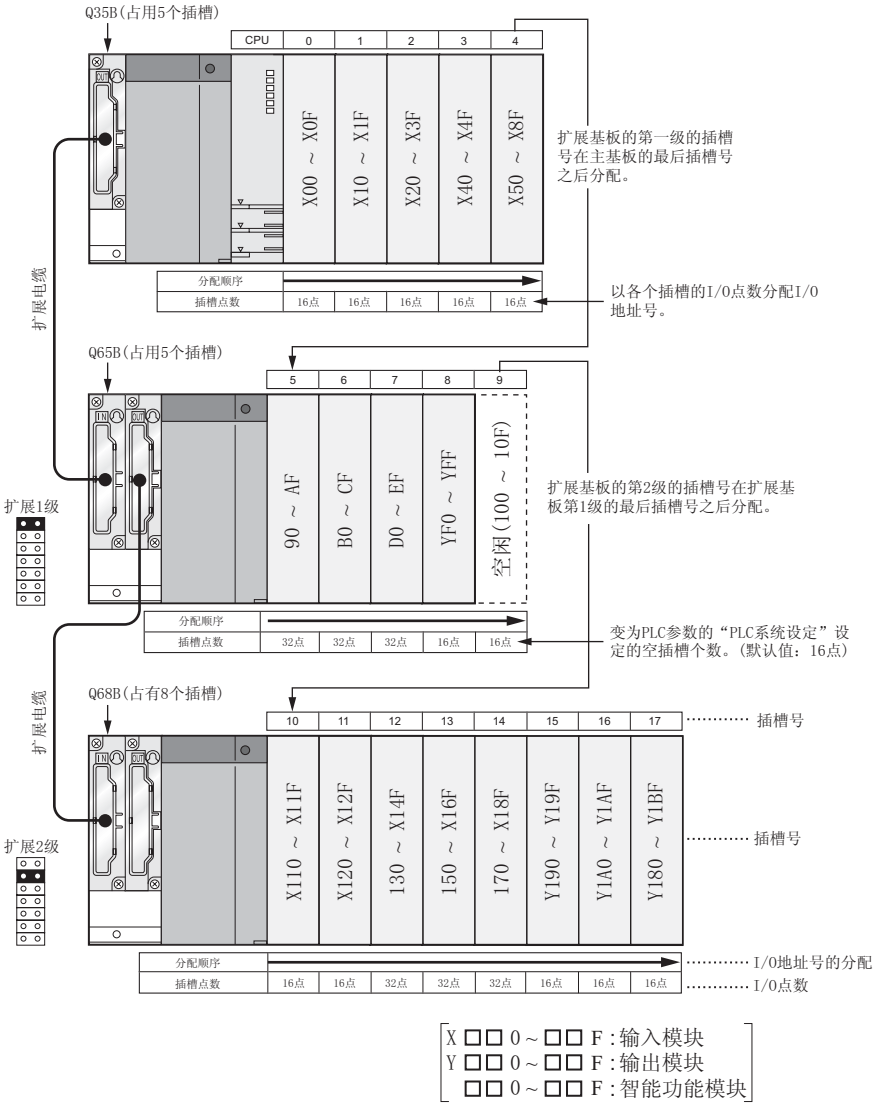


图 4.17 I/O 地址号的分配示例

表示智能功能模块的 I/O 点数在 32 点时的情况。
I/O 点数根据智能功能模块会有不同。
请通过智能功能模块手册确认 I/O 点数后，进行 I/O 地址号的分配。

要 点

即使不通过 GX Ceveloper 进行 I/O 分配，也可以进行 CPU 模块的控制。

在分配 I/O 地址号时，请根据下述的项目分配 I/O 地址号。

(1) 基板的插槽数

基板的插槽数在基本模式的设定中设定。(☞ 4.4 节)

(a) 自动模式的情况下

即为基板可以安装的模块数。

使用 5 个插槽的基板时为 5 插槽，使用 12 个插槽的基板时为 12 插槽。

(b) 详细模式的情况下

即为可编程控制器参数的 I/O 分配设定中设定的插槽数。

基本 注 4.3　过程 注 4.3　冗余 注 4.3

(2) I/O 地址号的分配顺序

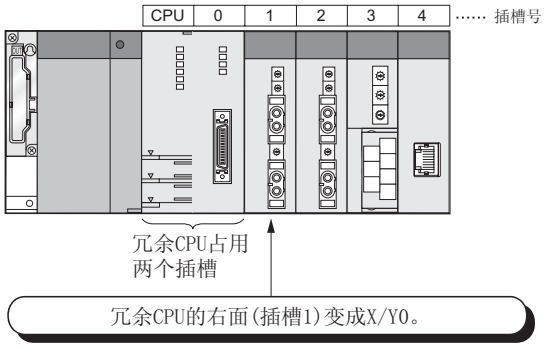
I/O 地址号在主基板 / 小型主基板注 4.3 的 CPU 模块的右面加上 0H，从左到右的顺序以连号的方式分配地址号。

要 点

在冗余 CPU 中，一个冗余 CPU 占用两个插槽。

因此，插槽 1(冗余 CPU 的右面) 的 I/O 地址号变成 X/Y0。

■主基板(5个插槽)



基本 注 4.3　过程 注 4.3　冗余 注 4.3

在 Q00JCPU、过程 CPU、冗余 CPU 中不能使用小型主基板。

(3) 扩展基板的 I/O 地址号的分配顺序

扩展基板在主基板的 I/O 地址号的下一个号开始分配 I/O 地址号。

扩展基板的 I/O 地址号分配按照扩展基板的级数设定连接器的设定顺序，如图 4.18 所示从左 (I/O 0) 到右的顺序连号分配地址号。

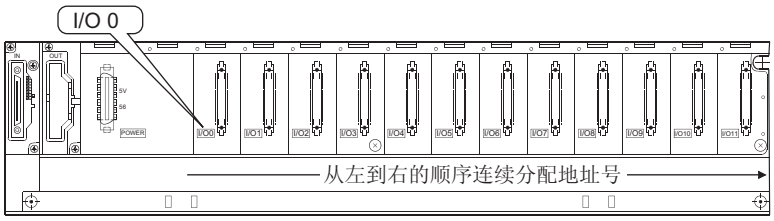


图 4.18 I/O 地址号的分配顺序

(4) 各个插槽号的 I/O 地址号

基板的各个插槽占用着所安装的的 I/O 模块、智能功能模块的 I/O 点数的 I/O 地址号。

在 CPU 模块的右面安装了 32 点的输入模块时，I/O 地址号变为 X0~1F。

(5) 空插槽的 I/O 地址号

基板中没有安装 I/O 模块、智能功能模块的空插槽将被分配为可编程控制器参数的可编程控制器系统设定中所设定的点数。(默认值：16 点)

要 点

基板的分配为自动模式时，即使通过基板的级数设定连接器跳跃地设定扩展级数，也不能预留空的扩展级数。(☞ 4.3 节 (2)(a))

如果想预先留有将来扩充用的空的扩展级数时，请通过可编程控制器参数进行基板的设定。

4.6.2 远程站的 I/O 地址号

基本 注 4.4

在 MELSECNET/H 远程 I/O 网络注 4.4、注 4.5、CC-Link 等的远程 I/O 系统中，可以将 CPU 模块软元件的输入 (X)、输出 (Y) 分配到远程站的 I/O 模块及智能功能模块，从而对模块进行控制。

高性能 注 4.5

另外，可以将输入 (X)、输出 (Y) 用于 MELSECNET/H 模块的链接 I/O(LX/LY) 的刷新对象 (CPU 模块侧的软元件)。

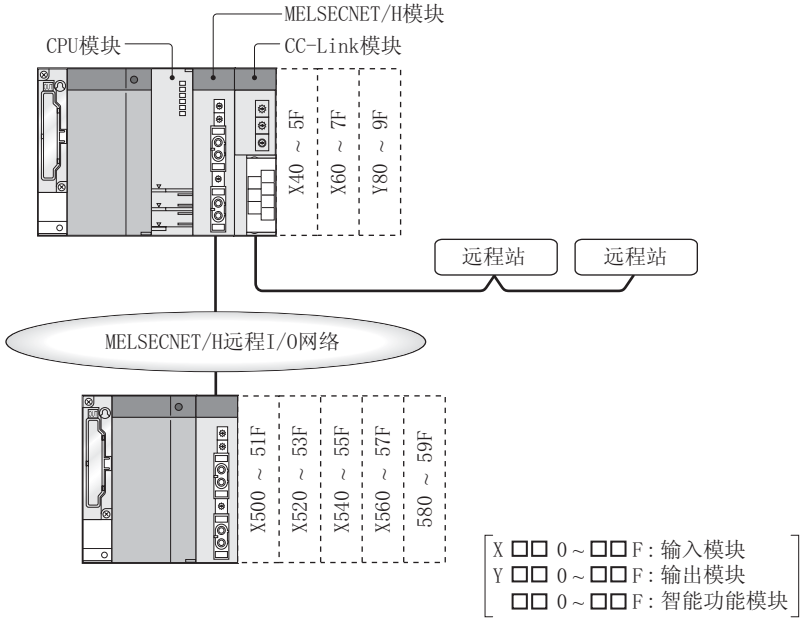


图 4.19 远程站的 I/O 地址号

(1) 远程站可以使用的 CPU 模块的 I/O 地址号

CPU 模块的输入 (X)、输出 (Y) 用于远程站的 I/O 地址号时，请不要使用 CPU 侧的 I/O 模块以及智能功能模块正在使用的 I/O 地址号，只能使用其后面的号进行分配。

例如：CPU 侧的 I/O 模块以及智能功能模块中使用 X/Y0~X/Y3FF(1024 点) 时，在远程站中 X/Y400 之后的可以使用。

基本 注 4.4

基本模式 QCPU 与 MELSECNET/H 远程 I/O 网络不相对应。

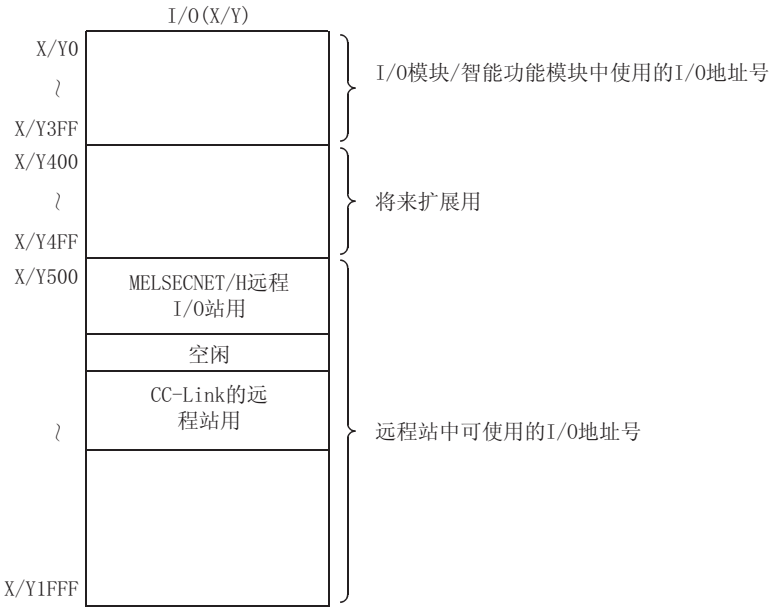
高性能 注 4.5

在高性能模式 QCPU 中使用 MELSECNET/H 远程 I/O 网络时，请确认 CPU 模块以及 GX Developer 的版本。(附录 4.2)

(2) 使用远程站的 I/O 地址号时的注意事项

(a) 考虑将来扩展时的设定

CPU 模块的输入 (X)、输出 (Y) 用于远程站的 I/O 地址号时，在设定时请考虑 CPU 模块侧的 I/O 模块以及智能功能模块的扩展。(☞ 图 4.20)



I/O模块/智能功能模块中使用X/Y0～3FF(1024点)，预留将来扩展用的
X/Y400～4FF(256点)时。

图 4.20 远程站的 I/O 地址号的分配

(b) 使用 MELSECNET/H 以及 CC-Link 时

请不要重复设定 MELSECNET/H 的刷新对象 (CPU 模块侧软元件) 的 I/O 地址号以及 CC-Link 远程 I/O 系统的 I/O 地址号。

(c) 关于 CPU 模块的 I/O 软元件个数

I/O 软元件点数在每个 CPU 模块中有所不同。

关于 CPU 模块的 I/O 软元件点数，请参照第 2 章。

要 点

1. CC-Link 系统中在没有进行网络参数的设定的情况下，对小号 CC-Link 的主、局部模块进行如下点数的分配。
   - 基本模式 QCPU
     X/Y400~7FF(1024 点)
   - 高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU
     X/Y1000~17FF(2048 点)
2. MELSECNET/H 远程 I/O 网络、CC-Link 等的 I/O 地址号的分配顺序没有限制。
3. 在 MELSECNET/H 远程 I/O 站用及 CC-Link 的远程站用的之间也可以设置空闲。

4.7 通过 GX Developer 进行的 I/O 分配

对通过 GX Developer 进行的 I/O 分配情况进行说明。

4.7.1 通过 GX Developer 进行的 I/O 分配的目的

通过 GX Developer 进行 I/O 分配的设定，在如下情形中进行。

(1) 变更为 16 点模块以外的模块的预留
为了以后将当前使用的模块变更为不同点数的模块时不变更 I/O 地址号，可以预先预留点数。
例如：将安装了 16 点输入模块的插槽按 32 点输入模块进行分配。

(2) 防止更换模块时的 I/O 地址号的变化
在 16 点以外的 I/O 模块、智能功能模块出现故障而拆除模块的情况下，可以防止 I/O 地址号的变化。

(3) 对程序中使用的 I/O 地址号的变更
在设计的程序中使用的 I/O 地址号与实际系统的 I/O 地址号不一致的情况下，可以将基板的各个模块的 I/O 地址号变更为程序的 I/O 地址号。

(4) 输入响应时间的设定（I/O 响应时间）
可以将输入模块、I/O 混和模块、中断模块的输入响应时间变更为与系统相吻合的时间。(☞ 6.7 节)

(5) 出错时的输出模式的设定
可以设定输出模块、I/O 混和模块、智能功能模块、中断模块出错时的输出模式。
(☞ 6.8 节)

(6) 智能功能模块、中断模块的开关设定
进行智能功能模块、中断模块的开关设定。(☞ 6.10 节)

(7) CPU 模块出错时的输出设定
设定 CPU 模块在停止出错中停止运算时的输出模块与智能功能模块的输出（保持 / 清除状态）。(☞ 6.8 节)

(8) 智能功能模块的硬件出错时的 CPU 模块的动作设定
设定智能功能模块的硬件发生出错时 CPU 模块的动作（继续运行 / 停止）状态。
(☞ 6.9 节)

要 点

1. I/O 分配设定在可编程控制器的电源由 OFF 到 ON 或者 CPU 模块进行复位解除时有效。
2. 在进行输入模块的响应时间的变更、智能功能模块的开关设定时，有必要进行 I/O 分配。
3. 在不通过 GX Developer 进行 I/O 分配的状态下，如果 16 点以外的 I/O 模块发生故障，那么在其之后的 I/O 地址号有发生变化并引起误动作的情况。
   因此，推荐您通过 GX Developer 进行 I/O 分配设定。

4.7.2 对通过 GX Developer 进行 I/O 分配的思考

在 I/O 分配中，在基板的各个插槽可以分别设定 “种类”（模块类别）、“点数”（I/O 点数）、“起始 XY”（起始 I/O 地址号）。
例如在变更指定的插槽的 I/O 点数的情况下，可以只设定点数。
没有设定的项目即为基板的安装状态。

(1) I/O 分配的设定

I/O 分配在可编程控制器参数的 I/O 分配的设定中进行。

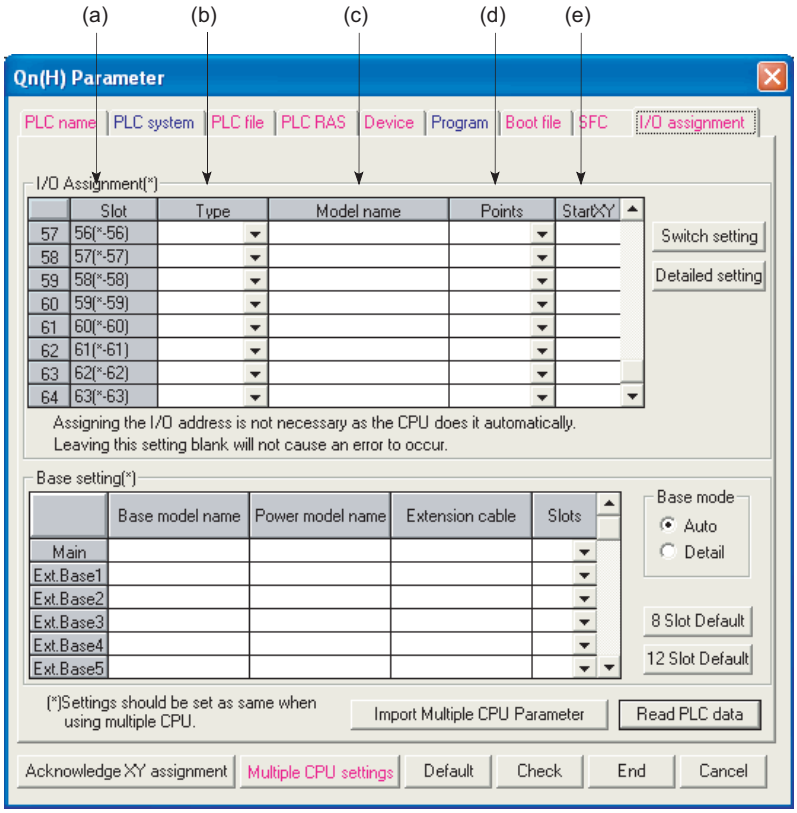


图 4.21 I/O 分配设定

(a) 插槽

表示插槽号与基板的第几级的第几个插槽。
在自动模式中设定基板的情况下，基板的第几级变为 “*”，第几个插槽变为在主基板的第 0 个插槽开始的插槽数。

(b) 种类（在 CPU 模块中使用）

在下述项目中选择安装模块的种类。

- 空闲（空插槽）
- 输入（输入模块）
- 高速输入（高速输入模块）*1
- 输出（输出模块）
- I/O 混和（I/O 混和模块）
- 智能（智能功能模块、AnS/A 时应特殊功能模块）注 4.6
- 中断（中断模块）*2

基本 过程 冗余 注 4.6 注 4.6 注 4.6

通用 UD 注 4.6

没有设定种类的插槽将被视为所安装模块的种类。

* 1: “高速输入”在 GX Developer Version5(SW5D5C-GPPW) 以后的产品中可以设定。
* 2: “中断”在 GX Developer Version6(SW6D5C-GPPW) 以后的产品中可以设定。

(c) 型号名称

安装模块的型号名称设定为半角 16 号字以下。

CPU 模块中不使用设定的型号名称。（仅作为用户的记录）

(d) 点数（在 CPU 模块中使用）

变更各插槽的 I/O 点数时，在下述中选择点数。

• 0(0 点)	• 16(16 点)	• 32(32 点)
• 48(48 点)	• 64(64 点)	• 128(128 点)
• 256(256 点)	• 512(512 点) *1	• 1024(1024 点) *1

未设定点数的插槽将被视为所安装的模块的点数。

* 1: 在 Q00JCPU 中不能进行设定。
（由于 Q00JCPU 的 I/O 点数为 256 点）

(e) 起始 XY(CPU 模块中使用)

在变更各个插槽的 I/O 地址号时，设定变更后的起始 I/O 地址号。

没有设定起始 XY 的插槽将被分配到设定的插槽之后的连续 I/O 地址号

基本 过程 冗余 注 4.6 注 4.6 注 4.6

通用 UD 注 4.6

在基本模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 中，不能使用 AnS/A 系列对应的特殊功能模块。

(2) I/O 分配时的注意事项

(a) 已进行了 I/O 分配的插槽的状态
已进行了 I/O 分配设定的插槽与模块的实际安装没有关系， 而以 I/O 分配设定为优先。

1) 设定的点数少于实际安装的 I/O 模块点数时的情况
实际安装的 I/O 模块的实际使用点数将减少。
例如：安装了 32 点输入模块的插槽，在 I/O 分配设定中如果设定的是 16 点输入模块的话，32 点输入模块的后半个的 16 点将不能使用。

2) 设定点数小于实际安装的智能功能模块的点数时
会出现 “SP.UNIT LAY ERR.”。

3) 设定点数大于实际安装的 I/O 模块点数的情况
超出实际安装点数的部分在 I/O 模块不使用。

4) 安装的模块与 I/O 分配的种类
安装模块的种类与 I/O 分配设定的种类应相同。
I/O 分配设定与安装的模块的种类在不一致的情况下，不能进行正常的动作。
另外，智能功能模块也要设定相同的 I/O 点数。
安装的模块与 I/O 分配设定的种类在不一致时的动作如表 4.4 所示。

表 4.4 实际安装的模块与 I/O 分配设定的不一致时的动作一览表

安装模块	I/O 分配的设定	结果
输入模块 输出模块 I/O 混和模块	智能 / 中断	出错 (SP.UNIT.LAY.ERR.)
智能功能模块	输入 / 高速输入 输出 /I/O 混和	出错 (SP.UNIT.LAY.ERR.)
空插槽	输入 / 高速输入 输出 /I/O 混合 智能 / 中断	被作为空插槽处理
所有的模块	空闲	被作为空插槽处理
其它的组合		虽然不出错，但不能正常执行动作。

5) 最后的 I/O 地址号
进行 I/O 分配时，最后的 I/O 地址号请设定在 I/O 点数最大值（☞第 2 章）的范围内。
最后的 I/O 地址号的设定如果超出 I/O 点数的最大值时，将会出错“SP. UNIT LAY ERR”。（在 GX Developer 的系统监视中，I/O 地址被表示为 ***。）

要 点

在扩展基板上混和使用 Q5□B/Q6□B/Q6□RB 与 QA1S6□B、QA6□B 时，需注意以下事项：

- 应按靠近主基板的一方首先连接 Q5□B/Q6□B，然后再连接 QA1S6□B、QA6□B 的顺序集中连接。
- 基板上安装的模块的 I/O 地址号应按各系列（Q 系列或 A 系列）分别集中分配。
  模块的 I/O 地址号如果未按上述要求分配，将会出现错误（“SP.UNIT LAY ERR.”）。

(b) CPU 模块自动分配起始 XY 时的注意事项

在起始 XY 没有输入的情况下，CPU 模块将自动分配起始 XY。
因此，下述的 1) 或 2) 的情况下，各个插槽的起始 XY 设定与 CPU 模块分配的起始 XY 设定有可能相互重复。

1) 在起始 XY 设定中将前后的 I/O 地址号进行了调换的设定。

2) 设定了起始 XY 的插槽与未设定的（自动分配的插槽）混和使用。

起始 XY 重复的情形如图 4.22 所示。

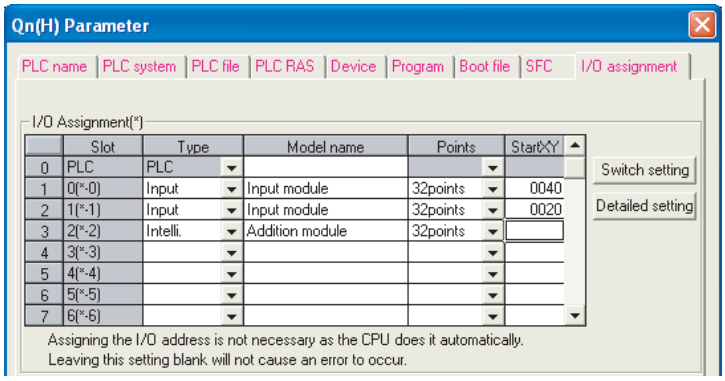


图 4.22 起始 XY 重复的 I/O 分配设定

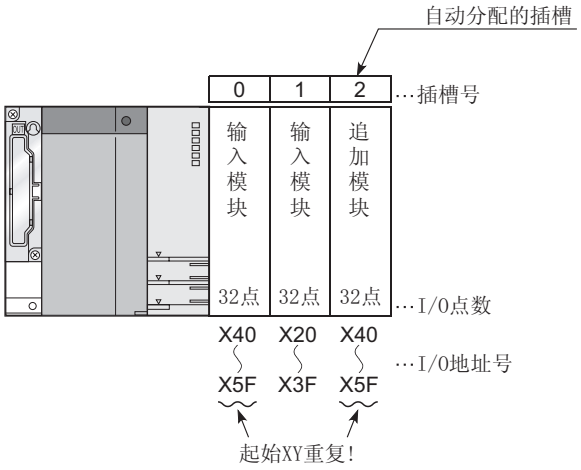


图 4.23 进行了如上所示的 I/O 分配设定时的起始 XY

请十分注意不要让各个插槽的起始 XY 重复。
当起始 XY 重复时，会出现 “SP.UNIT LAY ERR.” 的错误。

4.8 I/O 分配示例

说明通过 GX Developer 进行 I/O 分配设定时的 I/O 地址号的分配示例。

(1) 空插槽的点数由 16 点变更为 32 点的情况

在当前空插槽的位置上（插槽 No.3），将来安装 32 点的输入模块时，为了不变更 No.4 插槽以后的 I/O 地址号，请进行 32 点的预留。（插槽 No.12 的空插槽的 16 点不变更）

(a) 系统构成与 I/O 分配前的 I/O 地址号的分配

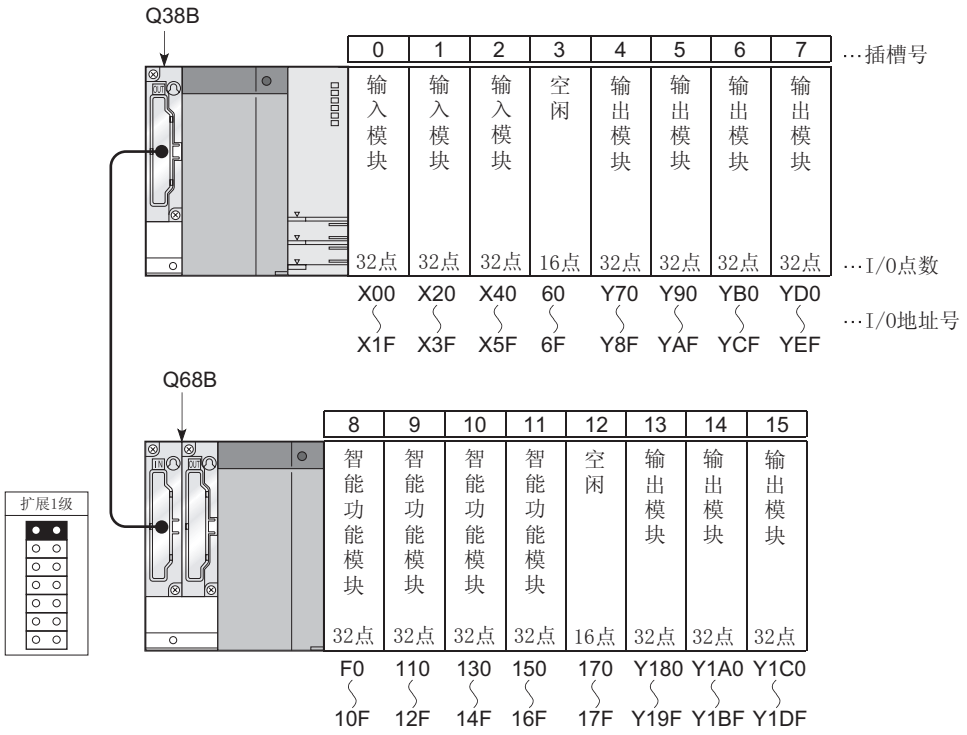


图 4.24 I/O 地址号的分配（空插槽点数变更前）

(b) 通过 GX Developer 进行 I/O 分配设定

在 GX Developer 的 I/O 分配设定画面中将插槽 No.3 设定为 “32 点”。

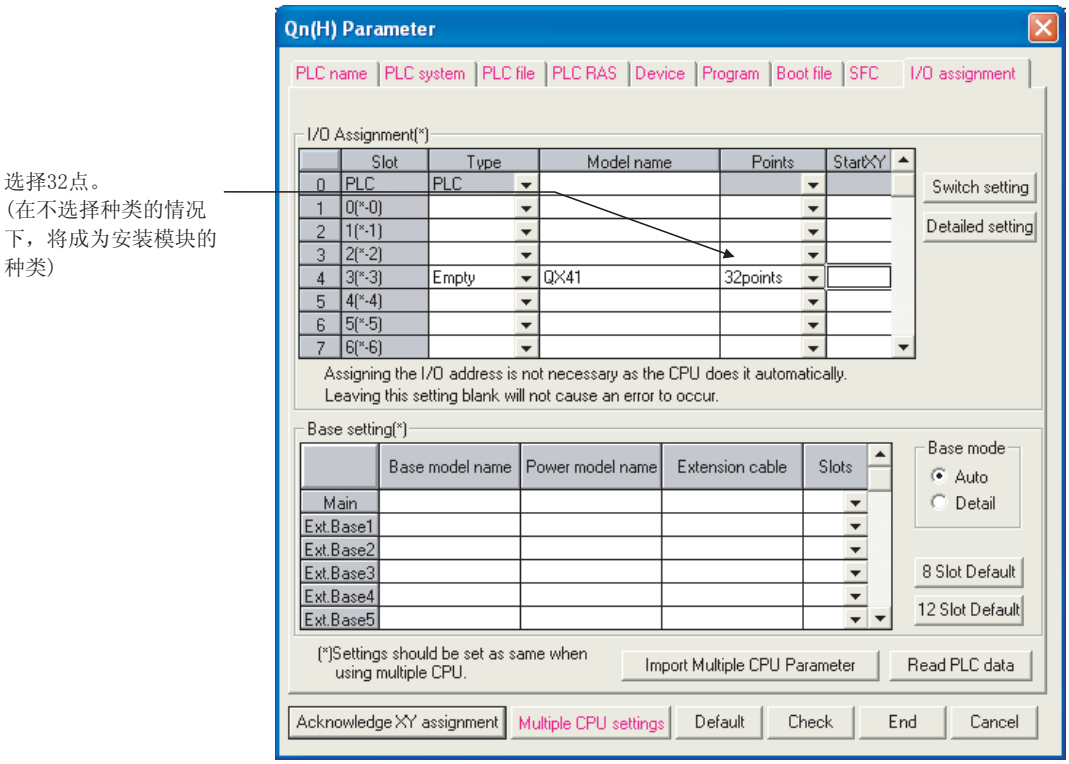


图 4.25 I/O 分配设定（变更插槽 3 空点数的情况）

(c) I/O 分配后的 I/O 地址号的分配

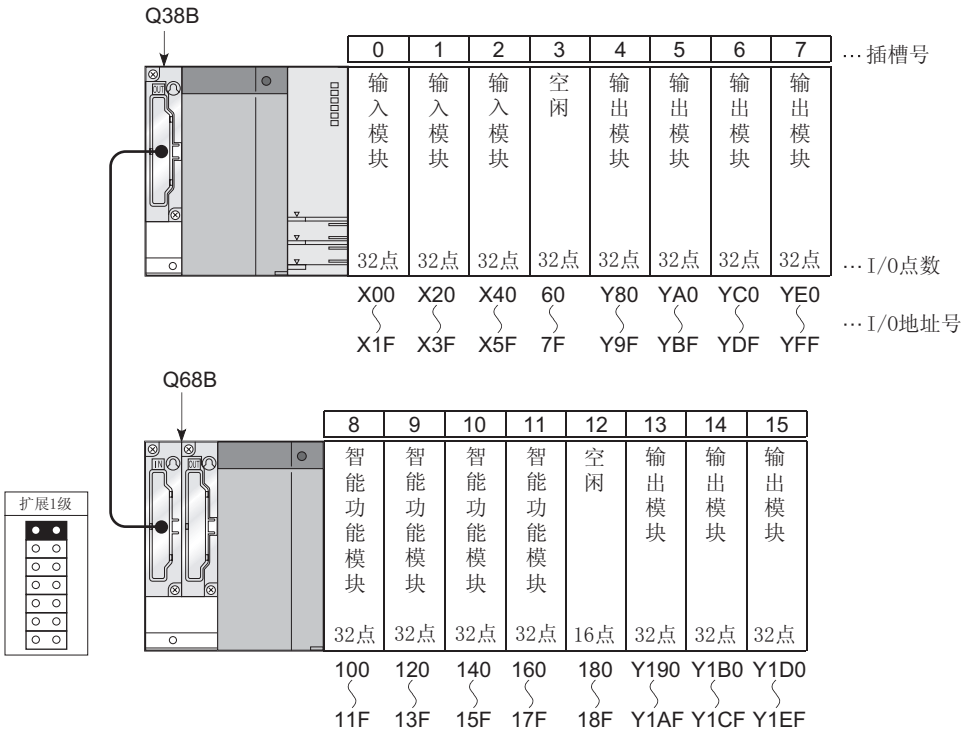


图 4.26 I/O 分配后的 I/O 地址号（变更空插槽的空点数后）

(2) 变更插槽的 I/O 地址号

根据在当前空插槽的位置上（插槽 No.3）安装 32 点 I/O 的输入模块，不要变更 No.4 以后的 I/O 地址号，请将插槽 No.3 的 I/O 地址号变更为 X200~21F。

(a) 系统构成与 I/O 分配前的 I/O 地址号的分配

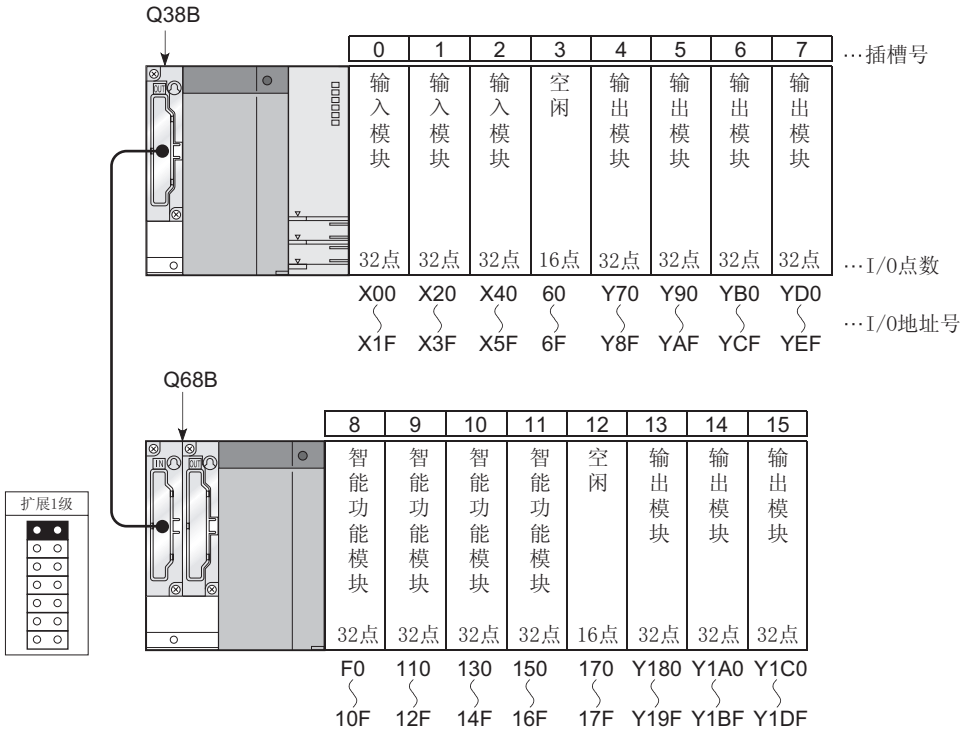


图 4.27 I/O 地址号的分配（插槽的 I/O 地址号变更前）

(b) 通过 GX Developer 进行 I/O 分配

在 GX Developer 进行 I/O 分配设定画面中将插槽 No.3 设定为 “200”，将插槽 No.4 设定为 “70”。

将起始I/O地址号设定为“200”。

将起始I/O地址号设定为“70”。
(如果不设定起始地址号，第3个插槽的下一个I/O地址号将被分配。)

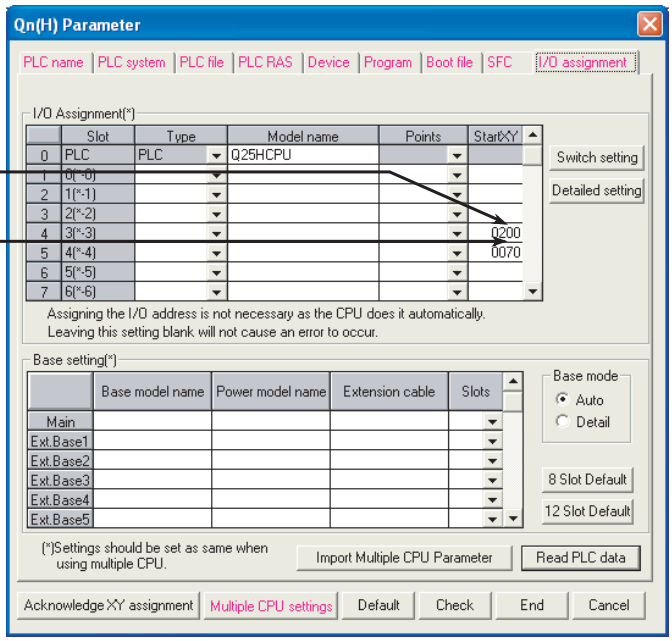


图 4.28 I/O 分配设定（变更插槽 3 的 I/O 地址号的情况）

(c) I/O 分配后的 I/O 地址号的分配情况

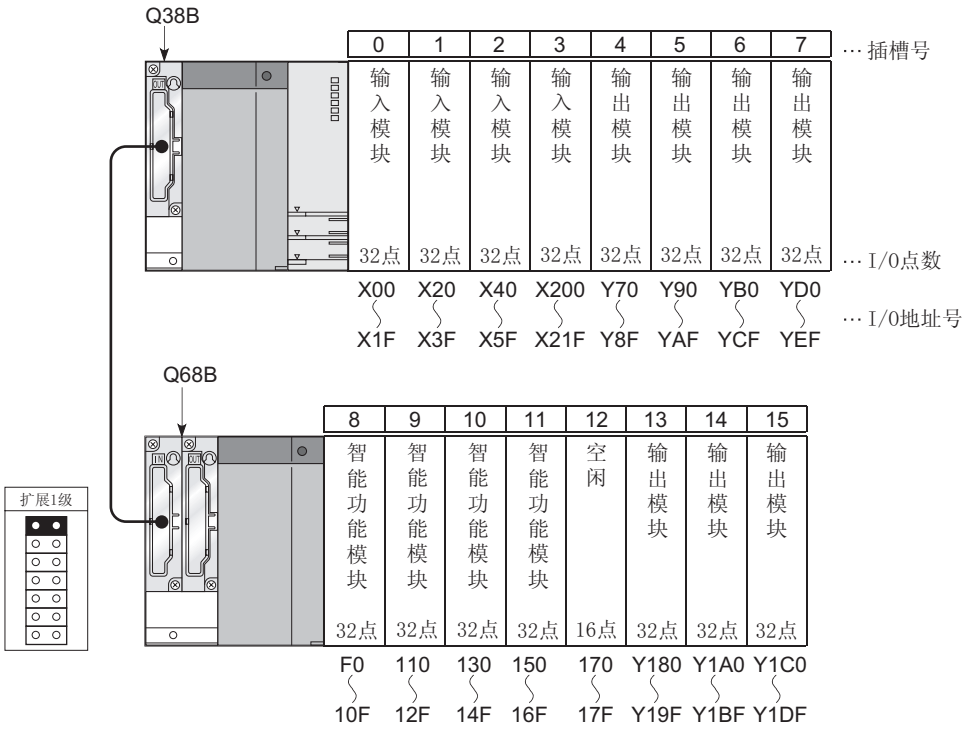


图 4.29 I/O 地址号的分配（插槽的 I/O 地址号变更后）

4.9 I/O 地址号的确认

通过 GX Developer 的系统监视，可以确认 CPU 模块的安装模块与 I/O 地址号。
(☞ 6.20 节)

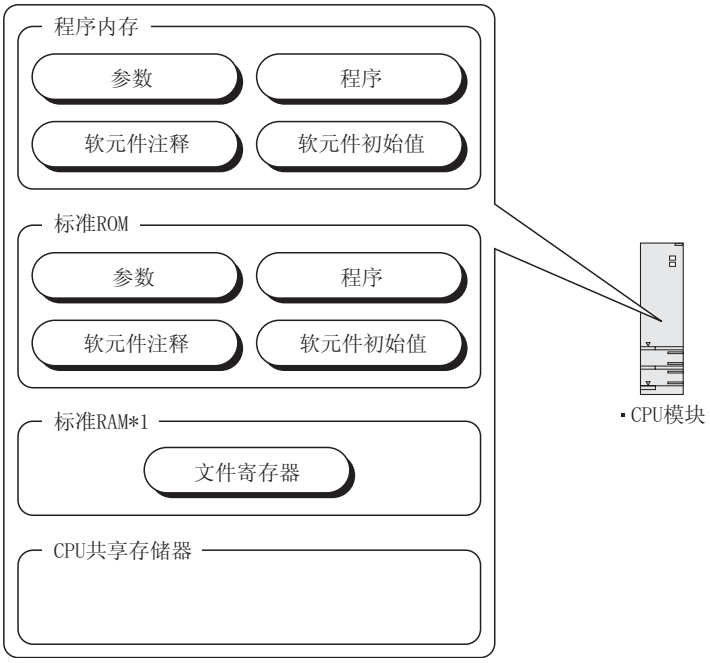
第 5 章　关于在 CPU 模块中使用的存储器与文件

5.1　基本模式 QCPU

5.1.1　存储器的构成与可以存储的数据

下面对基本模式 QCPU 中使用的存储器及可以存储的数据进行说明。

(1) 存储器的构成



*1 ： 在 Q00JCPU 中没有标准 RAM。
不能使用文件寄存器。

图 5.1 在基本模式 QCPU 中使用的数据

(a) 程序内存 (☞ 5.1.2 项)
程序内存是指为存储基本模式 QCPU 进行运算的程序的存储器。

(b) 标准 ROM(☞ 5.1.3 项)
标准 ROM 是指在基本模式 QCPU 中为进行引导运行的存储器。

(c) 标准 RAM(☞ 5.1.4 项)
标准 RAM 是指文件寄存器用的存储器。

(d) CPU 共享存储器 (☞ QCPU 用户手册 (多 CPU 系统篇))
CPU 共享存储器是指为进行多 CPU 系统的各个 CPU 模块间的数据的写入 / 读取的存储器。

(2) 各个存储器可以存储的数据

可以存储在程序内存、标准 RAM、标准 ROM 内的数据与驱动号如表 5.1 所示。

表 5.1 可以存储的数据与存储目标

	CPU 模块内部存储器			文件名与扩展名
	程序内存	标准 RAM*5	标准 ROM	
驱动 No.	0	3	4	
参数	◎	×	○	PARAM.QPA
智能功能模块的参数	○	×	○	IPARAM.QPA
顺控程序	◎*4	×	○*1	MAIN.QPG
SFC 程序	◎*4	×	○*1	MAIN-SFC.QPG
文件寄存器	×	○*3	×	MAIN.QDR
软元件注释	○*2	×	○*2	MAIN.QCD
软元件初始值	○	×	○	MAIN.QDI
用户设定的系统区域*6	○	×	×	—

◎：必要的数据，○：可以存储的数据，×：不可以存储的数据

*1：执行存储在标准 ROM 的程序时，有必要通过可编程控制器参数向程序内存中进行引导指定。
*2：只有从 GX Developer 才可以写入。
不能通过顺控程序指令使用软元件注释。
*3：标准 RAM 只能对文件寄存器存储一个文件。
关于可能存储的文件寄存器的个数，请参照第 2 章
*4：需要顺控程序、ST 程序或者 SFC 程序中任何一个的数据。
*5：Q00JCPU 中没有标准 RAM。
*6：设定系统中使用的区域。(☞ 5.1.2 项 (3)(b))

(3) 存储器的容量与格式化要求与否

存储器的容量与格式化要求与否如表 5.2 所示。

表 5.2 格式化的要求与否

	Q00JCPU	Q00CPU	Q01CPU	格式要求与否
程序内存	58k 字节	94k 字节	94k 字节	要求*1
标准 ROM	58k 字节	94k 字节	94k 字节	不要求
标准 RAM	无	128 字节		*2

*1：在使用之前，请务必通过 GX Developer 格式化。
*2：在使用功能版本 A 的 Q00CPU、Q01CPU 之前，必须通过 GX Developer 对标准 RAM 进行格式化。
在使用功能版本 B 的 Q00CPU、Q01CPU 之前，不需要进行格式化。
（如果通过 GX Developer 对功能版本 B 的 Q00CPU、Q01CPU 的标准 RAM 进行格式化，GX Developer 中将显示出错代码：4150H。）

要 点

将数据写入存储器时，根据写入 CPU 模块与存储区，存储容量的单位会有不同。
(☞ 5.4.4 项)

5.1.2 关于程序内存

(1) 关于程序内存

程序内存是指为存储基本模式 QCPU 进行运算的程序的存储器。
存储在标准 ROM 内的程序从程序内存中导出（读取）并进行运算。

(2) 可以存储的数据

程序内存中可以存储参数、智能功能模块的参数、程序、软元件注释、软元件初始值、用户设定的系统区域。
关于各个存储器中可以存储的数据请参阅 5.1.1 项 (2) 中的一览表。

要 点

存储在程序内存中数据的总容量在超出程序内存的程序容量时，请斟酌减少用户设定的系统区域。

(3) 在使用程序内存之前

在使用之前，请务必通过 GX Developer 格式化。

(a) 格式化的执行

格式化操作是指通过 GX Developer 的［在线］→［可编程控制器存储器格式化］对目标内存选择“程序内存 / 软元件内存”并进行格式化。

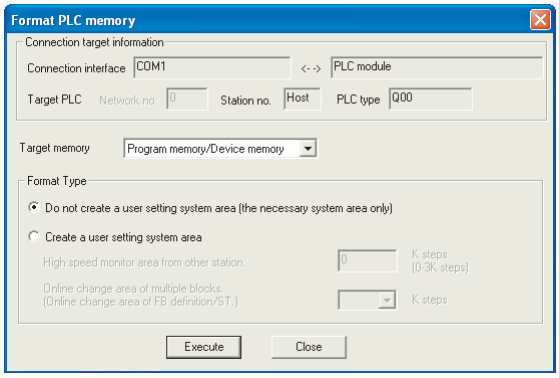


图 5.2 程序内存的格式化

(b) 用户设定的系统区域的创建

在对程序内存进行格式化时，设定用户设定的系统区域的容量。

1) 不创建用户设定的系统区域

在不创建用户设定的系统区域的状态下，进行格式化。

2) 创建用户设定的系统区域

在格式化时创建用户设定的系统区域。

用户设定的系统区域的形式如表 5.3 所示。

表 5.3 用户设定的系统区域的形式

系统区域形式	内容
为高速进行从其它站的监视而创建的区域	如果设定本区域，会使连接在串行口通讯模块等上的 GX Developer 的监视速度加快。 本区域在同时进行来自多个地方的监视时，使用来自连接在串行口通讯的上的 GX Developer 的监视数据登录。
为进行多块运行中写入而创建的区域 (FB 定义 /ST 运行中写入的区域)	如果设定本区域，将可以进行多块的运行中写入。 关于设定本区域时进行运行中写入的可能的块数的详细情况，请参照下述手册。 ☞ GX Developer 操作手册。

要 点

如果创建了用户设定的系统区域，可用区域的减少量仅为创建区域的所需的步数。存储器的容量可以在 GX Developer 的可编程控制器读出画面中确认。(☞ 本项 (3)(c))

(c) 对格式化后的存储容量的确认

存储容量通过 GX Developer 的 [在线]→[可编程控制器读出] 进行确认。

1) 在可编程控制器读出画面的对象内存中，选择 “程序内存 / 软元件内存”。

2) 点击 Free space volume 按钮。

3) 全空容量栏里显示出存储容量。

图 5.3 存储容量的确认步骤

(4) 对程序内存的写入

通过 GX Developer 的 [在线]→[可编程控制器写入] 进行程序内存数据的写入。

在可编程控制器读出画面的对象内存中，选择 “程序内存 / 软元件内存”进行可编程控制器的写入。

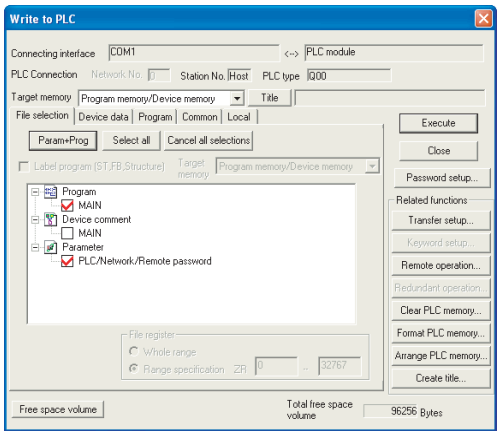


图 5.4 可编程控制器写入画面

要 点

文件的大小有最小单位。(☞ 5.4.4 项)

占用的存储容量有比实际文件大的情况。

5.1.3 关于标准 ROM

(1) 关于标准 ROM

标准 ROM 是指在基本模式 QCPU 中进行引导运行用的存储器。
标准 ROM 在没有电池备份的情况下保存程序与参数等时使用。
存储在标准 ROM 中的程序，从程序内存（☞ 5.1.2 项）中被引导出来（读出）并进行运算。

(2) 可以存储的数据

标准 ROM 可以存储参数、智能功能模块的参数、程序、软元件注释、软元件初始值。
各个存储器可以存储的数据一览如 5.1.1 项 (2) 所示，请参照。

(3) 存储容量的确认

存储容量通过 GX Developer 的 [在线]→[可编程控制器读出] 进行确认。

1) 在可编程控制器读出画面的对象内存中选择标准 ROM

2) 点击 Free space volume 按钮。

3) 在全部空闲容量栏中显示出存储容量。

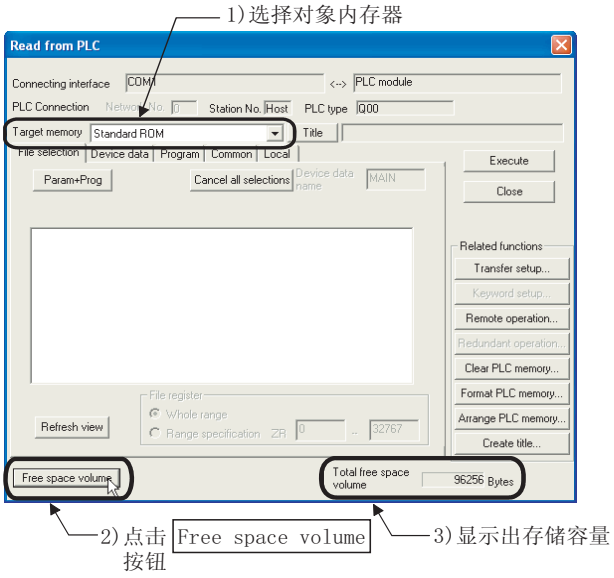


图 5.5 存储容量的确认步骤

(4) 对标准 ROM 的写入

标准 ROM 的写入通过 GX Developer 的 [在线]→[可编程控制器写入 (快闪卡)]→[程序内存的 ROM 化] 进行。(☞ 5.1.5 项)

要 点

文件的大小有最小单位。(☞ 5.4.4 项)
占用的存储容量有比实际文件大的情况。

(5) 存储在标准 ROM 中的程序的使用方法

由于存储在标准 ROM 中的程序不能执行运算的原因，因此，存储在标准 ROM 内的程序从程序内存里引导出 (读出) 使用。(☞ 5.1.5 项)

5.1.4 关于标准 RAM

(1) 关于标准 RAM

标准 RAM 是指文件寄存器用的存储器。(仅为 Q00CPU，Q01CPU)
标准 RAM 的文件寄存器可以与数据寄存器一样进行高速的存取。

(2) 存储的数据

标准 RAM 可以存储文件寄存器的一个文件。
各个存储器可以存储的数据如 5.1.1 项 (2) 所示，请参照。

(3) 使用标准 RAM 之前

- 在使用功能版本 A 的 Q00CPU、Q01CPU 之前，务必通过 GX Developer 对标准 RAM 进行格式化。
- 在使用功能版本 B 的 Q00CPU、Q01CPU 之前不需要进行格式化。
  (如果通过 GX Developer 对功能版本 B 的 Q00CPU、Q01CPU 执行了标准 RAM 的格式化，在 GX Developer 中将显示出错代码 :4150H。)

(a) 执行格式化

格式化通过 GX Developer 的 [在线]→[可编程控制器内存格式化] 选择对象内存的 “标准 RAM” 进行。

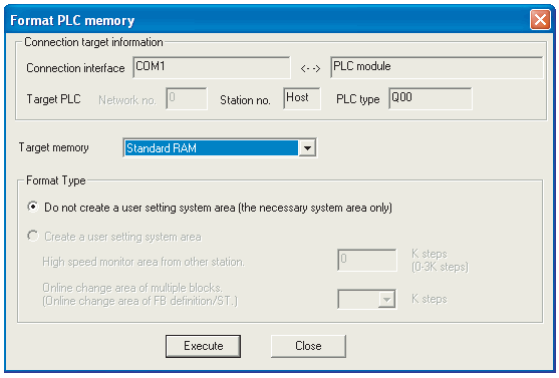


图 5.6 标准 RAM 的格式化

(b) 对格式化后存储容量的确认

存储容量通过 GX Developer 的 [在线]→[可编程控制器读出] 进行确认。

1) 可编程控制器读出画面的对象内存中选择 “标准 RAM”

2) 点击 Free space volume 按钮。

3) 在全部空闲容量栏中显示出存储容量。

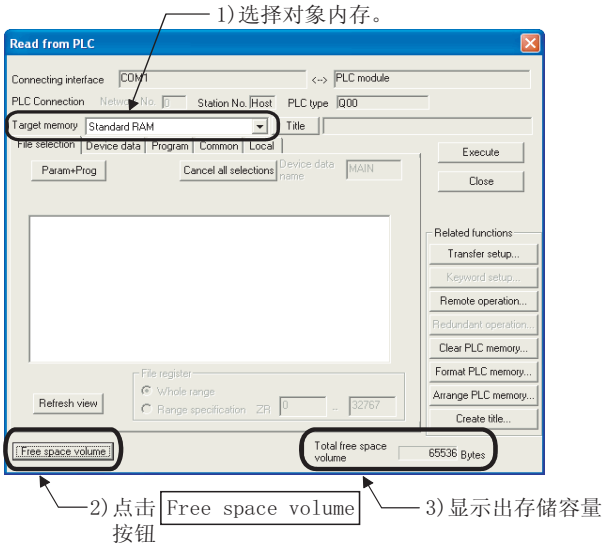


图 5.7 存储容量的确认步骤

(4) 对标准 RAM 进行的写入

标准 RAM 的数据写入通过 GX Developer 的 [在线]→[可编程控制器写入] 进行。

在可编程控制器写入画面的对象内存中选择 “标准 RAM”，进行可编程控制器写入。

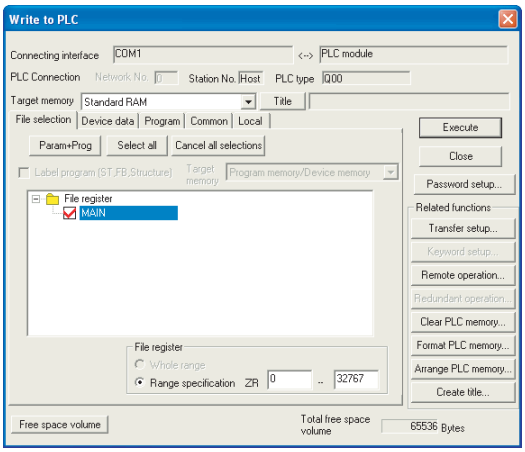


图 5.8 可编程控制器写入的画面

要 点

文件的大小有最小单位。(☞ 5.4.4 项)

占用的存储容量有比实际文件大的情况。

5.1.5 标准 ROM 的程序的执行（引导运行）与写入

(1) 标准 ROM 的程序的执行（引导运行）

(a) 关于标准 ROM 的程序的执行

基本模式 QCPU 进行存储在程序内存中的程序的运算。
存储在标准 ROM 中的程序不能进行运算。

存储在标准 ROM 中的程序从程序内存中引导（读出）并进行运算。

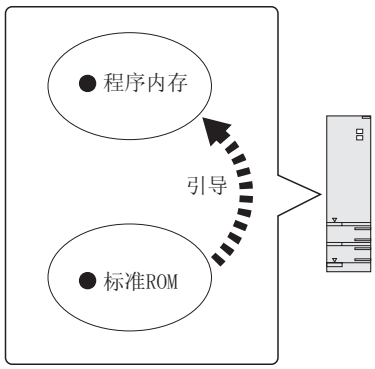


图 5.9 引导运行

(b) 进行引导运行前的准备工作

进行引导运行前的准备工作如下所示。

1) 通过 GX Developer 创建程序
制作进行引导运行的程序。

2) 通过 GX Developer 进行引导设定
通过可编程控制器参数的引导文件设定，设定 “进行从标准 ROM 的引导”。

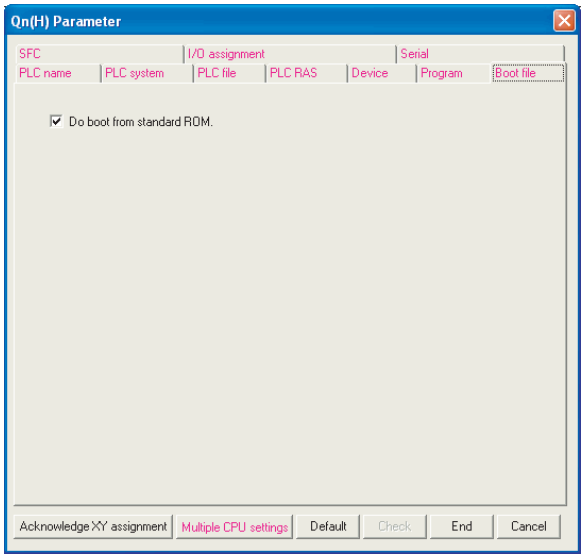


图 5.10 引导文件的设定

3) 通过 GX Developer 对标准 ROM 的写入
- 通过 GX Developer 的 [在线]→[可编程控制器写入] 将文件写入程序内存内。
- 通过 [在线]→[可编程控制器写入 (快闪卡)]→[程序内存的 ROM 化] 将程序内存中写入的文件写入标准 ROM 中。(☞ 本项 (2))

4) 程序的执行
通过基本模式 QCPU 的 RUN/STOP/RESET 开关复位时，从标准 ROM 进行引导。

5) 确认引导是否完成
引导是否正常完成通过特殊继电器 (SM660) 的状态进行确认。
关于特殊继电器的情况，请参照附录 1。

(c) 中止引导运行的操作
中止引导运行，通过写在程序内存中的参数程序执行运行时，请通过 GX Developer 进行以下的操作。

1) 进行程序内存的格式化

2) 进行 [在线]→[可编程控制器写入 (快闪卡)]→[程序内存的 ROM 化]。
(标准 ROM 的参数、顺控程序被消除)。

3) 向程序内存写入参数、顺控程序。

(d) 执行标准 ROM 的程序时的注意事项

1) 关于存储在标准 ROM 的文件
进行引导运行时，请将以下的文件存储在标准 ROM 中。
- 参数 *1
- 智能功能模块参数
- 程序 *2
- 软元件注释
- 软元件初始值

*1：请务必存储在标准 ROM 中。
*2：需要顺控程序、ST 程序与 SFC 程序中的任何一个。

2) 关于引导运行时的运行中写入
从标准 ROM 中进行引导运行时，即使进行程序内存的程序运行中写入，也不反映引导源的标准 ROM 的程序的变更内容。
因此，在变为 STOP 状态时，请向标准 ROM 中进行写入。(☞ 本项 (2))

3) 电源处于 OFF→ON 或者复位解除时程序的内容发生变化的情况
向程序内存中写入顺控程序过程中发生了可编程控制器电源 OFF→ON 或者复位解除时，程序内存内容写入的变化被视为引导运行。
请参照 [本项 (1)(c) 中止引导运行时的操作]，中止引导运行。

(2) 对标准 ROM 进行的写入

对标准 ROM 进行的写入是指将程序内存的文件批量的复制到标准 ROM 中。

(a) 写入前

在将文件写入标准 ROM 之前，请确认下述要点。

1) 标准 ROM 内的文件的保存

向标准 ROM 写入文件时，保存在标准 ROM 内的全部文件将自动消除。
在对标准 ROM 写入之前，请使用 GX Developer 的 [在线]→[可编程控制器写入]，预先将存储的文件通过 GX Developer 进行保存。

2) 写入文件的准备

向标准 ROM 写入文件时，由于存储在标准 ROM 内的全部文件被自动清除，请预先准备存储的所有的文件。

(b) 写入步骤

对向标准 ROM 写入文件的步骤进行说明。

1) 选择 GX Developer 的 [在线]→[可编程控制器写入 (快闪卡)]→[程序内存的 ROM 化]

2) 显示程序内存的 ROM 化的画面。

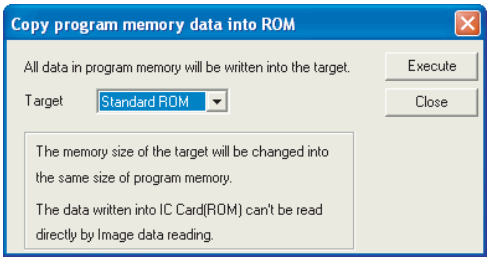


图 5.11 程序内存的 ROM 化的画面

3) 选择写入对象，将程序内存的文件复制到标准 ROM 中。

(3) 对标准 ROM 的文件进行的追加、变更

向标准 ROM 写入文件时，由于存储在标准 ROM 内的全部文件被自动清除，因此，不能对存储的文件直接进行追加、变更。

请通过下述方法进行追加、变更。

1) 通过 GX Developer 的 [在线]→[可编程控制器读出] 来读出标准 ROM 的所有文件。

2) 对读出的文件进行追加、变更。

3) 将进行了追加变更的文件写入到程序内存中。

4) 通过 [在线]→[可编程控制器写入 (快闪卡)]→[程序内存的 ROM 化] 将文件复制到标准 ROM 中。

(4) 注意事项

(a) 关于 GX Developer 的通讯时间的检查时间的设定

向标准 ROM 写入文件时，GX Developer 的通讯检查时间设定在 180 秒以下的情况下，进行 180 秒的检查。

(b) 经由 CC-Link 从其它站的 GX Developer 进行写入的情况

向标准 ROM 写入文件时由于处理需要花费时间，请将 CC-Link 的 CPU 监视时间设定 (SWOA) 设定在 180 秒以上。

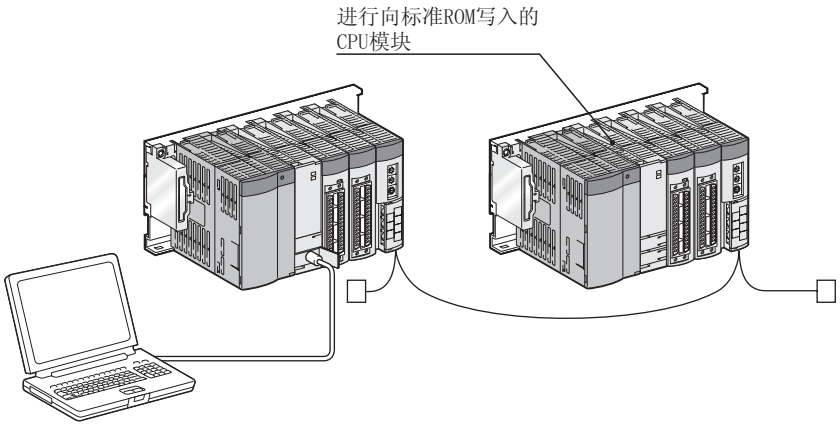


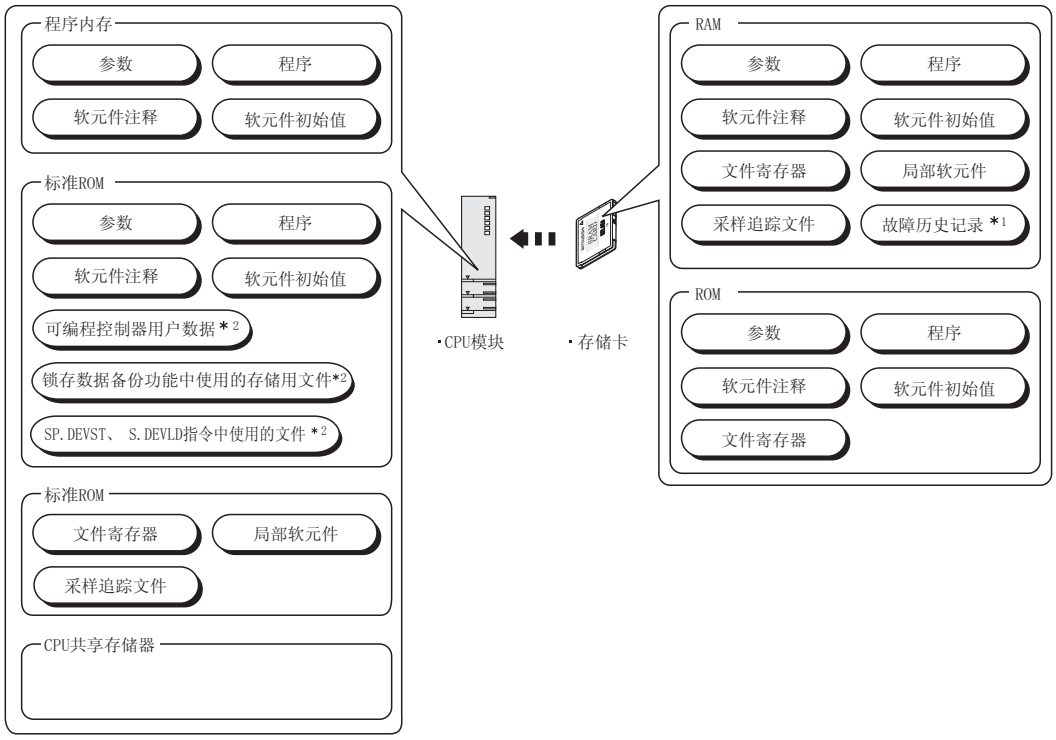
图 5.12 从其它站的 GX Developer 进行的写入 (经由 CC-Link)

5.2 高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU

5.2.1 存储器的构成与可以存储的数据

对在高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 中使用的存储器及可以存储的数据进行说明。

(1) 存储器的构成



*1: 在通用型 QCPU 中，不能将故障历史记录存储到存储卡中。
*2: 仅在使用通用型 QCPU 时才可以存储。

图 5.13 高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 的存储器构成

(a) 程序内存 (☞ 5.2.2 项)
程序内存是指，为进行高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 运算而存储程序的存储器。
存储在标准 ROM 注 5.1、存储卡中的程序被引导（读取）到程序内存中并进行运算。(☞ 5.2.9 项)

通用 UD 注 5.1

(b) 标准 ROM(☞ 5.2.4 项)
标准 ROM 是存储参数、程序等数据的存储器。
在高性能模式 QCPU、过程 CPU、冗余 CPU 中，也可以将其指定为引导源的存储驱动器使用。注 5.1

通用 UD 注 5.1

(c) 标准 RAM(☞ 5.2.5 项)
标准 RAM 是指为使用未安装存储卡的文件寄存器、局部软元件的存储器。

(d) 存储卡 (☞ 5.2.6 项)
存储卡在高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 的内置存储器扩展时使用。
可以使用的存储卡有 SRAM 卡、Flash 卡、ATA 卡。

(e) CPU 共享存储器 (☞ QCPU 用户手册（多 CPU 系统篇))
存储卡在高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 的内置存储器扩展时使用。

通用 UD 注 5.1

在通用型 QCPU 中，不能执行从标准 ROM 向程序内存的引导。(☞ 5.2.4 项)

(2) 可以存储在各个存储器的数据

可以存储在程序内存、标准 RAM、标准 ROM 以及存储卡的数据与驱动 No. 如表 5.4 所示。

表 5.4 可以存储的数据与存储对象

	CPU 模块内置存储器			存储卡 (RAM)	存储卡 (ROM)		文件名及扩展名	备注
	程序内存	标准 RAM	标准 ROM	SRAM 卡	Flash 卡	ATA 卡		
驱动号 *9	0	3	4	1	2			
参数	○	×	○	○	○	○	PARAM.QPA	1 数据 / 驱动器
智能功能模块参数 *12	○	×	○	○	○	○	IPARAM.QPA	1 数据 / 驱动器
程序	◎	×	○*1	○*1	○*1	○*1	***.QPG	--
软元件注释	○*2	×	○*3	○*3	○*3	○*3	***.QCD	--
软元件初始值	○	×	○	○	○	○	***.QDI	--
软元件数据	×	×	△*11	×	×	×	***.QST	--
文件寄存器	×	○*5*6	×	○	○*4	×	***.QDR	--
局部软元件	×	○*5	×	○	×	×	***.QDL	1 数据 /CPU 模块
采样追踪文件	×	○*5*10	×	○	×	×	***.QTD	--
故障历史记录数据 *13	×	×	×	○	×	×	***.QFD	--
可编程控制器用户数据	×	×	△*11	×	×	○*7	***.***	--
用户设置的系统区 *8	○	×	×	×	×	×	--	--

◎: 必要数据 , ○: 可以存储的数据 , ×: 不可以存储的数据

*1 : 执行存储在标准 ROM、存储卡中的程序时，需要通过可编程控制器参数向程序内存中进行引导指定。但是，在通用型 QCPU 中，不能执行从标准 ROM 向程序内存的引导。(☞ 5.2.3 项)

*2 : 只有通过 GX Developer 才可以写入。
不能通过顺控程序指令使用软元件注释。

*3 : 从顺控程序中读出需要数个扫描。

*4 : 通过顺控程序只能进行读出。
不能通过顺控程序进行写入。

*5 : 标准 RAM 中可以存储文件寄存器、局部软元件以及采样追踪文件的各一个文件。

*6 : 关于可以存储的文件寄存器的点数，请参照第 2 章。

*7 : 通过下述指令可以进行数据的读写。
• SP.FREAD (从存储卡的指定文件中批量读出)
• SP.FWRITE (向存储卡的指定文件中批量写入)

*8 : 设定系统中使用的区域。(☞ 5.2.2 项 (3)(b))

*9 : 驱动器 No. 是用于指定从使用了顺控程序与 MC 协议的外部设备等进行读出 / 写入的对象内存。
在 GX Developer 中，由于指定了对象内存名，所以无需理会驱动器 No.。

*10: 将采样追踪文件存储到标准 RAM 中时，应确认 CPU 模块及 GX Developer 的版本。(☞ 附录 4)

*11 : 只有通用型 QCPU 才可以存储到标准 ROM 中。

*12 : 应将智能功能模块参数与参数存储到同一个驱动器中。
如果存储到不同的驱动器中，智能功能模块参数将不能变为有效。

*13 : 在通用型 QCPU 中，不能将故障历史记录存储到存储卡中。(☞ 6.18.2 项)

(3) 存储器容量与是否格式化

各个存储器容量与是否进行格式化如表 5.5 所示。

高性能

注 5.2

表 5.5 是否进行格式化

		Q02CPU	Q02HCPU	Q06HCPU	Q12HCPU	Q25HCPU	Q12PHCPU	Q25PHCPU	是否需要格式化
程序内存		112k 字节 (28k 步)	112k 字节 (28k 步)	240k 字节 (60k 步)	496k 字节 (124k 步)	1008k 字节 (252k 步)	496k 字节 (124k 步)	1008k 字节 (252k 步)	*1
标准 ROM		112k 字节	112k 字节	240k 字节	496k 字节	1008k 字节	496k 字节	1008k 字节	不需要
标准 RAM *注 5.2*		64k 字节	128k 字节		256k 字节				*1
存储卡	SRAM 卡	Q2MEM-1MBS : 1M 字节 Q2MEM-2MBS : 2M 字节							需要 (通过 GX Developer 进行)
	Flash 卡	Q2MEM-2MBF : 2M 字节 Q2MEM-4MBF : 4M 字节							不需要
	ATA 卡	Q2MEM-8MBA : 8M 字节 Q2MEM-16MBA : 16M 字节 Q2MEM-32MBA : 32M 字节							需要 (通过 GX Developer 进行)

		Q12PRHCPU	Q25PRHCPU	Q02UCPU	Q03UDCPU	Q04UDHCPU	Q06UDHCPU	是否需要格式化
程序内存		496k 字节 (124k 步)	1008k 字节 (252k 步)	80k 字节 (20k 步)	120k 字节 (30k 步)	160k 字节 (40k 步)	240k 字节 (60k 步)	*1
标准 ROM		496k 字节	1008k 字节	512k 字节	1024k 字节	1024k 字节	1024k 字节	不需要
标准 RAM *注 5.2*		256k 字节	256k 字节	128k 字节	192k 字节	256k 字节	768k 字节	*1
存储卡	SRAM 卡	Q2MEM-1MBS : 1M 字节 Q2MEM-2MBS : 2M 字节		Q2MEM-1MBS : 1M 字节 Q2MEM-2MBS : 2M 字节 Q3MEM-4MBS : 4M 字节 Q3MEM-8MBS : 8M 字节				需要 (通过 GX Developer 进行)
	Flash 卡	Q2MEM-2MBF : 2M 字节 Q2MEM-4MBF : 4M 字节		Q2MEM-2MBF : 2M 字节 Q2MEM-4MBF : 4M 字节				不需要
	ATA 卡	Q2MEM-8MBA : 8M 字节 Q2MEM-16MBA : 16M 字节 Q2MEM-32MBA : 32M 字节		Q2MEM-8MBA : 8M 字节 Q2MEM-16MBA : 16M 字节 Q2MEM-32MBA : 32M 字节				需要 (通过 GX Developer 进行)

*1 : 存储器在初始状态或者是切断电池供应中如果出现存储内容不稳定，虽然可编程控制器的电源处于 OFF→ON 或者在复位时自动进行格式化，但请务必在使用之前通过 GX Developer 进行格式化。

要 点

1. 将文件写入各个存储器时，根据写入 CPU 模块与存储区域，存储容量的单位会有不同。(☞ 5.4.4 项)
2. 存储容量的计算为：一步等于 4 字节。

高性能

注 5.2

在高性能模式 QCPU 中通过功能升级标准 RAM 的容量会有不同。(☞ 附录 4.2)

5.2.2 关于程序内存

(1) 关于程序内存

程序内存是为进行高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 运算而存储程序的存储器。

在通用型 QCPU 中，程序的运算是从程序内存传送至程序高速缓冲存储器中进行。(☞ 5.2.3 项)

通用 UD 注 5.3

存储在标准 ROM 注 5.3 、存储卡中的程序被引导（读取）到程序内存中进行运算。(☞ 5.2.9 项)

(2) 可以存储的数据

程序内存可以存储参数、智能功能模块参数、程序、软元件注释、软元件初始值、用户设定的系统区域。

关于各个存储器可以存储的数据请参阅 5.2.1 项 (2) 的一览表。

要 点

程序内存的容量与各个模块的程序容量 (☞ 5.2.1 项 (3) 相同。

存储在程序内存中的数据的总容量在超出程序内存的程序容量时请争取如下措施：

1. 减少用户设定的系统区域
2. 将程序以外的数据移到标准 ROM 或者是存储卡上。

(3) 在程序使用之前

在使用之前，请务必通过通过 GX Developer 对程序内存进行格式化。

(a) 执行格式化

格式化通过 GX Developer 的 [在线]→[可编程控制器内存格式化] 选择对象内存的 “程序内存 / 软元件内存” 后进行。

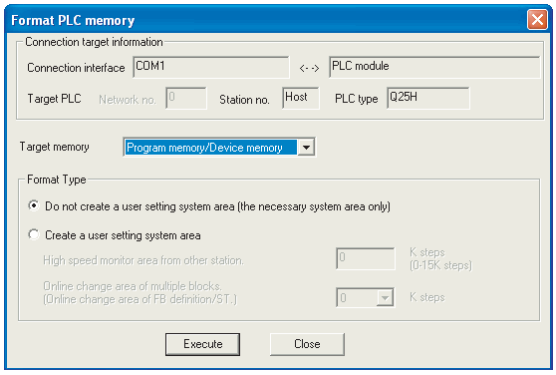


图 5.14 程序内存的格式化

通用 UD 注 5.3

在通用型 QCPU 中，不能执行从标准 ROM 向程序内存的引导。(☞ 5.2.3 项)

(b) 用户设定的系统区域的创建

在对程序内存进行格式化时，设定用户设定的系统区域的容量。

1) 不创建用户设定的系统区域

在不创建用户设定的系统区域的状态下，进行格式化。

2) 创建用户设定的系统区域

在格式化时创建用户设定的系统区域。

用户设定的系统区域的如表 5.6 所示。

表 5.6 用户设定的系统区域的形式

系统区域形式	内容
为高速进行从其它站的监视而创建的区域	如果设定本区域，会使来自于连接在串行口通讯等上的 GX Developer 的监视速度加快。 在通过 GX Developer 同时使用 RS-232 与 USB 时，本区域被用于来自连接在串行口通讯模块上的 GX Developer 的监视数据登陆。
为进行多块运行中写入而创建的区域 (FB 定义 /ST 的运行中写入用的区域)	• 高性能模式 QCPU、过程 CPU、冗余 CPU 时 如果设定本区域，将可以进行多块的运行中写入。 • 通用型 QCPU 时 即使未设定本区域，也可以进行多块的运行中写入。(不需要进行本区域的设定。) 关于可以进行运行中写入的块数的详细情况，请参照下述手册。 ☞ GX Developer 操作手册。

要 点

如果创建用户设定的系统区域，所减少的可以使用的区域仅为创建区域所需的步数。

存储容量可以在 GX Developer 的可编程控制器读出画面中确认。(☞ 本项 (3)(c))

(c) 对格式化后的存储容量的确认

存储容量可以通过 GX Developer 的 [在线]→[可编程控制器读出] 进行确认。

1) 在可编程控制器读出画面的对象内存中，选择 “程序内存 / 软元件内存”。

2) 点击 Free space volume 按钮。

3) 在全部空闲容量栏中显示出存储容量。

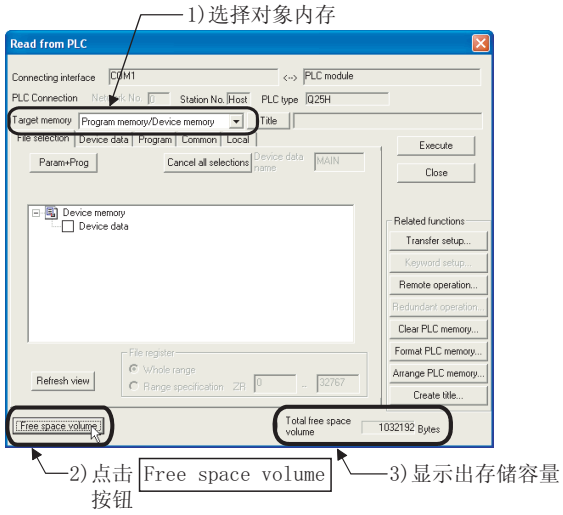


图 5.15 存储容量的确认步骤

(4) 对程序内存的写入

对程序内存数据的写入通过 GX Developer 的 [在线]→[可编程控制器写入] 进行。在可编程控制器写入画面的对象内存中，选择 “程序内存 / 软元件内存”进行可编程控制器的写入。

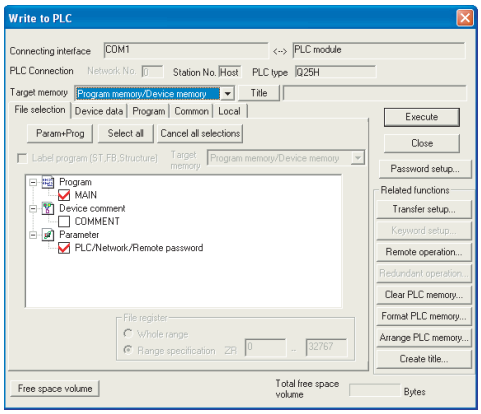


图 5.16 可编程控制器写入画面

要 点

文件的大小有最小单位。(☞ 5.4.4 项)

占用的存储容量有比实际文件大的情况。

如果文件数目增多，请注意占用的存储容量与实际的文件大小的差额将会变大。

5.2.3 关于程序高速缓冲存储器（仅通用型 QCPU）

高性能 注 5.4
过程 注 5.4
冗余 注 5.4

(1) 程序高速缓冲存储器的概要

程序高速缓冲存储器是在通用型 QCPU 中进行程序运算的存储器。程序的运算是将程序内存中存储的程序传送至程序高速缓冲存储器中进行。
(☞ 本项 (3))

从程序内存至程序高速缓冲存储器的传送时机如下所示。

- 电源 ON 时的初始化处理
- 复位解除时的初始化处理

程序的运算流程如图 5.17 所示。

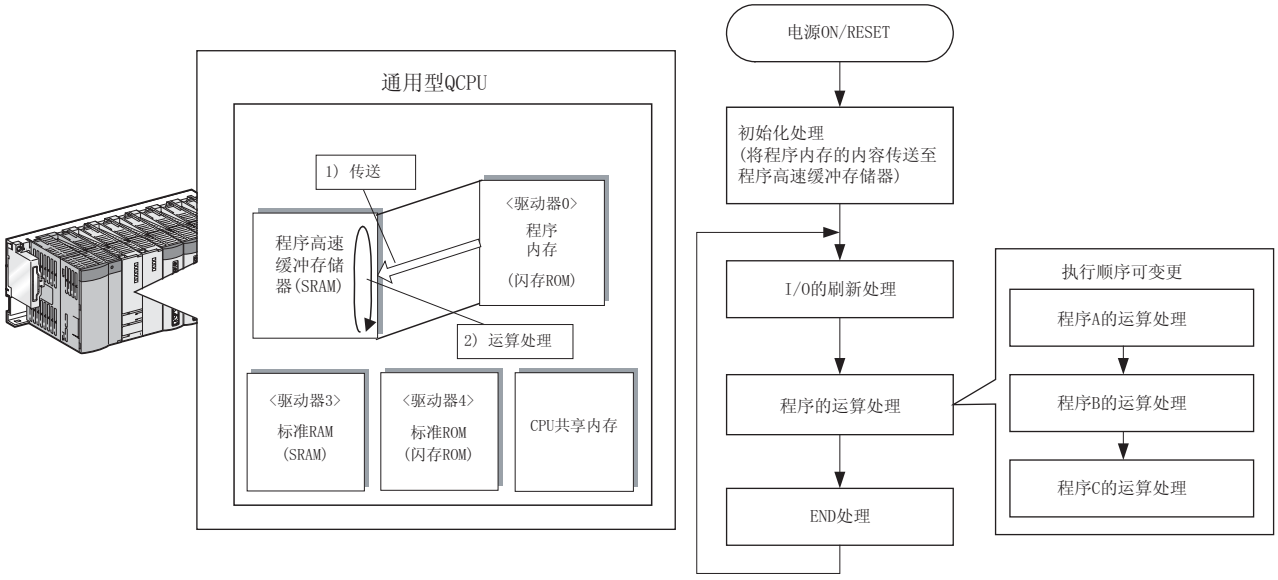


图 5.17 程序运算的流程

高性能 注 5.4　过程 注 5.4　冗余 注 5.4

在高性能模式 QCPU、过程 CPU、冗余 CPU 中，不使用程序高速缓冲存储器，因此无需理会本项的内容。

(2) 程序的写入

通过 GX Developer 进行写入操作时，程序、参数等被写入到 CPU 模块的程序高速缓冲存储器中后，将被立即传送到程序内存中。

程序的写入流程如图 5.18 所示。

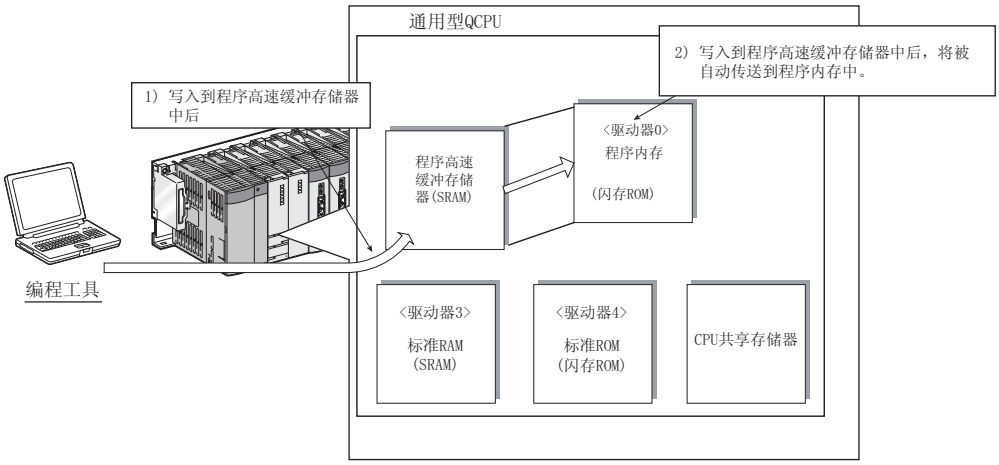


图 5.18 程序的写入流程

(3) 通过 GX Developer 传送至程序内存

通过 GX Developer 的［在线］→［程序内存批量传送］也可以进行从程序高速缓冲存储器至程序内存的批量传送。

此外，在 CPU 模块的 RUN 状态下进行至程序内存的批量传送时，需要耗费表 5.7 所示的传送时间。

表 5.7 至程序内存的批量传送的传送时间

CPU 模块	传送时间
Q02UCPU	Ts × 320 + 4.8 (s)
Q03UDCPU, Q04UDHCPU, Q06UDHCPU	Ts × 260 + 4.7 (s)

Ts: 扫描时间 (s)

(4) 至程序内存的传送状况确认

至程序内存的传送状况可通过 GX Developer 的进程画面、特殊继电器以及特殊寄存器确认。

(a) 通过进程画面的确认方法

GX Developer 的进程画面如图 5.19 所示。

图 5.19 至程序内存的传送状况的进程画面

(b) 通过特殊继电器、特殊寄存器的确认方法

至程序内存的传送状况可通过 SM681、SD681 进行确认。

5.2.4 关于标准 ROM

(1) 关于标准 ROM

通用 UD 注 5.5

标准 ROM 是存储参数、程序等的数据的存储器。
在高性能模式 QCPU、过程 CPU、冗余 CPU 中，也可以将其指定为引导源的存储驱动器使用。*注 5.5*
标准 ROM 被用于在没有电池备份的情况下保存程序与参数等。

(2) 可以存储的数据

标准 ROM 可以存储参数、智能功能模块的参数、程序、软元件注释、软元件初始值。
各个存储器可以存储的数据如 5.2.1 项 (2) 中一览表所示，请参照。

(3) 存储容量的确认

存储容量可以通过 GX Developer 的 [在线]→[可编程控制器读出] 进行确认。

1) 在可编程控制器读出画面的对象内存中，选择 “标准 ROM”。

2) 点击 Free space volume 按钮。

3) 在全部空闲容量栏中显示出存储容量。

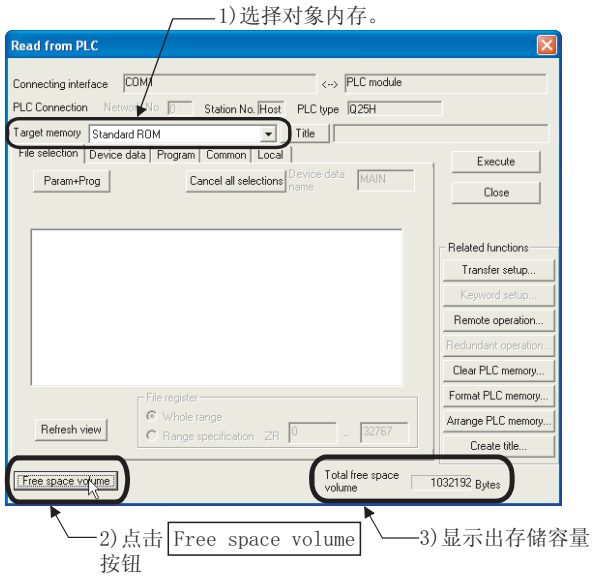


图 5.20 存储容量的确认步骤

通用 UD 注 5.5

在通用型 QCPU 中，不能执行从标准 ROM 向程序内存的引导。(☞ 5.2.3 项)

(4) 对标准 ROM 的写入

至标准 ROM 的写入方法根据 CPU 模块的不同而有所不同。

(a) 高性能模式 QCPU、过程 CPU、冗余 CPU 的情况

- 通过 GX Developer 的 [在线]→[可编程控制器写入 (快闪卡)]→[程序内存的 ROM 化] 将程序内存的数据批量的复制到标准 ROM 中的方法。(☞ 5.2.7 项)
- 通过 GX Developer 的 [在线]→[可编程控制器写入 (快闪卡)]→[可编程控制器写入 (快闪卡)] 进行写入的方法。(☞ 5.2.7 项)
- 使用存储卡存储卡→标准 ROM 全部数据自动写入功能进行写入的方法。(☞ 5.2.8 项)

(b) 通用型 QCPU 的情况

通过 GX Developer 的 [在线]→[可编程控制器写入] 的写入方法 (☞ 5.2.2 项)

应在对象内存中选择 “标准 ROM” 后进行。

要 点

文件的大小有最小单位。(☞ 5.4.4 项)

占用的存储容量有比实际文件大的情况。

如果文件数目增多，请注意占用的存储容量与实际的文件大小的差额将会变大。

备 注

关于标准 ROM 写入方法的选择标准、用途、步骤、注意事项的详细情况，请参照上述标明的参照对象。

(5) 存储在标准 ROM 中程序的使用方法

通用 UD 注 5.6

由于存储在标准 ROM 中的程序不能执行运算，因此，将存储在标准 ROM 内的程序引导 (读取) 注 5.6 到程序内存中使用。(☞ 5.2.9 项)

通用 UD 注 5.6

在通用型 QCPU 中，不能执行从标准 ROM 向程序内存的引导。(☞ 5.2.3 项)

5.2.5 关于标准 RAM

(1) 关于标准 RAM

标准 RAM 是指为使用未安装存储卡的文件寄存器、局部软元件而使用的存储器。
标准 RAM 作为文件寄存器使用时，可以与数据寄存器一样进行高速的存取。

(2) 存储的数据

标准 RAM 可以存储文件寄存器与局部软元件的各一个文件（共两个文件）。
各个存储器可以存储的数据如 5.2.1 项 (2) 的一览表所示，请参照。

要 点

1. 要存储到标准 RAM 中的文件的容量超出标准 RAM 的容量时，请采取如下措施：
   - 将文件存储到存储卡中
   - 酌情减少文件寄存器、局部软元件、采样追踪文件的点数。
     但是，在将文件寄存器存储到存储卡的情况下，请注意与标准 RAM 相比其存取速度将会变慢。
2. 将采样追踪文件存储到标准 RAM 中时，应确认 CPU 模块及 GX Developer 的版本。(☞附录 4)

(3) 使用标准 RAM 之前

在使用之前，请务必通过 GX Developer 对标准 RAM 进行格式化。

(a) 执行格式化

在 GX Developer 的 [在线]→[可编程控制器内存格式化] 的对象内存里选择“标准 RAM”，进行格式化。

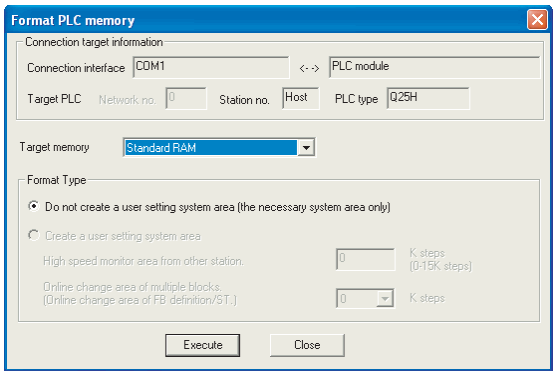


图 5.21 标准 RAM 的格式化

(b) 格式化后的存储容量的确认

存储容量可以通过 GX Developer 的 [在线]→[可编程控制器读出] 进行确认。

1) 可编程控制器读出画面的对象内存中选择 “标准 RAM”

2) 点击 Free space volume 按钮。

3) 在全部空闲容量栏中显示出存储容量。

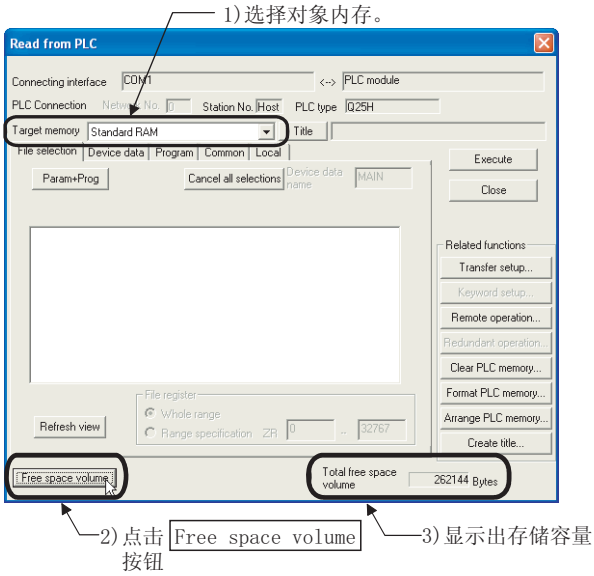


图 5.22 存储容量的确认步骤

(4) 对标准 RAM 进行的写入

标准 RAM 的数据写入通过 GX Developer 的 [在线]→[可编程控制器写入] 进行。

在可编程控制器写入画面的对象内存中选择 “标准 RAM”，进行可编程控制器写入。

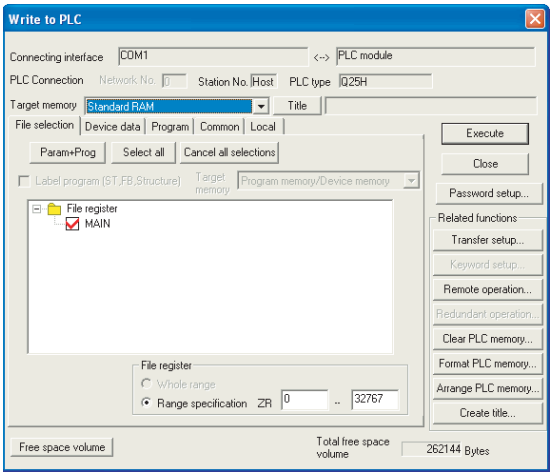


图 5.23 可编程控制器写入的画面

要 点

文件的大小有最小单位。(☞ 5.4.4 项)
占用的存储容量有比实际文件大的情况。
如果文件数目增多，请注意占用的存储容量与实际的文件大小的差额将会变大。

5.2.6 关于存储卡

(1) 存储卡

存储卡在 CPU 模块的存储器扩展时使用。
可以使用的存储卡有 SRAM 卡、Flash 卡、ATA 卡。

(a) SRAM 卡

通过顺控程序可以对 SRAM 卡中存储的文件寄存器进行写入 / 读出。
SRAM 卡在下述情形下使用。

- 文件寄存器的点数多于标准 RAM 的容量时
- 使用采样追踪功能时 (☞ 6.14 节)
- 希望保存 17 个以上的故障历史记录数据时注 5.7(☞ 6.18 节)

通用 UD 注 5.7

如果将 SRAM 卡作为文件寄存器使用，通过顺控程序最多可进行如下点数的写入 / 读出。

- 高性能模式 QCPU、过程 CPU、冗余 CPU .... 最多 1017k 点
- 通用型 QCPU ..... 最多 4086k 点

(b) Flash 卡

只有通过顺控程序才能读出。
数据的写入通过 GX Developer 进行，通过顺控程序读出并使用。
在不进行数据变更时使用。
最多可以存储以下点数的文件寄存器。

- 高性能模式 QCPU、过程 CPU、冗余 CPU .... 最多 1018k 点
- 通用型 QCPU ..... 最多 2039k 点

(c) ATA 卡

使用可编程控制器用户数据 (通用数据)。
通过顺控程序使用文件存取指令 (FWRITE 指令等)，以 CSV 格式 /2 进制格式对 ATA 卡的可编程控制器用户数据进行存取。

通用 UD 注 5.6 在通用型 QCPU 中，不能将故障历史记录数据存储到存储卡中。(☞ 6.18.2 项)

(2) 可以存储的数据

存储卡可以存储的数据如表 5.8 所示。

表 5.8 存储卡可以存储的数据

数据名	存储卡 (RAM)	存储卡 (ROM)	
	SRAM 卡	Flash 卡	ATA 卡
参数	○	○	○
智能功能模块的参数	○	○	○
程序	○	○	○
软元件注释	○	○	○
软元件初始值	○	○	○
文件寄存器	○	○	×
局部软元件	○	×	×
采样追踪文件	○	×	×
故障历史记录数据 *注 5.8*	○	×	×
可编程控制器用户数据	×	×	○
用户设定的系统区域	×	×	×

通用 UD 注 5.8

○：可以存储的数据，×：不可以存储的数据

各个存储器可以存储的数据一览如 5.2.1 项 (2) 的所示，请参照。

通用 UD 注 5.8

在通用型 QCPU 中，不能将故障历史记录数据存储到存储卡中。(☞ 6.18.2 项)

(3) SRAM 卡、ATA 卡使用之前

在使用之前，务必通过 GX Developer 对 SRAM 卡、ATA 卡进行格式化。

(a) 格式化的执行

格式化通过 GX Developer 的 [在线]→[可编程控制器内存格式化] 进行。

对 SRAM 卡进行格式化时，在对象内存中选择 “存储卡 (RAM)” 进行。

对 ATA 卡进行格式化时，在对象内存中选择 “存储卡 (ROM)” 进行。

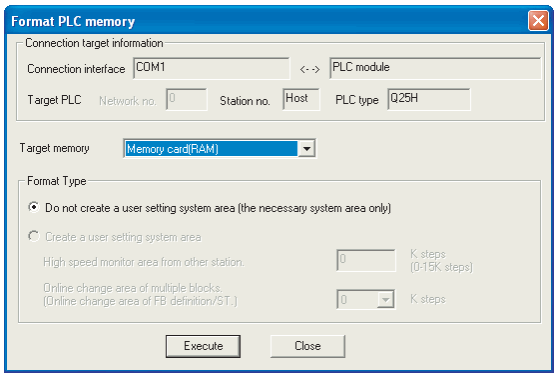


图 5.24 SRAM 卡、ATA 卡的格式化

要 点

1. 请不要通过 GX Developer 以外的设备对 ATA 卡进行格式化。
   通过 Microsoft® Windows® 的格式化功能等进行格式化的情况下，安装在 CPU 模块上有不能使用的情况。
2. 如果进行 SRAM 卡以及 ATA 卡的格式化，由于自动预留存储卡信息区域的原因，因此，容量将减少相当于存储卡信息区域的容量。
   存储容量可以通过 GX Developer 的可编程控制器读出画面进行确认。
   (☞ 本项 (3)(b))

备 注

Flash 卡没有必要进行格式化。

(b) 对格式化后的存储容量的确认

存储容量可以通过 GX Developer 的 [在线]→[可编程控制器读出] 进行确认。

1) 在可编程控制器读出画面的对象内存中，选择 “标准 (RAM)” 或者 “存储卡 (ROM)”。

2) 点击 Free space volume 按钮。

3) 在全部空闲容量栏中显示出存储容量。

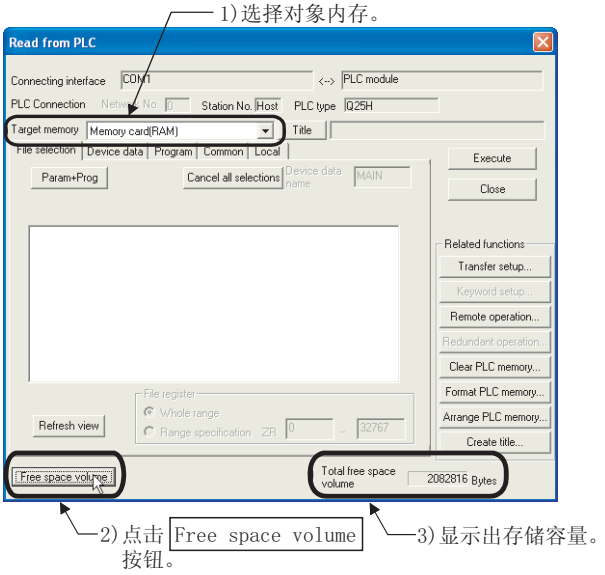


图 5.25 存储容量的确认步骤

(4) 对存储卡的写入

对存储卡写入前的操作与写入方法的种类进行说明。

(a) 对 SRAM 卡、ATA 卡的写入

对 SRAM 卡、ATA 卡的数据写入通过 GX Developer 的 [在线]→[可编程控制器写入] 进行。

对 SRAM 卡写入时，在可编程控制器画面的对象内存中选择 “存储卡 (RAM)”，进行可编程控制器写入。

对 ATA 卡写入时，在可编程控制器画面的对象内存中选择 “存储卡 (ROM)”，进行可编程控制器写入。

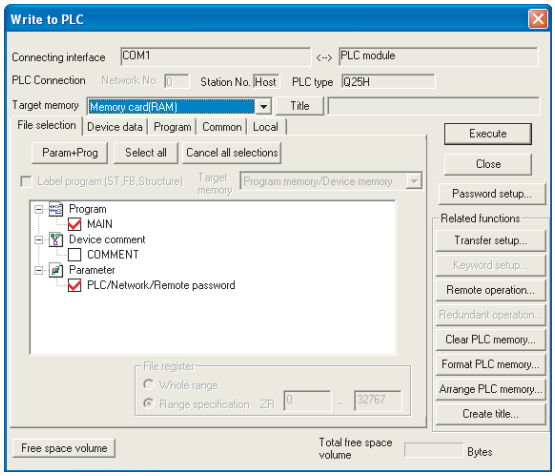


图 5.26 可编程控制器写入的画面

(b) 对 Flash 卡的写入

对 Flash 卡的写入，有如下 2 种方法：

- 通过 GX Developer 的 [在线]→[可编程控制器写入 (快闪卡)]→[程序内存的 ROM 化] 将程序内存的数据批量的写入到 Flash 卡中的方法。(☞ 5.2.7 项)
- 通过 GX Developer 的 [在线]→[可编程控制器写入 (快闪卡)]→[可编程控制器写入 (快闪卡)] 进行写入的方法。(☞ 5.2.7 项)

要 点

文件的大小有最小单位。(☞ 5.4.4 项)
占用的存储容量有比实际文件大的情况。
如果文件数目增多，请注意占用的存储容量与实际的文件大小的差额将会变大。

备 注

关于对 Flash 卡写入方法的选择标准、用途、步骤、注意事项的详细情况，请参照 5.2.7 项。

(5) 存储在存储卡中程序的使用方法

由于存储在存储卡中的程序不能执行运算，因此，将存储在存储卡上的程序从程序内存引导出 (读出) 并使用。(☞ 5.2.9 项)

5.2.7 通过 GX Developer 向标准 ROM、Flash 卡的写入

(1) 向标准 ROM、Flash 卡的写入的种类与用途

向标准 ROM、Flash 卡的写入数据的方法如图 5.27 所示。

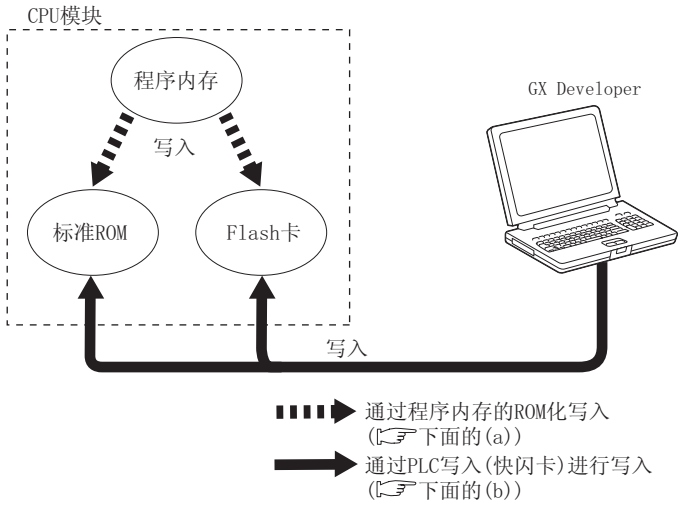


图 5.27 向标准 ROM、Flash 卡写入方法的种类

通过 [程序内存的 ROM 化] 的方法注 5.9

通用 UD 注 5.9

将程序内存的内容批量的复制到标准 ROM 或者 Flash 卡的方法。

程序内存的 ROM 化在下述情况下使用。

- 通过程序内存的程序进行调试，在原状态下将数据复制到标准 ROM 或者 Flash 卡并进行引导运行（☞ 5.2.9 项）时。
- 在没有后备电池的情况下将程序内存的数据保存到标准 ROM 或者 Flash 卡中时。

通过 GX Developer[可编程控制器写入（快闪卡）] 的方法注 5.9

通用 UD 注 5.9

通过 GX Developer 将指定的文件批量的复制到标准 ROM 或者 Flash 的方法。

可编程控制器写入（快闪卡）在下述的情况下使用。

- 将不能存储到程序内存中的参数、软元件初始值、软元件注释存储到标准 ROM 或者 Flash 卡时
- 将文件寄存器存储到 Flash 卡中并使用时

通用 UD 注 5.9

在通用型 QCPU 中，不能通过 “程序内存的 ROM 化”、“可编程控制器写入（闪存 ROM)” 对标准 ROM 进行写入。

(2) 对标准 ROM、Flash 卡的写入

对向标准 ROM 或者 Flash 卡写入之前的操作与写入方法进行说明。

(a) 写入前

将文件写入标准 ROM 或者 Flash 卡之前，请确认下述要点。

1) 标准 ROM 或者 Flash 卡内的文件的保存

将文件写入标准 ROM 或者 Flash 卡时，存储在标准 ROM 以及 Flash 卡的全部文件将自动被清除。

在对标准 ROM 写入之前，请使用 GX Developer 的 [在线]→[可编程控制器读出]，预先将存储的文件通过 GX Developer 进行保存等。

2) 写入文件的准备

向标准 ROM、Flash 卡写入文件时，由于存储在标准 ROM 以及 Flash 卡内的全部文件将被自动清除，请预先做准备存储好所有的文件。

3) 进行引导运行的情况

在进行引导运行之际将参数存储到标准 ROM 或者 Flash 卡时，请进行 5.2.9 项所示的引导文件的设定。

(b) 写入步骤

对向标准 ROM、Flash 卡写入文件的步骤进行说明。

1) GX Developer[程序内存的 ROM 化] 的步骤

- 选择 GX Developer 的 [在线]→[可编程控制器写入 (快闪卡)]→[程序内存的 ROM 化]。
- 显示程序内存的 ROM 化的画面。

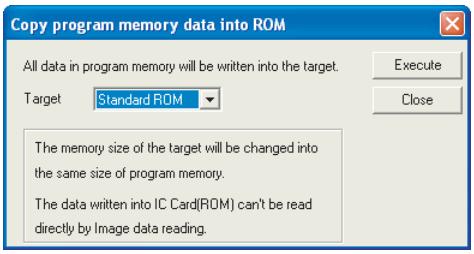


图 5.28 程序内存的 ROM 化的画面。

- 选择写入对象，将程序内存的文件复制到标准 ROM 或者 Flash 卡中。

要 点

1. 在通过程序内存的 ROM 化写入时，写入对象的使用存储容量成为在程序内存中使用的容量。
   若想充分使用写入对象的存储容量时，请使用可编程控制器写入 (快闪卡) 进行写入。
2. 在向 Flash 卡中写入程序内存中不能存储的数据 (文件寄存器) 时，请使用可编程控制器写入 (快闪卡) 进行写入。

2) GX Developer 的 [可编程控制器写入 (快闪卡)] 的步骤

- 选择 GX Developer 的 [在线]→[可编程控制器写入 (快闪卡)]→[可编程控制器写入 (快闪卡)]。
- 显示可编程控制器写入 (快闪卡) 的画面。

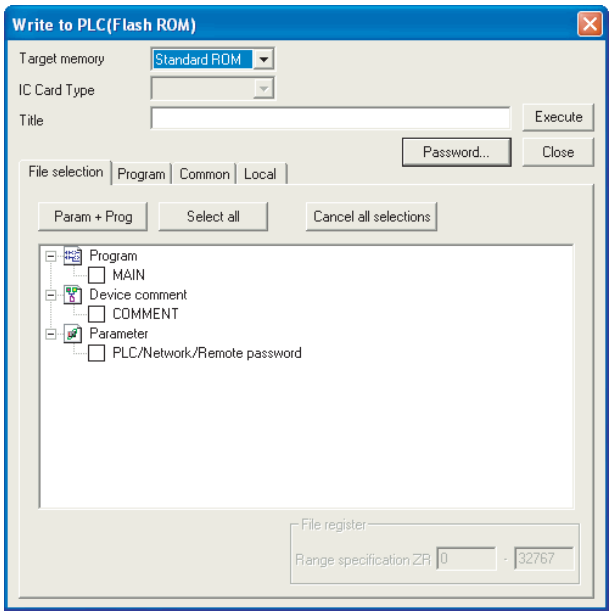


图 5.29 程序内存的 ROM 化画面

- 选择对象内存
- 选择写入文件，将文件写入到标准 ROM 或者 Flash 卡中。

(3) 对标准 ROM、Flash 卡内的文件进行的追加、变更

向标准 ROM、Flash 卡写入文件时，由于存储在标准 ROM、Flash 卡内的全部文件被自动清除的原因，因此，不能对存储的文件直接进行追加、变更。
通过下述方法进行追加、变更。

(a) 使用 GX Developer 的 [程序内存的 ROM 化] 写入的情况

- 通过 GX Developer 的 [在线]→[可编程控制器读出] 来读出程序内存的所有文件。
- 对读出的文件进行追加、变更。
- 将进行了追加、变更的文件写入到程序内存中。
- 通过 [在线]→[可编程控制器写入 (快闪卡)]→[程序内存的 ROM 化] 将文件写入到标准 ROM、Flash 卡中。

(b) 使用 GX Developer 的 [可编程控制器写入 (快闪卡)] 写入的情况

- 通过 GX Developer 的 [在线]→[可编程控制器读出] 来读出标准 ROM 或者 Flash 卡的所有文件。
- 对读出的文件进行追加、变更。
- 将进行了追加变更的文件通过 [在线]→[可编程控制器写入 (快闪卡)]→[可编程控制器写入
(快闪卡)] 写入到标准 ROM 或 Flash 卡中。

(4) 注意事项

(a) 关于 GX Developer 通讯检查时间的设定

向标准 ROM、Flash 卡中写入文件时，由于处理需要时间，请将 GX Developer 的通讯检查时间设定为 60 秒以上。

如果通讯检查时间过短，GX Developer 会出现超时。

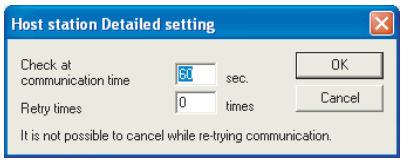


图 5.30 通讯检查时间设定

(b) 经由 CC-Link 从其它站的 GX Developer 进行写入的情况

向标准 ROM、Flash 卡写入文件时由于处理需要时间，请将 CC-Link 的 CPU 监视时间设定 (SW000A) 设定在 60 秒以上。

（由于默认值为 90 秒，可以使用默认值）

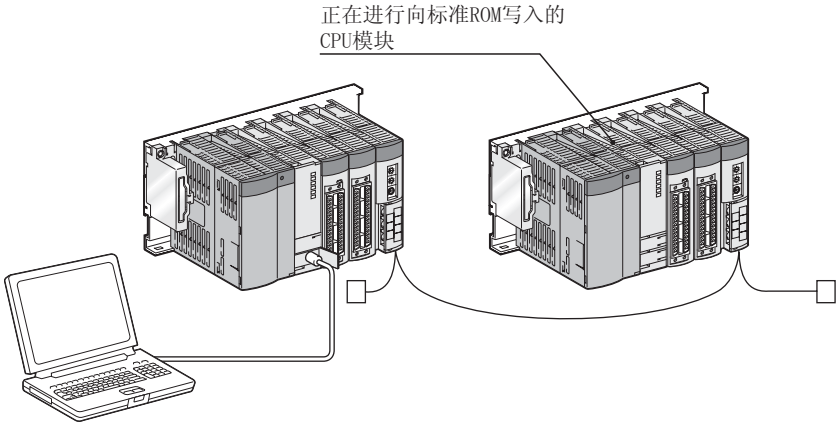


图 5.31 从其它站的 GX Developer 进行的写入（经由 CC-Link）

(c) 可编程控制器写入（快闪卡）所需时间
通过可编程控制器写入（快闪卡）进行的写入将写入标准 ROM 或者 Flash 卡的全部容量。
因此，即使向 Flash 卡写入小步数程序，由于进行 Flash 卡的全部容量的写入，因此，到结束需花费时间。
（对 Q2MEM-4MBF 使用 RS-232 接口，通讯速度为 115.2kbps 时，大约花费 14 分钟）
向 Flash 卡写入时，如果希望加快传送速度，请使用 USB。
另外，从其它站进行可编程控制器写入（快闪卡）也需花费通讯时间。

(d) 运行中的写入（只有可编程控制器写入（快闪卡）可以）
在运行中可以执行可编程控制器写入（快闪卡）。
但是，在下述情况下，请在 STOP 状态下执行可编程控制器写入（快闪卡）。

1) 通过顺控程序使用 Flash 卡的文件寄存器。

2) 通过可编程控制器参数将文件寄存器设定为 “不使用”，通过顺控程序使用文件寄存器。

在上述 1)，2) 的情况下，进行运行中的可编程控制器写入（快闪卡）会出现出错，高性能模式 QCPU、过程 CPU、冗余 CPU 有可能出现停止的情况。

(e) 可编程控制器写入（快闪卡）的写入 / 读出
在执行可编程控制器写入（快闪卡）过程中，不能从其它模块进行写入 / 读出。
因此，在其它模块中会出现超时。

(f) 在停止状态中的进行可编程控制器写入（快闪卡）的情况
在停止状态中进行可编程控制器写入（快闪卡）时，在写入过程中不要转换为运行状态。
在可编程控制器写入（快闪卡）过程中不能进行正常的运行。
可编程控制器写入（快闪卡）结束后请投入运行。

高性能
注 5.10

通用
注 5.11

5.2.8 存储卡向标准 ROM 中自动进行的全部数据写入

(1) 关于存储卡向标准 ROM 中自动进行的全部数据写入

存储卡向标准 ROM 中自动进行的全部数据写入功能（以下简称为向标准 ROM 的自动写入）是指将预先写在存储卡上的参数与程序自动的写入到标准 ROM 中的功能。

向标准 ROM 的自动写入如图 5.32 所示将参数、顺控程序等由存储卡被引导到程序内存中，再由程序内存写入标准 ROM 中。

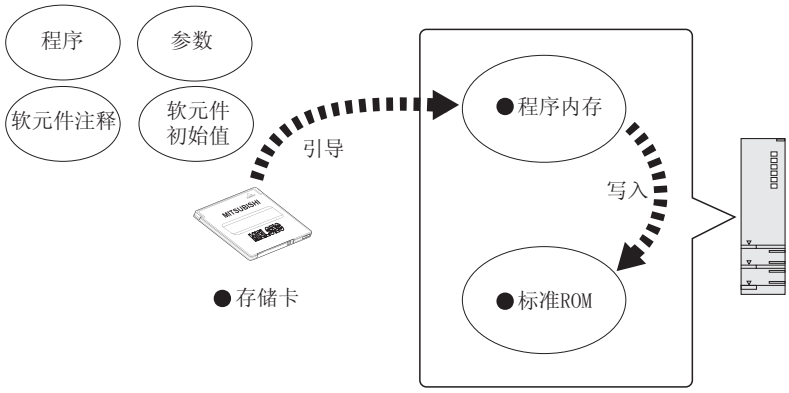


图 5.32 向标准 ROM 的自动写入

(2) 向标准 ROM 的自动写入的用途

如果使用向标准 ROM 的自动写入，可以将预先写在存储卡上的参数与程序不通过使用 GX Developer(PC) 直接写入到标准 ROM 中。

因此，在下述情况下使用时比较方便。

- 希望将相同的参数与程序写入多台的 CPU 模块中时
- 希望在较远的现场实现同样的环境时

高性能
注 5.10

高性能模式 QCPU 中使用向标准 ROM 的自动写入时，请确认 CPU 模块以及 GX Developer 的版本。(☞ 附录 4.2)

将设定了向标准 ROM 的自动写入的存储卡，安装在与标准 ROM 的自动写入不对应的高性能模式 QCPU 时，变为从标准 ROM 进行引导运行。

通用
注 5.11

在通用型 QCPU 中，不能进行存储卡→标准 ROM 的自动写入。

(3) 向标准 ROM 自动写入的执行步骤

向标准 ROM 的自动写入按照下述执行步骤进行。

(a) 通过 GX Developer 进行的操作（向标准 ROM 自动写入的设定）

1) 检查可编程控制器参数的引导文件设定中为 “清除程序内存”与 “存储卡向标准 ROM 全部数据自动写入”。
通过引导文件设定来设定引导的参数、程序等。
（传送源设定为 “标准 ROM”）

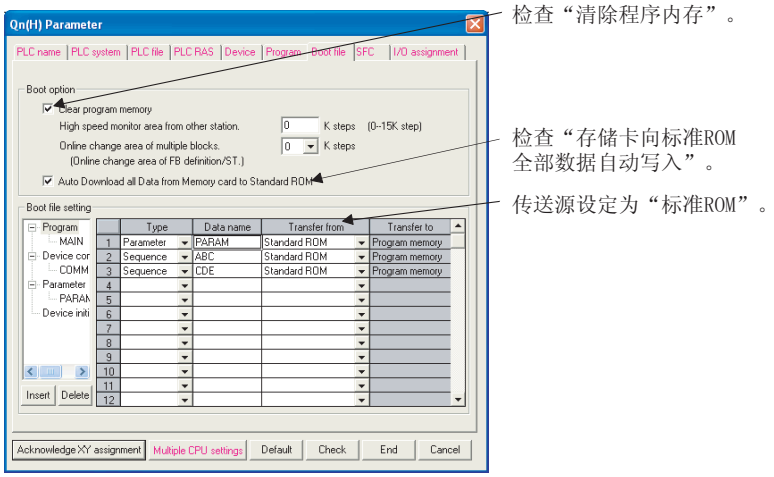


图 5.33 存储卡向标准 ROM 自动写入的设定

2) 设定的参数与引导程序存储在存储卡上。（☞ 5.2.6 项 (4)）

(b) 高性能模式 QCPU、过程 CPU、冗余 CPU 中的操作（向标准 ROM 自动写入）

1) 关闭可编程控制器的电源

2) 在高性能模式 QCPU、过程 CPU、冗余 CPU 上安装存储了参数与引导程序的存储卡。

3) 将高性能模式 QCPU、过程 CPU、冗余 CPU 的插杆开关的参数的有效驱动设定为安装了存储卡。

- 安装 SRAM 卡时 ..... SW2：ON, SW3：OFF
- 安装 Flash/ATA 卡时 ..... SW2：OFF, SW3：ON

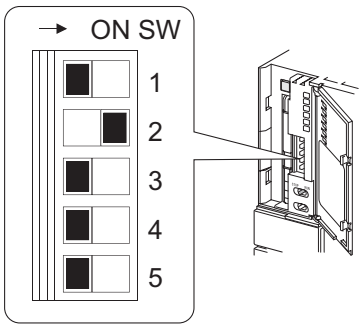


图 5.34 安装了 SRAM 卡的插杆开关

4) 接通可编程控制器的电源
将存储卡中指定的文件引导到程序内存中。
引导结束后，将程序内存的内容写入标准 ROM 中。

5) 向标准 ROM 的自动写入如果完成，BOOT LED 与 ERR. LED 将闪烁，高性能模式 QCPU、过程 CPU、冗余 CPU 变为停止出错状态。

6) 断开可编程控制器的电源

7) 拔出存储卡，将高性能模式 QCPU、过程 CPU、冗余 CPU 的插杆开关的参数的有效驱动设定为标准 ROM。

• 标准 ROM.....SW2：ON， SW3：ON

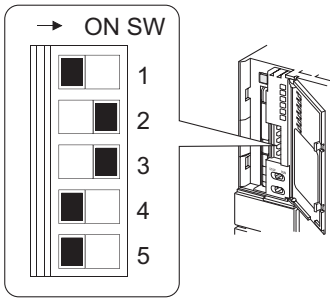


图 5.35 将设定插杆开关的参数的有效驱动设定为标准 ROM 的情况

8) 接通可编程控制器的电源
接通可编程控制器的电源后，进行从标准 ROM 向程序内存的引导，可以投入实际的运行。

(4) 注意事项
向标准 ROM 进行自动写入时需注意以下事项。

(a) 程序内存中存在有相同文件名的文件时
在程序内存中如果存在有与从存储卡引导的文件同名的文件时，用存储卡的数据覆盖。
另外，在程序内存中如果存在有与从存储卡引导的文件同名的文件时，需向程序内存中进行追加。
此时，如果超出程序内存的容量会出现 “FILE SET ERROR(出错代码：2401)”。

(b) 关于引导时程序内存的清除
从存储卡向程序内存中进行引导时，可以选择进行了程序内存的清除后引导，或是不进行程序内存的清除引导。
在进行向标准 ROM 的自动写入时，如果设定为进行程序内存的清除后引导，可以防止引导时超出程序内存的容量的现象。

(c) 关于本功能使用时的插杆开关的有效参数驱动的设定
“存储卡向标准 ROM 中全部数据自动进行写入”的设定，只有存储卡将插杆开关的参数有效驱动设定为存储卡时才有效。
因此，向标准 ROM 中的自动写入结束后，投入实际的运行时，没有必要取消引导文件设定的 “存储卡向标准 ROM 中全部数据自动写入”的检查。

通用 UD
注 5.12

5.2.9 标准 ROM/ 存储卡的程序的执行（引导运行）

在本项中，对存储在标准 ROM 注 5.12 以及存储卡内的程序的运算方法进行说明。

(1) 执行标准 ROM/ 存储卡的程序的方法。

CPU 模块进行存储在程序内存中的程序的运算。
存储在标准 ROM 以及存储卡内的程序不进行运算。
为进行存储在标准 ROM 以及存储卡内的程序的运算，需进行如下设定：在电源断开→接通或者复位解除时，将标准 ROM 以及存储卡内的程序引导（读出）到程序内存中。(☞ 本项 (4),(5))

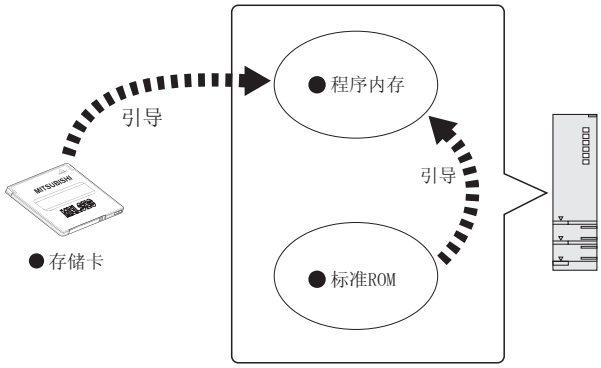


图 5.36 引导运行

引导文件设定中指定的文件名的程序，在电源处于断开→接通或者复位解除时，从标准 ROM 以及存储卡内被引导到程序内存中。

通用 UD
注 5.12

在通用型 QCPU 中，不能从标准 ROM 执行程序。

(2) 可引导的文件、传送源以及传送目标

根据 CPU 模块的不同，可引导的文件、传送源以及传送目标也有所不同。
可引导的组合如下表所示。

(a) 高性能模式 QCPU、过程 CPU、冗余 CPU 的情况

表 5.9 可引导的文件

文件名	传送源		传送目标
	存储卡	标准 ROM	
参数	○	○	程序内存
顺控程序	○	○	
软元件注释	○	○	
软元件初始值	○	○	

○ : 可引导； × : 不可引导

(b) 通用型 QCPU 的情况

表 5.10 可引导的文件

文件名	传送源	传送目标	
		程序内存	标准 ROM
参数	程序卡	○	○
顺控程序		○	×
软元件注释		○	○
软元件初始值		○	○
标签程序		○	○

○ : 可引导； × : 不可引导

(3) 进行引导运行的步骤

(a) 进行引导运行之前的准备工作如下
进行引导运行并创建程序。

(b) 通过 GX Developer 进行引导文件设定
为执行标准 ROM 以及存储卡的程序，在可编程控制器参数的引导文件设定中设定从程序内存中引导出（读出）的文件名。

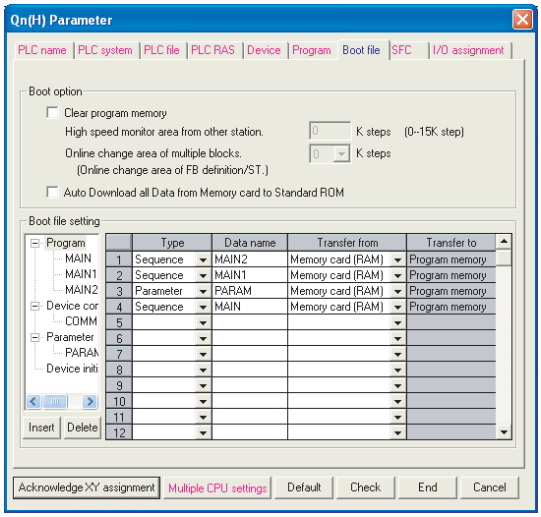


图 5.37 引导文件的设定

(c) 为引导运行进行的硬件设定

通用 UD 注 5.13

将插杆开关的参数有效驱动设定为存储参数的存储器。*注 5.13*
进行引导运行时，请将参数的存储对象设定为标准 ROM 或者存储卡。(☞ 本项 (6)(a))

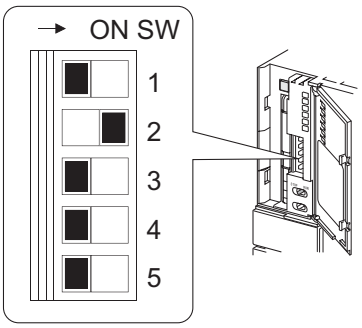


图 5.38 将插杆开关的参数有效驱动设定为 SRAM 卡的情况

通用 UD 注 5.13

在通用型 QCPU 中，不能从标准 ROM 执行程序。

(d) 存储卡的安装
通过引导运行将文件存储到存储卡内时，在高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 上安装存储卡。

(e) 通过 GX Developer 进行的参数、程序的写入

1) 高性能模式 QCPU、过程 CPU、冗余 CPU 的情况
向插杆开关的参数有效驱动中设定的存储器中写入参数。
另外，将 (b) 的引导文件设定中设定的文件写入到传送源存储器中。

2) 通用型 QCPU 的情况
将参数以及引导文件设定中设置的文件写入到存储卡中。

(f) 程序的执行
通过 RESET/L.CLR 开关（通用型 QCPU 时，为 RUN/STOP/RESET 开关）进行复位。
从指定的存储器的引导结束后，BOOT LED 将亮灯。

(g) 对引导是否正常结束的确认
引导是否正常的结束通过以下来进行确认。

- BOOT LED 亮灯
- 特殊继电器 (SM660) 处于 ON
- 通过 GX Developer 的 [在线]→[可编程控制器校验] 进行写入在发送源的存储中的数据与程序内存中数据的校验。

(4) 中止引导运行时的操作

中止引导运行，通过写在程序内存中的参数与程序文件执行运行时，请通过 GX Develper 进行以下的操作。

(a) 高性能模式 QCPU、过程 CPU、冗余 CPU 的情况

1) 将未设定引导文件的参数写入程序内存中。

2) 将 CPU 模块的插杆参数有效驱动设定为 “程序内存”。(SW2：OFF，SW3：OFF)

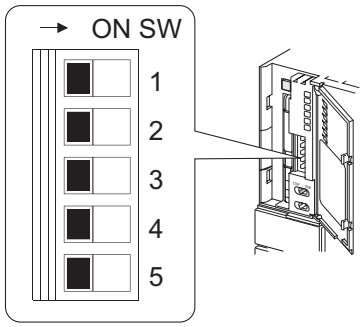


图 5.39 将插杆开关的参数有效驱动设定为程序内存的情况

3) 可编程控制器电源的再次投入或者对 CPU 模块进行复位。

(b) 通用型 QCPU 的情况

1) 卸下存储卡，将未进行引导文件设定的参数写入到程序内存中。

2) 重新接通可编程控制器的电源，或者对 CPU 模块进行复位。

冗余 注 5.14　通用 注 5.14

(5) 在运行过程中对程序文件的变更注 5.14

(a) 变更方法

高性能模式 QCPU、过程 CPU 在运行中，从标准 ROM 或者存储卡向程序内存进行文件的追加 / 变更 / 删除，通过顺控程序的下述指令可以进行。

- PLOADP 指令（从存储卡向程序内存中传送程序）
- PUNLOADP（从程序内存中删除程序）
- PSWAPP 指令（从程序内存中删除程序与从存储卡向程序内存传送程序）

PLOADP 指令、PUNLOADP 指令、PSWAPP 指令的详细情况，请参照下述手册。

☞ QCPU(Q 模式)/QnACPU 编程手册（公共指令篇）

(b) 关于变更程序文件时的程序设定

高性能模式 QCPU、过程 CPU 在运行中即使变更程序文件，可编程控制器参数的程序设定也不能被变更。

因此，在高性能模式 QCPU、过程 CPU 停止状态下，将可编程控制器参数的程序设定修正（程序名的追加、变更、删除）为高性能模式 QCPU、过程 CPU 在运行中变更的内容。

在不变更可编程控制器参数的程序设定的情况下，从停止状态切换到运行状态会出现错误。

(6) 执行标准 ROM/ 存储卡的程序时的注意事项

(a) 关于参数的存储对象

1) 高性能模式 QCPU、过程 CPU、冗余 CPU 的情况

- 进行引导运行时，请将进行了引导文件设定的参数（可编程控制器参数）存储到标准 ROM 或者存储卡中。
- 如果将参数存储在程序内存中，即使将插杆开关的参数有效驱动器设定为程序内存，CPU 模块也将忽略程序内存的可编程控制器参数的引导文件设定。CPU 模块在可编程控制器的电源 OFF → ON 或者 CPU 模块的复位解除时将不进行引导。

2) 通用型 QCPU 的情况

- 进行引导运行时，应将进行了引导文件设定的参数（可编程控制器参数）存储到存储卡中。
- 即使将进行了引导文件设定的参数存储到程序内存、标准 ROM 中，CPU 模块也将忽略。CPU 模块在可编程控制器的电源 OFF → ON 或者 CPU 模块的复位解除时将不进行引导。(☞5.2.11 项)
- 在下述条件 a) 及 b) 重叠时，CPU 模块将不以存储卡的参数动作，而以程序内存的参数动作。
  a) 程序内存中存在有参数
  b) 未将存储卡内的参数设置到引导文件设定中

(b) 关于引导运行过程中的运行中写入

1) 存储卡 (RAM)

如果对程序内存内的程序进行运行中写入 (☞ 6.12 节)，在引导源的存储卡 (RAM) 的程序中可以反映变更的内容。

冗余 注 5.14　通用 注 5.14

在冗余 CPU、通用型 QCPU 中，不能通过 PLOADP、PUNLOADP、PSWAPP 指令进行运行中的程序文件变更。

2) 标准 ROM/ 存储卡 (ROM)
即使进行程序内存的程序的运行中写入，其变更内容也不会反映到引导源的标准 ROM/ 存储卡 (ROM) 的程序中。
因此，请将与写入程序内存的程序相同的文件通过可编程控制器写入（快闪卡）写入到标准 ROM/ 存储卡 (ROM) 中。(☞ 5.2.7 项）

(c) 关于可以设定的最大引导文件数
请将可编程控制器参数的引导文件设定中可以设定的最大引导文件数设定为与程序内存中可以存储的文件数相同的数目。
但是，在进行下述设定中，引导文件分别减少一个文件。

- 设定标题时
- 引导存储在标准 ROM/ 存储卡上的可编程控制器参数（进行引导文件设定的参数）时

(d) 关于使用 ATA 卡时的引导运行
在下述状态中如果进行引导运行，在引导时每 1k 步 (4 字节）最大需要 200ms。

- 从 ATA 卡引导时
- 在安装 ATA 卡的状态下，从标准 ROM 引导时

(e) 电源处于 OFF→ON 或者复位解除时变更程序内存的内容的情况
向程序内存内写入顺控程序，并将可编程控制器的电源进行 OFF→ON/ 复位时，改写程序内存内容的情况被视为引导运行。
高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 前面的 BOOT LED 在亮灯时表示正在进行引导运行。
请参照本项 [(3) 中止引导运行时的操作]，中止引导运行。

(f) 关于从存储卡引导时的容量
存储在各个存储器中文件大小的单位根据存储卡与程序内存而不同。(☞ 5.4.4 项）
因此，从存储卡向程序内存传送的文件在传送之前与传送之后存储容量会发生改变，请加以注意。

5.2.10 关于写入文件的详细情况

写入 CPU 模块的各文件中，被附加了通过 GX Developer 创建时设定的文件名、文件大小、文件的写入日期等信息。
如果通过 GX Developer 的［在线］→［可编程控制器读出］进行文件的监视，各个文件显示情况如下。

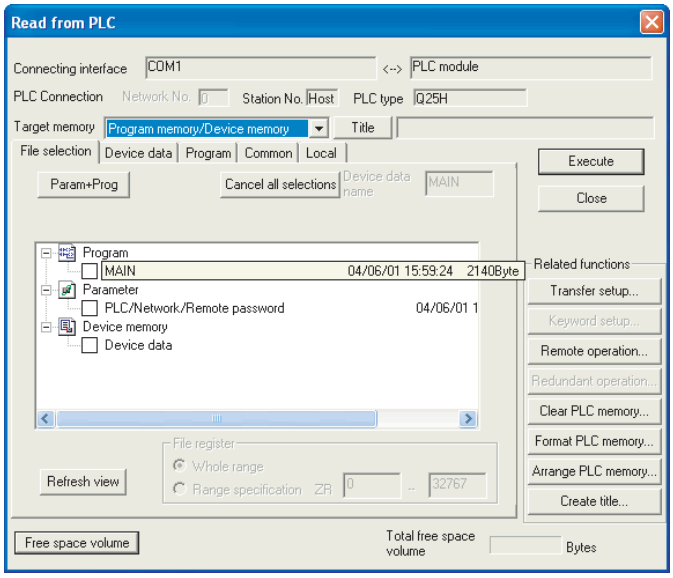


图 5.40 文件详细情况的显示

(a) 文件名

1) 文件名的构成与文件的指定
各个文件由文件名（最大半角 8 号字体 / 全角 4 号字体）与扩展子文件名（半角 3 号字体）构成。文件名全部用大写字母创建。
扩展名将根据文件创建时设定的类型被自动附加。

2) 文件名中不能使用的文字
下述显示的 Microsoft® Windows® 的保留字不能作为文件名使用。

- COM1-COM9
- LPT1-LPT9
- AUX
- CON
- PRN
- NUL
- CLOCK$

3) 顺控程序中的文件名
顺控程序中的文件名的指定方法根据 CPU 模块而有所不同。
- 高性能模式 QCPU、过程 CPU、冗余 CPU 的情况
  由于区分半角小写字母、半角大写字母，应以大写字母进行指定。
  * 区分 “ABC”与 “abc”进行处理。
- 通用型 QCPU 的情况
  由于不区分半角小写字母、半角大写字母，所以无论以哪一个进行指定都可以。
  * 指定 “ABC”或 “abc”时，均按 “ABC”进行处理。

备 注

由于区分全角小写字母、全角大写字母，因此应以大写字母进行指定。
* 区分 “ABC”与 “abc”进行处理。

(b) 日期、时间
是指通过 GX Developer 将文件写入到 CPU 模块中的日期及时间。
但是，设定的时间与日期为 GX Developer(PC) 的时间与日期。

(c) 大小
大小是指，从 GX Developer 向 CPU 模块进行写入时以字节为单位表示的文件容量。（最新的数据需点击 Refresh view ）
除文件寄存器以外的文件需附加用户创建的最低的文件容量 64 字节（程序时为 136 字节）。

5.2.11 有效参数的指定（参数有效驱动器设定）

参数有效驱动器设定是指，指定有效参数的存储驱动器（存储器）的功能。
根据 CPU 模块的不同，参数有效驱动器的指定方法也有所不同。

(1) 高性能模式 QCPU、过程 CPU、冗余 CPU 的情况

通过 CPU 模块本体的插杆开关进行指定。（ QCPU 用户手册（硬件设计 / 维护点检篇））

(a) 参数有效驱动器的指定方法

在 CPU 模块本体的插杆开关 SW2、SW3 的设定中指定参数有效驱动器。

表 5.11 通过 SW2、SW3 进行驱动器指定

SW2	SW3	参数的驱动器
OFF	ON	驱动器 0（程序内存）
ON	OFF	驱动器 1（存储卡 RAM）
OFF	ON	驱动器 2（存储卡 ROM）
ON	ON	驱动器 3（标准 ROM）

(b) 有效参数的确定时机

有效参数的确定时机如下所示。

- 电源 OFF → ON
- 复位解除

CPU 模块在上述确定时机使 SW2、SW3 中指定的驱动器的参数生效，以生效的参数设定执行动作。

(2) 通用型 QCPU 的情况

系统自动判断有效参数存储启动器并进行指定。
用户不能进行参数存储驱动器指定。

(a) 参数有效驱动器的指定方法

有效的参数取决于存储参数的驱动器的优先顺序。
存储参数的驱动器的优先顺序如表 5.12 所示。
应将希望使其有效的参数写入到驱动器中。

表 5.12 驱动器的优先顺序

优先顺序		参数的驱动器
高	1	驱动器 0(程序内存)
↑	2	驱动器 1(存储卡 RAM)
↓	3	驱动器 2(存储卡 ROM)
低	4	驱动器 3(标准 ROM)

CPU 模块指定参数存储驱动器的动作流程如图 5.41 所示。

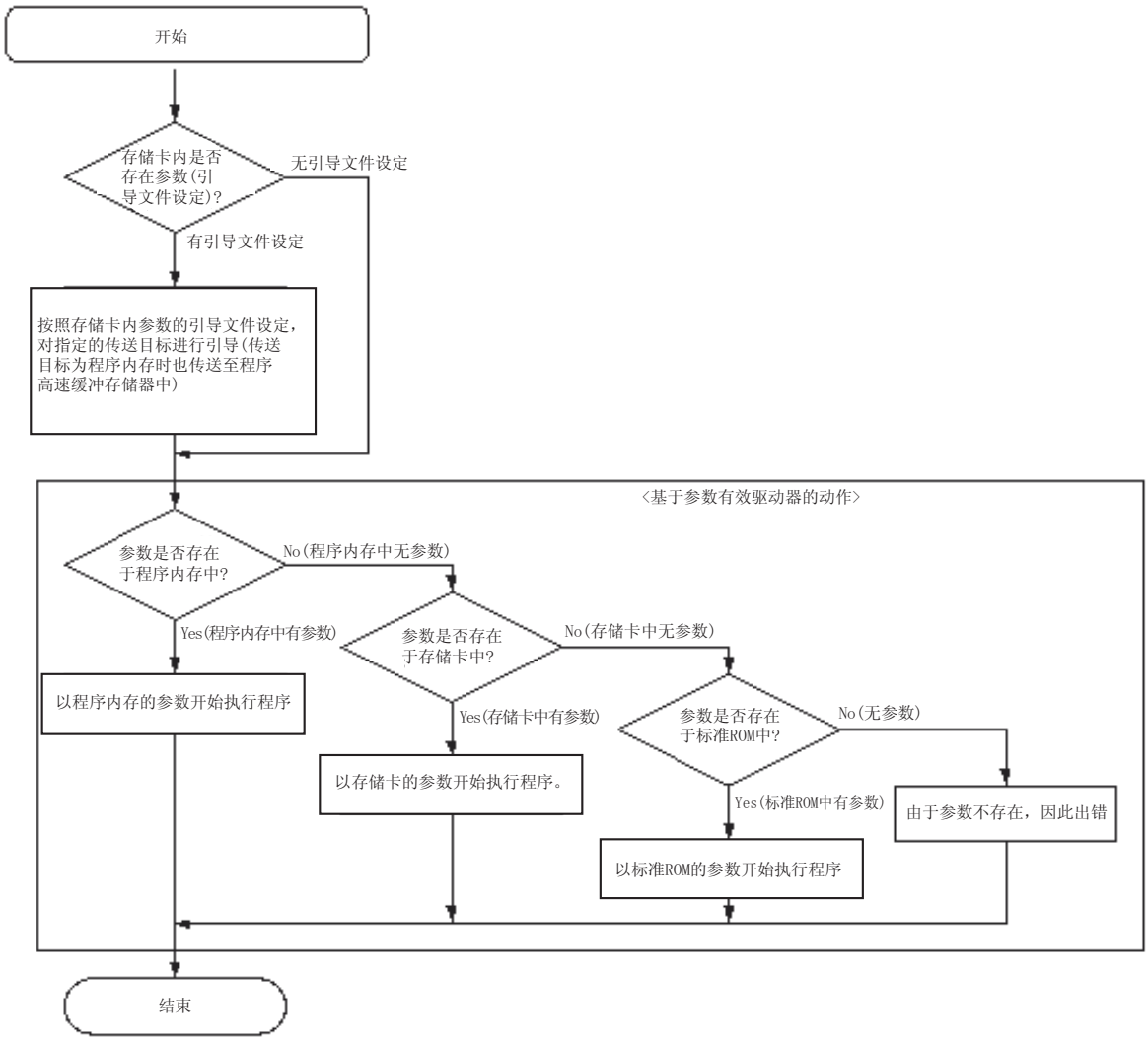


图 5.41 指定参数存储驱动器的动作流程

(b) 有效参数的确定时机

有效参数的确定时机如下所示：

- 电源 OFF → ON
- 复位解除

CPU 模块在上述确定时机自动检测参数，根据存在有参数的驱动器的参数设定执行动作。

通过 GX Developer 进行可编程控制器写入操作存储参数时，根据存储目标的驱动器，参数有效时机有所不同。

1) 将参数存储到与动作中的参数不同的驱动器中的情况
按照电源 OFF → ON 时 / 复位解除时驱动器的优先顺序，使参数有效。

2) 将参数存储到与动作中的参数相同的驱动器中的情况
仅参数内的软元件设定，在可编程控制器写入操作结束时有效。
若要使参数的全部设定有效，应进行电源 OFF → ON 或者复位解除操作。

5.3 程序文件的构成

程序文件由文件起始、执行文件、运行中写入用预留容量（单位：步）构成。

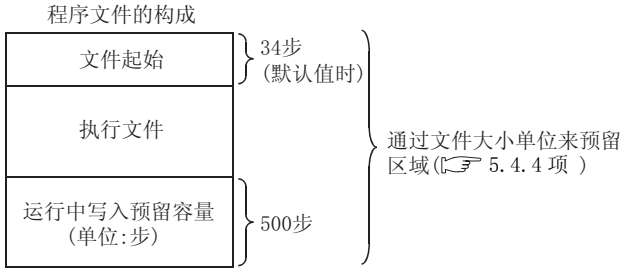


图 5.42 程序文件的构成

(1) 各构成的详细情况

存储在 CPU 模块的程序内存中的程序的容量为上述三种类型区域的合计容量。

(a) 文件起始

是指存储文件名、文件大小、文件创建日期等的区域。

文件起始通过变更可编程控制器参数的软元件设定变为 25-35 步 (100-140 字节)。

（默认值时为 34 步）

(b) 程序的执行

是指存储创建的程序的区域。

(c) 运行中写入预留容量（单位：步）

是指通过 GX Developer 进行增加步数运行中写入时使用的区域。

如果通过 GX Developer 进行增加步数运行中写入，将显示运行中写入预留容量（单位：步）的剩余步数。

1) 运行中写入预留容量（单位：步）的默认值

默认值被设定为 500 步 (2000 字节)。

2) 运行中写入预留容量（单位：步）的变更

运行中写入预留容量（单位：步）通过 GX Developer([在线]→[可编程控制器写
入] 的 << 程序 >> 制表) 可以变更。

另外，在运行中写入时，在运行中写入预留容量（单位：步）不足的情况下，可以再度设定运行中预留容量（单位：步）。(☞ 6.12.1 项)

(2) 在 GX Developer 显示的程序容量

GX Developer 在编程时，如图 5.43 所示通过步数显示出程序容量（文件起始容量与创建程序的步数的合计）。

创建程序时，可以对创建的程序的容量进行确认。

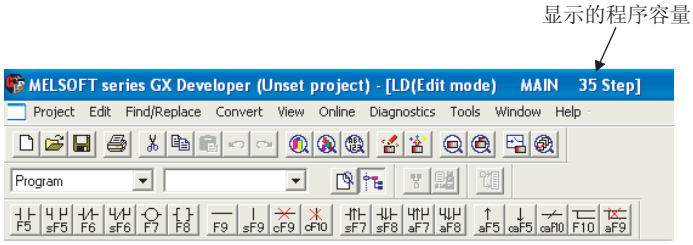


图 5.43 程序容量的显示

要 点

1. GX Developer 在编程时显示的程序容量是指文件起始加上执行文件的容量，不包括运行中写入预留容量（单位：步）的容量（500 步）。

   （示例）执行程序的部分为 491 步的程序，在 GX Developer 上的容量如下图所示。（文件起始为固定的 34 步）

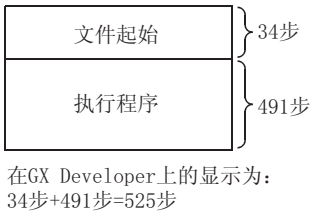


图 5.44 GX Developer 上的文件的状态

2. 在程序内存中，文件是以文件大小单位来存储的，因此，GX Developer 上编程时表示的程序容量与 CPU 模块上的程序文件的容量有不一致的情况。
   (☞ 5.4.4 项)

5.4 通过 GX Developer 进行文件操作以及使用时的注意事项

5.4.1 文件的操作

基本 注 5.15

存储在程序内存、标准 ROM、存储卡注 5.15 中的文件通过 GX Developer 可以进行文件的在线操作（如表 5.13 所示）。

但是，通过 GX Developer 的口令登陆，根据 CPU 模块的系统保护开关注 5.16 的状态，CPU 模块的 RUN/STOP 状态可以执行的文件操作会有不同。

基本 注 5.16　通用 UD 注 5.16

表 5.13 GX Developer 可执行的文件操作一览

文件的操作	操作内容	可否进行操作 *1（基本模式 QCPU）			可否进行操作 *1（高性能模式 QCPU）				
		A	B	C	A	B	C	D	
可编程控制器读出	从对象内存中读出文件。	○	△	○	○	△	○	○	
可编程控制器写入	向程序内存、SRAM 卡内写入文件。	△	△	×	△	△	○	×	
	向标准 ROM 中写入文件。	×	×	×	×	×	×	×	
可编程控制器校验	校验对象内存与 GX Developer 的文件。	△	△	○	△	△	○	○	
程序内存的 ROM 化	将存储在程序内存中的文件批量地写入 Flash 卡中。	○	○	×	○	○	○	×	
	将存储在程序内存中的文件批量地写入标准 ROM 中。	○	○	×	○	○	○	×	
可编程控制器写入（闪存 ROM）	将通过 GX Developer 指定的文件批量地写入 Flash 卡中。	×	×	×	△	△	○	×	
	将通过 GX Developer 指定的文件批量地写入标准 ROM 中。	×	×	×	△	△	○	×	
可编程控制器数据清除	清除存储器上存储的文件。	△	△	×	△	△	×	×	
可编程控制器存储器格式化	进行存储器的格式化。	○	○	×	○	○	×	×	
可编程控制器存储器整理	对存储器中配置不连续的文件进行再配置。	○	○	×	○	○	×	×	
梯形图模式中的运行中写入	将梯形图模式中变更的内容写入程序内存中。	△	△	○	△	△	○	×	

基本 注 5.15　在基本模式 QCPU 中，不能使用存储卡及系统保护开关。

基本 注 5.16　通用 UD 注 5.16　在基本模式 QCPU、通用型 QCPU 中，不能使用系统保护开关。

	可否进行操作 *1 (过程 CPU)				可否进行操作 *1 (冗余 CPU)				可否进行操作 *1 (通用型 QCPU)		
	A	B	C	D	A	B	C	D	A	B	C
	○	△	○	○	○	△	○	○	○	△	○
	△	△	○	×	△	△	○	×	△	△	○
	×	×	×	×	×	×	×	×	△	△	○
	△	△	○	○	△	△	○	○	△	△	○
	○	○	○	×	○	○	○	×	○	○	○
	○	○	○	×	○	○	○	×	×	×	×
	△	△	○	×	△	△	○	×	△	△	○
	△	△	○	×	△	△	○	×	×	×	×
	△	△	×	×	△	△	×	×	×	×	×
	○	○	×	×	○	○	×	×	○	○	×
	○	○	×	×	○	○	×	×	○	○	×
	△	△	○	×	△	△	○	×	△	△	○

○: 可以执行； △: 口令一致时可以执行； ×: 不可以执行

*1: 是否可以进行操作的符号的内容如下所示。

符号	内容
A	设定禁止文件写入的口令时
B	设定禁止文件读出 / 写入的口令时
C	CPU 模块处于运行中状态时
D	CPU 模块的系统保护开关处于 ON 时

5.4.2 文件使用时的注意事项

(1) 关于文件操作时电源处于断开的情况（包括复位）

在文件操作时断开电源或者对 CPU 模块进行复位时，各个存储器的文件如表 5.14 所示。

表 5.14 在文件操作中断开电源时文件的状态

CPU 模块	各个存储器的状态
基本模式 QCPU	各存储器的文件不定
高性能模式 QCPU 过程 CPU 冗余 CPU 通用型 QCPU	各存储器中的文件不被破坏。 （表示使用存储卡时，电源在断开时不将存储卡从 CPU 模块上拆除，在原状态下电源接通的情况。）

要 点

在移动文件的操作过程中发生可编程控制器的电源断开的情况下，操作过程中的数据将保存在 CPU 模块的内部存储器中。
保存的数据在电源由断开→接通时将恢复。
为了保存内部存储器的数据，因此，有必要通过电池进行备份。

(2) 关于从多个 GX Developer 向同一文件同时写入的情况

在 CPU 模块中，不能对写入中的文件通过其它的 GX Developer 进行访问。
另外，不能对在存取过程中的文件通过其它 GX Developer 进行写入。
在从多个 GX Developer 向同一文件写入的情况下，请在结束一个 GX Developer 的处理后，再进行下一个 GX Developer 的处理。

(3) 关于从多个 GX Developer 向不同文件同时存取的情况

在 CPU 模块中，从其它 GX Developer 对同一 CPU 模块的不同文件最大可以同时存取 10 个。

5.4.3 文件的存储容量

在 CPU 模块中使用的文件根据种类其文件的大小会有不同。
在本项中，说明各个 CPU 模块的文件的存储容量。
另外，将文件写入存储区域时，根据写入的 CPU 模块与存储区域，存储的容量的单位也会有不同。(☞ 5.4.4 项)

(1) 使用基本模式 QCPU 时

使用程序内存、标准 RAM、标准 ROM 时，各个文件大小的如表 5.15 计算得出。

表 5.15 文件的存储容量的计算（基本模式 QCPU）

<table>
<tr><th>功能</th><th>文件大致容量（单位：字节）</th></tr>
<tr><td>驱动标题</td><td>64</td></tr>
<tr><td>参数</td><td>默认值 :522(通过参数的设定增加)
参考
引导设定 → 94
MELSECNET/H 设定→最大增加 6180
Ethernet 设定→最大增加 922
CC-Link 设定→最大增加如下表所示的容量（下表的值表示每个模块的增加量。）

<table><tr><td rowspan="2">CC-Link 设定</td><td colspan="3">模式设置</td></tr><tr><td>版本 1 模式</td><td>版本 2 模式</td><td>版本 2 追加模式</td></tr><tr><td>第 1 个</td><td>550 字节</td><td>572 字节</td><td>624 字节</td></tr><tr><td>第 2 个 ～ 第 4 个</td><td>536 字节</td><td>558 字节</td><td>610 字节</td></tr><tr><td>第 5 个</td><td>550 字节</td><td>566 字节</td><td>618 字节</td></tr><tr><td>第 6 个 ～ 第 8 个</td><td>536 字节</td><td>558 字节</td><td>610 字节</td></tr></table>
远程口令设定→ 64+20+(对象模块数× 10)，最大增加 164</td></tr>
<tr><td>顺控程序</td><td>136*1+(4 × ((步数)+(运行中写入预留容量)))</td></tr>
<tr><td>软元件注释</td><td>74+(各软元件的注释数据大小的合计)
•1 个软元件的注释数据的大小 = 10 + 10250 × a + 40 × b
•a:((软元件点数)/256) 的商
•b:((软元件点数)/256) 的余数</td></tr>
<tr><td>文件寄存器</td><td>2 ×(文件寄存器点数)</td></tr>
<tr><td>软元件初始值</td><td>66+44 × n+2 ×(软元件初始值中设定软元件点数的合计)
•n: 软元件初始值的设定数</td></tr>
<tr><td>智能参数</td><td>68+(24 ×设定台数)+ 各个实用程序的参数大小</td></tr>
<tr><td>用户设定区域</td><td>格式化时的设定值 (0-3k)</td></tr>
<tr><td>设定多个块的运行中写入</td><td>格式化时的设定值 (0/1.25k/2.5k)</td></tr>
</table>

*1：默认值为 136。(通过参数设定增减)

(2) 使用高性能模式 QCPU、过程 CPU、冗余 CPU 时

使用程序内存、标准 RAM、标准 ROM、存储卡时，各个文件的大小如表 5.16 计算得出。

表 5.16 文件的存储容量的计算（高性能模式 QCPU、过程 CPU、冗余 CPU）

<table>
<tr><th>功能</th><th>文件大致容量（单位：字节）</th></tr>
<tr><td>驱动标题</td><td>64</td></tr>
<tr><td>参数</td><td>默认值 :564(通过参数的设定增加)
参考
引导设定 → 70+(18 × (文件数))
MELSECNET/G*3*4 设定→最大增加 7214/ 模块
MELSECNET/H 设定 → 最大增加 6180/ 模块
Ethernet 设定 → 最大增加 922/ 模块
CC-Link 设定→最大增加如下表所示的容量（下表的值表示每个模块的增加量。）

<table><tr><th rowspan="2">CC-Link 设定</th><th colspan="3">模式设置</th></tr><tr><th>版本 1 模式</th><th>版本 2 模式</th><th>版本 2 追加模式</th></tr><tr><td>第 1 个</td><td>550 字节</td><td>572 字节</td><td>624 字节</td></tr><tr><td>第 2 个 ～ 第 4 个</td><td>536 字节</td><td>558 字节</td><td>610 字节</td></tr><tr><td>第 5 个</td><td>550 字节</td><td>566 字节</td><td>618 字节</td></tr><tr><td>第 6 个 ～ 第 8 个</td><td>536 字节</td><td>558 字节</td><td>610 字节</td></tr></table>
远程口令设定 → 64+20+(对象模块数× 10)，最大增加 164</td></tr>
<tr><td>顺控程序</td><td>136*1+(4 × ((步数)+(运行中写入预留容量)))</td></tr>
<tr><td>软元件注释</td><td>74+(各软元件的注释数据大小的合计)
• 1 个软元件的注释数据的大小 =10+10250 × a+40 × b
• a:((软元件点数)/256) 的商
• b:((软元件点数)/256) 的余数</td></tr>
<tr><td>软元件初始值</td><td>66+44 × n+2 × (软元件初始值中设定软元件点数的合计)
• n: 软元件初始值的设定数</td></tr>
<tr><td>用户设定区域</td><td>格式化时的设定值 (0-15k)</td></tr>
<tr><td>设定多个块的运行中写入</td><td>格式化时的设定值 (0/2k/4k)</td></tr>
<tr><td>文件寄存器</td><td>2 × (文件寄存器点数)</td></tr>
<tr><td>采样追踪文件</td><td>362+(字软元件点数＋位软元件点数) × 12 +(N1+N2+N3+ 字软元件点数× 2+(位软元件点数 /16) × 2) ×追踪次数(总次数) *2
• 对于 N1 ～ N3，根据在追踪条件设置画面的追踪附加信息中设置的项目，需要加上下述的值
(☞ 6.14 节 (4)(b))
N1: 设置时间时加上 “4”
N2: 设置步号时加上 “10”
N3: 设置各程序名时加上 “8”</td></tr>
<tr><td>故障历史记录数据</td><td>72+54 × (存储故障数)</td></tr>
</table>

表 5.16 文件的存储容量的计算（ 高性能模式 QCPU、过程 CPU、冗余 CPU)（续）

功能	文件大致容量（单位：字节）
局部软元件	72+6（设定软元件种类）+(2 ×（(M、V 的合计点数）/16+(D 个数） +18(T、ST、C 的合计点数）/16)）×（程序个数） •M、V、D、T、ST、C 表示设定的下述软元件 M ：内部继电器 V ：变址继电器 D ：数据寄存器 T ：计时器 ST: 累计计时器 C ：计数器

*1: 默认值为 136。（通过参数设定增减）
*2: 位软元件点数 /16 在小数点以下舍去。
*3: 在高性能模式 QCPU 中使用 MELSECNET/G 时，请确认 CPU 模块以及 GX Developer 的版本。
（ 附录 4.2)
*4: 在过程 CPU、冗余 CPU 中，不能使用 MELSECNET/G。

备 注

关于存储容量的计算请参照 5.4.4 项。

(3) 使用通用型 QCPU 时

使用程序内存、标准 RAM、标准 ROM、存储卡时，应根据表 5.18 大致计算出各文件的大小。

表 5.17 文件的存储容量的计算（通用型 QCPU）

功能	大致文件容量（单位：字节）
驱动器标题	72
参数	默认值 :64(通过参数的设定增加)*1 参考 引导设定 → 84 + (18 × (文件数))*2 MELSECNET/G 设定 → 72+(各模块的参数大小的合计)+(路由设定的大小) + (数据链接间传送设定的大小) • 各模块的参数大小： 最大 10368(仅设定 LB/LW(1) 时 1826 + 16 × (刷新的传送点数)) • 路由设定的大小：6 + 8 × (路由设定数) • 数据链接间传送设定的大小：6+12 × (传送设定数) + 86 × (模块数) MELSECNET/H 设定 → 最大增加 6180/ 模块 Ethernet 设定 → 最大增加 922/ 模块 CC-Link 设定 → 最大增加如下表所示的容量（下表的值表示每个模块的增加量。） 远程口令设定 → 92 + (对象模块数 × 10)，最大增加 172
顺控程序	148*3+ ((4 × (步数) + (运行中写入预留容量)))
软元件注释	74 + 8 + (各软元件的注释数据大小的合计) •1 个软元件的注释数据的大小 = 10 + 10250 × a + 40 × b •a : ((软元件点数)/256) 的商 •b : ((软元件点数)/256) 的余数
软元件初始值	66 + 44 × n + 2 × (软元件初始值中设定软元件点数的合计) + 8 •n : 软元件初始值的设定数
用户设定区域	格式化时的设定值 (0-15k)
文件寄存器	2 × (文件寄存器点数)
采样追踪文件	362 + (字软元件点数 + 位软元件点数) × 12 + (N1 + N2 + N3 + 字软元件点数 × 2 + (位软元件点数 /16) × 2) × 追踪次数(总次数) *4 •对于 N1 ～ N3，根据在追踪条件设置画面的追踪附加信息中设置的项目，需要加上下述的值 (☞ 6.14 节 (4)(b)) N1: 设置时间时加上 “4” N2: 设置步号时加上 “10” N3: 设置各程序名时加上 “8”
软元件数据备份用文件	格式化时的设定值 (2 ～ 1024k)

CC-Link 设定 | 模式设置

CC-Link 设定	版本 1 模式	版本 2 模式	版本 3 模式
第 1 个	550 字节	572 字节	624 字节
第 2 个 ～ 第 4 个	536 字节	558 字节	610 字节
第 5 个	550 字节	566 字节	618 字节
第 6 个 ～ 第 8 个	536 字节	558 字节	610 字节

表 5.17 文件的存储容量的计算（通用型 QCPU)（续）

功能	大致文件容量（单位：字节）
局部软元件	70 + 6 ×（设定软元件种类）+（2 ×（(M、V 的合计点数)/16 +（D 个数） +18 ×（T、ST、C 的合计点数)/16)）×（程序个数）*2 •M、V、D、T、ST、C 表示设定的下述软元件。 M ：内部继电器 V ：变址继电器 D ：数据寄存器 T ：计时器 ST ：累计计时器 C ：计数器

*1: 通过系统将其与网络参数设定的合计字节数调整为 4 的倍数。
*2: 通过系统将字节数调整为 4 的倍数。
*3: 默认值为 148。（通过参数设定增减。）
*4: 位软元件点数 /16 在小数点以下舍去。

备 注

关于存储容量的计算请参照 5.4.4 项。

5.4.4 文件的大小单位

(1) 关于文件的大小单位

将文件写入存储区域时，根据写入的 CPU 模块与存储区域其存储容量的单位会有不同。

这个单位称作文件大小单位。

(a) 各存储区域中文件大小单位

说明写入的 CPU 模块与各存储区域中文件大小单位。

表 5.18 每个 CPU 模块的文件大小单位（各存储区域）

CPU 模块的型号	存储区域	
	程序内存、标准 ROM、Flash 卡 *1 的文件大小单位	标准 RAM
Q00JCPU	1 步 /4 字节	-
Q00CPU		4 字节
Q01CPU		
Q02CPU	128 步 /512 字节 *2	512 字节
Q02HCPU		
Q06HCPU		
Q12HCPU	256 步 /1024 字节 *2	1024 字节 *4
Q25HCPU	512 步 /2048 字节 *2	
Q12PHCPU	256 步 /1024 字节 *3	1024 字节
Q25PHCPU	512 步 /2048 字节 *3	
Q12PRHCPU	256 步 /1024 字节	
Q25PRHCPU	512 步 /2048 字节	
Q02UCPU	程序内存： 1 步 /4 字节 标准 ROM、Flash 卡： 128 步 /512 字节	512 字节
Q03UDCPU		
Q04UDHCPU		
Q06UDHCPU		

基本 注 5.17

*1 ： Flash 卡的文件大小单位适用于通过 GX Developer 的［程序内存的 ROM 化］（☞ 5.2.7 项 (1)(a)）向 Flash 卡写入时。注 5.17

*2 ： 序列号的高 5 位为 “04121” 以前的高性能模式 QCPU 为 1024 步 /4096 字节。

*3 ： 序列号的高 5 位为 “07031” 以前的过程 CPU 为 1024 步 /4096 字节。

*4 ： 序列号的高 5 位为 “02091” 以前的 Q12HCPU、Q25HCPU 为 512 字节。

基本 注 5.17

在基本模式 QCPU 中不能使用 Flash 卡。

基本 注 5.18

(b) 各存储卡的文件大小单位注 5.18

表 5.19 文件大小单位（各存储卡）

类别	存储卡型号	文件大小单位（簇大小）
SRAM 卡	Q2MEM-1MBS	512 字节
	Q2MEM-2MBS	1024 字节
	Q3MEM-4MBS注 5.19	1024 字节
	Q3MEM-8MBS注 5.19	4096 字节
Flash 卡 *1	Q2MEM-2MBF	1024 字节
	Q2MEM-4MBF	1024 字节
ATA 卡	Q2MEM-8MBA	4096 字节
	Q2MEM-16MBA	4096 字节
	Q2MEM-32MBA	2048 字节

基本 注 5.19 高性能 注 5.19 过程 注 5.19

冗余 注 5.19

*1 ：Flash 卡的文件大小单位适用于下述情况。
- 通过 GX Developer 的 [可编程控制器写入（快闪卡）]（☞ 5.2.7 项 (1)(b)）将文件向 Flash 卡写入时。
- 利用 GX Developer 将文件不经由 CPU 模块直接写入 Flash 卡时。

(2) 存储容量的计算示例

说明将参数与顺控程序写入程序内存时存储容量的计算示例。

(a) 条件

1) 写入对象 CPU 模块：Q25HCPU

2) 写入的文件

表 5.20 各文件的容量

文件名	存储容量 *1
PARAM.QPA（参数文件）	564 字节
MAIN.QPG（顺控程序）	525 步 /2100 字节 *2

*1 ：关于文件的存储容量请参照 5.4.3 项。

*2 ：表示通过 GX Developer 表示的程序容量（文件起始 + 执行程序）（☞ 5.3 节）

3) 运行中写入预留容量（单位：步）：500 步 /2000 字节

基本 注 5.18

在基本模式 QCPU 中不能使用存储卡。

基本 注 5.19 高性能 注 5.19 过程 注 5.19

冗余 注 5.19

在基本模式 QCPU、高性能模式 QCPU、过程 CPU、冗余 CPU 中，不能使用 Q3MEM-4MBS、Q3MEM-8MBS。

(b) 存储容量的计算

存储容量的计算是以写入对象 CPU 模块的文件大小单位为基准进行计算的。
(☞ 本项 (1))
本例中的 Q25HCPU 根据本项 (1) 文件大小单位为 512 步 /2048 字节。

1) 参数文件的容量计算

参数文件的容量虽然为 564 字节，但要按照程序内存上文件大小单位进行存储的原因，因次，占用 512 步 /2048 字节的容量。

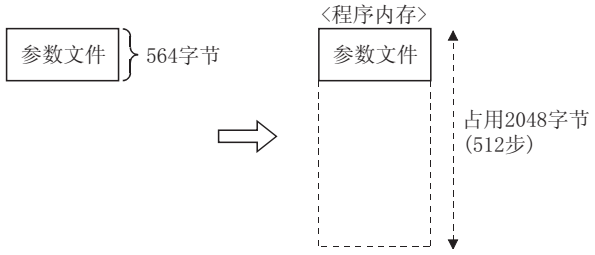


图 5.45 文件大小单位的占用 (参数文件)

2) 程序容量的计算

程序的容量为顺控程序的容量 + 运行中写入预留容量 (单位：步)。
例如：525 步 +500 步 =1025 步，由于使用程序内存上文件大小单位 (本例中的 Q25HCPU 中为 512 步单位) 进行存储的原因，因此，占用 1536 步 /6144 字节的容量。

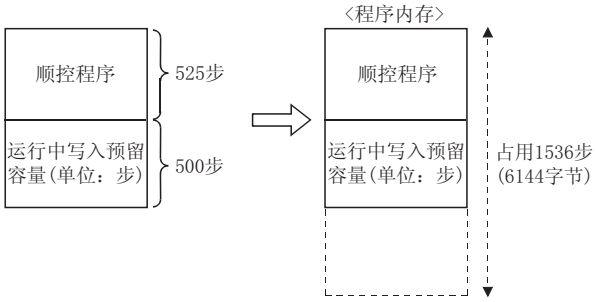


图 5.46 文件大小单位的占用 (程序文件)

3) 计算结果

表 5.21 存储容量的计算结果

文件名	文件容量		存储容量
PARAM.QPA	564 字节		512 步 (2048 字节)
MAIN.QPG	顺控程序容量	525 步	1536 步 (6144 字节)
	运行中写入预留容量 (单位：步)	500 步	
	合计	1025 步	
存储容量合计			2048 步 (8192 字节)

要 点

以下的 CPU 模块的文件大小单位将发生变化。

- 序列号的高 5 位为 “04112” 以后的高性能模式 QCPU
- 序列号的高 5 位为 “07032” 以后的过程 CPU

因此应注意以下几点：

1. 在上述序列号以后版本的 CPU 模块中运行的文件由于文件容量的原因，有可能不能被存储到上述序列号以前版本的 CPU 模块中。
2. 关于可编程控制器写入及与 GX Developer 的组合
   通过 GX Developer 将文件从高性能模式 QCPU/ 过程 CPU 中读出后，可编程控制器写入到其它的高性能模式 QCPU/ 过程 CPU 中时，高性能模式 QCPU/ 过程 CPU 与 GX Developer 的版本的组合如下所示：

(高性能模式 QCPU 时)

范例 ◎: 可以进行可编程控制器写入；○: 对可编程控制器写入有限制事项

写入对象	写入源			
	GX Developer 版本 8		GX Developer 版本 7	
	序列号为 “04122” 或以后的 CPU 的运行文件	序列号为 “04121” 或以前的 CPU 的运行文件	序列号为 “04122” 或以后的 CPU 的运行文件	序列号为 “04121” 或以前的 CPU 的运行文件
序列号 “04122” 或以后	◎	◎	◎*2	○*2
序列号 “04121” 或以前	○*1	◎	○*1 *2	○*2

*1 ： 由于文件大小单位不同的原因，根据文件容量有时会发生不能存储到高性能模式 QCPU 中的情况。
*2 ： 如果不减少运行中写入预留容量 (单位：步)，根据文件容量有时会发生不能存储到高性能模式 QCPU 中的情况。

(过程 CPU 时)

范例 ◎: 可以进行可编程控制器写入；○: 对可编程控制器写入有限制事项

写入对象	写入源			
	GX Developer 版本 8		GX Developer 版本 7	
	序列号为 “07032” 或以后的 CPU 的运行文件	序列号为 “07031” 或以前的 CPU 的运行文件	序列号为 “07032” 或以后的 CPU 的运行文件	序列号为 “07031” 或以前的 CPU 的运行文件
序列号 “07032” 或以后	◎	◎	◎*2	○*2
序列号 “07031” 或以前	○*1	◎	○*1 *2	○*2

*1 ： 由于文件大小单位不同的原因，根据文件容量有时会发生不能存储到高性能模式 QCPU 中的情况。
*2 ： 如果不减少运行中写入预留容量 (单位：步)，根据文件容量有时会发生不能存储到高性能模式 QCPU 中的情况。

备忘录

第 6 章　功能

对 CPU 模块的功能进行说明

6.1　功能一览表

CPU 模块的功能一览表如表 6.2 所示。
“CPU 模块”栏显示的号码与 CPU 模块的对应情况如下所示。

表 6.1　“CPU 模块”栏显示的号码与 CPU 模块的对应情况

编号	CPU 模块
1)	基本模式 QCPU
2)	高性能模式 QCPU
3)	过程 CPU
4)	冗余 CPU
5)	通用型 QCPU

表 6.2 CPU 模块的功能一览表

项目	内容	CPU 模块					参考
		1)	2)	3)	4)	5)	
恒定扫描	使程序在一定时间间隔内执行的功能。	○	○	○	○	○	6.2 节
锁存功能	电源进行断开，复位操作时保存软元件数据的功能。	○	○	○	○	○	6.3 节
停止－运行时的输出状态的选择功能	CPU 模块由停止－运行时选择输出 (Y) 状态（停止前的输出的再输出 / 运算执行后的输出）的功能。	○	○	○	○	○	6.4 节
计时功能	使 CPU 模块内部计时的功能。	○	○	○	○	○	6.5 节
远程运行 / 停止	使 CPU 模块运算停止、执行的功能。	○	○	○	○	○	6.6.1 项
远程中止	保持 CPU 模块的输出 (Y) 状态，停止 CPU 模块的运算的功能。	○	○	○	○	○	6.6.2 项
远程复位	CPU 模块处于停止状态时，对 CPU 模块进行复位的功能。	○	○	○	○	○	6.6.3 项
远程锁存清除	CPU 模块处于停止状态时，清除 CPU 模块的锁存数据的功能。	○	○	○	○	○	6.6.4 项
输入响应时间的选择	选择 Q 系列对应的输入模块、I/O 混和模块、高速输入模块、中断模块的响应时间的功能。	○	○	○	○	○	6.7 节

○：可以使用　　×：不可以使用

（下页续）

表 6.2 CPU 模块的功能一览表（续）

项目	内容	CPU 模块					参考
		1)	2)	3)	4)	5)	
出错时的输出模式	设定在 CPU 模块出现停止出错时，清除还是保存向 Q 系列对应的输出模块、I/O 混和模块、智能功能模块的输出的功能。	○	○	○	○	○	6.8 节
硬件出错时的 CPU 动作模式设定	设定智能功能模块的硬件发生出错时， CPU 模块运算是否停止的功能。	○	○	○	○	○	6.9 节
智能功能模块的开关设定	对智能功能模块进行各种设定的功能（关于设定的内容，请参照各个智能功能模块手册）	○	○	○	○	○	6.10 节
监视功能	通过 GX Developer 读出 CPU 模块的程序、软元件的状态的功能。	○	○	○	○	○	6.11 节
监视条件的设定	通过 CPU 模块的微小计时进行监视的功能	×	○	○	○	×	6.11.1 项
局部软元件的监视、测试	通过 GX Developer 进行指定程序的局部软元件的监视、测试的功能。	×	○	○	○	○	6.11.2 项
外部 I/O 的强制 ON/OFF	通过 GX Developer 对 CPU 模块的外部 I/O 强制 ON/OFF 的功能。	×	○	○	○	×	6.11.3 项
运行中写入	在 CPU 模块的运行中写入程序的功能。	○	○	○	○	○	6.12 节
程序一览监视	显示执行中的程序的处理时间的功能。	○	○	○	○	○	6.13.1 项
中断程序一览监视	显示中断程序的执行次数的功能。	○	○	○	○	○	6.13.2 项
扫描时间的测量	测量程序的任意步间的执行时间的功能。	×	○	○	○	×	6.13.3 项
采样追踪功能	在指定时间内连续收集指定软元件数据的功能。	×	○	○	○	○	6.14 节
多人进行的调试功能	从多个 GX Developer 同时进行调试的功能。	○	○	○	○	○	6.15 节
看门狗定时器	对 CPU 模块的硬件、程序异常而出现的运算停滞进行监视的功能。	○	○	○	○	○	6.16 节
自检测功能	CPU 模块自身进行有无异常检测的功能。	○	○	○	○	○	6.17 节
故障历史纪录	将自检测的结果以故障历史记录的形式存储在存储器内的功能。	○	○	○	○	○	6.18 节
系统保护	防止来自 GX Developer、串行口通讯模块、Etherent 模块等的程序变更的功能。	○	○	○	○	*2 △	6.19 节

○：可以使用　△：部分可以使用　×：不可以使用

*2:　在通用型 QCPU 中，只能使用远程口令。

（下页续）

表 6.2 CPU 模块的功能一览表（续）

项目	内容	CPU 模块					参考
		1)	2)	3)	4)	5)	
口令登录	禁止通过 GX Developer 读出 / 写入 CPU 模块的各文件的功能。	○	○	○	○	○	6.19.1 项
远程口令	在串行口通讯模块、Etherent 模块中防止来自于外部的非法存取的功能。	○	○	○	○	○	6.19.2 项
系统显示	连接 GX Developer，监视系统构成的功能。	○	○	○	○	○	6.20 节
LED 显示	通过 CPU 模块前面的 LED 显示 CPU 模块的动作状态的功能。	○	○	○	○	○	6.21 节
优先顺序的设定	通过对故障设定优先顺序，使 LED 显示处于灭灯状态。	○	○	○	○	×	6.21.2 项
高速中断功能	使用中断指针 I49，根据 0.2ms-1.0ms 间隔的恒定周期中断，执行中断程序的功能。	×	○	×	×	×	6.22 节
来自智能功能模块的中断	根据智能功能模块的中断请求，执行中断程序的功能。	○	○	○	○	○	6.23 节
串行口通讯功能	将 Q00CPU、Q01CPU 的 RS-232 接口与个人电脑 / 显示器等通过 RS-232 电缆连接，通过 MC 协议进行通信的功能。	○	×	×	×	×	6.24 节
模块服务间隔时间的读出	对智能功能模块、网络模块、周边设备的存取间隔时间（从 CPU 模块的存取受理起到下一次存取受理为止的时间）进行监视的功能。	×	○	○	○	×	6.25.1 项
服务处理设定	对 END 处理中执行的服务处理的次数 / 时间进行设定的功能。	×	×	×	×	○	6.25.2 项
软元件初始值	通过程序将程序使用的数据登录到软元件、智能功能模块的缓冲存储器上的功能。	○	○	○	○	○	6.26 节
电池长寿功能	通过使其仅保持时钟数据，延长电池使用寿命的功能。	×	×	×	×	○	6.27 节
存储器检查功能	检查是否由于过量的电磁噪声等导致 CPU 模块的存储器内容被改写的功能。	×	○	○	×	○	6.28 节
至标准 ROM 的锁存数据备份功能	在不使用电池的情况下保持软元件数据及故障历史记录等的锁存数据的功能。	×	×	×	×	○	6.29 节
向标准 ROM 进行软元件数据的写入 / 读取	向标准 ROM 进行软元件数据的写入 / 读取的功能。	×	×	×	×	○	6.30 节

○：可以使用　×：不可以使用

（下页续）

表 6.2 CPU 模块的功能一览表（续）

项目	内容	CPU 模块					参考
		1)	2)	3)	4)	5)	
在线模块的更换	在线中更换 Q 系列的输入 / 输出模块、功能版本 C 的智能功能模块、安装在 MELSECNET/H 远程 I/O 站的输入 / 输出模块、智能功能模块的功能。 将电源冗余的情况下，也可以在在线中更换电源模块。	×	×	○	○	×	*3
自动调节功能	自动调节功能是进行 PID 常量的初期设定的功能。 自动调节可以通过 S.PID、S.2PID 指令在温度调节等响应比较缓慢的系统中使用。	×	×	○	○	×	*4
冗余功能	将 CPU 模块、电源模块、网络模块、主基板冗余化的功能。	×	×	×	○	×	*5
系统切换（控制系统与待机系统的切换）功能	切换控制系统与待机系统（控制系统向待机系统、待机系统向控制系统切换。）的功能。 有系统切换与用户切换两种类型。	×	×	×	○	×	
更换运行模式	切换分开模式与备份模式的功能。	×	×	×	○	×	
热备传送功能	使控制系统与待机系统共享数据的功能。（将控制系统的数据传送到待机系统中。） 即使在控制系统故障、异常时发生系统切换，也可以通过相同的数据继续进行控制。	×	×	×	○	×	
在线程序写入的冗余跟踪功能	将通过可编程控制器写入、运行中写入写入到控制系统的数据传送到待机系统的功能。	×	×	×	○	×	
从控制系统向待机系统的存储复制功能	将控制系统的程序内存的内容复制到待机系统的功能。 （更换待机系统的 CPU 模块时，可以使控制系统与待机系统的程序内存的内容相同。） 有通过 GX Developer 的方法及通过特殊继电器、特殊寄存器的方法这两种类型。	×	×	×	○	×	

○：可以使用　×：不可以使用

*3: 关于可以进行在线模块更换的模块以及条件请参照下述手册。
☞ QCPU 用户手册（硬件设计 / 维护点检篇）

*4: 请参照下述手册。
☞ QnPHCPU/QnPRHCPU 编程手册（过程控制指令篇）

*5: 请参照下述手册。
☞ QnPRHCPU 用户手册（冗余系统篇）

6.2 恒定扫描

(1) 关于恒定扫描

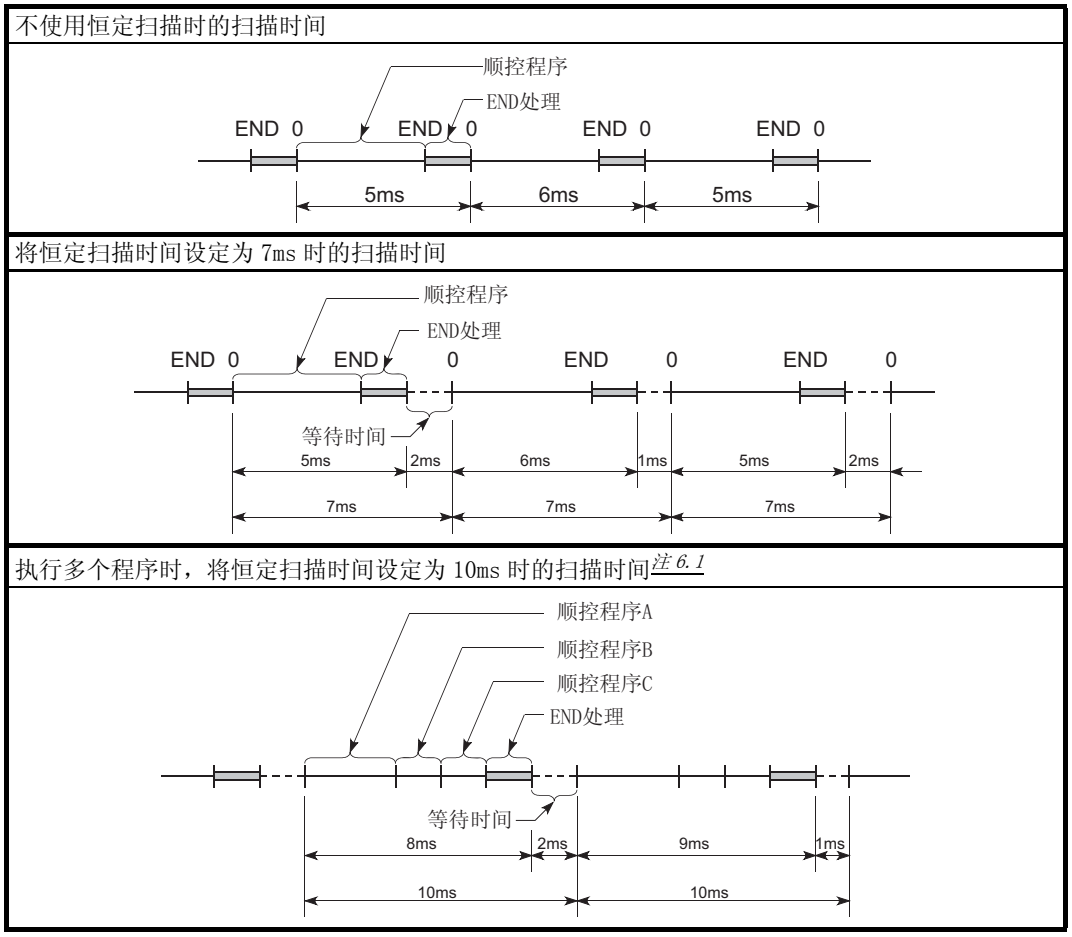
根据顺控程序中所使用的指令的执行与否，扫描时间的处理时间会有不同，在每个扫描周期发生不同的变化。

恒定扫描周期是指在保持一定的扫描时间的同时反复执行顺控程序的功能。

(2) 恒定扫描的用途

I/O 刷新在顺控程序执行前进行。

通过使用恒定扫描功能，即使顺控程序的执行时间发生变化，也可以保持一定的 I/O 刷新的时间间隔。



基本 注 6.1

图 6.1 恒定扫描的动作

基本 注 6.1

在基本模式 QCPU 中，由于不能执行多个程序的原因，因此，没有必要注意执行多个程序时的扫描时间。

(3) 恒定扫描时间的设定

恒定扫描时间的设定在可编程控制器参数的可编程控制器 RAS 设定中进行
恒定扫描时间在下述范围内可以进行设定。

- 基本模式 QCPU 的情况
  1-2000ms(设定单位为 1ms)
- 高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 的情况
  0.5-2000ms(设定单位为 0.5ms)

执行恒定扫描时，设定恒定扫描时间。
不执行恒定扫描时间时，将恒定扫描时间设定为 “空白”。

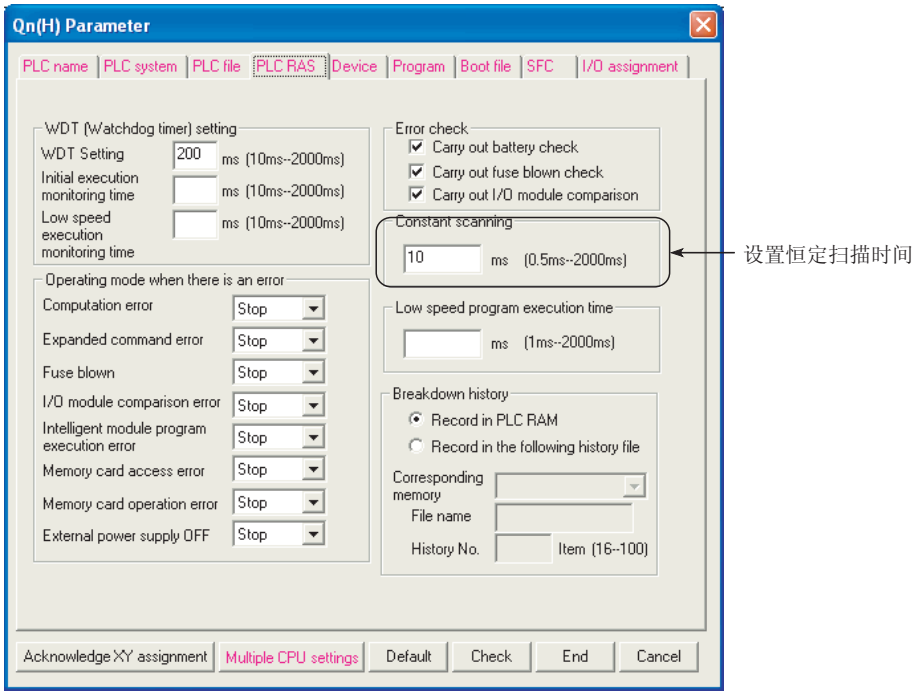


图 6.2 将恒定扫描时间设定为 10ms 的情况

(a) 设定时间的条件

请将恒定扫描的设定时间设定为满足下述关系式的值。

(WDT 的设定时间) > (恒定扫描的设定时间) > (顺控程序的最大扫描时间)

在顺控程序的扫描时间比恒定扫描的设定时间长的情况下，CPU 模块会检测出 PRG. TIME OVER(出错代码： 5010) 的错误。

此时，恒定扫描时间将被忽略，而根据顺控程序的扫描时间来执行。

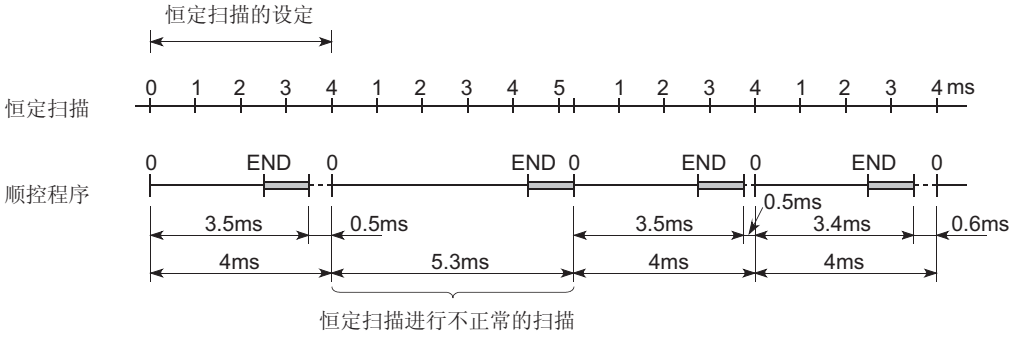


图 6.3 扫描时间比恒定扫描时间长时的动作

顺控程序的扫描时间比 WDT 的设定时间长的情况下，CPU 模块会检测出 WDT 出错。

此时，停止执行的程序。

(4) 关于从 END 处理执行开始到下一个扫描开始之前的等待时间

从顺控程序的 END 处理执行开始到下一个扫描开始之前的等待时间内，终止顺控程序的处理。

基本 注 6.2 冗余 注 6.2 通用 注 6.2

(a) 执行低速执行类型程序时*注 6.2*

执行低速执行类型程序时，在下述显示的时间内中断低速执行类型程序的执行。

（恒定扫描）-0.5ms

(b) 到下一个扫描开始之前的等待时间发生中断因子时

到下一个扫描开始之前的等待时间发生中断因子时，执行下述的程序：

基本 注 6.2

- 中断程序
- 恒定周期执行类型程序*注 6.2*

(c) 进行服务处理设定的情况 *注 6.3*

通过进行服务处理设定，在至下一个扫描开始为止的等待时间可以进行与外围设备（GX Developer 等）以及智能功能模块的通信服务的处理。(☞ 6.25.2 项)

基本 注 6.3 高性能 注 6.3 过程 注 6.3

冗余 注 6.3

基本 注 6.2 冗余 注 6.2 通用 注 6.2

在基本模式 QCPU 中，由于不能使用低速执行类型程序以及恒定周期执行类型程序，因此无需理会本内容。

在冗余 CPU、通用型 QCPU 中，由于不能使用低速执行类型程序，因此无需理会本内容。

基本 注 6.3 高性能 注 6.3 过程 注 6.3

冗余 注 6.3

在基本模式 QCPU、高性能模式 QCPU、过程 CPU、冗余 CPU 中，不能使用服务处理。

(5) 恒定扫描的精度

关于恒定扫描的精度，请参照第 10 章。

但是，执行下述 (a) 或者 (b) 显示的程序时，有偏离恒定扫描的可能性。

基本 注 6.4　冗余 注 6.4　通用 UD 注 6.4

(a) 执行低速执行类型程序时注 6.5

执行低速执行类型程序时，有 0.5ms 的等待时间。(☞图 3.31，图 3.32)

- 执行一个指令的最大处理时间为 0.5ms 以内的情况
  恒定扫描的误差与 10.2 节说明的恒定扫描的精度相同。
- 执行一个指令的最大处理时间超出 0.5ms 的情况
  恒定扫描时间只有在超出 0.5ms 时才有发生偏离的情况。
  另外，将检测出 PRG. TIME OVER(出错代码 :5010) 的错误。

关于一个指令大的最大处理时间，请参照下述手册。

☞QCPU(Q 模式)/QnACPU 编程手册 (公共指令篇)

基本 注 6.4

(b) 执行中断程序 / 恒定周期执行类型程序的情况注 6.5

在执行中断程序、恒定周期执行类型程序过程中，将处于中断禁止状态。

因此，在中断程序 / 恒定周期执行类型程序执行过程中，即使达到恒定扫描周期时间，也不能终止恒定扫描。

执行中断程序 / 恒定周期执行类型程序时，只有中断程序 / 恒定周期执行类型程序的执行时间有与恒定扫描时间发生偏离的情形。

基本 注 6.5　冗余 注 6.5　通用 UD 注 6.5

在基本模式 QCPU 中，由于不能使用低速执行类型程序以及恒定周期执行类型程序，因此无需理会本内容。

在冗余 CPU、通用型 QCPU 中，由于不能使用低速执行类型程序，因此无需理会本内容。

6.3 锁存功能

(1) 关于锁存功能

CPU 模块的各个软元件在下述情况下被清除，变为默认值（位软元件：OFF，字软元件：0）。

- 顺控程序的电源由 OFF→ON 时
- 进行复位操作时
- 超出瞬间掉电允许时间的停电

锁存功能是指在上述情况下保存软元件的内容的功能。

(2) 锁存使用时的运算

程序的运算与锁存的有无没有关系，都是一样的。

(3) 锁存功能的用途

在连续控制中进行数据的管理时，即使发生可编程控制器的电源 OFF、复位操作、超出瞬间掉电允许时间的停电等情况时，锁存也可以保持上述数据并继续进行控制。

(4) 可以使用锁存的的软元件

可以使用锁存的的软元件如下所示。

（默认值为只在锁存继电器中锁存才有效。）

- 锁存继电器 (L)
- 链接继电器 (B)
- 报警器 (F)
- 变址继电器 (V)
- 计时器 (T)
- 累计计时器 (ST)
- 计数器 (C)
- 数据寄存器 (D)
- 链接寄存器 (W)

要 点

如果设定了电池长寿功能（仅通用型 QCPU），则不能对锁存继电器进行锁存。

(5) 锁存范围的设定

锁存范围的设定在 GX Developer 的可编程控制器参数的软元件设定中进行。
锁存范围设定有锁存清除操作的有效范围与无效范围两种。

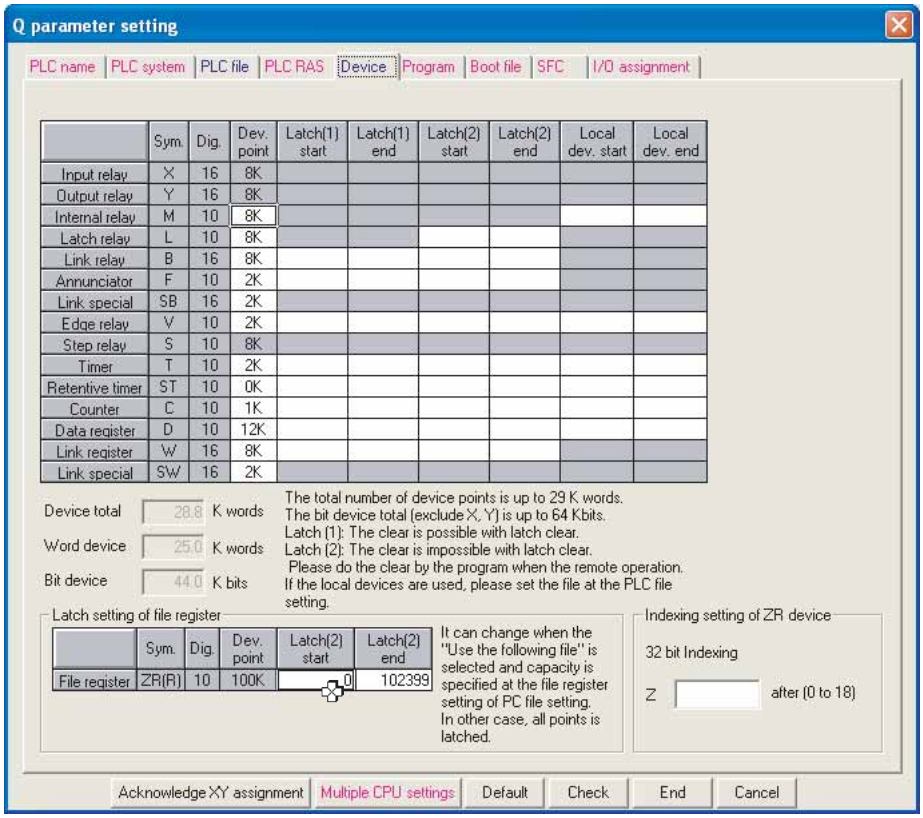


图 6.4 锁存范围的设定

要 点

在通用型 QCPU 中，可以设定文件寄存器的锁存范围。锁存范围以外的内容，在可编程控制器的电源 OFF → ON 时、CPU 模块的复位操作时将被清除。
在可编程控制器文件设定中选择 “使用以下文件”，进行了文件寄存器的容量设定后，设定锁存范围。
如果在可编程控制器文件设定中选择了 “使用与程序相同的文件名”，将不能设定文件寄存器的锁存范围。（保持文件寄存器的全部内容。）

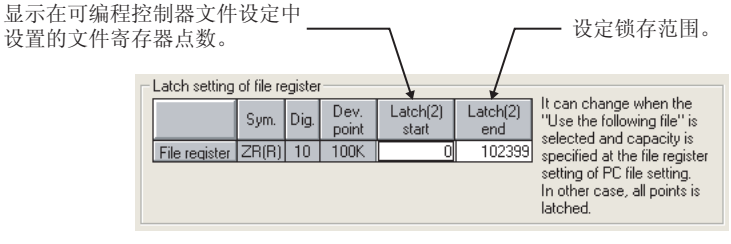


图 6.5 文件寄存器的锁存范围设定

通过 QDRSET 指令切换使用的文件寄存器时，文件寄存器的锁存范围设定将变为无效。切换后，与锁存范围的设定无关，将对文件寄存器的全部范围进行停电保持。

(6) 锁存范围的软元件数据的保持方法及对扫描时间的影响

(a) 基本模式 CPU、高性能模式 QCPU、过程 CPU、冗余 CPU 的情况
在将数据写入到锁存范围的软元件中的同时进行锁存。
由于不进行用于锁存的处理，因此锁存不会对扫描时间产生影响。

(b) 通用型 QCPU 的情况
END 处理时进行用于锁存的处理。因此，将发生锁存导致的扫描时间延迟。对软元件进行锁存时，应考虑扫描时间的延迟时间。(☞ 10.1.2 项 (11))

要 点

在通用型 QCPU 中，为了尽量减小锁存导致的扫描时间延迟，应尽可能地削减锁存点数 (锁存 (1) 设定、锁存 (2) 设定、锁存继电器 (L))。
通过以下替代方法可以削减锁存点数。

- 将锁存数据移至文件寄存器中。
- 将更新频度较少的软元件数据使用 SP.DEVST 指令存储到标准 ROM 中。(标准 ROM 中存储的软元件数据可以通过 S(P).DEVLD 读取)
  ☞ QCPU(Q 模式)/QnACPU 编程手册 (公共指令篇)

(7) 锁存范围的软元件数据的清除
进行锁存清除时的软元件的状态如表 6.3 所示。

表 6.3 进行锁存清除时的软元件的状态

锁存的有无	根据锁存清除进行的清除 / 保持
在锁存范围内未指定的软元件	清除
锁存 (1) 设定 (设定为 “可以通过锁存清除进行清除” 的软元件)	清除
锁存 (2) 设定 (设定为 “不可以通过锁存清除进行清除” 的软元件)	保持*

*: 关于清除方法，请参照 3.7 节。

要 点

文件寄存器 (R，ZR) 即使进行锁存清除也不能被清除。
文件寄存器 (R，ZR) 的清除通过顺控程序或者是 GX Developer 进行数据清除操作。(☞ 9.7 节 (3))

(8) 注意事项

基本
注 6.5

(a) 进行局部软元件或者软元件初始值指定的情况
即使是进行了锁存指定的软元件，如果进行了局部软元件指定[注 6.5] 或者软元件初始值指定便不能进行锁存。

(b) 关于电池的使用
锁存范围的软元件内容，通过在安装在 CPU 模块上的电池加以保持。
- CPU 模块即使在进行引导运行时，在进行锁存时也需要电池。
- 可编程控制器的电源处于 OFF 中时，如果将电池的连接器从 CPU 模块的连接器上拆除，将不能保存锁存范围的软元件的内容，会变成不确定的值，请加以注意。

(c) 冗余 CPU 的启动模式为热启动模式时
启动模式为热启动模式时，没有设定锁存范围的数据也将被保存。（变址寄存器以及步进继电器等一部分软元件除外）
在清除这些数据时，请进行锁存清除操作（☞ 3.7 节 (2)(b)）。

基本
注 6.5

在基本模式 QCPU 中，不能使用局部软元件指定。

6.4 停止状态与运行状态相互切换时的输出 (Y) 状态的设定

(1) 关于从停止状态到运行状态时的输出 (Y) 状态的设定

CPU 模块如果从运行等状态转向停止状态，则运行状态的输出 (Y) 将记忆在可编程控制器的内部并 OFF 所有的输出 (Y)。

在 GX Developer 的参数设置中，由 STOP 状态变为 RUN 状态时的状态可在下面的两种中选择：

- 输出停止前的输出 (Y) 状态
- 清除输出 (Y)

(2) 设定用途

在保持电路等中，在由停止状态→运行状态时，可以选择是否开启上次的输出状态。

图 6.6 保持电路

- 设定为输出停止前的输出 (Y) 状态

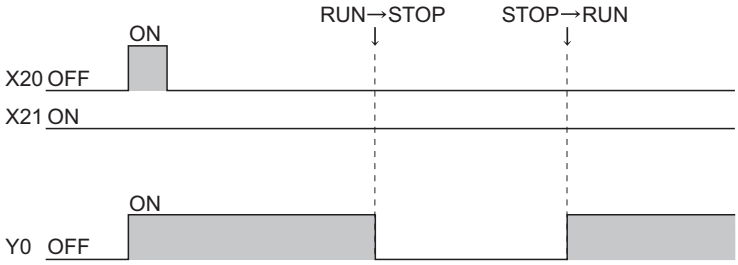


图 6.7 设定为输出停止前的输出 (Y) 状态时的时间图

- 设定为清除输出 (Y) 的情况

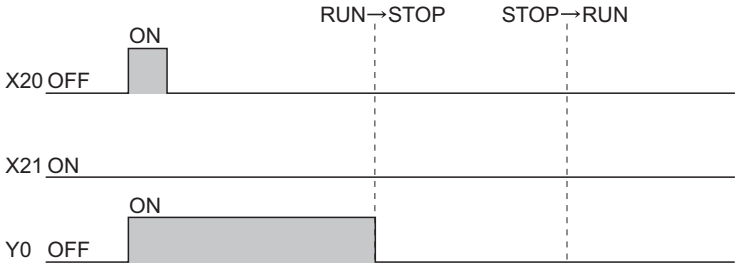


图 6.8 设定为清除输出 (Y) 时的时间图

(3) 从 STOP 状态变为 RUN 状态时的动作

(a) 输出停止前的输出 (Y) 状态 (默认)
在输出停止状态之前的输出状态后，进行顺控程序的运算。

(b) 输出 (Y) 清除
输出变为 OFF 状态。
在执行顺控程序的运算之后进行输出 (Y) 的输出。
关于 STOP 状态时输出 (Y) 被强制 ON 时的动作，请参阅 (5)。

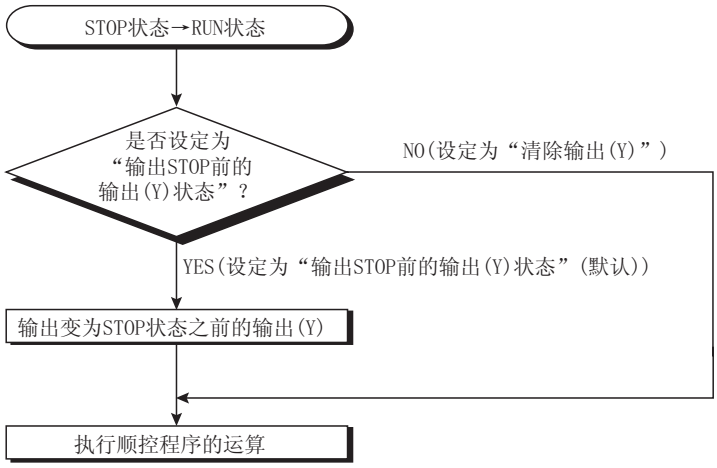


图 6.9 由 STOP 状态→ RUN 状态时的处理

(4) 由停止状态向运行状态转变时输出 (Y) 状态的设定

由停止状态向运行状态转变时输出 (Y) 状态的设定在可编程控制器参数的可编程控制器系统设定中进行。

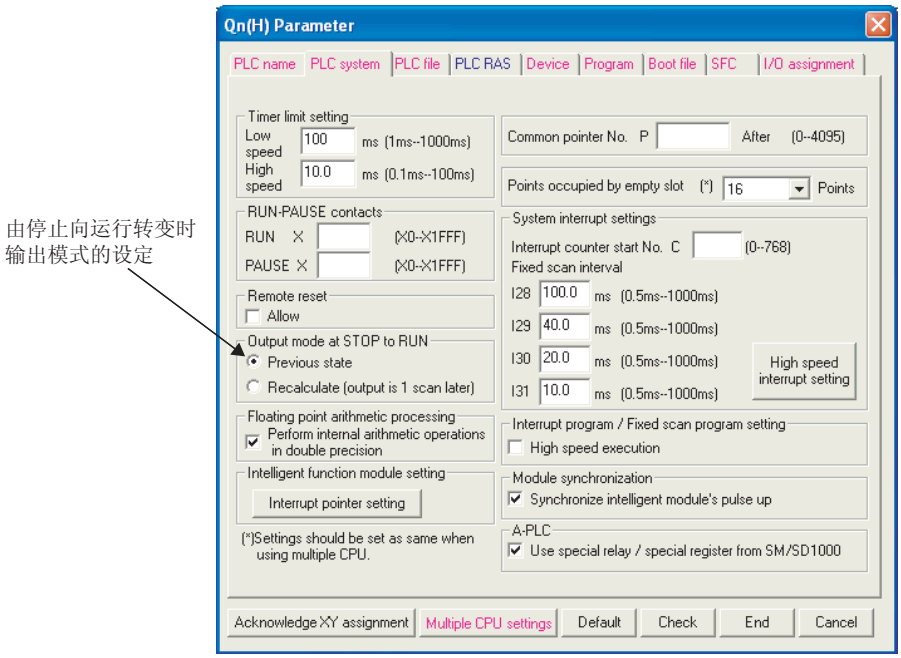


图 6.10 可编程控制器系统设定画面

(5) 注意事项

CPU 模块处于停止状态时对输出 (Y) 进行强制 ON 的情况下，由 STOP 状态变为 RUN 状态时的输出如表 6.4 所示。

表 6.4 STOP 状态时对输出 (Y) 进行强制 ON 后，由 STOP 状态变为 RUN 状态时的输出

STOP → RUN 时的输出模式	由 STOP 状态变为 RUN 状态时的输出
输出 STOP 前的输出 (Y) 状态	输出 STOP 前的输出 (Y) 状态。 STOP 前输出 (Y) 为 OFF 时，不保持 ON 状态。
清除输出 (Y)	保持 ON 状态。

6.5 时钟功能

(1) 关于时钟功能

时钟功能是指将 CPU 模块内部的时钟数据通过顺控程序读出，用于时间管理的功能。
时钟数据被用于向故障历史记录存储日期等 CPU 模块系统功能的时间管理等。

(2) 电源 OFF 以及瞬间掉电时的时钟动作

即使在可编程控制器的电源处于 OFF 状态中或者发生超出瞬间掉电允许时间的停电时，时钟动作可以通过 CPU 模块的内部电池继续进行。

(3) 时钟数据

时钟数据为 CPU 模块的内部中使用的时钟数据，如表 6.5 所示的内容。

表 6.5 时钟数据的内容

数据名称	内容	
年	阳历 4 位（可以测量 1980 年 -2079 年）	
月	1-12	
日	1-31（自动判别闰年）	
时	0-23(24 时间制)	
分	0-59	
秒	0-59	
星期	0	星期日
	1	星期一
	2	星期二
	3	星期三
	4	星期四
	5	星期五
	6	星期六

(4) 时钟数据的改变与读取

(a) 时钟数据的改变

时间数据的改变方法有通过 GX Developer 或通过程序进行改变的 2 种方法。

1) 通过 GX Developer 的方法

使用 GX Developer 时，通过［在线］→［时钟设定］显示时钟设定窗口，改变时钟数据。

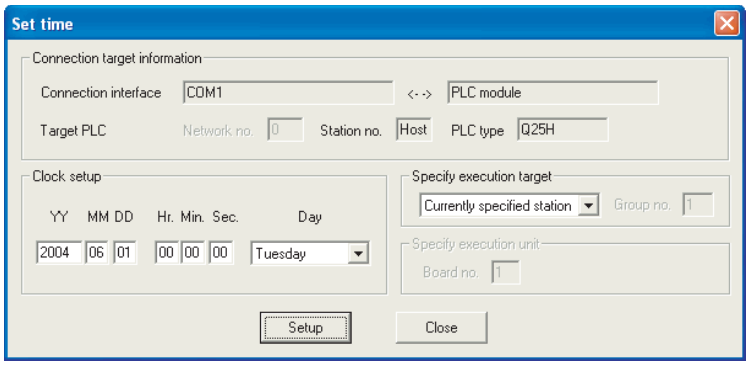


图 6.11 通过 GX Developer 的时钟数据写入

2) 通过程序的方法

在程序中通过时间数据的写入指令 (DATEWR) 改变时钟数据。

时钟数据的写入指令 (DATEWR) 中，D0~6 设定的时钟数据的写入程序如图 6.12 所示。

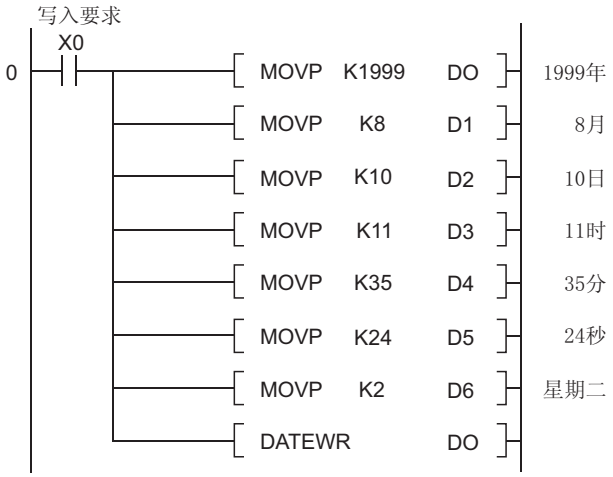


图 6.12 通过顺控程序对时钟数据的写入

关于 DATEWR 指令的详细情况，请参照下述手册。

☞QCPU(Q 模式)/QnACPU 编程手册（公共指令篇）

要 点

1. 如果通过 GX Developer 或者程序更改了时钟数据，精度为 1/1000 秒的时钟将被复位为 0。
2. 在 GX Developer 中，时钟数据的年的最大设定值为 2037 年。

(b) 时钟数据的读出

将时钟数据读取到数据寄存器中时，应在程序中使用以下的某一个指令；

- 时钟数据的读出指令 (DATERD)
- 扩展时钟的读取指令 (S(P).DATERD)

将通过时钟数据的读取指令 (DATERD) 读取的时钟数据存储到 D10 ～ 16 中的程序示例如图 6.12 所示。

图 6.13 存储时钟数据的程序

*: 存储在 D10-16 的数据如图 6.14 所示。

D10	2004	阳历4位
D11	4	月
D12	1	日
D13	11	时
D14	35	分
D15	24	秒
D16	2	星期

（☞本节 (3)）

图 6.14 时钟数据的读出

关于 DATERD 指令的详细情况，请参照下述手册。

☞QCPU(Q 模式)/QnACPU 编程手册（公共指令篇）

要 点

时钟数据的写入 / 读出也可以通过特殊继电器 (SM210-213) 与特殊寄存器 (SD210-213) 进行。

关于特殊继电器的详细情况请参照附录 1，关于特殊寄存器的详细情况请参照附录 2。

(5) 注意事项

(a) 初次使用时的时钟数据的设定
时钟数据出厂时没有设定。
时钟数据在故障历史记录等 CPU 模块的系统以及智能功能模块中使用。
初次使用 CPU 模块时，请务必设定正确的时间。

(b) 时钟数据的修改
即使修改时钟数据的一部分，也有必要将所有数据再次写入 CPU 模块中。

(c) 时钟数据改变的范围
改变时钟数据时，请在本节 (3) 所示的范围内写入。
将不可能的时间数据写入 CPU 模块时，不能进行正常的时钟动作。
但是，在本节 (3) 说明的范围内即使写入不可能的时间数据，也不会出现出错。

表 6.6 写入时钟数据的数据示例

	对 CPU 模块的写入动作	CPU 模块的动作状态
2 月 30 日	执行	没有检测出出错
13 月 32 日	不执行	执行 DATEWR 指令时：OPERATION ERROR （出错代码：4100） SM210ON 时：SM211ON

(d) 精度为 1/1000 秒的时钟数据的使用

1) 可使用精度为 1/1000 秒的时钟数据的功能
只有在以下的指令中可以使用：
- S(P).DATERD
- S(P).DATE+
- S(P).DATE-

在除上述指令以外的指令中不能使用精度为 1/1000 秒的时钟数据。
（通过 SM/SD 进行的读取、故障历史记录中存储的出错的发生时间、通过 GX Developer 的读取、通过其它模块的专用指令的读取等。）

2) 更改了时钟数据时
如果通过 GX Developer 或者指令（包括其它模块的专用指令）更改了时钟数据，1/1000 秒的时钟数据将被复位为 0。

(6) 时钟数据的精度

时钟功能的精度根据周围温度而不同，如下所示。

表 6.7 基本模式 QCPU 的精度

周围的温度（℃）	精度（每日误差，S）
0	-3.2 ～ +5.27(TYP.+1.98)
+25	-2.57 ～ +5.27(TYP.+2.22)
+55	-11.68 ～ +3.65(TYP.-2.64)

表 6.8 高性能模式 QCPU、过程 CPU 的精度

周围的温度（℃）	精度（每日误差，S）
0	-3.18 ～ +5.25(TYP.+2.12)
+25	-3.93 ～ +5.25(TYP.+1.9)
+55	-14.69 ～ +3.53(TYP.-3.67)

表 6.9 冗余 CPU 的精度

周围的温度（℃）	精度（每日误差，S）
0	-3.2 ～ +5.27(TYP.+2.07)
+25	2.77 ～ +5.27(TYP.+2.22)
+55	-12.14～ +3.65(TYP.-2.89)

表 6.10 通用型 QCPU 的精度

周围的温度（℃）	精度（每日误差，S）
0	-2.96 ～ +3.74(TYP.+1.42)
+25	-3.18 ～ +3.74(TYP.+1.50)
+55	-13.20～ +2.12(TYP.-3.54)

(7) 时钟数据的比较

将时钟数据读出并通过顺控程序进行比较时，请通过时钟数据读出指令 (DATERD) 读出时钟数据。

在时钟数据读出指令 (DATERD) 中，由于年份数据是以阳历的 4 位读出的，因此，可以直接在原状态下用比较指令进行比较。

6.6 远程操作

远程操作是指通过从外部（GX Developer、使用 MC 协议的外部设备、MELSECNET/G 注 6.6 注 6.7 或者 MELSECNET/H 网络模块的链接专用指令、远程触点等）的操作来变更 CPU 模块的动作状态的操作。

远程操作有下述 4 种类型

- 远程 RUN/STOP : ☞6.6.1 项
- 远程 PAUSE : ☞6.6.2 项
- 远程复位 : ☞6.6.3 项
- 远程锁存清除 : ☞6.6.4 项

6.6.1 远程 RUN/STOP

(1) 关于远程 RUN/STOP

远程 RUN/STOP 是指将 CPU 模块的 RUN/STOP 开关（基本模式 QCPU、通用型 QCPU 时为 RUN/STOP/RESET 开关）置于 RUN 的位置并通过外部进行 CPU 模块的 RUN/STOP 的操作。

(2) 远程 RUN/STOP 的用途

在下述情况下，通过隔离操作进行远程 RUN/STOP 比较方便。

- CPU 模块处于不在身边的情况。
- 通过外部信号对控制盘的 CPU 模块进行 RUN/STOP 时。

(3) 远程 RUN/STOP 时的运算

进行远程 RUN/STOP 时的运算如下所示。

(a) 远程 STOP

在程序执行至 END 指令后，进入 STOP 状态。

(b) 远程 RUN

通过远程 STOP 进入停止状态时，如果进行远程运行则将再次进入 RUN 状态并从步 0 开始执行程序。

在高性能模式 QCPU 中使用 MELSECNET/G 时，应确认 CPU 模块以及 GX Developer 的版本。
（☞ 附录 4.2）

在基本模式 QCPU、过程 CPU、冗余 CPU 中，不能使用 MELSECNET/G。

(4) 远程 RUN/STOP 的方法

高性能 注 6.8

远程 RUN/STOP 有通过 “RUN 触点”以及通过 “GX Developer、使用 MC 协议的外部设备、MELSECNET/G 注 6.8 注 6.9 或者 MELSECNET/H 网络模块的链接专用指令”这两种方法。

基本 过程 冗余 注 6.9 注 6.9 注 6.9

(a) 通过 RUN 触点的方法

RUN 触点在可编程控制器参数的可编程控制器系统设定中进行设定。

可以设定的软元件范围如表 6.11 所示。

表 6.11 RUN 触点可以设定的软元件范围

CPU 模块	可以设定的软元件范围
基本模式 QCPU	输入 X0-7FF
高性能模式 QCPU 过程 CPU 冗余 CPU 通用型 QCPU	输入 X0-1FFF

通过设定 RUN 触点的 ON/OFF，可以进行远程 RUN/STOP。

- RUN 触点在 OFF 时，CPU 模块处于 RUN 状态。
- 运行触点在 ON 时，CPU 模块处于 STOP 状态。

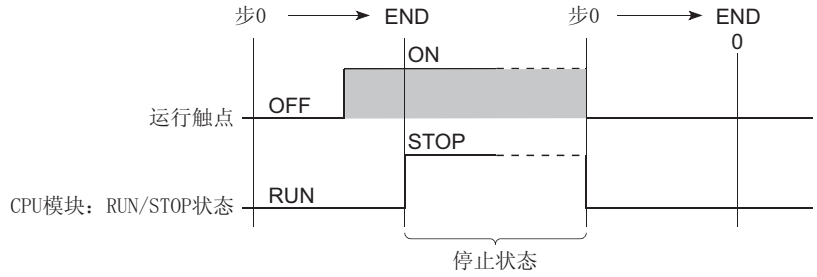


图 6.15 通过 RON 触点的 RUN/STOP

高性能 注 6.8

在高性能模式 QCPU 中使用 MELSECNET/G 时，应确认 CPU 模块以及 GX Developer 的版本。
(附录 4.2)

基本 过程 冗余 注 6.9 注 6.9 注 6.9

在基本模式 QCPU、过程 CPU、冗余 CPU 中，不能使用 MELSECNET/G。

(b) 通过 GX Developer、使用 MC 协议的外部设备等的方法

通过 GX Developer、使用 MC 协议的外部机器进行的远程 RUN/STOP 操作可以进行 CPU 模块的 RUN/STOP。

GX Developer 的操作通过 [在线]→[远程操作] 进行。

来自外部设备的 RUN/STOP 操作通过 MC 协议的指令进行。

☞Q 系列对应 MELSEC 通讯协议参考手册

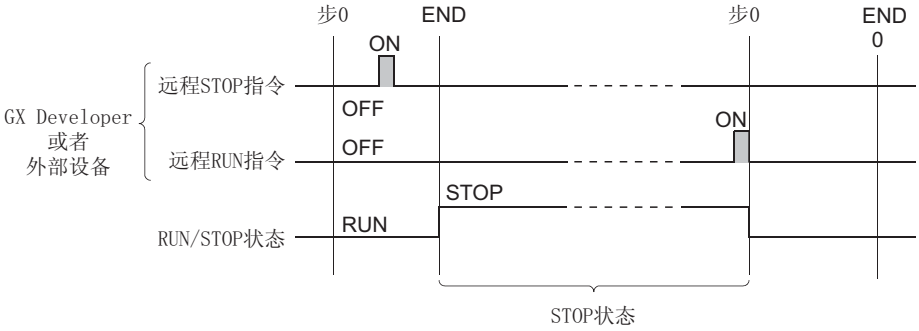


图 6.16 通过 GX Developer、外部机器进行的远程 RUN/STOP

(5) 注意事项

由于 CPU 模块为 STOP 优先的原因，请注意以下要点：

(a) 进入 STOP 状态的时机

如果使用 RUN 触点、GX Developer、使用 MC 协议的外部设备，MELSECNET/H 网络模块的链接专用指令中的任何一个进行远程 STOP，CPU 模块将变为 STOP 状态。

(b) 远程 STOP 后再度进入 RUN 状态的情况

通过远程 STOP 使 CPU 模块处于 STOP 状态后，再度进入 RUN 状态时，请按照最初进行远程 STOP 的顺序进行远程 RUN。

(c) 构筑冗余 CPU 的冗余系统时

1) 不通过远程操作画面指定两系统的情况

只对通过链接对象指定等指定的系统进行远程 RUN/STOP。

2) 通过远程操作画面进行两系统指定的情况

远程 RUN/STOP 在控制系统与待机系统两系统中进行。

但是，通过两系统指定的远程操作只在冗余 CPU 处于备份模式时才可能进行。

要 点

1. RUN/STOP 状态如下所示。
   - RUN 状态 ..... 反复执行顺控程序的步 0-END/FEND 指令的运算的状态。
   - STOP 状态 ..... 在顺控程序的运算 STOP 状态下，输出 (Y) 的所有点将 OFF。
2. 如果进行 CPU 模块的复位，复位后将进入 RUN/STOP 开关设定的状态。

6.6.2 远程 PAUSE

(1) 关于远程 PAUSE

远程 PAUSE 是指，将 CPU 模块的 RUN/STOP 开关（基本模式 QCPU、通用型 QCPU 时为 RUN/STOP/RESET 开关）置于 RUN 的位置并通过外部使 CPU 模块进入 PAUSE 状态的操作。

PAUSE 状态是指保持整个输出 (Y) 原有的 ON/OFF 状态不变并停止 CPU 模块运算的状态。

(2) 远程 PAUSE 的用途

在过程控制等情况下，CPU 模块 RUN 状态时输出 (Y) 为 ON 时，如果想在 STOP 状态时也使其保持为 ON，这时便可使用远程 PAUSE 功能。

(3) 远程 PAUSE 的方法

远程 PAUSE 有通过 “PAUSE 触点”以及通过 “GX Developer、使用 MC 协议的外部设备”这两种方法。

(a) 通过 PAUSE 触点的方法

PAUSE 触点在可编程控制器参数的可编程控制器系统设定中进行设定。

可以设定的软元件范围如表 6.12 所示。

表 6.12 PAUSE 触点可以设定的软元件范围

CPU 模块	可以设定的软元件范围
基本模式 QCPU	输入 X0-7FF
高性能模式 QCPU 过程 CPU 冗余 CPU 通用型 QCPU	输入 X0-1FFF

- PAUSE 触点与 PAUSE 允许线圈 (SM206) 同时 ON 的扫描 END 处理执行时，PAUSE 触点 (SM204) 将 ON。
  在 PAUSE 触点 ON 的下一次扫描执行 END 指令时，将进入 PAUSE 状态并停止运算。
- 如果 PAUSE 触点 OFF 或者是 SM206OFF，PAUSE 状态将被解除，将再度从步 0 开始进行顺控程序的运算。

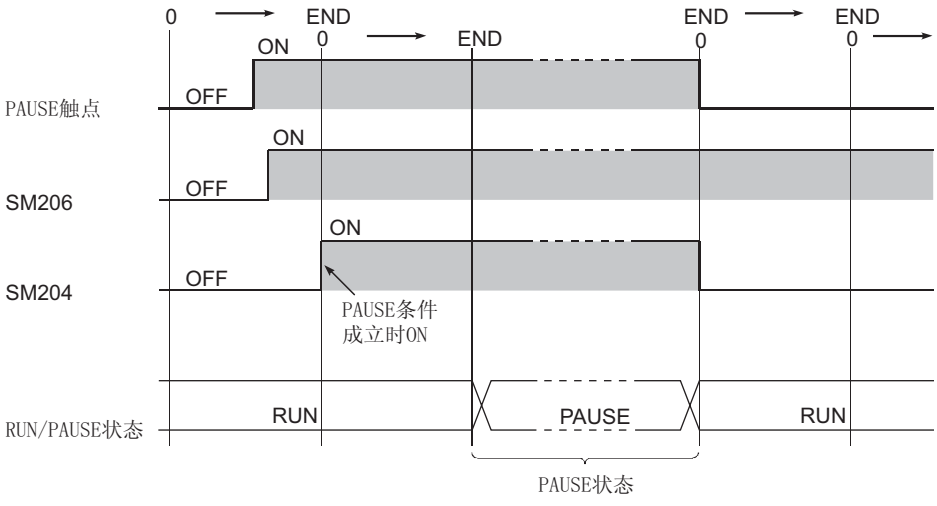


图 6.17 通过 PAUSE 触点 PAUSE 时的时间图

要 点

不能只对 PAUSE 触点进行设定。
在设定 PAUSE 触点时，请对 RUN 触点也进行设定。

(b) 通过 GX Developer、使用 MC 协议的外部设备的方法

通过 GX Developer、使用 MC 协议的外部设备进行的远程 PAUSE 操作可以进行 CPU 模块的远程 PAUSE。
GX Developer 的操作通过 [在线]→[远程操作] 进行。

来自使用 MC 协议的外部设备的情况下，通过 MC 协议的指令进行。
☞Q 系列对应 MELSEC 通讯协议参考手册

- 在执行输入了 PAUSE 触点指令的扫描 END 处理时，PAUSE 触点 (SM204) 将 ON。在 PAUSE 触点 ON 的下一次扫描执行 END 指令时，将进入 PAUSE 状态并停止运算。
- 如果输入远程运行指令，将再度从步 0 开始进行顺控程序的运算。

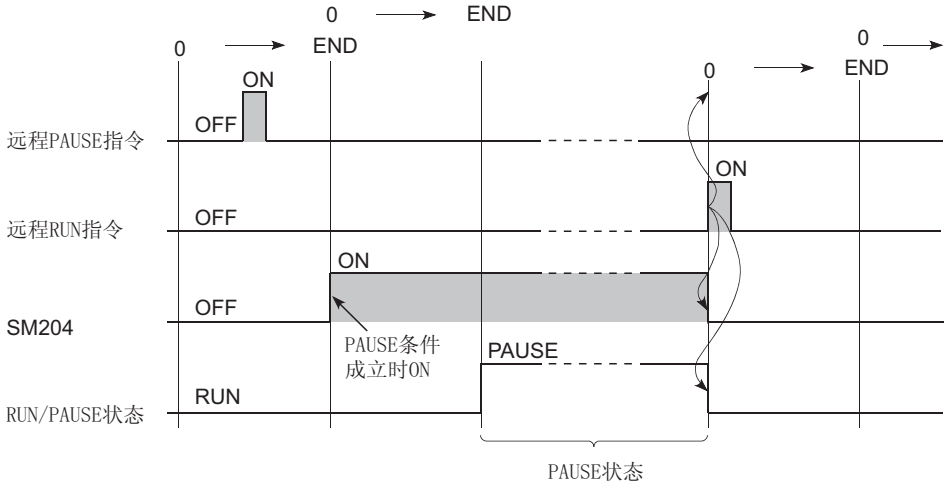


图 6.18 通过 GX DeveloperPAUSE 时的时间图

(4) 注意事项

(a) 预置强制 ON 或者 OFF 的状态的情况

预置 PAUSE 状态下的强制 ON 或者 OFF 状态时，请取消 PAUSE 触点 (SM204) 的互锁

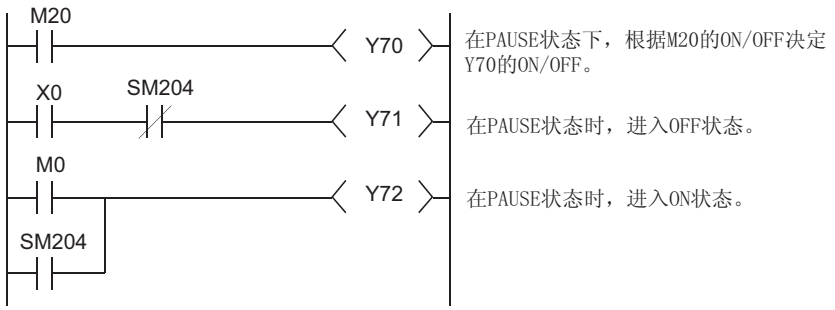


图 6.19 PAUSE 状态下为进行输出 (Y) 的强制 ON 或者 OFF 状态的程序示例

(b) 冗余 CPU 中冗余系统的构筑

1) 通过远程操作画面不进行两系统指定的情况
只对通过链接对象指定等指定的系统进行远程 PAUSE。

2) 通过远程操作画面进行两系统指定的情况
远程 PAUSE 在控制系统与待机系统两系统中进行。
但是，通过两系统指定的远程操作只在冗余 CPU 处于备份模式时才可能进行。

6.6.3 远程 RESET（远程复位）

(1) 关于远程复位

远程复位是指 CPU 模块处于 STOP 状态时，通过来自外部的操作进行 CPU 模块的复位操作。

另外，即使 RUN/STOP 开关（基本模式 QCPU、通用型 QCPU 时为 RUN/STOP/RESET 开关）处于 RUN 的位置，CPU 模块通过自诊断功能检测出发生的错误并处于停止状态时可以进行复位。

(2) 远程复位的用途

在手触及不到的远处的 CPU 模块发生出错时，通过远程操作可以对其进行复位。

(3) 远程复位的方法

远程复位只有通过来自 GX Developer、装有 MC 协议的外部设备等的操作才能进行。

远程复位按照如下的步骤进行。

- 在可编程控制器参数的系统设定中，将远程复位设定为 “许可”，对 CPU 模块写入可编程控制器参数。

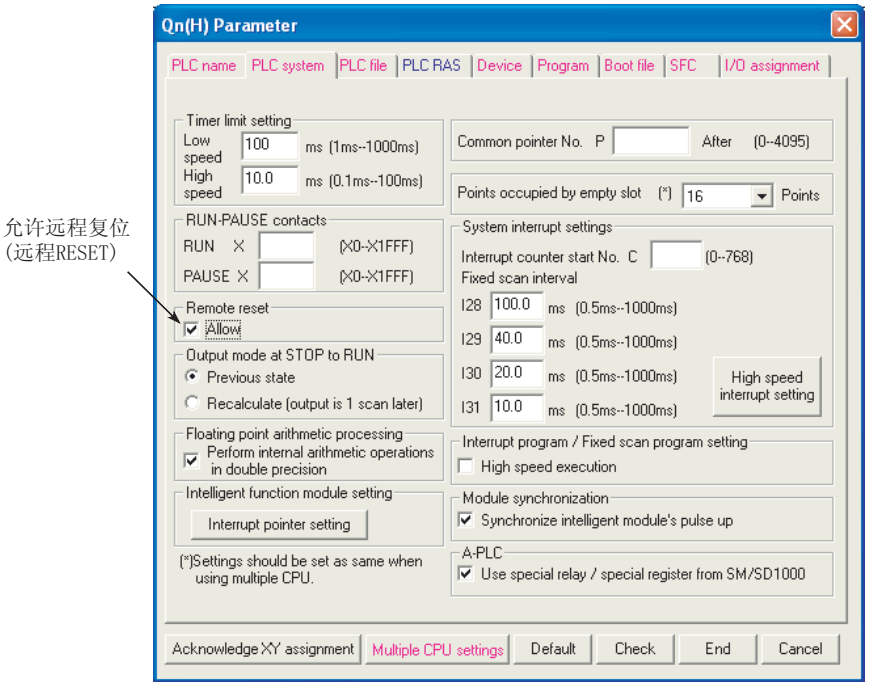


图 6.20 远程 RESET 必要的设定

- CPU 模块处于运行状态时，通过远程停止操作使其进入停止状态。
- 通过远程复位操作对 CPU 模块进行复位。

  使用 GX Developer 的情况下，通过［在线］→［远程操作］进行。

  使用装有 MC 协议的外部设备的情况下，通过 MC 协议的指令进行。

  ☞Q 系列对应 MELSEC 通讯协议参考手册。

(4) 注意事项

(a) 远程复位时的设定
进行远程复位时，在可编程控制器参数的可编程控制器系统设定中将远程复位设定为 “许可”，对 CPU 模块写入可编程控制器参数。
没有将远程复位设定为 “许可”的情况下，不能进行远程复位。

(b) 运行状态时的远程复位
CPU 模块处于运行状态时，不能通过远程复位进行复位。
通过远程停止等操作使 CPU 模块处于停止状态后，再进行远程复位。

(c) 复位处理结束后的状态
进行了远程复位的 CPU 模块在复位处理结束后，CPU 模块将变为 RUN/STOP 开关（基本模式 QCPU、通用型 QCPU 时为 RUN/STOP/RESET 开关）中设定的运行状态。

- RUN/STOP 开关（基本模式 QCPU、通用型 QCPU 时为 RUN/STOP/RESET 开关）处于 STOP 的位置时，复位结束后 CPU 模块变为 STOP 状态。
- RUN/STOP 开关（基本模式 QCPU、通用型 QCPU 时为 RUN/STOP/RESET 开关）处于 RUN 的位置时，复位结束后 CPU 模块变为 RUN 状态。

(d) 噪声干扰引起异常发生时
在由于噪声干扰引起 CPU 模块发生异常时，通过远程复位有不能进行复位的情况，请加以注意。
在不能通过远程复位进行复位时，请通过 RESET/L.CLR 开关（基本模式 QCPU 、通用型 QCPU 时为 RUN/STOP/RESET 开关） 进行复位或者重新启动可编程控制器的电源。

(e) 冗余 CPU 中构筑冗余系统时
备份模式时的远程复位是对控制系统进行的，如果进行远程复位，两系统都将被复位。
在独立模式与调试模式时，只能对通过 GX Developer 链接对象指定等指定的系统进行复位。

1) 备份模式时的远程复位对象
备份模式时的远程复位，请对控制系统进行。（如果对控制系统进行远程复位，两系统都将被复位）
如果对待机系统进行复位，会出现出错（出错代码：4240H）。

2) 备份模式时的控制系统与待机系统的 RUN/STOP 状态不一致的情况
控制系统处于停止状态下进行远程复位时，请将待机系统也设为停止状态。
控制系统为停止状态，待机系统为运行状态下如果进行远程复位，将发生系统切换。
另外，控制系统为运行状态、待机系统为停止状态下如果进行远程复位，则只有待机系统被复位

3) 备份模式时待机系统中发生 WDT 出错时
待机系统中发生 WDT 出错时，即使只对控制系统进行远程复位，待机系统也将被复位。
此时，请通过下述所示的路径（不经由热备电缆的通讯路径）进行远程复位。
- 将 PC(GX Developer) 直接连接在待机系统的冗余 CPU 上。
- 经由待机系统的模块进行远程复位。（通过 MC 协议进行远程复位等）

4) 备份模式时通过其它路径进行远程复位时
如 6.6.5 项 (3) 所示，对正在进行远程操作的 CPU 模块不能进行来自其它 GX Developer 的远程操作。
如图 6.21 所示，通过其它路径对控制系统以及待机系统进行远程操作时，即使对控制系统进行远程操作，待机系统也有不被复位的情况。

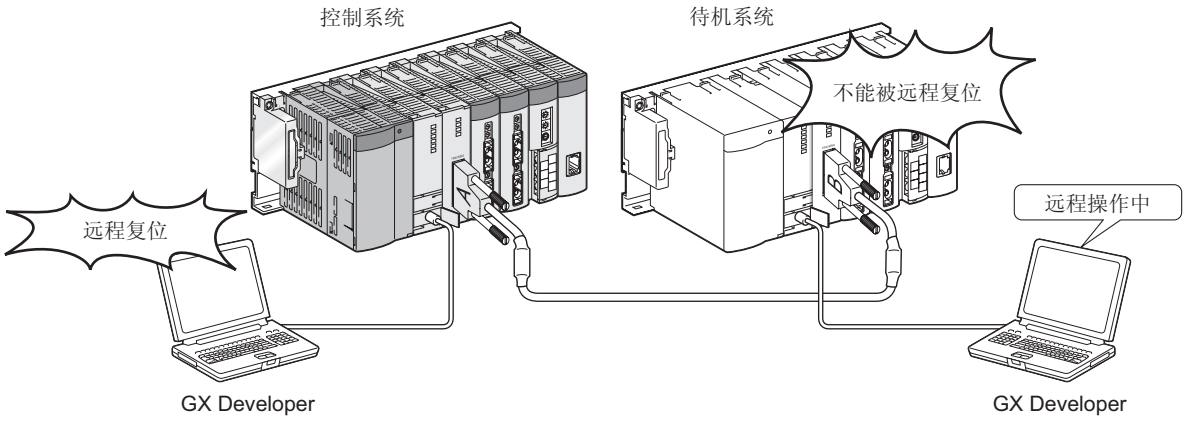


图 6.21 经由其它路径进行远程操作的情况

对待机系统进行远程操作时，请解除对待机系统的远程操作后，进行对控制系统的远程复位。

要 点

1. CPU 模块由于出错处于停止状态时，如果进行远程复位，在复位处理结束时 CPU 模块将处于 RUN/STOP 开关（ 基本模式 QCPU、通用型 QCPU 时为 RUN/STOP/RESET 开关）中设定的运行状态，请加以注意。
2. 即使未将可编程控制器参数的可编程控制器系统设定中的远程复位设定为“许可”，GX Developer 的远程处理也会结束。
   但是，由于 CPU 模块没有进行复位处理，因此不能被复位。
   即使通过 GX Developer 进行远程复位 CPU 模块的状态也没有发生变化时，请确认可编程控制器参数的可编程控制器系统设定中远程复位是否被设定为“许可”。

6.6.4 远程锁存清除

(1) 关于远程锁存清除

远程锁存清除是指在 CPU 模块处于停止状态时，通过 GX Developer 等对被锁存的软元件数据进行复位的操作。

(2) 远程锁存清除的用途

在 CPU 模块安装在下述场所的情况下，希望进行锁存清除时比较方便。
此时，与远程 RUN/STOP 组合使用。

- CPU 模块安装在手触及不到的场所时。
- 通过外部对控制盘内的 CPU 模块进行锁存清除时。

(3) 远程锁存清除的方法

远程锁存清除只有通过来自 GX Developer、装有 MC 协议的外部设备等的操作才能进行。
远程锁存清除按照如下的步骤进行。

- 通过远程停止使 CPU 模块进入停止状态。
- 通过远程锁存清除操作对 CPU 模块进行锁存清除。
  GX Developer 的操作通过 [在线]→[远程操作] 进行。
  通过装有 MC 协议的外部设备进行控制时，通过 MC 协议的指令进行。
  ☞Q 系列对应 MELSEC 通讯协议参考手册。
- 远程锁存清除结束后，希望进入 RUN 状态时，请进行远程 RUN 操作。

(4) 注意事项

(a) 运行状态下的锁存清除

CPU 处于运行状态时，不能进行锁存清除。

(b) 锁存清除操作有效 / 无效的范围

在可编程控制器参数的软元件设定中设定的软元件的锁存范围中，存在着锁存操作有效范围与无效范围。
在远程锁存清除操作中，软元件范围只有设定在 “锁存 (1)” 的范围内才能被复位。
关于被设定为锁存清除无效的软元件的复位方法，请参照 3.7 节 (2)(c)。

(c) 执行远程锁存清除时被复位的软元件
在执行远程锁存清除时，没有被锁存的软元件也将被复位。

(d) 在冗余 CPU 中构筑冗余系统时

1) 通过远程操作画面未进行两系统指定的情况
只对通过链接对象指定等指定的系统进行远程锁存清除。

2) 通过远程操作画面进行了两系统指定的情况
远程锁存清除在控制系统与待机系统两系统中进行。
但是，通过两系统指定的远程操作只在冗余 CPU 处于备份模式时才可以进行。

6.6.5 远程操作与 CPU 模块的 RUN/STOP 状态的关系

(1) 远程操作与 CPU 模块的 RUN/STOP 状态的关系

根据远程操作与 CPU 模块的 RUN/STOP 状态的组合，CPU 模块的动作状态如表 6.13 所示。

表 6.13 RUN/STOP 状态与远程操作的关系

RUN/STOP 状态	远程操作				
	运行 *1	停止	PAUSE *2	复位 *3	锁存清除
运行	运行	停止	PAUSE	不可以操作 *4	不可以操作 *4
停止	停止	停止	停止	复位 *5	锁存清除

*1: 通过 RUN 触点进行时，有必要在可编程控制器参数的可编程控制器系统设定中设定 “RUN-PAUSE 触点”。

*2: 通过 PAUSE 触点进行时，有必要在设定可编程控制器参数的可编程控制器系统设定中设定 “RUN-PAUSE 触点”。另外，有必要预先使远程 PAUSE 许可线圈 (SM206)ON。

*3: 有必要将可编程控制器参数的可编程控制器系统设定中的 “远程复位” 设定为 “许可”。

*4: 通过远程 STOP 使 CPU 模块进入 STOP 状态时，可以进行复位或者锁存清除。

*5: 包括 CPU 模块由于出错而处于停止的情况。

(2) 来自同一个 GX Developer 的远程操作

通过同一个 GX Developer 进行远程操作时，进入其后执行的远程操作的状态。

(3) 来自多个 GX Developer 的远程操作

对通过一个 GX Developer 正在进行远程操作的 CPU 模块，不能从另一个 Developers 对其进行远程操作。

(a) 通过正在进行远程操作的 GX Developer 进行远程 RUN

如果打算通过其它 GX Developer 进行远程操作时，请先由正在进行远程操作的 GX Developer 进行远程 RUN，并解除远程操作。

例如：通过一个 Developers 正在进行远程 PAUSE 时，即使通过其它 GX Developer 进行远程 STOP/ 远程 RUN，仍会保持 PAUSE 状态不变。

只有通过正在进行远程 PAUSE 的 GX Developer 进行远程 RUN，并解除远程操作后，才可以通过其它 GX Developer 进行远程操作。

6.7 Q 系列对应模块的输入响应时间的选择（I/O 响应时间）

(1) 关于输入响应时间的选择

是指将 Q 系列对应模块的输入响应时间以模块为单位进行变更的功能。

可以进行输入响应时间变更的模块与可以设定的时间如表 6.14 所示。

表 6.14 可以进行输入响应时间变更的模块

可以进行输入响应时间变更的模块	类型	可设置的时间
输入模块	输入	1ms、5ms、10ms、20ms、70ms
I/O 混和模块	I/O 混和	（默认值：10ms）
高速输入模块	高速输入	0.1ms、0.2ms、0.4ms、0.6ms、1ms
中断模块	中断	（默认值：0.2ms）

上述显示的 Q 系列对应的模块以设定的输入响应时间接受外部输入。

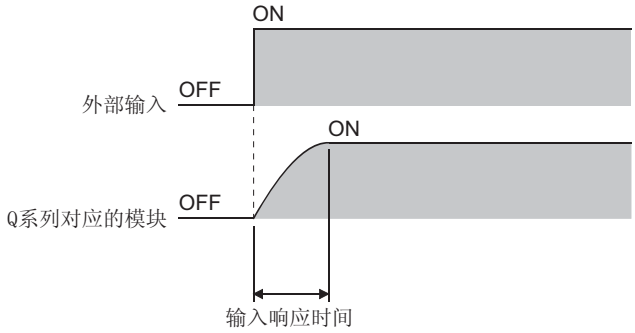


图 6.22 输入响应时间

(2) 输入响应时间的设定

输入响应时间的设定在可编程控制器参数的 I/O 分配设定中进行。

1) 进行 I/O 分配设定

2) 选择 Detailed setting（详细设定）按钮

3) 在 I/O 模块、智能功能模块的详细设定画面中，设定输入响应时间。

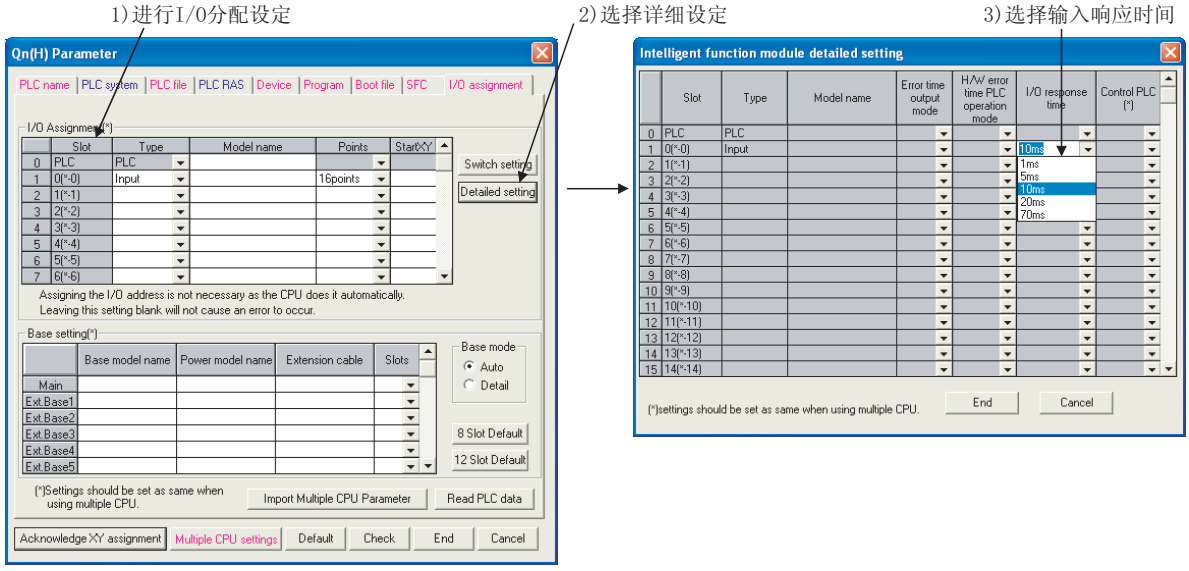


图 6.23 输入响应时间的设定

(3) 注意事项

(a) 关于可以设定模块的 GX Developer 版本的限制

变更高速输入模块或者是中断模块的输入响应时间时，请使用表 6.15 所示版本以后的 GX Developer。

使用表 6.15 所示版本以前的 GX Developer 时，以输入响应时间的默认值进行动作。

表 6.15 可以设定的 GX Developer 的版本

模块	可以设定的 GX Developer 的版本
高速输入模块	GX Developer 版本 5(SW5D5C-GPPW) 以后
中断模块	GX Developer 版本 6(SW6D5C-GPPW) 以后

(b) 缩短输入响应时间时

如果将输入响应时间的设定设定为高速，其抗噪声干扰的的能力会变弱。

请考虑使用的环境进行输入响应时间的设定。

基本 过程 冗余
注 6.10 注 6.10 注 6.10

(c) 使用 Ans/A 系列对应的输入模块时*注 6.10*

不能变更 Ans/A 系列对应的输入模块的输入响应速度。

对 Ans/A 系列对应的输入模块或者中断模块的插槽进行输入响应速度的设定时将视为无效。

通用
UD
注 6.10

(d) 有效设定的时机

输入响应时间的设定在下述情况下有效。

- 可编程控制器的电源处于 OFF→ON 时
- CPU 模块处于复位解除时

基本 过程 冗余
注 6.10 注 6.10 注 6.10

通用
UD
注 6.10

基本模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 中不能使用 Ans/A 系列对应的模块。

6.8 出错时输出模式的设定

(1) 关于出错时输出模式设定

出错时输出模式设定是指在 CPU 模块出现停止出错时对 Q 系列对应的输出模块、I/O 混和模块、智能功能模块、中断模块的输出是清除还是保持的设定。

(2) 出错时的输出模式的设定

出错时的输出模式的设定在可编程控制器参数的 I/O 分配设定中进行。

1) 进行 I/O 分配设定

2) 选择 Detailed setting（详细设定）按钮

3) 在插槽的出错时输出模式的设定中选择 “清除”或者 “保持”。（默认为 “清除”）

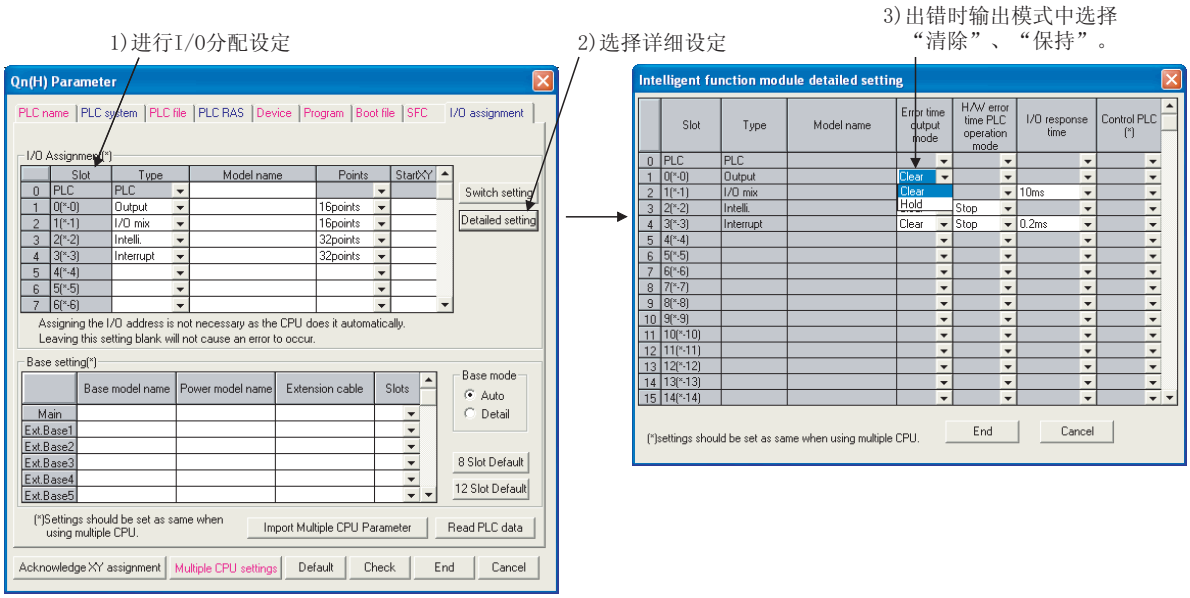


图 6.24 出错时输出模式设定的流程

(3) 注意事项

出错时输出模式的设定在下述情况下有效。

- 可编程控制器的电源处于 OFF→ON 时
- CPU 模块处于复位解除时

变更出错时输出模式的设定后，不进行上述操作时会出现 “PARAMETER ERROR（出错代码：3000）的错误。

6.9 硬件出错时 CPU 动作模式的设定

(1) 关于硬件出错时 CPU 动作模式的设定

硬件出错时 CPU 动作模式的设定是指在智能功能模块、中断模块中发生硬件出错时对 CPU 模块的运算是停止还是继续进行的设定。

(2) 硬件出错时 CPU 动作模式的设定

硬件出错时 CPU 动作模式的设定在可编程控制器参数的 I/O 分配设定中进行。

1) 进行 I/O 分配设定

2) 选择 Detailed setting（详细设定）按钮

3) 在插槽的硬件出错时 CPU 模块动作模式的设定中设定硬件出错时 CPU 模块的动作模式。（默认为 “停止”）

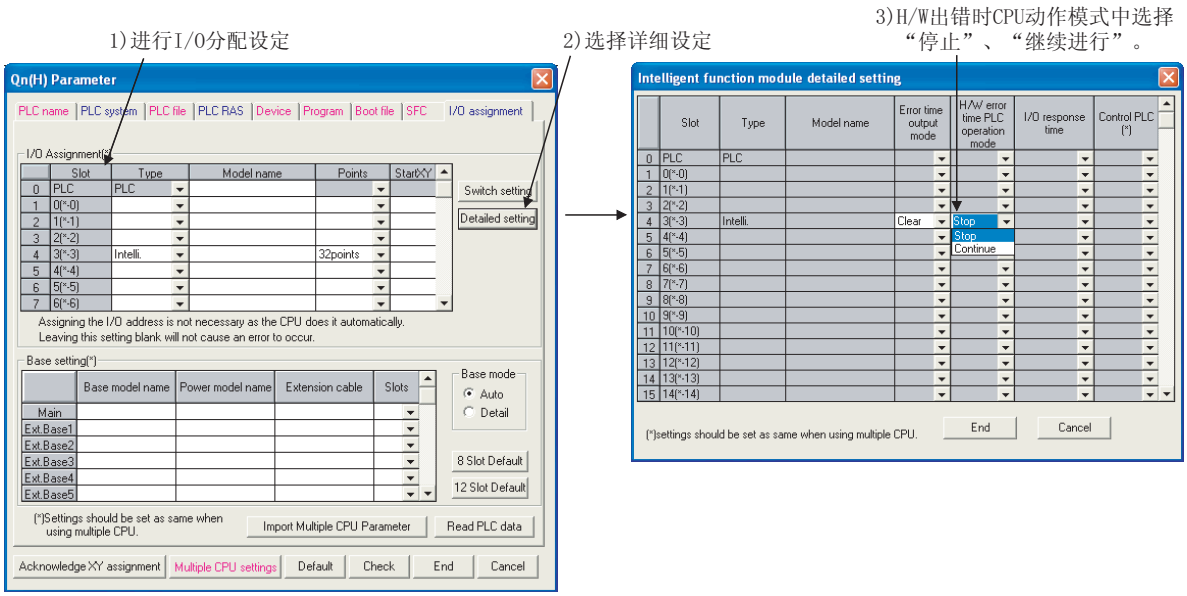


图 6.25 硬件出错时 CPU 动作模式设定的流程

(3) 注意事项

硬件出错时的 CPU 模块动作模式的设定在下述情况下有效。

- 可编程控制器的电源处于 OFF→接通时
- CPU 模块处于复位解除时

变更硬件出错时 CPU 动作模式的设定后，不进行上述操作时会出现 “PARAMETER ERROR（出错代码：3000）的错误。

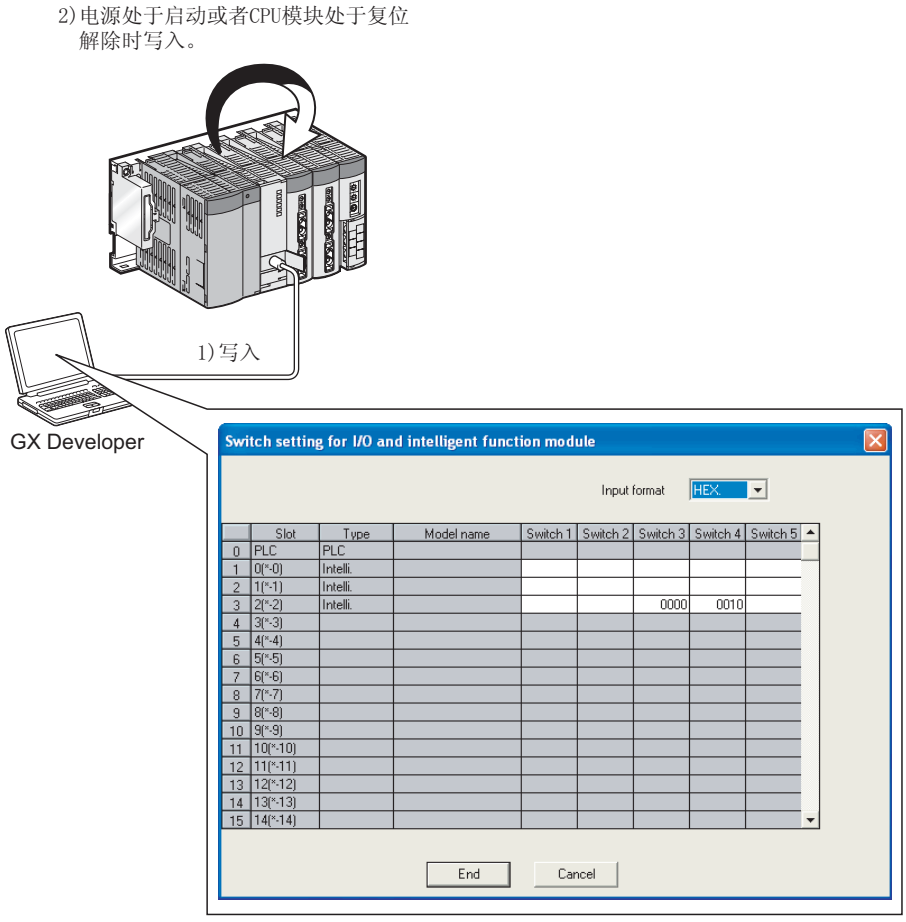
6.10 智能功能模块的开关设定

(1) 关于智能功能模块的开关设定

智能功能模块的开关设定是指通过 GX Developer 设定 Q 系列对应的智能功能模块、中断模块的开关内容的操作。

(2) 写入开关设定的时机

设定的开关设定在可编程控制器的电源处于启动或者 CPU 模块处于复位解除时，进行从 CPU 模块向各个智能功能模块、中断模块的写入。



智能功能模块的开关设定

图 6.26 将开关设定写入模块的流程

(3) 智能功能模块、中断模块的开关设定

智能功能模块、中断模块的开关设定在可编程控制器参数的 I/O 分配设定中进行。

1) 进行 I/O 分配设定

2) 选择 Switch setting （开关设定）按钮

3) 进行模块的开关设定

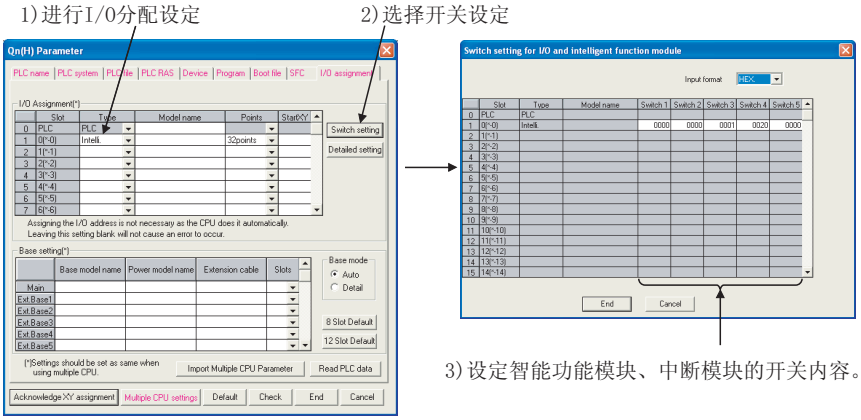


图 6.27 开关设定的流程

(4) 注意事项

基本 过程 冗余
注 6.11 注 6.11 注 6.11

通用
注 6.11

(a) 使用 AnS/A 系列对应的模块时*注 6.11*

请不要在 AnS/A 系列对应的特殊功能模块上进行开关设定。

对 AnS/A 系列对应的模块进行开关设定时，会出现 “SP.PARA.ERRORR” 的错误。

(b) 各个模块的开关设定内容

关于智能功能模块、中断模块的开关设定内容请参照所用的智能功能模块、中断模块的手册。

(c) 根据不同的 GX Developer 版本进行的不同设定

通过 GX Developer 版本 6(SW6D5C-GPPW) 以后的产品设定中断模块的开关设定时，将类别设定为 “中断” 后进行。

通过 GX Developer 版本 5(SW5D5C-GPPW) 以前的产品设定中断模块的开关设定时，将类别设定为 “智能” 后进行。

关于中断模块的开关设定的详细情况，请参照下述手册。

☞ 组件 I/O 模块用户手册

(d) 有效设定的时机

智能功能模块、中断模块的设定在下述情况下有效。

- 可编程控制器的电源 OFF→ON 时
- 复位解除时

基本 过程 冗余
注 6.11 注 6.11 注 6.11

通用
注 6.11

基本模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 中不能使用 Ans/A 系列对应的模块。

6.11 监视功能

(1) 关于监视功能

监视功能是指通过 GX Developer 读出 CPU 模块的程序、软元件、智能功能模块的状态的功能。×

表 6.16 监视功能一览表与是否可以使用

监视功能	各个 CPU 模块可否使用					参考
	基本模式 QCPU	高性能模式 QCPU	过程 CPU	冗余 CPU	通用型 QCPU	
梯形图监视	○	○	○	○	○	GX Developer 操作手册
软元件 / 缓冲存储器批量监视	○	○	○	○	○	
软元件登录监视	○	○	○	○	○	
软元件测试	○	○	○	○	○	
梯形图登录监视	○	○	○	○	○	
监视条件的设定	×	○	○	○	×	6.11.1 项
局部软元件的监视、测试	×*1	○	○	○	○	6.11.2 项
外部 I/O 的强制 ON/OFF	△*2	○	○	○	×	6.11.3 项

○：可以使用　△：一部分可以使用　×：不可以使用

*1: 在基本模式 QCPU 中，全局软元件、局部软元件没有区别（☞ 9.13.1 项）。
因此，没有必要通过 GX Developer 进行局部软元件监视的设定。
*2: 在基本模式 QCPU 中，只有通过软元件测试才可以进行强制 ON/OFF。

(2) 监视要求的处理时机与显示的数据

CPU 模块通过 END 处理进行来自 Developers 的监视要求的处理。
因此，GX Developer 中显示 CPU 模块 END 处理时的数据。

基本 通用 注 6.12 注 6.12

(3) 设定了监视条件的监视器*注 6.12*

在进行调试时，通过 GX Developer 对监视条件的设定，可以在指定条件下对 CPU 模块的运算状态进行监视。
另外，通过对监视停止条件的设定，可以在指定条件下保持监视状态。

基本 注 6.13

(4) 局部软元件的监视*注 6.13*

在执行多个程序并使用局部软元件的情况下，也可以对各个程序的局部软元件的数据进行监视。

基本 通用 注 6.12 注 6.12

在基本模式 QCPU、通用型 QCPU 中，不能进行设定了监视条件的监视。

基本 注 6.13

在基本模式 QCPU 中，由于不能执行多个程序，因此，全局软元件与局部软元件（☞ 9.13.1 项）没有区别。
因此，没有必要通过 GX Developer 进行局部软元件的监视设定。

基本
注 6.14

6.11.1 监视条件的设定

通用
注 6.14

监视条件的设定在指定条件下进行监视时设定。

(1) 梯形图监视时的监视执行条件的设定

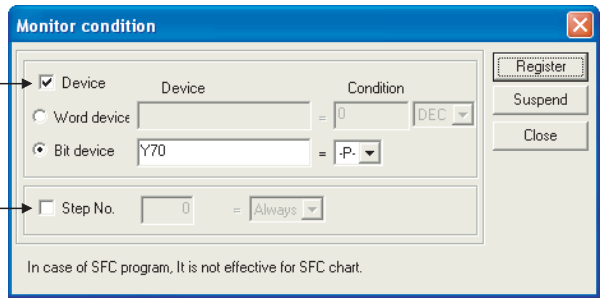
通过 GX Developer 设为监视模式。

通过 [在线]→[监视]→[监视条件的设定]，将显示出下面的窗口。

在启动 Y70 时进行监视的情况，请进行下述的设定。

选择对软元件的内容进行监视

选择对步No.进行监视



条件成立时进行监视(显示条件成立时的运算状态)

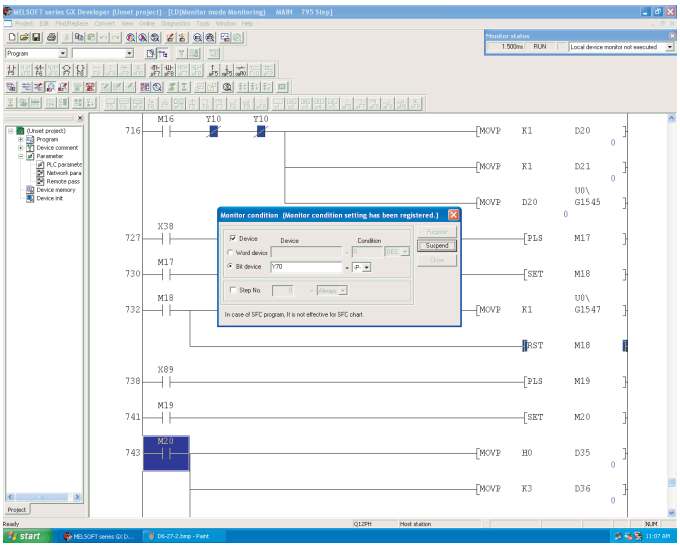


图 6.28 监视条件的设定画面

(a) 只指定步 No. 的情况

监视数据的收集时机为由指定步执行之前的状态变为指定状态时。

可以指定的执行条件如下所示。

- 指定步的运算由非执行状态变为执行状态时 :<-P->
- 指定步的运算由执行状态变为非执行状态时 :<-F->
- 指定步的运算只在执行中的任何时候 :<ON>
- 指定步的运算只在非执行中的任何时候 :<OFF>
- 与指定步的运算状态没有关系的任何时候 :< 任何时候 >

基本
注 6.14

通用
UD
注 6.14

在基本模式 QCPU、通用型 QCPU 中，不能进行监视条件设定。

要 点

1. 将 AND/OR 块中途的步作为监视条件进行指定时，监视数据的收集时机为：来自块内的 LD 指令的指定步执行之前的状态变为指定状态时。
因此，根据指定步的梯形图的不同，监视时机也会不同。
作为执行条件的步 2 “ON”的情况下进行监视时（步 No.(2)=(ON)）时，如下所示。

   • 通过 AND 指令连接步 2 时
   如图 6.29 中，X0 与 X1 同时 ON 时，监视的执行条件成立。

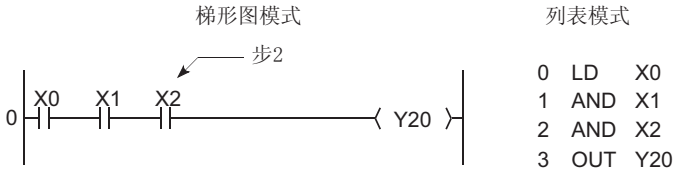


图 6.29 通过 AND 指令连接步 2 的情况

   • 步 2 在 AND/OR 块的中途被连接时
   在图 6.30 中 “X1”为 ON 状态下执行条件成立。(X0 的 ON/OFF 与执行条件的成立没有关系）

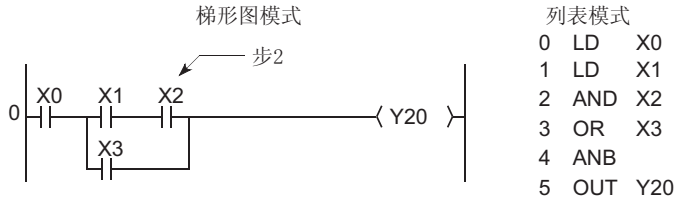


图 6.30 步 2 在 AND/OR 块的中途被连接时

   • 作为详细条件对步 No. 指定步 0 以外的起始梯形图块时，在由指令执行之前的状态变为指定状态时进行监视数据的收集。
   在下述梯形图中指定（步 No.(2)=(ON)）时，在 OUTY10 “ON”的状态下进行监视数据的收集。

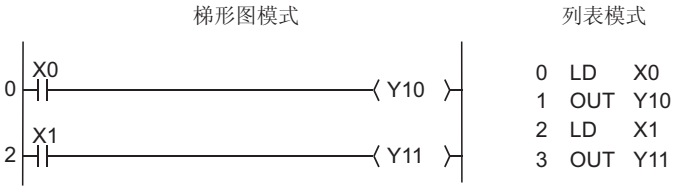


图 6.31 对步 No. 指定步 0 以外的起始梯形图块的情况

2. 指定步 No. 为 “0”的地方，请务必将条件设定为 “常时（任何时候）”。

(b) 只指定软元件的情况
可以指定字元件或者位软元件。

1) 指定字元件的情况
监视数据的收集时机在由指定字元件的当前值变为指定值时。
当前值以 10 进数或者 16 进数指定。

2) 指定位软元件的情况
监视数据的收集时机在由指定位字元件的执行状态进入指定状态时。
实行条件在上升时或者下降时可以进行指定。

(c) 指定步 No. 以及软元件的情况
监视数据的收集时机在由指定步的执行前的状态或者指定位软元件（字元件）的状态（当前值）进入指定状态时。

要 点

作为执行条件指定步 100 上升、D1=5（步 No.(100)=<-P->，字元件 (D1)=(K5)）的情况下，步 100 的上升与 D1=5 时监视的执行条件成立。

图 6.32 指定步 100 的上升与 D1=5 的情况

但是，GX Developer 的监视间隔周期要根据 GX Developer 的处理速度来决定。监视的执行条件在比 GX Developer 的监视间隔周期短的情况下，如果在 GX Developer 的监视间隔中监视的执行条件成立，则进行监视。

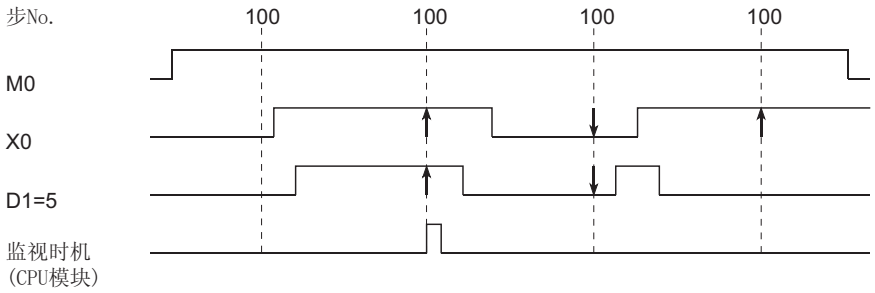


图 6.33 GX Developer 的监视时机

(2) 监视停止条件的设定

设定在 [在线] → [监视] → [监视停止条件设定] 中进行 (如图 6.34 的窗口所示)。
在 Y71 上升时停止监视的情况下如图 6.34 所示进行设定。

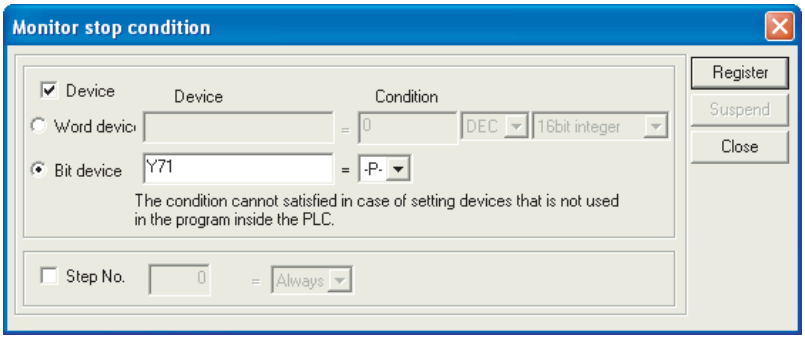


图 6.34 监视停止条件画面

(a) 指定 “步 No.” 的情况

监视停止的时机是在由监视条件指定的步 No. 的执行状态进入指定状态时。
可以指定的执行条件如下所示。

- 指定步的运算由非执行状态进入执行状态时 :< P >
- 指定步的运算由执行状态进入非执行状态时 :< F >
- 指定步的运算只在执行中的任何时候 :<ON>
- 指定步的运算只在非执行中的任何时候 :<OFF>
- 与指定步的运算状态没有关系的任何时候 :< 常时 >

在不指定 “步 No.” 的情况下，监视停止的时机是在 CPU 模块的 END 处理后。

(b) 指定了软元件的情况

可以指定字元件或者位软元件。

1) 指定字元件的情况

监视时机时机在由指定字元件的当前值变为指定值时。
当前值的指定可以选择 10 进数 / 16 进数与 16 位整数 /32 位整数 / 实数。

2) 指定位软元件的情况

监视停止时机在由指定位软元件的执行状态进入指定状态时。
实行条件在上升时或者下降时可以进行指定。

(3) 注意事项

(a) 进行监视的 CPU 模块的文件
设定监视条件并执行监视时，对显示在 GX Developer 上的文件进行监视。
执行 GX Developer 的 [在线] → [可编程控制器读出]，请保持进行监视的 CPU 模块的文件名与 GX Developer 上的文件名一致。

(b) 未设定文件寄存器的情况
在未设定文件寄存器的情况下，如果对文件寄存器进行监视将显示 0。(在通用型 QCPU 中，显示为 FFFFH。)

(c) 监视时的 CPU 模块与 GX Developer 的软元件分配
在监视时，请保持 CPU 模块与 GX Developer 的软元件分配的一致。

(d) 对智能功能模块的缓冲存储器进行监视的情况
在对智能功能模块的缓冲存储器进行监视的情况下，与执行 FROM/TO 指令时一样，扫描时间将被延长。

(e) 多人同时执行监视的情况
多人可以同时执行监视。
在多人同时执行监视的情况下，有以下的注意事项。

- 在设定程序内存的格式化或者可编程控制器参数的引导文件时，通过对其它站的每个监视文件增加 1k 步的系统区域，可以进行高速监视。
  其它站的监视文件最大可以设定为 15 站，但会占用相应的程序区域。
- 只有一个人可以设定监视的详细条件。

(f) 可以设定监视的详细条件的状态
监视的详细条件的设定只可以在梯形图监视中设定。

(g) 作为条件指定同一个软元件的情况
作为监视条件与监视停止条件指定同一个软元件时，请指定 “ON”或者是 “OFF”。

(h) 在监视条件中指定步 No. 的情况
在监视条件中指定步 No. 时，如下所示，不执行指定步的指令时监视条件不成立。

- 通过 CJ 指令、SCJ 指令、JMP 指令跳过设定步时
- 在设定步的 END 指令中，在程序中存在 FEND 指令，END 指令不执行时。

(i) 监视条件登录过程中
在监视登录过程中，请不要对 CPU 模块进行复位。

基本 注 6.15

6.11.2 局部软元件的监视、测试

通用 UD 注 6.16

可编程控制器参数的软元件设定中，可以对设定为局部软元件的软元件通过 GX Developer 进行监视以及测试。*注 6.16* *注 6.16*
根据本功能，在通过 GX Developer 对监视的程序中使用的软元件（☞9.13.1 项）的内容进行确认的同时，可以进行调试。

(1) 局部软元件的监视
将局部软元件设定为 D0-99，执行 CPU 模块中程序名为 “A”、“B”、“C”三个程序时的动作如表 6.17 所示。
（程序的执行顺序按照 A→B→C→(END 处理)→A→B... 进行。）

表 6.17 执行三个程序时显示的数据

设定	监视软元件	
	D0（局部软元件）	D100（全局软元件）
未设定局部软元件时	监视程序 C 的 D0。	监视执行程序 C 后的 D100。
设定局部软元件时	监视显示程序的 D0。	监视显示的程序执行后的 D100。

基本 注 6.15

在基本模式 QCPU 中，由于不能执行多个程序的原因，因此，全局软元件与局部软元件没有区别。（☞9.13.1 项）
因此，没有必要通过 GX Developer 进行局部软元件的监视设定。

通用 UD 注 6.16

在通用型 QCPU 中，不能进行局部软元件的测试。

例如，进行局部软元件的监视设定，显示并监视程序名为 B 的程序时，可以对程序 B 的局部软元件进行监视。

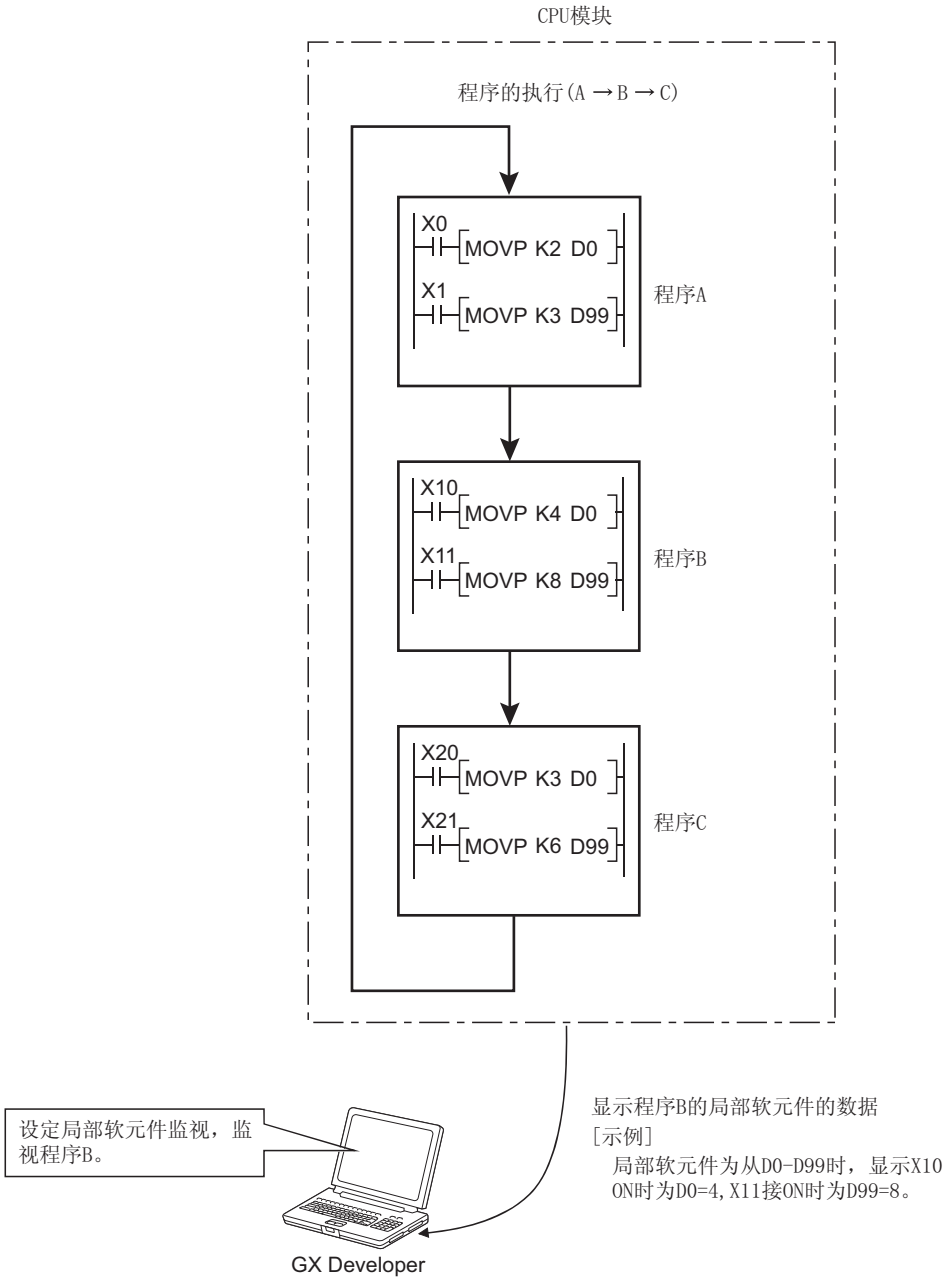


图 6.35 局部软元件的监视示例

(2) 局部软元件的监视顺序

局部软元件的监视按照下述的顺序进行。

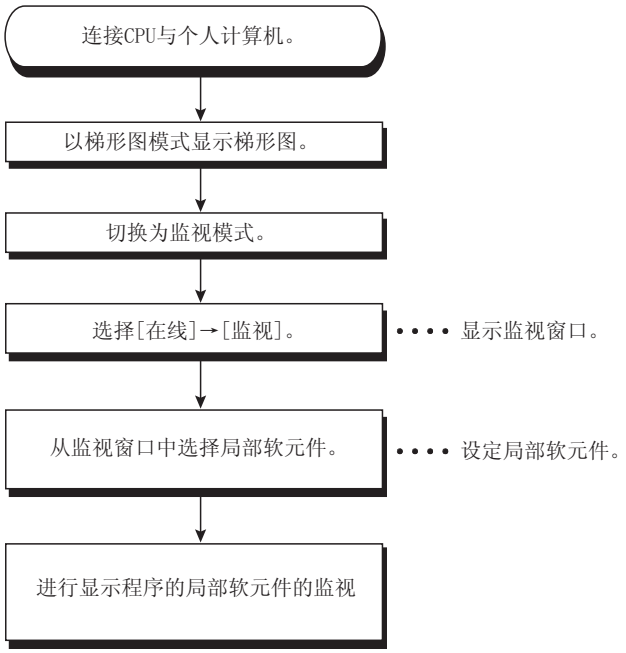


图 6.36 局部软元件的监视顺序流程

(3) 注意事项

(a) 通过一个 GX Developer 可以监视、测试的局部软元件

通过一个 GX Developer 只可以监视、测试一个程序的局部软元件。
不能通过一个 GX Developer 同时监视、测试多个程序的局部软元件。

(b) 可以监视、测试的程序数

通过连接在 CPU 模块的 RS-232、串行口通讯模块等上的多个 GX Developer 可以最多同时监视测试 16 个程序的局部软元件。

(c) 待机类型程序的局部软元件的监视

进行待机类型程序的局部软元件监视时，由于进行局部软元件数据的退避与复归的原因，扫描时间将延长。(☞9.13.1 项)

(d) 恒定周期执行类型程序的局部软元件监视

不能对恒定周期执行类型程序的局部软元件数据进行监视。

高性能
注 6.17

通用
注 6.18

6.11.3 外部 I/O 的强制 ON/OFF

通过 GX Developer 的强制 ON/OFF 操作，可以对外部 I/O 进行强制 ON/OFF。
通过 GX Developer 的操作解除 ON/OFF 登录的信息。

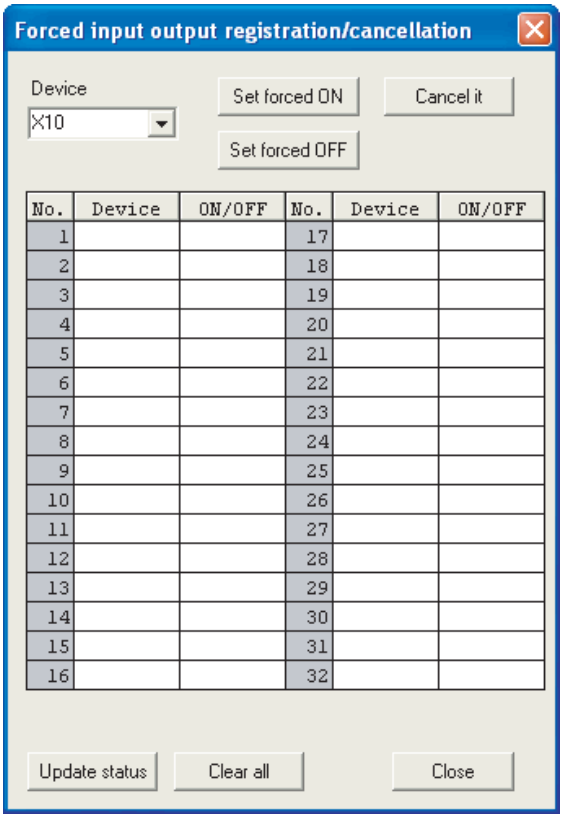


图 6.37 强制 I/O 登录 / 解除画面

高性能
注 6.17

使用高性能模式 QCPU 中的外部 I/O 的强制 ON/OFF 时，请确认 CPU 模块以及 GX Developer 的版本。(☞ 附录 4.2)

通用
注 6.18

在通用型 QCPU 中，不能进行外部 I/O 的强制 ON/OFF。

(1) 强制 ON/OFF 时的动作

在强制 ON/OFF 中可以进行强制 ON（强制 ON 登录）、强制 OFF（强制 OFF 登录）、强制 ON/OFF 的解除（登录解除）。注 6.19

基本 注 6.19

强制 ON、强制 OFF、强制 ON/OFF 的解除操作时的动作如表 6.18 所示。

表 6.18 进行强制 ON/OFF 的解除操作时的动作

操作	输入 (X) 的动作	输出 (Y) 的动作
解除（未操作）时	通过外部输入进行顺控程序的运算。	将顺控程序的运算结果输出到外部。注 6.20
强制 ON 时	在强制 ON 状态下，进行顺控程序的运算。	与顺控程序的运算结果无关，将“ON”输出到外部。注 6.20
强制 OFF 时	在强制 OFF 状态下，进行顺控程序的运算。	与顺控程序的运算结果无关，将“OFF”输出到外部。注 6.20

冗余 注 6.20

基本 注 6.19

在基本模式 QCPU 中，不能使用强制 I/O 登录 / 解除（☞本项 (3)）进行强制 ON 登录和强制 OFF 登录。
在基本模式 QCPU 中，只能通过 GX Developer 的软元件测试进行强制 ON/OFF。

冗余 注 6.20

在备份模式中，不能进行待机系统输出 (Y) 的强制 ON/OFF。

进行强制 ON/OFF 时的动作如图 6.38 所示。

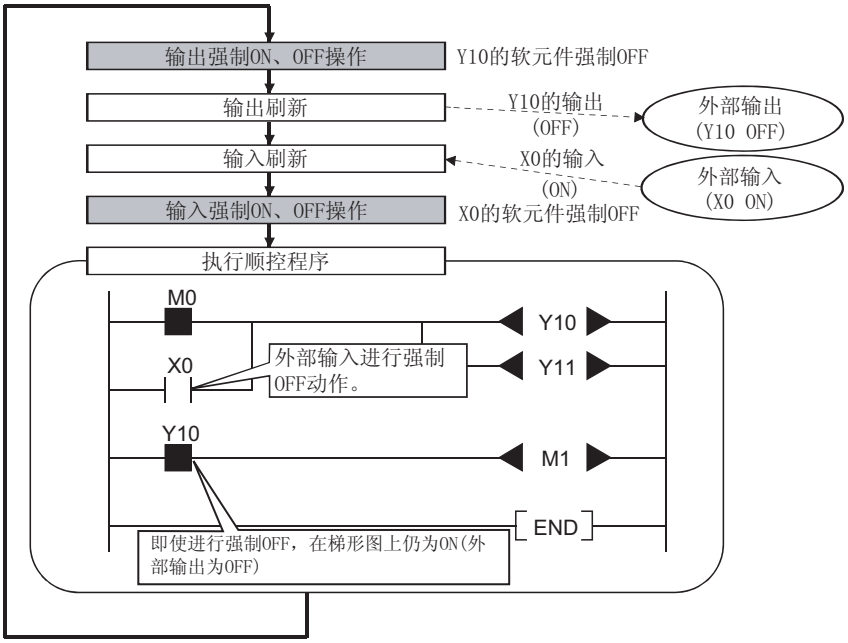


图 6.38 进行强制 ON/OFF 时的动作

要 点

1. 使用本功能需使用 GX Developer 版本 6 以后的产品。
2. 在冗余 CPU 中发生系统切换时，新控制系统将以系统切换发生前登录到控制系统上的强制 ON/OFF 登录内容，继续进行强制 ON/OFF。

(2) 规格说明

(a) 可以进行强制 ON/OFF 的 CPU 模块的状态
可以不管 CPU 模块的 RUN/STOP 状态进行强制 ON/OFF。
但是，在发生停止出错时只可以进行输入的强制 ON/OFF。
输出只对软元件 Y 进行。

(b) 可以登录的软元件
可以登录的软元件为 CPU 模块的 I/O 软元件点数。
(☞第 2 章)注 6.21

基本 注 6.21

(c) 变为强制 ON/OFF 对象的 I/O
变为强制 ON/OFF 对象的 I/O 如下所示。

- 安装在基板上模块的输入 (X)、输出 (Y)
- MELSECNET/G 注 6.22, 注 6.23 或者 MELSECNET/H 模块的 LX/LY 的刷新目标的 CPU 模块的 I/O(X/Y)。
- CC-Link 的 RX/RY 的刷新对象的 CPU 模块的 I/O(X/Y)。

高性能 注 6.22

对上述刷新范围以外的软元件（示例：空插槽）进行强制 ON/OFF 时，只对 CPU 模块的软元件内存 ON/OFF，不进行向外部的输出。

基本 注 6.23　过程 注 6.23　冗余 注 6.23

基本 注 6.21
在基本模式 QCPU 中，不能使用 [Online] → [Debug] → [Forced input output registration/cancellation](☞本项 (3)) 进行强制 ON 登录以及强制 OFF 登录。
在基本模式 QCPU 中，只能通过 GX Developer 的软元件测试进行强制 ON/OFF。

高性能 注 6.22
在高性能模式 QCPU 中使用 MELSECNET/G 时，请确认 CPU 模块以及 GX Developer 的版本。
(☞附录 4.2)

基本 注 6.23　过程 注 6.23　冗余 注 6.23
在基本模式 QCPU、过程 CPU、冗余 CPU 中，不能使用 MELSECNET/G。

基本
注 6.24

(d) ON/OFF 登录的信息的解除注 6.24

ON/OFF 登录的信息的解除可以通过 GX Developer 的操作进行。
如果解除 ON/OFF 登录的信息，强制 ON/OFF 的软元件如下所示。

表 6.19 解除 ON/OFF 登录信息后的软元件的状态

强制 ON/OFF 软元件		强制 ON/OFF 软元件	不通过顺控程序进行 ON/OFF
输入	来自安装在基板上的模块的输入	变为从模块读取的 ON/OFF 状态。	
	MELSECNET/G*1*2 或者 MELSECNET/H 模块的 LX 的刷新对象的 CPU 模块的输入	变为从 MELSECNET/H 刷新过来的 ON/OFF 状态。	
	CC-Link 的 RX 的刷新对象的 CPU 模块的输入	变为从 CC-Link 刷新过来的 ON/OFF 状态。	
	上述以外的输入（刷新范围以外）	保持被强制 ON/OFF 的状态。	
输出	来自安装在基板上的模块的输出	输出顺控程序的运算结果。	输出 OFF。
	MELSECNET/G*1*2 或者 MELSECNET/H 模块的 LX 的刷新对象的 CPU 模块的输出	输出顺控程序的运算结果。	输出 OFF。
	CC-Link 的 RX 的刷新对象的 CPU 模块的输出	输出顺控程序的运算结果。	输出 OFF。
	上述以外的输出（刷新范围以外）	CPU 模块的输出变为顺控程序的运算结果（不输出到外部）	CPU 模块的输出变为 OFF 状态不变。（不输出到外部）

*1: 在高性能模式 QCPU 中使用 MELSECNET/G 时，请确认 CPU 模块以及 GX Developer 的版本。
(☞ 附录 4.2)
*2: 在基本模式 QCPU、过程 CPU、冗余 CPU 中，不能使用 MELSECNET/G。

强制 ON/OFF 的设定，可以通过下述的操作清除。

- 电源由 OFF→ON
- 通过 CPU 模块的 RESET/L.CLR 开关进行复位
- 通过远程复位操作进行复位

基本
注 6.24

在基本模式 QCPU 中，不能使用 [Online] → [Debug] → [Forced input output registration/cancellation](☞ 本项 (3)) 进行强制 ON 登录以及强制 OFF 登录。
在基本模式 QCPU 中，只能通过 GX Developer 的软元件测试进行强制 ON/OFF。

(e) 外部 I/O 的强制 ON/OFF 时机

外部 I/O 的强制 ON/OFF 时机如表 6.20 所示。

表 6.20 强制 ON/OFF 的时机

刷新区域	输入	输出
安装在基板上的模块的I/O	• END 处理（输入刷新）时 • 执行直接存取输入 (DX) 所用的指令时 (LD、LDI、AND、ANI、OR、ORI、LDP、LDF、ANDP、ANDF、ORP、ORF)	• END 处理（输出刷新）时 • 执行直接存取输出 (DY) 所用的指令时 (OUT、SET、DELTA、RST、PLS、PLF、FF、MC)
MELSECNET/G*1*2 或者 MELSECNET/H 模块的 LX、LY 的刷新对象的 CPU 模块的 I/O	• END 处理 (MELSECNET/H 刷新）时 • 执行 COM 指令时 • 执行 ZCOM 指令时	
CC-Link 的 RX、RY 刷新对象的 CPU 模块的 I/O	• END 处理（自动刷新）时 • 执行 COM 指令时 • 执行 ZCOM 指令时	

*1: 在高性能模式 QCPU 中使用 MELSECNET/G 时，请确认 CPU 模块以及 GX Developer 的版本。
(☞ 附录 4.2)

*2: 在基本模式 QCPU、过程 CPU、冗余 CPU 中，不能使用 MELSECNET/G。

基本 注 6.25

(f) 可以登录的软元件数*注 6.25*

可以登录的软元件数加上强制 ON/OFF 最大为 32 个。

(g) 顺控程序中使用输出 Y 的触点的情况

顺控程序中使用输出 Y 的触点时，顺控程序的动作将被优先。

基本 注 6.25

(h) 强制 ON/OFF 以及解除状况的确认*注 6.25*

强制 ON/OFF、解除（包括未设定）的状况通过 GX Developer 可以得到确认。

另外，即使只登录一个软元件时，也可以通过 MODE LED 进行确认。(MODE LED 闪烁）*注 6.26*

冗余 注 6.26

基本 注 6.25

(i) 通过多个 GX Developer 进行的强制 ON/OFF*注 6.25*

通过连接在网络上的多个 GX Developer 可以对同一个 CPU 模块进行外部 I/O 的强制 ON/OFF 登录。

但是，通过多个 GX Developer 对同一个软元件进行强制 ON/OFF 登录的情况下，其后将进入所登录的 ON/OFF 状态。

初次执行强制 ON/OFF 的 GX Developer 有显示出与 CPU 模块的 ON/OFF 信息不同的信息的情况。

通过多个 GX Developer 进行强制 ON/OFF 登录的情况下，请在通过 “读出登录状况”开关读出最新的数据后执行强制 ON/OFF。

基本 注 6.25

在基本模式 QCPU 中，不能使用 [Online] → [Debug] → [Forced input output registration/cancellation](☞ 本项 (3)) 进行强制 ON 登录及强制 OFF 登录。

在基本模式 QCPU 中，只能通过 GX Developer 的软元件测试进行强制 ON/OFF。

冗余 注 6.26

在冗余 CPU 中，两系统冗余 CPU 的 MODE LED 闪烁。

基本
注 6.27

(3) 操作顺序*注 6.27*

操作顺序如下所示。

- 对指定软元件进行强制 ON/OFF 登录。
  [在线] → [调试] → [强制 I/O 登录 / 解除]
- 在 [强制 I/O 登录 / 解除] 的设定画面中设定软元件后，通过选择 [强制 ON 登录]/[强制 OFF 登录]，可以对指定的软元件进行强制 ON/ 强制 OFF。

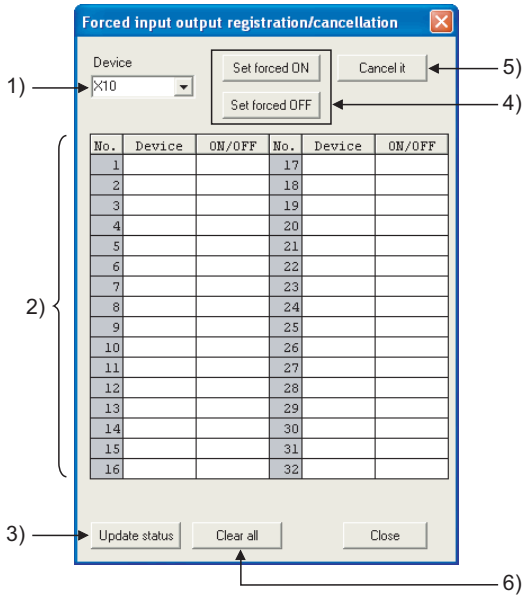


图 6.39 强制 I/O 登录画面

表 6.21 强制 I/O 登录画面的项目

No.	设定项目名称	功能说明
1)	软元件	输入强制 ON/OFF 或者解除强制 ON/OFF 的 I/O 地址号。
2)	登录状况显示区域	显示登录的强制 I/O 登录状况。
3)	读出登录状况	显示从 CPU 模块读出的登录状况。
4)	强制 ON/OFF 登录	登录设定的软元件的强制 ON/OFF。
5)	登录解除	解除登录的软元件的强制 ON/OFF。
6)	登录批量解除	解除所有的强制 I/O 登录。

基本
注 6.27

在基本模式 QCPU 中，不能使用强制 I/O 登录 / 解除进行强制 ON 登录、强制 OFF 登录。
在基本模式 QCPU 中，只能通过 GX Developer 的软元件测试进行强制 ON/OFF。

(4) 使用冗余 CPU 的注意事项

(a) 关于进行登录 / 解除的系统
在使用冗余 CPU 时进行强制 I/O 登录 / 解除的情况下，请对控制系统进行操作。
（对控制系统以及待机系统不能分别地进行登录 / 解除）
在系统切换后，请对新控制系统（通过系统切换从待机系统变为控制系统的系统）进行登录 / 解除。
在对待机系统进行登录 / 解除时，会出现错误（出错代码：4240H）。

(b) 关于独立模式时对电源进行由 OFF→ON 或者复位
独立模式时如果对控制系统的电源进行 OFF 或者复位，强制 I/O 的登录将被解除。
登录解除后的动作与表 6.18 所示的 “解除（未操作）时”动作一致。
但是，MELSECNET/H 远程 I/O 站上的模块的输出如下所示。

1) 电源处于 OFF 或者复位时
保持控制系统电源的 OFF 或复位时的输出状态。

2) 电源由 OFF→ON 或者进行复位解除时
输出顺控程序的运算结果。
（与强制 I/O 输出解除时一样）

6.12 CPU 模块在运行中进行的程序的写入

CPU 模块在运行中进行的程序的写入有如表 6.22 所示的种类。

表 6.22 运行中写入的种类

CPU 模块	运行中写入的种类
基本模式 QCPU	• 梯形图模式中的运行中写入（☞ 6.12.1 项） • 使用指针进行的运行中写入（☞ 6.15.2 项）
高性能模式 QCPU 过程 CPU 冗余 CPU 通用型 QCPU	• 梯形图模式中的运行中写入（☞ 6.12.1 项） • 文件的运行中写入（☞ 6.12.2 项） • 使用指针进行的运行中写入（☞ 6.15.2 项）

要 点

关于使用冗余 CPU 时的运行中写入（向控制系统与待机系统的写入顺序、执行运行中写入时可否执行热备传送等），请参照下述手册。

☞ QnPRHCPU 用户手册（冗余系统篇）

6.12.1 梯形图模式中的运行中写入

(1) 关于梯形图模式中的运行中写入

梯形图模式中的运行中写入是指向运行中的 CPU 模块进行程序的写入的功能。

如果使用梯形图模式中的运行中写入，不停止 CPU 模块的程序运算也可以变更程序。

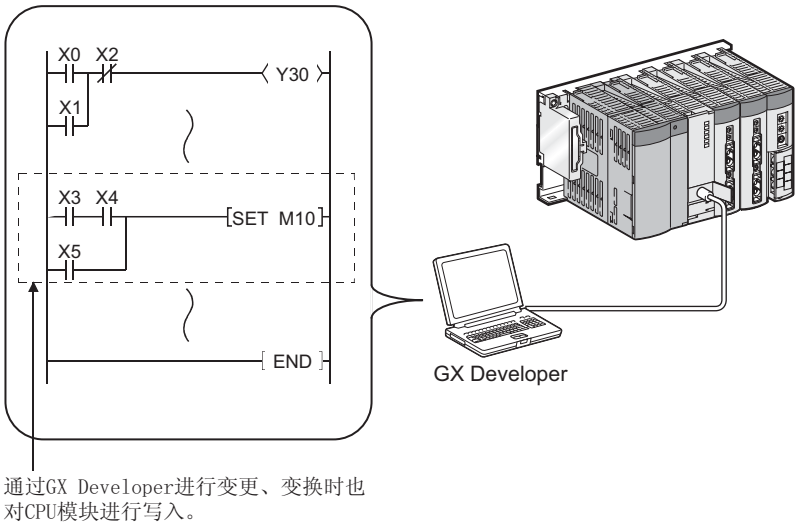


图 6.40 梯形图模式的运行中写入的概要

另外，可以通过连接在网络上的其它站上的 GX Developer 进行程序的运行中写入。

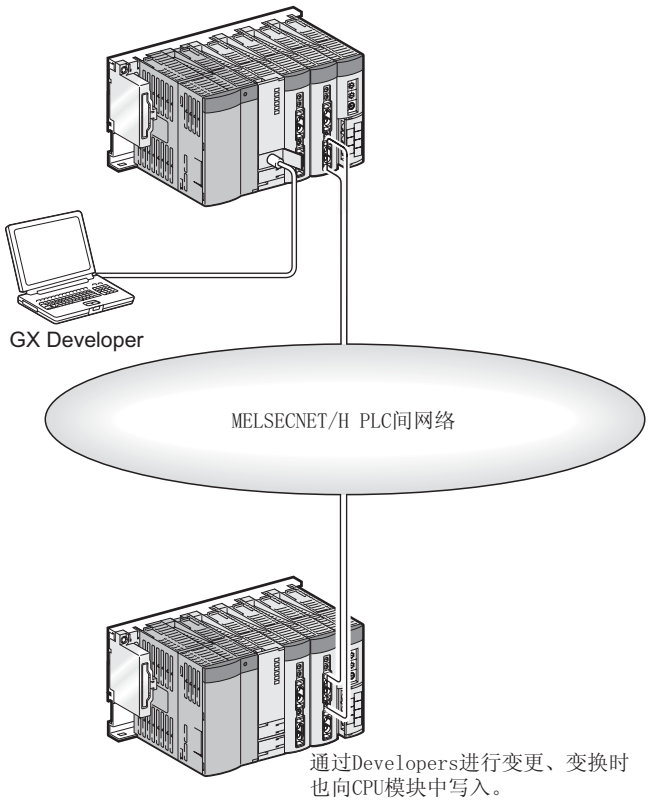


图 6.41 经由网络的运行中写入的概要

(2) 可进行运行中写入的存储器

(a) 基本模式 QCPU、高性能模式 QCPU、过程 CPU、冗余 CPU 的情况
可进行运行中写入的存储器为程序内存。

(b) 通用型 QCPU 的情况
可进行运行中写入的存储器为程序高速缓冲存储器（程序内存）。

(3) 一次运行中写入时可写入的步数
一次运行中写入时可写入的步数最多为 512 步。

基本 冗余 通用
注 6.28 注 6.28 注 6.28

(4) 低速执行类型程序的运行中写入的执行时机 注 6.28

低速执行类型程序的运行中写入，是在所有的低速执行类型程序结束后的下一个扫描的 END 处理时进行。

此外，在低速执行类型程序的运行中写入过程中，中断低速执行类型程序的执行。

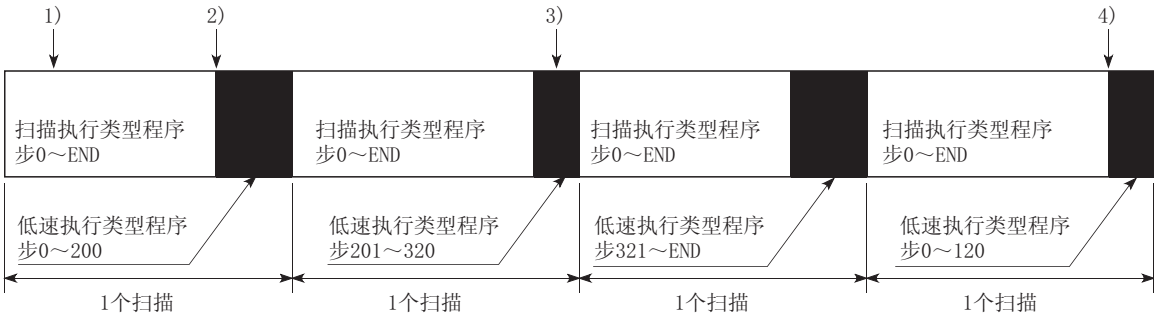


图 6.42 执行低速执行类型程序时的运行中写入处理

1)：扫描执行类型程序的运行中写入指令

2)：扫描执行类型程序的运行中写入的执行

3)：低速执行类型程序的运行中写入指令

4)：低速执行类型程序的运行中写入的执行

基本 冗余 通用
注 6.29 注 6.29 注 6.29

(5) 执行 PLOADP、PUNLOADP、PSWAPP 指令过程中的运行中写入 注 6.29

如果在 PLOADP、PUNLOADP、PSWAPP 指令的执行过程中进行运行中写入，那么需等到执行中的指令的处理结束后才进行运行中写入的处理。

另外，在执行运行中写入过程中，如果执行 PLOADP、PUNLOADP、PSWAPP 指令，那么需等到运行中写入的处理结束后才会开始执行 PLOADP、PUNLOADP、PSWAPP 的指令。

基本 冗余 通用
注 6.28 注 6.28 注 6.28

在基本模式 QCPU、冗余 CPU、通用型 QCPU 中，由于不能使用低速执行类型程序，因此无需理会低速执行类型程序的运行中写入。

基本 冗余 通用
注 6.29 注 6.29 注 6.29

在基本模式 QCPU、冗余 CPU、通用型 QCPU 中，不能通过 PLOADP、PUNLOADP、PSWAPP 指令进行运行中的程序文件的变更。

(6) 运行中写入时的 “运行中写入预留容量（单位：步）”的变更

对在运行中写入时变更 “运行中写入预留容量（单位：步）”的注意事项进行说明。

(a) 关于运行中写入预留容量（单位：步）

在程序文件中，为保证程序文件的容量与变化的运行中写入相对应，留有用于运行中写入的预留容量（单位：步）。

程序文件的容量变为创建的文件的容量上加上 “运行中写入预留容量（单位：步）”的值。

(b) 程序文件的容量比预留容量（ 单位：步）有所增加的情况

在运行中写入之际，在程序文件的容量超出了预留的容量（包括运行中写入预留容量（ 单位：步）的容量） 的情况下，可以在运行中写入过程中对运行中写入预留容量（单位：步）进行再设定。

因此，在用户存储区域有空闲区域的情况下，可以进行运行中写入。

(c) 进行运行中写入预留容量（单位：步）的再设定时的扫描时间

进行运行中写入预留容量（单位：步）的再设定时，扫描时间将延长。

关于扫描时间的延长，请参阅 10.1.3 项。

基本 高性能 过程
注 6.30 注 6.30 注 6.30

冗余
注 6.30

(7) 运行中写入时、T/C 设定值变更时以及从程序高速缓冲存储器传送至程序内存时的禁止操作 *注 6.30*

关于运行中写入时、T/C 设定值变更时以及从程序高速缓冲存储器传送至程序内存时的禁止操作，请参阅 6.12.3 项 (2)。

(8) 在运行中写入时不能正常动作的指令

关于运行中写入时不能正常动作的指令，请参阅 6.12.3 项 (3)。

基本 高性能 过程
注 6.30 注 6.30 注 6.30

冗余
注 6.30

在基本模式 QCPU、高性能模式 QCPU、过程 CPU、冗余 CPU 中，由于不能使用程序高速缓冲存储器，因此无需理会从程序高速缓冲存储器至程序内存的传送的有关内容。

基本
注 6.31

6.12.2 文件的运行中写入

(1) 关于文件的运行中写入功能

是指 CPU 模块在运行过程中通过 GX Developer 的在线操作，向 CPU 模块内批量写入如表 6.23 所示的文件的功能。

表 6.23 可以进行文件运行中写入的文件

存储器名称	CPU 模块内部			存储卡 (RAM)	存储卡 (ROM)	
	程序内存	标准 RAM	标准 ROM	SRAM 卡	快闪卡	ATA 卡
参数	×	×	×	×	×	×
智能功能模块参数	×	×	×	×	×	×
程序	○	×	△ *2	○	×	○
软元件注释	○	×	△ *1*2	△ *1	×	△ *1
软元件初始值	×	×	×	×	×	×
文件寄存器	×	△ *1	×	△ *1	×	×
局部软元件	×	×	×	×	×	×
采样追踪文件	×	○	×	○	×	×
故障历史记录数据	×	×	×	×	×	×
可编程控制器用户数据	×	×	△ *2	×	×	○

○：可以写入； ×：不可写入； △：有限制

*1：在不处于通过顺控程序进行存取的状态时可以写入。
*2：在通用型 QCPU 中，可以写入。

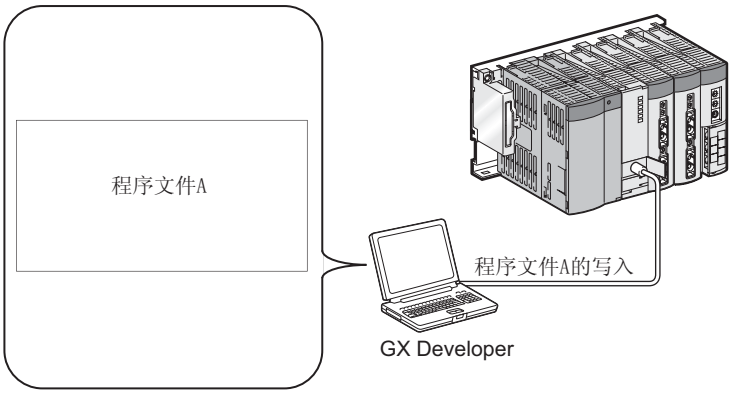


图 6.43 文件的运行中写入的概要

基本
注 6.31

在基本模式 QCPU 中，不能进行文件的运行中写入。

(2) 文件的运行中写入的执行可否

(a) 基本模式 QCPU、高性能模式 QCPU、过程 CPU、冗余 CPU 的情况

文件的运行中写入的执行可否如表 6.24 所示。

表 6.24 文件的运行中写入的执行可否

写入程序文件以上的空闲区域		文件的运行中写入
程序内存	存储卡	
有	无 / 有	可以执行
无	有	可以执行 *1
无	无	不可执行

*1: 通过 GX Developer 的选项菜单在运行中写入时的指令动作设定中设置了 “不执行下降沿指令” 时，如果在写入的程序文件内包含有下降沿指令，将不能进行运行中写入。
但是，写入的程序为 SFC 程序时，即使在写入的程序文件内包含有下降沿指令，也可以进行文件的运行中写入。
（关于通过 GX Developer 的选项菜单在运行中写入时的指令动作设定中设置了 “不执行下降沿指令” 时的动作，请参阅 6.12.3 项。）

(b) 通用 QCPU 的情况

与是否有写入的程序文件以上的空闲区域无关，可以进行文件的运行中写入。

(3) 扫描时间的延长时间

如果进行运行中的文件的写入，扫描时间将延长。
关于扫描时间的延长时间，请参阅 10.1.3 项。

(4) 通过顺控程序的指令进行了文件存取时

在执行文件的运行中写入的过程中，不能通过程序指令进行文件存取，应加以注意。
在文件的运行中写入的执行过程中文件存取指令将变为非执行。

(5) 不能从多处进行文件的运行中写入

请不要从多处同时进行文件的运行中写入。
同时从多处进行文件的运行中写入时，程序文件的内容有消失的可能性。

(6) SFC 程序的文件的运行中写入 注 6.32 注 6.33

对于 SFC 程序，如果执行文件的运行中写入，执行后将进行初始化启动。

(7) 在运行中写入时动作不正常的指令运行中写入时、T/C 设定值变更时以及从程序高速缓冲存储器传送至程序内存时的禁止操作 注 6.34

关于运行中写入时、T/C 设定值变更时以及从程序高速缓冲存储器传送至程序内存时的禁止操作，请参阅 6.12.3 项 (2)。

(8) 文件的运行中写入时不能正常动作的指令

关于文件的运行中写入时不能正常动作的指令，请参阅 6.12.3 项 (3)。

在执行高性能模式 QCPU、过程 CPU、SFC 程序的文件的运行中写入时，请确认 CPU 模块以及 GX Developer 的版本。(☞ 附录 4.2、附录 4.3)

在通用型 QCPU 中，不能进行 SFC 程序的文件的运行中写入。

在基本模式 QCPU、高性能模式 QCPU、过程 CPU、冗余 CPU 中，由于不能使用程序高速缓冲存储器，因此无需理会从程序高速缓冲存储器至程序内存的传送的有关内容。

6.12.3 运行中写入时的注意事项

运行中写入时的注意事项如下所示。

(1) 引导运行中的运行中写入

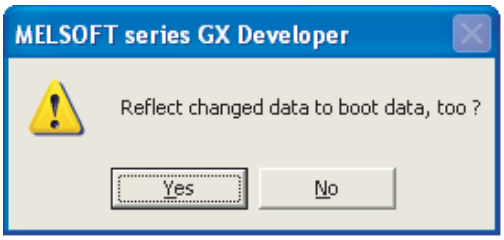
在引导运行过程中执行了运行中写入时，引导源程序的状态根据所使用的 CPU 模块及引导源存储器的不同而有所不同。

在引导运行的过程中执行运行中写入时的引导源程序的状态如表 6.25 所示。

表 6.25 执行了运行中写入时的引导源程序的状态

引导源		引导源的程序状态			
		基本模式 QCPU	• 高性能模式 QCPU • 过程 CPU	冗余 CPU	通用型 QCPU
存储卡	SRAM 卡 ATA 卡	----	更改 *1	更改 *1	不更改 *2
	快闪卡	----	不更改 *2	更改 *1*3	不更改 *2
标准 ROM		不更改 *2	不更改 *2	更改 *1*3	----

*1: 在显示如下图所示的信息框时如果选择 “Yes”，将进行引导源的程序的更改。
此时，至运行中写入的结束需要耗费一定的时间。



如果选择了 “No”，将不更改引导源程序。
在运行中写入后，进行可编程控制器的电源 OFF 或者 CPU 模块的复位之前，应将程序内存的内容写入到存储卡 注 6.35、标准 ROM 注 6.36 中。

基本 注 6.35

*2: 在运行中写入后，进行可编程控制器的电源 OFF 或者 CPU 模块的复位之前，应将程序内存的内容写入到存储卡 注 6.35、标准 ROM 注 6.36 中。

*3: 引导源存储器为快闪卡、标准 ROM 时，引导源的文件将全部被删除，将被程序内存上的文件所覆盖。对于引导运行时的必要文件，应在参数的引导文件设定中设置为传送到程序内存中。

通用 注 6.36

基本 注 6.35

在基本模式 QCPU 中，不能使用存储卡。

基本 注 6.36

在通用型 QCPU 中，不能进行从标准 ROM 至程序内存的引导。（☞ 5.2.3 项）

(2) 运行中写入时、T/C 设定值变更时以及从程序高速缓冲存储器传送至程序内存时的禁止操作 *注 6.37*

运行中写入时、T/C 设定值变更时以及从程序高速缓冲存储器传送至程序内存时，不要执行以下操作。

基本 注 6.37　高性能 注 6.37　过程 注 6.37

冗余 注 6.37

(a) 电源 OFF、复位操作

在运行中写入时、T/C 设定值变更时以及从程序高速缓冲存储器传送至程序内存时，如果执行电源 OFF 或者复位操作，操作将无法正常结束。

进行了电源 OFF 或者复位操作时，应重新进行可编程控制器写入。

(b) 通过 GX Developer 的操作

在运行中写入时、T/C 设定值变更时以及从程序高速缓冲存储器传送至程序内存时，不能进行如下操作，应在操作结束后再执行。

如果执行了以下操作，GX Developer 中将显示操作。

- 运行中写入（梯形图模式、文件的运行中写入、功能块）
- T/C 设定值变更
- 至程序内存的传送
- 可编程控制器写入（快闪卡）

基本 注 6.37　高性能 注 6.37　过程 注 6.37

冗余 注 6.37

在基本模式 QCPU、高性能模式 QCPU、过程 CPU、冗余 CPU 中，由于不能使用程序高速缓冲存储器，因此无需理会从程序高速缓冲存储器至程序内存的传送的有关内容。

(3) 运行中写入时动作不正常的指令

进行运行中写入时，动作不正常的指令有以下 3 种：

- 下降沿指令
- 上升沿指令
- SCJ 指令

(a) 下降沿指令

在运行中写入的写入范围内有下降沿指令时，运行中写入结束时即使下降沿指令的执行条件 (ON → OFF) 不成立也将执行下降沿指令。

关于运行中写入时防止执行下降沿指令的方法，请参阅要点。

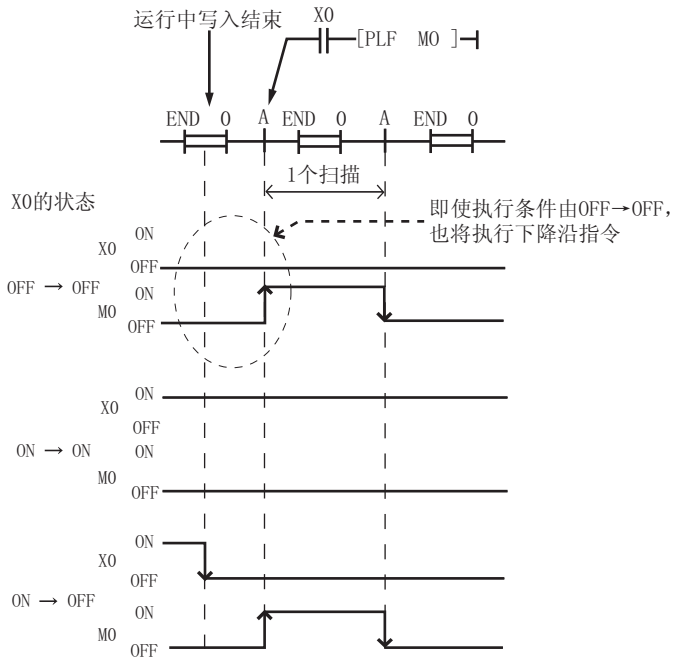


图 6.44 下降沿指令的动作

对象的下降沿指令为 LDF、ANDF、ORF、MEF、PLF、FCALLP、EFCALLP 指令。

(b) 上升沿指令

在运行中写入的写入范围内有上升沿指令时，运行中写入结束时即使上升沿指令的执行条件 (OFF → ON) 成立也不执行上升沿指令。

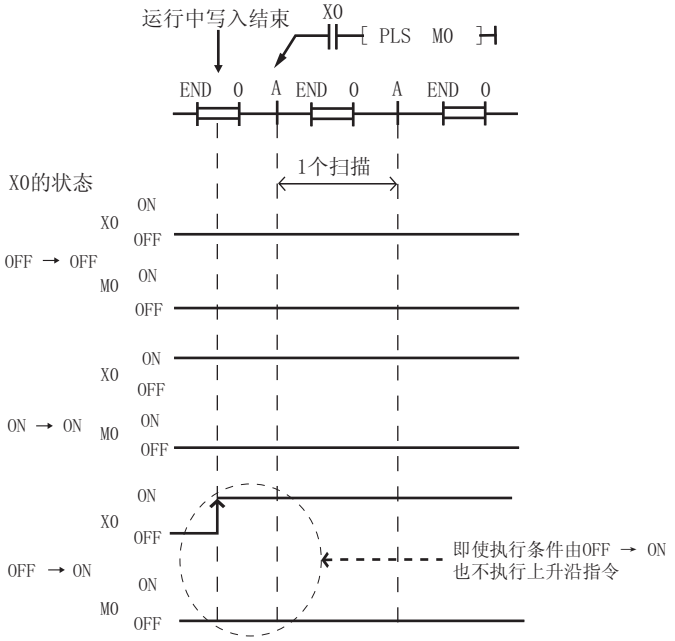


图 6.45 上升沿指令的动作

对象的上升沿指令为 PLS、□P 指令。

(c) SCJ 指令

在运行中写入的写入范围内有 SCJ 指令时，如果运行中写入结束时 SCJ 指令的执行条件为 ON，将不执行 1 个扫描等待而跳转到指定指针。

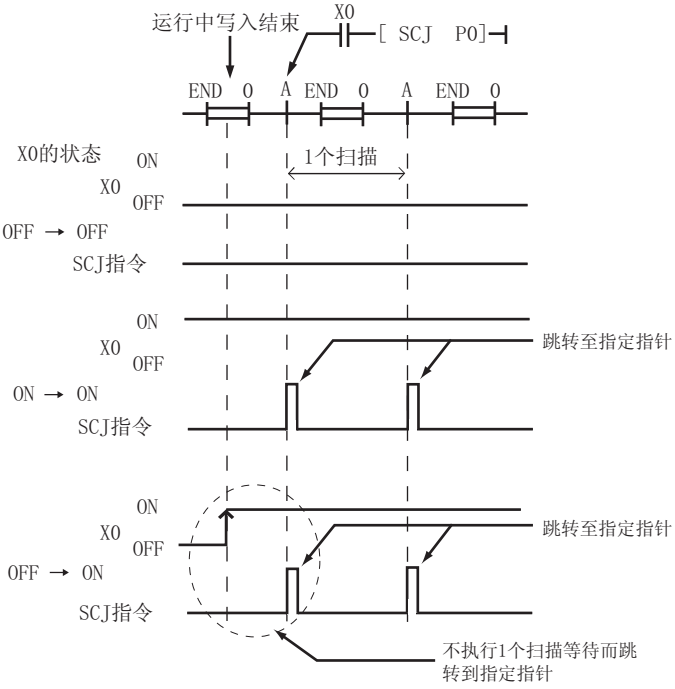


图 6.46 SCF 指令的动作

要 点

高性能 过程 冗余
注 6.38 注 6.38 注 6.38

通过在 GX Developer 的工具的选项菜单中进行 “运行中写入时的指令动作设置”，可以防止执行运行中写入时，即使下降沿指令的执行条件（ON → OFF）不成立，也将执行下降沿指令的现象。 注 6.38 注 6.39

默认情况因 CPU 模块而异。

基本
注 6.39

- 高性能模式 QCPU、过程 CPU、冗余 CPU 的情况
  执行执行条件为 OFF 的下降沿指令
- 通用型 QCPU 的情况
  不执行执行条件为 OFF 的下降沿指令

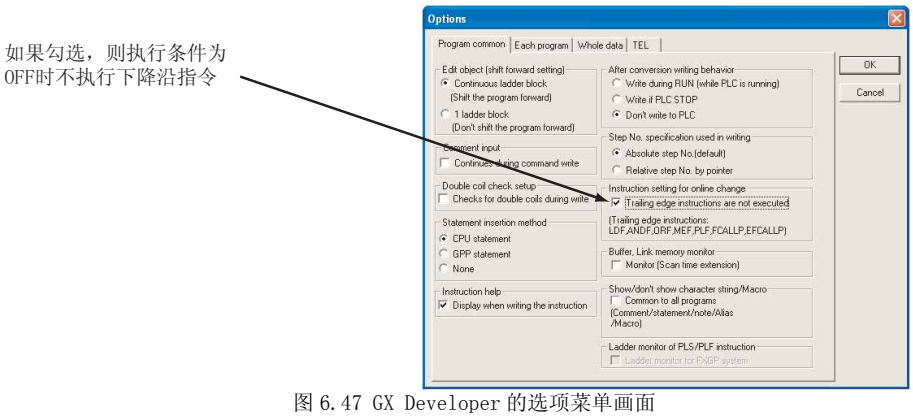
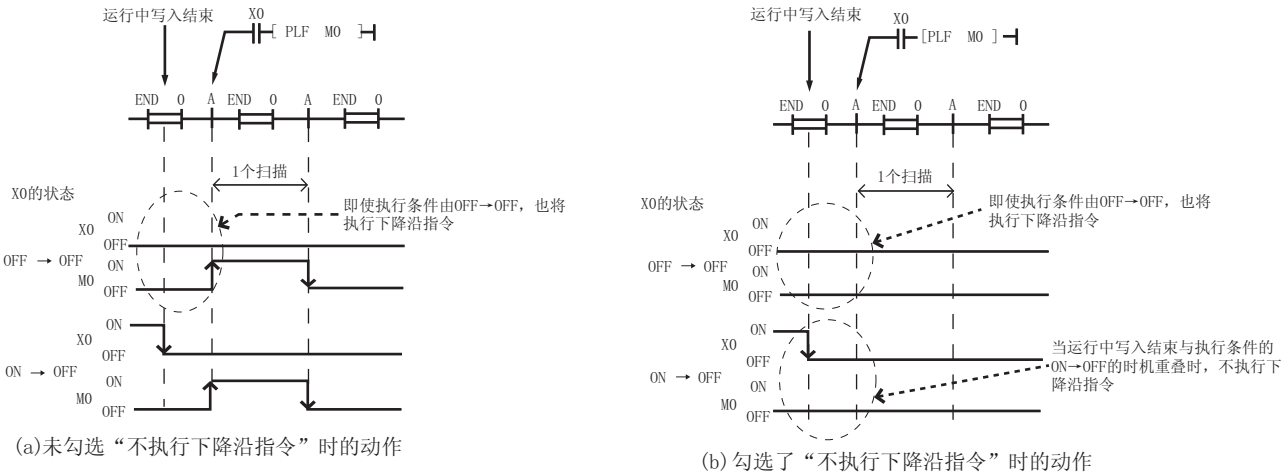


图 6.47 GX Developer 的选项菜单画面

根据 GX Developer 的 “运行中写入时的指令动作设置”，下降沿指令的动作如图 6.48 所示。



(a)未勾选“不执行下降沿指令”时的动作

(b) 勾选了“不执行下降沿指令”时的动作

图 6.48 下降沿指令的动作比较

高性能 过程 冗余
注 6.38 注 6.38 注 6.38

只有当下述的 CPU 模块与 GX Developer 组合时才可以执行。

- 序列号的高 5 位为 07092 或以后的高性能模式 CPU、过程 CPU、冗余 CPU。（冗余 CPU 时两个系统的冗余 CPU 均应为序列号的高 5 位为 07092 或以后的产品。）
- GX Developer：版本 8.27D 或以后

如果使用了序列号的高 5 位为 07091 或以前的 CPU 模块，或者版本为 8.26C 或以前的 GX Developer，将不能通过 GX Developer 的设置防止下降沿指令的执行。

基本
注 6.39

在基本模式 QCPU 中，不能通过 GX Developer 的设置防止下降沿指令的执行。

(4) 使用通用型 QCPU 时的注意事项

(a) 运行中写入以及 T/C 设定值变更时的至程序内存的写入

将通过运行中写入或者 T/C 设定值变更所更改的内容与至程序高速缓冲存储器的写入同时自动传送到程序内存中。

由于自动传送至程序内存，运行中写入以及 T/C 设定值变更所需的时间将延长如表 6.26 中所示的时间。

表 6.26 自动传送至程序内存导致的延长时间

CPU 模块	传送时间
Q02UCPU	Ts × 320 + 4.8 (s)
Q03UDCPU, Q04UDHCPU, Q06UDHCPU	Ts × 260 + 4.7 (s)

Ts: 扫描时间 (s)

此外，由于至程序内存（快闪卡）的写入次数是有限制（10 万次）的，频繁地进行运行中写入或者 T/C 设定值变更的情况下，应将至程序内存的自动传送设置为无效。

要 点

1. 通过 GX Developer 的选项设置可以将至程序内存的自动传送设置为无效。

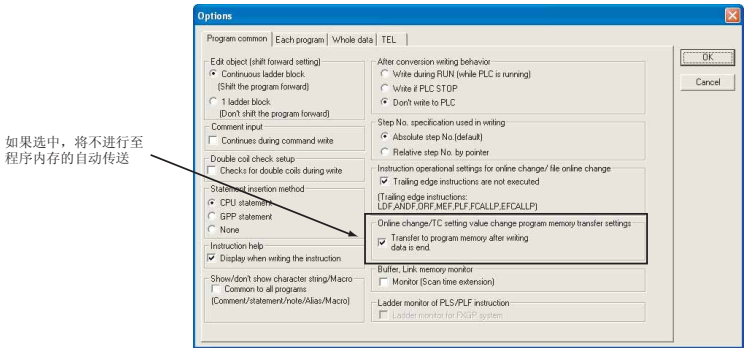
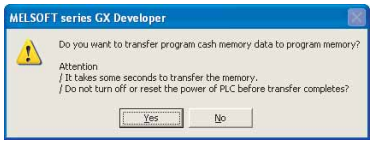


图 6.49 运行中写入、T/C 设定值变更程序内存传送设置画面

2. 将自动传送设置为无效时，运行中写入结束后将显示以下信息框。



如果选择 “是”，将执行至程序内存的传送。

如果选择 “否”，则不执行至程序内存的传送。

3. 在上述选择中选择了 “否”时，必须通过 GX Developer 进行程序内存批量传送。

(☞GX Developer 操作手册)

6.13 执行时间的测量

(1) 关于执行时间的测量

是指显示执行中的程序的处理时间的功能。

(2) 执行时间测量的用途与种类

在系统进行调整时，为了解各个程序的处理时间对于全部扫描时间的影响时使用。

执行时间的测量有下述三类。

- 程序监视一览表 :(☞6.13.1 项)
- 中断程序监视一览表 :(☞6.13.2 项)
- 扫描时间测量 :(☞6.13.3 项)

6.13.1 程序监视一览表

(1) 关于程序监视一览表

程序监视一览表是指显示执行中的程序的处理时间的功能。

显示各个程序的扫描时间、执行次数以及各项目的处理时间。

(2) 程序监视一览表的执行

程序监视一览表通过 [在线] → [监视] → [程序监视一览表] 显示程序监视一览表画面。

程序一览表的执行示例如图 6.50 所示。

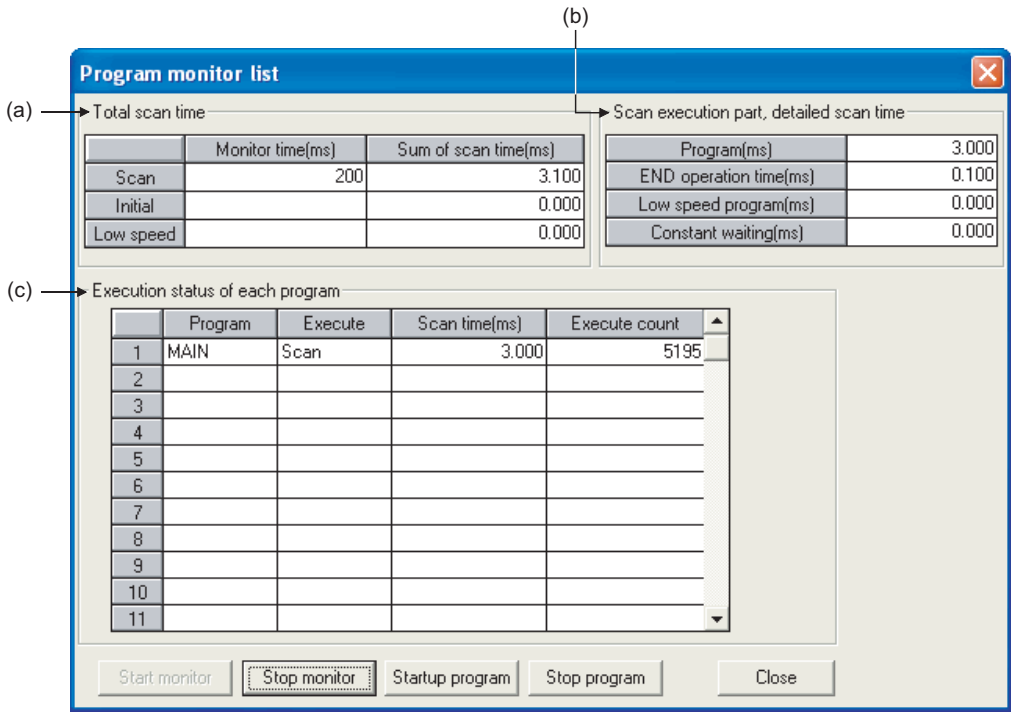


图 6.50 程序监视一览表

基本 冗余 通用
注 6.40 注 6.40 注 6.40

(a) 全部的扫描时间*注 6.40*

显示可编程控制器参数的 “可编程控制器 RAS 设定”的 WDT(看门狗定时器)设定中设定的监视时间以及 CPU 模块执行的各程序执行类型的扫描时间的合计时间。

1) 监视时间
显示各个程序的监视时间。
扫描时间如果超出这个时间，CPU 模块中会出现看门狗定时器溢出。

2) 扫描时间合计
显示 “扫描执行时的扫描时间的详细情况”中的各个项目的合计时间。
设定恒定扫描时间的情况下，显示恒定扫描时间。

(b) 扫描执行时的扫描时间的详细情况
显示扫描时间的详细情况。

1) 程序
显示扫描执行类型程序的执行时间的合计时间。

2) END 处理时间
显示 END 处理时间。

基本 冗余 通用
注 6.41 注 6.41 注 6.41

3) 低速程序*注 6.41*
显示低速执行类型程序的执行时间。
另外，在设定恒定扫描时间时，显示低速执行类型程序的执行时间的合计时间。

4) 恒定等待
在设定恒定扫描时间时，显示恒定扫描的等待时间。
但是，在同时设定低速程序执行时间的情况下，将显示 0.000ms。

基本 冗余 通用
注 6.40 注 6.40 注 6.40

在基本模式 QCPU 中，仅显示扫描执行类型程序的监视时间。
在冗余 CPU、通用型 QCPU 中，不显示低速执行类型程序的监视时间。

基本 冗余 通用
注 6.41 注 6.41 注 6.41

在基本模式 QCPU、冗余 CPU、通用型 QCPU 中，不显示低速程序的项目。

(c) 各程序的执行状态

基本 注 6.42

显示在可编程控制器参数的程序设定中指定程序的执行状态。注 6.42

1) 程序
按照可编程控制器参数设定的顺序号显示程序名。

2) 执行
显示在可编程控制器参数中设定的程序的执行类型。

3) 扫描时间
显示实际的扫描时间（当前值）。
在程序停止（待机）状态下，扫描时间显示为 0.000ms。

4) 执行次数
测量开始的时点为第 0 次，显示一直到监视时的执行的次数。
显示的执行次数最大为 65535 次，在第 65536 次的测量将返回到 0 次开始。
即使程序停止的时候执行的次数也将被保留。

备 注

如果通过 POFF 指令更改了程序的执行类型，执行次数的情况如下所示。

- 基本模式 QCPU、高性能模式 QCPU、过程 CPU、冗余 CPU ..... 不计入执行次数。
- 通用型 QCPU ..... 计入执行次数。

详细内容请参阅以下手册中有关 POFF 指令的项目。

☞ QCPU(Q 模式)/QnACPU 编程手册（公共指令篇）

基本 注 6.42

在基本模式 QCPU 中，由于不能执行多个程序的原因，各个程序的执行状况如下所示。
程序 ：MAIN 或者 MAIN-SFC
执行 ：扫描

基本 通用
注 6.43 注 6.43

(3) 程序的启动*注 6.43*

通过程序监视一览表画面进行程序的启动。

如果点击程序监视一览表画面（☞本项 (2)）上的 Startup program （程序启动）的按钮，将显示下述的对话框。

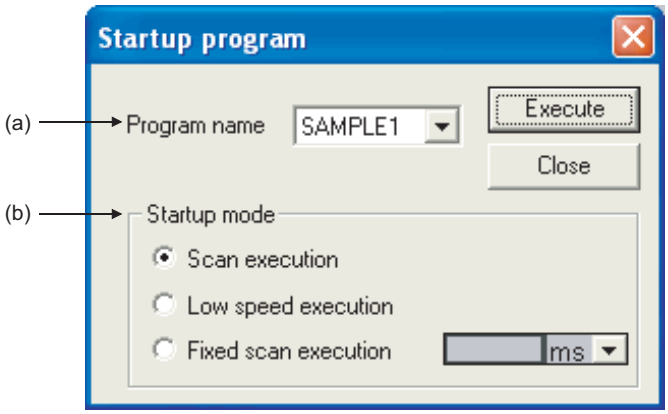


图 6.51 程序启动画面

(a) 程序名

选择可编程控制器参数的程序设定中设定的程序。

不能任意输入程序名。

(b) 启动模式

可以将待机类型程序设定为 “扫描执行”、“低速执行”*注 6.44*、“恒定周期执行”。

冗余 通用
注 6.44 注 6.44

恒定周期的默认值显示在可编程控制器参数的程序设定中设定的值。单位可以选择

ms、s。

基本 通用
注 6.43 注 6.43

在基本模式 QCPU、通用型 QCPU 中，不能通过程序监视一览表画面进行程序的启动。

冗余 通用
注 6.44 注 6.44

在冗余 CPU、通用型 QCPU 中，由于不能使用低速执行类型程序，因此在启动模式中不能选择 “低速执行”。

基本 通用
注 6.45 注 6.45

(4) 程序的停止[注 6.45]

通过程序监视一览表画面进行程序的停止。

如果点击程序监视一览表（☞ 本项 (2)）上的 Stop program （程序停止）按钮，将显示下述的对话框。

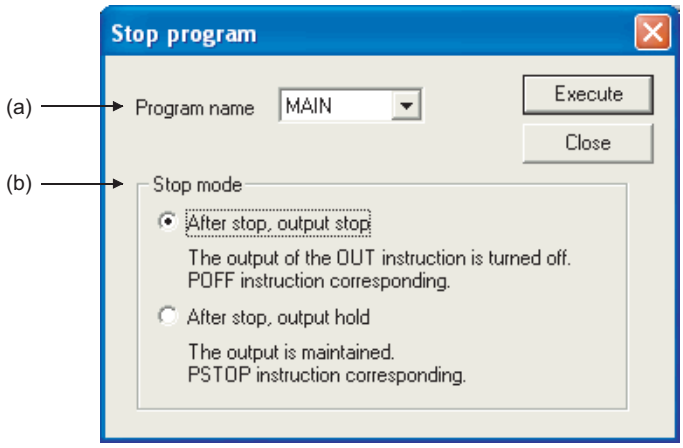


图 6.52 程序停止画面

(a) 程序名

选择可编程控制器参数的程序设定中设定的程序。

不能任意输入程序名。

(b) 停止模式

冗余 通用
注 6.46 注 6.46

- 如果对扫描执行类型程序执行 “停止之后，输出停止”，在下一个扫描周期内输出将 OFF（非执行处理）。
  再下一个扫描周期以后进入待机状态。（执行 POFF 指令时为同样的动作）
- 如果对低速执行类型程序[注 6.46]执行 “停止之后，输出停止”，将中断低速执行类型程序并在下一个扫描周期内将输出 OFF（非执行处理）。
  再下一个扫描周期以后成为待机类型程序。（执行 POFF 指令时为同样的动作）
- 如果对待机类型程序执行 “停止之后，输出停止”，其将作为扫描执行类型在执行了一个扫描周期 OFF 后停止。
  因此，“执行次数”需要加 1。
- 待机类型程序在执行一个扫描周期 OFF 过程中即使由于 RET/IRET 指令出现出错时，“执行次数”也需加 1。
  此时的执行类型成为 “扫描执行”。

基本 通用
注 6.45 注 6.45

在基本模式 QCPU、通用型 QCPU 中，不能通过程序监视一览表画面进行程序的停止。

冗余 通用
注 6.46 注 6.46

在冗余 CPU、通用型 QCPU 中，不能使用低速执行类型程序。

要 点

即使执行 “停止之后，停止输出”，根据指令输出也有不 OFF 的情况。
关于详细的情况，请参照下述手册的 POFF 指令的项目。
☞QCPU(Q 模式)/QnACPU 编程手册（公共指令篇）

(5) 注意事项

执行恒定周期类型程序时，不显示恒定周期执行类型程序的扫描时间。
在扫描时间的栏中将显示横杠（—）。

6.13.2 中断程序监视一览表

(1) 关于中断程序监视一览表

是指表示中断程序执行次数的功能。

在确认中断程序的执行状态时使用。

(2) 中断程序监视一览表的执行

中断程序监视一览表通过 [在线] → [监视] → [中断程序监视一览表] 显示中断程序监视一览表窗口。

中断程序一览表的执行示例如图 6.53 所示。

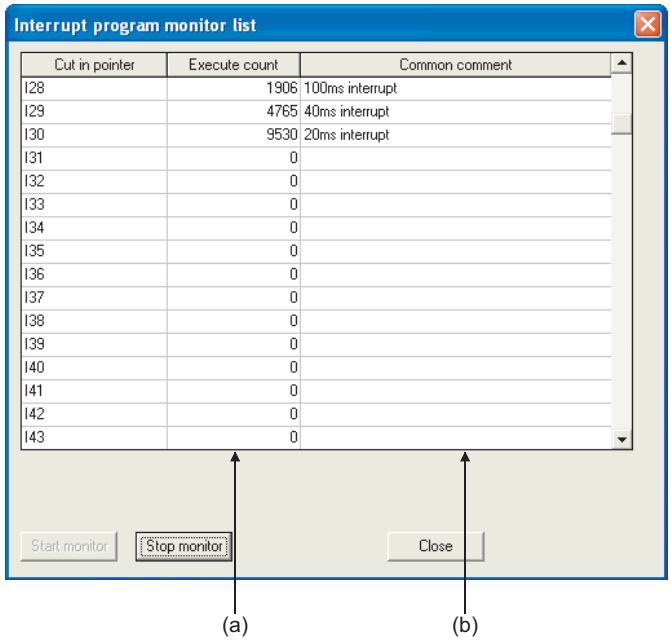


图 6.53 中断程序监视一览表

(a) 执行次数

显示执行中断程序执行的次数。

CPU 模块进入运行状态时开始测量，在达到最大 65536 次时将返回到 0。

(b) 公共注释

显示创建中断指针的软元件注释。

基本 注 6.47

通用 注 6.47

6.13.3 扫描时间的测量

(1) 关于扫描时间的测量

扫描时间的测量是指在执行梯形图监视时显示程序的任意区间的处理时间的功能。
也可以测量副程序、中断程序内的时间。

(2) 指定扫描时间的测量范围

扫描时间的测量范围的指定有下述两种方法。

- 通过梯形图监视画面指定。
- 通过扫描时间测量窗口指定。

(3) 存在副程序的调用指令的情况

在扫描时间的测量范围内存在副程序的调用指令（CALL 指令）时，时间将包含副程序的处理时间。

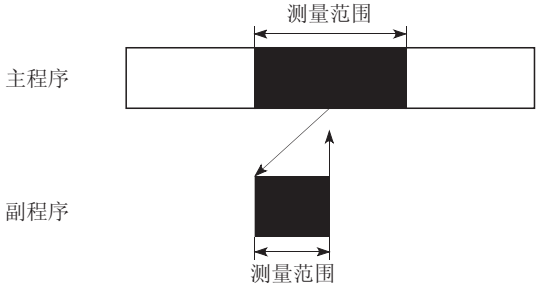


图 6.54 在测量范围内包含副程序的情况

(4) 执行中断程序 / 恒定周期执行类型程序的情况

执行中断程序 / 恒定周期执行类型程序时，加上了执行时间。

(5) 扫描时间测量的执行

扫描时间的测量按照下述的顺序进行。

- 通过 GX Developer 显示进行扫描时间测量的梯形图起始后，进入监视模式。

基本 注 6.47　通用 注 6.47

在基本模式 QCPU、通用型 QCPU 中，不能使用扫描时间测量。

• 指定扫描时间测量范围。（指定部分为反影）

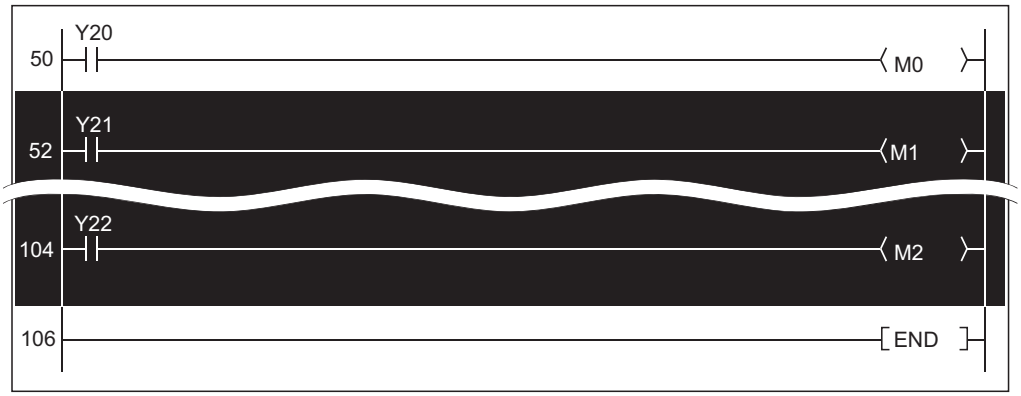
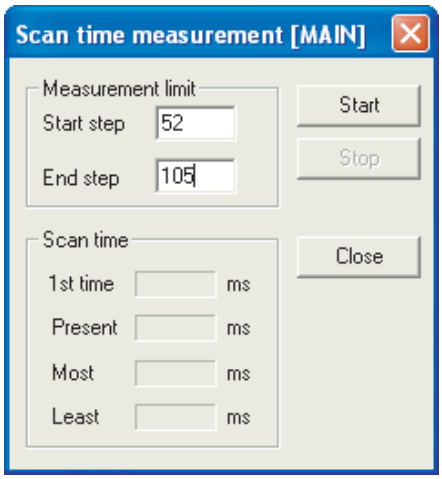


图 6.55 测量范围的指定

• 通过 [在线] → [监视] → [扫描时间测量] 显示扫描时间测量窗口。



• 选择 Start （开始）按钮。

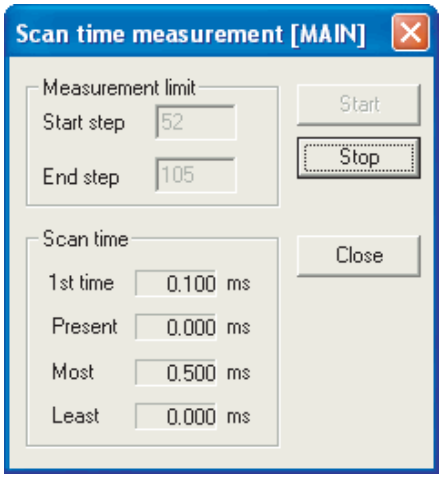


图 6.56 扫描时间测量窗口

(6) 注意事项

(a) 设定测量范围时
请设定为（起始步）<（结束步）。

(b) 跨程序文件的扫描时间的测量
不能进行跨程序文件的扫描时间的测量。

(c) 测量时间在没有达到 0.100ms 时
测量时间在没有达到 0.100ms 时，显示 0.0000ms。

(d) 指定 FOR~NEXT 之间指令的情况
在指定 FOR~NEXT 之间指令的情况下，为执行了一次指定步的时间

基本 注 6.48 6.14 采样追踪功能

(1) 采样追踪功能

采样追踪功能是指在指定的时机连续收集 CPU 模块的指定软元件内容的功能。
采样追踪在一定的间隔期内（采样周期）对指定软元件的内容进行采样，将追踪的结果存储在内存卡内的采样追踪文件内。

(2) 采样追踪的用途

在指定时机可以确认在调试时程序中使用的软元件的内容的变化。
另外，也进行触发条件成立时的软元件的内容的接收。

(3) 采样追踪的动作

在采样追踪文件中，存储执行采样追踪用的追踪设定以及追踪结果。
如果通过 GX Developer 执行追踪开始，则进行设定的追踪次数的追踪。
采样追踪用的区域最大为 60k 字节。（在通用型 QCPU 中，无容量限制。）
追踪次数为采样追踪用区域的字节数除以设定的软元件的字节数 (N1+N2+N3+ 字软元件点数× 2+(位软元件点数 /16) × 2) 的值。*1*2

*1: 关于计算式中的 “位软元件点数 /16”，请将小数点以后的数进位。
*2: N1 ～ N3 为根据追踪条件设置画面的追踪附加信息中设置的项目而相加的值。
N1: 设置时间时加上 “4”
N2: 设置步号时加上 “10”
N3: 设置程序名时加上 “8”

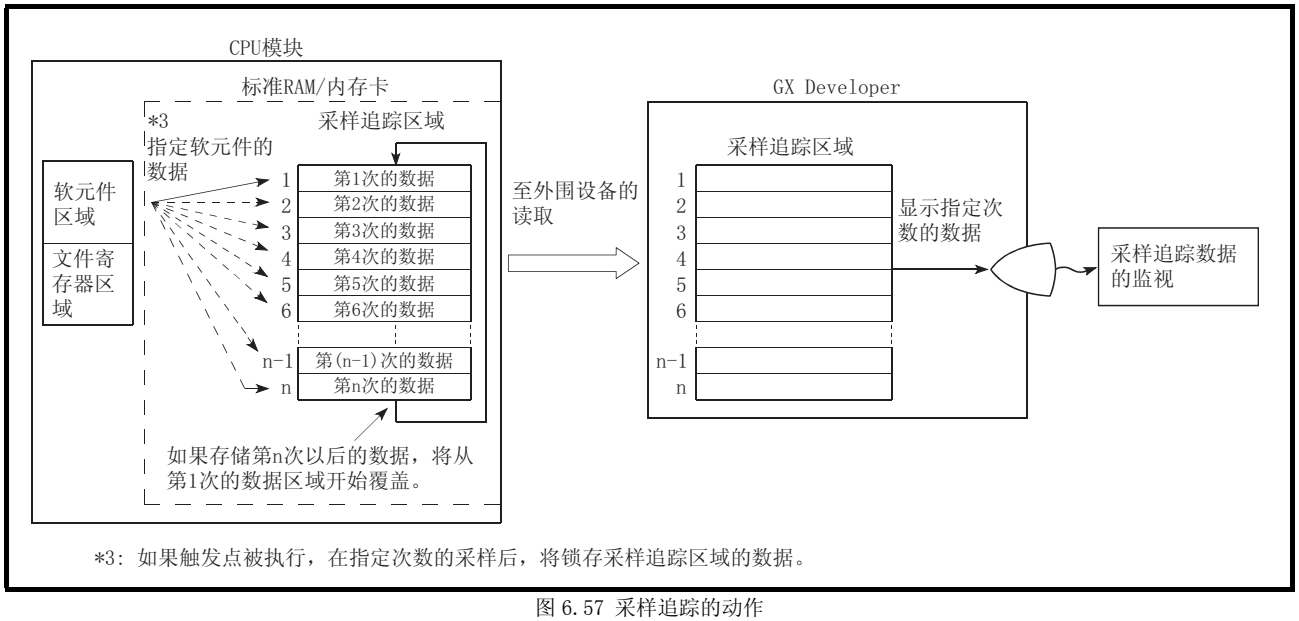


图 6.57 采样追踪的动作

基本 注 6.48

在基本模式 QCPU 中，不能使用采样追踪功能。

要 点

1. 采样追踪文件只能存储在标准 RAM 或者 SRAM 卡中。(☞ 5.2.1 项 (2))
2. 将采样追踪文件存储到标准 RAM 中时，应确认 CPU 模块以及 GX Developer 的版本。(☞ 附录 4)

(a) 执行采样追踪时

采样追踪功能的执行状态存储在特殊继电器 (SM800, SM802, SM804, SM805) 中。
另外，在采样追踪功能执行中如果发生出错，SM826 将 ON。
由于顺控程序中使用了上述特殊继电器，因此可以确认采样追踪功能的执行状态。

1) 采样追踪的准备

通过 GX Developer 设定的 “追踪数据”、“追踪条件”至 CPU 模块的写入结束后，SM800（采样追踪准备）将 ON。
通过 SM800 可以确认采样追踪是否执行。

2) 采样追踪的开始

如果采样追踪开始要求成立，则开始采样追踪，SM802（采样追踪执行中）将 ON。
通过 SM802 可以确认采样追踪是否执行。

- 通过 GX Developer 的跟踪开始
- SM801 ON

3) 触发条件的确认

下述的触发条件的任何一个成立的话，则 SM804 “ON”（采样追踪触发后）。
通过 SM804 可以确认触发条件是否成立。

- 通过 GX Developer 的触发执行
- TRACE 指令的执行
- SM803 ON
- 详细设定（软元件号、步号）

(b) 采样追踪结束时

如果采样追踪的执行结束，则 SM805(采样追踪结束)ON。

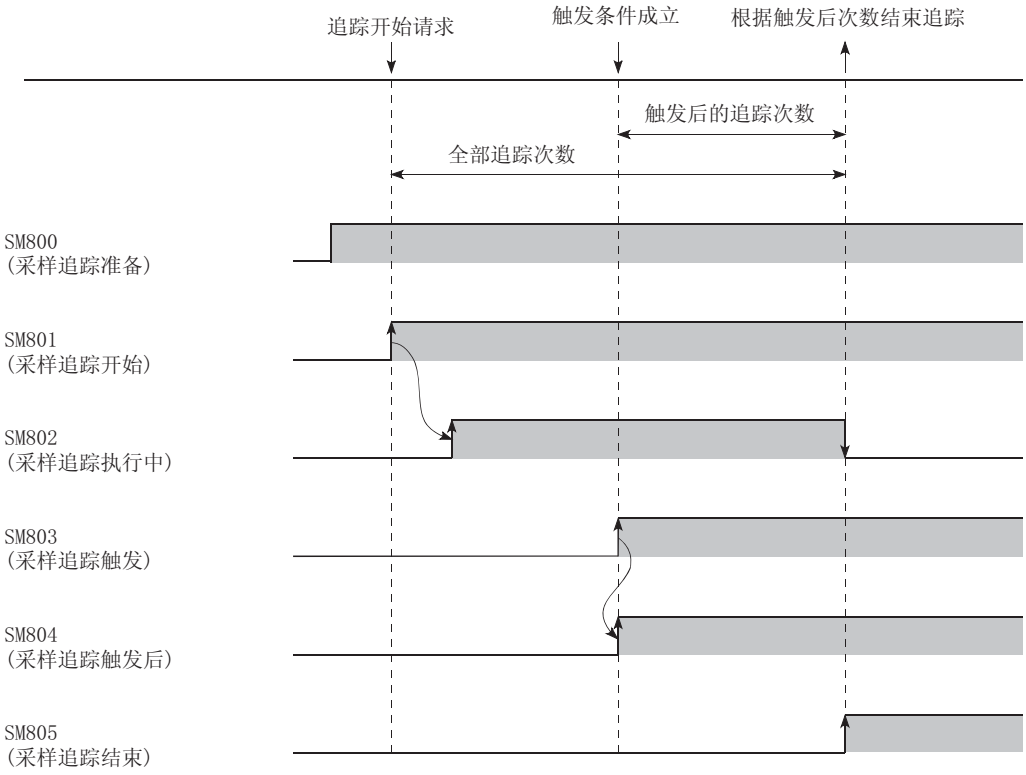
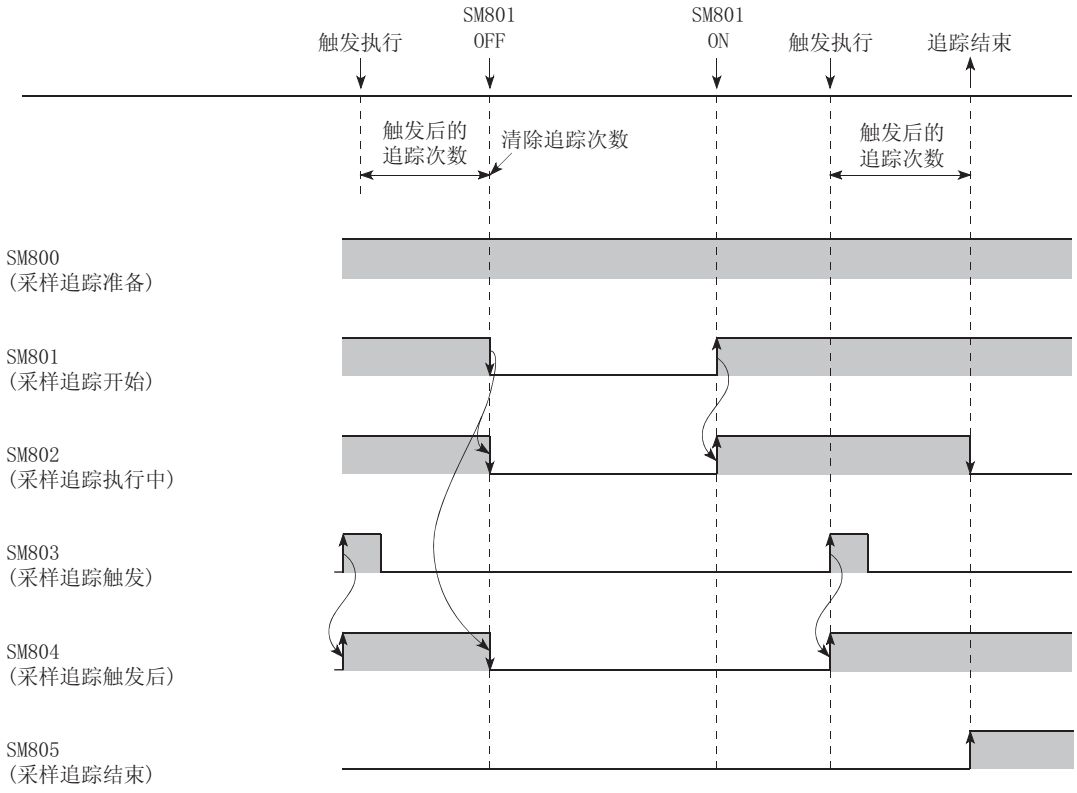


图 6.58 采样追踪时序图

(c) 采样追踪中断时

在采样追踪过程中如果 SM801(采样追踪开始)OFF，则中断采样追踪 OFF。
如果采样追踪 OFF，将清除跟踪次数。
如果 SM801 再次 ON，将再次开始采样追踪。



* 通过GX Developer使追踪中断时，SM800也将OFF。

图 6.59 采样追踪时序图(采样追踪中断时)

(4) 操作顺序

采样追踪的操作顺序有下述方法。

- 使用向导的方法（☞GX Developer 操作手册）
- 个别进行详细设定的方法（☞如下）

各个操作通过［在线］→［跟踪］→［采样追踪］菜单进行。

(a) 跟踪数据（设定 + 结果）的存储目标、跟踪执行方法的设定

进行跟踪数据的存储目标与跟踪执行方法的设定。

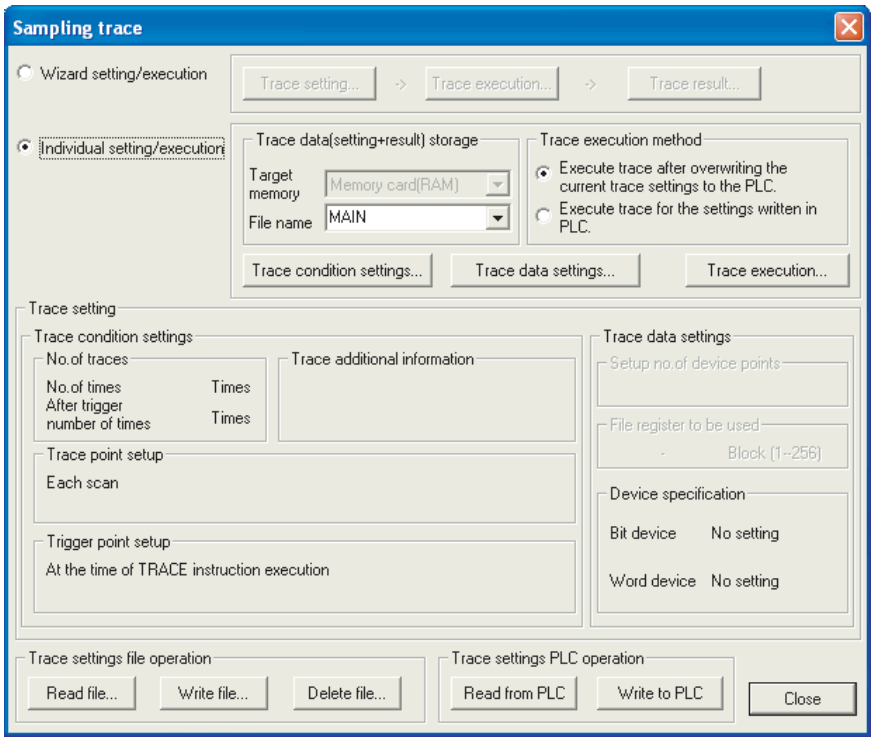


图 6.60 采样追踪画面

1) 追踪数据（设定 + 结果）的存储目标

设定存储追踪数据的对象存储器以及写入追踪条件的文件名。

选择 “标准 RAM” 或者 “存储卡 (RAM)” 之一作为对象存储器。

以指定的文件名将追踪结果存储到本设置中设定的对象存储器中。

2) 追踪执行方法

从下述方法中选择追踪执行方法：

- 将当前的追踪设定覆盖到 CPU 模块后执行

  将追踪设定覆盖到原有的采样追踪文件中。
- 根据写入到 CPU 模块中的追踪设定执行

  按照追踪数据（设定 + 结果）存储对象中指定的采样追踪文件的追踪设定执行。

(b) 跟踪条件的设定

在采样追踪画面中点击 Trace condition setting（跟踪条件设定）按钮，设定跟踪条件。

在跟踪条件设定中，可以设定跟踪次数、跟踪点设定、触发点设定以及跟踪附加信息。

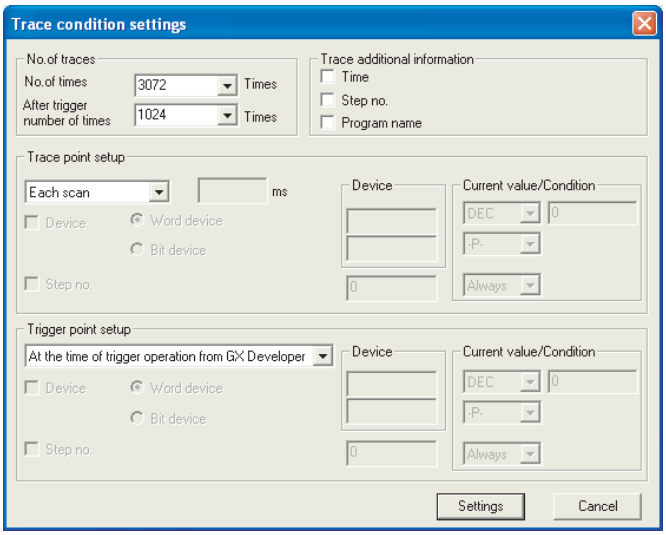


图 6.61 跟踪条件设定画面

1）跟踪次数

跟踪次数是指设定跟踪开始到结束的采样追踪执行次数。

触发后的次数是指设定从触发执行开始到跟踪结束的采样追踪执行的次数。

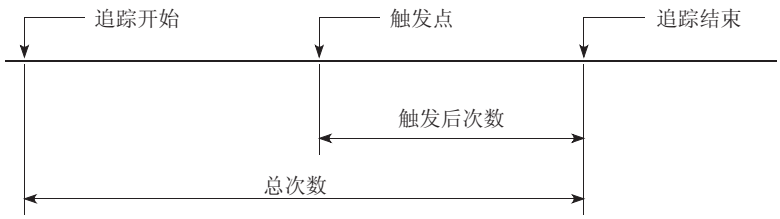


图 6.62 总次数与触发后次数的关系

各个次数的设定范围如下所示。

（触发后的次数）≤（总次数）≤（8192 次）

2) 跟踪点的设定
设定收集跟踪数据的时机。
在下述中选择一个

- 每个扫描周期
  在每个扫描周期的 END 处理时进行数据收集。
- 每个时间段
  在每个指定时间内进行数据的收集。
- 多 CPU 间各高速通信周期（仅通用型 QCPU）
  在 0.88ms ×指定间隔的各时间进行数据采集。
  由于在执行 I45 的中断程序的时机进行追踪数据的采集，因此只有在以下条件全部成立时才可以进行追踪数据的采集。
  1) 使用多 CPU 间高速主基板 (Q3□DB)。
  2) 在 CPU 个数为 2 个以上的多 CPU 系统中，且被设定为使用多 CPU 间高速通信功能。
  3) 程序中存在有 I45 的中断指针。
  4) 处于 EI 状态，且 I45 的中断屏蔽已被解除。
- 详细设定
  设定跟踪点的软元件以及步号。
  设定方法以及跟踪数据收集的时机与 6.11.1 项的监视条件的设定相同。
  此外，在通用型 QCPU 中，除上述以外，可以将采集时机选择为字软元件值的变更时。
  在详细设定中可以设定的软元件如下所示。

  •位软元件 :X(DX)、Y(DY)、M、L、F、SM、V、B、SB、T（触点）、ST（触点）、C（触点）、FX、FY、J□\X、J□\Y、J□\B、J□\SB、BL□\S
  •字元件 :T（当前值）、ST（当前值）、C（当前值）、D、SD、W、SW、R、Z、ZR、FD、U□\G、J□\W、J□\SW、U3E□\G（仅通用型 QCPU）

  针对上述软元件，可以进行以下的修饰：

  •指定位软元件的位数
  •指定字元件的位号

3) 跟踪附加信息
在每个跟踪中设定附加信息，从下述中可以选择多个。（也可以不选择）

- 时间
  存储执行跟踪的时间。
- 步 No.
  存储执行跟踪的步 No.。
- 程序名
  存储执行跟踪的程序名。

4) 触发点的设定
设定执行触发的点。从下述中选择一个。

- 执行 TRACE 指令时
  在执行 TRACE 指令时进行触发。
- 从周边进行触发操作时
  通过 GX Developer 执行触发时进行触发。
- 详细设定
  设定触发点的软元件以及步号。
  设定方法以及触发执行的时机与 6.11.1 项的监视条件的设定相同。
  此外，在通用型 QCPU 中，除上述以外，可以将采集时机选择为字软元件的写入时。
  在详细设定中可设置的软元件如下所示。

  •位软元件 :X(DX)、Y(DY)、M、L、F、SM、V、B、SB、T(触点)、ST(触点)、C(触点)、FX、FY
  •字元件 :T(当前值)、ST(当前值)、C(当前值)、D、SD、W、SW、R、ZR。

  对上述软元件可以进行以下修饰。
  •字软元件的位号指定。

(c) 跟踪数据的设定

选择采样追踪画面的 “个别设定 / 执行”。

点击 Trace data setting （跟踪数据设定）按钮，设定执行采样追踪的软元件。

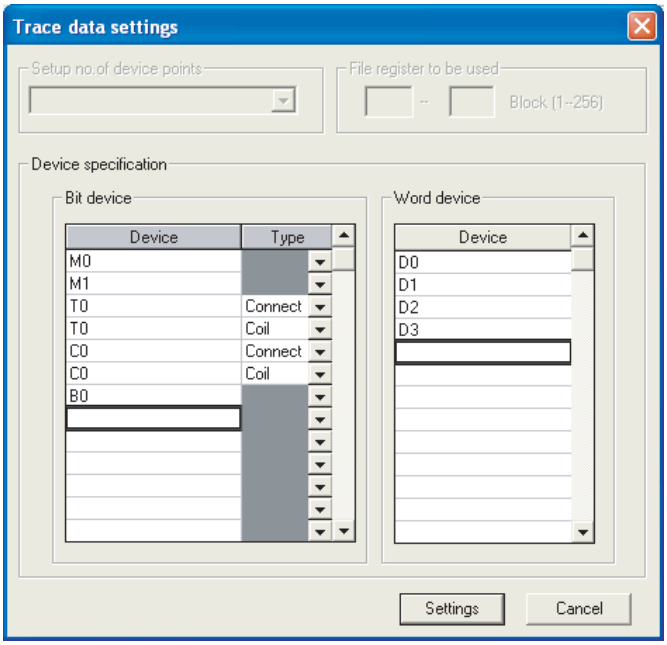


图 6.63 跟踪数据的设定画面

1) 位软元件

下述位软元件最大可以设定 50 点。

X、DX、Y、DY、M、L、F、SM、V、B、SB
T(触点)、T(线圈)、ST(触点)、ST(线圈)、C(触点)、C(线圈)
J□\X、J□\Y、J□\B、J□\SB、BL□\S

2) 字元件

下述字元件最大可以设定 50 点。

T(当前值)、ST(当前值)、C(当前值)、D、SD、W、SW、R、Z、ZR
U□\G、J□\W、J□\SW、U3E□\G(仅通用型 QCPU)

(d) 追踪数据以及追踪条件的写入

创建的追踪数据设定、追踪条件设定被作为采样追踪文件写入到“追踪数据（设定 + 结果）存储对象”中设置的对象存储器中。

采样追踪文件的写入是通过采样追踪画面的 Write to PLC （可编程控制器写入）按钮进行。

要 点

将采样追踪文件存储到存储卡（SRAM 卡）中时，通过更改文件名，可以存储多个采样追踪文件。
存储到标准 RAM 中时，只能存储 1 个采样追踪文件。
使用多个采样追踪文件时，应使用存储卡（SRAM 卡）。

(e) 采样追踪的执行

执行采样追踪。

点击采样追踪画面的 Execute （跟踪执行）按钮，显示采样追踪执行画面。

执行采样追踪。

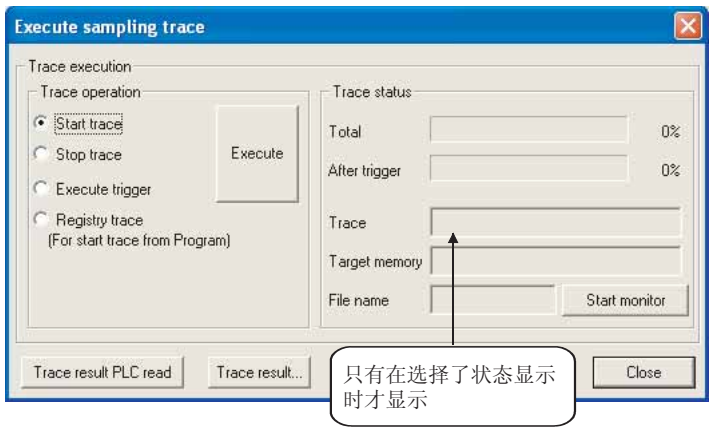


图 6.64 执行、状态显示画面

在执行、状态显示画面中可以设定跟踪操作。

- 跟踪开始
  开始跟踪并开始跟踪次数的计数。
- 中断跟踪
  中断跟踪。清除计数的跟踪次数、触发后的次数。
  （重新启动追踪时，请再次选择 “跟踪开始”）
- 触发执行
  开始触发后次数的计数。
  在计数到设定的触发后次数时，终止跟踪。
- 跟踪登录（通过程序执行时使用）
  通过程序执行时进行跟踪登录。

(f) 跟踪结果的显示

将跟踪的结果从 CPU 模块中读出并显示。

- 通过采样追踪执行画面的 Trace result PLC read （跟踪结果可编程控制器读出）按钮从 CPU 模块中读出跟踪结果。
- 显示通过 Trace result （跟踪结果）按钮读出的跟踪结果。

跟踪结果显示每个采样周期的位软元件的 ON/OFF 状态以及字元件的当前值。

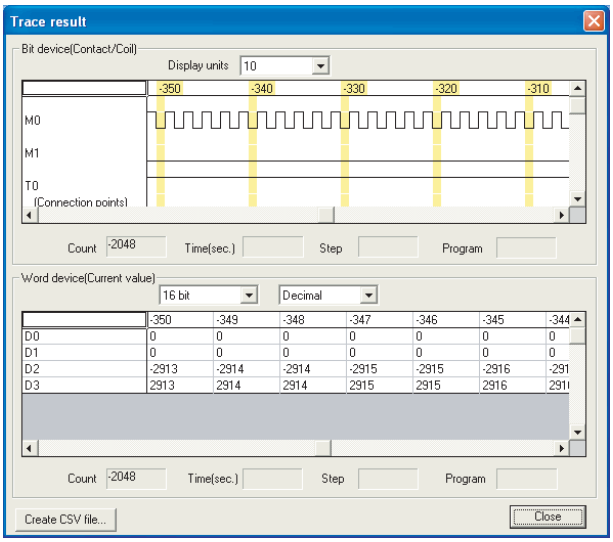


图 6.65 跟踪结果的显示

要 点

1. 在触发点设定中设定的条件（触发条件）成立状态下接收指定的软元件的内容。

   在每次扫描周期中进行采样并通过周边的触发操作结束采样时，由于触发条件成立与采样为同一时间，因此，进行两次数据的收集。

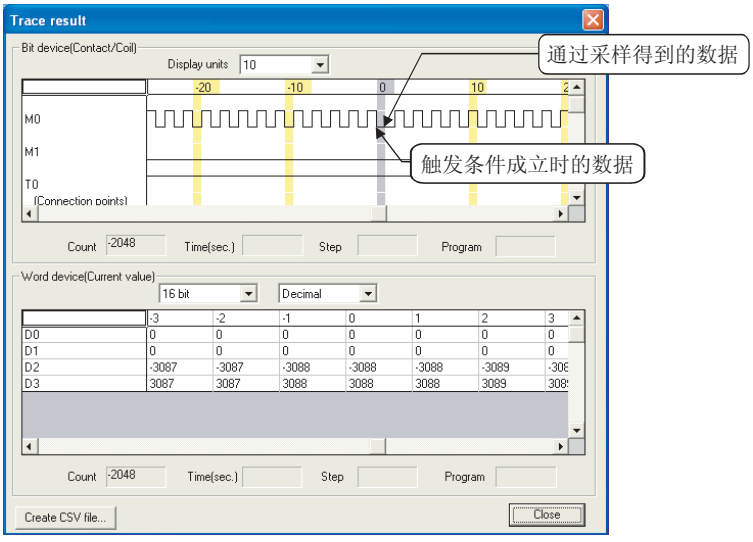


图 6.66 跟踪结果的显示

2. 如果执行一次采样追踪，则不执行第二次。

   需再度执行的情况下，请执行 TRACER 指令并对采样追踪进行复位。

(5) 追踪执行状态的清除方法

通用 UD 注 6.49

通过 RESET/L.CLR 开关 注 6.49 或者远程锁存清除操作进行的锁存清除可以清除追踪执行状态。锁存清除后，再次进行采样追踪时应选择 “追踪开始”或者 “追踪登录”之后再执行。

(6) 注意事项

(a) 可进行采样追踪的位置

从网络上的其它站或者串行口通信模块也可执行采样追踪。
但是，不能从多个位置同时进行追踪。
在 CPU 模块中一次只能从一个位置执行采样追踪。

(b) 关于追踪设定的保持与清除

登录在 CPU 模块上的追踪设定（采样追踪文件）将被锁存。
即使进行了电源的投入 (OFF → ON)/CPU 模块的复位，通过登录时的追踪设定也可以再次执行采样追踪。但是，无法读取上一次的追踪结果。
此外，在下述情况下，如果采样追踪的触发条件成立，由于未被识别为触发条件，因此锁存的追踪设定将被清除 (SM800（追踪准备）将 OFF。)
应通过 GX Developer 再次登录追踪设定。

1) 在对象存储器中选择了 “标准 RAM”时

- 进行了改变标准 RAM 内的局部软元件的大小设置 *1、将可编程控制器参数写入到 CPU 模块后，进行了电源的投入 (OFF → ON)、CPU 模块的复位或者 STOP → RUN。
- 在采样追踪文件已被损坏的状态下，进行了电源的投入 (OFF → ON) 或者 CPU 模块的复位。

*1: 也包括新建局部软元件时。

要 点

按以下步骤可以将追踪结果保存到个人计算机中。

- 通过采样追踪执行画面的 Trace result PLC read （追踪结果可编程控制器读取） 按钮，将追踪结果读取到个人计算机中。(☞ 本项 (4)(e))
- 点击 Trace result （追踪结果）按钮，显示追踪结果。
- 点击 Create CSV file （创建 CSV 文件）按钮，将追踪结果以 CSV 格式保存。

通用 UD 注 6.49

在通用型 QCPU 中，不能通过开关进行锁存清除。

2) 在对象存储器中选择了 “存储卡 (RAM)” 时
- 在未安装登录了采样追踪文件的 SRAM 卡的状态下，进行了电源的投入 (OFF → ON) 或者 CPU 模块的复位。
- 在采样追踪文件已被损坏的状态下，进行了电源的投入 (OFF → ON) 或者 CPU 模块的复位

(c) 执行采样追踪时与 CPU 模块的连接
采样追踪是通过将 GX Developer 连接在 CPU 模块上进行的。

(d) 关于 STOP 状态时追踪结果的读出
在 CPU 模块处于 STOP 状态时，不能读出追踪结果。
需要读出追踪结果时，应在运行状态下进行读取。

(e) 关于执行采样追踪时的触发条件
执行采样追踪时，应将触发点设定的触发条件设定为不成立。
执行采样追踪时触发条件成立的情况下，由于不将其识别为触发条件，因此追踪信息将被清除。(SM800 (追踪准备) 将 OFF。)
但是，在通用型 QCPU 中，通过将 SM829 (追踪设定的强制登录指定) 置于 ON，即使追踪条件或者触发条件成立，也可以登录采样追踪设定。

(f) 通过追踪条件的详细设置在指定软元件中指定了文件寄存器时
通过追踪点以及触发点设置的详细设置在指定软元件中指定了文件寄存器时，追踪登录后，不要更改文件寄存器文件以及文件寄存器的块号。
否则可能导致不能正常采集追踪数据。

高性能 过程 冗余
注 6.50 注 6.50 注 6.50

(g) 关于追踪点设定 *注 6.50*
在各时间或者各多 CPU 间高速通信周期中进行了追踪点设定时，由于被作为中断处理执行，因此应注意采样间隔及一次采样处理时间。如果一次采样处理时间过长，有可能发生看门狗定时器错误。

高性能 过程 冗余
注 6.50 注 6.50 注 6.50

在高性能模式 QCPU、过程 CPU、冗余 CPU 中，不能对各多 CPU 间高速通信周期进行设置。

6.15 多人进行的调试功能

(1) 关于多人进行的调试功能

多人进行的调试功能是指通过连接在 CPU 模块、串行口通讯模块等的多个 GX Developer 同时进行调试的功能。

基本

对各工程、各功能等进行文件分配时，通过多个 GX Developer 同时对其它文件进行调试时使用。注 6.51

注 6.51

(2) 功能说明

通过多人进行调试功能的组合如下所示。

表 6.27 调试功能的组合

执行中的功能	之后执行的功能			
	监视	运行中写入	执行时间计测	采样追踪
监视	○*1	○*2	○	○
运行中写入	○*2	×*4	×	×
执行时间计测	○	×	×*3	○
采样追踪 注 6.52	○	×	○	×*3

基本

注 6.52

○：可以同时执行。但是，由于监视条件的设定（6.11.1 项）只对来自一个 GX Developer 有效，因此其它的 GX Developer 中不可以设定监视条件。

×：只能通过一个 GX Developer 执行。

*1：由于监视条件的设定（6.11.1 项）只对来自一个 GX Developer 有效，因此其它的 GX Developer 中不可以设定监视条件。注 6.53

基本 通用

注 6.53 注 6.53

*2：不能同时执行设置了监视条件的监视及运行中写入。

*3：不能多人同时执行。

从其它的 GX Developer 执行时，应停止或中断执行中的执行时间计测以及采样追踪之后再执行。

*4：关于多人对 1 个文件进行运行中写入的方法，请参阅 6.15.2 项。

基本

注 6.51

在基本模式 QCPU 中，由于不能对各工程、功能分配文件，因此，不能使用对其它文件调试的功能。

基本

注 6.52

在基本模式 QCPU 中，不能使用采样追踪。

基本 通用

注 6.53 注 6.53

在基本模式 QCPU、通用型 QCPU 中，不能使用设定了监视条件的监视。

6.15.1 多个人同时进行监视的功能

(1) 关于多人同时进行监视的功能

是指通过连接在 CPU 模块、串行口通讯模块等的多个 GX Developer 同时进行监视的功能。

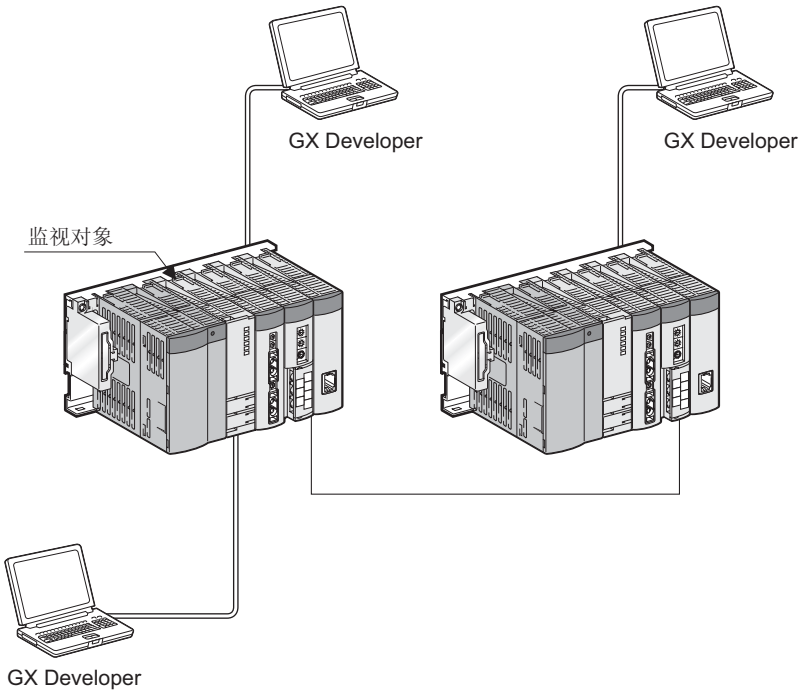


图 6.67 同时监视

根据创建的用户设定的系统区域可以通过多个 GX Developer 进行高速监视。（不需要设定本站用监视文件）

(2) 多个人同时进行监视的设定

多个人同时进行监视时，按如下步骤创建用户设定的系统文件。

- 选择 GX Developer 的 [在线]→[可编程控制器内存格式化]，显示可编程控制器内存格式化窗口。
- 在对象存储器中选择 “程序内存 / 软元件内存”。
- 将格式化形式设定为 “创建用户设定的系统文件”。
- 设定系统区域的步数 (1k 步单位)。

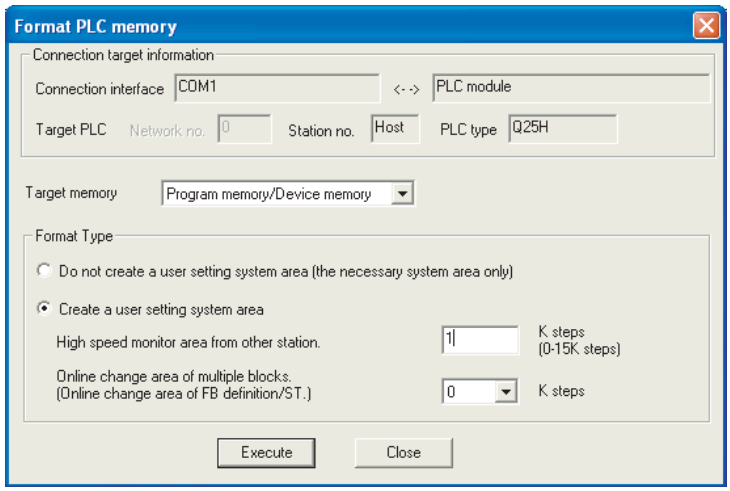


图 6.68 系统区域的设定 (设定 1k 步时)

作为系统区域可以设定的最大的步数如表 6.28 所示。

表 6.28 作为系统区域可以设定的最大的步数

CPU 模块	可以设定的最大步数	其它站监视用的系统区域
基本模式 QCPU	最大 3k 步	最大 3 个
高性能模式 QCPU 过程 CPU 冗余 CPU	最大 15k 步	最大值 15 个

来自其它站的一个监视文件只对应 1k 步。

(3) 注意事项

(a) 关于监视条件的设定*注 6.54*

监视条件的设定只能通过一个场所进行。

(b) 关于是否进行系统区域的设定

即使不创建用户设定的系统区域也可以通过其它站同时进行监视，但是，监视速度将变慢。

系统区域由于设定在程序内存中，因此程序的存储区域减少的量就等于设定的系统区域所占的量。

(c) 通过设定可以进行高速监视的监视器个数

对于一个 CPU 模块，同时可以进行高速监视的监视器个数为 “用户设定的系统区域数 (k 步数)+1”。

例如：创建了 15k 步的用户设定的系统区域时，对于一个 CPU 模块，可以同时从 16 个地点的监视器进行高速监视。

在基本模式 QCPU、通用型 QCPU 中，不能使用设定了监视条件的监视。

基本
注 6.55

6.15.2 多人同时进行运行中写入的功能

(1) 关于多人同时进行运行中写入的功能

是多人同时向一个文件或其它文件进行运行中写入的功能。

由多人同时向一个文件进行运行中写入时，变为 “Relative step No. by pointer（根据指针进行相对的运行中写入）”。

对其它的文件进行运行中写入时，即使不使用 “根据指针进行相对的运行中写入”也可执行。

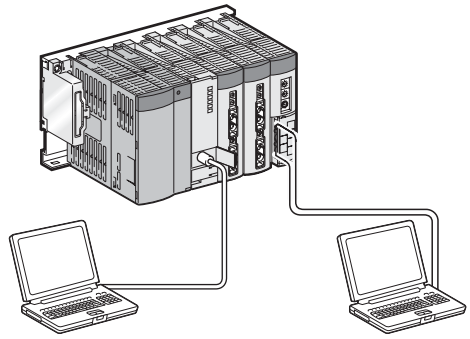


图 6.69 多人同时进行的运行中写入

(2) 对 1 个文件进行运行中写入时的操作步骤

多人同时对一个文件进行运行中写入时，需预先设定运行中写入用的指针并选择 “根据指针进行相对的运行中写入”。

(a) << 公共程序 >> 选项卡的显示

• 选择 GX Developer 中的 [工具] → [选项]，选择 << 公共程序 >> 选项卡。

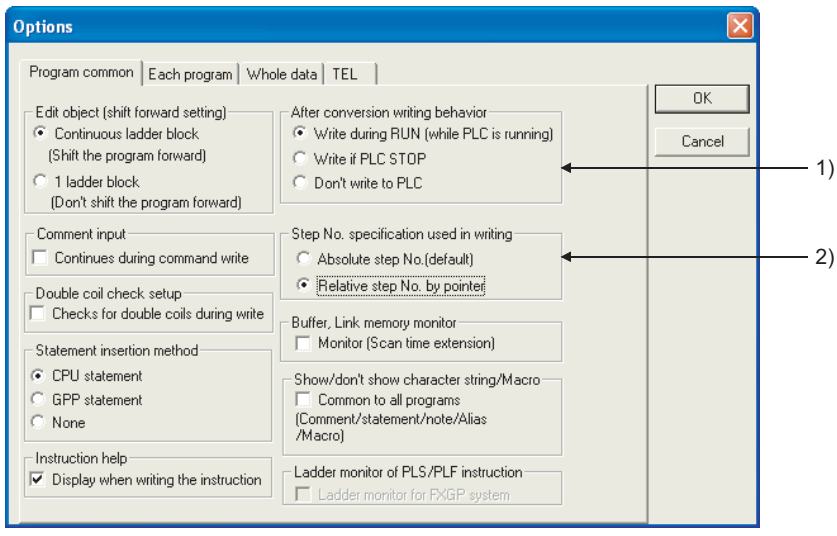


图 6.70 选项画面

基本
注 6.55

在基本模式 QCPU 中，多人同时进行运行中写入时，请确认 CPU 模块以及 GX Developer 的版本。
(附录 4.1)

(b) 运行中写入设定以及运行中写入方式的设定

• 设置运行中写入设定以及运行中写入方式

1)在运行中写入设定中设定 “变换后向可编程控制器进行运行中写入”。

2)在运行中写入方式中选择 “一般的运行中写入”或者 “根据指针进行相对的运行中写入”。

(c) 运行中写入的执行

• 显示指定指针的梯形图，对变更的梯形图进行运行中写入。

(3) 注意事项

运行中写入时的注意事项与 6.12.3 项的运行中写入相同。

6.16 看门狗定时器 (WDT)

(1) 关于看门狗定时器 (WDT)

看门狗定时器是指为检测 CPU 模块的硬件与顺控程序的异常的 CPU 模块内部的时钟。

(2) 看门狗定时器的设定与复位

(a) 看门狗定时器的设定

看门狗定时器的设定时间可以通过可编程控制器参数的可编程控制器 RAS 设定进行变更。

看门狗定时器的默认值被设定为 200ms。

看门狗定时器可以在 10-2000ms 中进行变更。(单位为 10ms)

(b) 看门狗定时器的复位

CPU 模块在执行 END 处理的过程中进行看门狗定时器的复位。

- CPU 模块正常工作并在看门狗的设定值以内执行 END/FEND 指令时，看门狗定时器不会出现时间到的情况。
- CPU 模块出现硬件异常或者执行中断程序 / 恒定周期执行类型程序*注 6.56* 等而延长顺控程序的扫描时间，在看门狗定时器的设定值以内不能执行 END/FEND 指令时，看门狗定时器时间到。

基本 注 6.56

(3) 看门狗定时器时间到的情况

看门狗定时器时间到的情况

1) CPU 模块所有的输出 OFF

2) 前面的 RUN LED 灭灯，ERR LED 闪烁。

3) SM1 将 ON，出错代码 5000、5001(WDT ERROR) 存储在 SD0 中。

基本 注 6.56

在基本模式 QCPU 中，由于没有恒定周期执行类型程序的原因，所以没有根据恒定周期执行类型程序延长扫描时间的情况。

(4) 注意事项

(a) 关于看门狗定时器的误差

看门狗定时器的测量时间的误差在 0-10ms 内。
设定看门狗定时器时，请考虑上述误差值进行设定。

(b) 关于通过 FOR~NEXT 指令反复执行程序时的看门狗定时器复位

看门狗定时器可以根据在顺控程序中执行的 WDT 指令进行复位。
通过 FOR 指令及 NEXT 指令反复执行程序的情况下，看门狗定时器时间到时，通过 WDT 指令复位看门狗定时器。

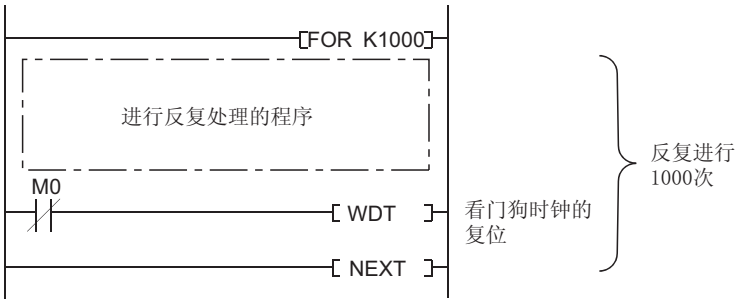


图 6.71 执行 FOR~NEXT 指令时的看门狗定时器的复位

(c) 关于使用 WDT 指令时的扫描时间

即使在顺控程序中进行看门狗定时器复位，扫描时间的值也不复位。

扫描时间为直到 END 指令为止所测量的值。

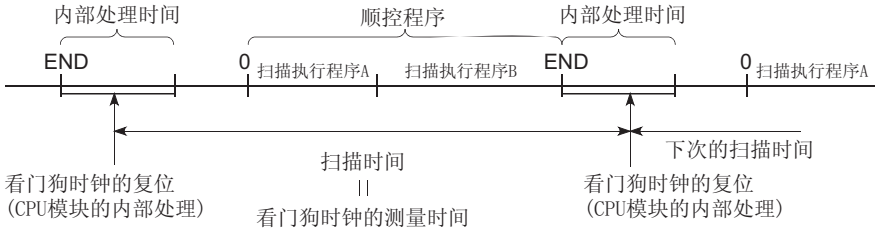


图 6.72 看门狗定时器的复位

要 点

1. 扫描时间是指 CPU 模块从执行顺控程序的运算的步 0 开始到下一次执行同一个文件名的顺控程序的步 0 为止的时间。

   扫描时间在每个扫描周期内不一定相等，由于如下所示的原因而有差异。

   基本 注 6.57

   - 执行 / 不执行正在使用的指令
   - 执行 / 不执行中断程序、恒定周期执行类型程序 (注 6.57)

2. 若想让每个扫描周期都以相同的扫描时间执行的话，请使用恒定扫描功能 (☞本节)。

基本 注 6.57 在基本模式 QCPU 中，不能执行恒定周期执行类型程序。

6.17 自检测功能

(1) 关于自检测功能

自检测功能是 CPU 模块自身进行有无异常检测的功能。
自检测功能具有防止 CPU 模块误动作的功能，同时也达到预防维护的目的。

(2) 自检测的计时

CPU 模块的电源投入时或者 CPU 模块的 RUN/STOP 中发生异常时，通过自诊断功能检测出异常并显示出错后，进行停止 CPU 模块的运算等动作。
但是，根据异常发生状态及执行的指令，有时会发生不能通过自诊断功能检测出异常的现象。应在可编程控制器的外部设置安全电路，以保证当不能通过自诊断停止运算时整个系统的安全运行。

(3) 异常现象的确认

(a) LED 的亮灯

CPU 模块在检测出异常时将点亮 ERR.LED 灯等。

(b) 确认异常内容的存储目标

CPU 模块如果检测出异常，将使特殊继电器 (SM0、SM1)ON 并将异常的内容（出错代码）存储在特殊寄存器 (SD0) 中。
在检测出多个异常时，最新出错的出错代码将被存储在 SD0 中。
在程序上使用特殊继电器、特殊寄存器时，请进行可编程控制器或者机械系统的互锁。

要 点

1. 在冗余 CPU 的情况下，在其它系统中发生的异常内容可以存储在特殊继电器 (SM1610-SM1626) 以及特殊寄存器 (SD1610-SD1636) 中。
   但是，在下述的情况下，不能存储其它系统的异常内容。
   - 其它系统处于电源断开、复位、硬件故障时
   - 发生看门狗定时器溢出时（出错代码：5000，5001）
   - 热备电缆发生异常时（未安装、断线、故障等）
2. 如下所示的表示 CPU 模块的状态的出错中，异常内容不能存储在特殊继电器 (SM0、SM1) 以及特殊寄存器 (SD0 到 SD26) 中。
   另外，ERR.LED 也不亮灯。
   出错内容被存储在故障历史记录中 (☞本节)。
   - 待机系统向控制系统切换时
     （出错信息：CONTROL EXE。出错代码：6200）
   - 控制系统向待机系统切换时
     （出错信息：STANDBY。出错代码：6210）

(4) 故障历史记录的确认

CPU 模块记录最新的出错代码。(☞本节)

通过 GX Developer 的 [检测]→[可编程控制器检测] 可以确认故障的历史记录。

故障的历史记录即使断开可编程控制器的电源，也可以通过电池进行备份。

(5) 检测出异常时的 CPU 模块的动作

(a) 检测出异常时的模式

根据自检测检测出异常时，CPU 模块的动作有如下两种类型的模式。

1) 停止 CPU 模块运算的模式

CPU 模块在检测出异常时将停止运算，并将可编程控制器参数 I/O 分配设定的“出错时的输出模式”中设定为 “清除”(默认值) 的模块的外部输出全部 OFF。(保持软元件内存的输出 (Y)。)

但是，与在 “出错时的输出模式”中设定为 “保持”的模块相关的外部输出将被保持。(保持软元件内存的输出 (Y))。

2) 继续进行 CPU 模块的运算的模式

CPU 模块如果检测出异常时，仅停止执行发生异常的程序 (指令)，继续执行其它的程序。

(b) 可以选择继续进行 / 停止运算的出错
下述的出错可以选择 “继续进行 / 停止”运算。

1) 可编程控制器参数的可编程控制器 RAS 设定中可以选择的出错

- 运算出错 (包括 SFC 程序)
- 扩展指令出错 (用于以后扩展的设定)
- 保险丝熔断
- 模块校验出错
- 智能模块程序执行出错
- 文件存取出错注 6.58
- 内存卡操作出错注 6.58
- 外部电源供给断开 (用于以后扩展的设定)注 6.58

基本 注 6.58

例如，将模块校验出错设定为 “继续进行”时，如果发生出错将根据出错前的 I/O 地址号继续进行运算。
关于出错的详细内容，请参阅自诊断一览。((☞ 本项 (7))

2) 可编程控制器参数的 I/O 分配设定的详细设定中可以选择的出错
- 智能功能模块出错

(6) 出错检查的选择
下面的出错检查通过可编程控制器参数的可编程控制器 RAS 设定可以选择 “进行 / 不进行”。
(参数的默认值被设定为 “进行”所有的检查)
(a) 电池检查
(b) 保险丝断开检查
(c) 模块校验
(d) 变址修饰时的软元件范围检查

基本 注 6.58 在基本模式 QCPU 中，不能设定内存卡文件存取出错、内存卡操作出错、外部电源供给断开。

(7) 自检测一览表

显示 CPU 模块进行自检测的一览表

表 6.29 所示的 “出错信息”栏中的出错信息可以通过 GX Developer 的 [检测] → [可编程控制器检测] 进行确认。

另外，“CPU 模块”栏所示的编号与 CPU 模块的对应情况如下所示。

表 6.29 “CPU 模块”栏所示的编号与 CPU 模块的对应情况

编号	CPU 模块
1)	基本模式 QCPU
2)	高性能模式 QCPU
3)	过程 CPU
4)	冗余 CPU
5)	通用型 QCPU

表 6.30 自我检测一览表

检测内容		出错信息	检测时机	CPU 模块的状态	LED 的状态		CPU 模块				
					RUN	ERR.	1)	2)	3)	4)	5)
硬件异常	CPU 异常	MAIN CPU DOWN	• 常时	停止	灭灯	闪烁	○	○	○	○	○
	不执行 END 指令	END NOT EXECUTE	• 执行 END 指令时	停止	灭灯	闪烁	○	○	○	○	○
	SFC 程序执行出错	SFCP.END ERROR	• 执行 SFC 程序时	停止	灭灯	闪烁	○	×	×	×	×
	RAM 检查	RAM ERROR	• 电源接通以及复位时	停止	灭灯	闪烁	○	○	○	○	○
	热备存储器以及热备硬件异常	TRK.CIR. ERROR	• 电源接通以及复位时 • 运行过程中	停止	灭灯	闪烁	×	×	×	○	×
	运算梯形图检查	OPE.CIRCUIT ERR.	• 电源接通以及复位时 • 执行 END 处理时	停止	灭灯	闪烁	○	○	○	○	○
	保险丝断开 *1	FUSE BREAK OFF	• 执行 END 指令（默认值... 检查）*2	停止 / 继续进行	灭灯 / 亮灯	闪烁 / 亮灯	○	○	○	○	○
	I/O 中断出错	I/O INT.ERROR	• 发生中断时	停止	灭灯	闪烁	○	○	○	○	○

○：进行自检测　×：不进行自检测

*1: 通过 GX Developer 在参数设定中可以变更为 “继续进行”。

*2: 通过 GX Developer 在参数设定中可以设定为 “不进行检查”。另外，在 SM251 “ON” 时不进行检查。

（下页续）

表 6.30 自我检测一览表（续）

检测内容		出错信息	检测时机	CPU 模块的状态	LED 的状态		CPU 模块				
					RUN	ERR.	1)	2)	3)	4)	5)
硬件异常	智能功能模块出错 *1	SP.UNIT DOWN	•电源接通以及复位时 •执行 FROM/TO 指令时 •执行智能功能模块专用指令时 •执行 END 指令时	停止 / 继续运行	灭灯 / 亮灯	闪烁 / 亮灯	○	○	○	○	○
	控制总线出错	CONTROL-BUS ERR.	•接通电源时 •执行 END 处理时 •执行 FROM/TO 指令时 •执行智能功能模块专用指令时 •常时	停止	灭灯	闪烁	○	○	○	○	○
	发生瞬间停止	AC/DC DOWN	•常时	继续运行	亮灯	灭灯	○	○	○	○	○
	多 CPU 间高速总线出错	MULTI-C.BUS ERR.	•电源 ON 以及复位时	停止	灭灯	闪烁	×	×	×	×	○
	冗余基板的电源电压过低	SINGLE PS. DOWN	•常时	继续运行	亮灯	亮灯	×	○*4	○*4	○	×
	冗余电源模块异常	SINGLE PS. ERROR	•常时	继续运行	亮灯	亮灯	×	○*4	○*4	○	×
	电池电量低	BATTERY ERROR	•常时 •（默认 ... 检查）*3	继续运行	亮灯	BAT.ALM LED 亮灯	○	○	○	○	○
	闪存 ROM 出错	FLASH ROM ERROR	•ROM 写入时	继续运行	亮灯	亮灯	×	×	×	×	○
使用异常	模块校验 *1	UNIT VERIFY ERR.	•执行 END 指令时 •（默认... 检查）*2	停止 / 继续运行	灭灯 / 亮灯	闪烁 / 亮灯	○	○	○	○	○
	基板分配出错	BASE LAY ERROR	•电源接通或者复位时	停止	灭灯	闪烁	○	×	×	×	○
	智能功能模块分配出错	SP.UNIT LAY ERR.	•电源接通或者复位时 •由 STOP 切换为 RUN 时	停止	灭灯	闪烁	○	○	○	○	○

○：进行自诊断　×：不进行自诊断

*1: 通过 GX Developer 在参数设定中可以变更为 “继续运行”。
*2: 通过 GX Developer 在参数设定中可以设定为 “不进行检查”。另外，在 SM251 为 ON 时不进行检查。
*3: 通过 GX Developer 在参数设定中可以设定为 “不进行检查”。
*4: 多 CPU 系统配置时，所有的 CPU 模块的序列号的高 5 位均为 “07032”以后时仅在 1 号机的 CPU 模块中可以检测出该错误。

（转下页）

表 6.30 自我检测一览表（续）

检测内容		出错信息	检测时机	CPU 模块的状态	LED 的状态		CPU 模块				
					RUN	ERR.	1)	2)	3)	4)	5)
使用异常	智能程序执行出错 *1	SP.UNIT ERROR	•执行 FROM/TO 指令时	停止 / 继续运行	灭灯 / 亮灯	闪烁 / 亮灯	×	○	○	○	○
	智能功能模块版本出错	SP.UNIT VER.ERR	•电源接通或者复位时	停止	灭灯	闪烁	×	○	○	○	○
	无参数	MISSING PARA.	•电源接通或者复位时	停止	灭灯	闪烁	○	○	○	○	○
	引导出错	BOOT ERROR	•电源接通或者复位时	停止	灭灯	闪烁	○	○	○	○	○
	备份出错	ESTORE ERROR	•电源接通或者复位时	停止	灭灯	闪烁	×	×	×	×	○
	内存卡操作出错 *1	ICM.OPE. ERROR	•内存卡脱落时	停止 / 继续运行	灭灯 / 亮灯	闪烁 / 亮灯	×	○	○	○	○
	文件设置出错	FILE SET ERROR	•电源接通或者复位时	停止	灭灯	闪烁	○	○	○	○	○
	文件存取出错 *1	FILE OPE. ERROR	•执行指令时	停止 / 继续运行	灭灯 / 亮灯	闪烁 / 亮灯	×	○	○	○	○
	不能执行指令	CAN'T EXE.PRG.	•电源接通或者复位时	停止	灭灯	闪烁	○	○	○	○	○
参数异常	参数设定检查	PARAMETER ERROR	•电源接通或者复位时 •从 STOP 切换为 RUN 时	停止	灭灯	闪烁	○	○	○	○	○
	链接参数出错	LINK PARA.ERROR	•电源接通或者复位时 •从 STOP 切换为 RUN 时	停止	灭灯	闪烁	○	○	○	○	○
	SFC 参数出错	SFC PARA.ERROR	•从 STOP 切换为 RUN 时	停止	灭灯	闪烁	○	○	○	○	○
	智能参数出错	SP.PARA. ERROR	•电源接通或者复位时	停止	灭灯	闪烁	○	○	○	○	○
口令异常		REMOTE PASS.ERR	•电源接通或者复位时 •从 STOP 切换为 RUN 时	停止	灭灯	闪烁	○	○	○	○	○
程序异常	指令代码检查	INSTRUCT. CODE ERR	•电源接通或者复位时 •从 STOP 切换为 RUN 时 •执行指令时	停止	灭灯	闪烁	○	○	○	○	○
	无 END 指令	MISSING END INS.	•电源接通或者复位时 •从 STOP 切换为 RUN 时	停止	灭灯	闪烁	○	○	○	○	○

○：进行自诊断　×：不进行自诊断

*1 ： 通过 GX Developer 在参数设定中可以变更为 “继续运行”。

（转下页）

表 6.30 自我检测一览表（续）

检测内容		出错信息	检测时机	CPU 模块的状态	LED 的状态		CPU 模块				
					RUN	ERR.	1)	2)	3)	4)	5)
程序异常	指针设定出错	CAN'T SET(P)	•电源接通或者复位时 •从 STOP 切换为 RUN 时	停止	灭灯	闪烁	○	○	○	○	○
	指针设定出错	CAN'T SET(I)	•电源接通或者复位时 •从 STOP 切换为 RUN 时	停止	灭灯	闪烁	○	○	○	○	○
	运算出错 *1 *2	OPERATION ERROR	•执行指令时	停止 / 继续运行	灭灯 / 亮灯	闪烁 / 亮灯	○	○	○	○	○
	FOR ～ NEXT 指令构成出错	FOR NEXT ERROR	•执行指令时	停止	灭灯	闪烁	○	○	○	○	○
	CALL ～ RET 指令构成出错	CAN'T EXECUTE(P)	•执行指令时	停止	灭灯	闪烁	○	○	○	○	○
	中断程序出错	CAN'T EXECUTE(I)	•执行指令时	停止	灭灯	闪烁	○	○	○	○	○
	不能执行指令	INST.FORMAT ERR.	•执行指令时	停止	灭灯	闪烁	○	○	○	○	○
	多 CPU 间高速总线对应专用指令出错	MULTI COM. ERROR	•执行指令时	停止	灭灯	闪烁	×	×	×	×	○
	SFC 程序构成出错	SFCP.CODE ERROR	•从 STOP 切换为 RUN 时	停止	灭灯	闪烁	×	○	○	○	×
	SFC 块构成出错	CAN'T SET(BL)	•从 STOP 切换为 RUN 时	停止	灭灯	闪烁	○	○	○	○	○
	SFC 步构成出错	CAN'T SET(S)	•从 STOP 切换为 RUN 时	停止	灭灯	闪烁	○	○	○	○	○
	SFC 执行出错	SFC.EXE. ERROR	•从 STOP 切换为 RUN 时	停止	灭灯	闪烁	○	×	×	×	○
	SFC 句法结构出错	SFCP. FORMAT ERR.	•从 STOP 切换为 RUN 时	停止	灭灯	闪烁	○	○	○	○	○
	SFC 运算检查出错 *1	SFCP.OPE. ERROR	•执行指令时	停止 / 继续运行	灭灯 / 亮灯	闪烁 / 亮灯	×	○	○	○	×
	SFC 程序执行出错	SFCP.EXE. ERROR	•从 STOP 切换为 RUN 时	继续运行	亮灯	亮灯	○	○	○	○	×
	SFC 块执行出错	BLOCK EXE.ERROR	•执行指令时	停止	灭灯	闪烁	○	○	○	○	○
	SFC 步执行出错	STEP EXE.ERROR	•执行指令时	停止	灭灯	闪烁	○	○	○	○	○

○：进行自诊断　×：不进行自诊断

*1:　通过 GX Developer 在参数设定中可以变更为 “继续运行”。
*2:　也包含进行了变址修饰时的软元件范围检查情况下的运算出错。

（转下页）

表 6.30 自我检测一览表（续）

检测内容		出错信息	检测时机	CPU 模块的状态	LED 的状态		CPU 模块				
					RUN	ERR.	1)	2)	3)	4)	5)
CPU 异常	运算停滞监视	WDT ERROR	• 常时	停止	灭灯	闪烁	○	○	○	○	○
	超出程序时间	PRG.TIME OVER	• 常时	继续运行	亮灯	亮灯	○	○	○	○	○
冗余系统异常	程序、参数、插杆开关不一致	FILE DIFF.	• 常时 • 电源接通或者复位时 • 安装热备电缆时 • 变更运行模式时	停止	灭灯	闪烁	×	×	×	○	×
	运行状态、键开关不一致	OPE.MODE DIFF.	• 电源接通或者复位时 • 常时	继续运行 / 停止	灭灯 / 亮灯	闪烁 / 亮灯	×	×	×	○	×
	模块安装构成不一致	UNIT LAY. DIFF.	• 常时 • 电源接通或者复位时 • 安装热备电缆时 • 变更运行模式时	停止	灭灯	闪烁	×	×	×	○	×
	内存卡安装状态不一致	CARD TYPE DIFF.	• 电源接通或者复位时	停止	灭灯	闪烁	×	×	×	○	×
	当前模式下功能不能执行	CAN'T EXE. MODE	• 常时	继续运行	亮灯	亮灯	×	×	×	○	×
	两系统文件一致性发生异常 参数有效驱动一致性发生异常	CPU MODE DIFF.	• 电源接通或者复位时 • 执行 END 指令时 • 安装热备电缆时	停止	灭灯	闪烁	×	×	×	○	×
	热备数据通讯出错	TRK.TRANS. ERR.	• 常时	继续运行	亮灯	亮灯	×	×	×	○	×
	超出热备容量出错	TRK.SIZE ERROR	• 执行 END 指令时	继续运行	亮灯	亮灯	×	×	×	○	×
	热备电缆异常、热备传送用硬件故障	TRK.CABLE ERR.	• 电源接通或者复位时	停止	灭灯	闪烁	×	×	×	○	×
	热备电缆未连接・故障、热备传送用硬件故障	TRK. DISCONNECT	• 常时	继续运行	亮灯	亮灯	×	×	×	○	×
	热备传送初始化出错	TRK.INIT. ERROR	• 电源接通或者复位时	停止	灭灯	闪烁	×	×	×	○	×

○：进行自诊断　×：不进行自诊断

（转下页）

表 6.30 自我检测一览表（续）

检测内容		出错信息	检测时机	CPU 模块的状态	LED 的状态		CPU 模块				
					RUN	ERR.	1)	2)	3)	4)	5)
冗余系统异常	发生由待机系统向控制系统切换 *4	CONTROL EXE.	• 常时	继续运行	亮灯	灭灯	×	×	×	○	×
	发生由控制系统向待机系统切换 *4	STANDBY	• 常时	继续运行	亮灯	灭灯	×	×	×	○	×
	系统切换出错	CAN'T SWITCH	• 执行系统切换时	继续运行	亮灯	亮灯	×	×	×	○	×
	待机系统未启动 / 停止出错	STANDBY SYS.DOWN	• 常时	继续运行	亮灯	亮灯	×	×	×	○	×
	控制系统未启动 / 停止出错	CONTROL SYS.DOWN	• 常时	停止	灭灯	闪烁	×	×	×	○	×
	实施程序内存清除	PRG.MEM. CLEAR	• 执行程序内存复制功能时	停止	灭灯	闪烁	×	×	×	○	×
	实施存储器复制功能	MEM.COPY EXE.	• 执行存储器复制功能时	继续进行	亮灯	亮灯	×	×	×	○	×
	热备设定参数出错	TRK.PARA. ERROR	• 电源接通或者复位时	停止	灭灯	闪烁	×	×	×	○	×
	不可以构成多 CPU 系统	CPU LAY ERROR	• 电源接通或者复位时	停止	灭灯	闪烁	×	×	×	○	×
多 CPU 系统异常	其它号机严重异常	MULTI CPU DOWN	• 常时 • 电源接通或者复位时	停止	灭灯	闪烁	○	○	○	×	○
	多 CPU 执行出错	MULTI EXE.ERROR	• 电源接通或者复位时	停止	灭灯	闪烁	○	×	×	×	○
	多 CPU 一致性出错	CPU LAY.ERROR	• 电源接通或者复位时	停止	灭灯	闪烁	○	○	○	×	○
	其它号机轻度异常	MULTI CPU ERROR	• 常时	继续运行	亮灯	亮灯	○	○	○	×	○
文件诊断确认		INCORRECT FILE	• 电源接通或者复位时 • 从 STOP 切换为 RUN 时 • 可编程控制器写入时	停止	灭灯	灭灯	×	×	×	×	○
报警器检查		F****	• 执行指令时	继续运行	亮灯	USER LED 亮灯	×	○	○	○	○
CHK 指令检查		<CHK>ERR***-***	• 执行指令时	继续运行	亮灯	USER LED 亮灯	×	○	○	○	×
引导 OK		BOOT OK	• 电源接通或者复位时	停止	灭灯	闪烁	×	○	○	○	×

○：进行自诊断　×：不进行自诊断

*4 ： 由于是表示 CPU 模块的状态的诊断，因此在 GX Developer 的可编程控制器诊断画面的 “当前出错”栏中不能显示出错信息。
此出错只显示在出错历史记录栏中。

要 点

冗余 CPU 情况下，其它系统发生的异常内容被存储在特殊继电器 (SM1610 ～ 1626) 以及特殊寄存器 (SD1610 ～ 1636) 中。
但是，在下述情况下，其它系统的异常内容不能被存储。

- 其它系统处于电源 OFF、复位、硬件故障中时
- 发生看门狗定时器出错时（出错代码 :5000、5001）
- 热备电缆异常（未安装、断线、故障等）

6.17.1 根据发生出错进行的中断

基本 注 6.59 通用 注 6.59

CPU 模块根据发生出错可以对出错对象执行中断指针的中断程序。

(1) 根据可编程控制器 RAS 设定中可设定为继续进行 / 停止的出错而产生的中断

对于在可编程控制器参数的可编程控制器 RAS 设定中可以对运算设定 “继续运行 / 停止”的出错，只对设定了 “继续进行”的出错执行中断。

对于设定为 “停止”的出错将执行停止出错全部的中断程序 (I32)。

高性能 注 6.60 过程 注 6.60

(2) 与中断指针相对应的出错*注 6.60*

与中断指针相对应的出错如图 6.73 所示。

中断指针	对应的出错信息
I32	全部停止出错 *1
I33	SINGLE PS.DOWN*2
I34	UNIT VERIFY ERR. FUSE BREAK OFF SP.UNIT ERROR MULTI CPU ERROR
I35	OPERATION ERROR SFCP OPE.ERROR SFCP EXE.ERROR
I36	ICM.OPE.ERROR FILE OPE.ERROR
I37	空闲
I38	PRG.TIME OVER
I39	CHK指令 检测出报警
I40	CAN'T SWITCH
I41	STANDBY
I42～48	空闲

发生出错时运行模式为继续运行的出错，或者是可选择“停止/继续运行”的出错中设定为“继续运行”的出错。

*1: 发生下述严重出错时，不能执行 I32 的中断程序。
- MAIN CPU DOWN
- END NOT EXECUTE
- RAM ERROR
- OPE CIRCUIT ERR.

*2: 在高性能模式 QCPU 或者过程 CPU 中进行了多 CPU 系统配置时，只有在 1 号机的 CPU 模块中才可以使用。

图 6.73 根据出错发生中断的中断指针一览表

基本 注 6.59 通用 注 6.59

在基本模式 QCPU、通用型 QCPU 中，由于没有出错发生时的中断指针，因此不能通过 CPU 模块自身发生的出错进行中断。

高性能 注 6.60 过程 注 6.60

在高性能模式 QCPU、过程 CPU 中，不能使用中断指针 I33、I40、I41。

此外，I33 只对应于序列号的高 5 位为 “07032”或以后的 CPU。

(3) 注意事项

(a) 使用中断指针 I41 的中断程序时的注意事项

中断指针 I41 是指由控制系统向待机系统切换时的中断指针。
由于控制系统切换到待机系统后的新待机系统（由于系统切换控制系统变为待机系统的系统）中执行 I41 的中断程序的原因，请注意下述要点。

1) 特殊继电器的控制系统判别标志 - 待机系统判别标志 (SM1515、SM1516)
控制系统判别标志 - 待机系统判别标志变为待机系统 (SM1515：OFF、SM1516：ON)。

2) 变更热备对象的软元件的情况
为在 I41 中断程序中变更热备对象的软元件而进行编程时，新待机系统的软元件将被新控制系统（系统切换中由待机系统切换为控制系统的系统）的热备传送所覆盖。

在 I41 的中断程序中变更热备传送的软元件时，请进行如下操作。

- 在冗余参数的热备设定中 (☞ 本节 (2))，将进行变更的软元件从热备传送的对象中剔除。
- 对编程进行如下变更：将热备对象的软元件的内容移动到其它的软元件中。

3) 安装在主基板上的模块以及网络模块的输出
由于 I41 的中断程序是在新待机系统中被执行，因此不进行下述所示的输出。

- 向安装在主基板上的模块的输出 (Y)
- 向 MELSENET/H 远程 I/O 网络的输出 (Y)
- 向 MELSENET/H 远程 I/O 网络、MELSENET/H 可编程控制器间网络的其它站的链接继电器 (B)、链接寄存器 (W) 的发信。

要 点

中断指针 I32-41 在电源启动 /CPU 模块的复位时，进入执行禁止状态。
使用 I32-41 时，请使用 IMASK 指令以及 EI 指令进入执行允许状态。
关于 IMASK 指令以及 EI 指令请参照下述手册。
☞QCPU(Q 模式)/QnACPU 编程手册（公共指令篇）

6.17.2 发生出错的 LED 显示

出错发生时，CPU 模块前面的 LED 亮灯 / 闪烁。(☞ 本节)

6.17.3 出错的解除

CPU 模块在程序中可以进行出错解除的操作只局限于程序运算继续进行的出错。

(1) 出错解除的顺序

出错解除按照下述顺序进行。

- 清除出错的原因。
- 向特殊寄存器 SD50 内存储解除的出错代码。
- 将特殊继电器 SM50 由 OFF→ON 转换。
- 对象出错被解除。

(a) 发生多个出错时的出错解除顺序

如果解除最后发生的出错（存储在特殊寄存器 SD0 中的出错），由于存有出错信息的特殊继电器 / 寄存器 (SM0,SM1,SM5,SM16,SD0-26) 的内容被清除的原因，没有解除的出错的信息不能从特殊继电器 / 寄存器中获得。

对于没有解除的出错，请可以通过故障历史记录（☞ 本节）获得过去发生的出错并解除。

(2) 出错解除后的状态

通过出错解除使 CPU 模块回复原来状态时，与出错相关的特殊继电器、特殊寄存器以及 LED 会返回到出错前的状态。

在进行出错解除之后再次发生同样的出错时，将再次被登录到故障历史记录中。

(3) 报警器的解除

对多个检测出的报警器的解除，只有最初检测出的 F 号被解除。(☞9.2.5 项)

要 点

1. 将解除的出错代码存储到 SD50 中并进行了出错解除时，低 2 位的代码编号将被忽略。

   [例]

   发生了出错代码为 2100 及 2101 的错误时，如果将 2100 存储到 SD50 中并进行解除，则 2101 也将被解除。

   发生了出错代码为 2100 及 2111 的错误时，如果将 2100 存储到 SD50 中并进行解除，则 2111 也将被解除。

2. 由于 CPU 模块以外的原因发生出错时，即使通过特殊继电器 SM50 以及特殊寄存器 SD50 进行出错解除也不能清除出错原因。

   [例]

   “SP.UNIT.DOWN”由于是在 Q 总线上发生的出错，即使通过特殊继电器 SM50 以及特殊寄存器 SD50 进行出错解除也不能清除出错原因。

   请参阅 QCPU 用户手册（硬件设计 / 维护点检篇）中记载的出错代码一览表并清除出错原因。

6.18 故障历史记录

CPU模块根据自检测功能在检测出的结果上附有检测出的时间并可以作为故障历史记录存储在存储器中。
故障历史记录可以通过GX Developer的[检测]→[可编程控制器检测]确认。

要 点

检测时间由于使用的是CPU模块的内部时钟时间，因此在使用CPU模块时，请务必设定最初的正确的时间。(☞本节)

6.18.1 基本模式QCPU

(1) 存储区域
最新的16个故障被存储在锁存的基本模式QCPU的故障历史记录存储器中。

(2) 存储数据
可编程控制器的电源处于接通状态下同一个出错多次发生时，只向故障历史记录存储器中进行一次数据的存储。

(3) 故障历史记录清除的方法
故障历史记录清除通过GX Developer[检测]→[可编程控制器检测]的
“Clear log”(历史记录清除)按钮进行清除。
如果进行故障历史记录清除，基本模式QCPU的故障历史记录存储器的数据将全部被清除。

6.18.2 高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU

(1) 存储区域

根据 CPU 模块的不同，存储区域也有所不同。

(a) 高性能模式 QCPU、过程 CPU、冗余 CPU 的情况

最新的 16 个故障被存储在锁存的 CPU 的故障历史记录存储器中。
在存储 17 个以上的情况下，根据可编程控制器参数的可编程控制器 RAS 设定可以向内存卡内的文件进行存储。

(b) 通用型 QCPU 的情况

将可存储的部分存储到 CPU 模块的故障历史记录存储内存中。

表 6.31 故障历史记录文件的存储区域

项目	高性能模式 QCPU、过程 CPU、冗余 CPU	通用型 QCPU
存储区域	内存以及设置的存储卡中的文件	内存
可存储数量	最多 100 个（可更改）*1	最多 100 个 *1

*1 ： 超出可存储数量时，删除最旧的历史记录，存储最新的历史记录。

(2) 存储的数据

在高性能模式 QCPU、过程 CPU、冗余 CPU 中，进行如下操作的参数与内存卡的历史记录数不一致的情况下，在清除了内存卡的历史记录文件的内容之后，将 CPU 模块的故障历史记录存储内存中的 16 个数据传送到历史记录文件中。

- 参数的历史记录文件的历史记录数据中途发生变更的情况
- 安装了与参数设定的历史记录数据不一致的内存卡的情况

要 点

通过参数设定的故障历史记录文件即使不存在于内存卡内，CPU 模块也不会发生出错。
此时，CPU 模块只将发生的故障存储到 CPU 模块的故障历史记录存储器中。

(3) 故障历史记录的清除方法

故障历史记录存储器 / 故障历史记录文件的清除通过 GX Developer[检测]→[可编程控制器检测] 的 “ Clear log ”（历史记录清除）按钮进行清除。
如果进行故障历史记录清除，CPU 模块的故障历史记录存储器的数据、内存卡的故障历史记录文件的数据将全部被清除。

6.19 系统保护

CPU 模块具有多个防止来自设计者以外的第三者的 GX Developer、串行口通讯等对程序进行变更的保护功能（系统保护）。

表 6.32 系统保护的种类

保护对象	保护有效的文件	保护内容	方法	有效时机	参考
CPU 全体 *注 6.61*	所有的文件	从整体上禁止 GX Develop 等来自外部的对 CPU 模块的写入 / 控制指示	使 CPU 模块的系统设定开关 SW1ON。	常时	对软元件也有效
内存卡单位 *注 6.61*	所有的文件	对内存卡进行写保护以及禁止写入	使内存卡的写保护开关 ON	常时	—
文件单位	程序 软元件注释 软元件初始值	对每个文件的属性进行如下的变更。 禁止读写 禁止写入	口令登录中进行文件属性的变更。	常时	—

基本 注 6.61

基本 注 6.61

＊在上表中，控制指示、读写以及写入的内容如下所示。

项目	内容
控制指示	通过远程操作的 CPU 模块的动作指示 （远程 RUN、远程 STOP 等）
读写	程序的读出、写入等操作
写入	与程序的写入、测试等写入处理相关的操作

要 点

即使将 CPU 模块本身的插杆开关 SW1ON，使其处于系统保护的状态，也可执行通过可编程控制器参数、CPU 模块的插杆开关设定的下述功能。*注 6.62*

- 从标准 ROM、内存卡进行引导
- 向标准 ROM 进行自动写入

基本 注 6.62　通用 注 6.62

基本 注 6.61

在基本模式 QCPU 中，不能进行 CPU 全体以及内存卡单位的系统保护。

基本 注 6.62　通用 注 6.62

在基本模式 QCPU、通用型 QCPU 中，由于没有通过插杆开关进行系统保护的功能，因此，无需理会要点所示的限制。

6.19.1 口令登录

口令登录可以禁止通过 GX Developer 进行的读出以及改写 CPU 模块内的程序、软元件注释等的数据。

(1) 口令的有效范围

基本

注 6.63

禁止读出以及改写的范围是以指定的存储器（程序内存 / 标准 ROM/ 内存卡）的程序文件、软元件注释文件、软元件初始值文件为对象。*注 6.63*

(2) 通过口令禁止可以禁止的操作

登录的内容有以下两种类型。

- 不能进行文件名的写入 / 读取。
- 不能写入文件（可以读出）

在进行口令登录的情况下，如果不输入相同的口令，则不能通过 GX Developer 进行文件操作。

基本

注 6.63

基本模式 QCPU 的口令的有效范围是以程序内存的程序文件、软元件注释文件、软元件初始值为对象的。
标准 ROM、内存卡不为有效对象。

(3) 口令的登录

口令的登录在 GX Developer 的口令登录 / 变更画面中进行。

通过 GX Developer 的 [在线]→[口令登录] 或者可编程控制器写入画面的 Password setup (口令设定) 按钮来显示口令登录 / 变更画面。

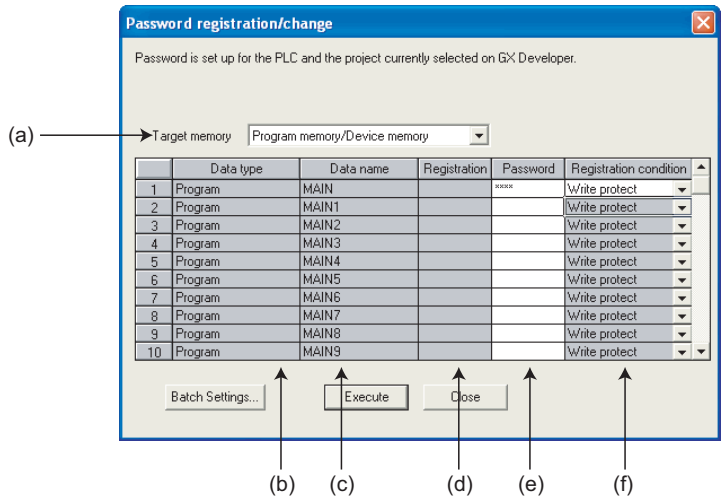


图 6.74 口令登录 / 变更画面

说明各项的内容。

(a) 对象内存

指定存储登录口令文件的存储器。

(b) 类别

显示存储在对象内存中文件的类别。

(c) 数据名称

显示存储在对象内存中的文件名。

(d) 登录状态

口令已被登录时显示 “*”。

(e) 口令

设定新登录的口令或者当前设定的口令。

如果指定口令，可以设定登录条件。

(f) 登录条件

1) 禁止写入

禁止向指定了口令的文件进行写入。(可以读出)

2) 禁止读出 / 写入

禁止向指定了口令的文件进行写入 / 读出。

3) 取消

取消设定的口令。

(在口令栏中设定当前设定的口令。)

(4) 注意事项

登录到文件的口令不能从文件中读出。
如果忘记了登录的口令，将不能进行下述之外的文件操作。
请将登录的口令记录在书本上并认真保管。

(a) 基本模式 QCPU 的情况

- 程序内存 ： 格式化可编程控制器内存
- 标准 ROM ： ROM 化程序内存

(b) 高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 的情况

- 程序内存 / 内存卡 ： 格式化可编程控制器内存
- 标准 ROM ： 批量写入

6.19.2 远程口令

基本 注 6.64

高性能 注 6.64

(1) 关于远程口令

远程口令是指防止远处的用户向 CPU 模块内进行不正常的存取的功能。
如果设定远程口令，有远处的用户向 CPU 模块内进行存取时，将进行远程口令的检查。

(2) 从远程口令设定到反映的流程

远程口令在 GX Developer 中设定并写入 CPU 模块中。(☞ 本项 (6))
远程口令在可编程控制器的电源断开到接通或者 CPU 模块进行复位解除时，远程口令被传送到可以设定的模块内。(☞ 本项 (3))

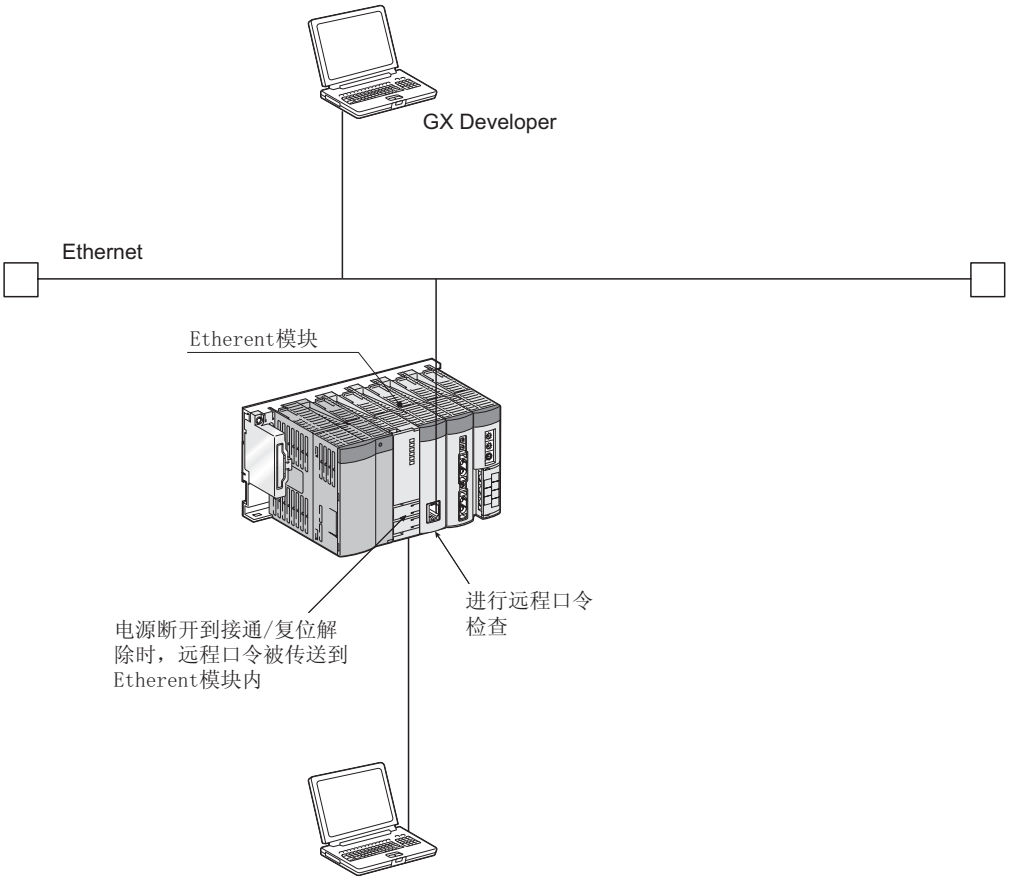


图 6.75 远程口令的概要

基本 注 6.64　高性能 注 6.64

在基本模式 QCPU、高性能模式 QCPU 中使用远程口令时，请确认 CPU 模块以及 GX Developer 的版本。(☞ 附录 4.1，附录 4.2)

(3) 可以设定远程口令的模块

可以设定远程口令的模块如下所示。

- 串行口通讯模块
- Ethernet 模块
- 调制解调器接口模块

(4) 远程口令的锁定 / 非锁定处理

经由调制解调器的串行口通讯模块或者经由 Etherent 的 Etherent 模块进行远程口令的非锁定。

远程口令一致的情况下，可以向 CPU 模块内进行存取。

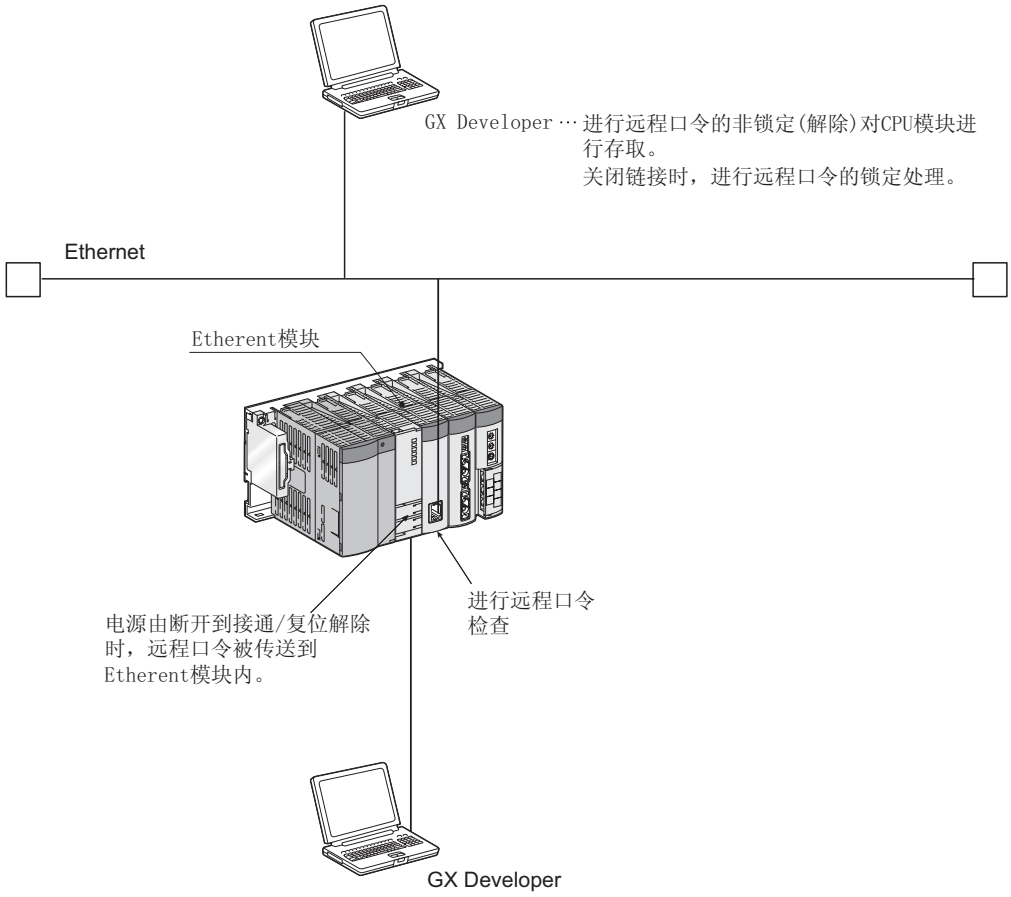


图 6.76 Etherent 模块进行口令的锁定 / 非锁定处理的概要

(5) 远程口令的设定个数

远程口令的设定个数根据使用的 GX Developer 版本不同而不同。
根据 GX Developer 的版本远程口令的设定个数如表 6.33 所示。

表 6.33 根据 GX Developer 的版本的远程口令的设定个数

GX Developer 的版本	模块名称	最大设定个数
版本 6 版本 7	Etherent 模块	4 个
	串行口通讯模块	4 个
	调制解调器接口模块	
版本 8 以后的产品	Etherent 模块	4 个
	串行口通讯模块	8 个
	调制解调器接口模块	

要 点

上表的最大设定个数是指根据 GX Developer 版本的远程口令的设定个数
并不是使用 CPU 模块的系统中的最大安装个数。
关于系统中的最大安装个数请参照下述手册。
☞QCPU 用户手册（ 硬件设计 / 维护点检篇 ）

(6) 远程口令设定、变更、取消的顺序

(a) 远程口令的设定

• 在 GX Developer 的工程数据一览表中选择 [参数]→[远程口令]，显示远程口令设定画面。
进行远程口令的设定

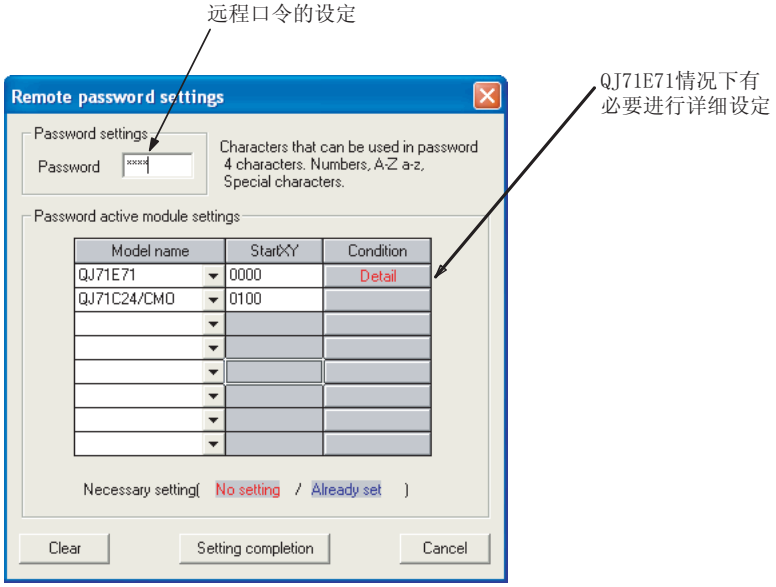


图 6.77 远程口令的设定画面

表 6.34 远程口令设定画面的设定项目

项目			设定内容	设定范围 / 选择项
口令设定			远程口令输入	4 号字以内（英、数字、特殊符号）
口令有效模块设定		型号	选择型号	QJ71E71/QJ71C24/QJ71CMO
		起始 X/Y	设定模块的起始地址	• 基本模式 QCPU 使用时：0000_H~$03E0_H$ • 高性能模式 QCPU、过程 QCPU、冗余 CPU 使用时：0000_H~$0FE0_H$
详细设定			—	—
	用户用链接号有效的设定		设定用户用链接号	链接号 1~ 链接号 16
	系统用链接有效设定	自动打开 UPD 端口	检查远程口令有效端口	—
		FTP 通讯端口 (TCP/IP)		
		GX Developer 通讯端口 (TCP/IP)		
		GX Developer 通讯端口 (UDP/IP)		
		HTTP 端口		

冗余 注 6.65

- 将 GX Developer 与 CPU 模块相连接。
  将设置的远程口令写入 CPU 模块。
  构成多 CPU 系统时，将远程密码写入要设置的模块的管理 CPU。注 6.65
- 将可编程控制器的电源 OFF→ON，或者使 CPU 模块复位，远程口令将对模块有效。

要 点

基本 注 6.66

1. 设置了远程口令的情况下，请将参数存储到程序内存中（0 驱动）。
   将参数存储到程序存储区（0 驱动）以外的区域时，远程口令功能将不能正常动作。注 6.66 注 6.67

通用 UD 注 6.67

2. 进行引导运行时，请将参数文件存储在标准 ROM 或者存储卡中，在 GX Developer 的引导文件设定中进行如下设定：参数文件被发送到程序内存中。注 6.66 注 6.67
   此时，请将插杆开关的参数有效驱动设为存储了参数文件的标准 ROM 或者存储卡。（将插杆开关的参数有效驱动设为程序内存时，引导运行将不能正常动作。）

(b) 远程口令的变更

- 在 GX Developer 的工程数据一览中，选择（参数）→（远程口令），将显示远程口令设置画面。
- 变更口令后，将已变更的口令写入 CPU 模块。

(c) 远程口令的删除

- 在 GX Developer 的工程数据一览中，选择（参数）→（远程口令），将显示远程口令设置画面。
- 点击 Clear （清除）按钮，所设置的远程口令将被删除。
- 利用 GX Developer 写入远程口令。

冗余 注 6.65

在冗余 CPU 中不能使用多 CPU 系统。

基本 注 6.66

在基本模式 QCPU 中，由于参数必须被存储到程序内存中，因此以上的要点内容不适用。

通用 UD 注 6.67

在通用型 QCPU 中，如果参数被存储到有效驱动中，远程口令将有效，因此无需理会要点的内容。

备 注

关于远程口令功能的详细内容，请参照以下手册。

- 使用串行口通讯模块时
  ☞Q 对应串行口通讯模块用户手册（应用篇）
- 使用 Ethernet 模块时
  ☞Q 对应 Ethernet 接口模块用户手册（基本篇）
- 使用 Modem 接口模块时
  ☞QJ71CMO 型 Modem 接口模块用户手册（详细篇）

6.20 利用 GX Developer 显示 CPU 模块的系统

将 CPU 模块与 GX Developer 连接，通过系统监控器（参照图 6.78，图 6.79）可确认以下项目。

- 安装状态
- 动作状态
- 模块详细信息
- 产品信息

在本节中，通过 DX Developer Version6 以后的系统监控器画面进行说明。

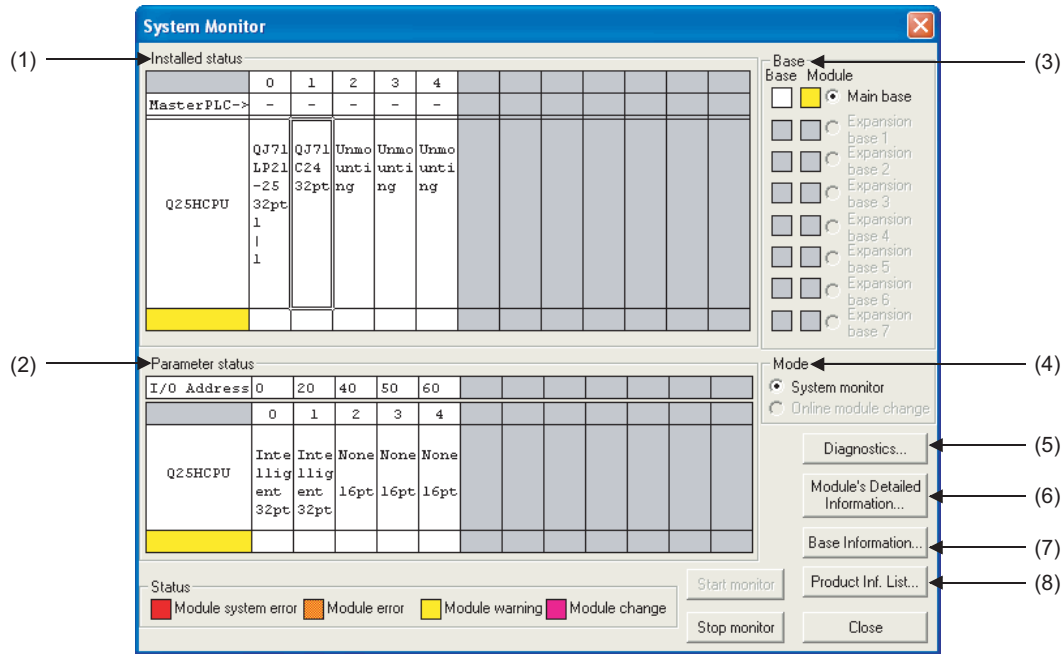


图 6.78 系统监控器画面（冗余 CPU 除外）

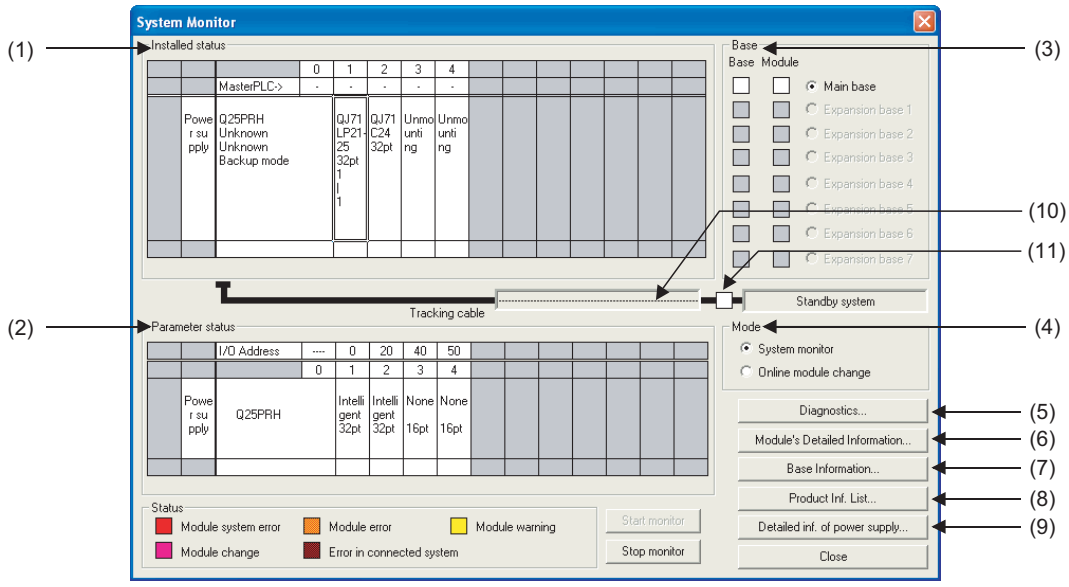


图 6.79 系统监控器画面（冗余 CPU）

基本 注 6.68

(1) 安装状态

可确认所选定的基板上安装的模块的管理 CPU注 6.68、模块型号、点数。
对于没有安装模块的插槽，将显示“未安装”。
对于在 PC 参数的 I/O 分配设定中设为“空”的插槽，即使安装了模块，也不会显示模块型号。

基本 注 6.69

使用冗余基板时，也将显示电源模块的安装状态。注 6.69
使用冗余 CPU 时，也将显示冗余 CPU 的状态。

(2) 动作状态

可确认所选定的基板的各插槽的 I/O 地址号、模块类别、点数。
动作状态显示为空 0 点、分配错误时，说明可编程控制器参数的 I/O 分配与安装状态不一致。

基本 注 6.69

请使可编程控制器参数的 I/O 分配与安装状态一致。

使用冗余基板时，也将显示电源模块的安装状态。注 6.69

(3) 基板

可确认所使用的基板与所安装的模块的状态。
只要存在一个异常模块，模块栏中便会变为该异常模块的状态色。

基本 注 6.70　高性能 注 6.70　通用 UD 注 6.70

(4) 模式注 6.70

用于进行在线模块更换时。
关于在线模块更换，请参照以下手册。
☞QCPU 用户手册（硬件设计 / 维护点检篇）
☞在线模块更换对应的模块的手册

(5) 检测 注 6.70

用于确认所选定的模块状态以及错误。

基本 注 6.68

使用基本模式 QCPU 时，不能显示模块的管理 CPU。

基本 注 6.69

使用基本模式 QCPU 时，不能显示电源模块的安装状态。

基本 注 6.70　高性能 注 6.70　通用 UD 注 6.70

基本模式 QCPU、高性能模式 QCPU、通用型 QCPU 不能使用在线模块更换，因此不能进行模式选择。

(6) 模块详细信息

用于确认所选定的模块的详细信息。

关于智能模块的详细信息，请参照各智能模块的手册。

(7) 基本信息

基本信息中，可确认“整体信息”与“基本信息”。

(a) 整体信息

可确认所使用的基板数、基板上安装的模块数。

(b) 基本信息

可确认所选定基板的基板的名称、插槽数、基板类型、基板上安装的模块数。

(8) 产品信息一览

可以确认所安装的 CPU 模块、I/O 模块、智能模块的个别信息（类别、系列、型号、点数、起始 I/O、管理 CPU、系列 No.、功能版本、生产编号 *1）。

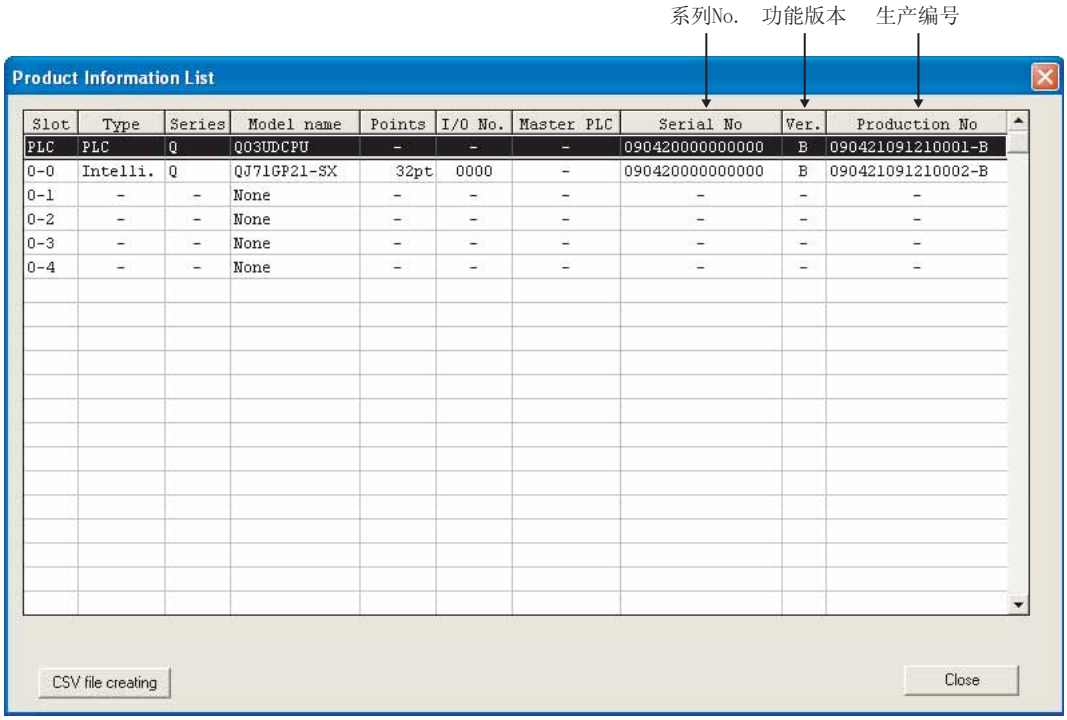


图 6.80 产品信息一览

*1: 只有使用通用型 QCPU 时才可以显示生产编号。
在基本模式 QCPU、高性能模式 QCPU、过程 CPU、冗余 CPU 中，由于不兼容生产编号，因此显示为“0000000000000000”。

(9) 电源详细信息

将显示电源模块的 ON/OFF 状态、有无异常、瞬间掉电次数。

使用冗余基板以及支持电源详细信息的电源模块时，可使用电源详细信息。

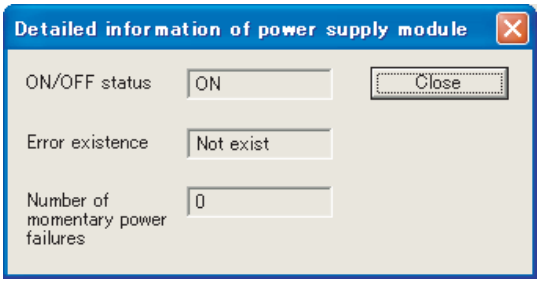


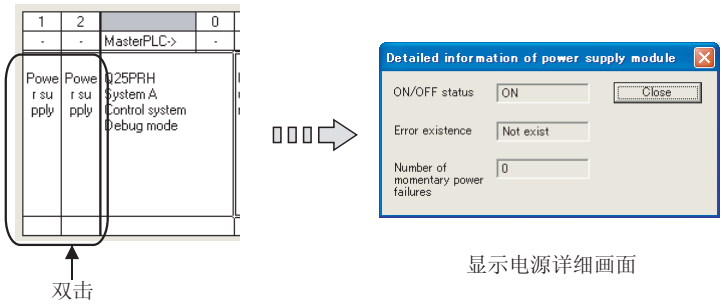
图 6.81 电源详细画面

表 6.35 电源详细画面的显示内容

项目	内容
ON/OFF 状态	显示至冗余电源模块的输入电源状态。
有无异常	显示有无冗余电源模块的故障（异常）。
瞬时掉电次数	• 选择冗余主基板的冗余电源模块时 显示安装在冗余基板的冗余电源的瞬时掉电次数。 （显示范围 :0 ～ 65535） • 选择冗余扩展基板的冗余电源模块时 瞬时掉电次数显示为 “-”。 瞬时掉电次数不能被计数。

要 点

1. 多 CPU 系统配置时，只有将 GX Developer 与 1 号机相连接时才可以显示电源详细信息。
2. 只有在所有的 CPU 模块的序列号的高 5 位均为 07032 或以后的情况下，才可以显示多 CPU 系统配置时的电源详细信息。
3. 通过鼠标双击安装状态的电源模块部分，也可以显示电源详细画面。



显示电源详细画面

(10)内存复制状态

将显示从控制系统到待机系统的内存复制功能的执行状态。

- 通常运行时

- 从控制系统到待机系统的内存复制时

- 热备电缆异常时

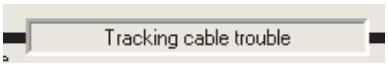


图 6.82 内存复制状态

(11)其它系统状态

将显示其它系统的状态。

- 其它系统正常时

- 其它系统发生异常时

图 6.83 其它系统状态

但是，冗余 CPU 为调试模式时，即使其它系统发生异常时也显示为正常。（不进行异常发生的显示。）

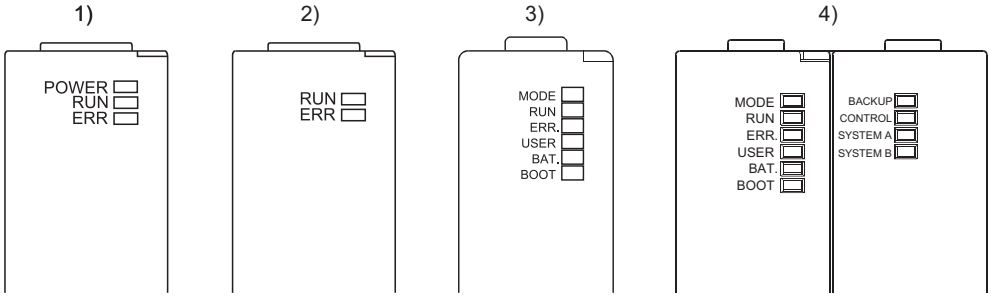
备 注

关于 GX Developer 的系统监视的详细内容，请参照以下手册。

☞GX Developer 操作手册

6.21 LED 的显示

CPU 模块的前面带有显示 CPU 模块动作状态的 LED。



1)：基本模式QCPU(Q00JCPU)
2)：基本模式QCPU(Q00CPU、Q01CPU)
3)：高性能模式QCPU、过程CPU、通用型QCPU
4)：冗余CPU

图 6.84 CPU 模块前面的 LED

备 注

关于各 LED 的显示内容，请参照以下手册。

☞QCPU 用户手册（ 硬件设计 / 维护点检篇 ）

6.21.1 LED 的灭灯方法

(1) LED 的灭灯方法

(a) 基本模式 CPU

对于亮灯的 ERR.LED，消除错误原因后，通过操作特殊继电器 SM50、特殊寄存器 SD50 解除错误后即可使其灭灯。
（复位操作除外。）

(b) 高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 的情况

通过以下操作可使亮灯的 LED 灭灯。
（复位操作除外。）

表 6.36 LED 的灭灯方法

灭灯方法	对象 LED			
	ERR.	USER	BAT.	BOOT
消除出错原因后，执行 LEDR 指令。	○	○	○	×
消除出错原因后，操作特殊继电器 SM50、特殊寄存器 SD50，消除错误。 （仅限于继续运行错误。）*1	○	○	○	×
操作特殊继电器 SM202、特殊寄存器 SD202，使 LED 灭灯。*1	×	○	×	○

○：有效　×：无效

*1: 关于特殊继电器、特殊寄存器的内容
SM50 ... OFF→ON 时，将解除 SD50 中存储的出错代码。
SD50 ... 存储要解除错误的出错代码。
关于出错代码，请参照以下手册。
☞ QCPU 用户手册（硬件设计 / 维护点检篇）
SM202 .. OFF→ON 时，与 SD202 的各个位相对应的 LED 将灭灯。
SD202 .. 指定要灭灯的 LED。

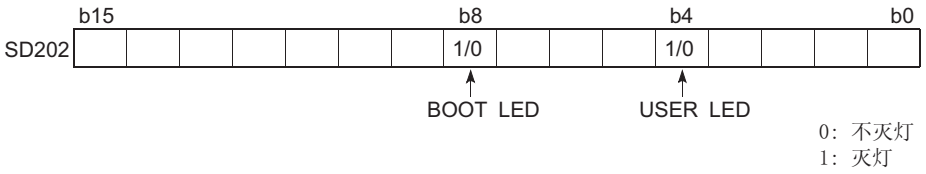


图 6.85 特殊寄存器 SD202 的位构成

使各 LED 灭灯的设定如下。（全部为 16 进制）

- BOOT LED、USER LED 同时灭灯时　:SD202=110_H
- 仅为 BOOT LED 灭灯时　:SD202=100_H
- 仅为 USER LED 灭灯时　:SD202=10_H

(2) 使 ERR.LED、USER LED、BAT.LED 不亮灯的方法

ERR.LED、USER LED、BAT.LED 具有一定的优先顺序。（☞本项）
从优先顺序中删除各 LED 的对象原因编号之后，即使发生了原因编号的错误，LED 也不会亮灯。

通用 UD 注 6.71

6.21.2 优先顺序的设定

以下将要说明的是，发生异常时，显示器中显示的错误信息的优先顺序的设定。

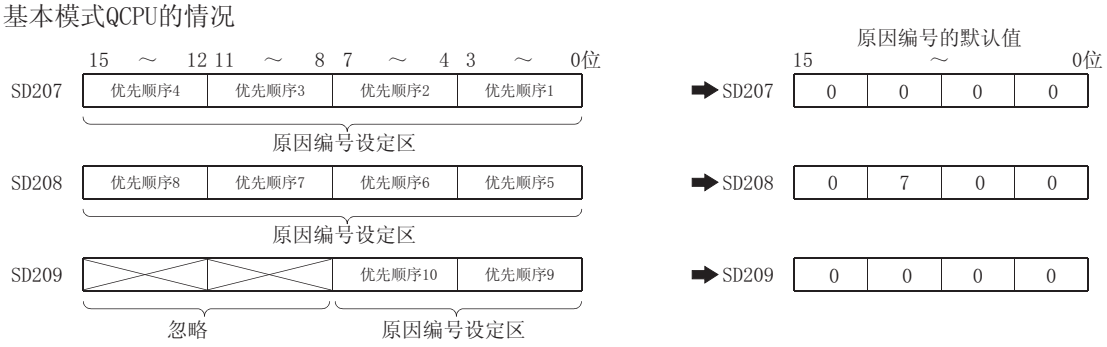
(1) 所显示的错误信息与优先顺序

发生了多个显示原因时，将根据以下条件进行显示。

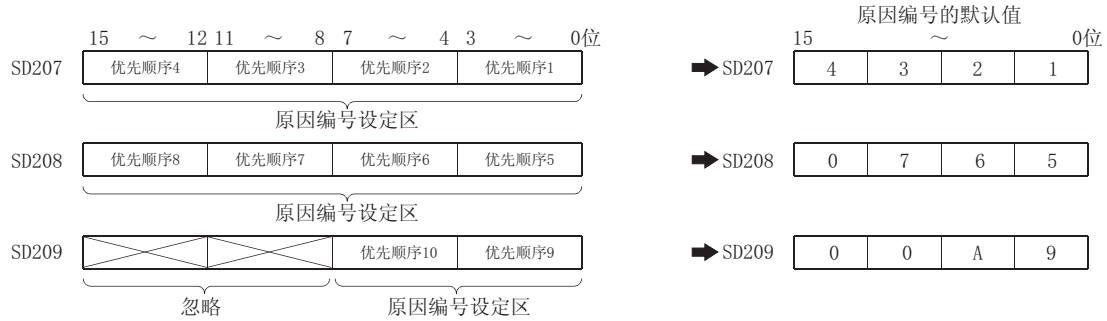
- 停止错误被无条件设定为显示器数据 (SD220~227)。
- 对于继续运行错误，根据本项所示优先顺序的原因编号进行显示。
  优先顺序可进行变更。(通过特殊寄存器 SD207~209 进行设定。)
- 同一优先顺序的错误发生时，先检测出的将被显示。

优先顺序是通过特殊寄存器 SD207~209 进行如下设定。

基本模式QCPU的情况



高性能模式QCPU、过程CPU、通用型QCPU的情况



冗余CPU的情况

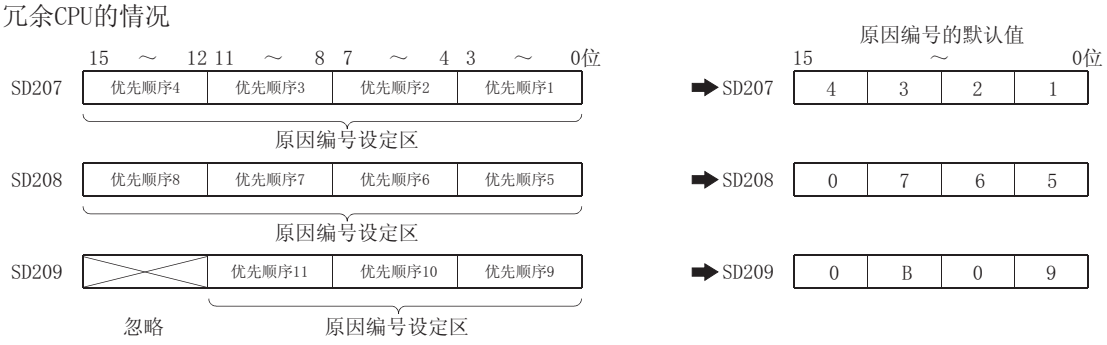


图 6.86 与优先顺序的设定相关的特殊寄存器及位构成

通用 UD 注 6.71

在通用型 QCPU 中，不能进行 LED 显示优先顺序的设定。只能设定显示・隐藏。

(2) 优先顺序与原因编号

特殊寄存器 SD207~209 中设定的原因编号内容与优先顺序的缺省值如下：

关于特殊寄存器 SD207~209，请参照附录 2。

表 6.37 原因编号与优先顺序的缺省值一览

优先顺序	原因编号 注 6.72 (16 进制数)	显示的出错信息	备注
1	1	AC/DC DOWN SINGLE PS.DOWN 注 6.73 SINGLE PS.ERROR 注 6.73	电源断开 冗余基板的电源电压过低 冗余电源模块异常
2	2	UNIT VERIFY ERR. FUSE BREAK OFF SP.UNIT ERROR SP.UNIT DOWN	输入输出模块校验出错 保险丝熔断 智能功能模块校验出错
3	3	OPERATION ERROR SFCP OPE.ERROR SFCP EXE.ERROR	运算出错 SFC 指令运算出错 SFC 程序执行出错
4	4	ICM.OPE.ERROR FILE OPE.ERROR OPE. MODE DIFF. 注 6.74 CAN'T EXE.MODE 注 6.74 TRK.TRANS.ERR. 注 6.74 TRK.SIZE ERROR 注 6.74 TRK.DISCONNECT 注 6.74 FLASH ROM ERROR 注 6.75	存储卡操作出错 文件访问出错 运行状态、开关不一致 当前模式时不能执行功能 热备数据通信出错 热备容量溢出出错 热备电缆未连接 / 故障 快闪卡访问次数溢出出错

(转下页)

基本 注 6.72

高性能 过程 通用 注 6.73

高性能 过程 通用 注 6.74

基本 高性能 过程 注 6.75

冗余 注 6.75

基本 注 6.72

基本模式 QCPU 中仅能设定原因编号 7(报警器)。

高性能 过程 通用 注 6.73

应确认与高性能模式 QCPU、过程控制 CPU 相对应的 CPU 模块的版本 (附录 4)。
在通用型 QCPU 中不能显示。

高性能 过程 通用 注 6.74

在高性能模式 QCPU、过程控制 CPU、通用型 QCPU 中不能显示。

基本 高性能 过程 注 6.75

冗余 注 6.75

在基本模式 QCPU、高性能模式 QCPU、过程 CPU、冗余 CPU 中不能显示。

表 6.37 原因编号与优先顺序的缺省值一览（续）

优先顺序	原因编号 注 6.76 (16 进制数)	显示的出错信息	备注
5	5	PRG. TIME OVER MULTI CPU ERROR	恒定扫描设置时间溢出 低速执行监视时间溢出 注 6.77 多 CPU 系统构成时的他号机异常
6	6	CHK 指令 注 6.78	--
7	7	报警器	--
8	8	--	--
9	9	BATTERY ERROR	--
10	A	--	--
11	B	CAN'T SWITCH 注 6.79 STANDBY SYS. DOWN 注 6.79 MEM. COPY EXE. 注 6.79	系统切换出错 待机系统未启动出错 / 停止出错 实施存储器复制功能

基本 注 6.76　冗余 通用 注 6.77　通用 注 6.78　高性能 过程 通用 注 6.79

要 点

1. 如果要使 LED 在发生错误时不亮灯，请将存储了 SD207~209 相应原因编号的原因编号设定区域（各 4 位）设为 0。

   （例）检测出保险丝断裂错误时，将原因编号为 “2” 的原因编号设定区域设为 0，即可使 LED 不亮灯。

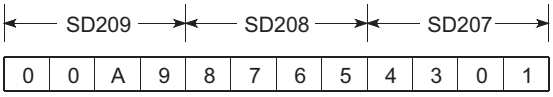


图 6.87 SD207~209 中存储的原因编号

   由于原因编号 “2” 未被设定，所以，即使检测出保险丝断裂 ERR. LED 也不会亮灯。
   此时，即使检测出原因编号 “2” 的其它错误（I/O 校对错误、智能功能模块校对错误），ERR. LED 也不会亮灯。

2. 即使将原因编号设定区域设为 0(LED 不亮灯的设定）之后，仍然可执行 SM0（检测错误标志）的 ON、SM1（自我检测错误标志）的 ON 以及将错误编码存储至 SD0（检测错误寄存器）。

基本 注 6.76
在基本模式 QCPU 中，只能设定原因编号 7（报警器）。

冗余 通用 注 6.77
在冗余 CPU、通用型 QCPU 中，由于不能使用低速执行类型程序，因此不会发生低速执行监视时间溢出。

通用 注 6.78
在通用型 QCPU 中，不能使用 CHK 指令。

高性能 过程 通用 注 6.79
应确认与高性能模式 QCPU、过程控制 CPU 相对应的 CPU 模块的版本。（☞ 附录 4）
在通用型 QCPU 中不能显示。

基本 注 6.80 过程 注 6.80 冗余 注 6.80 通用 注 6.80 高性能 注 6.81

6.22 高速中断功能

QnHCPU 中，使用中断指针 I49 编写中断程序后，可通过 0.2ms~1.0ms 间隔的高速恒定周期中断来执行程序。
此外，通过在高速中断程序的执行前后刷新设定了参数的范围的 I/O 信号，以及智能功能模块的缓冲存储器，可提高 QnHCPU 的 I/O 的响应性。
由此，精密的位置检测等高精度控制得以实现。

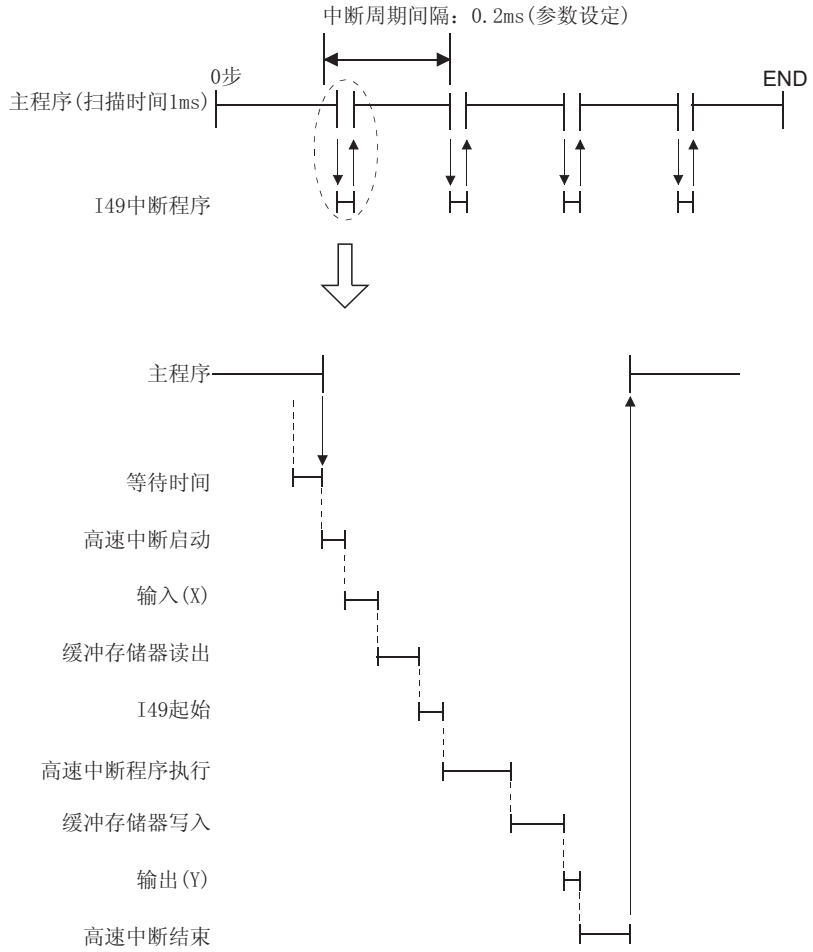


图 6.88 高速中断时间图

基本 注 6.80 过程 注 6.80 冗余 注 6.80 通用 注 6.80

在基本模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 中不能使用高速中断功能。

高性能 注 6.81

Q02CPU 不能使用高速中断功能。
此外，在 QnHCPU 中使用高速中断功能时，请确认 CPU 模块以及 GX Developer 的版本。
(☞ 附录 4.2)

(1) 对应的 CPU 模块

表 6.38 高速中断功能对应的 CPU 模块

对应 CPU 模块	备注
Q02HCPU、Q06HCPU、Q12HCPU、Q25HCPU	CPU 模块的版本有限制 (☞ 附录 4.2(2))

(2) 高速中断功能的规格

表 6.39 高速中断功能的规格

项目	内容	备注
中断周期间隔	0.2~1.0ms	以 0.1ms 为单位
中断程序数	1 个	中断指针 I49

高速中断功能的设定步骤为：可编程控制器参数的 “可编程控制器系统设定”→“系统中断设定”→“高速中断设定”。

(3) 高速中断功能的详细项目

表 6.40 高速中断功能的详细项目

项目	内容	参照
中断程序的执行	执行 I49 中编写的中断程序。	本项
高速 I/O 刷新	以中断周期间隔在 I/O 模块、智能功能模块与 CPU 模块之间更新 I/O 信号。	本项
高速缓冲发送	以中断周期间隔在智能功能模块的缓冲内存数据与 CPU 模块的软元件数据之间进行更新。	

要 点

由于高速中断功能是由中断指针 I49 以 0.2ms~1.0ms 间隔执行中断，因此请勿执行其它中断指针的中断程序以及恒定周期程序。
执行了中断程序以及恒定周期程序后，不能按设定的中断周期间隔来执行高速中断。
关于上述以外的限制事项，请参照本项。

基本 注 6.82
过程 注 6.82
冗余 注 6.82
通用 注 6.82
高性能 注 6.83

6.22.1 高速中断程序执行

高速中断程序执行功能是指根据高速中断指针 I49 来执行中断程序的功能。

(1) 设定方法

高速中断指针 I49 的设定步骤如下：可编程控制器参数的 “可编程控制器系统设定” →“系统中断设定”→设定 “高速中断设定”中的 “高速中断 I49 恒定周期间隔”。

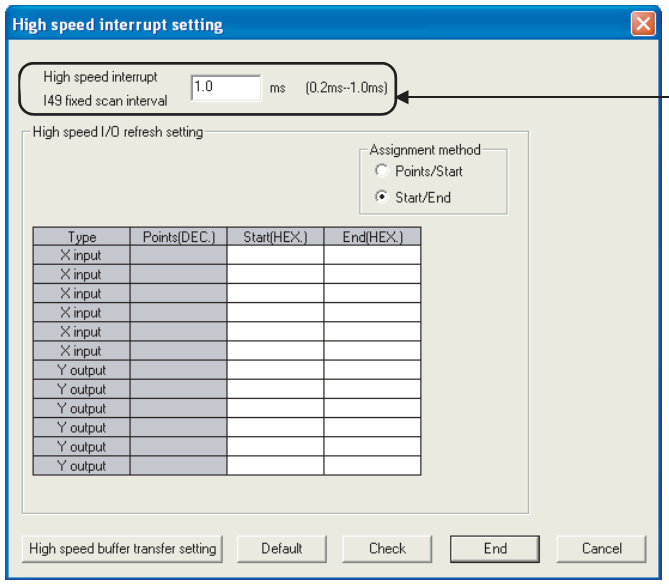


图 6.89 高速中断设定

(2) 设定注意事项

(a) 关于中断禁止中的高速中断

在中断禁止中不能执行高速中断程序。

通过解除中断禁止后执行。

关于因中断禁止而导致高速中断启动等待的项目，请参照 6.22.4 项 (3)。

(b) 关于高速中断被忽略时

中断禁止持续期间超过所设定的中断周期间隔时，高速中断可能被忽略。

中断禁止中发生了 2 次高速中断时，其中 1 次高速中断将被忽略。

基本 注 6.82　过程 注 6.82　冗余 注 6.82
通用 注 6.82

在基本模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 中不能使用高速中断程序功能。

高性能 注 6.83

Q02CPU 不能使用高速中断程序功能。

(c) 高速中断程序的执行可否

本功能在全部满足以下条件时执行。

- EI 指令中
- CPU 模块处于 RUN 状态
- IMASK 指令中 I49 未被掩蔽

缺省状态下，IMASK 指令中 I49 未被掩蔽。

备 注

关于 IMASK 指令以及 EI 指令，请参照以下手册。

☞QCPU(Q 模式)/QnACPU 编程手册（公共指令篇）

基本 注 6.84
过程 注 6.84
冗余 注 6.84
通用 注 6.84
高性能 注 6.85

6.22.2 高速 I/O 刷新、高速缓冲发送

高速 I/O 刷新是指以中断周期间隔在 I/O 模块、智能功能模块与 CPU 模块之间更新 I/O 信号的功能。
此外，高速缓冲发送是指以中断周期间隔在智能功能模块的缓冲存储器数据与 CPU 模块的软元件数据之间进行更新的功能。

(1) 设定方法

为执行本功能，需要设定本项中所示的 “高速中断 I49 恒定周期间隔”、“高速 I/O 刷新设定”以及 “高速缓冲发送设定”。

(a) 高速 I/O 刷新设定

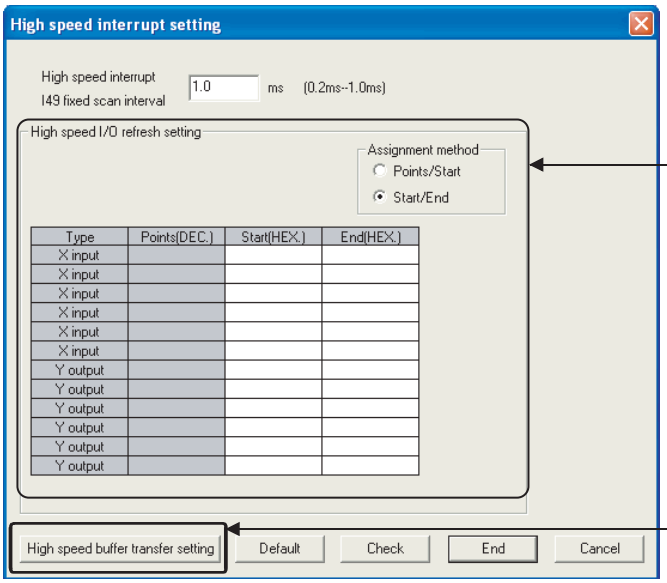


图 6.90 高速中断设定画面

基本 注 6.84　过程 注 6.84　冗余 注 6.84
通用 注 6.84

在基本模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 中不能使用高速 I/O 刷新以及高速缓冲传送功能。

高性能 注 6.85

Q02CPU 不能使用高速 I/O 刷新和高速缓冲发送功能。

(b) 高速缓冲发送设定

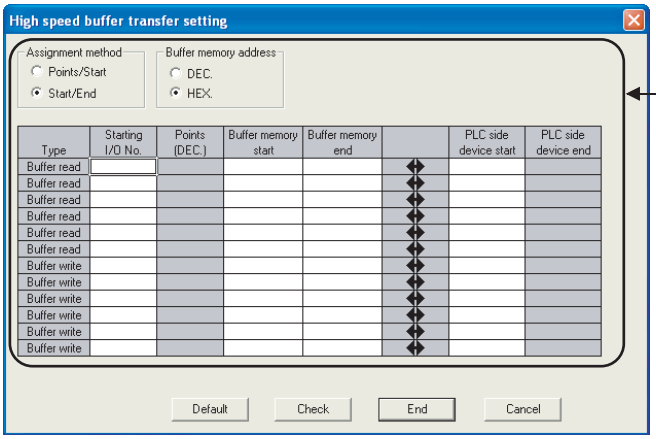


图 6.91 高速缓冲发送设定画面

表 6.41 高速 I/O 刷新设定以及高速缓冲发送设定

项目	设置项目	设置内容	限制事项	设置数
高速 I/O 刷新设定	起始 (X/Y)	起始软元件 No. (X0 ～ FF0/Y0 ～ FF0)	I/O 模块、智能功能模块 只能指定为 16 的倍数 *1	最多可设置 X 输入 /Y 输出 各 6 个
	点数	传送位数 (16 ～ 4096)		
高速缓冲发送设定	起始 I/O No.	起始 I/O No. /10H(0~FFH)	仅限于智能功能模块 *2	最多可设置 读出 / 写入 各 6 个
	点数	发送字数 (1，2~FFFEH)	仅限于智能功能模块 仅限于偶数地址、偶数字指定 *3	
	缓冲内存起始	起始地址 (0~FFFFH)		
	CPU 侧软元件起始	起始软元件 No.	仅限于 D, W, R, ZR	

*1: 起始软元件 No. 以及发送位数均只能设定为 16 的倍数。
*2: 由于不能连接 QA 基板 (QA1S6□B、QA6□B)，因此 A/QnA 用智能功能模块不在对象之内。
(连接 QA 基板 (QA1S6□B、QA6□B) 时，将检测出 “PARAMETER ERROR(3006)”。)
此外，智能功能模块安装检查、缓冲存储器空间检查中出错时，也将检测出 “PARAMETER ERROR(3006)”。
*3: 只有发送字数被指定为 1 时才可以是奇数地址。

要 点

对于具有本功能的对象模块，建议安装在主基板上。
(因为与扩展基板相比，主基板的模块存取速度快。)

(2) 本功能的执行可否

本功能在全部满足以下条件时执行。

- EI 指令中
- CPU 模块处于 RUN 状态
- IMASK 指令中 I49 未被掩蔽

缺省状态下，IMASK 指令中 I49 未被掩蔽。

备 注

关于 IMASK 指令以及 EI 指令，请参照以下手册。

☞QCPU(Q 模式)/QnACPU 编程手册（公共指令篇）

基本 注 6.86
过程 注 6.86
冗余 注 6.86
通用 注 6.86
高性能 注 6.87

6.22.3 处理时间

高速中断功能显示从启动到结束为止的各个处理时间。

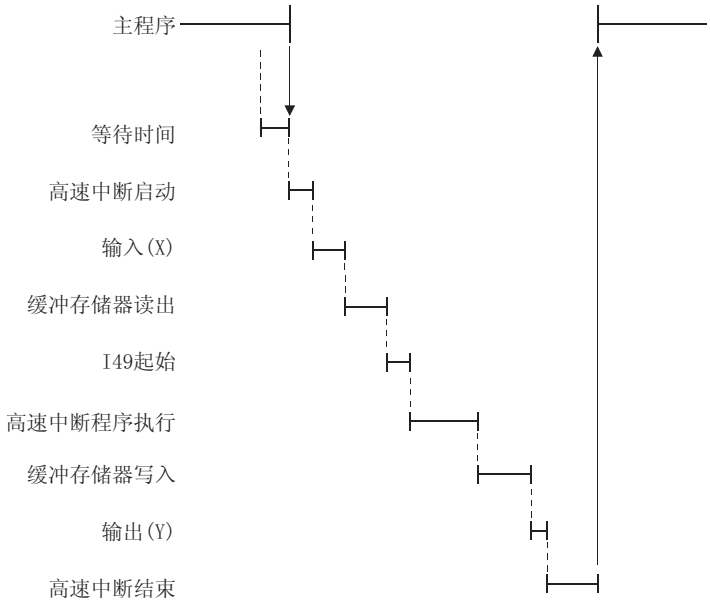


图 6.92 高速中断功能从启动至结束的处理

基本 注 6.86　过程 注 6.86　冗余 注 6.86
通用 注 6.86

由于在基本模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 中不能使用高速中断功能，因此无需理会本项的内容。

高性能 注 6.87

Q02CPU 不能使用高速中断功能，因此本项内容不适用。

表 6.42 与高速 I/O 刷新、高速缓冲发送相关的处理时间

处理项目	处理时间
等待时间	• 最长 37.5μs 或 37.5μs 以上的指令处理时间 • 将 MELSECNET/G*1、MELSECNET/H、CC-Link、智能功能模块安装到扩展基板上时，最大 40μs
高速中断启动＋高速中断结束	22μs
X 输入	〈主基板〉 时间 =0.14×(X 总点数)+0.65×(设定数)+0.85 〈扩展基板〉 时间 =0.21×(X 总点数)+0.65×(设定数)+0.85 (计算示例) 主基板安装中设定数为 1，X 点数为 16 点时，时间为 3.74μs
缓冲内存读出	〈主基板〉 •16 字以下时 时间 =0.47×(发送总字数)+2.85×(设定数)+0.95 •超过 16 字时 时间 =0.5×(发送总字数)+0.95 〈扩展基板〉 •16 字以下时 时间 =1.07×(发送总字数)+2.85×(设定数)+0.95 •超过 16 字时 时间 =1.1×(发送总字数)+0.95 (计算示例) 主基板安装中设定数为 1，字数为 2 时，时间为 4.74μs
I49 起始	41μs
高速中断程序执行	依据用户编写的中断程序。
缓冲内存写入	〈主基板〉 •16 字以下时 时间 =0.47×(发送总字数)+2.65×(设定数)+0.95 •超过 16 字时 时间 =0.55×(发送总字数)+0.95 〈扩展基板〉 •16 字以下时 时间 =1.07×(发送总字数)+2.65×(设定数)+0.95 •超过 16 字时 时间 =1.15×(发送总字数)+0.95 (计算示例) 主基板安装中设定数为 1，字数为 2 时，时间为 4.54μs
Y 输出	〈主基板〉 时间 =0.13×(Y 总点数)+1.55 〈扩展基板〉 时间 =0.2×(Y 总点数)+1.55 (计算示例) 主基板安装中设定数为 1，字数为 16 时，时间为 3.63μs

*1: 在高性能模式 QCPU 中使用 MELSECNET/G 时，请确认 CPU 模块以及 GX Developer 的版本。
(附录 4.2)

基本 注 6.88 过程 注 6.88 冗余 注 6.88 通用 注 6.88 高性能 注 6.89

6.22.4 限制事项

在本项中，对执行高速中断功能时的注意事项进行说明。
如果执行了以下注意事项所列的错误项目时将发生 WDT 错误，将不能以设定周期间隔执行高速中断。
限制事项注意有以下 4 大类。

(1) 高速中断设定时全部不能使用的项目
(2) 只在高速中断内不能使用的项目
(3) 由于高速中断禁止而导致高速中断启动等待的项目
(4) (1)~(3) 以外的注意事项

此外，请确保 1 次中断程序所需时间不超过中断周期间隔的设定时间。（如果 1 次中断程序所需时间超过了中断周期间隔的设定时间，将不能保证高速中断的动作。）

(1) 高速中断设定时全部不能使用的项目

表 6.43 高速中断设定时全部不能使用的项目

No.	项目	限制事项	使用后
1	不对应的 CPU 模块	高速中断功能只能在 6.22 节 (1) 所示的 CPU 模块中使用	检测出参数错误。
2	基板	不能连接 QA1S6□8，QA6□B 基板	检测出参数错误。
3	多 CPU 系统	不能构成多 CPU 系统	在 GX Developer 的参数设定时进行检查。
4	指令	不能执行 PR/PRC, UDCNT1/2, PLSY, PWM, SPD, PLOADP/PUNLOADP/PSWAPP 指令	不执行左边的指令，检测出错误。
5	指令	不能使用处理时间超过高速中断周期的指令	因为执行指令时中断禁止，因此不能按设定的周期执行高速中断。
6	编程模块	不能连接编程模块	指令查找的响应变慢。 此外，可能发生编程模块侧的通讯错误。
7	SFC	不能执行以下两个 SFC 功能： •SM90~99.SD90~99 的 SFC 转移监视检查功能 •定时执行块执行功能	不执行左边的功能，忽略。
8	采样追踪	不能使用各时间的采样追踪（各扫描周期、详细条件执行时可使用。）	不执行采样追踪，忽略。 （跟踪读出时，数据可能未被设定。）
9	中断程序 (I0~48，I50~255)、恒定周期程序	不能执行中断程序 (I0~48，I50~255)、恒定周期程序	由于多重中断禁止，在执行中断程序、恒定周期程序时，不能按设定的周期执行高速中断。

基本 注 6.88 过程 注 6.88 冗余 注 6.88 通用 注 6.88

由于在基本模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 中不能使用高速中断功能，因此无需理会本项的内容。

高性能 注 6.89

由于 Q02CPU 不能使用高速中断功能，因此本项内容不适用。

表 6.43 高速中断设定时全部不能使用的项目（续）

No.	项目	限制事项	使用后
10	RUN 中写入，文件的 RUN 中批量写入	不能执行 RUN 中写入，不能执行文件的 RUN 中批量写入	由于执行 RUN 中写入时中断禁止，其间高速中断启动变慢，因此不能按设定的周期执行高速中断。 （所需时间如下： •RUN 中写入的最长时间为 102μs， •文件的 RUN 中批量写入的最长时间为 300ms）
11	与程序名相同的文件寄存器	不能使用本地软元件	由于与程序名相同的文件寄存器在切换时中断禁止，因此不能按设定的周期执行高速中断。 （所需时间如下： •标准 RAM 为 410μs •SRAM 卡为 400μs+100μs×程序文件数）
12	本地软元件	不能使用本地软元件	由于本地软元件切换时中断禁止，因此不能按设定的周期执行高速中断。 （所需时间如下： •标准 RAM 为 390μs+170μs×n， •SRAM 卡为 390μs+950μs×n •n：程序文件数）
13	对 CPU 模块进行存取的智能功能模块的指令	对于向 QJ71C24,QJ71E71 等 CPU 模块进行存取的智能功能模块，不能由其发出 CPU 存取指令	由于发出 CPU 存取指令时中断禁止，因此其间高速中断启动变慢，不能按设定的周期执行高速中断。 N 点读出 / 写入：(0.07×N+34)μs N 点随机读出 / 写入：(0.07×N+101)μs
14	经由其它站点的监视	自站点监视中不能执行经由 MELSECNET/G*1、MELSECNET/H、QJ71C24 等智能功能模块的监视。	自站点监视请求与经由智能功能模块的监视请求重复时，中断禁止的处理时间将延长，其间高速中断启动变慢 (102μs)，因此不能按设定的周期执行高速中断。
15	中断计数器	不能使用与中断指针 I49 对应的中断计数器	即使有中断计数设定，对 I49 的设定也被忽略，高速中断 I49 仍将被执行。 （其它的中断指针不执行中断程序，执行中断计数。）

*1: 在高性能模式 QCPU 中使用 MELSECNET/G 时，请确认 CPU 模块以及 GX Developer 的版本。
（☞ 附录 4.2）

(2) 只在高速中断内不能使用的项目

表 6.44 只在高速中断内不能使用的项目

No.	项目	限制事项	使用后
1	软元件注释	对于高速中断程序内与程序名相同的软元件注释，不能退避 / 复位	高速中断程序的软元件注释将被改写。
2	变址寄存器	高速中断梯形图内，变址寄存器不能退避 / 复位	高速中断程序的变址寄存器将被改写。
3	存取执行标志 SM390	高速中断程序内，存取执行标志不能退避 / 复位	高速中断程序的 SM390 的值将被改写。
4	强制 ON/OFF	高速 X/Y 刷新区域不能强制 ON/OFF	高速中断内，不执行，忽略。（不会导致超时错误。）
5	详细条件监视	高速中断程序内不能指定。	不能正常执行。（不会导致超时错误。）
6	执行时间测量	高速中断程序内不能指定。	不执行，忽略。（不会导致超时错误。）

(3) 由于高速中断禁止而导致高速中断的启动等待的项目

表 6.45 由于高速中断禁止而导致高速中断的启动等待的项目

No.	项目	注意事项
1	指令	执行指令时中断禁止。
2	链接刷新	刷新（总线存取）时中断禁止。 在 MELSECNET/G*1、MELSECNET/H、CC-Link、智能功能模块的刷新中，各模块安装在主基板时的等待时间最长为 37.5μs，安装在扩展基板时的等待时间最长为 40μs。
3	执行多个程序	执行多个程序时，切换程序时中断禁止。等待 30μs。 设定高速中断功能时，程序个数建议设定为 1。
4	监视	梯形图监视、软元件批量监视、软元件登录监视的等待时间如下： (0.096×软元件点数 +20)μs
5	AC DOWN 时	高速中断启动等待时间最长为 20ms。

*1: 在高性能模式 QCPU 中使用 MELSECNET/G 时，请确认 CPU 模块以及 GX Developer 的版本。
（☞ 附录 4.2）

(4) (1)~(3) 以外的注意事项

(a) 关于可编程控制器参数的中断程序 / 恒定周期程序设定
GX Developer 的可编程控制器系统设定 “中断程序 / 恒定周期程序设定”的 “高速执行”对高速中断功能无效。

(b) 关于高速缓冲发送时
高速缓冲发送时，使用了设定范围外的文件寄存器（超过最大点数的范围）时，为了避免出错，不实施范围以外的发送。（不会破坏其它软元件的内容。）

(c) 程序编写的注意事项
程序编写的注意事项与其它中断程序相同。(☞ 3.1.3 项 (7))

6.23 智能功能模块发出的中断

基本 注 6.90

CPU 模块中，可根据智能功能模块发出的中断请求，执行中断程序（I□）。
例如，串行口通讯模块中，在执行以下数据通讯功能时可通过中断程序进行数据接收处理。

- 利用无序通讯协议进行通讯时的数据接收
- 利用双向通讯协议进行通讯时的数据接收

通过中断程序进行数据接收处理，可加快 CPU 模块的数据接收。

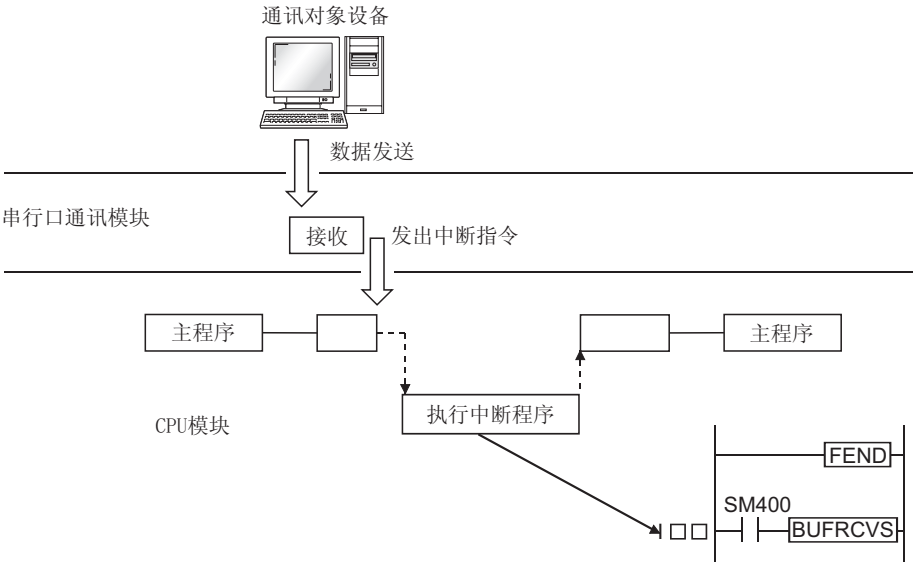


图 6.93 由串行口通讯模块发出的中断

(1) 智能功能模块发出的中断的设定

由智能模块发出的中断而执行中断程序时，需要在可编程控制器参数的可编程控制器系统设定中进行“智能功能模块设定（中断指针设定）”。
此外，智能功能模块中需要进行“系统设定”。
由智能功能模块发出的中断而执行中断程序时，请参照以下手册。
☞所使用的智能功能模块的手册

备 注

关于执行由智能模块发出的中断时可使用的中断指针编号，请参照 9.10 节。

基本 注 6.90

在基本模式 QCPU 中，使用由智能功能模块发出的中断时，请先确认 CPU 模块以及 GX Developer 的版本。(☞附录 4.1)

基本 注 6.91

6.24 串行口通讯功能

高性能 注 6.92
过程 注 6.92
冗余 注 6.92
通用 注 6.92

(1) 什么是串行口通讯功能

串行口通讯功能是指利用 RS-232 电缆将 CPU 模块的 RS-232 接口与 PC、显示器等连接，通过 MC 协议 *1 进行通讯的功能。

利用串行口通讯功能的通讯只能在 Q00CPU,Q01CPU 中执行。

（其它 CPU 模块不具备串行口通讯功能。）

使用串行口通讯功能与 PC、显示器等进行通讯时，所需的规格、功能、各种设定说明如下。

*1: MC 协议是 MELSEC 通讯协议的简称。
MELSEC 通讯协议是指根据 Q 系列可编程控制器（串行口通讯模块、Ethernet 接口模块等）的通讯步骤，从对象设备向 CPU 模块进行存取的通讯方式的名称。
关于 MELSEC 通讯协议，请参照以下手册。
☞ Q 对应 MELSEC 通讯协议参考手册

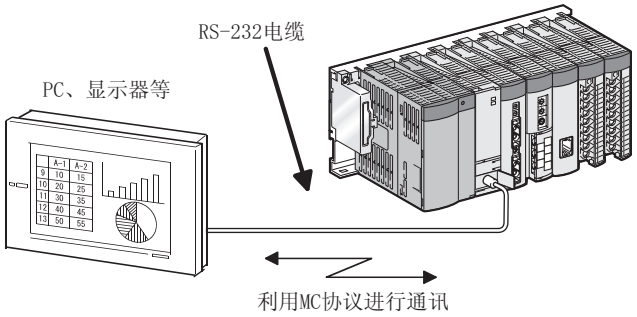


图 6.94 与 PC、显示器等的通讯

要 点

1. PC、显示器等只能通过串行口通讯功能与所连接的 CPU 模块进行通讯。
   PC、显示器等不能经由所连接的 CPU 模块与 MELSECNET/H、Ethernet、CC-Link 的其它站点进行通讯。
2. GX Developer,GX Configurater 与 CPU 模块的连接不能使用串行口通讯功能。

基本 注 6.91

在 Q00JCPU 中，不能使用串行口通讯功能。

高性能 注 6.92　过程 注 6.92　冗余 注 6.92
通用 注 6.92

在高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 中，不能使用串行口通讯功能。

(2) 规格

(a) 发送规格

CPU 模块的串行口通讯功能中使用的 RS-232 的发送规格如表 6.46 所示。

请务必确认 PC / 显示器等的规格符合表 6.46 以后，再使用串行口通讯功能。

表 6.46 串行口通讯功能的发送规格

项目	缺省	设定范围
通讯方式	全双工通讯	—
同步方式	调步同步方式	—
发送速度 *1	19.2kbps	9.6kbps, 19.2kbps, 38.4kbps, 57.6kbps, 115.2kbps
数据形式	起始位：1 数据位：8 奇偶校验位：奇数 停止位：1	—
MC 协议形式 *2（自动判断）	形式 4(ASCII) 形式 5(2 进制）	—
帧 *2	QnA 兼容 3C 帧 QnA 兼容 4C 帧	—
发送控制	DTR/DSR 控制	—
总数检查 *1	无	有，无
发送等待时间 *1	无等待	无等待，10ms~150ms（以 10ms 为单位）
RUN 中写入设定 *1	不许可	许可，不许可
延长距离	15m	—

*1: 可以在 GX Developer 的可编程控制器参数设定中进行设定。

*2: MC 协议格式与帧的关系如表 6.47 所示。

表 6.47 MC 协议格式与帧的关系

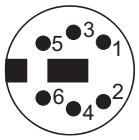
功能		型号 4	型号 5
ASCII 编码通讯	QnA 兼容 3C 帧	○	×
	QnA 兼容 4C 帧	○	×
2 进制码通讯	QnA 兼容 4C 帧	×	○

○：可使用， ×：不可使用

(b) RS-232 连接器规格

CPU 模块的 RS-232 连接器的用途如表 6.48 所示。

表 6.48 RS-232 连接器的规格

外观	针号	信号名	信号名称
 Mini-Din 6针脚 (凹型)	1	RD(RXD)	接收数据
	2	SD(TXD)	发送数据
	3	SG	信号地
	4	—	—
	5	DSR(DR)	数据设定就绪
	6	DTR(ER)	数据终端就绪

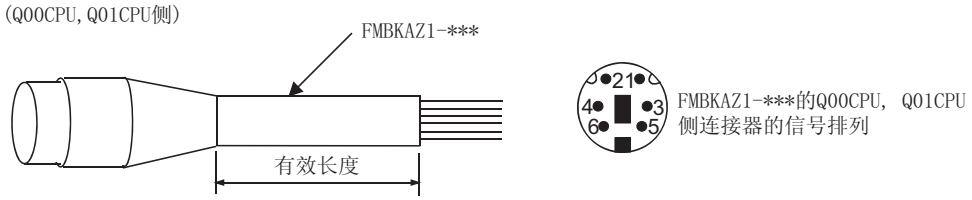
(c) RS-232 电缆

PC/ 显示器等与 CPU 模块的连接可使用以下 RS-232 电缆。

- QC30R2(电缆长：3m)
- FMBKAZ1-***(仓茂电工株式会社制)

  单侧：带有迷你 DIN 连接器，单侧：无连接器的电缆

  *** 表示电缆长度，以 0.1m 为单位，可指定至 15m。



针脚编号	1	2	3	4	5	6	金属壳
信号名	RD	SD	SG	—	DR	ER	
线芯	红	黑	绿 / 白	—	黄	褐	屏蔽

图 6.95 RS-232 电缆的有效长度与信号排列

(3) 功能

串行口通讯功能可执行表 6.49 的 MC 协议的指令。

关于 MC 协议的详细内容。请参照以下手册。

☞Q 对应 MELSEC 通讯协议参考手册

表 6.49 串行口通讯功能支持的 MC 协议指令一览

功能			指令	处理内容	处理点数
软元件内存	批量读出	位单位	0401(00□1)	以 1 点为单位读出位软元件。	ASCII：3584 点 BIN：7168 点
		字单位	0401(00□0)	以 16 点为单位读出位软元件。	480 字（7680 点）
				以 1 点为单位读出字元件。	480 点
	批量读入 *1	位单位	1401(00□1)	以 1 点为单位写入位软元件。	ASCII：3584 点 BIN：7168 点
		字单位	1401(00□0)	以 16 点为单位写入位软元件。	480 字（7680 点）
				以 1 点为单位写入字元件。	480 点
	随机读出	字单位	0403(00□0)	以 16 点、32 点为单位，随机指定位软元件及其软元件编号后读出。	96 点
				以 1 点、2 点为单位，随机指定字元件及其软元件编号后读出。	
	测试 *1 （随机写入）	位单位	1402(00□1)	以 1 点为单位，随机指定位软元件及其软元件编号后进行设定和复位。	94 点
		字单位	1402(00□0)	以 16 点、32 点为单位，随机指定位软元件及其软元件编号后进行设定和复位。	*2
				以 1 点、2 点为单位，随机指定字软元件及其软元件编号后写入。	
	监视登录	字单位	0801(00□0)	以 16 点、32 点为单位，将要监视的位软元件进行登录。	96 点
				以 1 点、2 点为单位，将要监视的字元件进行登录。	96 点
	监视	字单位	0802(00□0)	对进行了登录的软元件进行监视。	监视登录点数

*1: 在 CPU 模块 RUN 中写入时，将 RUN 中写入设定设为 “许可”。

*2: 处理点数在以下范围内设定。

（字存取点数）×12+（双字存取点数）×14≤960

•对于位软元件，字存取时 1 点为 16 位，双字存取时 1 点为 32 位。

•对于字元件，字存取时 1 点为 1 字，双字存取时 1 点为 2 字。

(4) 可存取的软元件

表 6.50 串行口通讯功能中可存取的软元件

分类	软元件		软元件编码	软元件编号范围 *1 （缺省值）		写入	读出
内部系统软元件	内部系统软元件		FX*2	000000~00000F	16 进制	×	×
	功能输出		FY*2	000000~00000F	16 进制		
	功能寄存		FD	000000~000004	10 进制		
	特殊继电器		SM	000000~001023	10 进制	○	○
	特殊寄存器		SD	000000~001023	10 进制		
内部用户软元件	输入		X	000000~0007FF	16 进制		
	输入		Y	000000~0007FF	16 进制		
	内部继电器		M	000000~008191	10 进制		
	锁存继电器		L	000000~002047	10 进制		
	报警器		F	000000~001023	10 进制		
	变址继电器		V	000000~001023	10 进制		
	链接继电器		B	000000~0007FF	16 进制		
	数据寄存器		D	000000~011135	10 进制		
	链接寄存器		W	000000~0007FF	16 进制		
	计时器	触点	TS	000000~000511	10 进制		
		线圈	TC				
		当前值	TN				
	累积计时器	触点	SS	—	10 进制		
		线圈	SC				
		当前值	SN				
	计数器	触点	CS	000000~000511	10 进制		
		线圈	CC				
		当前值	CN				
	链接特殊继电器		SB	000000~0003FF	16 进制		
	链接特殊寄存器		SW	000000~0003FF	16 进制		
	步进继电器		S	000000~002047	10 进制	×	
	直接输入		DX	000000~0007FF	16 进制	○	
	直接输出		DY	000000~0007FF	16 进制		
	变址寄存器		Z	000000~000009	10 进制		
	文件寄存器		R	000000~032767	10 进制		
			ZR	000000~00FFFF	16 进制		

○：可以写入 / 读出， ×：不可写入

*1: 用 GX Developer 变更了 CPU 模块的软元件点数时，请在变更后的软元件编号范围内使用。
10 进制、16 进制表示 MC 协议的指令中的指定为 10 进制、16 进制。

*2: FX、FY 的 000005 ～ 00000F 中将被输入不定值。

(5) 发送规格的设定

串行通讯口功能的发送速度、总数检查、发送等待时间、RUN 中写入设定等，均可通过可编程控制器参数的串行口通讯设定进行。

- 使用串行口通讯功能与 PC、显示器具等进行通讯时，设定为“使用串行口通讯功能”。
- 进行发送速度、总数检查、发送等待时间、RUN 中写入设定等设定。

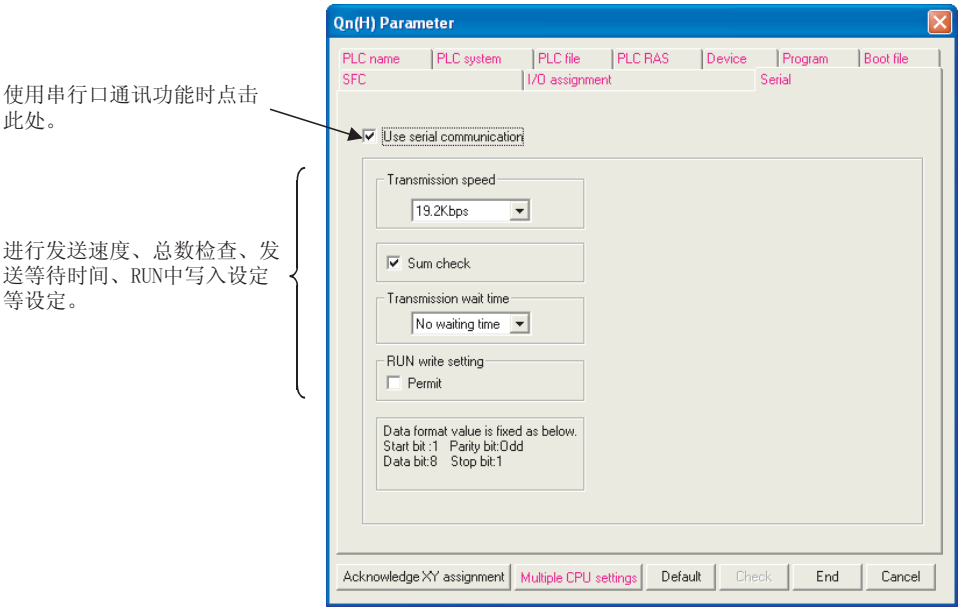


图 6.96 串行口通讯设定画面

(6) 注意事项

(a) 关于与显示器等进行通讯中切换为连接 GX Developer 时

使用串行口通讯功能与 PC、显示器等进行通讯中可切换为连接 GX Developer。重新将 PC、显示器等连接至 CPU 模块后，PC、显示器等的启动方法请参照所使用的设备的手册。

(b) 关于在连接对象指定画面中设定的发送速度

设定为“使用串行口通讯功能”后，GX Developer 的连接对象指定画面中设定的发送速度将无效。

要 点

串行口通讯设定中设定的数据在以下情况有效。

- 可编程控制器的电源 ON 时
- CPU 模块复位时

(7) 利用串行口通讯功能进行通讯时的出错代码

利用串行口通讯功能进行通讯时若发生错误，CPU 模块向外部设备发送的出错代码、错误内容及其处理如表 6.51 所示。

表 6.51 CPU 模块向外部设备发送的出错代码一览

错误码（16 进制）	错误项目	错误内容	处理方法
4000_H ~ $4FFF_H$	—	• CPU 模块检测出错误（串行通讯口功能以外发生的错误）	• 参照 QCPU 用户手册（硬件设计 / 维护点检篇）的附录进行处理。
7153_H	帧长错误	• 接收信息的长度超过了容许范围。	• 改正发送信息。 • 使信息构成符合存取点数容许范围
7155_H	监视未登录错误	• 监视登录前请求监视。	• 事先将要监视的软元件登录后再请求监视。
7164_H	请求内容错误	• 请求内容或软元件指定方法有误。	• 确认对象设备的发送信息 / 请求内容，修正后重新通讯。
7167_H	RUN 中不可	• 在设定为 RUN 中写入否时指定了写入指令。	• 将设定变更为可以 RUN 中写入，重新通讯。
7168_H		• 指定了不能在 RUN 中执行的指令	• 停止 CPU 模块后重新通讯。
$716D_H$	监视登录错误	• 没有以 QnA 兼容 3C/4C 帧进行监视登录。	• 重新进行监视登录。
$7E40_H$	指令错误	• 指定了不存在的指令、或是副指令。	• 确认对象设备的发送信息，修正后重新通讯。
$7E41_H$	数据长度错误	• 指定时超过了随机写入 / 读出时可通讯的点数。	• 确认对象设备的发送信息，修正后重新通讯。
$7E42_H$	数据数错误	• 请求点数超过了指令范围。	• 确认对象设备的发送信息，修正后重新通讯。
$7E43_H$	软元件错误	• 指定了不存在的软元件。 • 指定了该指令不能指定的软元件。	• 确认对象设备的发送信息，修正后重新通讯。
$7E47_H$	连续请求错误	• 返回响应信息前接收到了下一个请求。	• 对象设备不发出连续请求。 • 使计时器 1 的监视时间与对象设备的超时时间一致。
$7E4F_H$	软元件点数错误	• 存取点数不正确。	• 确认对象设备的发送信息，修正后重新通讯。
$7E5F_H$	请求方模块 I/O 地址号错误	• 请求方模块 I/O 地址号有误。	• 修正数据发送目标的模块 I/O 地址号。
$7E64_H$	登录点数范围错误	• 登录点数（字 / 位）不在范围内	• 修正登录点数（字 / 位）的设定值。
$7F01_H$	缓冲溢出错误	• 接收数据处理完成前接收到了下一个数据。	• 通过与对象设备握手（反馈检验）等，腾出发送间隔时间。
$7F21_H$	接收头部错误	• 指令（帧）部分指定有误。 • 接收了不能转换为二进制的 ASCII 编码。	• 确认对象设备的发送信息，修正后重新通讯。

（续下页）

表 6.51 CPU 模块向外部设备发送的出错代码一览（续）

出错代码（16 进制）	错误项目	错误内容	处理方法
7F22H	指令错误	• 指定了不存在的指令或软元件。 • 远程口令长度有误。	• 确认对象设备的发送信息，修正后重新通讯。
7F23H	MC 协议信息错误	• 字符部后面的数据（ETX,CR-LF 等）不存在，或指定错误。	• 确认对象设备的发送信息，修正后重新通讯。
7F24H	总数检查错误	• 计算的总数检查与接收的总数检查不一致。	• 改正对象设备的总数确认。
7F67H	溢出错误	• CPU 模块完成接收数据处理前接收到了下一个数据。	• 降低通讯速度后重新通讯。 • 确认通讯模块没有发生瞬停。（CPU 模块的情况下，可用特殊寄存器 SD53 进行确认） 发生了瞬停时，解除其原因。
7F69H	奇偶错误	• 奇偶位的设定不一致。	• 使 CPU 模块与对象设备一致。
7F6AH	缓冲溢出错误	• OS 的接收缓冲溢出，接收数据读出跳转。	• 进行 DTR 控制，确保通讯时不会发生缓冲溢出。
F□□□H	—	• MELSECNET/H 网络系统检测出错误	• 确认对象设备的发送信息，修正后重新通讯。 （站点编号可能已指定。 不能与 MELSECNET/H,Etnernet 其它站点进行通讯。）

6.25 服务处理

6.25.1 用户服务间隔读出

通用
注 6.93

模块服务间隔表示的是监视、测试、程序的读出 / 写入等瞬时请求的间隔。
CPU 模块可监视智能功能模块、网络模块或 GX Developer 的服务间隔时间（两次受理服务之间的时间）。
由此，可以监视到外部对 CPU 模块的存取频率。

(1) 读出方法
读出模块服务的间隔时间需操作表 6.52、表 6.53 所示的特殊继电器和特殊寄存器。

表 6.52 特殊继电器

编号	名称	内容
SM551	模块服务间隔读出	OFF→ON，则特殊寄存器 SD550 指定的智能功能模块的模块服务间隔时间读出到 SD551~552 中。 ON ：读出 OFF：无处理

表 6.53 特殊寄存器

编号	名称	内容
SD550	服务间隔测定模块	对测定模块服务间隔时间的模块的 I/O 地址号进行设定。 连接在 CPU 模块的 RS-232 或者 USB 接口的外围设备的 I/O 地址号为 $FFFF_H$。
SD551、SD552	服务间隔时间	将 SM551 置于 ON 时，存储通过 SD550 指定的模块的服务间隔时间。 SD551：1ms 单位（0 ～ 65535 范围） SD552：100μs 单位 （0 ～ 900 范围、每次存储 100μs） （例）模块服务间隔时间为 123.4ms 时 SD551=123，SD552=400

通用
注 6.93

在通用型 QCPU 中，不能使用模块间隔时间的读出。

(2)（程序示例）

读出 X/Y160 的智能功能模块的模块服务间隔时间时的程序示例如图 6.97 所示。

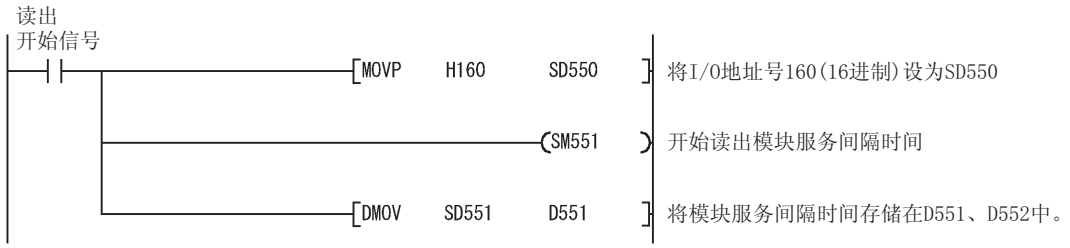


图 6.97 模块服务间隔时间读出程序示例

要 点

1. 网络模块发出的循环通讯的存取间隔将不能存储。
2. 在读出网络上的其它站点的 GX Developer 所存取的服务间隔时间时，请设定网络模块的 I/O 地址号。

6.25.2 服务成立设定

基本 注 6.94
高性能 注 6.94
过程 注 6.94
冗余 注 6.94

(1) 服务成立设定的概要

服务处理设定是指，通过参数任意指定 END 处理内执行的服务处理的次数或者时间的功能。

通过服务成立设定功能，可以提高与外围设备的通信响应，抑制服务处理导致的扫描时间延迟。由此，可以构筑系统最佳服务处理环境。

要 点

服务处理是指，与外围设备（GX Developer 等）以及智能功能模块的通信服务的处理。但是，不包括与 MELSECNET/H 网络模块、CC-Link 网络模块等的刷新处理。

此外，通过使用 COM 指令，在执行程序的过程中也可以执行与 END 处理相同的服务处理。因此，即使是在扫描时间较长的情况下，也可以实现高速服务处理响应。

基本 注 6.94　高性能 注 6.94　过程 注 6.94
冗余 注 6.94

在基本模式 QCPU、高性能模式 QCPU、过程 CPU、冗余 CPU 中，不能使用服务处理设定。

(2) 参数设定

服务处理设定是在可编程控制器参数的可编程控制器系统设定中设置。

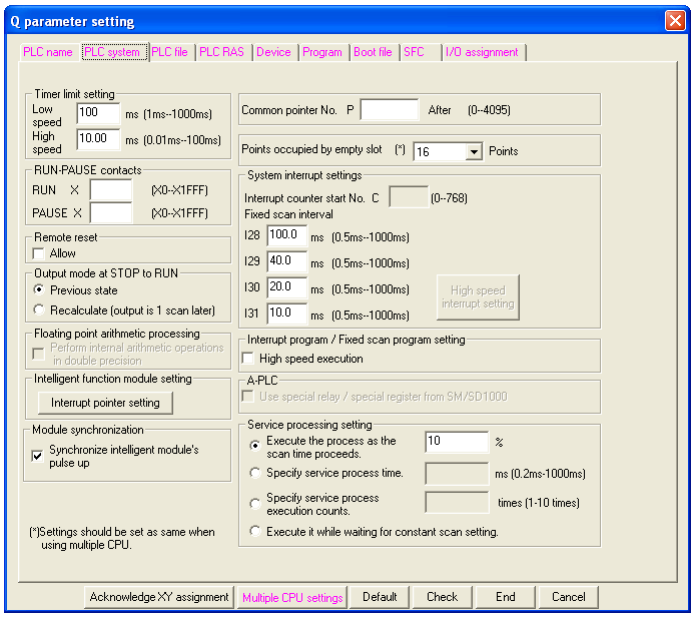


图 6.98 参数设定画面

服务处理的执行方法可从表 6.54 中的参数项目中任选其一。未选择的参数的设定值将不能输入。（默认：根据扫描时间的比例执行 =10%）

表 6.54 参数项目一览表

项目名称	设定内容	设定范围	备注
根据扫描时间的比例执行	指定 1 个扫描中执行的服务处理的比例。	• 范围： 1 ～ 99% • 单位： 1%	选择时默认 =10%
指定服务处理时间	指定 1 个扫描中执行的服务处理的时间。	• 范围： 0.2ms ～ 1000ms • 单位： 0.1ms	选择时默认 =0.2ms
指定服务处理次数	指定 1 个扫描中执行的服务处理的次数。	• 范围： 1 ～ 10 次 • 单位： 1 次	选择时默认 =1 次
在恒定扫描设定的等待时间执行	指定在恒定扫描设定时的等待时间是否执行服务处理。	------	在执行等待时间为 0.2ms 以下的服务处理时，将服务处理时间 (0.2ms) 加到扫描时间中。

(3) 各服务处理设定时的动作

执行各服务处理设定时的动作的详细情况如下所示。

(a) 根据扫描时间的比例执行时的动作详细情况

1) 设定 “扫描时间的比例 =10%”时的动作

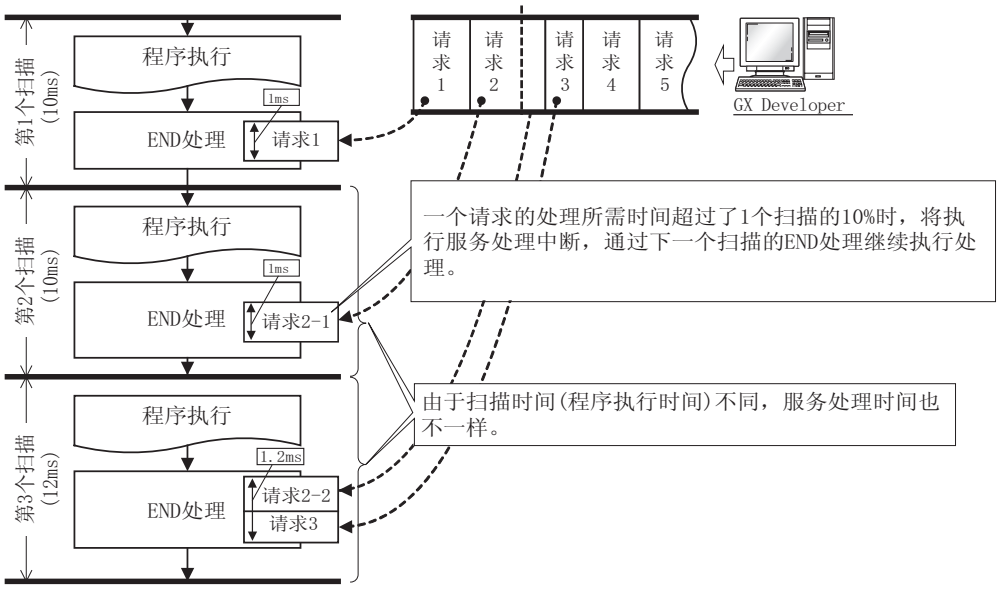


图 6.99 设定 “扫描时间的比例 =10%”时的动作

要 点

服务处理的请求数据不存在时，END 处理的速度将快相当于请求处理的时间。(不执行请求等待处理。)

2) 恒定扫描设定时的动作

服务处理时间的计算不是对扫描时间，而是对从扫描时间中减去恒定扫描的等待时间后的时间进行比例计算。

[例]

设定 “扫描时间的比例 =50%”的情况

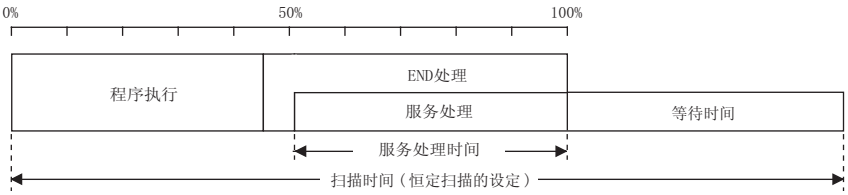


图 6.100 恒定扫描设定时的动作

要 点

设定了恒定扫描时，通过选择 “在恒定扫描设定的等待时间执行服务处理”可以提高服务处理的效率。(☞ 本项 (3)(d))

(b) 指定服务处理次数时的动作详细情况

1) 设定 “服务处理次数 =1 次”时的动作

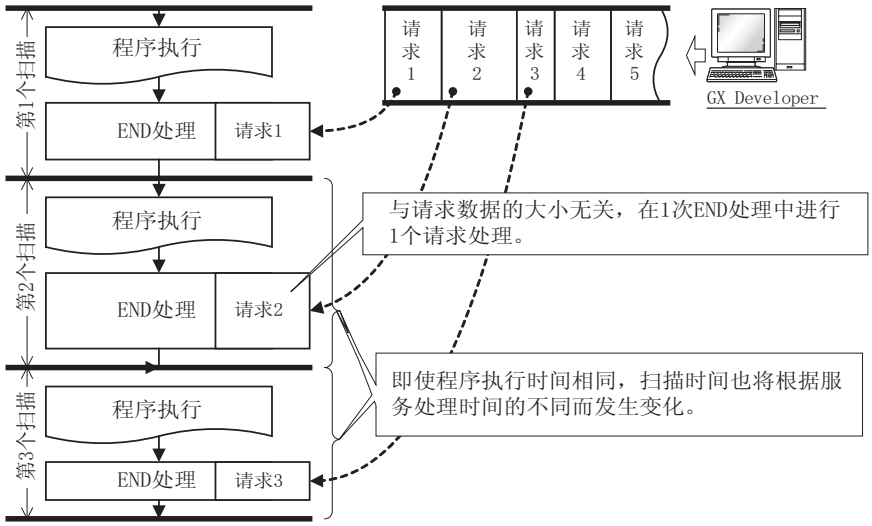


图 6.101 设定 “服务处理次数 =1 次”时的动作

2) 设定 “服务处理次数 =2 次”时的动作

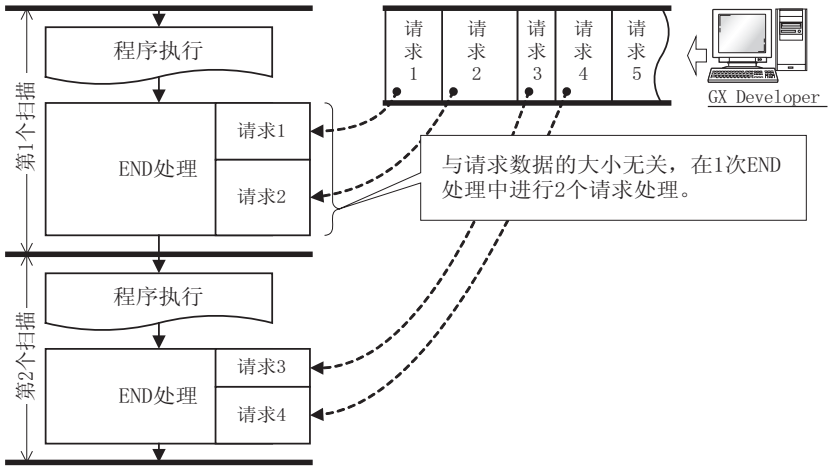


图 6.102 设定 “服务处理次数 =2 次”时的动作

要 点

1. 将多台设备与 1 个 CPU 模块相连接时，将从各个设备分别接受服务处理。
   从多个设备的请求重叠时，如果按设备的台数设定服务处理次数，由于 1 次的 END 处理可以同时接受多台设备的请求，因此可以提高响应性能。（但是，扫描时间将延长相当于服务处理的时间。）
2. 在指定服务处理次数时请求数据不存在的情况下，END 处理将变快相当于请求处理的时间。（不执行请求等待处理。）

(c) 指定服务处理时间时的动作详细情况

1) 设定 “服务处理时间 =0.5ms” 时的动作

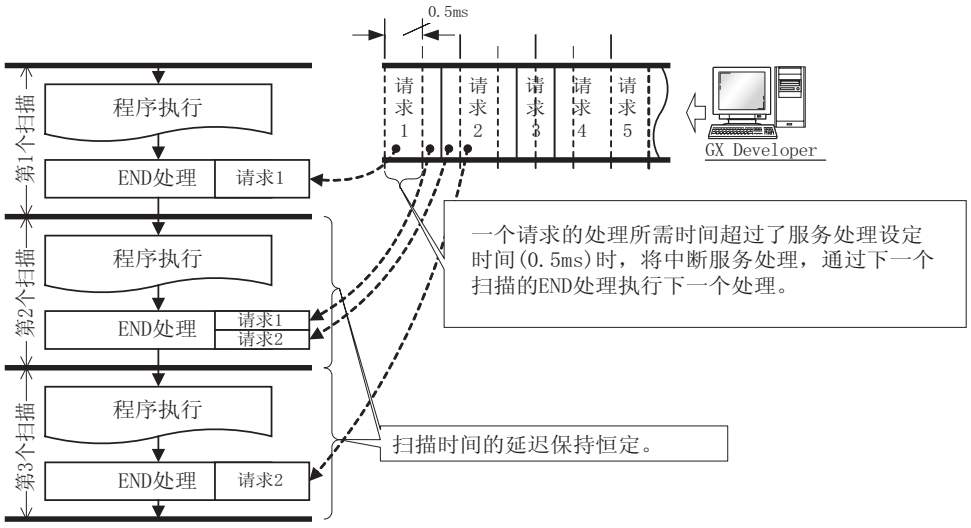


图 6.103 设定 “服务处理时间 =0.5ms” 时的动作

2) 设定设定 “服务处理时间 =1ms” 时的动作

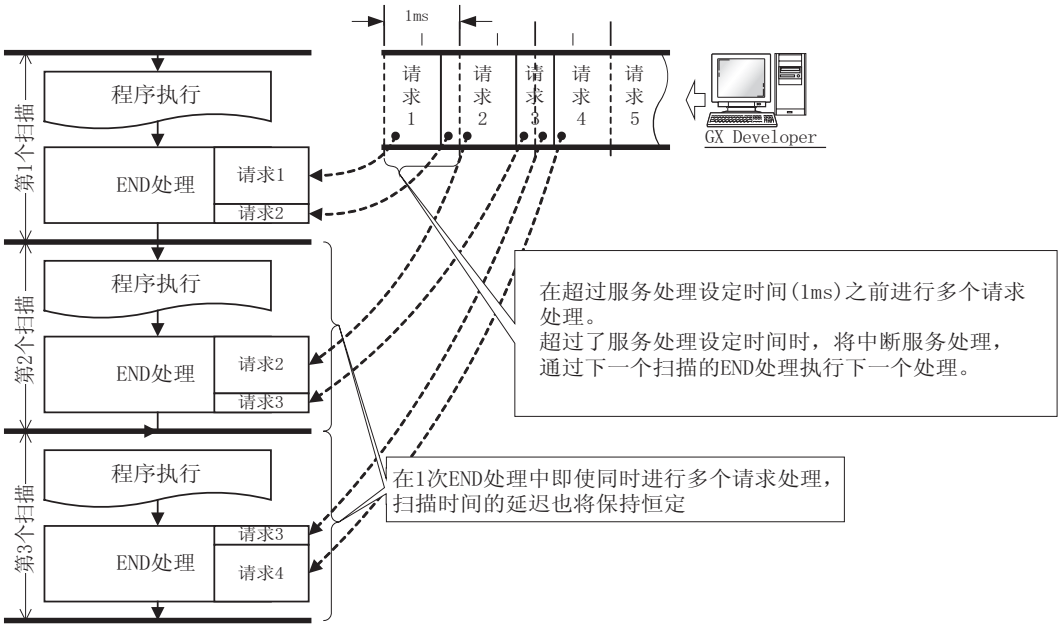


图 6.104 设定 “服务处理时间 =1ms” 时的动作

要 点

在指定服务处理时间时请求数据不存在的情况下，END 处理将变快相当于请求处理的时间。(不执行请求等待处理。)

(d) 在恒定扫描设定的等待时间执行服务处理时的动作详细情况

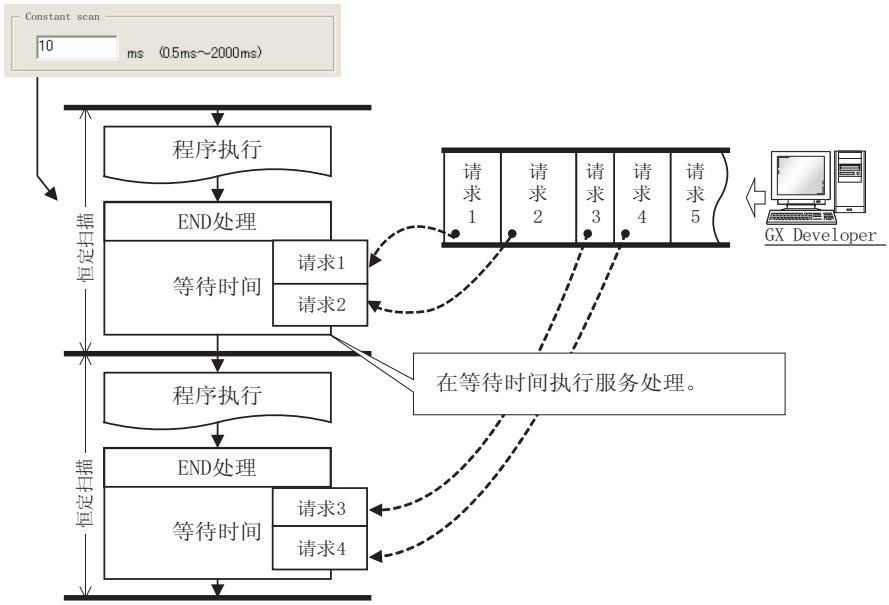
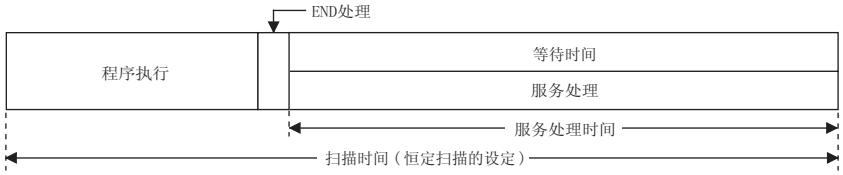


图 6.105 设定 “在恒定扫描设定的等待时间执行服务处理”时的动作

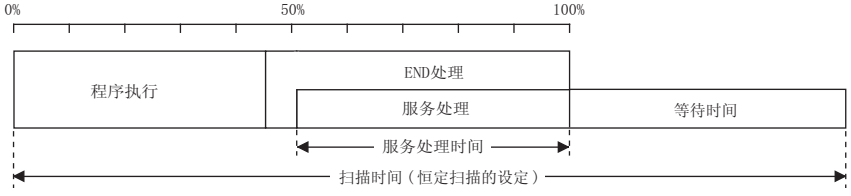
要 点

1. 设定了恒定扫描时，如果选择了 “在恒定扫描设定的等待时间执行服务处理”，可以提高服务处理的效率。

   • 在恒定扫描设定的等待时间执行服务处理时



   • 根据扫描时间的比例执行时（设定为扫描时间的比例 =50% 时）



2. 即使是在无等待时间时，也将进行服务处理 (0.2ms)。因此，等待时间为 0.2ms 以下时，有可能会超过恒定扫描时间。

(4) 注意事项

进行服务处理设定时的注意事项如下所示。

1) 对于服务处理过程中的以下功能，即使进行了指定服务处理时间设定，也会发生超出指定时间使扫描时间延迟的现象，应加以注意。
   - 运行中写入
   - T/C 设定值变更
   - 局部软元件监视
   - 程序内存的备份
   - 至文件寄存器的写入以及读出（写入容量及读出容量较大等情况下将发生扫描时间的延迟。）
   - 智能功能模块的缓冲存储器写入以及读出（写入容量及读出容量较大等情况下将发生扫描时间的延迟。）
   - 至网络模块的访问
     a) 诊断功能（网络诊断、以太网诊断、CC-Link 诊断）
     b) 监视功能（模块访问软元件、直接链接软元件）

2) 设定了较多的服务处理次数时，如果同时接受多个请求，有可能大幅度延长扫描时间，应加以注意。

3) 如果设定的服务处理时间大幅度短于扫描时间时，将会大大降低服务处理的响应性能。在设定服务处理时间时，应考虑外围设备的超时时间。

4) 在服务处理时间指定中，与实际的处理时间相比，将会发生 -20μs ～ +30μs 的误差。

6.26 软元件初始值

基本 注 6.95

(1) 软元件初始值

基本 注 6.96

软元件初始值是指将程序使用的数据非程序化并登录到软元件、智能功能模块 / 特殊功能模块注 6.96 的缓冲存储器的功能。

过程 注 6.96

(2) 软元件初始值的用途

冗余 注 6.96

使用软元件初始值，可省略通过初始化处理程序向软元件进行数据设定的程序。

通用 UD 注 6.96

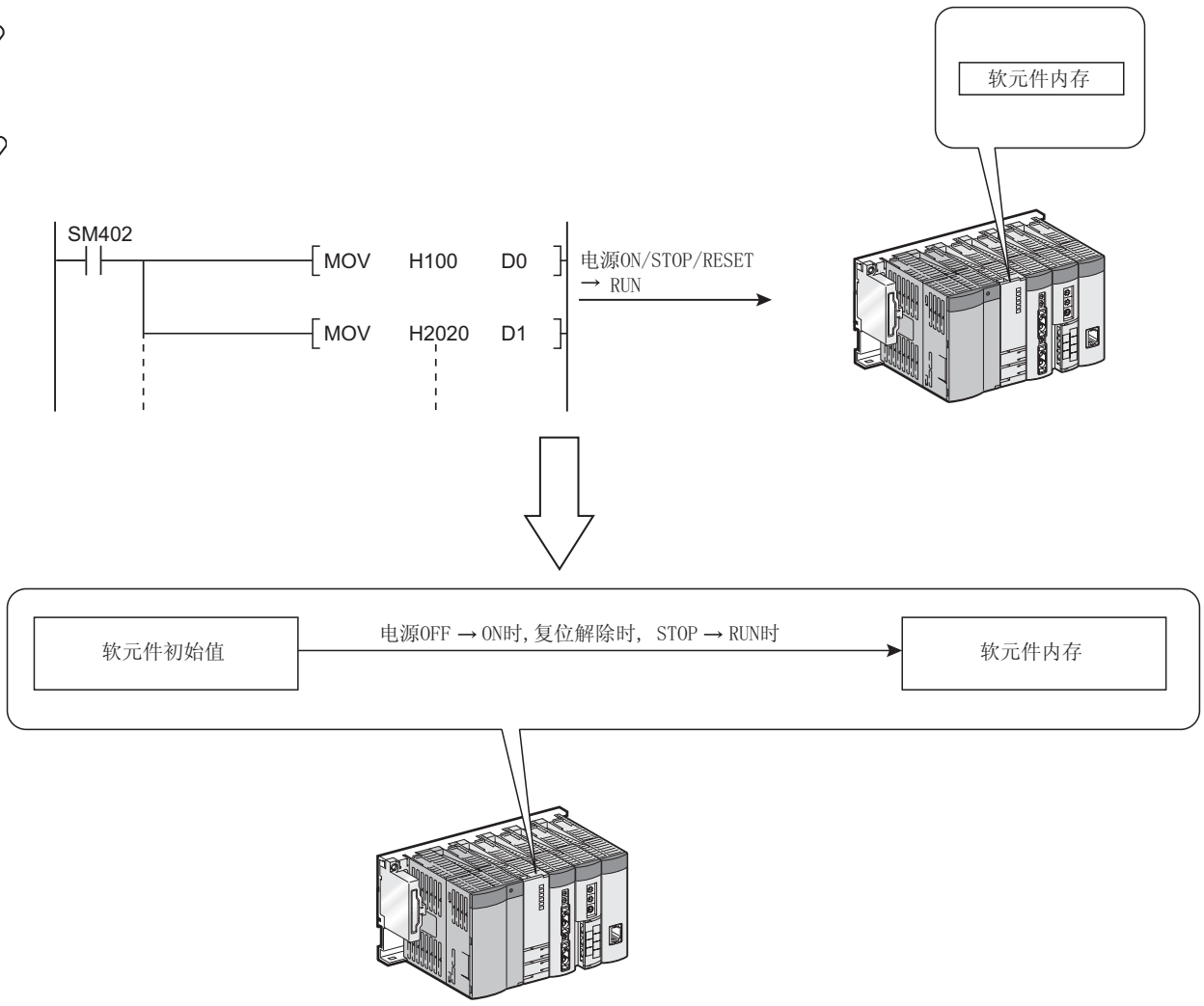


图 6.106 利用初始程序进行数据设定

基本 注 6.95

在基本模式 QCPU 中使用软元件初始值时，请先确认 CPU 模块以及 GX Developer 的版本。
(☞ 附录 4.1)

基本 注 6.96　过程 注 6.96　冗余 注 6.96

通用 UD 注 6.96

在基本模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 中，不能使用 AnS/A 系列对应的特殊功能模块。

(3) 软元件初始值被写入所指定的软元件的时间

在 CPU 模块中，可编程控制器电源 OFF → ON 时、复位解除时或者 CPU 模块 STOP → RUN 时，将所指定的软元件初始值文件的数据写入指定的软元件、智能功能模块的缓冲存储器中。

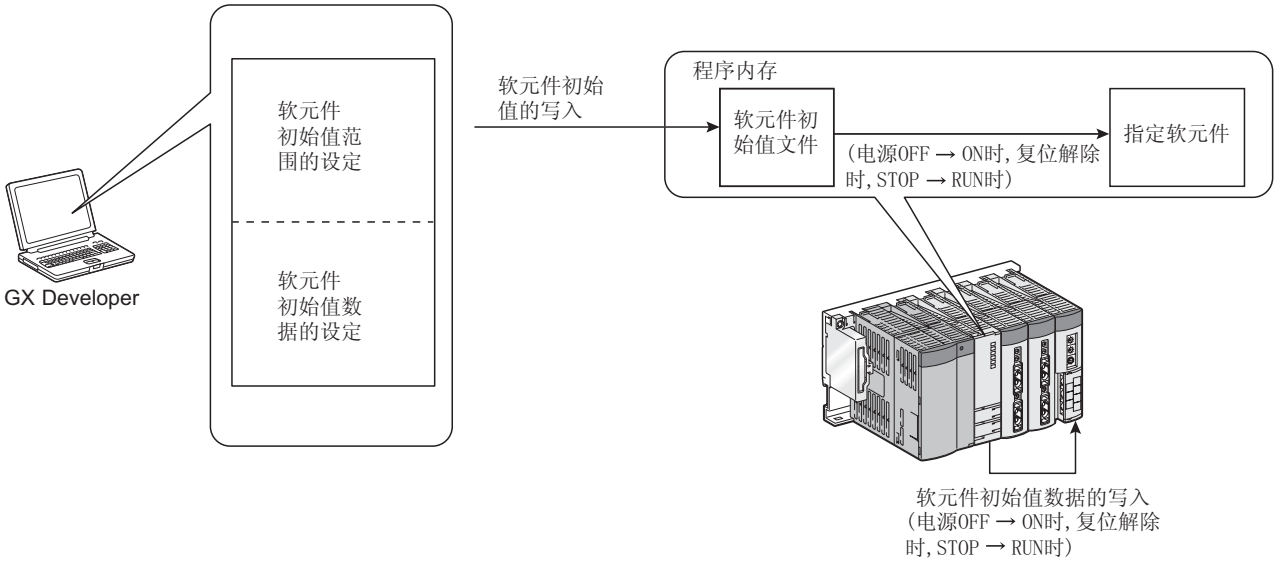


图 6.107 软元件初始值写入流程

(4) 可使用的软元件 *1

软元件初始值可使用的软元件如下所示。

- 计时器的当前值 (T)
- 累积计时器的当前值 (ST)
- 计数器的当前值 (C)
- 数据寄存器 (D)
- 特殊寄存器 (SD)
- 链接寄存器 (W)
- 链接特殊寄存器 (SW)
- 文件寄存器 (R)
- 文件寄存器 (ZR)
- 智能功能模块软元件 (U□\G□)
- 链接直接软元件 (J□\W□、J□\SW□)

*1: 关于可使用的范围，请参照 9.1 节。

(5) 使用软元件初始值的步骤与设定

使用软元件初始值时，需要事先用 GX Developer 编制软元件初始值数据，并将其存储到 CPU 模块的程序内存、标准 ROM 或者存储卡中注 6.97，作为软元件初始值文件。

基本
注 6.97

- 在 GX Developer 的工程数据一览中，添加软元件初始值数据。
  显示软元件初始值范围设定画面后，设定软元件初始值的范围。
  一个范围设定最多可设定 8000 点。
- 在 GX Developer 的工程数据一览中，添加软元件内存的数据。
  显示软元件内存画面后，在上述设定的软元件初始值范围中设定软元件初始值数据。

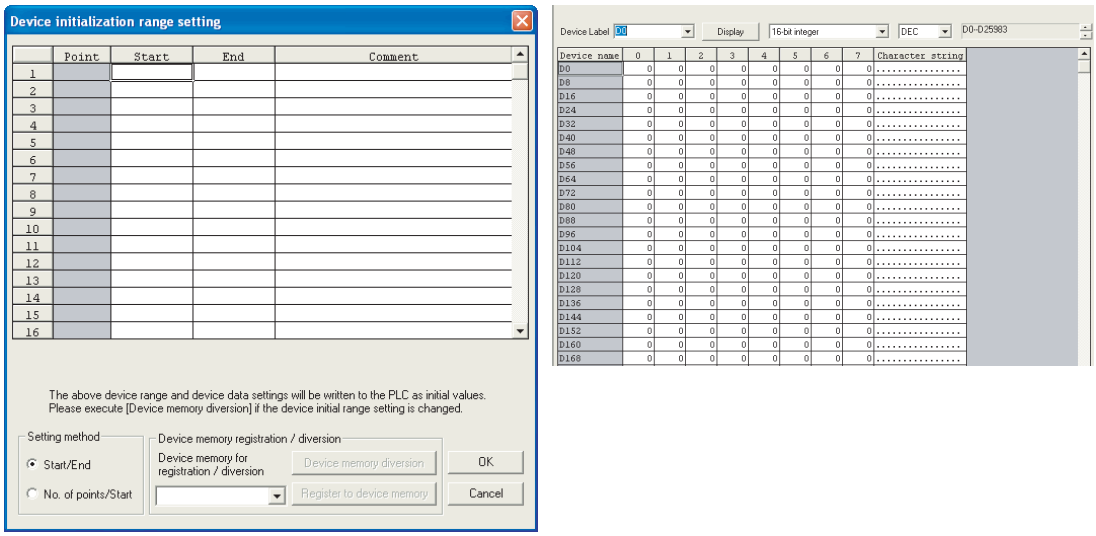


图 6.108 软元件初始值设定画面，软元件内存画面

基本
注 6.97

在基本模式 QCPU 中不能使用存储卡。

• 在可编程控制器参数的可编程控制器文件设定中，设定存储了所使用的软元件初始值数据的文件名。

1）基本模式 QCPU
在可编程控制器参数的可编程控制器文件设定中，将软元件初始值设为“使用”。

2）高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 的情况
在可编程控制器参数的 LPC 文件设定中，设定存储了所使用的软元件初始值数据的文件名。

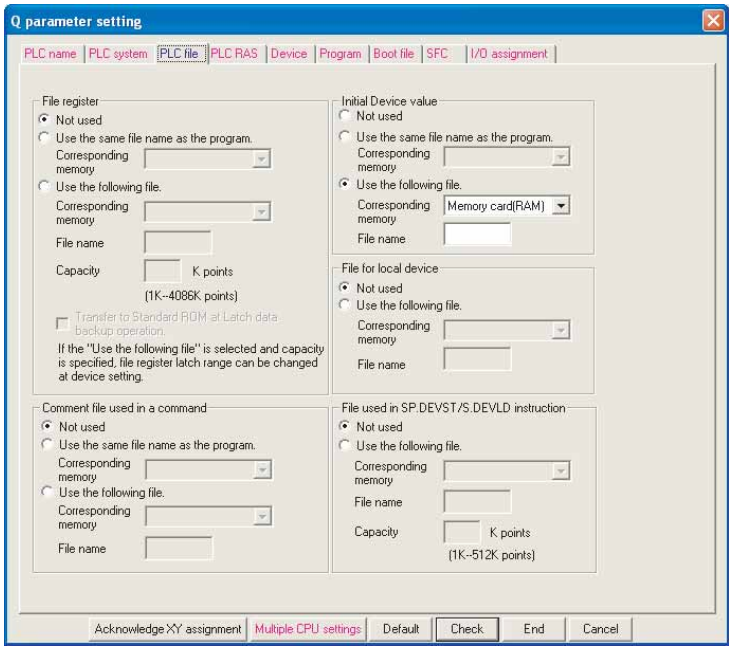


图 6.109 可编程控制器文件设定画面

• 将利用 GX Developer 设定的软元件初始值以及参数写入 CPU 模块。

(6) 使用软元件初始值的注意事项

(a) 软元件初始值与锁存范围重复时
软元件初始值与锁存范围重复时，以软元件初始值为优先。
因此，锁存范围的数据在电源 OFF→ON 时，将被改写为软元件初始值数据。

(b) STOP→RUN 时不想设定的区域
在 STOP→RUN 时也将反映软元件初始值。
STOP→RUN 时不想设定的区域中（电源 OFF→ON 时设定并且在程序中变化的数据），不能使用软元件初始值。
请在主程序中通过 NOV 指令等，编写将初始值设定到指定软元件的程序。
此外，智能功能模块的情况下，使用 TO 指令写入缓冲存储器。

(c) 需要进行模块同步设定的软元件
在软元件初始值范围设定中指定以下软元件时，请通过可编程控制器参数的可编程控制器系统设定设为“模块同步设定”。
如果没有进行模块同步设定，软元件初始值可能无法正常设定到对象模块中。

- 智能功能模块软元件 (U\G)
- 链接直接软元件 (J\W, J\SW)

备 注

关于软元件初始值范围的设定、软元件初始值数据的设定操作以及将软元件初始值写入到 CPU 模块的操作的详细内容，请参照以下手册。
☞GX Developer 操作手册

基本 注 6.98
高性能 注 6.98
过程 注 6.98
冗余 注 6.98

6.27 电池长寿功能

(1) 关于电池长寿功能

电池长寿功能是指，通过使 CPU 模块本体的电池仅进行时钟数据的数据保持，从而延长电池使用寿命的功能。

如果使用电池长寿功能，在电源 OFF 时以及复位解除时，时钟数据以外的数据将全部被初始化。

表 6.55 通过电池保持的数据的初始化内容

通过电池保持的数据			初始化内容
故障历史记录			使故障历史记录件数为 0
锁存软元件 (L)			清零
锁存范围的软元件			清零
标准 RAM			格式化（清零）
	标准 RAM 中分配的文件寄存器	设置为 “使用与程序相同的文件名”	删除文件
		设置为 “使用以下文件”	删除文件。（电源 ON 时及复位解除时重新生成文件。（数据将被清零））

基本 注 6.98
高性能 注 6.98
过程 注 6.98
冗余 注 6.98

在基本模式 QCPU、高性能模式 QCPU、过程 CPU、冗余 CPU 中，不能使用电池长寿功能。

(2) 电池长寿功能的设定

电池长寿功能的设定是在可编程控制器参数的 I/O 分配设定中进行。

1) 进行 I/O 分配设定。

2) 选择 开关设定 按钮。

3) 在 CPU 插槽的开关 3 中输入 0001H。（即使进行至他号机的 CPU 插槽的输入，也将被忽略。）

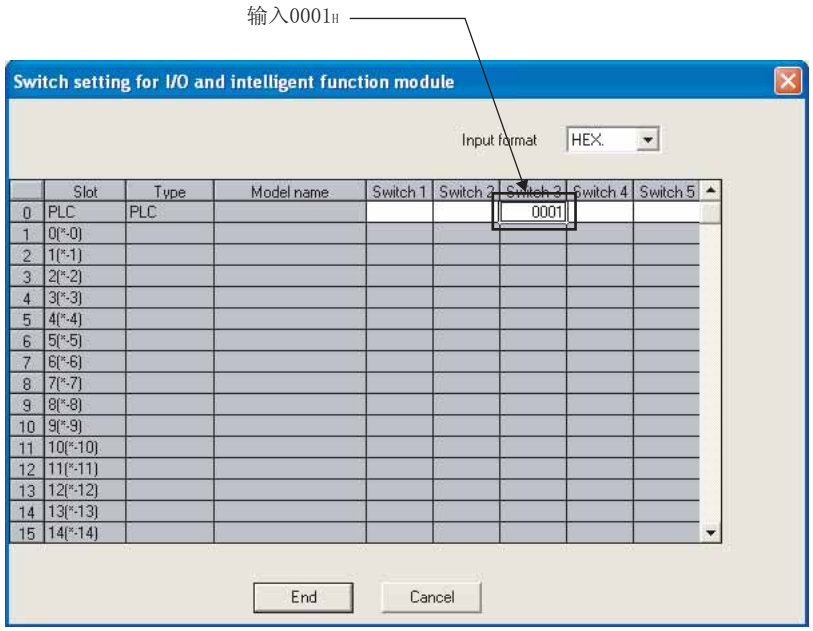


图 6.110 开关设定画面

(3) 电池寿命

关于使用了电池长寿功能时的 CPU 模块本体的电池寿命，请参照以下手册。

☞ QCPU 用户手册（硬件设计 / 维护点检篇）

基本 注 6.99

6.28 内存检查功能

高性能 注 6.99

内存检查功能是指，检查是否由于过度的电气噪声等导致 CPU 模块的内存的内容被改写的功能。
根据各 CPU 模块的不同，可检查的对象内存也不一样。成为对象的内存如表 6.56 所示。

表 6.56 可检查的对象内存

CPU 型号	对象内存	
	程序内存	软元件内存
过程 CPU	○	×
冗余 CPU	○	×
通用型 QCPUQ	○	○

○：进行检查； ×：不进行检查

(1) 各 CPU 模块的内存检查功能的概要

(a) 过程 CPU、冗余 CPU 的情况

以 CPU 模块的程序内存中写入的用户程序、参数等的数据为基准，在由 STOP 切换为 RUN 时以及 END 处理时，进行执行中的程序内存的数据与基准数据是否一致的检查。(☞ 本节 (3))
基准数据与程序内存的数据不相同时，将作出程序内存的数据已被改写的判断，变为停止出错的 RAM ERROR(出错代码 :1106) 状态。

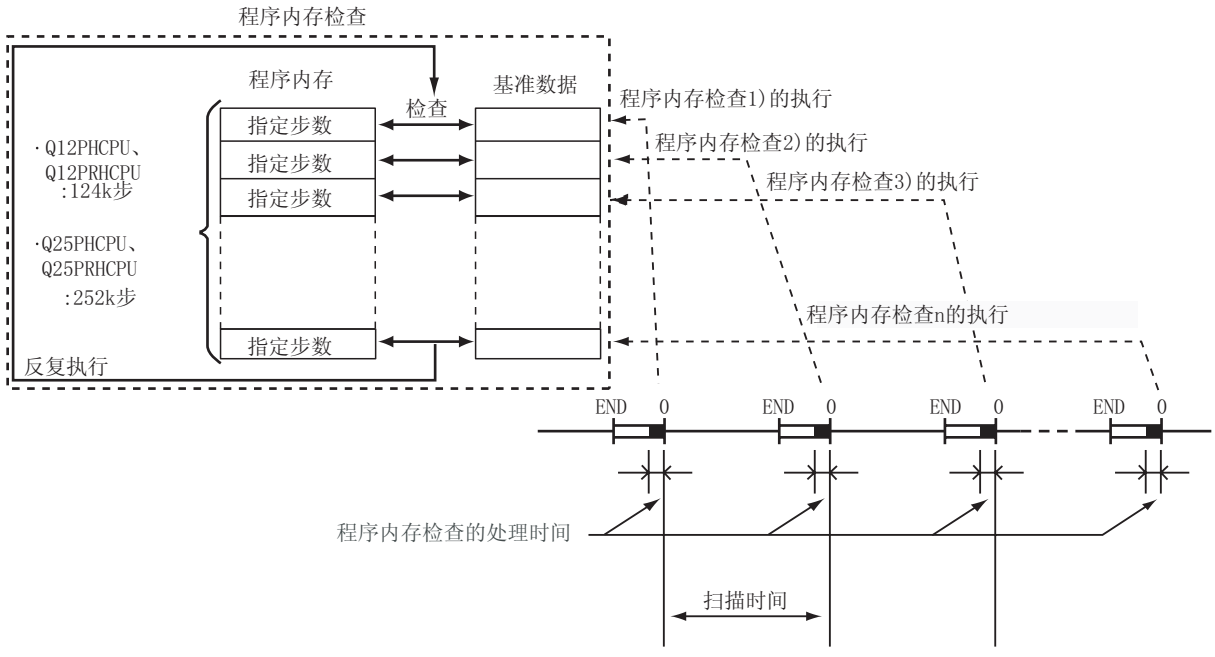


图 6.111 过程 CPU、冗余 CPU 的程序内存检查的概要

基本 注 6.99　高性能 注 6.99

在基本模式 QCPU、高性能模式 QCPU 中，没有程序内存检查功能。

(b) 通用型 QCPU 的情况

以 CPU 模块的程序内存中写入的用户程序数据为基准，自动进行执行中的程序内存的数据与基准数据是否一致的检查。

基准数据与程序内存的数据不相同时，将变为停止出错的 RAM ERROR(出错代码:1160)状态。

此外，对于软元件内存，在不能作为软元件读出时，将变为停止出错的 RAM ERROR(出错代码:1161)状态。

(2) 用于执行程序内存检查的设置

(a) 过程 CPU、冗余 CPU 的情况

执行程序内存检查时，在可编程控制器参数的可编程控制器 RAS 设置(2)中，勾选“Check program memory(检查程序内存)”，设置“Capacity to be checked at one time(1 次检查的容量)”。

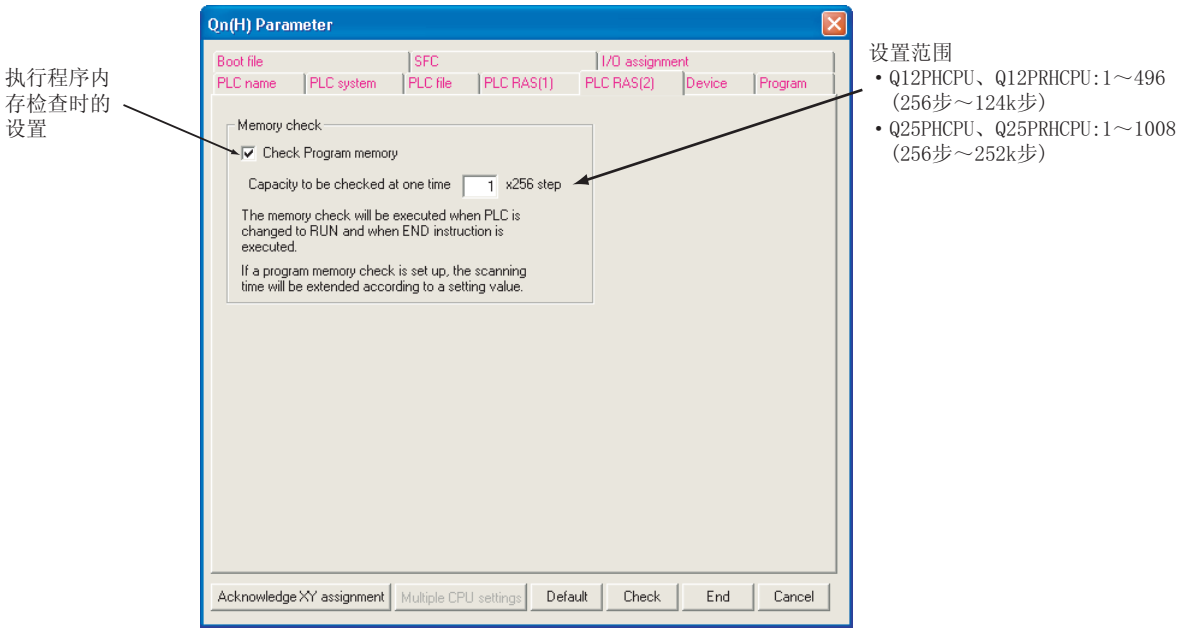


图 6.112 可编程控制器 RAS(2)设置画面

对于 Q12PRHCPU、Q25PRHCPU，通过设置为“执行程序内存检查”，可以对控制系统 CPU 与待机系统 CPU 均进行程序内存检查。

但是，对于备份模式的待机系统 CPU，与 1 次检查的容量的设置无关，固定为“1(256 步)”进行内存检查。

(b) 通用型 QCPU 的情况

不存在用于执行程序内存及软元件内存检查的设定。

(3) 内存检查的执行

(a) 过程 CPU、冗余 CPU 的情况

内存的检查是在 “由 STOP 切换为 RUN 时”以及运行中的 “END 处理时”进行。

- 由 STOP 切换为 RUN 时 : 进行程序内存的全部区域的检查。
- END 处理时 : 以在可编程控制器参数的可编程控制器 RAS 设置 (2) 中设置的 (1 次检查的容量) × 256(步) 逐次进行检查。

(b) 通用型 QCPU 的情况

程序内存、软元件内存、参数的检查是在以下时机自动进行。

1) 程序内存

a) 程序 : 执行程序时

b) 参数 :

- 电源源 OFF → ON 时
- 复位解除
- 可编程控制器写入后的 STOP → RUN 时

2) 软元件内存 : 读出软元件时

(4) 程序内存检查的处理时间

(a) 过程 CPU、冗余 CPU 的情况

1) 由 STOP 切换为 RUN 时

如果进行程序内存检查，由 STOP → RUN 的切换将发生如下所示的内存检查的处理时间的延迟。

- Q12PHCPU、Q12PRHCPU: 434ms
- Q25PHCPU、Q25PRHCPU: 882ms

2) 运行中的 END 处理时

如果执行程序内存检查，扫描时间将延迟。

在可编程控制器参数的可编程控制器 RAS 设置 (2) 中，对 1 次检查的容量应在考虑扫描时间的延迟时间的基础上进行设置。

扫描时间的延迟时间应按下述公式进行计算 :

$$(\text{扫描时间的延迟时间}) = 3.5 \times \frac{(\text{1次检查的容量})^{*1} \times 256}{1024} \ (\text{ms})$$

例如，在可编程控制器参数的可编程控制器 RAS 设置 (2) 中将 1 次检查的容量设置为 4 时的处理时间为 3.5 × 4 × 256 ÷ 1024 = 3.5(ms)。

*1 : 表示在可编程控制器参数的可编程控制器 RAS 设置 (2) 中设置的 “1 次检查的容量”。
(☞本节 (2))

(b) 通用型 QCPU 的情况

无需用于内存检查的处理时间。

(5) 注意事项

在过程 CPU、冗余 CPU 中的内存检查中，应注意如下所示的项目。

(a) 出错检测的最大延迟时间

从发生程序内存的数据的改写时起，至检测出数据的改写为止，将有如下式所示的最大延迟时间。

由于在检测出错误之前顺控程序处于执行状态，在检测出 RAM ERROR 错误之前有可能检测出其它的错误，或者由于程序内存化导致异常动作的发生。

$$\text{出错检测的最大延迟时间} = \frac{\text{(最大程序内存容量)}}{\text{(1次检查的容量)}^{*3} \times 256} \times \text{(扫描时间)(ms)}$$

例如，在 Q12PHCPU 中将 1 次检查的容量设置为 4，扫描时间为 10ms 时的最大延迟时间为 (124 × 1024) ÷ (4 × 256) × 10=1240(ms)。

*2 : 各 CPU 模块的最大程序内存容量如下所示：
Q12PHCPU、Q12PRHCPU:124k 步 (124 × 1024 步)
Q25PHCPU、Q25PRHCPU:252k 步 (252 × 1024 步)

程序内存的格式化时，在设置了用户设置的系统区域的情况下，程序内存容量将减少相当于所设置的容量的量。

关于用户设置的系统区域、程序内存容量的确认方法，请参阅 5.2.2 项。

*3 : 表示在可编程控制器参数的可编程控制器 RAS(2) 设置中设置的 “1 次检查的容量”。
(☞ 本节 (2))

冗余
注6.100

(b) 执行内存检查时不能使用的指令
在可编程控制器参数中将内存检查设置为有效时，如果执行下述指令，将发生OPERATION ERROR(出错代码： 4105)。注 6.100

- PLOADP 指令
- PUNLOADP 指令
- PSWAPP 指令

(c) 可编程控制器参数的有效条件
在将可编程控制器参数写入到 CPU 模块中后，如果进行了以下操作，内存检查的执行 / 不执行将生效。

- 可编程控制器的电源的重新启动
- CPU 模块的复位

(d) 执行 COM 指令时的内存检查
执行了 COM 指令时不进行内存检查。

冗余
注 6.100

冗余 CPU 中没有 PLOADP、PUNLOADP、PSWAPP 指令。

6.29 至标准 ROM 的锁存备份功能

基本 注 6.101
高性能 注 6.101
过程 注 6.101
冗余 注 6.101

(1) 关于至标准 ROM 的锁存备份功能

至标准 ROM 的锁存备份功能是指，长时间停止系统时，在不能使用电池的状况下将软元件数据及故障历史记录等锁存数据保持（备份）到标准 ROM 中的功能。由此，可以实现电池的长寿化。

备 注

如果执行至标准 ROM 的锁存数据备份功能，与电池长寿化功能的参数设定无关，电池长寿化功能将生效。

此外，进行了备份数据的还原后，电池长寿化功能将被切换为无效状态。

电池长寿化的有效 / 无效状态可以通过 SD119（电池长寿化有效 / 无效）进行确认。

关于电池长寿化功能，请参照 6.27 节。

(2) 备份对象数据

成为备份对象的数据如表 6.57 所示。

表 6.57 成为备份对象的数据一览表

备份的数据	数据内容	备注
软元件数据	• 内部用户软元件（M、L、B、F、V、T、ST、C、D、W） • 变址寄存器（Z）/ 通用运算寄存器（Z） • 文件寄存器（R、ZR）	对于文件寄存器，仅在使用标准 RAM 的文件寄存器时进行备份。
故障历史记录	至标准 ROM 的锁存数据备份之前为止的故障历史记录信息	-----
SFC 程序继续运行启动信息	用于进行 SFC 程序继续运行启动信息。	-----
追踪设定（采样追踪文件）	通过采样追踪功能创建的追踪条件设定、追踪数据设定。	在以下情况下不能进行备份。 • 将追踪设定设定到内存卡中。 • 追踪设定未登录到 CPU 中。

基本 注 6.101　高性能 注 6.101　过程 注 6.101
冗余 注 6.101

在基本模式 QCPU、高性能模式 QCPU、过程 CPU、冗余 CPU 中，不能使用至标准 ROM 的锁存数据备份功能。

要 点

执行文件寄存器的备份时，应注意以下几点。

- 只有在设定了使用标准 RAM 的文件寄存器的情况下才进行备份。
- 应在可编程控制器的可编程控制器文件设定中将“锁存数据备份操作时传送至标准 ROM”设置为有效。

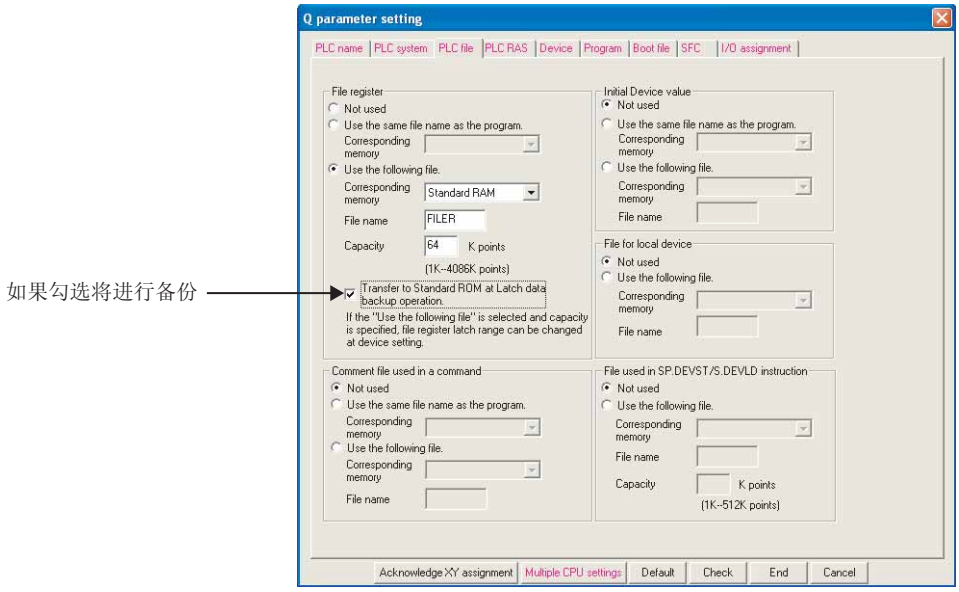


图 6.113 文件寄存器设定画面

(3) 备份数据存储用文件容量

执行至标准 ROM 的锁存数据备份时，用于存储各备份数据的文件是由系统自动地在标准 ROM 中创建。

在至标准 ROM 的锁存数据备份功能中使用的存储用文件的文件容量如表 6.58 所示。

表 6.58 存储用文件的文件容量一览表

备份的数据		文件容量
软元件	• 内部用户软元件 (M、L、B、F、V、T、ST、C、D、W) • 变址寄存器 (Z) / 通用运算寄存器 (Z)	110818 字节 *1 (默认的软元件分配设定时)*2
故障历史记录		
SFC 程序继续运行启动信息		
文件寄存器 (R、ZR)*3		64 + 2 × 文件寄存器点数
追踪信息 (采样追踪文件)*4		16 + 采样追踪文件容量

*1: Q02UCPU 中的文件容量为 87138 字节。
*2: 通过参数的设定增减。
*3: 仅在备份文件寄存器时，创建存储用文件。
*4: 仅在进行了追踪登录的情况下，才创建存储用文件。

(4) 执行方法

(a) 通过触点执行

1) 设定方法

在可编程控制器参数的可编程控制器系统设定中，设定至标准 ROM 的锁存数据备份开始触点。（可用于触点的软元件为 X、M、B。）

2) 执行方法

通过触点的上升沿（OFF → ON 时），开始备份。

备份结束后，CPU 模块的 BAT.LED 将绿灯闪烁，变为可以进行电源 OFF 的备用状态。

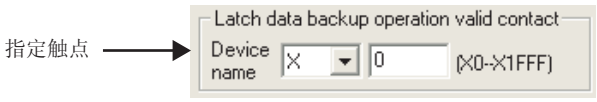


图 6.114 至标准 ROM 的锁存数据备份开始触点的设定画面

3) 注意事项

a) 由于备份的各数据为触点 ON 时（END 处理时）的数据，因此在执行本操作后，如果不进行电源的重新接通或者复位解除，则不能重新变为 RUN 状态。

b) 由于在执行 END 指令时对至标准 ROM 的锁存数据备份开始触点的状态进行检查，因此即使在 1 个扫描处理内对触点进行 ON → OFF → ON、OFF → ON → OFF 的操作，也不进行备份。

c) 在如下所示的情况下，如果不对至标准 ROM 的锁存数据备份开始触点进行 OFF 后重新 ON 操作，将不开始备份。

- 将至标准 ROM 的锁存数据备份开始触点设定为 X，通过触点的 OFF → ON 执行备份后，在未将触点 OFF 的状况下进行了电源 OFF → ON 或者复位解除时。
- 将至标准 ROM 的锁存数据备份开始触点设定为 M、B，通过触点的 OFF → ON 执行了备份时。

(b) 通过远程操作执行

1) 执行方法

通过 GX Developer 的［在线］→［锁存数据备份操作］执行。

备份结束后，CPU 模块的 BAT.LED 将绿灯闪烁，变为可以进行电源 OFF 的备用状态。

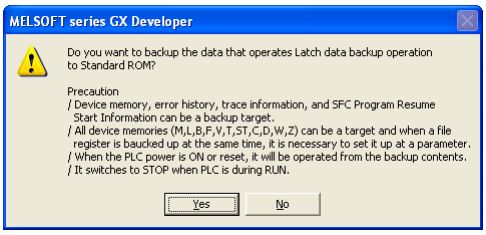


图 6.115 远程操作执行画面

备份的各数据是执行远程操作时的数据。

(5) 备份数据的还原

备份的各数据通过以下操作自动还原。

- 电源 OFF → ON 时
- 复位解除时

在备份操作时通过 SM676（还原反复执行指定）指定执行备份后的还原是仅执行 1 次，还是在上述各操作时执行。

表 6.59 SM676 的状态及还原动作

备份操作时的 SM676 的状态	还原的动作
在 SM676=OFF 的状态下执行备份操作	在执行备份操作后的电源 OFF → ON 或者复位解除时仅执行 1 次还原。
在 SM676=ON 的状态下执行备份操作	执行备份操作后，在电源 OFF → ON 或者复位解除时执行还原。在执行备份数据的删除或者下一个锁存数据备份之前，反复进行还原。

备份数据的还原结束后，CPU 模块的 BAT.LED 将亮绿灯（5 秒）。

要 点

如果参数设定中设置的各软元件点数与备份时的各软元件点数不相同，备份数据的还原时，将发生 RESTORE ERROR（出错代码 :2220），备份数据的还原将不能正常结束。（在下一次的电源 OFF → ON 以及复位解除时再次执行还原。）

为了正常结束还原，应执行以下某个操作。

- 恢复为参数备份时的数据。
- 删除备份数据。
- 重新进行备份。

(6) 备份数据的删除

备份数据的删除是在 GX Developer 的［在线］→［锁存数据备份操作］进行。（在 RUN 状态下不能执行。应置于 STOP 状态后再执行。）

此外，通过删除备份数据，还可以使特殊寄存器 (SD671 ～ 675) 的信息初始化（清零）。

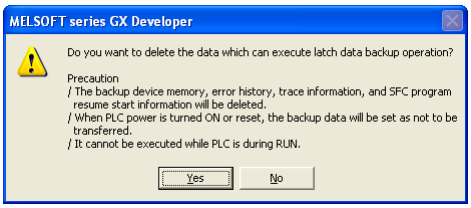


图 6.116 删除备份数据时的画面

(7) 通过特殊继电器、特殊寄存器进行确认的方法

至标准 ROM 的锁存数据备份的执行、还原操作的状态可以通过 SM671、SM676、SD671 ～ 679 进行确认。

(8) 注意事项

执行锁存数据备份时的注意事项如下所示。

1) 在锁存数据的备份过程中不要进行 CPU 模块的电源 OFF 及复位操作。如果进行了电源 OFF 及复位操作，将发生 RESTORE ERROR(出错代码 :2221)，将不执行备份数据的还原。(将删除备份数据。)

2) 即使存在有备份数据，如果设定了软元件初始值，软元件初始值将优先。
因此，对于进行了软元件初始值设定的软元件，备份数据反映后，将被改写为软元件初始值的软元件数据。

3) 即使使用了锁存软元件以及锁存范围设定，备份数据也优先。
因此，备份结束后，即使更改了锁存软元件以及锁存范围设定的数据，在电源 OFF → ON 或者复位解除时将被改写为备份的数据。

4) 进行了局部软元件范围设定的软元件不进行备份。进行电源 OFF → ON 或者复位解除时，进行了局部软元件范围设定的软元件将被初始化(清零)。

5) 如果至标准 ROM 的写入次数超过了 10 万次 (FLASH ROM ERROR(出错代码 :1610))，有可能不能正常进行备份，应加以注意。

6) 只有通过远程操作进行备份数据的删除，或者对备份数据的存储目标内存(标准 ROM) 进行格式化，才可以删除备份数据。

7) 在锁存数据的备份过程中，不能执行以下操作，应在备份操作结束之后再执行以下操作。
如果执行了操作 GX Developer 中将显示出错。
- 可编程控制器内存格式化(仅标准 ROM)
- 通过远程操作进行锁存数据备份
- 运行中写入(梯形图模式、文件的运行中写入、功能块)
- 程序内存的 ROM 化
- 可编程控制器写入(快闪 ROM)

6.30 至标准 ROM 的软元件数据的写入 / 读出

基本 注 6.102
高性能 注 6.102
过程 注 6.102
冗余 注 6.102

(1) 关于至标准 ROM 的软元件数据的写入 / 读出

至标准 ROM 的软元件数据的写入 / 读出是指，将任意的软元件数据写入到标准 ROM 中的功能。通过将运算中使用的固定值及运算结果写入到标准 ROM 中，可以防止电池电量过低时的数据丢失。此外，对于写入到标准 ROM 中的数据，可以使用指令在任意的时机进行读出。

(2) 执行方法

将软元件数据写入到标准 ROM 中是通过 SP.DEVST 指令进行。此外，通过 S(P).DEVLD 指令，可以将写入到标准 ROM 中的软元件数据读取到指定的软元件中。

(3) 可写入到标准 ROM 中的软元件

可写入到标准 ROM 中的软元件如表 6.60 所示。

表 6.60 可写入到标准 ROM 中的软元件一览及区域

软元件名称	存储目标区域
X、Y、M、L、B、F、V、T、ST、C、D、W、SM、SD、SB、SW	标准 ROM(软元件数据存储目标文件)
R、ZR	

基本 注 6.102　高性能 注 6.102　过程 注 6.102
冗余 注 6.102

在基本模式 QCPU、高性能模式 QCPU、过程 CPU、冗余 CPU 中，不能使用至标准 ROM 的软元件数据的写入 / 读出。

(4) 设定方法

在可编程控制器参数的可编程控制器文件设定中，设定标准 ROM 中存储软元件数据的区域。

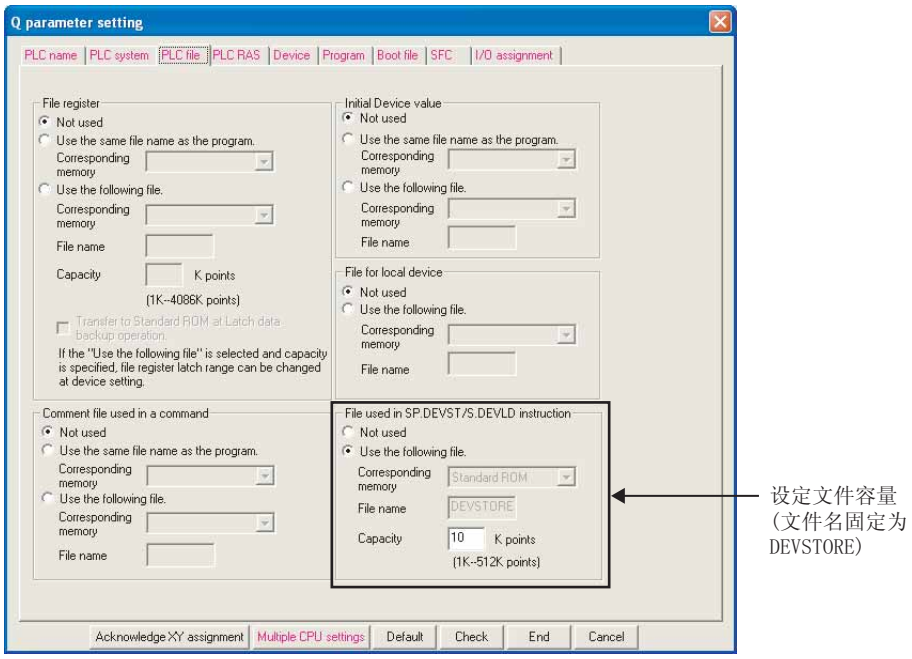


图 6.117 软元件数据存储用文件的设定画面

备 注

关于指令的详细内容，请参照以下手册。

☞ QCPU(Q 模式)/QnACPU 编程手册（公共指令篇）

备忘录

第 7 章　与智能功能模块的通讯

(1) 智能功能模块

智能功能模块是指将 I/O 模块不能处理的模拟量、高速脉冲等在 CPU 模块上进行处理的模块。

例如，通过智能功能模块的模拟 / 数字转换模块可将模拟量转换为数字值加以使用。

(2) 与智能功能模块的通讯

智能功能模块中具含有存储器（缓冲存储器），用来存储外部写入获取的数据以及向外部输出的数据。

CPU 模块可从智能功能模块的缓冲存储器中进行数据的写入 / 读出。

7.1　CPU 模块与智能功能模块的通讯

CPU 模块可通过以下方法与智能功能模块进行通讯。

- 利用 GX Configurator 进行初始设置设定、自动刷新设置设定 *1(☞ 7.1.1 项)
- 利用软元件初始值进行初始设置设定 (☞ 7.1.2 项)
- FROM/TO 指令 (☞ 7.1.3 项)
- 智能功能模块软元件 (☞ 7.1.4 项)
- 智能功能模块专用指令 *1(☞ 7.1.5 项)

*1: 在 AnS/A 系列对应的特殊功能模块中不能使用。

与智能功能模块的通讯方法与通讯时间，如表 7.1 所示。

表 7.1 与智能功能模块的通讯时间

与智能功能模块的通讯方法		通讯时间				
		电源 OFF→ON	CPU 模块复位	STOP→RUN *3	指令执行	END 处理
GX Configurator *1	初始设定	○	○	○	—	—
	自动刷新设定	—	—	—	—	○
软元件初始值		○	○	○	—	—
FROM/TO 指令 *2		—	—	—	○	—
智能功能模块软元件 *2		—	—	—	○	—
智能功能模块专用指令 *1*2		—	—	—	○	—

○: 执行　—: 不执行

*1: 在 AnS/A 系列对应的特殊功能模块中不能使用。

*2: 表示使用了智能功能模块软元件、FROM/TO 指令、智能功能模块专用指令的程序。

*3: 表示对 RUN/STOP 开关（基本模式 QCPU、通用型 QCPU 时为 RUN/STOP/RESET 开关）进行了以下操作的情况：STOP→RUN(RUN LED 闪烁)→STOP→RUN。

要 点

1. 与智能功能模块通讯所使用的数据（初始设定等）存储在 CPU 模块中。
   关于 CPU 模块中的存储区，请参照 5.1.1 项以及 5.2.1 项
2. 通过电源 OFF → ON 或者 CPU 模块的复位使 GX Configurator 中设定的初始设定生效时，应将 RUN/STOP 开关（ 基本模式 QCPU、通用型 QCPU 时为 RUN/STOP/RESET 开关）置于 RUN 状态并进行电源 OFF → ON 或者 CPU 模块的复位。

7.1.1 利用 GX Configurator 进行初始设定、自动刷新设定

将智能功能模块对应的 GX Configurator 添加到 GX Developer 中，便可进行智能功能模块的初始设定、自动刷新设定。
利用 GX Configurator 进行智能功能模块的初始设定及自动刷新设定时，不需要编写与智能功能模块的通讯程序，便可写入／读出数据。

(1) GX Configurator 的启动

选择 GX Developer 的（工具）→（智能功能模块多功能）→（启动），便可启动 GX Configurator。

(2) 利用 GX Configurator 进行设定

以 A/D 转换模块 Q64AD 的初始设定、自动刷新设定为例进行说明。

(a) 初始设定

Q64AD 的初始设定有以下 4 种。

- A/D 转换许可／禁止设定
- 采样／平均处理指定
- 时间平均／次数平均指定
- 平均时间／平均次数指定

Q64AD 的初始设定在图 7.1 所示的 GX Configurator 的初始设定画面中进行。

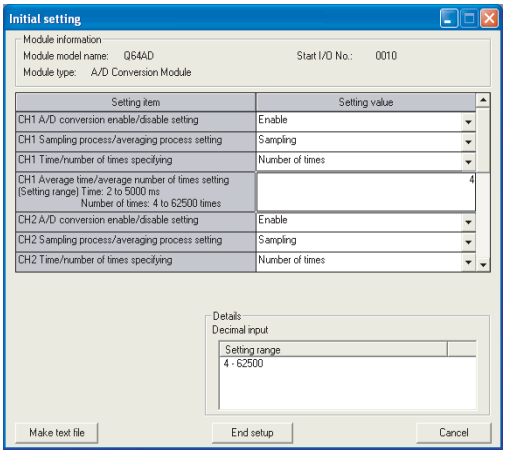


图 7.1 初始设定画面

所设定的初始设定数据存储在 CPU 模块的智能功能模块参数中。

(b) 自动刷新设定

自动刷新设定中，存储以下数据时，设定 CPU 模块的软元件。

- Q64AD 的数字输出
- Q64AD 的最大值 / 最小值
- 出错代码

Q64AD 的自动刷新设定在图 7.2 所示的 GX Configurator 的自动刷新设定画面中进行。

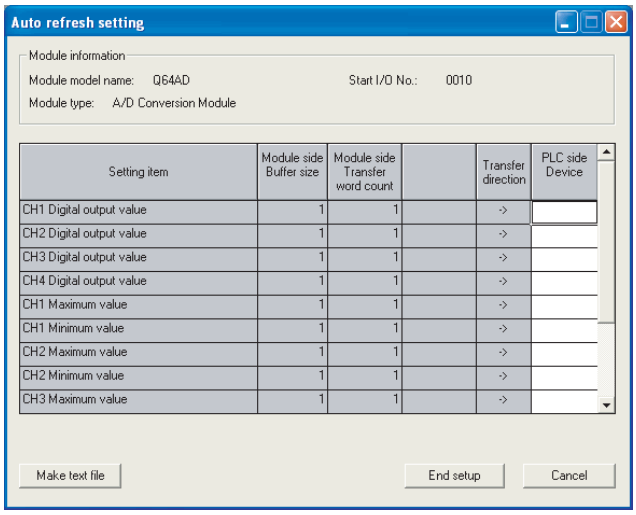


图 7.2 自动刷新设定画面

所设定的自动刷新数据存储在 CPU 模块的智能功能模块参数中。

(3) 关于参数设定个数的限制

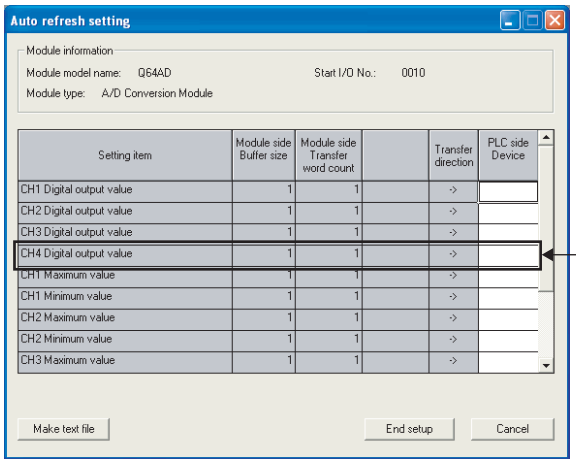
GX Configurator 中设定的参数（初始设定、自动刷新设定）中，可设定的项目数是有限制的。

安装了多个智能功能模块的情况下，在进行 GX Configurator 设定时，全部智能功能模块的参数设定个数的合计不要超过表 7.2 的最大参数设定个数。

表 7.2 在 GX Configurator 中可设定的参数个数

CPU 模块	参数设定个数	
	初始设定	自动刷新设定
Q00JCPU、Q00CPU、Q01CPU、Q02CPU、Q02HCPU、Q06HCPU、Q12HCPU、Q25HCPU、Q12PHCPU、Q25PHCPU、Q12PRHCPU、Q25PRHCPU、MELSECNET/H 远程 I/O 站	512	256
Q02UCPU	2048	1024
Q03UDCPU、Q04UDHCPU、Q06UDHCPU	4096	2048

<< 参数设定个数的思路 >>



该行中设定个数计数为1个。
空栏不计个数。
将该设定画面的全部设定项目相加后，与其它智能功能模块的个数进行合计。

图 7.3 参数设定个数的思路

(4) 注意事项

AnS/A 系列对应的特殊功能模块中，不能使用 GX Configurator 进行设定。

备 注

关于 GX Configurator 的详细内容，请参照所使用的智能功能模块手册。

7.1.2 利用软元件初始值进行初始设定

(1) 软元件初始值

通过使用软元件初始值（☞ 6.26 节），可以利用程序来进行智能功能模块的初始设定。

在 CPU 模块的电源 OFF → ON 时、复位解除时、STOP → RUN 时，所设定的软元件初始值从 CPU 模块被写入到智能功能模块中。

(2) 软元件初始值的设定

利用 GX Developer 进行以下设定。

- 将作为软元件初始值的智能功能模块软元件（☞ 9.5 节）的数据设定到软元件内存中。
- 通过软元件初始值设定，指定作为软元件初始值使用的智能功能模块软元件的范围。

7.1.3 利用 FROM/TO 指令进行通讯

FROM 指令将智能功能模块的缓冲存储器读出的数据存储到指定的软元件中。
TO 指令将指定的软元件的数据写入智能功能模块的缓冲存储器。

备 注

- 关于 FROM/TO 指令的详细内容，请参照以下手册。
  ☞ QCPU(Q 模式)/QnACPU 编程手册（公共指令篇）
- 关于智能功能模块的缓冲存储器的详细内容，请参照所使用的智能功能模块的手册。

7.1.4 利用智能功能模块软元件进行通讯

(1) 智能功能模块软元件

智能功能模块软元件（☞ 9.5 节）是指将智能功能模块的缓冲存储器作为 CPU 模块的软元件进行表达。

与软元件内存一样可通过顺控程序指令来处理智能功能模块的缓冲存储器中存储的数据。

例如，将 “100”写入 I/O 地址号为 X/Y20~2F 的智能功能模块的缓冲存储器地址 0 时，可进行以下编程。

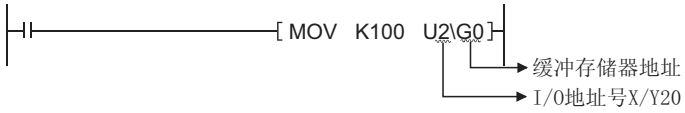


图 7.4 智能功能模块软元件的使用方法

(2) 与 FROM/TO 指令的不同点

智能功能模块软元件可作为 CPU 模块的软元件使用，因此可以用一个指令同时进行从智能功能模块读出以及数据处理。

例如，进行以下编程便可从智能功能模块读出并进行数据加法运算，与将结果存储到 D2 中的处理一样。

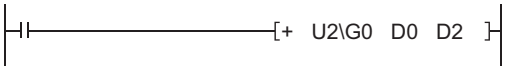


图 7.5 智能功能模块软元件的使用示例

由此可以减少程序整体的步数。

指令处理时间是指令的执行时间加上智能功能模块存取时间的和。

要 点

智能功能模块软元件在每次执行指令时向智能功能模块进行存取。
在顺控程序内使用多个智能功能模块软元件，对缓冲存储器的数据进行写入 / 读出时，请使用 FROM/TO 指令在程序的某处进行写入 / 读出。

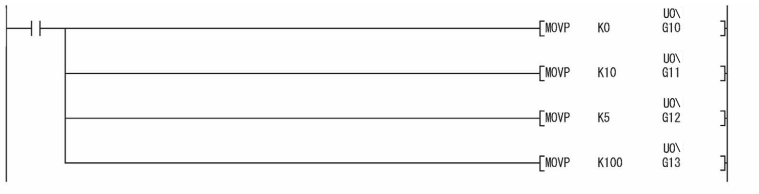


图 7.6 多次使用智能功能模块软元件写入的情况。

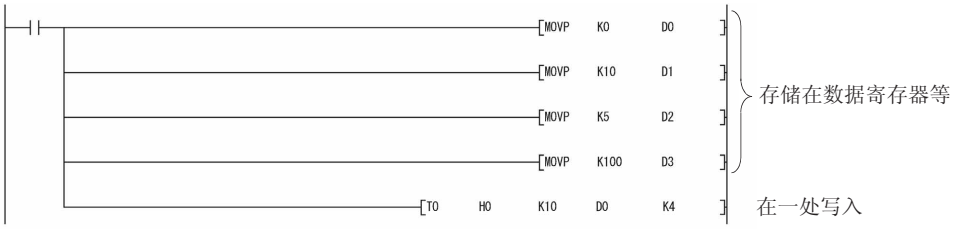


图 7.7 使用 TO 指令在程序的一处写入的情况

7.1.5 利用智能功能模块专用指令进行通讯

(1) 智能功能模块专用指令

智能功能模块专用指令是指在使用智能功能模块的功能的基础上，使编程容易化的指令。

(a) 串行口通讯模块专用指令（OUTPUT 指令）示例

使用 OUTPUT 指令，可以无需识别串行口功能通讯模块的缓冲存储器地址，通过无序协议与对象设备进行通讯。

[通过 OUTPUT 指令进行发送]

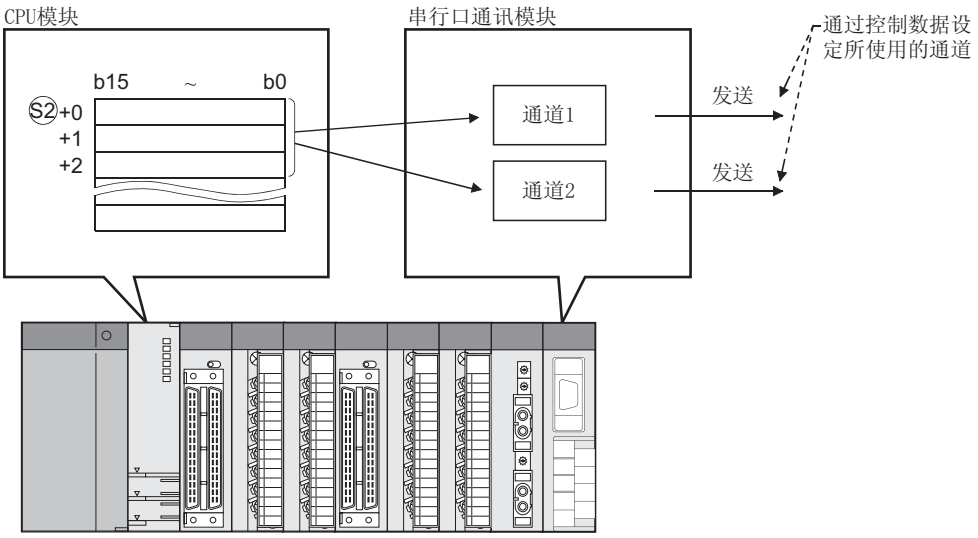


图 7.8 与使用了 OUTPUT 指令的对象设备进行通讯

(2) 智能功能模块专用指令的处理

依据智能功能模块专用指令，可指定结束软元件。

该结束软元件在指令执行结束时一个扫描周期 ON。

在同一智能功能模块中使用多个智能功能模块专用指令时，请在结束软元件 ON 之后再执行下一个智能功能模块专用指令。

(3) 注意事项

(a) 在结束软元件 ON 之前，由 RUN 变为 STOP 时
执行智能功能模块专用指令，在结束软元件 ON 之前将 CPU 模块由 RUN 变为 STOP，结束软元件将在下一个运行的 1 个扫描周期后 ON。

(b) 关于可使用的范围
对于 MELSECNET/H 的远程 I/O 站点上安装的智能功能模块，不能执行智能功能模块专用指令。
仅限于主基板、紧凑型主基板、扩展基板的智能功能模块可以执行。(AnS/A 系列对应的特殊功能模块除外。)

备 注

关于智能功能模块专用指令和结束软元件，请参照所使用的智能功能模块的手册。

基本 注 7.1

7.2 对 AnS/A 系列对应的特殊功能模块进行存取时

过程 注 7.1

冗余 注 7.1

(1) 对特殊功能模块的存取高速化的影响

由于处理的高速化，Q 系列 CPU 模块的扫描时间被缩短了。
对于特殊功能模块在较短的扫描周期内频繁执行 FROM/TO 指令时，作为对象的特殊功能模块的处理，可能不能正常完成。
此时，请利用特殊功能模块的计时器、恒定扫描等，使 FROM/TO 指令的执行间隔与处理时间、转换时间一致。（☞ 如下 (2)）

通用 注 7.1

(2) 特殊功能模块的存取高速化的对策

使用 AnS/A 系列对应的特殊功能模块时，请使用 SM415(2n ms 时钟)、SD415(2n ms 时钟设定）调节执行间隔。
将 SD415 的初始值设定为 “30”，将 SM415 用于 FROM/TO 指令的连锁，便会每隔 120ms 执行 FROM/TO 指令。

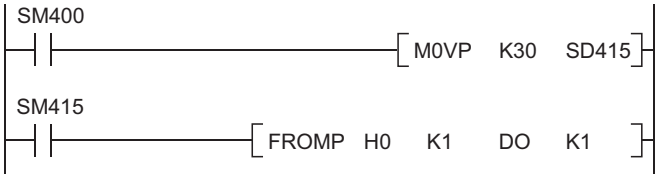


图 7.9 对策程序示例

要 点

要变更 SM415 的时钟时，请将变更值存储在 SD415 中。
SM415 的详细内容请参照附录 1，SD415 的详细内容请参照附录 2。

基本 注 7.1　过程 注 7.1　冗余 注 7.1

通用 注 7.1

在基本模式 QCPU、过程 CPU、冗余 CPU 和通用型 QCPU 中，不能使用 AnS 系列对应的特殊功能模块、因此本节内容不适用。

备忘录

第 8 章　参数

在本章中就可编程控制器系统建立时设定的参数进行说明。

(1) 参数的种类

CPU 模块的参数有以下几种。

- PC 参数 (☞ 8.1 节)
  单独使用可编程控制器时进行设定。
- 冗余参数 (☞ 8.2 节)
  使用冗余 CPU 建立冗余系统时进行设定。
- 网络参数 (☞ 8.3 节)
  将 MELSECNET/G 模块 注 8.1、注 8.2、MELSECNET/H 模块、Ethernet 模块和 CC-Link 模块与可编程控制器组合使用时进行设定。

- 远程口令 (☞ 8.4 节)
  使用 Ethernet 模块、串行通讯模块和 MODEM 接口模块的远程口令功能时进行设定。

(2) 参数的设定方法

PC 参数、冗余参数、网络参数和远程口令都是通过 GX Developer 进行设定。
关于 GX Developer 的设定操作，请参照以下手册。
☞ GX Developer 操作手册

要 点

在 GX Developer 中显示为灰色（不能选择）的设置项目是在所使用的 CPU 模块中不存在的功能，因此不需要进行设置。

备 注

- 本章的表格中所示的参数 No. 在参数设定出错时被存储到特殊寄存器 (SD16~26) 中。
  关于参数 No. 的一览表，请参阅附录 3。
- 关于参数的保存步骤，请参阅第 11 章。

在高性能 QCPU 中使用 MELSECNET/G 时，请确认 CPU 模块以及 GX Developer 的版本。
(☞ 附录 4.2)

在基本模式 QCPU、过程 CPU、冗余 CPU 中，不能使用 MELSECNET/G。

8.1 可编程控制器参数

对可编程控制器参数一览与各参数的内容进行说明。

8.1.1 基本模式 QCPU

(1) 可编程控制器名设定

设定所使用的 CPU 模块的标签、注释。

通过可编程控制器名设定标签、注释后对实际的动作并无影响。

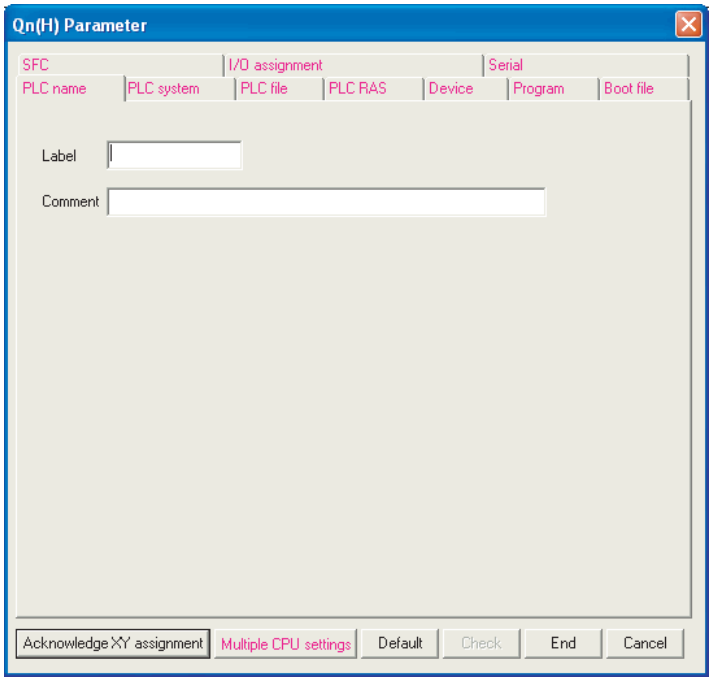


图 8.1 可编程控制器名设定

表 8.1 可编程控制器名设定一览

项目	参数 No.	内容	设定范围	缺省值	参照
标签	0000H	设定 CPU 模块的标签（名称、用途）。	半角文字 10 字符以内	无设定	—
注释	0001H	设定 CPU 模块的标签的注释。	半角文字 64 字符以内	无设定	—

(2) 可编程控制器系统设定

使用 CPU 模块时必须的设定。

直接使用缺省值也可进行控制。

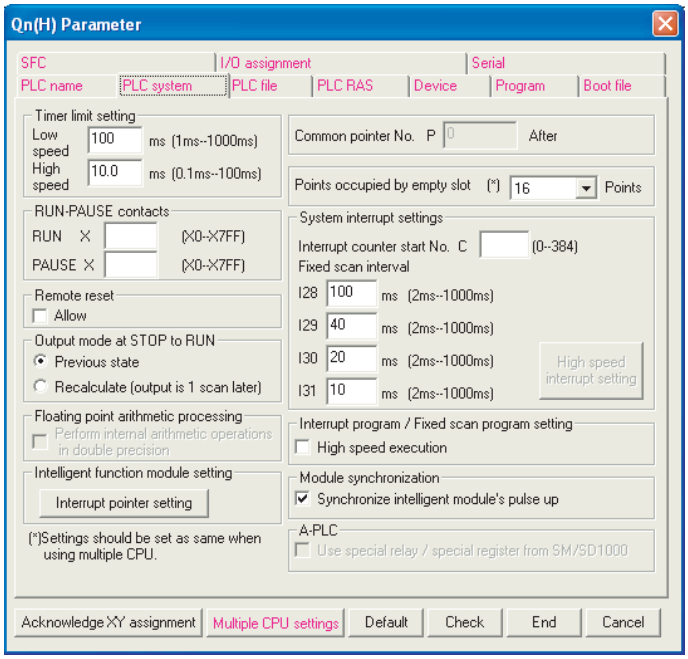


图 8.2 可编程控制器系统设定

表 8.2 可编程控制器系统设定一览

项目		参数 No.	内容	设定范围	缺省值	参照
计时器时限设定	低速	1000H	设定低速计时器 / 高速计时器的时限。	1ms~1000ms(1ms 单位)	100ms	9.2.10 项
	高速			0.1ms~100.0ms(0.1ms 单位)	10.0ms	9.2.10 项
RUN-PAUSE 触点	RUN	1001H	设定 CPU 模块的 RUN/PAUSE 的控制触点。 不能只进行 PAUSE 触点的设定。 (可设定 RUN 触点、RUN 触点 +PAUSE 触点。)	X0~7FF	无设定	6.6.1 项
	PAUSE					6.6.2 项
远程复位		1002H	设定 GX Developer 的远程复位操作的许可 / 禁止。	许可 / 不许可	不许可	6.6.3 项
STOP → RUN 时的输出模式		1003H	设定从 STOP 状态切换至 RUN 状态后的输出 (Y) 状态。	输出 STOP 前的输出 (Y) 状态 / 清除输出 (Y)(输出为 1 个扫描周期之后)	输出 STOP 前的输出 (Y) 状态	6.4 节
智能功能模块设定(中断指针设定)		100AH	设定中断指针 (150~127) 的分配、智能功能模块的起始 I/O No. 和起始 SI No.	起始 I/O No. 起始 SI No. I50~127	无设定	9.10 节
空插槽点数		1007H	设定主基板 / 扩展基板的空插槽点数。	[Q00JCPU] 0 点 /16 点 /32 点 /64 点 /128 点 /256 点 [Q00CPU, Q01CPU] 0 点 /16 点 /32 点 /64 点 /128 点 /256 点 /512 点 /1024 点	16 点	4.6.1 项 (5)

(接下一页)

表 8.2 可编程控制器系统设定一览（续）

项目		参数 No.	内容	设定范围	缺省值	参照
系统中断设定	中断计数器的起始 No.	1008H	设定中断计数器的起始 No.	C0~13184（计数器设定点数 -128 为止）	无设置	9.2.11 项 (4)
	In 固定周期间隔（n：28~31）		设定中断指针（128~31）的执行间隔。	2ms~1000ms（1ms 单位）	I28：100ms I29：40ms I30：20ms I31：10ms	9.10 节
中断程序 / 恒定周期程序设定		1008H	设定是否进行中断程序的高速执行。	不高速执行 / 高速执行	不高速执行	3.1.3 项
模块同步设定		100CH	设定是否同步启动 CPU 模块和智能功能模块。	同步 / 不同步启动智能功能模块	使智能功能模块的启动同步	—

(3) 可编程控制器文件设定

设定 CPU 模块中使用的文件。

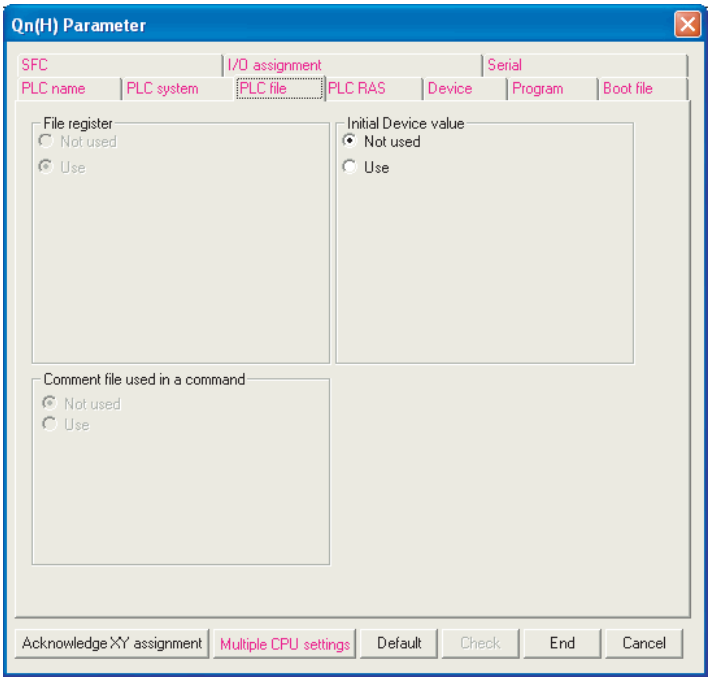


图 8.3 可编程控制器文件设定

表 8.3 可编程控制器文件设定一览

项目	参数 No.	内容	设定范围	缺省值	参照
软元件初始值	1102H	设定 CPU 模块使用的软元件初始值的文件。	• 不使用 • 使用	不使用	6.26 节

(4) 可编程控制器 RAS 设定

进行 RAS 功能所需的各种设定。

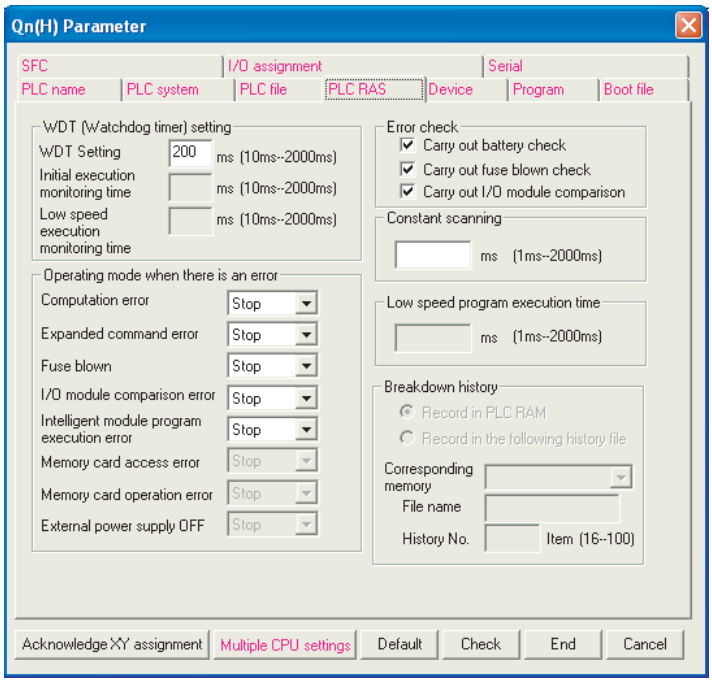


图 8.4 可编程控制器 RAS 设定

表 8.4 可编程控制器 RAS 设定一览

项目		参数 No.	内容	设定范围	缺省值	参照
WDT(看门狗计时器)设定	WDT 设定	3000H	设定 CPU 模块的看门狗计时器的时间。	10ms~2000ms(10ms 单位)	200ms	3.2 节
出错时的运行模式	运算错误	3002H	设定检测出错时 CPU 模块的动作模式。 *2	停止 / 继续运行	停止	6.17 节
	扩展指令错误 *1					
	保险丝断裂					
	模块核对错误					
	智能功能模块程序执行错误					
错误检查	进行电池检查	3001H	设定是否检测指定错误。	检查 / 不检查	检查	6.17 节
	进行保险丝断裂检查					
	进行 I/O 模块核对					
恒定扫描		3003H	设定恒定扫描时间。	1ms~2000ms(1ms 单位)	无设定	6.2 节

*1: 扩展指令出错是将来扩展用的设定。

*2: 关于出错的详细内容，请参照自诊断一览表。(6.17 节 (7))

(5) 软元件设定

设定每个软元件的使用点数和锁存范围。

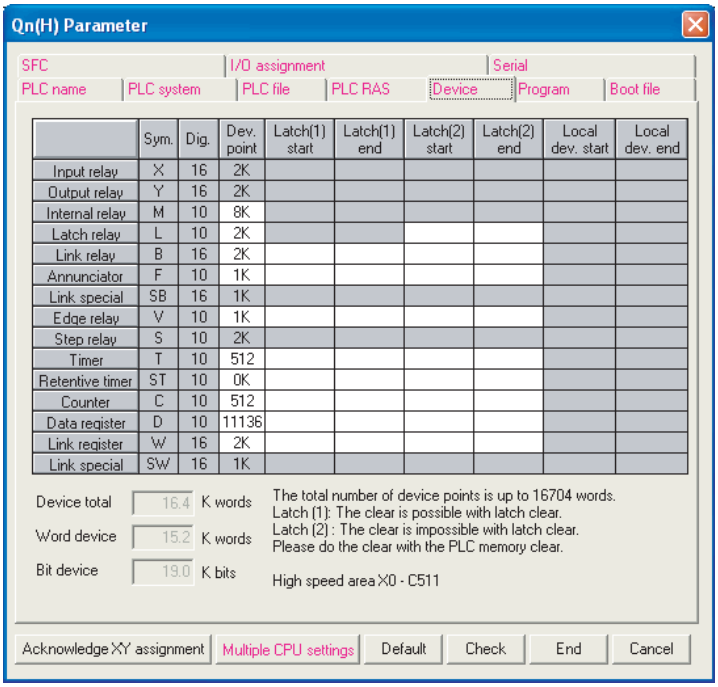


图 8.5 软元件设定

表 8.5 软元件设定一览

项目	参数 No.	内容	设定范围	缺省值	参照
软元件点数 *1	2000H	根据系统设定软元件的使用点数。	X(2k 点)、Y(2k 点)、S(2k 点)、SB(1k 点)、SW(1k 点)固定。 包括以上点数 (1.5k 字)，设定范围合计 16.4k 字。 • 1 软元件: 最大 32k 点 (位软元件的合计点数没有限制。)	X : 2k 点 Y : 2k 点 M : 8k 点 L : 2k 点 B : 2k 点 F : 1k 点 SB : 1k 点 V : 1k 点 S : 2k 点 T : 512 点 ST : 0k 点 C : 512 点 D : 11136 点 W : 2k 点 SW : 1k 点	9.1 节 9.2 节
锁存 (1) 起始 / 最终 (锁存清除有效)	2001H	设定远程锁存清除操作可清除的锁存范围 (起始软元件编号 / 最终软元件编号)。	B, F, V, T, ST, C, D, W 各软元件只能设定一个范围。	无设定	3.7 节 6.3 节
锁存 (2) 起始 / 最终 (锁存清除无效)	2002H	设定远程锁存清除操作不可清除的锁存范围 (起始软元件编号 / 最终软元件编号)。	L, B, F, V, T, ST, C, D, W 各软元件只能设定一个范围。	无设定	3.7 节 6.3 节

*1: 更改软元件点数时，必须注意不要让网络模块的刷新范围及智能模块的自动刷新范围超出软元件点数的范围。
超出了相应软元件范围时，数据有可能被写入到其它的软元件中。

(6) 引导文件设定

设定是否进行标准 ROM 的引导。

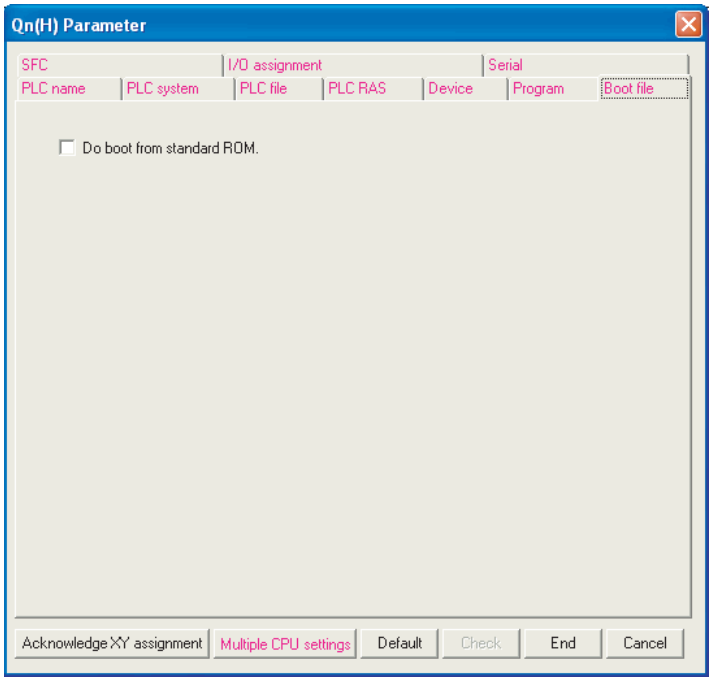


图 8.6 引导文件设定

表 8.6 引导文件设定

项目	参数 No.	内容	设定范围	缺省值	参照
引导文件设定	7000H	是否进行标准 ROM 的引导。	引导 / 不引导	不引导	5.1.5 项

(7) SFC 设定

设定使用 SFC 程序时的 SFC 程序启动模式、启动条件、块停止时的输出模式。

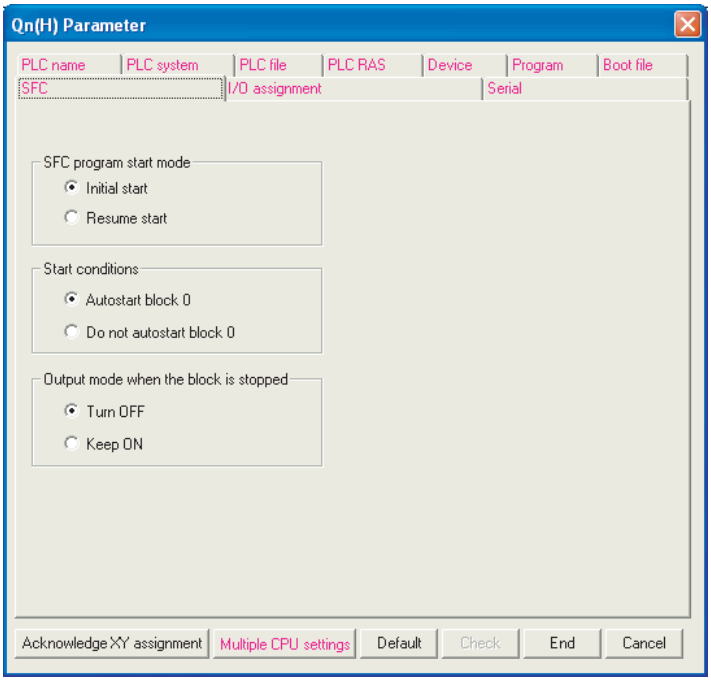


图 8.7 SFC 设定

表 8.7 SFC 设定一览

项目	参数 No.	内容	设定范围	缺省值	参照
SFC 程序启动模式	8002H	设定使用 SFC 程序时的 SFC 程序启动模式、启动条件、块停止时的输出模式。	QCPU(Q 模式)/QnACPU 编程手册 (SFC 篇)	初始启动	—
启动条件	8003H			自动启动块 0	
块停止时的输出模式	8006H			OFF	

(8) I/O 分配设定

设定系统各模块的安装状态。

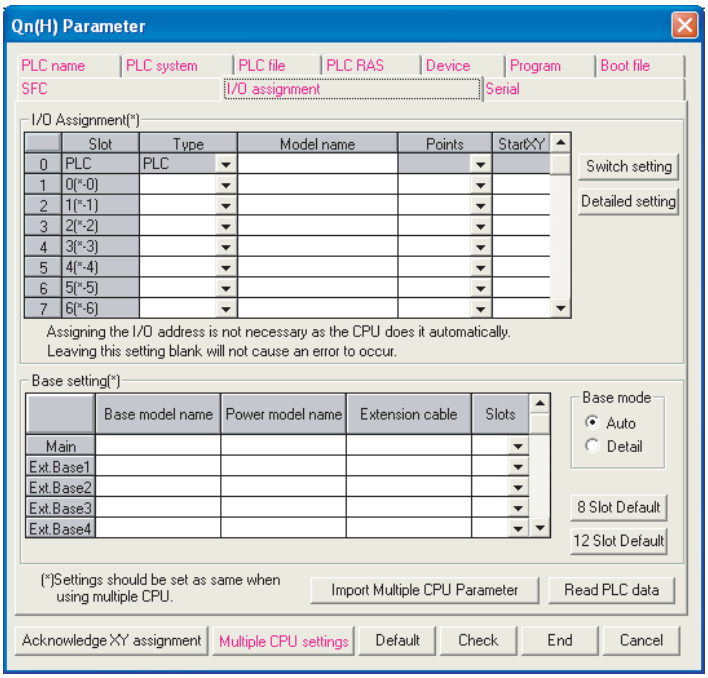


图 8.8 I/O 分配设定

表 8.8 I/O 分配设定一览

项目		参数 No.	内容	设定范围	缺省值	参照
I/O 分配	类别	0400H	设定所安装的模块的类别。	• CPU2 号机 ~3 号机：n 号机 / 空（不安装 CPU 模块的插槽设定为 “CPU(空)”。） • 空、输入、高速输入、输出、智能、I/O 混合、中断。	无设定	4.7 节
	型号		设定所安装的模块的型号。（用户备注。CPU 模块不使用。	半角 16 字	无设定	
	点数		设定各插槽的点数	[Q00JCPU] 0 点、16 点、32 点、48 点、64 点、128 点、256 点 [Q00CPU, Q01CPU] 0 点、16 点、32 点、48 点、64 点、128 点、256 点、512 点、1024 点	无设定	
	起始 XY（起始 I/O 编号）		设定各插槽的起始 I/O 编号	[Q00JCPU] 0H~F0H [Q00CPU, Q01CPU] 0H~3F0H	无设定	

（接下一页）

表 8.8 I/O 分配设定一览（续）

项目		参数 No.	内容	设定范围	缺省值	参照
基板设定	基板型号	0401H	设定所使用的主基板、扩展基板的型号。（用户备注。CPU 模块不使用。）	半角 16 字符	无设定	4.4 节
	电源型号		设定主基板、扩展基板上安装的电源模块的型号。（用户备注。CPU 模块不使用。）	半角 16 字符	无设定	
	扩展电缆型号		设定扩展电缆型号。（用户备注。CPU 模块不使用。）	半角 16 字符	无设定	
	插槽数		设定主基板、扩展基板的插槽数。 全部基板都进行插槽数的设置。	2，3，5，8，10，12	无设定	
开关设定		0407H	设定智能模块的各种开关。	参照所使用的智能模块的手册	无设定	6.10 节
详细设定	错误时输出模式	0403H	设定在管理 CPU 发生停止出错时是否清除输出。	清除 / 保持	清除	6.8 节
	H/W 错误时 CPU 动作模式	4004H	智能模块硬件异常时，停止 / 继续运行管理 CPU。	停止 / 继续运行	停止	6.9 节
	I/O 响应时间	0405H	设定输入模块、高速输入模块、I/O 混合模块和中断模块的响应时间。	• 输入、I/O 混合：1ms，5ms，10ms，20ms，70ms • 高速输入、中断：0.1ms，0.2ms，0.4ms，0.6ms，1ms	输入、I/O 混合：10ms 高速输入、中断：0.2ms	6.7 节
	管理 CPU	0406H	设定 I/O 模块和智能模块的管理 CPU。	• 1 号机、2 号机、3 号机	1 号机	QCPU 用户手册（多 CPU 系统篇）

(9) 串行口通讯设定

设定使用 Q00CPU、Q01CPU 的串行口通讯功能时的发送速度、总数确认、信息等待时间以及可否 RUN 中写入。

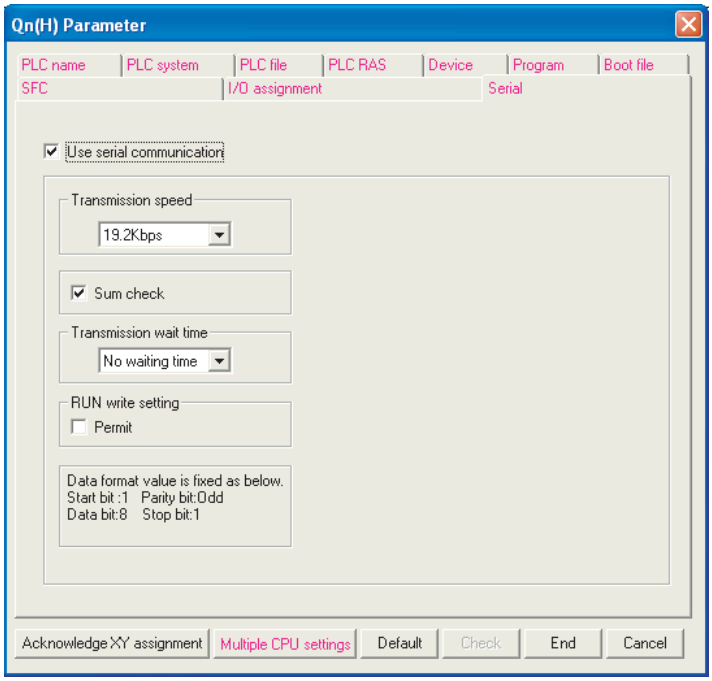


图 8.9 串行口通讯设定

表 8.9 串行口通讯设定一览

项目	参数 No.	内容	设定范围	缺省值	参照
使用串行口通讯功能	100E$_H$	使用串行口通讯功能时，带有确认标记。	使用 / 不使用串行口通讯功能	不使用串行口通讯功能	6.24 节
发送速度		设定与对象设备进行数据通讯所需的发送速度。	9.6kbps/19.2kbps/38.4kbps/57.6kbps/115.2kbps	19.2kbps	
总数确认		利用串行口通讯功能进行数据通讯时，根据对象设备的规格，设定是否在发送信息、接收信息中附加确认编码。	无 / 有	有	
信息等待时间		对象设备接收数据后，不能立即接收数据时的基本模式 QCPU 侧的发送等待时间。	无等待 /10ms~150ms (10ms 单位)	无等待	
RUN 中写入设定		从对象设备向 CPU 模块写入数据时，CPU 模块是否在 RUN 中写入。	不许可 / 许可	不许可	

(10)X/Y 分配确认

对 I/O 分配、MELSECNET/Ethernet 设定、CC-Link 设定中设定的内容进行确认。

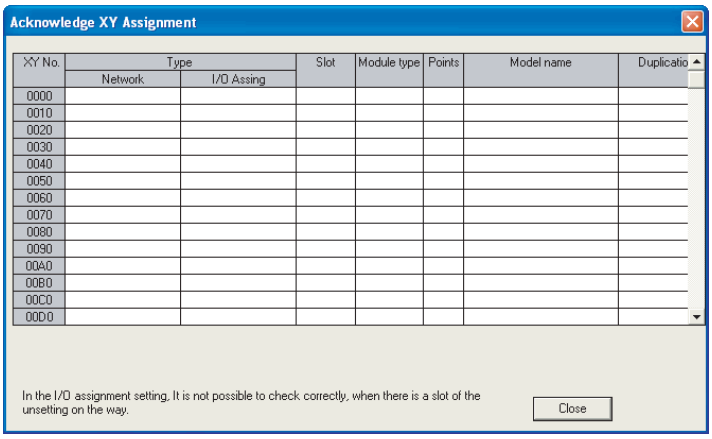


图 8.10 X/Y 分配确认

表 8.10 X/Y 分配确认一览

项目	参数 No.	内容	设定范围	缺省值	参照
X/Y 分配确认	—	对 I/O 分配、MELSECNET/Ethernet 设定、CC-Link 设定中设定的内容进行确认。	—	—	—

(11)多 CPU 设定

建立多 CPU 系统所需的设定。

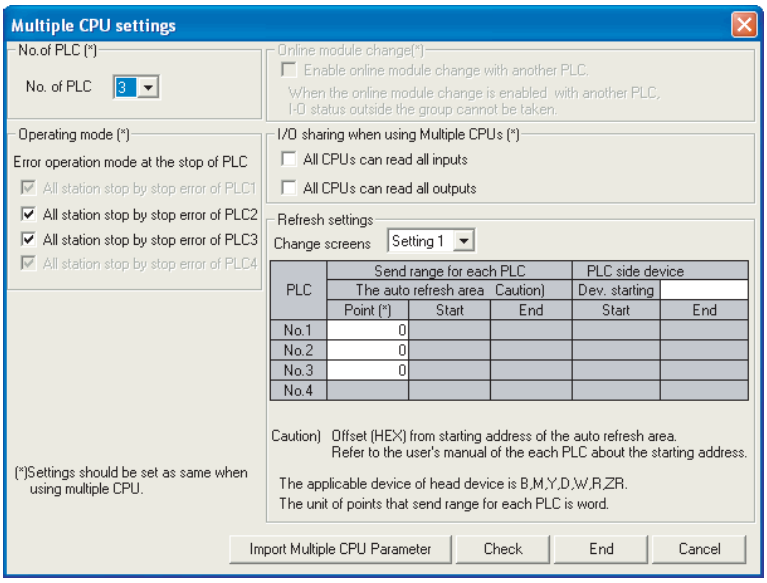


图 8.11 多 CPU 设定

表 8.11 多 CPU 设定一览

项目		参数 No.	内容	设定范围	缺省值	参照
CPU 数量		0E00H	设定多 CPU 系统中使用的 CPU 的数量。	1 台 ~3 台	1 台	QCPU 用户手册（多 CPU 系统篇）的“8.2.2 使用多 CPU 系统时需要的参数”。
动作模式		0E01H	2 号机 ~3 号机的 CPU 模块停止出错时多 CPU 系统的动作设定。 1 号机发生停止错误时多 CPU 模块会停止。（固定）	因为 n 号机的错误全部机器停止 / 不停止	因为 n 号机的错误全部机器停止	
组外的 I/O 设置	接收组外的输入状态	0E04H	设定是否接收他号机所管理的输入模块、智能模块的输入状态。	接收 / 不接收组外部的输入状态	不接收组外部的输入状态	
	接收组外的输出状态		设定是否接收他号机所管理的输出模块、智能模块的输出状态。	接收 / 不接收组外部的输出状态	不接收组外部的输出状态	
刷新设置		E002H E003H	设定多 CPU 系统的各 CPU 模块之间，通过自动刷新写入 / 读出数据的软元件与点数。	[各 CPU 模块的发送范围] 0~320 点 (2 点单位)/1 台，最大 4416/1 系统 [CPU 模块侧软元件] B、M、Y、D、R、ZR 占有从指定软元件编号开始至发送范围中设定的点数的软元件。 • 发送范围为 1 点时，B、M、Y 占有 16 点。 • 发送范围为 1 点时，D、W、R、ZR 占有 1 点。	无设置	

8.1.2 高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU

(1) 可编程控制器名设定

设定所使用的 CPU 模块的标签和注释。

通过 PC 名设定标签和注释后，对实际的动作并无影响。

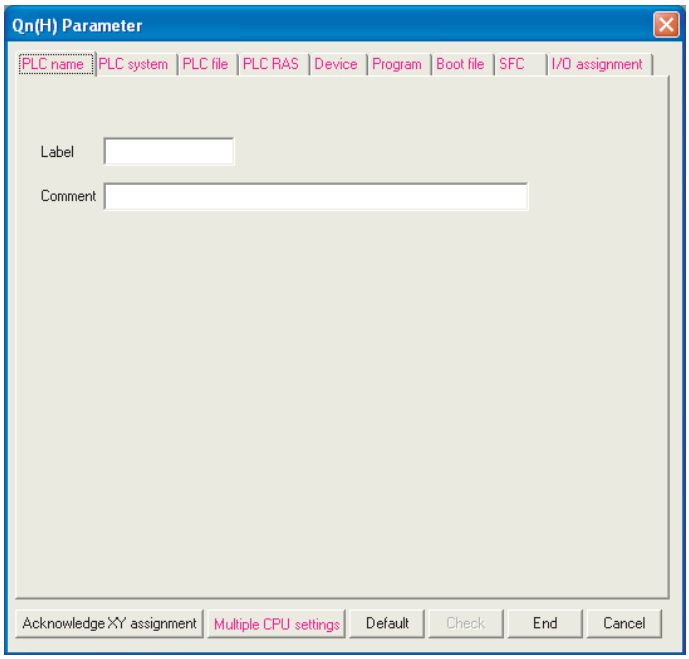


图 8.12 可编程控制器名设定

表 8.12 可编程控制器名设定一览

项目	参数 No.	内容	设定范围	缺省值	参照
标签	0000H	设定 CPU 模块的标签（名称、用途）	半角 10 字以内	无设定	—
注释	0001H	设定 CPU 模块的标签的注释	半角 64 字以内	无设定	—

(2) 可编程控制器系统设定

使用 CPU 模块时所需的设定。

使用缺省值也可进行控制。

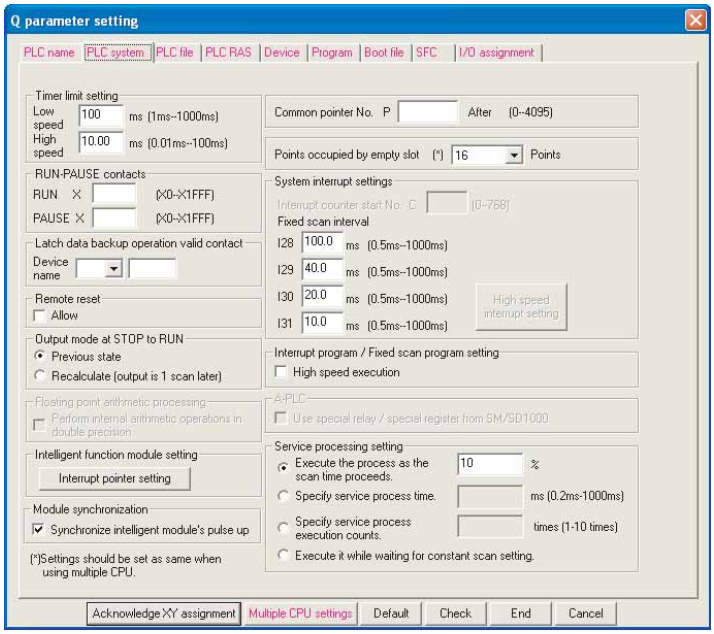


图 8.13 可编程控制器系统设定

表 8.13 可编程控制器系统设定一览

项目		参数 No.	内容	设定范围	缺省值	参照
计时器时限设定	低速	1000H	设定低速计时器 / 高速计时器的时限。	1ms~1000ms(1ms 单位)	100ms	9.2.10 项
	高速			0.1ms~100.0ms(0.1ms 单位) *2	10.0ms	9.2.10 项
RUN-PAUSE 触点	RUN	1001H	设定 CPU 模块的 RUN/PAUSE 的控制触点。 不能只进行 PAUSE 触点的设定。 (可设定 RUN 触点、RUN 触点＋PAUSE 触点。)	X0~1FFF	无设定	6.6.1 项
	PAUSE					6.6.2 项
远程复位		1002H	设定 GX Developer 的远程复位操作的许可 / 禁止。	许可 / 不许可	不许可	6.6.3 项
STOP → RUN 时的输出模式		1003H	设定从 STOP 状态切换至 RUN 状态后的输出 (Y) 状态。	输出 STOP 前的输出 (Y) 状态 / 清除输出 (Y)(输出为 1 个扫描周期之后)	输出 STOP 前的输出 (Y) 状态	6.4 节
浮动小数点运算处理 *1		1004H	是否采样倍数制进行浮动小数点的运算。	是否用双精度进行内部运算处理	用倍精度进行内部运算处理	3.9.4 项
智能模块设定(中断指针设定)		100AH	设定中断点 (150~255) 的分配、智能模块的起始 I/O NO. 和起始 SI NO.	起始 I/O No. 起始 SI No. I50~255	无设定	9.10 节
通用点 No.		1005H	设定作为通用点使用的点的起始 No.	P0~4095	无设定	9.9.2 项
空插槽点数		1007H	设定主基板 / 扩展基板的空插槽点数。	0 点 /16 点 /32 点 /64 点 /128 点 /256 点 /512 点 /1024 点	16 点	4.6.1 项 (5)

*1：使用过程 CPU、冗余 CPU、通用型 QCPU 时不能设定。

*2：在通用型 QCPU 中，设定范围变为 0.01ms ～ 100.0ms (单位：0.01ms)。

(接下一页)

表 8.13 可编程控制器系统设定一览（续）

项目			参数 No.	内容	设定范围	缺省值	参照
系统中断设定	中断计数器起始 No.		1008H	设定中断计数器的起始 No. *4	C0~22272（计数器设定点数 -256 为止）	无设定	9.2.11 项 (4)
	In 恒定周期间隔（n：28~31）			设定中断指针（I28~31）的执行间隔	0.5ms~1000ms（0.5ms 单位）	I28：100.0ms I29：40.0ms I30：20.0ms I31：10.0ms	9.10 节
	高速中断设定 *2	X 输入	100FH	高速中断指针 149 的固定周期间隔、高速 I/O 刷新设置、高速缓冲发送设定。	149：0.2ms~1.0ms（0.1ms 单位）	无设定	3.1.3 项 3.3.5 项
		Y 输出	1010H				
		缓冲读出	1011H				
		缓冲写入	1012H				
	服务处理设定 *3		1013H	从以下方法中选择服务处理的执行方法。 ・根据扫描时间的比例执行 ・服务处理的次数 ・服务处理的时间 ・在恒定扫描设定时的等待时间执行	・1 ～ 100%（单位：%） ・1 ～ 10 次（单位：次） ・0.2 ～ 1000ms（单位：0.1ms） ・无设定	・10% ・1 次 ・0.2ms ・无设定	6.25.2 项
锁存数据备份操作有效触点 *3			1014H	设定至标准 ROM 的锁存数据备份时有效触点软元件号。	—	不使用	6.29 节
中断程序 / 固定周期程序设定			1008H	是否高速执行中断程序	不高速执行 / 高速执行	不高速执行	3.1.3 项
模块同步设置			100CH	是否同步启动 CPU 模块和智能模块。	同步 / 不同步启动智能功能模块	同步启动智能功能模块	—
A 系列 CPU 兼容设置 *1			100DH	是否使用 MELSEC-A 系列用的特殊继电器 / 特殊寄存器（SM1000/SD1000~SM1299/SD1299）。	使用 / 不使用 SM1000、SD1000 以上的特殊继电器 / 特殊寄存器。	使用 SM1000、SD1000 以后的特殊继电器 / 特殊寄存器。	9.3.2 项 9.3.3 项

*1：使用冗余 CPU、通用型 QCPU 时不能设定。
*2：只有在使用高性能模式 QCPU 时才可以进行高速中断设定。
*3：只有在使用通用型 QCPU 时才可以设定。
*4：在通用型 QCPU 中，不能设定中断计数器的起始号。

(3) 可编程控制器文件设定

设定 CPU 模块中使用的各种文件。

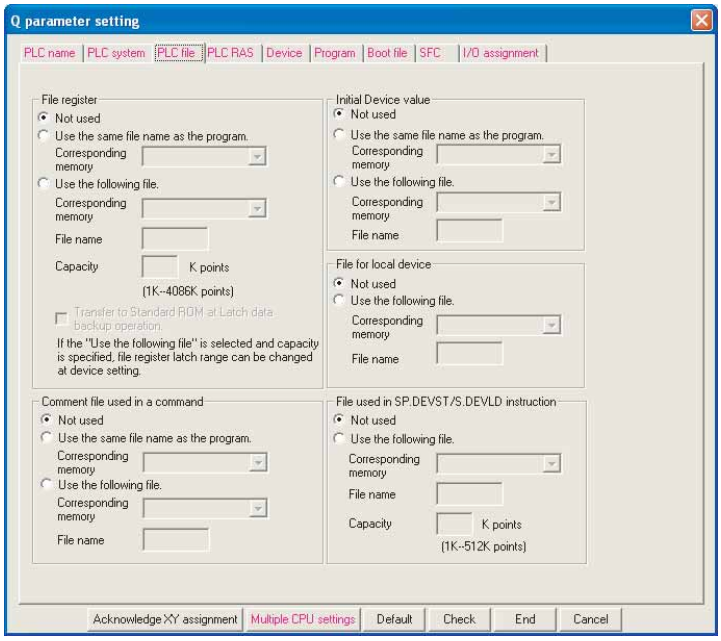


图 8.14 可编程控制器文件设定

表 8.14 可编程控制器文件设定一览

项目	参数 No.	内容	设定范围	缺省值	参照
文件寄存器	1100H	设定程序中使用的文件寄存器的文件。	• 不使用 • 使用与程序相同的文件名 • 使用以下文件名	不使用	9.7 节
指令使用的注释文件	1101H	设定程序中使用的软元件注释的文件。	• 不使用 • 使用与程序相同的文件名 • 使用以下文件名	不使用	—
软元件初始值	1102H	设定 CPU 模块中使用的软元件初始值的文件。	• 不使用 • 使用与程序相同的文件名 • 使用以下文件名	不使用	6.26 节
本地软元件用文件	1103H	设定程序中使用的本地软元件的文件。	• 不使用 • 使用以下文件名	不使用	9.13.1 项
锁存数据备份操作时传送至标准 ROM *1	1104H	设定至标准 ROM 的锁存数据备份时是否进行文件寄存器批量传送。	进行 / 不进行文件寄存器的批量传送	不使用	6.29 节
SP.DEVST/S.DEVLD 指令中使用的文件 *1	1105H	设定至软元件数据的标准 ROM 的写入 / 读出中使用的软元件数据存储用文件。	• 不使用 • 使用下述文件	无设定	6.30 节

*1：仅在使用通用型 QCPU 时才可以设定。

(4) 可编程控制器 RAS 设定 (可编程控制器 RAS 设定 (1)*1)

进行 RAS 功能所需的各种设定。

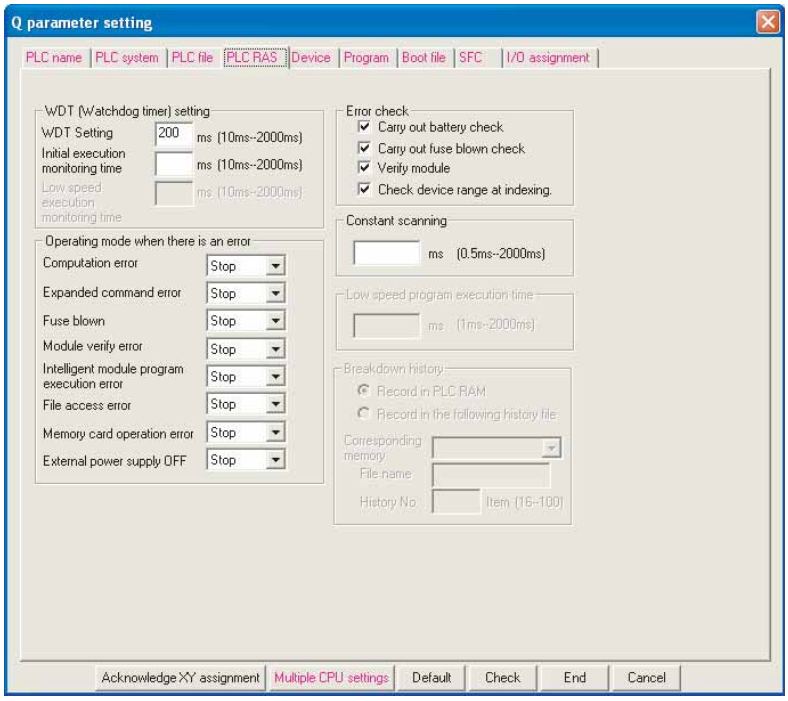


图 8.15 可编程控制器 RAS 的设定

表 8.15 可编程控制器 RAS 设定一览

项目		参数 No.	内容	设定范围	缺省值	参照
WDT (看门狗计时器) 设置	WDT 设定	3000H	设定 CPU 模块的看门狗计时器的时间。	10ms~2000ms (10ms 单位)	200ms	3.2 节
	初始执行监视时间		设定使用初始执行程序时看门狗计时器的时间。	10ms~2000ms (10ms 单位)	无设定	3.3.1 项
	低速执行监视时间 *2		设定使用低速执行程序时看门狗计时器的时间。	10ms~2000ms (10ms 单位)	无设定	3.3.3 项
出错时的运行模式	运算错误	3002H	设定检测出错时 CPU 模块的动作模式。 *4	停止 / 继续运行	停止	6.17 节
	扩展指令错误 *3					
	保险丝断裂					
	模块核对错误					
	智能程序执行错误					
	文件访问出错					
	存储卡操作错误					
	外部电源供给 OFF *3					

*1：冗余 CPU 的可编程控制器参数设定时，GX Developer 的可编程控制器参数画面中将显示 “可编程控制器 RAS 设定 (1)”。

*2：使用冗余 CPU、通用型 QCPU 时不能设定。

*3：扩展指令出错、外部电源供给 OFF 是将来扩展用的设定。

*4：关于出错的详细内容，请参照自诊断一览表。(6.17 节 (7))

(接下一页)

表 8.15 可编程控制器 RAS 设定一览（续）

项目		参数 No.	内容	设定范围	缺省值	参照
错误检查	进行电池检查 进行保险丝断裂检查 进行模块核对检查 对进行了变址修饰的软元件的范围进行检查	3001H	设定是否检测指定错误。	检查 / 不检查	检查	6.17 节
恒定扫描		3003H	设定恒定扫描时间。	0.5ms~2000ms(0.5ms 单位)	无设定	6.2 节
低速程序执行时间 *2		3006H	设定每次扫描的低速程序的执行时间。	1ms~2000ms	无设定	3.3.3 项
故障历史记录		3005H	设定 CPU 模块的故障历史记录的储存区域。*5	存储到程序内存 / 存储到以下记录文件	存储到程序内存	6.18 节

*2：使用冗余 CPU、通用型 QCPU 时不能设定。
*5：在通用型 QCPU 中，只能设定内存。

过程
注 8.3

(5) 可编程控制器 RAS 设定 (2) *注 8.3*

设定冗余 CPU 中 RAS 功能所需的各种设定。
本设定只能在使用冗余 CPU 时才能进行。

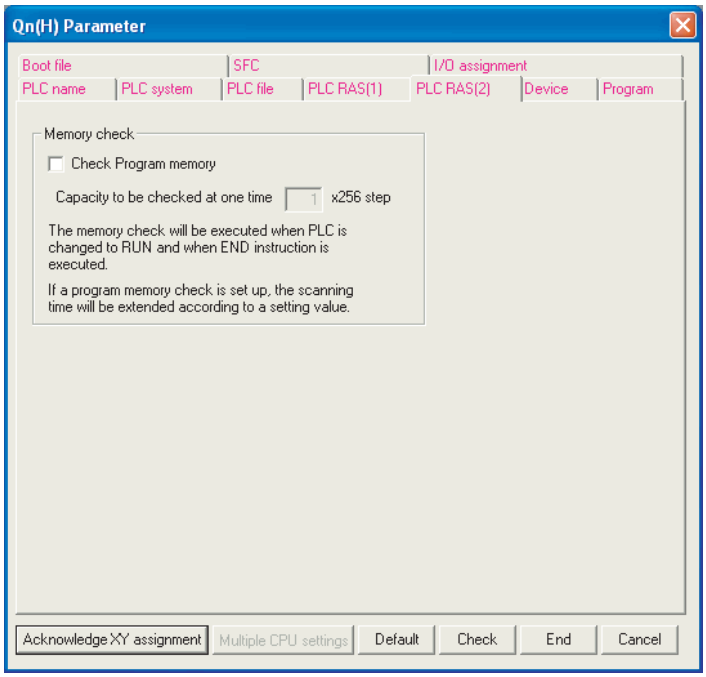


图 8.16 可编程控制器 RAS 设定 (2)

表 8.16 可编程控制器 RAS 设定 (2) 一览

项目		参数 No.	内容	设定范围	缺省值	参照
内存检查	进行程序内存检查	3008H	对于程序内存（系统区域除外）的用户区域，设定是否检查数据损坏。	不检查 / 检查程序内存	不检查程序内存	—
			已设定为“进行程序内存检查”时，设定检测步数。	Q12PRHCPU：1~496 Q25PRHCPU：1~1008	1×256 步	

要 点

检查全部的程序内存区域时，检查对象的步数可通过下述方法计算：

- 在 GX Developer 的可编程控制器读取画面中，确认程序内存的全部容量。确认全部容量时，应将程序内存格式化后，在未写入文件的状态下进行确认。(☞ 5.2.2 项 (3)(c))
- 通过下述计算公式算出检查对象的步数：

$$\text{检查对象步数} = \frac{\text{全部容量}}{4}$$

过程
注 8.3

在过程控制 CPU 中进行内存检查设置时，应确认 CPU 模块以及 GX Developer 的版本。(☞ 附录 4.3)

(6) 软元件设定

设定各软元件的使用点数、锁存范围和本地软元件范围。

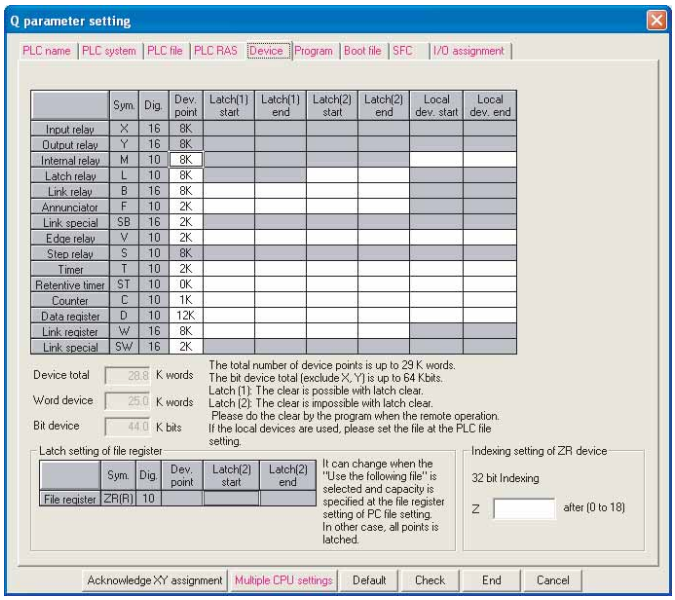


图 8.17 软元件设定

表 8.17 软元件设定一览

项目	参数 No.	内容	设定范围	缺省值	参照
软元件点数 *1	2000H	根据系统设置软元件的使用点数。	X(8k 点)、Y(8k 点)、S(8k 点)、SB(2k 点) *3 、SW(2k 点) *3 固定。 包括以上点数 (3.7k 字)，设定范围合计 29k 字。 • 1 软元件：最大 32k 点 • 位软元件的合计点数：最大 64k 点	X ：8k 点 Y ：8k 点 M ：8k 点 L ：8k 点 *4 B ：8k 点 F ：2k 点 SB ：2k 点 V ：2k 点 S ：8k 点 T ：2k 点 ST ：0k 点 C ：1k 点 D ：12k 点 W ：8k 点 SW ：2k 点	9.1 节 9.2 节
锁存 (1) 起始 / 最终 *4	2001H	设定通过 RESET/L.CLR 开关 *2、远程锁存清除操作可清除的锁存范围 (起始软元件编号 / 最终软元件编号)。	B，F，V，T，ST，C，D，W 各软元件只能设定一个范围。	无设置	3.7 节 6.3 节
锁存 (2) 起始 / 最终 *4	2002H	设定通过 RESET/L.CLR 开关 *2、远程锁存清除操作不可清除的锁存范围 (起始软元件编号 / 最终软元件编号)。	L，B，F，V，T，ST，C，D，W 各软元件只能设定一个范围。	无设置	3.7 节 6.3 节
本地软元件起始 / 最终	2003H	设定本地软元件使用的锁存范围 (起始软元件编号 / 最终软元件编号)。	M，V，T，ST，C，D，W 各软元件只能设定一个范围。	无设置	9.13.1 项

*1： 更改软元件点数时，必须使网络模块的刷新范围及智能模块的自动刷新范围不要超出软元件点数的范围。
超出了相应软元件范围时，数据可能会被写入到其它的软元件中。
*2： 在通用型 QCPU 中，不能通过开关操作进行锁存清除。
*3： 在通用型 QCPU 中，可以在范围内设定。
*4： 在通用型 QCPU 中，如果对软元件进行锁存将使扫描时间延迟。
对软元件进行锁存时，应考虑扫描时间的延迟时间。(10.1.2 项 (11))

(7) 程序设定

设定将多个程序写入 CPU 模块时，程序的文件名与执行型号（执行条件）。

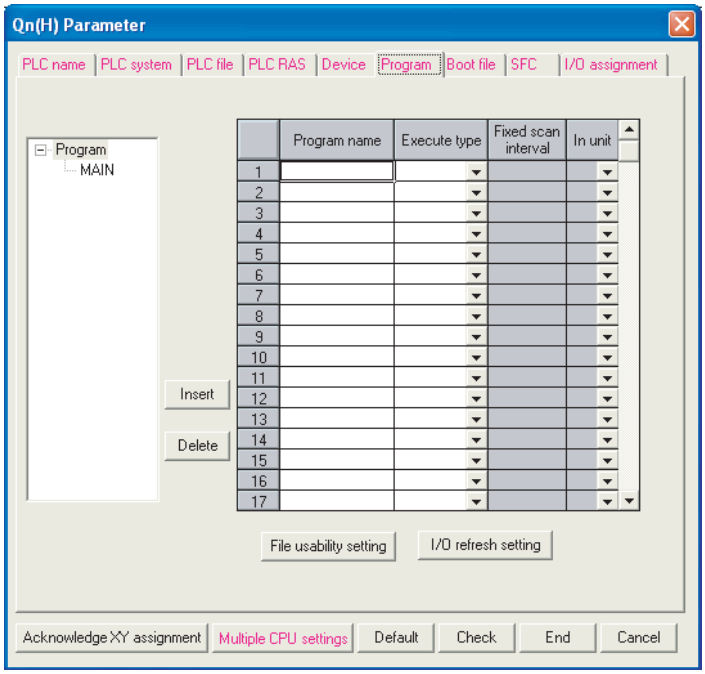


图 8.18 程序设定

表 8.18 程序设定一览

项目	参数 No.	内容	设定范围	缺省值	参照
程序设定	7000H	设定将多个程序写入 CPU 模块时，程序的文件名与执行型号（执行条件）。 此外，恒定周期间隔（恒定周期执行型程序）也将设定。	• 程序名 • 执行型（恒定周期选择时的恒定周期间隔） • 文件使用方法设置 • I/O 刷新设定	无设定	3.3.6 项

(8) 引导文件设置

通用 UD 注 8.4

进行引导运行、向标准 ROM 注 8.4 的自动写入所需的设置。

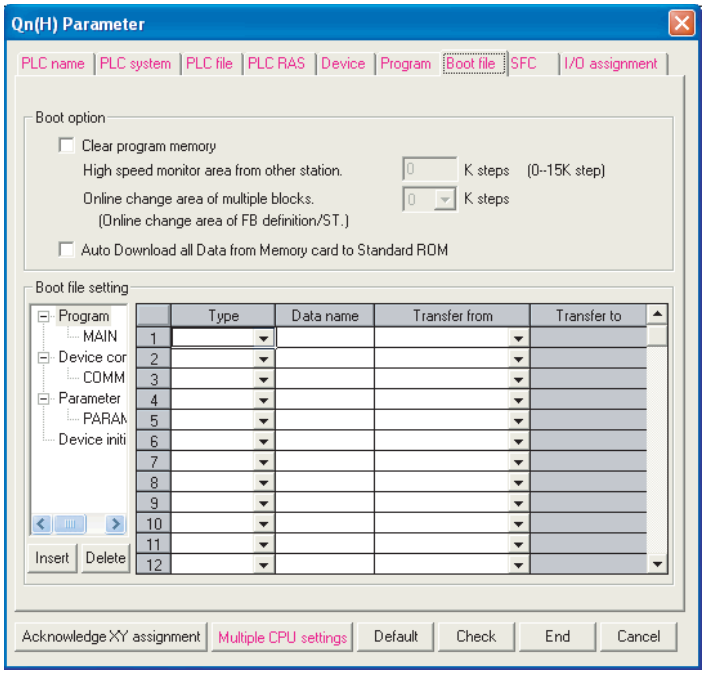


图 8.19 引导文件设定

表 8.19 引导文件设定

项目		参数 No.	内容	设定范围	缺省值	参照
引导选项	清除程序内存	7000H	设定引导运行时是否清除程序内存。	引导运行时不清除程序内存 / 引导运行时清除程序内存	引导运行时不清除程序内存	5.2.8 项 5.2.9 项
	存储卡 → 标准 ROM 全部数据自动写入		设定引导运行时是否将存储卡的数据自动写入标准 ROM。	存储卡 → 标准 ROM 全部数据不自动写入 / 自动写入	存储卡 → 标准 ROM 全部数据不自动写入	
引导文件设定			设定引导运行的文件的类别、数据名称和发送源驱动器。	类别、数据名称和发送源驱动器（发送目标驱动器在程序内存中自动设定。）	无设置	

通用 UD 注 8.4

在通用型 QCPU 中，不能进行至标准 ROM 的自动写入。

(9) SFC 设定

设定使用 SFC 程序时的 SFC 程序启动模式、启动条件、块停止时的输出模式。

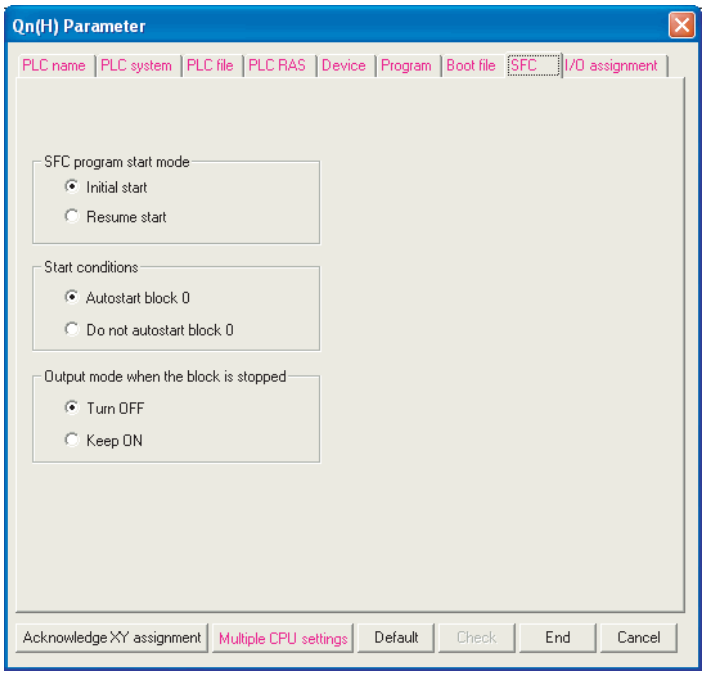


图 8.20 SFC 设定

表 8.20 SFC 设定一览

项目	参数 No.	内容	设定范围	缺省值	参照
SFC 程序启动模式	8002H	设定使用 SFC 程序时的 SFC 程序启动模式、启动条件、块停止时的输出模式。	参照 QCPU(Q 模式)/QnACPU 编程手册 (SFC 篇)	初始启动	—
启动条件	8003H			自动启动块 0	
块停止时的输出模式	8005H			OFF	

(10) I/O 分配设定

设定系统的各模块的安装状态。

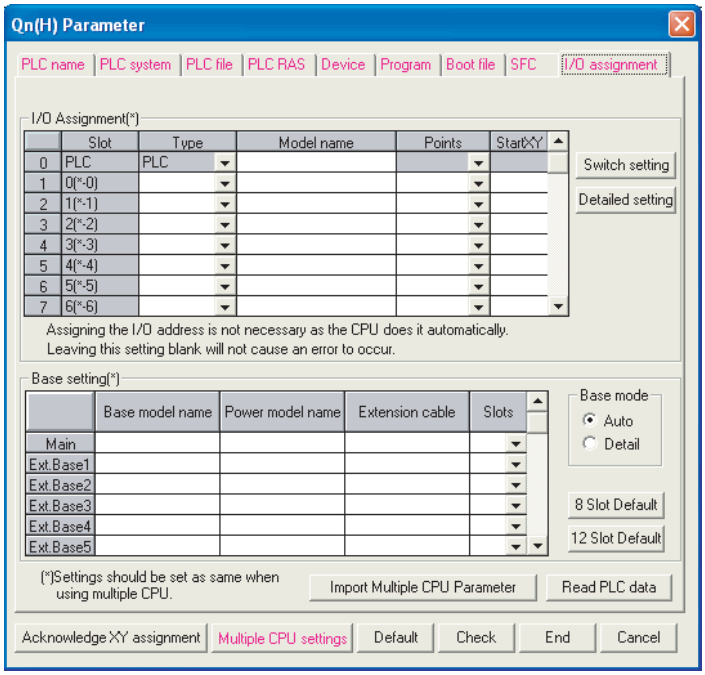


图 8.21 I/O 分配设定

表 8.21 I/O 分配设定一览

项目		参数 No.	内容	设定范围	缺省值	参照
I/O 分配	类别	0400H	设定所安装的模块的类别。	• CPU2 号机～4 号机 / 空（不安装 CPU 模块的插槽设为 “CPU（空）”。） • 空、输入、高速输入、输出、智能、I/O 混合、中断	无设定	4.7 节
	型号		设定所安装的模块的型号。（用户的备注。CPU 模块不使用。）	半角 16 字符	无设定	
	点数		设定各插槽的点数。	0 点、16 点、32 点、48 点、64 点、128 点、256 点、512 点、1024 点	无设定	
	起始 XY（起始 I/O 地址号）		设定各插槽的起始 I/O 地址号。	0H～FF0H	无设定	
基本设置	基板型号	0401H	设定所使用的主基板、扩展基板的型号。（用户的备注。CPU 模块不使用。）	半角 16 字符	无设定	4.4 节
	电源模块型号		设定主基板、扩展基板上安装的电源模块的型号。（用户备注。CPU 模块不使用。）	半角 16 字符	无设定	
	扩展电缆型号		设定扩展电缆的型号。（用户备注。CPU 模块不使用。）	半角 16 字符	无设定	
	插槽数		设定主基板、扩展基板的插槽数。 所有基板都进行插槽数的设定。	2，3，5，8，10，12	无设定	

（接下一页）

表 8.21 I/O 分配设定一览（续）

项目		参数 No.	内容	设定范围	缺省值	参考
开关设置		0407H	设定智能模块的各种开关。	参照所使用的智能模块的手册	无设定	6.10 节
详细设定	出错时的输出模式	0403H	设定在管理 CPU 方式停止错误时清除 / 保留输出。	清除 / 保留	清除	6.8 节
	H/W 出错时的 CPU 动作模式	4004H	设定在智能模块硬件异常时，停止 / 继续运行管理 CPU。	停止 / 继续运行	停止	6.9 节
	I/O 响应时间	0405H	设定输入模块、高速输入模块、I/O 混合模块、中断模块的响应时间。	• 输入、I/O 混合：1ms，5ms，10ms，20ms，70ms • 高速输入、中断：0.1ms，0.2ms，0.4ms，0.6ms，1ms	输入、I/O 混合：10ms 高速输入、中断：0.2ms	6.7 节
	管理 CPU*1	0406H	设定 I/O 模块、智能模块的管理 CPU。	• 1 号机，2 号机，3 号机，4 号机	1 号机	QCPU 模块用户手册（多 CPU 系统篇）

*1：使用冗余 CPU 时不能设定。

(11)X/Y 分配确认

对 I/O 分配、MELSECNET/Ethernet 设定、CC-Link 设定中所设定的内容进行确认。

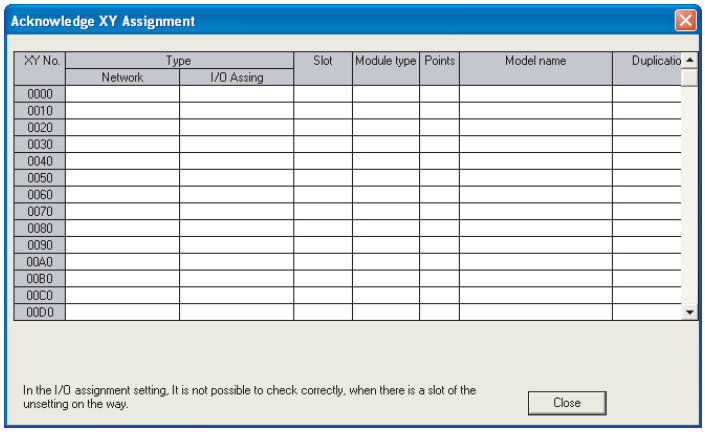


图 8.22 X/Y 分配确认

表 8.22 X/Y 分配确认一览

项目	参数 No.	内容	设定范围	缺省值	参考
X/Y 分配确认	—	可确认 I/O 分配、MELSECNET/Ethernet 设定、CC-Link 设定中设定的内容。	—	—	—

冗余
注 8.5

(12) 多 CPU 设定*注 8.5*

建立多 CPU 系统所需的设定。

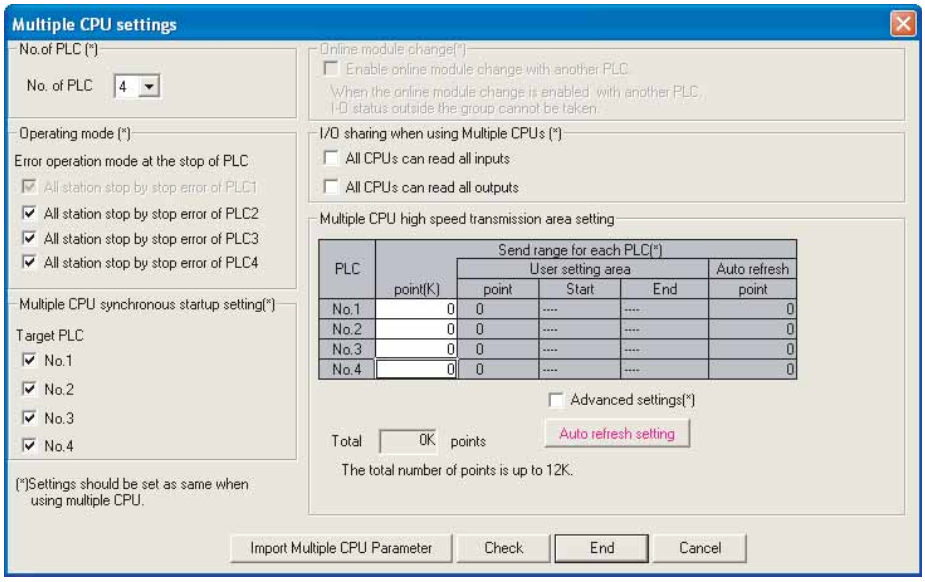


图 8.23 多 CPU 设定

表 8.23 多 CPU 设定一览

项目	参数 No.	内容	设定范围	缺省值	参照
CPU 个数	0E00H	设定多 CPU 系统中使用的 CPU 的个数。	1~4 个	1 个	QCPU 模块用户手册（多 CPU 系统篇）
动作模式	0E01H	设定 2 号机 ~4 号机的 CPU 模块方式停止错误时多 CPU 系统的动作。 1 号机方式停止错误时，多 CPU 系统停止。（固定）	由于 n 号机的错误全部机器停止 / 不停止	由于 n 号机的错误全部机器停止	
在线模块更换设定	E006H	设定是否允许在多 CPU 系统中进行在线模块更换。（设为允许时，不能接收组外的 I/O 状态。）	允许 / 不允许其它 CPU 模块的在线模块更换	• 高性能模式 QCPU：不允许 • 过程 CPU：允许 • 通用型 QCPU：不允许	QCPU 模块用户手册（多 CPU 系统篇）

（接下一页）

冗余
注 8.5

CPU 不能使用多 CPU 系统，因此不能进行多 CPU 设定。

表 8.23 多 CPU 设定一览（续）

项目		参数 No.	内容	设定范围	缺省值	参照
刷新参数详细软元件指定		E007H	指定将多 CPU 共享内存传送至 CPU 软元件时的传送目标软元件及传送点数。可以对各个 CPU 分别进行设定。	可使用的软元件：D、W、R、ZR、B、M、Y	无设定	QCPU 用户手册（多 CPU 系统篇）
多 CPU 间高速通信功能各 CPU 发送范围设定 *1		E008H	设定构成多 CPU 系统的各 CPU 模块中分配的多 CPU 间高速通信区的容量。	简单设定时： 0 ～ 14k 点（单位：1k） 高速设定时： 0 ～ 15k 点（单位：0.5k）	1k 点	
多 CPU 间高速通信功能自动刷新设定 *1		E009H	设定多 CPU 间高速通信区内的用户区中，通过自动刷新功能进行数据传送的范围。	可使用的软元件 *2：X、Y、M、L、B、D、W、R、ZR、SM、SD、SB、SW	—	
多 CPU 间高速通信功能自动刷新软元件设定 *1		E00AH				
多 CPU 间同步启动设定 *1		E00BH	设定构成多 CPU 系统的各 CPU 模块的启动时间是否同步。	1 号机～4 号机	多 CPU 间全部号机同步启动	
本机号设定 *1		E00CH	对设定多 CPU 设定参数的机号进行设定。（设定本机号。）	1 号机～4 号机	无设定	
组外的 I/O 设定	接收组外的输入状态	0E04H	设定是否接收其它机器所管理的输入模块、智能模块的输入状态。	不接收 / 接收组外的输入状态。	不接收组外的输入状态。	
	接收组外的输出状态		设定是否接收其它机器所管理的输出模块的输出状态。	不接收 / 接收组外的输出状态。	不接收组外的输出状态。	
刷新设定		E002H E003H	多 CPU 系统的各 CPU 模块之间，利用自动刷新进行数据的写入 / 读出的软元件与点数。	[设定各 CPU 起始软元件] 各 CPU 起始软元件设定 / 不设定	不设定各 CPU 起始软元件	
				[各 CPU 模块的发送范围] 0~2048 点 (2 点单位)/1 个 最大 4k 点 (4096 点)/1 个系统	无设定	
				[CPU 模块的软元件] B.M.Y.D.R.ZR 占有从指定软元件编号到发送范围中设定的点数的软元件。 ・发送范围 1 点，B，M，Y，占有 16 点。 ・发送范围 1 点，D，W，R，ZR 占有 1 点。		

*1：只有在使用通用型 QCPU（除 Q02UCPU 以外）时才可以设定。
*2：SM、SD、SB、SW 只有在被设定为发送软元件时才有效。

基本 注 8.6

8.2 冗余参数

高性能 注 8.6

过程 注 8.6

通用 注 8.6

对冗余参数的一览与各参数的内容进行说明。

表 8.24 冗余参数

项目	参数 No.	内容	设定范围	缺省值	参照
冗余参数	0D00H	冗余 CPU 中的动作模式以及热备发送设定。	—	—	—

(1) 动作模式设定

设定冗余系统的电源 ON 时的冗余 CPU 的动作。

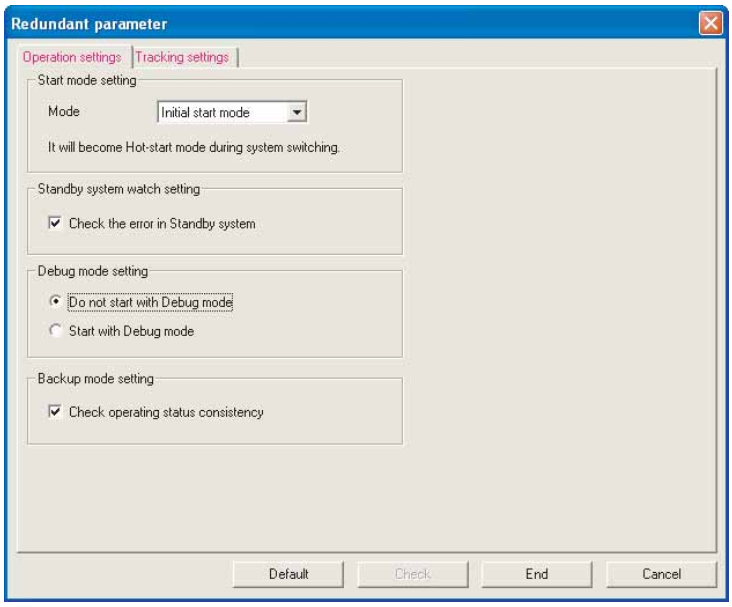


图 8.24 动作模式设定

表 8.25 动作模式设定一览

项目	参数 No.	内容	设定范围	缺省值	参照
电源 ON 时的设定	D001H	设定在电源 OFF → ON 时冗余 CPU 的启动模式。	初始启动模式 / 热启动模式	初始启动模式	QnPRHCPU 用户手册（冗余系统篇）
待机系统监视设定		设定是否监视待机系统的异常。	检查 / 不检查待机系统的异常	检查待机系统的异常	
调试模式设定		设定是否以调试模式启动冗余 CPU。	不以调试模式启动 / 以调试模式启动	不以调试模式启动	
备份模式设定		设定是否检查冗余 CPU 在备份模式下运行时两系统的动作状态的同一性。	检查 / 不检查动作状态同一性	检查动作状态同一性	

基本 注 8.6　高性能 注 8.6　过程 注 8.6

通用 注 8.6

在基本模式 QCPU、高性能模式 QCPU、过程 CPU 和通用型 QCPU 中，不能使用冗余参数。
本参数仅在使用冗余 CPU 时可设定。

(2) 热备设定

进行冗余 CPU 的热备发送功能的相关设定。

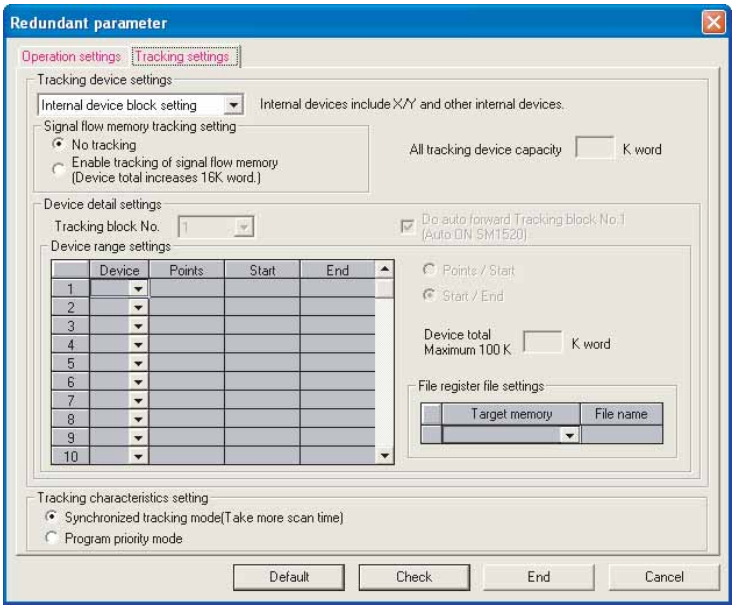


图 8.25 热备设定

表 8.26 热备设定一览

项目	参数 No.	内容	设定范围	缺省值	参照
热备软元件设定	D003H	设定是否任意设定热备发送的软元件数据的范围。	内部软元件批量设定 / 软元件详细设定	内部软元件批量设定	参照 QnPRHCPU 用户手册（冗余系统篇）的“5.5.3 热备发送设定数据”
上升 / 下降执行指令用记录（信号流）		设定是否实施信号流内存的热备。	不进行热备 / 进行热备	不进行热备	
软元件详细设定		用户任意设定热备发送的软元件时，设定热备发送的软元件。	—	—	
热备块号		选定热备发送的软元件的块号。选定块号后，通过软元件范围设定，设定热备软元件。	1~64	1	
自动发送热备块号 1（自动启动 SM1520）		设定是否自动发送热备块号 1	自动发送 / 不自动发送	自动发送	
软元件范围设定		设定热备发送的软元件与范围。	参照 QnPRHCPU 用户手册（冗余系统篇）的“5.5.3 热备发送设定数据”	—	
文件寄存器文件设定		在软元件范围设定中设置了文件寄存器时，设定进行热备发送的文件寄存器的对象存储器和文件名等。		—	
热备发送模式设定	D002H	实施热备发送时的热备发送动作。	热备同步模式 / 程序优先模式	热备同步模式	

备 注

关于热备发送功能与冗余参数设定时的注意事项，请参照以下手册。

☞ QnPRHCPU 用户手册（冗余系统篇）

8.3 网络参数

对网络参数的一览与各参数的内容进行说明。

■ 关于本节的参数 No. 栏中所示的 mn、**、M、N
本节的参数 No. 栏中所示的 mn、**、M、N 如下所示：

mn：表示起始 I/O No. ÷ 16 的值。
**：表示任意值。
N ：表示是第几个模块。
M ：表示网络类别。3

表 8.27 MELSECNET/G注 8.7、注 8.8、MELSECNET/H、Ethernet 设定的情况（☞ 本节 (1),(2),(3)）

高性能 注 8.7
基本 注 8.8　过程 注 8.8　冗余 注 8.8

M	网络类别
1H	MELSECNET/G 模式（控制站）注 8.7、注 8.8 MELSECNET/H 模式（控制站）、MELSECNET/H 扩展模式（控制站）、 MELSECNET/10 模式（控制站）
2H	MELSECNET/G 模式（控制站）注 8.7、注 8.8 MELSECNET/H 模式（一般站）、MELSECNET/H 扩展模式（一般站）、 MELSECNET/10 模式（一般站）
5H	MELSECNET/H（远程主站）
AH	MELSECNET/H（待机站）
BH	MELSECNET/H 模式多重远程 I/O 网络主站
DH	MELSECNET/H 模式多重远程 I/O 网络子站 （未进行参数设置时）
EH	MELSECNET/H 模式多重远程 I/O 网络子站 （进行了参数设置时）

表 8.28 CC-Link 设定的情况（☞ 本节 (4)）

M	网络类别
0H	主站
1H	本地站点
2H	待机主站

高性能 注 8.7

在高性能模式 QCPU 中使用 MELSECNET/G 时，请确认 CPU 模块以及 GX Developer 的版本。
（☞ 附录 4.2）

基本 注 8.8　过程 注 8.8　冗余 注 8.8

在基本模式 QCPU、过程 CPU、冗余 CPU 中，不能使用 MELSECNET/G。

高性能 注 8.9

(1) MELSECNET/G 设定 *注 8.9*, *注 8.10*

设定 MELSECNET/G 的网络参数。

基本 注 8.10　过程 注 8.10　冗余 注 8.10

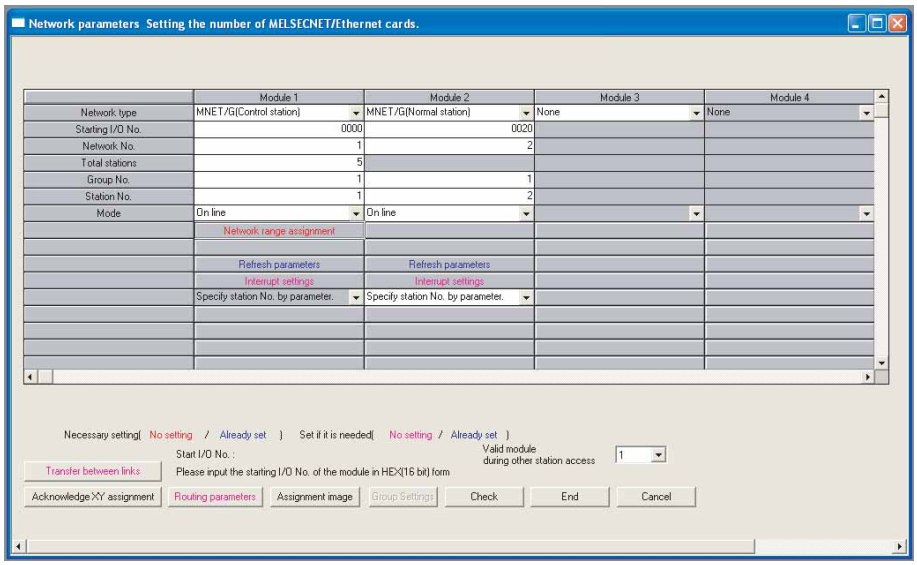


图 8.26 MELSECNET/Ethernet 个数设定 (MELSECNET/G 设定时)

表 8.29 MELSECNET/G 设定一览表

项目	参数号	内容	设定范围	默认值	参照
MELSECNET/G 个数设定	A000H	设定 MELSECNET/G 的网络参数。	参照 Q 系列 MELSECNET/G	----	----
起始 I/O No.	ANM0H				
网络 No.					
链接总（从）站数					
组 No.	0AmnH				
模式	ANM0H				
刷新参数	ANM1H				
公共参数	ANM2H				
站固有参数	ANM3H				
链接间传送（数据链接间传送参数）	A002H				
路由参数	A003H				

高性能 注 8.9

在高性能模式 QCPU 中使用 MELSECNET/G 时，请确认 CPU 模块以及 GX Developer 的版本。
(☞ 附录 4.2)

基本 注 8.10　过程 注 8.10　冗余 注 8.10

在基本模式 QCPU、过程 CPU、冗余 CPU 中，不能使用 MELSECNET/G。

(2) MELSECNET/H 设定

设定 MELSECNET/H 的网络参数。

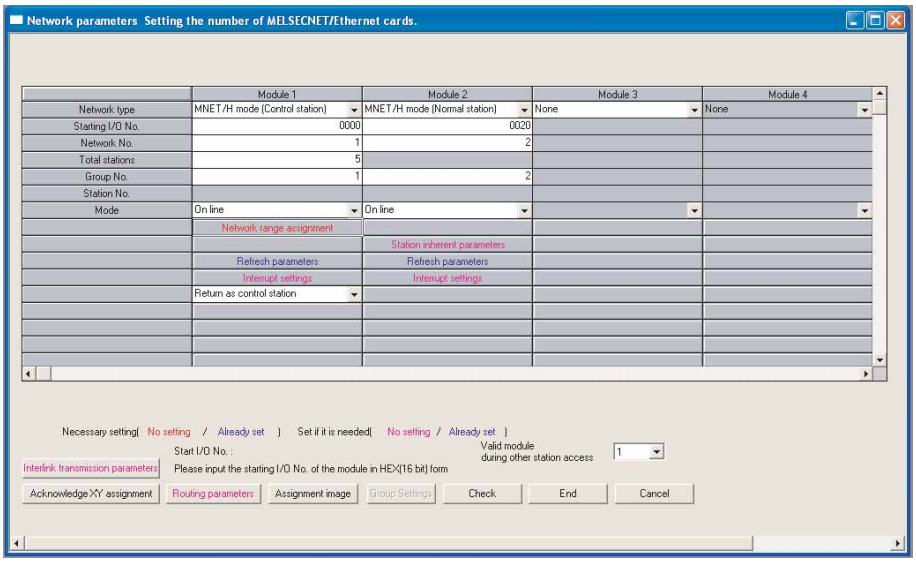


图 8.27 MELSECNET/Ethernet 个数设定 (MELSECNET/H 设定时)

表 8.30 MELSECNET/H 设定一览

项目	参数 No.	内容	设定范围	缺省值	参照
MELSECNET/H 数量设定	5000H	设定 MELSECNET/H 网络参数。	参照 Q 兼容 MELSECNET/H 的手册。	—	—
起始 I/O No.	5NM0H				
网络 No.					
链接总 (子) 站数					
组 No.	05mnH				
模式	5NM0H				
刷新参数	5NM1H				
公共参数	5NM2H				
站点固有参数	5NM3H				
子站用参数 *2	5NM5H				
通用参数 2	5NMAH				
站点固有参数 2	5NMBH				
中断设定					
组设定 *1	D004H				
冗余设定 *1	D5**H				
他站存取时的有效模块	5001H				
链接间发送 (数据链接间发送参数)	5002H				
路由参数	5003H				

*1：仅在使用冗余 CPU 时可设定。
*2：仅在使用过程 CPU 和冗余 CPU 时可设定。

(3) Ethernet 设定

设定 Ethernet 的网络参数。

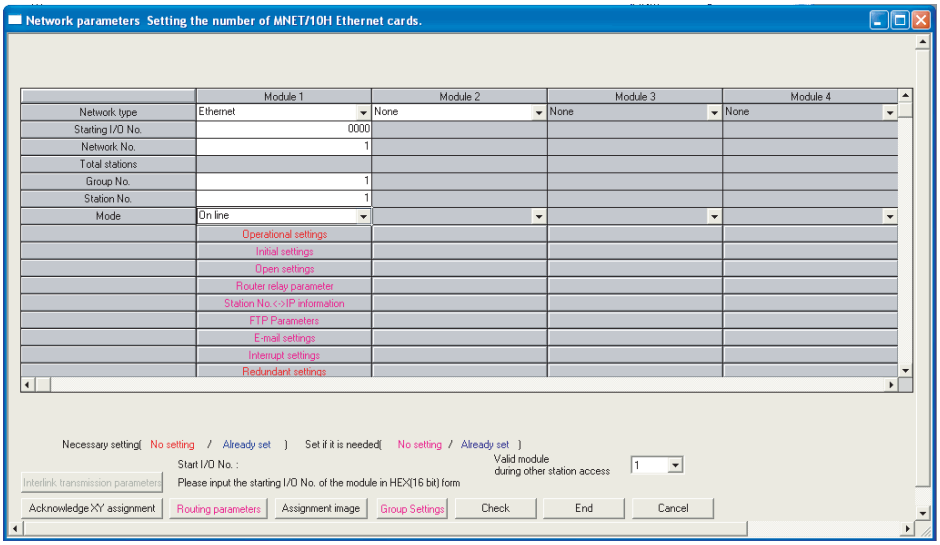


图 8.28 MELSECNET/Ethernet 个数设定 (Ethernet 设定时)

表 8.31 Ethernet 设定一览

项目	参数 No.	内容	设定范围	缺省值	参照
Ethernet 数量设定	9000H	设置 Ethernet 的网络参数	参照 Q 兼容 Ethernet 的手册	—	—
起始 I/O No.	9N00H				
网络 No.					
组 No.					
站点编号					
动作设定					
初始设定	9N01H				
启动设定	9N02H				
路由器中转参数	9N03H				
站点编号 <->IP 相关信息	9N05H				
FTP 参数	9N06H				
电子邮件设定	9N07H				
通告设定	9N08H				
中断设定	9N09H				
冗余设定 *1	D9**H				
其它站访问时的有效模块	5001H				
路由参数	9N04H				
组设定 *1	D004H				

*1：仅在使用冗余 CPU 时可设定。

(4) CC-Link 设定

设定 CC-Link 的参数。

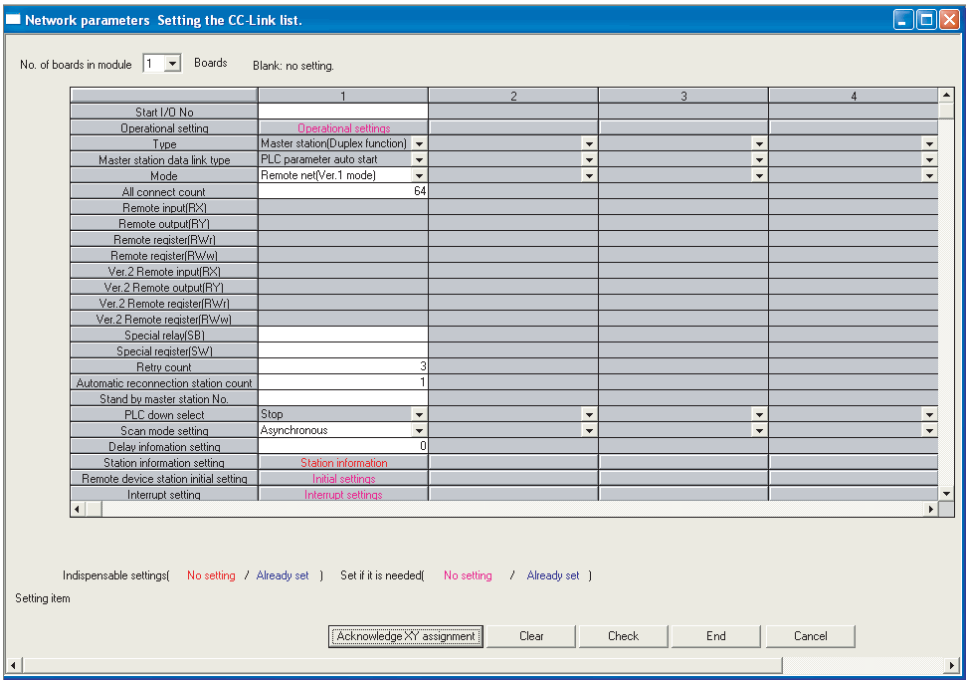


图 8.29 CC-Link 设定

表 8.32 CC-Link 设定一览

项目	参数 No.	内容	设定范围	缺省值	参照
模块数量设定	C000H	设置 CC-Link 的参数。	参照 CC-Link 的手册	—	—
起始 I/O No.	CNM2H				
动作设定					
总连接数量					
远程输入 (RX) 刷新软元件	CNM1H				
远程输出 (RY) 刷新软元件					
远程寄存器 (RWr) 刷新软元件					
远程寄存器 (RWw) 刷新软元件					
Ver.2 远程输入 (RX) 刷新软元件 *1					
Ver.2 远程输出 (RY) 刷新软元件 *1					
Ver.2 远程寄存器 (RWr) 刷新软元件 *1					
Ver.2 远程寄存器 (RWw) 刷新软元件 *1					

*1：只有在使用高性能 QCPU、过程控制 CPU、通用型 QCPU 时才可以设定。

（接下一页）

表 8.32 CC-Link 设定一览（续）

项目	参数 No.	内容	设定范围	缺省值	参照
特殊继电器 (SB) 刷新软元件	CNM1H	设置 CC-Link 的参数。	参照 CC-Link 的手册	—	—
特殊寄存器 (SW) 刷新软元件					
重试次数	CNM2H	设定 CC-Link 的参数。	参照 CC-Link 的手册	—	—
自动恢复数量					
待机主站号					
CPU 宕机指定					
扫描模式指定					
延时设定					
站点信息设定					
远程软元件站点初始设定					
中断设定					

8.4 远程口令

对远程口令的相关参数一览和各参数的内容进行说明。

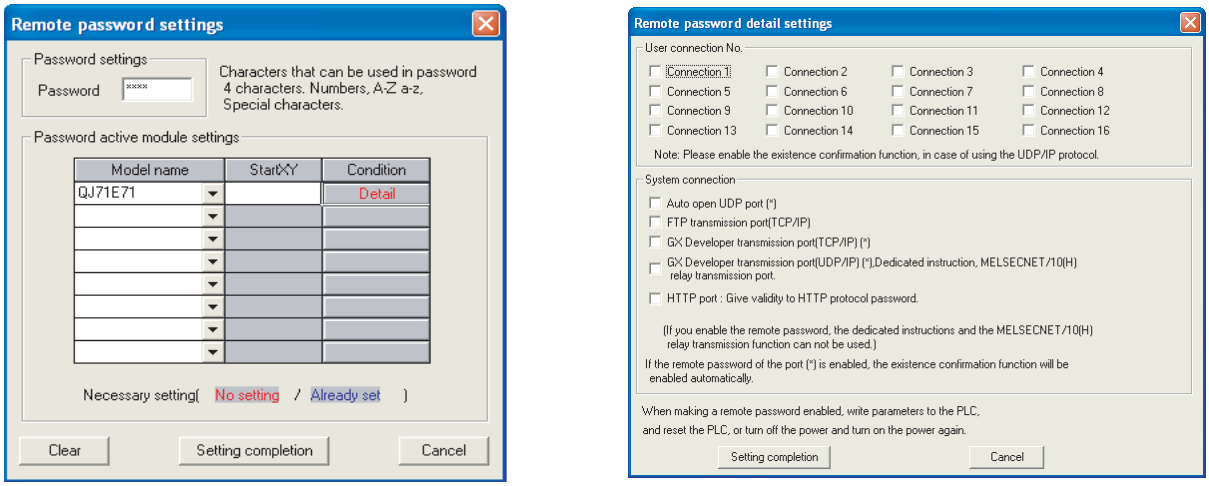


图 8.30 远程口令设定画面

设定 Ethernet 模块、串行口通讯模块和 MODEM 接口模块的远程口令。

表 8.33 远程口令设定一览

项目		参数 No.	内容	设定范围	缺省值	参照
口令设定		—	输入远程口令。	4 字以内（英文，数字，特殊符号）	—	Ethernet 模块，串行口通讯模块，MODEM 接口模块的手册
口令有效模块设定	型号	—	选定对 CPU 模块设定的远程口令进行检查的模块型号。	QJ71E71 QJ71C24/CMO	—	
	起始 XY	—	设定进行远程口令检查的模块的起始地址。	<基本模式 QCPU> 0000H~03E0H <高性能模式 QCPU，过程 CPU，冗余 CPU> 0000H~0FE0H	—	
详细设定		—	设定 QJ71E71 中的远程口令的详细内容。	—	—	
	用户用链接 No. 有效设定	—	设定用户用链接 No.	链接 No. 1~No. 16	—	
	系统用链接有效设定	—	指定系统用链接的远程口令有效端口	指定远程口令有效端口。 • 自动启动 UDP 端口 • FTP 通讯端口（TCP/IP） • GX Developer 通讯端口（TCP/IP） • GX Developer 通讯端口（UDP/IP），专用指令，MELSECNET/10(H) 中转通讯端口 • HTTP 端口	—	

第 9 章　软元件的说明

对可在 CPU 模块中使用的软元件进行说明。

9.1　软元件一览

在 CPU 模块中可使用的软元件名称与使用范围如表 9.1 和表 9.2 所示。

(1) 基本模式 QCPU

表 9.1 软元件一览表

分类	类别	软元件名	缺省值			参数设定范围	参照
			点数	使用范围			
内部用户软元件	位软元件	输入	2048 点	X0~7FF	16 进制	16.4k 字以内可变更 *2	9.2.1 项
		输出	2048 点	Y0~7FF	16 进制		9.2.2 项
		内部继电器	8192 点	M0~8191	10 进制		9.2.3 项
		锁存继电器	2048 点	L0~2047	10 进制		9.2.4 项
		报警器	1024 点	F0~1023	10 进制		9.2.5 项
		变址继电器	1024 点	V0~1023	10 进制		9.2.6 项
		步进继电器	2048 点	S0~127/ 块	10 进制		9.2.9 项
		链接继电器	2048 点	B0~7FF	16 进制		9.2.7 项
		链接特殊继电器	1024 点	SB0~3FF	16 进制		9.2.8 项
	字元件	定时器 *1	512 点	T0~511	10 进制		9.2.10 项
		累计定时器 *1	0 点	(ST0~511)	10 进制		
		计数器 *1	512 点	C0~511	10 进制		9.2.11 项
		数据寄存器	11136 点	D0~11135	10 进制		9.2.12 项
		链接寄存器	2048 点	W0~7FF	16 进制		9.2.13 项
		链接特殊寄存器	1024 点	SW0~3FF	16 进制		9.2.14 项
内部系统软元件	位软元件	功能输入	16 点	FX0~F	16 进制	禁止	9.3.1 项
		功能输出	16 点	FY0~F	16 进制		9.3.1 项
		特殊继电器	1000 点	SM0~999	10 进制		9.3.2 项
	字元件	功能寄存器	5 点	FD0~4	10 进制		9.3.1 项
		特殊寄存器	1000 点	SD0~999	10 进制		9.3.3 项
链接直接软元件	位软元件	链接输入	8192 点	Jn\X0~1FFF	16 进制	禁止	9.4 节
		链接输出	8192 点	Jn\Y0~1FFF	16 进制		
		链接继电器	16384 点	Jn\B0~3FFF	16 进制		
		链接特殊继电器	512 点	Jn\SB0~1FF	16 进制		
	字元件	链接寄存器	16384 点	Jn\W0~3FFF	16 进制		
		链接特殊寄存器	512 点	Jn\SW0~1FF	16 进制		
智能功能模块软元件	字元件	智能功能模块软元件	65536 点	Un\G0~65535*3	10 进制	禁止	9.5 节
变址寄存器	字元件	变址寄存器	10 点	Z0~9	10 进制	禁止	9.6 节

（接下一页）

表 9.1 软元件一览表（续）

分类	类别	软元件名		缺省值			参数设定范围	参照
				点数	使用范围			
文件寄存器	字元件	文件寄存器	Q00JCPU	0 点	-	-	禁止	9.7 节
			Q00CPU，Q01CPU	64k 点	R0~32767 ZR0~65535	10 进制		
嵌套结构	—	嵌套结构		15 点	N0~14	10 进制	禁止	9.8 节
指针	—	指针		300 点	P0~299	10 进制	禁止	9.9 节
		中断指针		128 点	I0~127	10 进制		9.10 节
其它	位软元件	SFC 块软元件		128 点	BL0~127	10 进制	禁止	9.11.1 项
	—	网络 No. 指定软元件		239 点	J1~239	10 进制	禁止	9.11.3 项
		I/O 号指定软元件	Q00JCPU	—	U0~F	16 进制		9.11.4 项
			Q00CPU，Q01CPU	—	U0~3F	16 进制		
	—	宏指令变量软元件		—	VD0~□	10 进制	禁止	9.11.5 项
常数	—	10 进制常数		K-2147483648~2147483647				9.12.1 项
		16 进制常数		H0~FFFFFFFF				9.12.2 项
		实数常数		E±1.17550-38~E±3.40282+38				9.12.3 项
		字符串常数		“ABC”，“123” *4				9.12.4 项

*1: 定时器、累计定时器、计数器中，触点・线圈变为位软元件，当前值变为字元件。
*2: 可通过 GX Developer 的可编程控制器参数进行变更。（输入、输出、步进继电器、链接特殊继电器和链接特殊寄存器除外。）(☞9.2 节)
*3: 实际上可使用的点数因智能功能模块而异。
对于缓冲存储器的点数，请参照所使用的智能功能模块的手册。
*4: 字符串只能在 $MOV，STR，DSTR，VAL，DVAL，ESTR，EVAL 指令中使用。
其它的指令不能使用字符串。

(2) 高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU

表 9.2 软元件一览表

分类	类别	软元件名	缺省值			参数设定范围	参照
			点数	使用范围			
内部用户软元件	位软元件	输入	8192 点	X0~1FFF	16 进制	29k 字以内可变更 *3	9.2.1 项
		输出	8192 点	Y0~1FFF	16 进制		9.2.2 项
		内部继电器	8192 点	M0~8191	10 进制		9.2.3 项
		锁存继电器	8192 点	L0~8191	10 进制		9.2.4 项
		报警器	2048 点	F0~2047	10 进制		9.2.5 项
		变址继电器	2048 点	V0~2047	10 进制		9.2.6 项
		步进继电器	8192 点	S0~511/ 块	10 进制		9.2.9 项
		链接特殊继电器	2048 点	SB0~7FF	16 进制		9.2.8 项
		链接继电器	8192 点	B0~1FFF	16 进制		9.2.7 项
	字元件	定时器 *1	2048 点	T0~2047	10 进制		9.2.10 项
		累计定时器 *1	0 点	(ST0~2047)	10 进制		
		计数器 *1	1024 点	C0~1023	10 进制		9.2.11 项
		数据寄存器	12288 点	D0~12287	10 进制		9.2.12 项
		链接寄存器	8192 点	W0~1FFF	16 进制		9.2.13 项
		链接特殊寄存器	2048 点	SW0~7FF	16 进制		9.2.14 项
内部系统软元件	位软元件	功能输入	5 点	FX0~F	16 进制	禁止	9.3.1 项
		功能输出	5 点	FY0~F	16 进制		9.3.1 项
		特殊继电器	2048 点	SM0~2047	10 进制		9.3.2 项
	字元件	功能寄存器	5 点	FD0~4	10 进制		9.3.1 项
		特殊寄存器	2048 点	SD0~2047	10 进制		9.3.3 项
链接直接软元件	位软元件	链接输入	8192 点	Jn\X0~1FFF	16 进制	禁止	9.4 节
		链接输出	8192 点	Jn\Y0~1FFF	16 进制		
		链接继电器	16384 点	Jn\B0~3FFF	16 进制		
		链接特殊继电器	512 点	Jn\SB0~1FF	16 进制		
	字元件	链接寄存器	16384 点	Jn\W0~3FFF	16 进制		
		链接特殊寄存器	512 点	Jn\SW0~1FF	16 进制		
模块访问软元件	字元件	智能功能模块软元件	65536 点	Un\G0~65535*2	10 进制	禁止	9.5 节
		多 CPU 间共享软元件 *4	4096 点 (14336 点)*5	U3En\G0 ～ 4095(U3En\G10000 ～ 24335) *6	10 进制	可以	
变址寄存器 / 通用运算寄存器	字元件	变址寄存器 / 通用运算寄存器	16 点 (20 点)*5	Z0 ～ 15 (Z0 ～ 19)*6	10 进制	禁止	9.6 节
文件寄存器	字元件	文件寄存器	0 点	—	—	0~1018k 点 (0~4086k 点) *5	9.7 节
嵌套结构	—	嵌套结构	15 点	N0~14	10 进制	禁止	9.8 节
指针	—	指针	4096 点	P0~4095	10 进制	禁止	9.9 节
		中断指针	256 点	I0~255	10 进制		9.10 节

(接下一页)

表 9.2 软元件一览表（续）

分类	类别	软元件名	缺省值			参数设定范围	参照
			点数	使用范围			
其它	位软元件	SFC 块软元件	320 点	BL0~319	10 进制	禁止	9.11.1 项
		SFC 转移软元件 *9	512 点	TR0~511	10 进制		9.11.2 项
	—	网络 No. 指定软元件	255 点	J1~255	10 进制		9.11.3 项
		I/O 地址号指定软元件	—	U0~FF	16 进制		9.11.4 项
		宏指令变量软元件	—	VD0~□	16 进制		9.11.5 项
常数	—	10 进制常数	K-2147483648~2147483647				9.12.1 项
		16 进制常数	H0~FFFFFFFF				9.12.2 项
		实数常数	单精度浮动小数点数据： E±1.17549435-38 ～ E±3.40282347+38				9.12.3 项
			双精度浮动小数点数据 *5*8： E±2.2250738585072014-308 ～ E±1.7976931348623157 +308				9.12.3 项
		字符串常数	“ABC”，“123”				9.12.4 项

基本 注 9.1　冗余 注 9.1　通用 UD 注 9.1

*1 ： 定时器、累计定时器、计数器中，触点・线圈变为位软元件，当前值变为字元件。
*2 ： 实际上可使用的点数因智能功能模块 / 特殊功能模块注 9.1 而异。
对于缓冲存储器的点数，请参照所使用的智能功能模块 / 特殊功能模块的手册。
*3 ： 可通过 GX Developer 的可编程控制器参数进行变更。（输入、输出、步进继电器、链接特殊继电器和链接特殊寄存器除外。）（☞ 9.2 节）
*4 ： 只有在多 CPU 系统构成时才可以使用。
*5 ： 括号内为使用通用型 QCPU 时的点数。
*6 ： 括号内为只有在使用通用型 QCPU 时才可以使用的点数。
*7 ： 只有在通用型 QCPU 中才可以使用双精度浮动小数点。
*8 ： 通过 GX Developer 可输入的位数为 15 位。
*9 ： 在通用型 QCPU 中，不能使用 SFC 转移软元件。

基本 注 9.1　冗余 注 9.1　通用 UD 注 9.1

在基本模式 QCPU、过程 CPU 和冗余 CPU 中，不能使用特殊功能模块。

9.2 内部用户软元件

(1) 内部用户软元件

内部用户软元件是指可按照用户的用途使用的软元件。
内部用户软元件预先设定了可使用的点数（缺省值）。
通过可编程控制器参数的软元件设定可变更可使用点数。

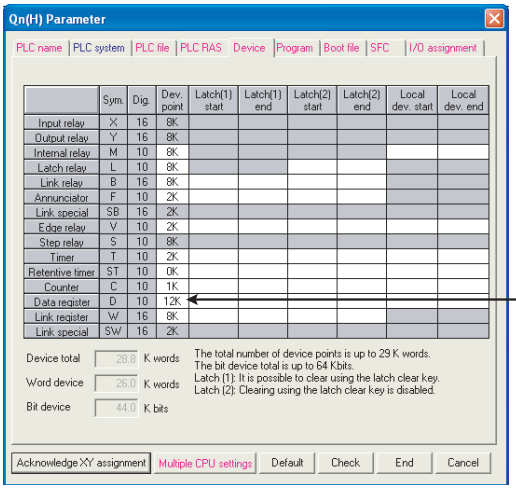


图 9.1 可编程控制器参数的软元件设定

(2) 内部用户软元件的设定范围

通用 UD 注 9.2

除 CPU 模块的输入 (X)、输出 (Y)、步进继电器 (S)、链接特殊继电器 (SB) 注 9.2、链接特殊寄存器 (SW) 注 9.2 以外的内部用户软元件，均可通过可编程控制器参数的软元件设定在表 9.3 所示的范围内变更使用点数。

表 9.3 内部软元件的设定范围

CPU 模块	可变更范围
基本模式 QCPU	16.4K 字 （含以上软元件部分 1.5k 字）
高性能模式 QCPU，过程 CPU 冗余 CPU 通用型 QCPU	29K 字 （含以上软元件部分 3.7k 字）

以下就变更内部用户软元件点数时的想法进行说明。

通用 UD 注 9.2

在通用型 QCPU 中，可以通过可编程控制器参数的软元件设定在表 9.3 所示的范围内变更链接特殊继电器 (SB)、链接特殊寄存器 (SW) 的使用点数。

(a) 关于设定范围

1 个软元件以 16 点为单位进行设定。

1 个软元件的最大点数为 32k 点。

定时器、累计定时器和计数器的 1 点，按照线圈 1 点和触点 1 点共计 2 点进行计算。

要 点

在高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 中，内部继电器、锁存继电器、报警器、变址继电器、链接继电器、链接特殊继电器、步进继电器、定时器、累计定时器和计数器的合计点数最大为 64k 点。

(3) 内存容量的思路

内部用户软元件的设定需满足以下公式。

(基本模式 QCPU)

(位软元件容量)+(字元件容量)+(定时器、累计定时器、计数器容量)≤16.4k 字

(高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU)

(位软元件容量)+(字元件容量)+(定时器、累计定时器、计数器容量)≤29k 字

(a) 位软元件的情况

位软元件以 16 点作为 1 字进行计算。

$$(\text{位软元件容量}) = \frac{(X+Y+M+L+B+F+SB+V+S)}{16} (\text{字})$$

(b) 定时器 (T)、累计定时器 (ST)、计数器 (C) 的情况

定时器 (T)、累计定时器 (ST)、计数器 (C) 是以 16 点作为 18 字进行计算。

$$(\text{定时器、累计定时器、计数器}) = \frac{(T+ST+C)}{16} \times 18(\text{字})$$

(c) 字元件的情况

对于数据寄存器 (D)、链接寄存器 (W)、链接特殊寄存器 (SW)，是将 16 点作为 16 字进行计算。

$$(\text{字元件容量}) = \frac{(D+W+SW)}{16} \times 16(\text{字})$$

要 点

利用参数变更内部用户软元件的使用点数后，按照变更前的参数编制的以下文件将不能原封不动地使用。

- 顺控程序
- SFC 程序
- ST 程序

变更内部用户软元件的使用点数后，请用 GX Developer 从 CPU 模块读出以上程序，并将其重新写入 CPU 模块。

(4) 软元件点数分配示例

高性能模式 QCPU 中的软元件点数分配示例如表 9.4 所示。

表 9.4 采用的是附录 7 中记载的软元件点数分配表。

表 9.4 软元件点数分配示例（高性能模式 QCPU 的情况）

软元件名	符号	进制	软元件点数 *1*2		限制检查			
			点数	编号	容量（字）*3		位点数 *2	
输入继电器	X	16	8k(8192) 点	X0000~1FFF	÷16	512 字	×1	8192 点
输出继电器	Y	16	8k(8192) 点	Y0000~1FFF	÷16	512 字	×1	8192 点
内部继电器	M	10	16k(16384) 点	M0~16383	÷16	1024 字	×1	16384 点
锁存继电器	L	10	4k(4096) 点	L0~4095	÷16	256 字	×1	4096 点
链接继电器	B	16	4k(4096) 点	B0000~0FFF	÷16	256 字	×1	4096 点
报警器	F	10	1k(1024) 点	F0~1023	÷16	64 字	×1	1024 点
链接特殊继电器	SB	16	2k(2048) 点	SB0000~07FF	÷16	128 字	×1	2048 点
变址继电器	V	10	1k(1024) 点	V0~1023	÷16	64 字	×1	1024 点
步进继电器	S	10	8k(8192) 点	S0~8191	÷16	512 字	×1	8192 点
定时器	T	10	2k(2048) 点	T0~2047	$\times\frac{18}{16}$	2304 字	×2	4096 点
累计定时器	ST	10	2k(2048) 点	ST0~2047	$\times\frac{18}{16}$	2304 字	×2	4096 点
计数器	C	10	1k(1024) 点	C0~1023	$\times\frac{18}{16}$	1152 字	×2	2048 点
数据寄存器	D	10	14k(14336) 点	D0~14335	×1	14336 字	—	
链接寄存器	W	16	4k(4096) 点	W0000~4095	×1	4096 字	—	
链接特殊寄存器	SW	16	2k(2048) 点	SW0000~07FF	×1	2048 字	—	
软元件合计						29568 字 (29696 字以下)		63488 点 (65536 点以下)

*1 : □内的点数是由系统所固定的。（不可变更）

*2 : 1 个软元件的最大点数为 32k 点。

*3 : 将软元件点数乘以（或者除以）容量（字）栏中的数字后的数值填写进去。

9.2.1 输入 (X)

(1) 输入

输入是指通过按钮、切换开关、限位开关、数字开关等外围设备，将指令与数据送到 CPU 模块。

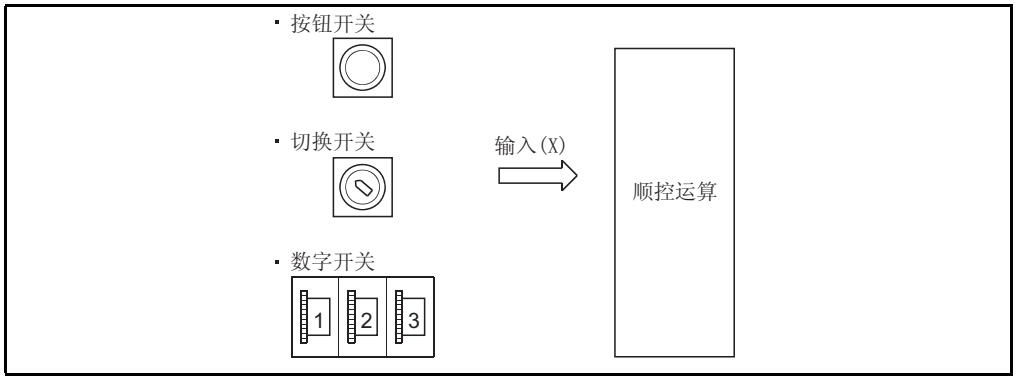


图 9.2 外围设备向 CPU 模块发出的指令

(2) 输入 (X) 的思路

关于输入点，可认为 CPU 模块内内置了假想的继电器 Xn，程序中使用该 Xn 的 a 触点和 b 触点。

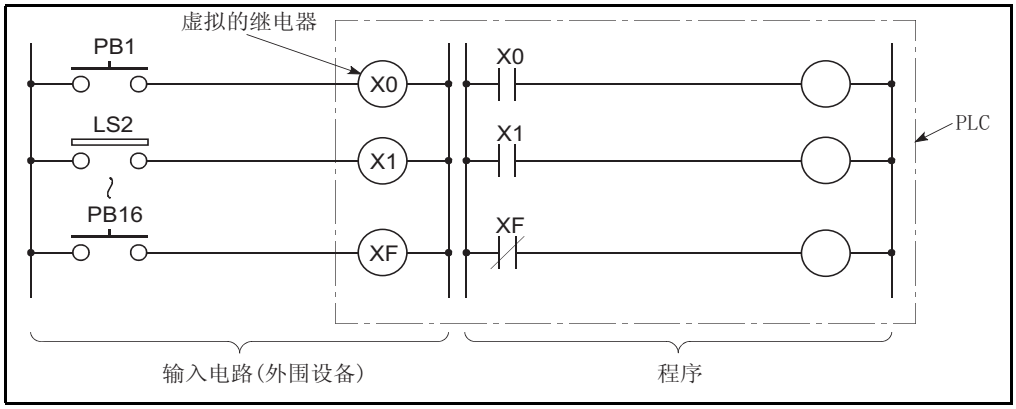


图 9.3 输入 (X) 的思路

(3) a 触点与 b 触点的使用数

程序内的 Xn 的 a 触点和 b 触点的使用数，如果在程序容量的范围之内则无限制。

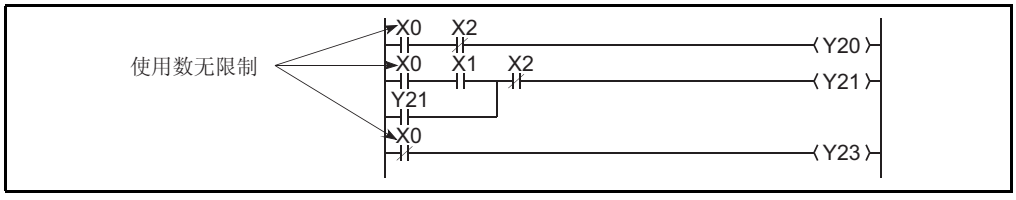


图 9.4 输入 (X) 在程序中的使用

要 点

1. 调试编写好的程序时，可用以下方法使输入 (X) ON/OFF。
   - GX Developer 的软元件测试
   - OUT Xn 指令

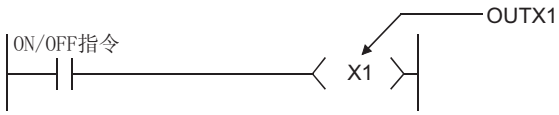


图 9.5 使用 OUT Xn 指令使输入 (X) ON/OFF。

2. 输入 (X) 在以下情况下也可使用。
   - CC-Link 的 RX 的刷新目标 (CPU 模块侧) 软元件
   - MELSECNET/G 注 9.3、注 9.4 或者 MELSECNET/H 的链接刷新目标 (CPU 模块侧) 软元件

高性能 注 9.3

基本 注 9.4　过程 注 9.4　冗余 注 9.4

高性能 注 9.3

在高性能模式 QCPU 中使用 MELSECNET/G 时，请确认 CPU 模块以及 GX Developer 的版本。
(附录 4.2)

基本 注 9.4　过程 注 9.4　冗余 注 9.4

在基本模式 QCPU、过程 CPU、冗余 CPU 中，不能使用 MELSECNET/G。

9.2.2 输出 (Y)

(1) 输出

输出是指将程序的控制结果向外部的信号灯、数字显示器、电磁开关（接触器）和螺线管等输出。

可向外部拿出相当于 1 个 a 触点的输出量。

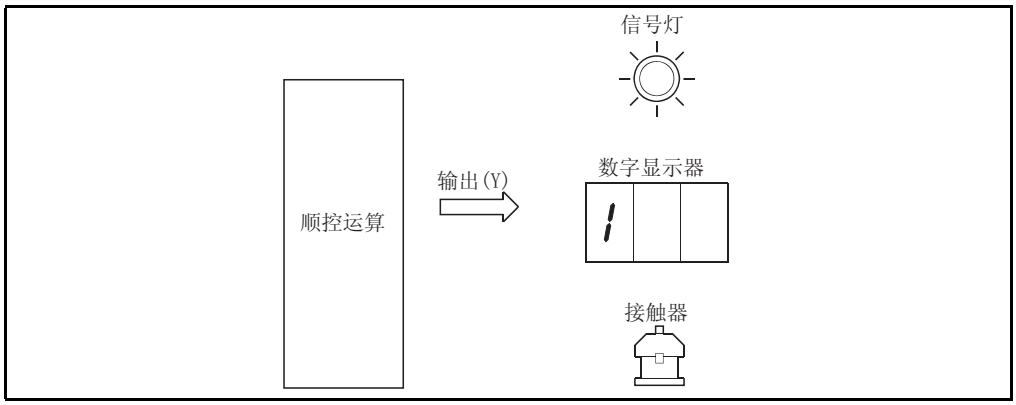


图 9.6 从 CPU 模块向外围设备的输出

(2) a 触点与 b 触点的使用数

程序内的输出 Yn 的 a 触点与 b 触点的使用数，如果在程序容量的范围内则无限制。

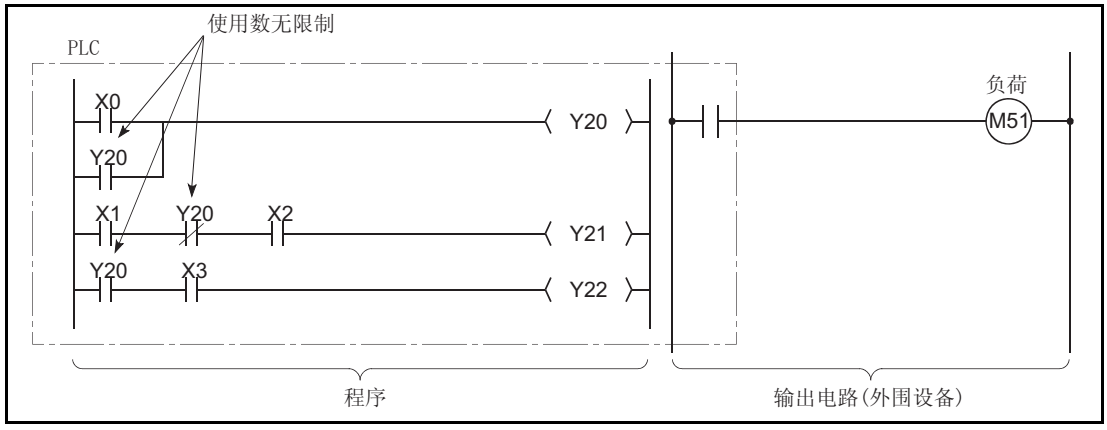


图 9.7 输出 (Y) 在程序中的使用

(3) 代替内部继电器 (M) 的使用

安装了输入模块的区域以及未安装模块的区域对应的输出 (Y)，可代替内部继电器 (M) 使用。

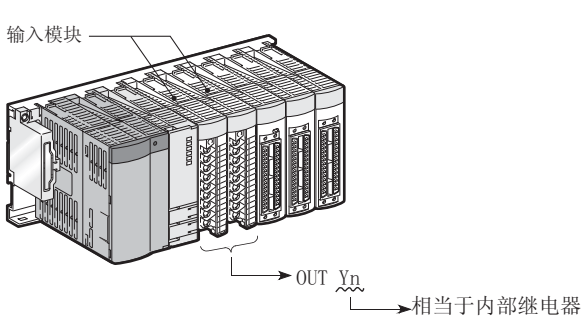


图 9.8 内部继电器的代用

9.2.3 内部继电器 (M)

(1) 内部继电器

内部继电器是指在 CPU 模块内部使用的辅助继电器。
进行以下操作将使所有内部继电器 OFF。

- 可编程控制器的电源 OFF→ON 时
- CPU 模块的复位操作时
- 锁存清除时 (☞6.3 节)

(2) 锁存 (停电保持) 的可否

(a) 基本模式 QCPU、高性能模式 QCPU、过程 CPU、通用型 QCPU 的情况
内部继电器不能锁存 (停电保持)。

(b) 冗余 CPU 的情况
启动模式为热启动时，可进行锁存 (停电保持)。启动模式为初始启动时，不进行锁存 (停电保持)。

(3) a 触点与 b 触点的使用数

程序内的触点 (a 触点、b 触点) 的使用数，如果在程序容量的范围内则无限制。

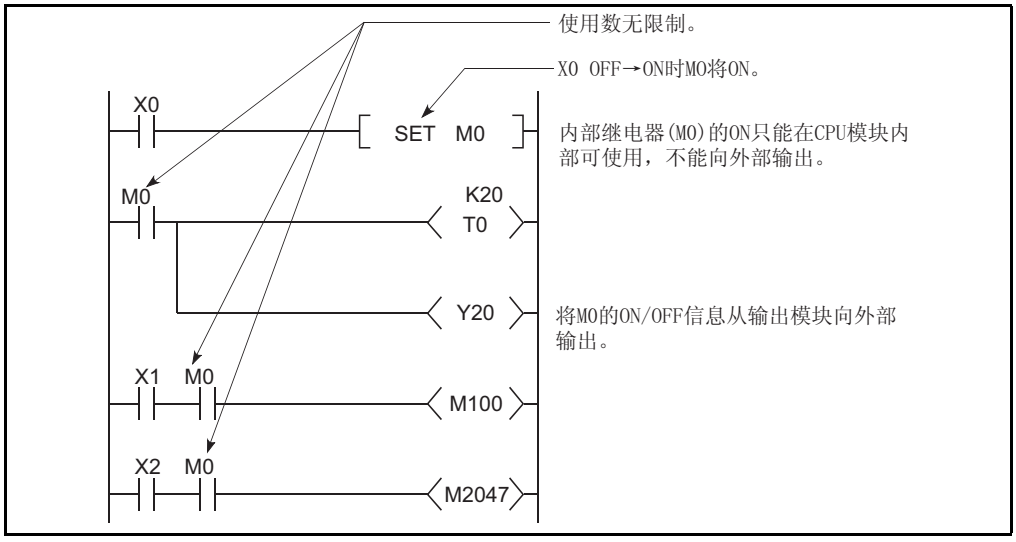


图 9.9 内部继电器在程序中的使用

(4) 向外部输出的方法

将顺控程序的运算结果向外部输出时，使用输出 (Y)。

要 点

需要进行锁存 (停电保持) 时，请使用锁存继电器 (L)。(☞9.2.4 项)

9.2.4 锁存继电器 (L)

(1) 锁存继电器

锁存继电器是指在 CPU 内部使用的能够进行锁存（停电保持）的辅助继电器。
进行以下操作后锁存继电器仍将保持运算结果 (ON/OFF 信息)。

- 可编程控制器的电源 OFF→ON 时
- CPU 模块复位操作时

锁存是通过 CPU 模块本体的电池进行的。

(2) 锁存继电器的清除

基本 通用 注 9.5 注 9.5

锁存清除操作可使锁存继电器 OFF。
但通过可编程控制器参数的软元件设定已设为 “锁存 (2) 起始 / 最终”的锁存继电器，即使通过 RESET/L.CLR 开关注 9.5/ 远程锁存清除进行锁存清除，也不会 OFF。

(3) a 触点与 b 触点的使用数

程序内的触点 (a 触点、b 触点) 的使用数，如果在程序容量的范围内则无限制。

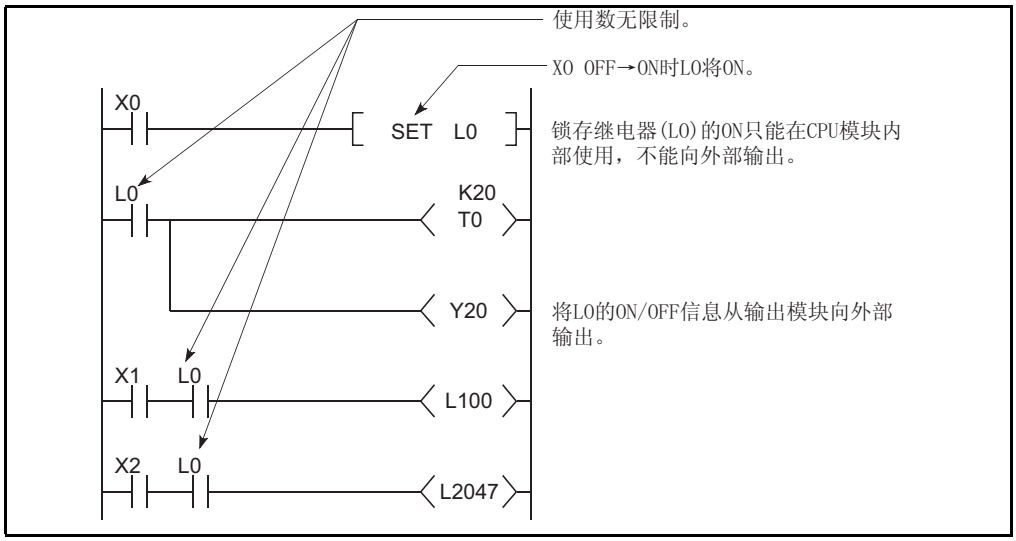


图 9.10 锁存继电器

要 点

在通用型 QCPU 中，如果使用锁存继电器 (L)，扫描时间将延迟。通过删减锁存继电器 (L) 的点数，可以缩短扫描时间的延迟时间。(☞ 10.1.2 项 (11))

基本 通用 注 9.5 注 9.5

在基本模式 QCPU、通用型 QCPU 中，不能通过开关操作进行锁存清除。

(4) 向外部输出的方法

将顺控程序的运算结果向外部输出时，使用输出 (Y)。

要 点

1. 不需要进行锁存时，请使用内部继电器 (M)。
   (☞9.2.3 项)
2. 关于锁存清除，可通过可编程控制器参数的软元件设定来设定每个软元件的锁存清除的无效范围。(☞6.3 节)

9.2.5 报警器 (F)

(1) 报警器

报警器是指能够给用户编写的用来检测设备异常和故障的程序带来便利的内部继电器。

(2) 报警器 ON 时的特殊继电器和特殊寄存器

报警器 ON 后，特殊继电器 (SM62) 将 ON，ON 的报警器的个数和编号将被存储到特殊寄存器 (SD62~79) 中。

- 特殊继电器 : SM62 ..... 只要有 1 个报警器 ON，特殊继电器就会 ON。
- 特殊寄存器 : SD62 ..... 存储最先 ON 的报警器编号。
  : SD63 ..... 存储 ON 的报警器个数。
  : SD64~79..... 按照 ON 的顺序存储报警器编号。
  (SD62 与 SD64 可存储同一报警器编号。)

此外，SD62 中存储的报警器编号还将被登陆到故障历史记录存储区中。

要 点

在基本模式 QCPU 中，可编程控制器的电源 ON 时存储到故障历史记录存储区中的报警器编号仅为 1 个。

(3) 报警器的用途

在故障检测程序中使用报警器后，在特殊继电器 (SM62)ON 时对特殊寄存器 (SD62~79) 进行监视，便可确认设备是否有异常和故障 (报警器编号)。

示例

报警器 (F5)ON 后，将 ON 的报警器编号向外部输出的程序

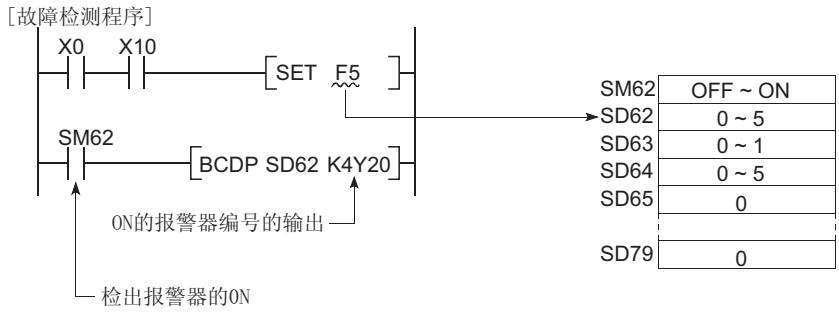


图 9.11 报警器的 ON 检测与存储

(4) a 触点与 b 触点的使用数

程序内的触点 (a 触点、b 触点) 的使用数，如果在程序容量的范围内则无限制。

(5) 报警器的 ON 方法与处理内容

(a) 报警器的 ON 方法
通过以下指令可使报警器 ON。

1) SET F□指令
SET F□指令只有在输入条件成立时 (OFF→ON) 使报警器 ON。
输入条件 OFF 后报警器仍然保持 ON 状态。
使用报警器较频繁时，使用 OUT F□指令可缩短扫描时间。

2) OUT F□指令
使用 OUT F□指令也可使报警器 ON/OFF，但需要进行扫描处理，因此比使用 SET F□指令速度要慢。
此外，使用 OUT F□指令使报警器 OFF 后，需要执行 RST F□指令 /LEDR 指令 注 9.6/BKRST 指令，因此请使用 SET F□指令来使报警器 ON。

基本 注 9.6

要 点

1. 如果使用 SET F□指令或 OUT F□指令以外的其它指令（如 MOV 指令）来使报警器 ON，将与内部继电器的动作相同。
不进行 SM62 的 ON 以及向 SD62、SD64~79 的报警器编号的存储。
2. 使用冗余 CPU 的热备发送功能，将报警器从控制系统热备到待机系统时，报警器在待机系统中不会能 ON。
关于热备报警器时的处理，请参照以下手册。
☞QnPRHCPU 用户手册（冗余系统篇）

基本 注 9.6

在基本模式 QCPU 中，不能使用 LEDR 指令。

(b) 报警器 ON 时的处理内容

1) 特殊寄存器 (SD62~79) 的存储数据

- ON 的报警器编号依次存储在 SD64~79 中。
- SD64 中存储的报警器编号被存储到 SD62 中。
- SD63 的内容将被 +1。

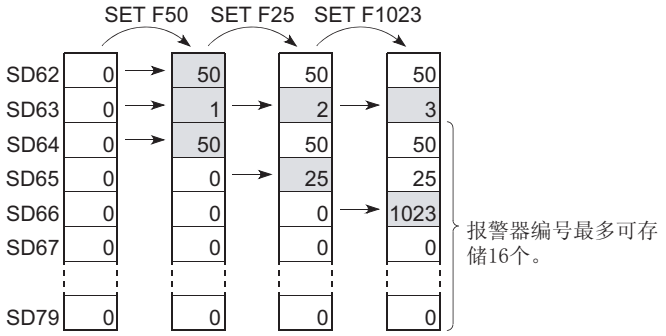


图 9.12 报警器 ON 时的处理内容

2) 在主单元中的处理

- 使用基本模式 QCPU 时
  模块全部的 ERR.LED 亮灯。
- 使用高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 时
  模块前面的 USER LED(红色) 亮灯。

3) LED 的亮灯 / 灭灯的选择

通过对 SD207 ～ 209 设定发生错误时的显示优先顺序，可以选择报警器 ON 时 USER.LED 的亮灯 / 灭灯。(在基本模式 QCPU 中，为 ERR.LED)

(6) 报警器的 OFF 方法与处理内容

(a) 报警器的 OFF 方法

使用以下指令可使报警器 OFF。

1) RST F□指令

通过 SET F□指令 ON 的报警器编号，通过 RST F□指令使其 OFF。

基本 注 9.7

2) LEDR 指令*注 9.7*

LEDR 指令用于使 SD62、SD64 中存储的报警器编号 OFF。

3) BKRST 指令

BKRST 指令用于使指定范围内的报警器编号成批 OFF。

4) OUT F□指令

OUT F□指令可使报警器编号 ON 也可使用 OFF。

但使用 OUT F□指令使报警器 OFF 后，不会进行本项 (6)(b) 中的报警器 OFF 时的处理。

使用 OUT F□指令使报警器 OFF 后，需要执行 RST F□指令、LEDR 指令或 BKRST 指令。

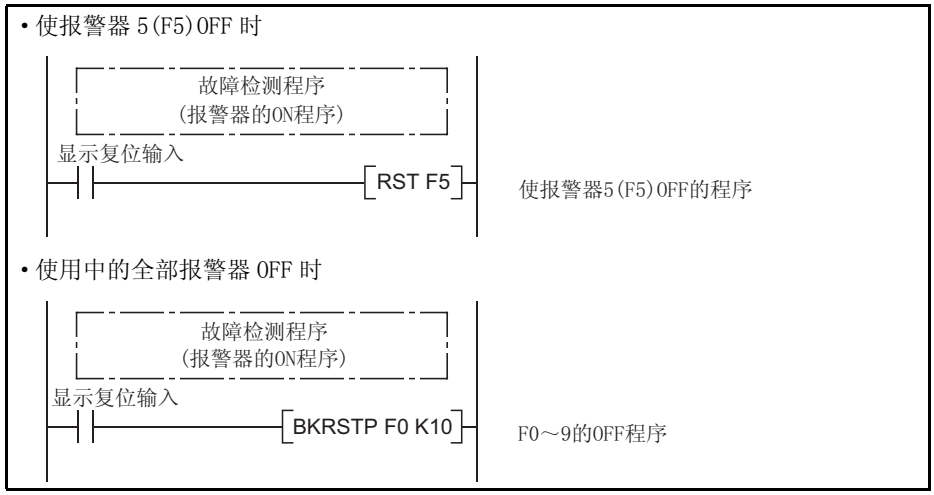


图 9.13 使报警器 OFF 的示例

备 注

关于各指令的详细内容，请参照以下手册。

☞QCPU(Q 模式)/QnACPU 编程手册（公共指令篇）

基本 注 9.7

在基本模式 QCPU 中，不能使用 LEDR 指令。

(b) 报警器 OFF 时的处理内容

基本
注 9.8

1) 执行 LEDR 指令时的特殊寄存器 (SD62~79) 的存储数据*注 9.8*

- SD64 中存储的报警器编号被清除，SD65 以后的寄存器中存储的报警器编号向前靠齐。
- SD64 中存储的报警器编号被存储到 SD62 中。
- SD63 的内容将被 -1。
- SD63 变为 0 时，SM62 将 OFF。

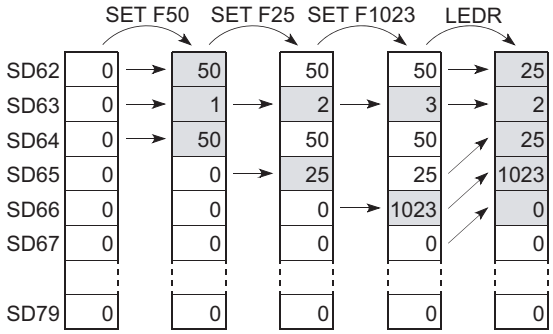


图 9.14 报警器 OFF 时的处理内容（执行 LEDR 指令时）

基本
注 9.8

在基本模式 QCPU 中，不能使用 LEDR 指令。

2) 通过执行 RST F□指令或 BKRST 指令使报警器 OFF 后的特殊寄存器 (SD62~79) 的存储数据。

- 使用 RST 指令 /BKRST 指令指定的报警器编号被清除，存储在被清除的报警器编号之后的报警器编号将向前靠齐。
- SD64 中存储的报警器编号 OFF 后，新存储到 SD64 中的报警器编号被存储到 SD62 中。
- SD63 的内容将被 “-1”。
- SD63 变为 0 时，SM62 将 OFF。

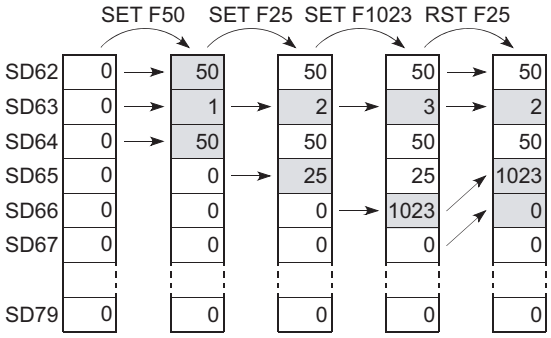


图 9.15 报警器 OFF 时的处理内容（执行 REST F□指令时）

3) LED 的显示

SD64~79 的报警器编号全部 OFF 时，报警器 ON 时亮灯的 LED（参照本项 (5)(b)）将灭灯。

要 点

报警器 ON，以及同时发生了比报警器优先顺序（☞6.21.2 项）更高的运算继续运行错误时，执行 LEDR 指令将解除比报警器优先顺序更高的错误。因此，即使执行 LEDR 指令报警器也不会 OFF。

基本 注 9.9

使用 LEDR 指令使报警器 OFF 时，请先消除比报警器优先顺序更高的错误因子后再执行 LEDR 指令。*注 9.9*

基本 注 9.9

在基本模式 QCPU 中，不能使用 LEDR 指令。

9.2.6 变址继电器 (V)

(1) 变址继电器

变址继电器是记忆梯形图块的起始运算结果 (ON/OFF 信息) 的软元件，只能作为触点使用。

变址继电器不能作为线圈使用。

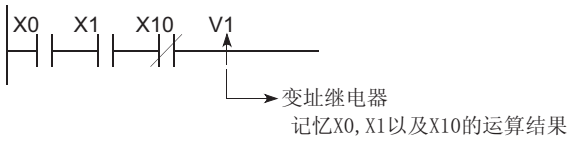
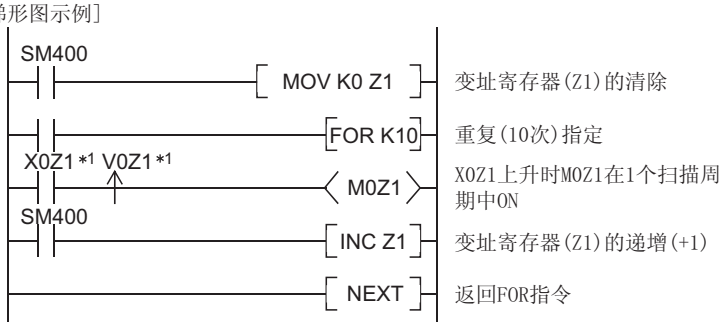


图 9.16 变址继电器

(2) 变址继电器的用途

变址继电器是在变址修饰的结构化程序中由于上升 (OFF → ON) 捡出而执行时使用。

[梯形图示例]



*1 : 将 X0Z1 的 ON/OFF 信息记忆在变址寄存器 V0Z1 中。
例如 X0 的 ON/OFF 信息记忆在 V0 中，X1 的 ON/OFF 信息记忆在 V1 中。

[时序图]

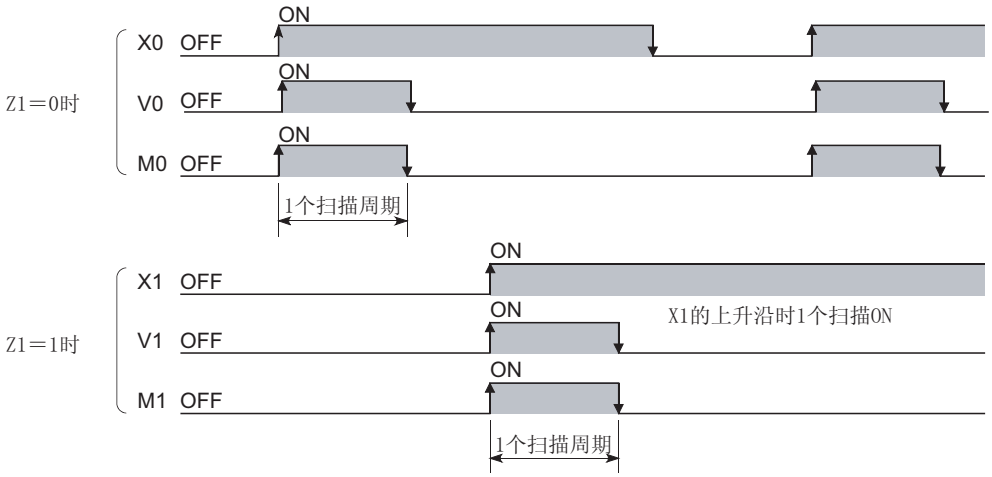


图 9.17 变址继电器的使用示例与时序图。

(3) 注意事项

在程序中，不能使用多个编号相同的变址继电器。

9.2.7 链接继电器 (B)

(1) 链接继电器

链接继电器是指，将 MELSECNET/H 网络模块等的链接继电器 (LB) 刷新到 CPU 模块时，或者将 CPU 模块内的数据刷新到 MELSECNET/H 网络模块等的链接继电器 (LB) 时所使用的 CPU 模块侧的继电器。

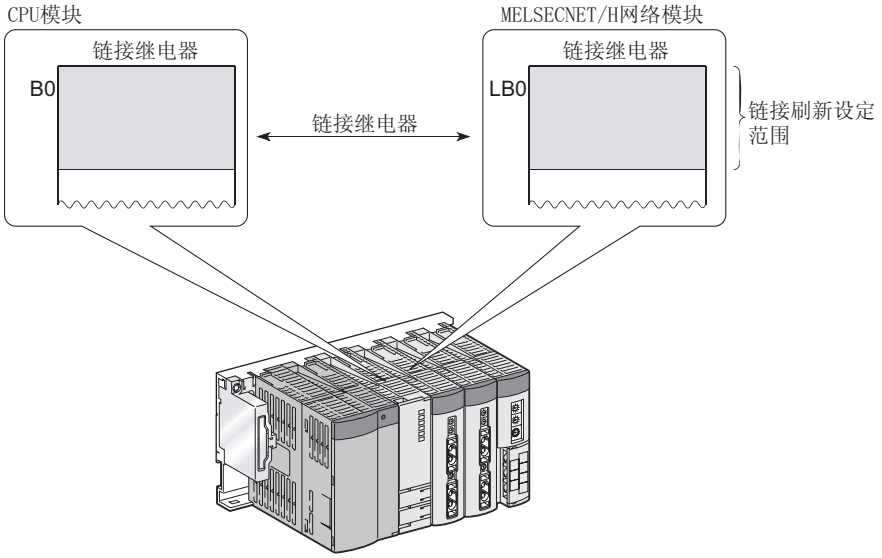


图 9.18 链接刷新

(2) a 触点与 b 触点的使用数

程序内的触点（a 触点、b 触点）的使用数，如果在程序容量的范围内则无限制。

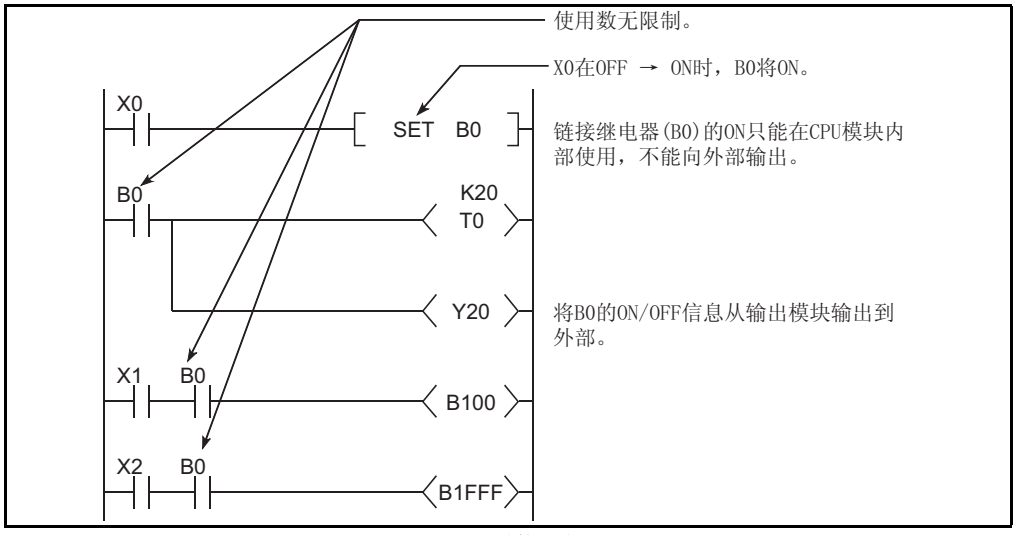


图 9.19 链接继电器

(3) 在网络系统中使用时

在网络系统中使用时，需要进行网络参数的设定。

在未设定网络参数的（不在 MELSECNET/H 网络系统等中使用）范围内，可作为内部继电器或锁存继电器使用。

- 使用链接继电器不进行锁存的范围……相当于内部继电器
- 使用链接继电器进行够锁存的范围……相当于锁存继电器

要 点

高性能 注 9.10

基本 注 9.11　过程 注 9.11　冗余 注 9.11

- MELSECNET/G 网络模块内 注 9.10、注 9.11 的链接继电器为 32768 点，但 CPU 模块的链接继电器的缺省值为 8192 点。
- MELSECNET/H 网络模块内的链接继电器为 16384 点，但 CPU 模块的链接继电器的缺省值分别为：基本模式 QCPU 为 2048 点，高性能模式 QCPU、过程 CPU 、冗余 CPU、通用型 QCPU 为 8192 点。

使用上述点数以后的链接继电器时，请通过可编程控制器参数的软元件设定变更链接继电器的点数。

备 注

关于网络参数，请参照以下手册。

☞ MELSECNET/G 网络系统参考手册（控制网络篇）

☞ Q 系列 MELSECNET/H 网络系统参考手册（可编程控制器网络篇）

☞ Q 系列 MELSECNET/H 网络系统参考手册（远程 I/O 网络篇）

高性能 注 9.10

在高性能模式 QCPU 中使用 MELSECNET/G 时，请确认 CPU 模块以及 GX Developer 的版本。
（☞ 附录 4.2）

基本 注 9.11　过程 注 9.11　冗余 注 9.11

在基本模式 QCPU、过程 CPU、冗余 CPU 中，不能使用 MELSECNET/G。

9.2.8 链接特殊继电器 (SB)

(1) 链接特殊继电器

链接特殊继电器是指表示 MELSECNET/H 网络模块等智能功能模块的通讯状态、异常检测的继电器。

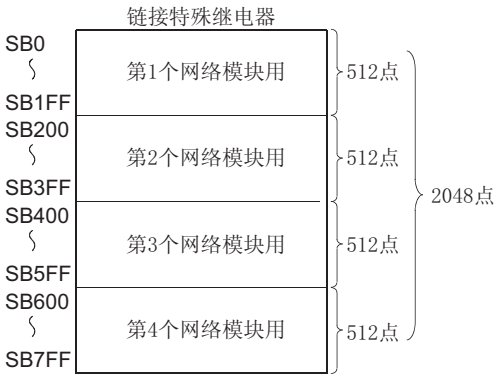
链接特殊继电器的 ON/OFF 是由数据链接时发生的各种原因进行控制的。

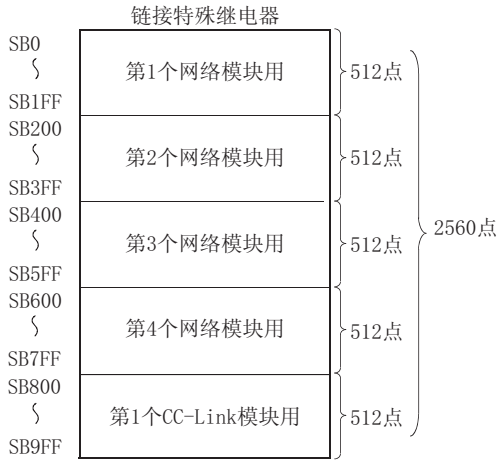
通过对链接特殊继电器进行监视，可把握数据链接的通讯状态、异常状态等。

(2) 链接特殊继电器点数

链接特殊继电器的点数如表 9.5 所示。

表 9.5 各种 CPU 模块的链接特殊继电器的点数

CPU 模块	链接特殊继电器点数
基本模式 QCPU	1024 点 (SB0~3FF) MELSECNET/H 网络模块等具备链接特殊继电器的智能功能模块为 512 点。
高性能模式 QCPU， 过程 CPU， 冗余 CPU	2048 点 (SB0~7FF)。 MELSECNET/H 网络模块等具备链接特殊继电器的智能功能模块为 512 点。 链接特殊继电器可按照下图进行分配。 

CPU 模块	链接特殊继电器点数
通用型 QCPU	2048 点 (SB0~7FF)。 但是，可以在可编程控制器参数的软元件设定中变更。 (☞ 9.1 节 (3)) MELSECNET/H 网络模块等具有链接特殊继电器的智能功能模块为 512 点。 通过将链接特殊继电器按照下图进行分配，可以将 CC-Link 的链接特殊继电器 (SB) 也刷新到 CPU 模块的链接特殊继电器 (SB) 中。 

备 注

链接特殊继电器的详细内容，请参照配备了链接特殊继电器的智能功能模块的手册。

9.2.9 步进继电器 (S)

步进继电器是 SFC 程序用的软元件。
关于步进继电器的使用方法，请参照以下手册。
☞QCPU(Q 模式)/QnACPU 编程手册 (SFC 篇)

要 点

步进继电器是 SFC 程序专用的软元件，因此不能在顺控程序中代替内部继电器使用。
将步进继电器代替内部继电器在顺控程序中使用时，可能导致 SFC 错误、系统宕机。

9.2.10 定时器 (T)

(1) 定时器

定时器 (T) 是指，定时器的线圈 ON 时开始计测，当前值超过设定值时即时间到，触点将 ON 的软元件。

定时器为加法。

定时器时间到时，当前值将变为与设定值相同的值。

(2) 定时器的种类

定时器有两种，一是线圈 OFF 时当前值变为 0 的定时器，一种是线圈 OFF 时保持当前值的累计定时器。

定时器有低速定时器和高速定时器，累计定时器有低速累计定时器和高速累计定时器。

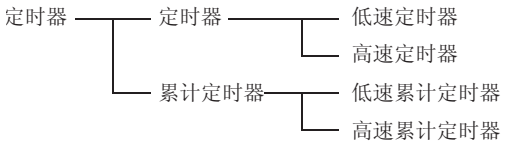


图 9.20 定时器的种类

(3) 定时器的使用方法

低速定时器与高速定时器为同一软元件，通过定时器的指定（指令的写法）可成为低速定时器或高速定时器。例如，指定为 OUT T0 将成为低速定时器，而指定为 OUTH T0 则成为高速定时器。

低速累计定时器与高速累计定时器为同一软元件，通过定时器的指定（指令的写法）可成为低速累计定时器或高速累计定时器。例如，指定为 OUT T0 将成为低速累计定时器，而指定为 OUTH ST0 则成为高速累计定时器。

(4) 低速定时器

(a) 低速定时器

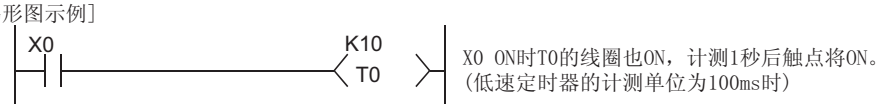
低速定时器是指计测单位为 1~1000ms 的定时器。

定时器只在线圈 ON 时有效。

定时器的线圈 ON 时即开始计测，时间到时触点即 ON。

定时器的线圈 OFF 时当前值变为 0，触点也将 OFF。

[梯形图示例]



[时序图]

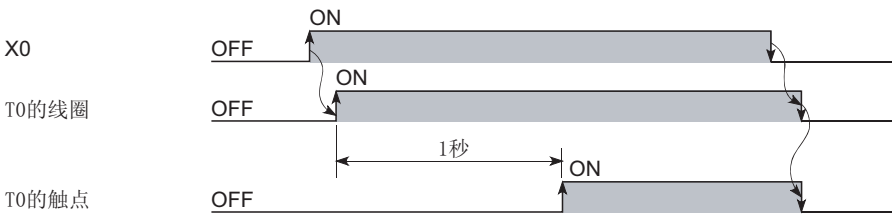


图 9.21 低速定时器的梯形图示例与时序图

(b) 计测单位

低速定时器的计测单位（时限）的缺省值为 100ms。

计测单位为 1~1000ms，以 1ms 为单位能够进行变更。

其设定在可编程控制器参数的可编程控制器系统设定中进行。

(5) 高速定时器

(a) 高速定时器

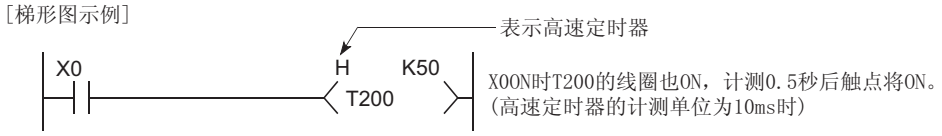
高速定时器是比低速定时器计量精度高的定时器。各 CPU 模块中的计量单位如表 9.6 所示。

定时器只在线圈 ON 时有效，符号为 “H”。

定时器的线圈 ON 时即开始计测，时间到后触点即 ON。

定时器的线圈 OFF 时当前值变为 0，触点也将 OFF。

[梯形图示例]



[时序图]

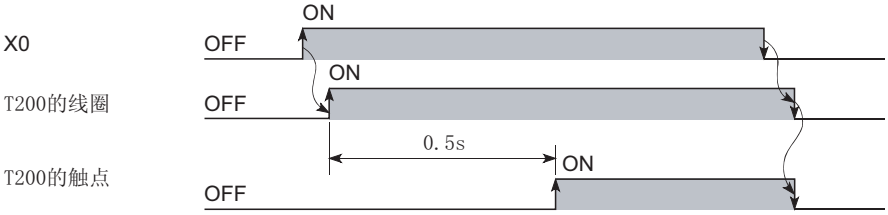


图 9.22 高速定时器的梯形图示例与时序图

(b) 计量单位

设定是在可编程控制器参数的可编程控制器系统设定中进行。

表 9.6 各 CPU 模块中的计量单位

CPU 模块	基本模式 QCPU、高性能模式 QCPU、过程 CPU、冗余 CPU	通用型 QCPU
计量单位	0.1 ～ 100ms(单位 0.1ms)	0.01 ～ 100ms(单位 0.01ms)

(6) 累计定时器

(a) 累计定时器

累计定时器是指在线圈 ON 时计测时间的定时器。
累计定时器的线圈 ON 时即开始计测时间，时间到时触点将 ON。
定时器的线圈 OFF 后当前值、触点的 ON/OFF 状态将保持。
线圈重新 ON 时，从所保持的当前值开始重新计测时间。

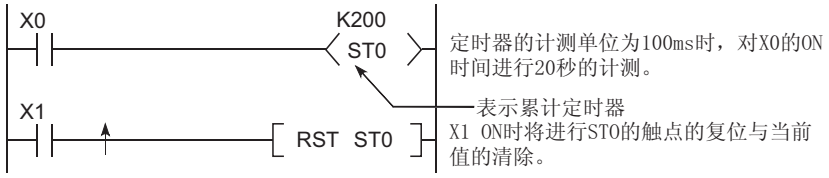
(b) 累计定时器的种类

累计定时器有低速累计定时器和高速累计定时器 2 种。

(c) 累计定时器的清除

当前值的清除和触点的 OFF 可通过 RST ST□指令进行。

[梯形图示例]



[时序图]

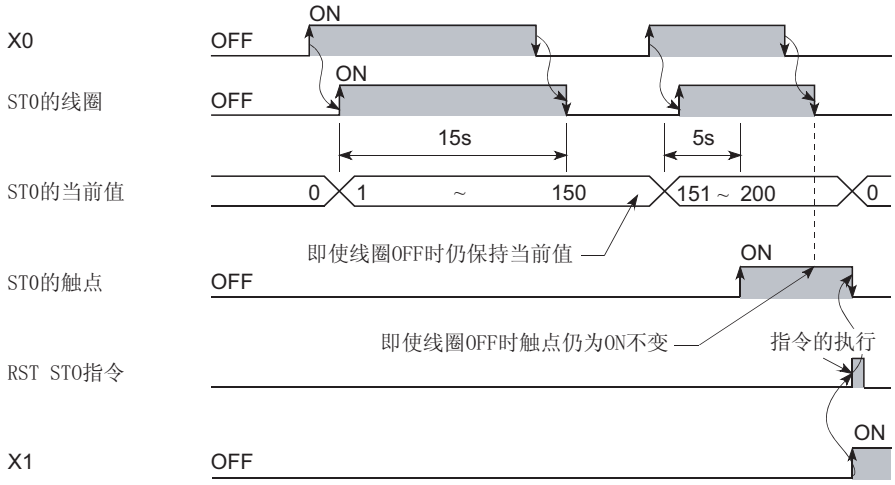


图 9.23 累计定时器的梯形图示例与时序图

(d) 计测单位

累计定时器的计测单位（时限）与低速定时器、高速定时器相同。

- 低速累计定时器：低速定时器
- 高速累计定时器：高速定时器

要 点

使用累计定时器时，需要通过可编程控制器参数的软元件设定来设定累计定时器的使用点数。

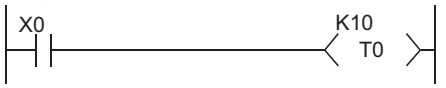
(7) 定时器的处理方法与精度

(a) 处理方法

执行 OUT T□指令时进行定时器线圈的 ON/OFF、当前值的更新以及触点的 ON/OFF 处理。

通过 END 处理不能进行定时器当前值的更新和触点的 ON/OFF 处理。

[梯形图示例]



[执行OUT T0指令时的处理内容]

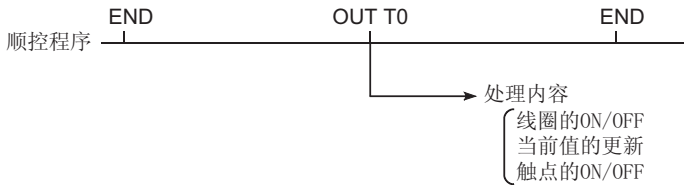


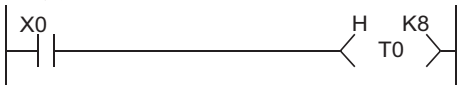
图 9.24 执行 OUT T0 指令时的处理内容

(b) 精度

当前值为将 END 指令的计数与执行 OUT T□指令时的值相加。

执行 OUT T□指令时定时器的线圈 OFF 的情况下，不进行当前值的更新。

[梯形图示例]



[当前值的更新时机]

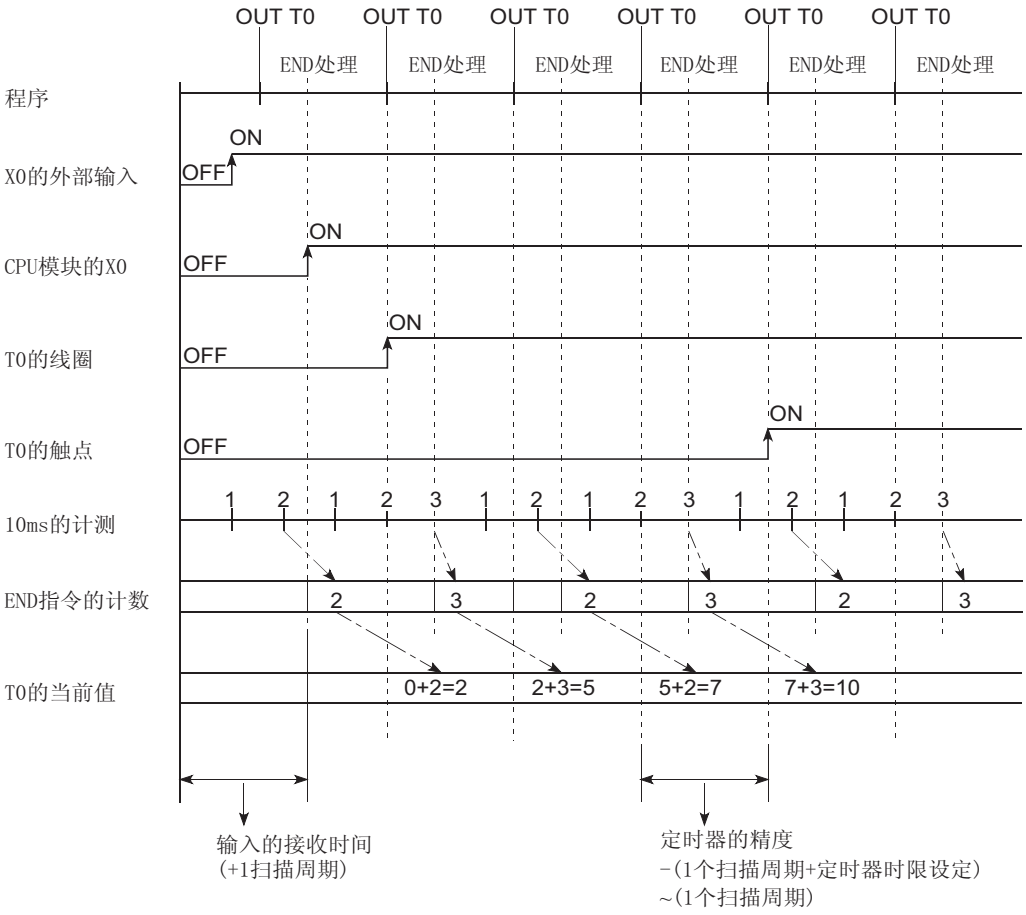


图 9.25 定时器的精度 (10ms 时)

接收输入 (X) 到输出为止的定时器的响应精度最大为 “2 个扫描周期＋定时器的时限设定”。

(8) 使用定时器时的注意事项

使用定时器时的注意事项如下所示。

(a) 关于同一定时器的使用

不能在 1 个扫描周期中使用多个 OUT T□记述同一定时器。

在 1 个扫描周期中使用多个 OUT T□记述同一定时器时，执行各 OUT T□指令时需要更新定时器的当前值，因此不能正常进行计测。

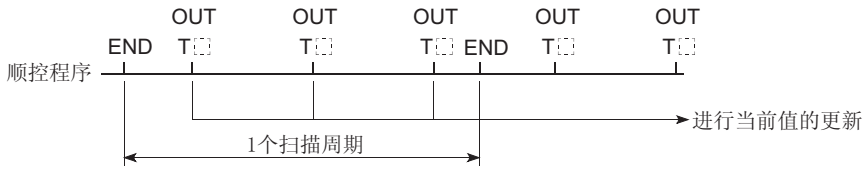


图 9.26 使用同一定时器时

(b) 当定时器未被在每个扫描中执行时

当定时器（例 :T1）的线圈处于 ON 时，不能使用 CJ 指令等跳转 OUT T1 指令。

OUT T□指令被跳转时，不能更新定时器的当前值。

此外，在副程序内存在有定时器的情况下，定时器（例 :T1）的线圈处于 ON 时，必须使包含有 OUT T1 指令的副程序每个扫描仅执行 1 次。

如果副程序未在每个扫描中执行，则定时器的当前值将不能被更新。

基本 注 9.12

(c) 关于不能使用定时器的程序

在中断程序与恒定周期执行型程序*注 9.12* 中不能使用定时器。

(d) 设定值为 0 时

执行 OUT T□指令时触点将 ON。

(e) 时间到后变更了设定值时

定时器时间到后，将设定值变更为比当前值大的值时。定时器也保持时间到状态而不动作。

基本 注 9.13　冗余 注 9.13　通用 注 9.13

(f) 使用低速执行型程序时*注 9.13*

在低速执行型程序中使用了定时器的情况下，在执行 OUT T□指令时当前值将被加上低速扫描时间（☞ 3.3.3 项）。

基本 注 9.12

在基本模式 QCPU 中，不能使用恒定周期执行类型程序。

基本 注 9.13　冗余 注 9.13　通用 注 9.13

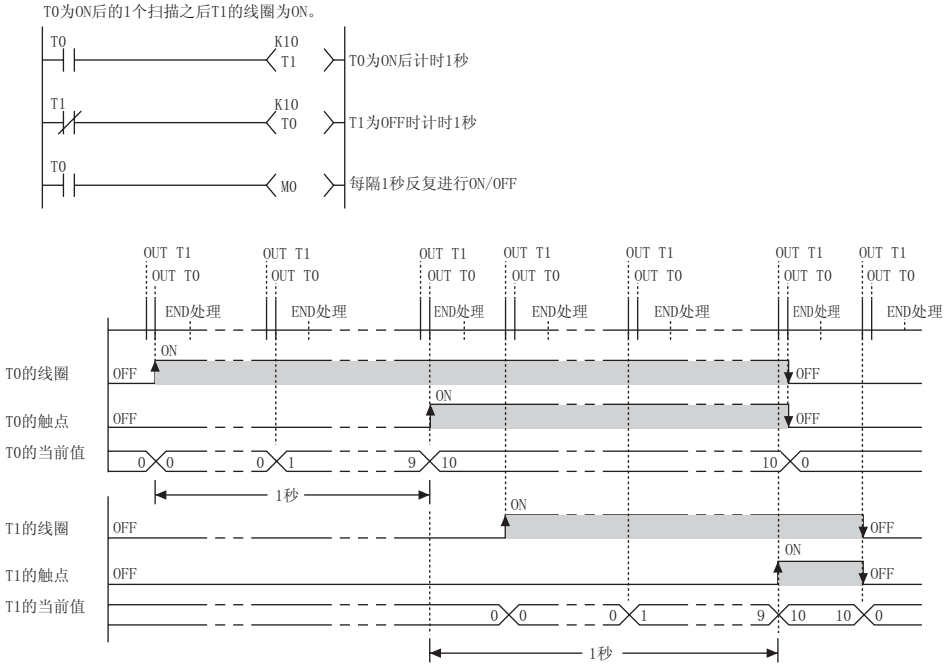
在基本模式 QCPU、冗余 CPU、通用型 QCPU 中，不能使用低速执行类型程序。

(g) 使用多个定时器时

由于执行各 OUT T□指令时进行定时器的当前值更新，因此使用多个定时器时，请注意梯形图的顺序。

例如，使用 2 个定时器创建 ON/OFF 梯形图时，请按图 9.27 中所示创建梯形图。

[正确的梯形图示例]



[错误的梯形图示例]

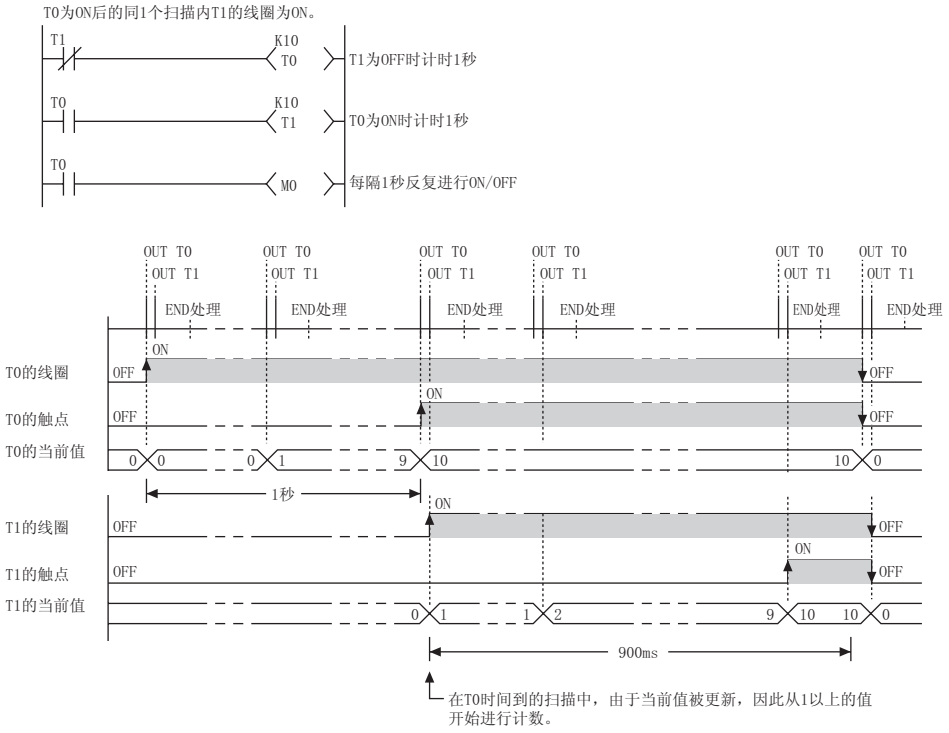


图 9.27 使用了 2 个定时器的 ON/OFF 梯形图

9.2.11 计数器 (C)

(1) 计数器

计数器是指在顺控程序中对输入条件的上升次数进行计数的软元件。
计数值与设定值相同时，计数到，触点 ON。
计数器为加法。

(2) 计数器的种类

计数器分为 2 种，一种是在顺控程序中对输入条件的上升次数进行计数的计数器，一种是对中断因子的发生次数进行计数的计数器。

(3) 计数处理

(a) 执行 OUT C□指令时

执行 OUT C□指令时进行计数器线圈的 ON/OFF 处理、当前值的更新（计数值＋ 1）以及触点的 ON/OFF 处理。
通过 END 处理更新计数器的当前值后，不进行触点的 ON/OFF 处理。

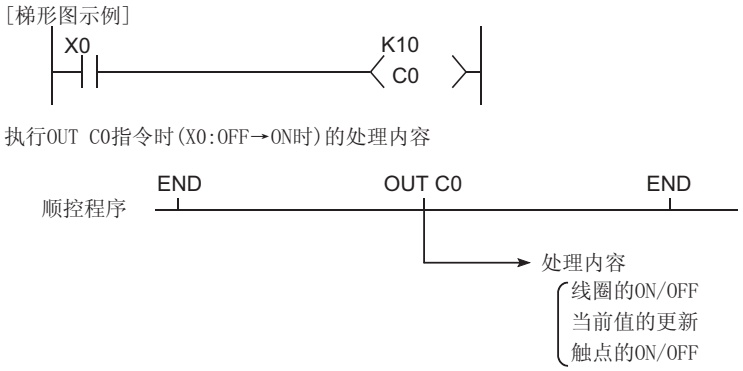


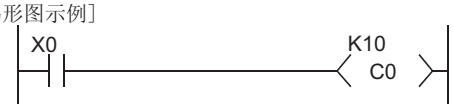
图 9.28 OUT C□指令的执行与处理内容

(b) 当前值的更新（计数值 +1）

当前值的更新（计数值 +1）在 OUT C□指令上升 (OFF → ON) 时进行。

OUT C□指令在 OFF、ON → ON 以及 ON → OFF 时，不更新当前值。

[梯形图示例]



[当前值的更新时间]

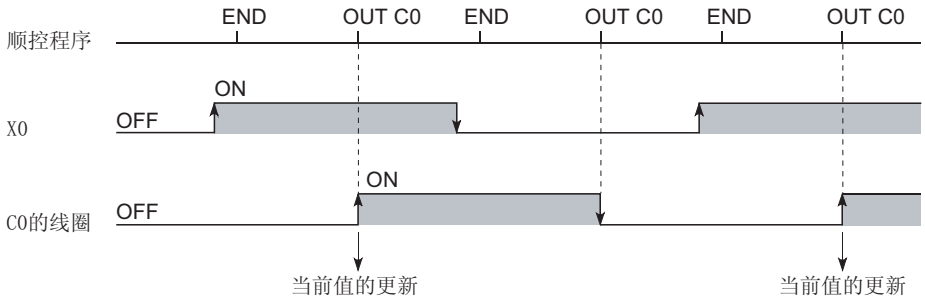


图 9.29 当前值的更新时间

要 点

在 1 个扫描周期中记述多个计数器，可最大可能地提高计数速度。

此时计数器的输入信号使用直接存取输入 (DX□)（☞ 3.8.2 项）。

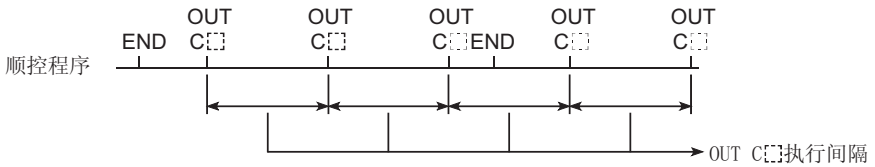


图 9.30 最大可能地提高计数速度的梯形图示例

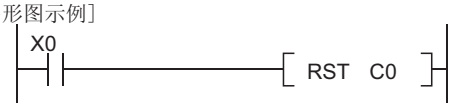
(c) 计数器的复位

计数器的当前值在 OUT C□指令 OFF 时也不会被清除。

计数器的当前值的清除（复位）与触点的 OFF 通过 RST C□指令进行。

执行 RST C□指令时计数器值被清除，触点也将 OFF。

[梯形图示例]



[计数器的复位时机]

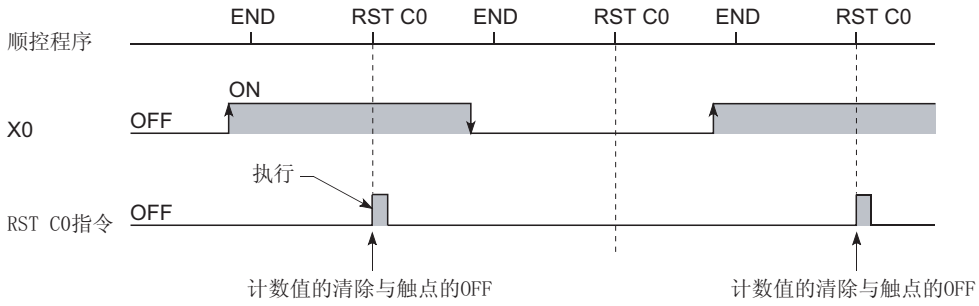


图 9.31 计数器的复位

1) 计数器复位时的注意事项

执行 RST C□指令，则 C□的线圈也将 OFF。

执行 RST C□指令后 OUT C□指令的执行条件仍为 ON 时，执行 OUT C□指令时 C□的线圈将 ON，进行当前值（计数值 +1）的更新。

[梯形图示例]

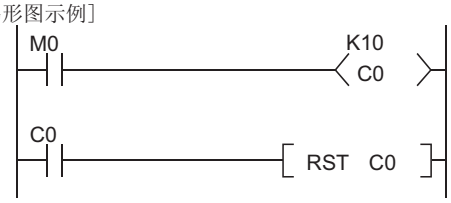


图 9.32 计数器复位的梯形图示例

上述梯形图中，M0 为 OFF → ON，C0 的线圈 ON，当前值被更新。

C0 计数到时，C0 的触点将 ON，通过执行 RST C0 指令 C0 的当前值将被清除。此时，C0 的线圈也将 OFF。

下一个扫描周期中 M0 为 ON 时，执行 OUT C0 指令时 C0 的线圈为 OFF → ON，进行当前值的更新。（当前值变为 1。）

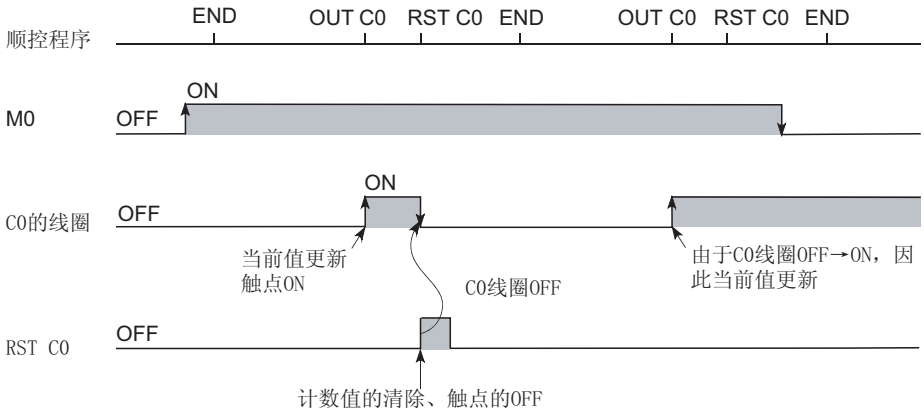


图 9.33 当前值更新的时间

作为上述应对，在 RST C0 指令的执行条件中插入 OUT C0 指令的执行条件的 b 触点，OUT C0 指令的执行条件 (M0) 为 ON 时，建议确保 C0 的线圈不会 OFF。

[变更梯形图示例]

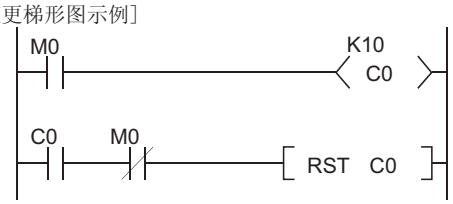


图 9.34 计数器复位的梯形图示例（建议示例）

(d) 计数器的最大计数速度

计数器只能在输入条件的 ON/OFF 时间大于同一 OUT C□指令的执行间隔时进行计数。

计数器的最大计数速度的计算公式如下。

$$\text{最大计数速度Cmax} = \frac{n}{100} \times \frac{1}{T} [\text{次/s}]$$

n：任务 (%)*1

T：OUT C□指令的执行间隔 (sec)

*1 ：任务 (n) 是将计数输入信号的 ON • OFF 时间的比用百分比 (%) 表示的值。

- T1 ≧ T2时，$n = \frac{T2}{T1+T2} \times 100\%$
- T1 ＜ T2时，$n = \frac{T1}{T1+T2} \times 100\%$

图 9.35 任务比

通用 UD 注 9.14

(4) 中断计数器*注 9.14*

(a) 中断计数器

中断计数器是指对中断因子的发生次数进行计数的软元件。

(b) 计数处理

1) 发生中断时

中断计数器在发生中断时更新计数器的当前值。

用中断计数器进行计数时，没有必要编写使用了中断计数器的程序。

2) 关于中断计数器的计数

中断计数器即使指定了设定值，也不会计数到。

将中断计数器用于控制时，请使用比较指令 (=，＜＝等）与设定值进行比较后使内部继电器 (M) 等 ON/OFF。

I0 的中断输入 10 次 ON 时，使 M0 也 ON 的情况下，编写图 9.36 所示的程序。（假设 I0 对应的中断计数器为 C300。）

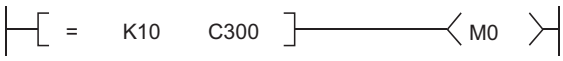


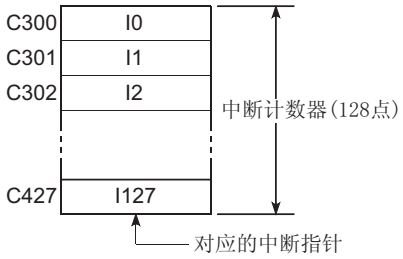
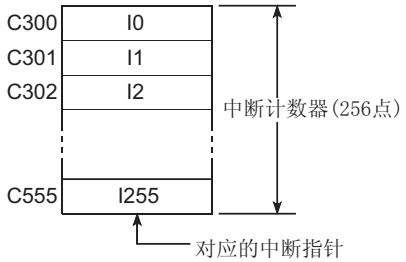
图 9.36 将中断计数器用于控制时的梯形图示例

(c) 中断计数器的设定

使用中断计数器时，通过可编程控制器参数的可编程控制器系统设定来设定中断计数器起始 No.。

从所设定的计数器编号到表 9.7 所示的点数成变为中断计数器。

表 9.7 各 CPU 模块的中断计数器的点数

基本模式 QCPU	高性能模式 QCPU，过程 CPU，冗余 CPU
从设定的计数器编号开始 128 点成为中断计数器。 将中断计数器起始 No. 设定为 C300 时，C300~427 成为中断计数器。 	从设定的计数器编号开始 256 点成为中断计数器。 将中断计数器起始 No. 设定为 C300 时，C300~555 成为中断计数器。 

使用中断计数器时，在主程序中通过 EI 指令进入中断许可状态。

通用 UD 注 9.14

在通用型 QCPU 中，不能使用中断计数器。

(5) 注意事项

(a) 关于中断计数器与中断程序的执行
在 1 个中断指针下，中断计数器不能既进行中断计数又执行中断程序。
通过可编程控制器参数的可编程控制器系统设定进行中断计数器的设定后，不能执行中断程序。

(b) 计数处理等待的处理
执行以下各处理时发生中断的情况下，计数器处理将等待，直到各处理的执行完成。
各处理执行完成后即进行计数处理。
但进行各处理时再次发生同一中断时，只能进行 1 次计数。

基本
注 9.15

- 执行顺控程序的各指令时
- 执行中断程序时
- 执行恒定周期执行类型程序时*注 9.15*

(c) 关于中断计数器的最大计数速度
中断计数器的最大计数速度取决于以下所示的处理时间中的最长的处理。

基本
注 9.15

- 程序中使用的指令中，处理时间最长的指令
- 中断程序的处理时间
- 恒定周期执行类型程序的处理时间*注 9.15*

(d) 多点数使用中断计数器时
多点数使用中断计数器，顺控程序的处理时间将变长，导致 “WDT ERROR”。
此时，请减少中断计数器的点数，或降低输入脉冲信号的计数速度。

(e) 关于中断计数器的复位
应使用主程序的 RST C□指令进行中断计数器的计数值的复位。

(f) 关于计数值的读出
中断计数器的计数值，可通过顺控程序的 MOV 指令读出。

基本
注 9.15

在基本模式 QCPU 中，不能执行恒定周期执行类型程序。

9.2.12 数据寄存器 (D)

(1) 数据寄存器

数据寄存器是指可存储数值数据 (-32768~32767 或 0000H~FFFFH) 的存储器。

(2) 数据寄存器的位构成

(a) 位构成与读出以及写入单位

1 个数据寄存器由 16 位构成，能够以 16 位为单位读出和写入。

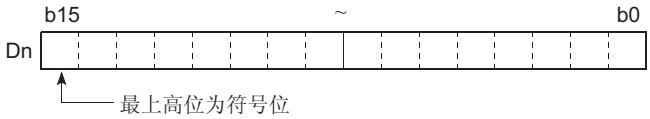


图 9.37 数据寄存器的位构成

要 点

数据寄存器是附带有符号来使用的。

HEX(16 进制) 的情况可存储 0000H~FFFFH，但由于最高为符号位，所以能够指定的数值范围为 -32768~32767。

(b) 通过 32 位指令使用数据寄存器

通过 32 位指令使用数据寄存器时，Dn 与 Dn+1 为处理对象。

通过顺控程序指定的数据寄存器编号 (Dn) 为低 16 位，通过顺控程序指定的数据寄存器编号 +1 的数据寄存器为高 16 位。

例如用 DMOV 指令指定了 D12 时，D12 为低 16 位，D13 为高 16 位。

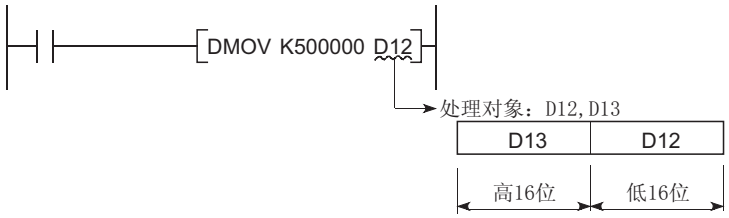


图 9.38 利用 32 位指令进行数据发送以及存储目标

用 2 个数据寄存器可存储 -2147483648~2147483647 或者 0H~FFFFFFFFH 的数据。(32 位构成时的最高为符号位。)

(3) 关于存储数据的保留

数据寄存器中存储的数据将保留到存储其它数据为止。

可编程控制器的电源 OFF 或者 CPU 模块复位时，数据寄存器中存储的数据将被初始化。

9.2.13 链接寄存器 (W)

(1) 链接寄存器

链接寄存器是指将 MELSECNET/H 网络模块等智能功能模块的链接寄存器 (LW) 的数据刷新到 CPU 模块时的 CPU 模块侧的存储器。

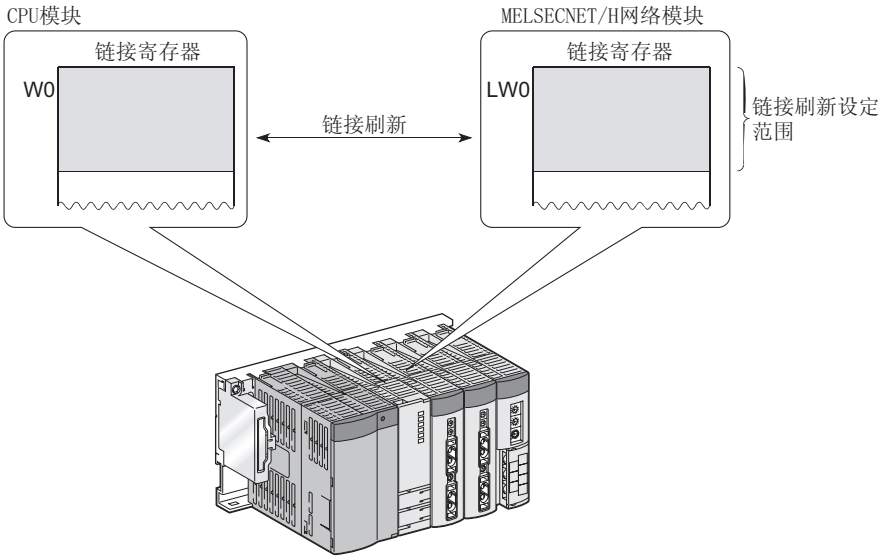


图 9.39 链接刷新

链接寄存器是指能够存储数值数据 (-32768~32767) 或者 0000H~FFFFH) 的存储器。

(2) 链接寄存器的位构成

(a) 位构成与读出以及写入单位

1 个链接寄存器由 16 位构成，能够以 16 位为单位读出和写入。

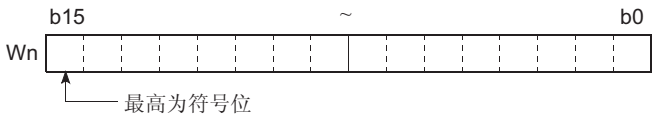


图 9.40 链接寄存器的位构成

要 点

1. 链接寄存器是附带有符号来使用的。
   HEX(16 进制) 的情况可存储 0000H~FFFFH，但由于最高位为符号位，所以能够指定的数值范围为 -32768~32767。
2. 在 MELSECNET/H 网络模块等中属不使用范围的链接寄存器可作为数据寄存器的代用品使用。

(b) 通过 32 位指令使用链接寄存器

通过 32 位指令使用链接寄存器时，以连续 2 个链接寄存器 (Wn 与 Wn+1) 为处理对象。

通过顺控程序指定的链接寄存器编号 (Wn) 为低 16 位，通过顺控程序指定的链接寄存器编号 +1 的链接寄存器为高 16 位。

例如用 DMOV 指令指定了 W12 时，W12 为低 16 位，W13 为高 16 位。

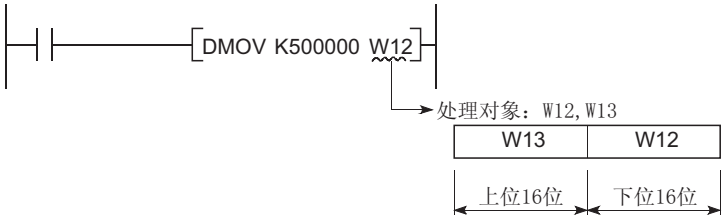


图 9.41 利用 32 位指令进行数据发送以及存储目标

2 个链接寄存器中，可存储 -2147483648~2147483647 或者 0H~FFFFFFFFH 的数据。(32 位构成时的最高为符号位。)

(3) 关于存储数据的保留

链接寄存器中存储的数据将保留到存储其它数据为止。

可编程控制器的电源 OFF 或者 CPU 模块复位时，链接寄存器中存储的数据将被初始化。

要 点

高性能 注 9.16

基本 注 9.17　过程 注 9.17　冗余 注 9.17

- MELSECNET/G 网络模块内 *注 9.16*、*注 9.17* 的链接寄存器为 131072 点，但 CPU 模块内的链接寄存器的缺省值为 8192 点。
- MELSECNET/H 网络模块内的链接寄存器为 16384 点，但 CPU 模块内的链接寄存器的缺省值分别为：基本模式 QCPU 为 2048 点，高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 为 8192 点。

使用上述点数以后的链接寄存器时，请通过可编程控制器参数的软元件设定变更链接寄存器的点数或者使用文件寄存器。

高性能 注 9.16

在高性能模式 QCPU 中使用 MELSECNET/G 时，请确认 CPU 模块以及 GX Developer 的版本。
(附录 4.2)

基本 注 9.17　过程 注 9.17　冗余 注 9.17

在基本模式 QCPU、过程 CPU、冗余 CPU 中，不能使用 MELSECNET/G。

(4) 在网络系统中使用时

在网络系统中使用时，需要进行网络参数的设定。

网络参数中未设定的链接寄存器可代替数据寄存器使用。

备 注

关于网络参数，请参照以下手册。

☞ MELSECNET/G 网络系统参考手册（控制网络篇）

☞ Q 系列 MELSECNET/H 网络系统参考手册（可编程控制器网络篇）

☞ Q 系列 MELSECNET/H 网络系统参考手册（远程 I/O 网络篇）

9.2.14 链接特殊寄存器 (SW)

(1) 链接特殊寄存器

链接特殊寄存器是指存储 MELSECNET/H 网络模块等智能功能模块的通讯状态、异常内容的寄存器。

由于数据链接时的信息是以数值的形式存储的，所以通过对链接特殊寄存器进行监视，便可调查异常处及其原因等。

(2) 链接特殊寄存器点数

链接特殊寄存器的点数如表 9.8 所示。

表 9.8 各种 CPU 模块的链接特殊寄存器的点数

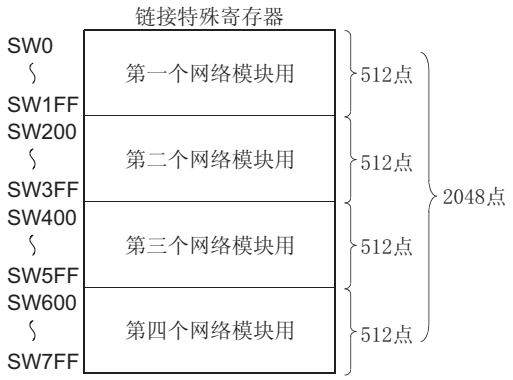
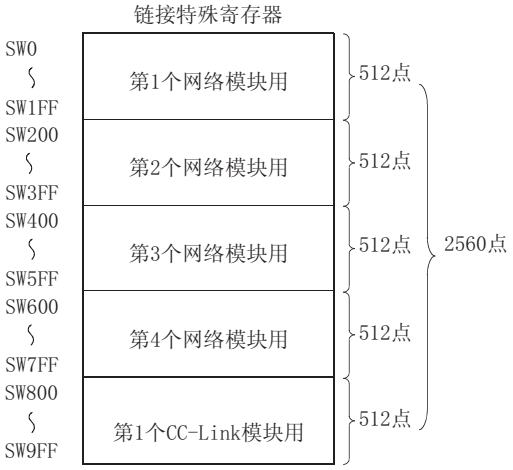
CPU 模块	链接特殊寄存器点数
基本模式 QCPU	1024 点 (SW0~3FF) MELSECNET/H 网络模块等智能功能模块为 512 点。
高性能模式 QCPU， 过程 CPU， 冗余 CPU	2048 点 (SW0~7FF)。 MELSECNET/H 网络模块等智能功能模块为 512 点。 链接特殊寄存器可按照下图进行分配。 

表 9.8 各种 CPU 模块的链接特殊寄存器的点数（续）

CPU 模块	链接特殊寄存器点数
通用型 QCPU	2048 点（SB0~7FF）。 但是，可以在可编程控制器参数的软元件设定中变更。 （ 9.1 节 (3)） MELSECNET/H 网络模块等具有链接特殊寄存器的智能功能模块为 512 点。 通过将链接特殊寄存器按照下图进行分配，可以将 CC-Link 的链接特殊寄存器（SW）也刷新到 CPU 模块的链接特殊寄存器（SW）中。 

备 注

关于能够使用的链接特殊寄存器的详细内容，请参照配备了链接特殊寄存器的智能功能模块的手册。

9.3 内部系统软元件

内部系统软元件是指系统用的软元件。
内部系统软元件的分配 / 容量是固定的，用户不能自行变更。

9.3.1 功能软元件 (FX，FY，FD)

(1) 功能软元件

功能软元件是指在带变量的副程序中使用的软元件。
功能软元件在带变量的副函数调用源与带变量的副程序之间进行数据的写入 / 读出。

示例

在副程序中使用FX0、FY1、FD2时，通过副程序调用指令指定X0、M0、D0的情况下，X0的ON/OFF数据将被传递到FX0，FY1的ON/OFF数据将被传递到M0，D0的数据将被传递到FD2。

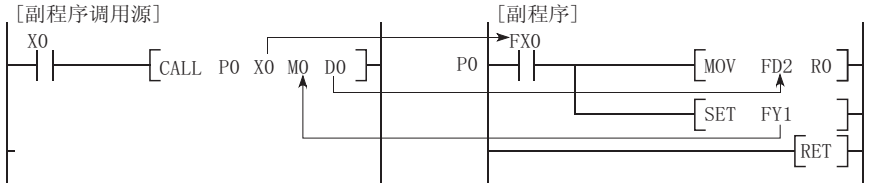


图 9.42 功能软元件的使用示例

(2) 功能软元件的用途

在副程序中使用功能软元件，便可决定在各副程序调用源中使用的软元件，因此即使使用同一副程序也无需在意其它副程序的调用源。

(3) 功能软元件的种类

功能软元件有 3 种：功能输入 (FX)，功能输出 (FY) 和功能寄存器 (FD)。

(a) 功能输入 (FX)

- 功能输入用于将 ON/OFF 数据传递至副程序。
- 副程序中，接收通过带变量的副函数调用指令指定的位数据，并用于运算。
- CPU 模块的位数据指定软元件全部可使用。

(b) 功能输出 (FY)

- 功能输出用于将副程序中的运算结果 (ON/OFF 数据) 传递到副函数调用源。
- 运算结果被存储在带变量的副程序中指定的软元件中。
- 可使用除 CPU 模块的输入 (X，DX) 以外的位数据指定软元件。

(c) 功能寄存器 (FD)

- 功能寄存器用于在副函数调用源与副程序之间写入 / 读出数据。
- 功能寄存器的输入 / 输出条件由 CPU 模块自动进行判断。
  如果是副程序中的源数据，将变为副程序的输入数据。
  如果是副程序中的目标数据，将变为来自副程序的输出数据。
- 1 个功能寄存器最多占有 4 个字。
  但使用的字数因副程序中的指令而异。

如果是 1 字指令，就仅使用 1 个字。

图 9.43 1 个功能寄存器占有 1 个字的情况

如果是 2 字指令，则使用 2 个字。

图 9.44 1 个功能寄存器占有 2 个字的情况

如果是 32 位的乘除运算的目标数据，则占有 4 个字。

图 9.45 1 个功能寄存器占有 4 字的情况

可以使用 CPU 模块的字数据指定软元件。

要 点

在带变量的副程序中，不能使用功能寄存器中使用的软元件。
如果在带变量的副程序中使用功能寄存器中使用的软元件，则会导致功能寄存器的值不能正常地传递至调用源中。

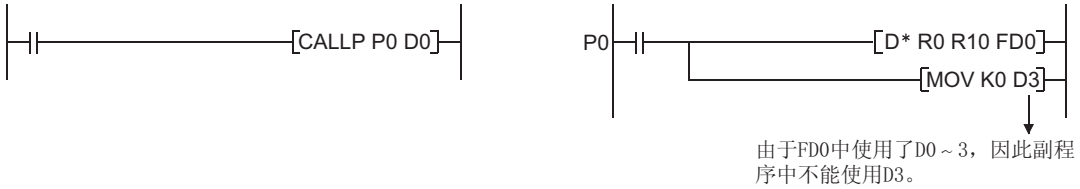


图 9.46 带变量的副程序中不能使用软元件的梯形图示例

备 注

关于功能软元件的使用方法，请参照以下手册。
☞ QCPU(Q 模式)/QnACPU 编程手册（公共指令篇）

9.3.2 特殊继电器 (SM)

(1) 特殊继电器

特殊继电器是指存储了 CPU 模块状态的继电器。

(2) 特殊继电器的分类

特殊继电器按照用途进行如表 9.9 所示的分类。

表 9.9 特殊继电器的分类一览

分类	特殊继电器	CPU 模块				
		基本模式 QCPU	高性能模式 QCPU	过程 CPU	冗余 CPU	通用型 QCPU
诊断信息	SM0 ～ 99*1	○	○	○	○	○
	SM0 ～ 199*2					
串行口通讯功能	SM100 ～ 129*1	○	×	×	×	×
系统信息	SM200 ～ 399	○	○	○	○	○
系统时钟 / 系统计数器	SM400 ～ 499	○	○	○	○	○
扫描信息	SM500 ～ 599	○	○	○	○	○
存储卡	SM600 ～ 699	○	○	○	○	○
指令相关	SM700 ～ 799	○	○	○	○	○
调试	SM800 ～ 899	×	○	○	○	○
锁存区域	SM900 ～ 999	×	○	○	○	○
A → Q/QnA 转换对应	SM1000 ～ 1299*3	×	○	○	×	×
过程控制指令	SM1500 ～ 1509	×	×	○	○	×
冗余系统（自系统 CPU 信息）	SM1510 ～ 1599	×	×	×	○	×
冗余系统（其它系统 CPU 信息）	SM1600 ～ 1699	×	×	×	○	×
冗余系统（热备信息）	SM1700 ～ 1779	×	×	×	○	×
冗余电源模块信息	SM1780 ～ 1799	×	○	○	○	○

○: 存在能够使用的特殊继电器
×: 不存在能够使用的特殊继电器

*1: 在基本模式 QCPU 中，SM0~99 将变为故障检测区域，SM100~129 将变为串行口通讯功能区域。
*2: 在高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 中，SM0~199 将变为故障诊断用的区域。
*3: 仅在通过可编程控制器参数的可编程控制器系统设定设为 “使用 SM1000，SD1000 以后的特殊继电器 / 特殊寄存器”时有效。

备 注

关于特殊继电器的详细内容，请参照附录 1。

9.3.3 特殊寄存器 (SD)

(1) 特殊寄存器

特殊寄存器是指存储了 CPU 模块状态（故障检测・系统信息等）的寄存器。

(2) 特殊寄存器的分类

特殊寄存器按照用途进行如表 9.10 所示的分类。

表 9.10 特殊寄存器的分类一览

分类	特殊继电器	CPU 模块				
		基本模式 QCPU	高性能模式 QCPU	过程 CPU	冗余 CPU	通用型 QCPU
诊断信息	SD0 ～ 99*1 SD0 ～ 199*2	○	○	○	○	○
串行口通讯功能	SD100 ～ 129*1	○	×	×	×	×
保险丝熔断模块	SD130 ～ 149*1	○	×	×	×	×
I/O 模块校验	SD150 ～ 199*1	○	×	×	×	×
系统信息	SD200 ～ 399	○	○	○	○	○
系统时钟 / 系统计数器	SD400 ～ 499	○	○	○	○	○
扫描信息	SD500 ～ 599	○	○	○	○	○
存储卡	SD600 ～ 699	○	○	○	○	○
指令相关	SD700 ～ 799	○	○	○	○	○
调试	SD800 ～ 899	×	○	×	×	○
锁存区域	SD900 ～ 929	×	○	×	×	○
冗余功能（停电保持信息）	SD952	×	×	×	○	×
A→ Q/QnA 转换对应	SD1000 ～ 1299	×	○	○	×	×
保险丝熔断模块	SD1300 ～ 1399*2	×	○	○	○	○
I/O 模块校验	SD1400 ～ 1499*2	×	○	○	○	○
过程控制指令	SD1500 ～ 1509	×	×	○	○	×
冗余系统（自系统 CPU 信息）	SD1510 ～ 1599	×	×	×	○	×
冗余系统（其它系统 CPU 信息）	SD1600 ～ 1699	×	×	×	○	×
冗余系统（热备信息）	SD1700 ～ 1779	×	×	×	○	×
冗余电源模块信息	SD1780 ～ 1799	×	○	○	○	○

○：存在能够使用的特殊寄存器

×：不存在能够使用的特殊寄存器

*1：在基本模式 QCPU 中，SD0~99 将变为故障检测区域。
串行口通讯功能、保险丝断裂模块和 I/O 模块核对的数据被存储在 SD100~199 中。

*2：在高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 中，SD0~199 将变为故障诊断用的区域。

*3：仅在通过可编程控制器参数的可编程控制器系统设定设为“使用 SM1000，SD1000 以后的特殊继电器 / 特殊寄存器”时有效。

备 注

关于特殊寄存器的详细内容，请参照附录 2。

9.4 链接直接软元件（J □ \ □）

(1) 链接直接软元件

高性能 注 9.18

链接直接软元件是可直接存取 MELSECNET/G 注 9.18、注 9.19 或者 MELSECNET/H 网络模块内的链接软元件的软元件。

基本 注 9.19　过程 注 9.19　冗余 注 9.19

CPU 模块与 MELSECNET/G 注 9.18、注 9.19 或者 MELSECNET/H 网络系统的网络模块之间的刷新（数据传送）是通过顺控程序的 END 处理进行的。

高性能 注 9.18

在高性能模式 QCPU 中使用 MELSECNET/G 时，请确认 CPU 模块以及 GX Developer 的版本。
（ 附录 4.2）

基本 注 9.19　过程 注 9.19　冗余 注 9.19

在基本模式 QCPU、过程 CPU、冗余 CPU 中，不能使用 MELSECNET/G。

(2) 链接直接软元件的指定方法与使用示例

(a) 指定方法

链接直接软元件的指定是通过网络 No. 与软元件编号进行的。

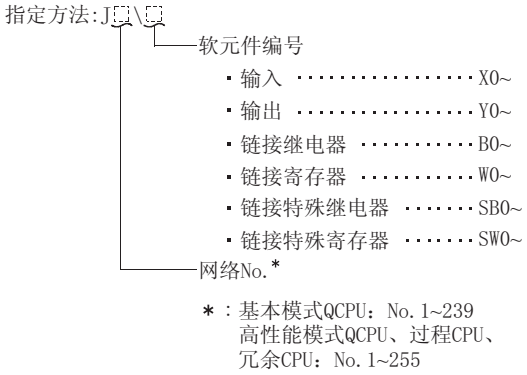


图 9.47 链接直接软元件的指定方法

(b) 使用示例

如果是网络 No.2 的链接寄存器 10(W0)，将变为 “J2\W10”。

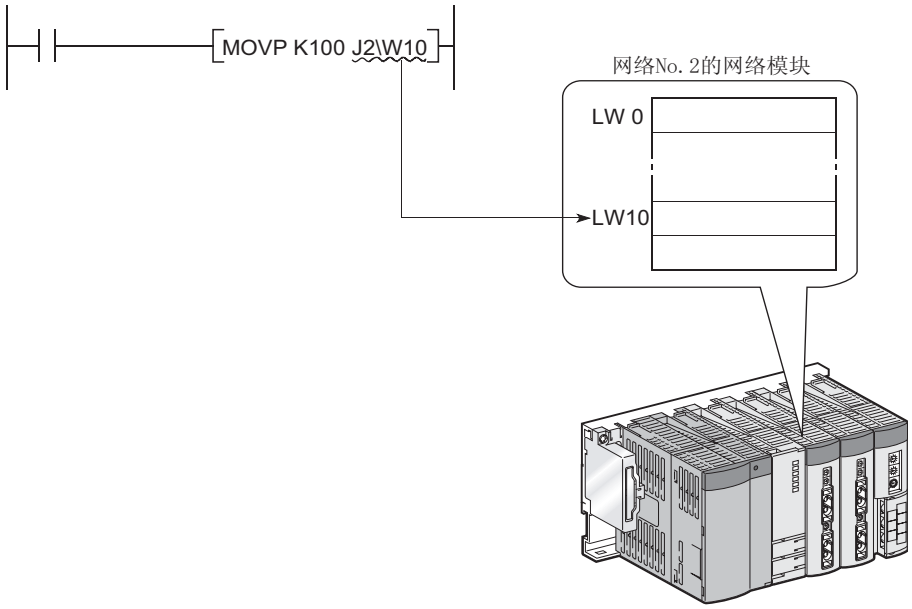


图 9.48 链接直接软元件的使用示例

• 如果是位软元件 (X，Y，B，SB)，进行位指定。
指定示例：J1\K1X0，J10\K4B0

(3) 指定范围

对于链接直接软元件，可以指定网络模块的链接软元件。
不在网络刷新参数设定范围内的链接软元件也可指定。

(a) 写入时

• 请在通过网络参数的公共参数设为发送范围的链接软元件范围内，以及未设为网络刷新参数刷新范围的范围内进行写入。

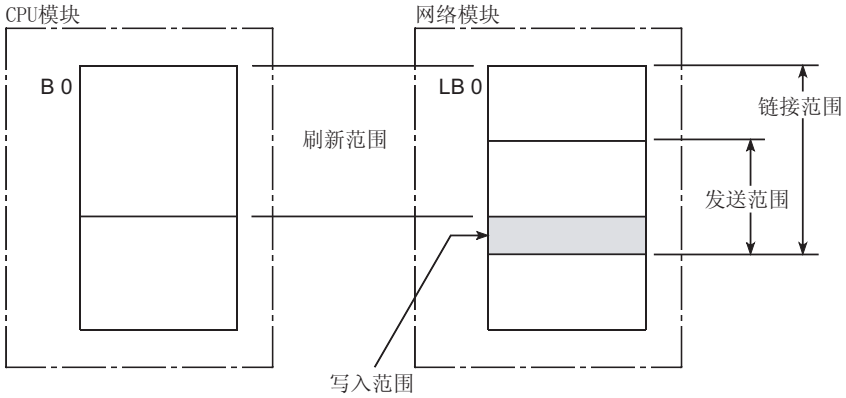


图 9.49 链接直接软元件的写入范围

• 通过刷新参数设为刷新范围的链接软元件范围内也可进行写入，但刷新时链接模块的链接软元件将被改写。
利用链接直接软元件进行写入时，通过刷新参数设定的 CPU 模块侧的相应软元件中，也请写入同一数据。

[刷新参数的设定]
• 网络模块号：1
• CPU 模块 (W0~3F)↔网络模块 (LW0~3F)

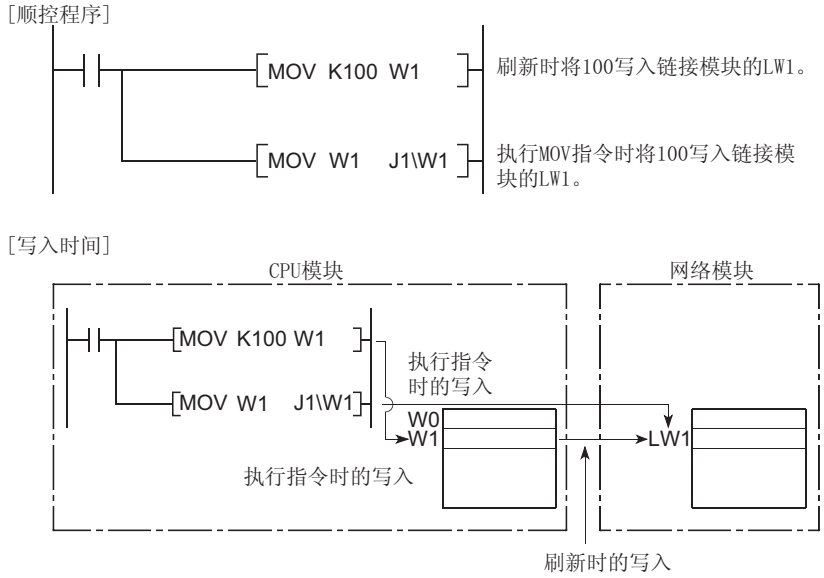


图 9.50 设定为刷新范围的链接软元件的写入

• 在其它站点的写入范围内利用链接软元件进行写入情况下，接收其它站点发送的数据时，接收的数据将被改写。

(b) 读出时

通过链接直接软元件读出时，在网络模块的链接软元件的范围内均可进行。

要 点

基本

注 9.20

能够利用链接直接软元件进行写入 / 读出的网络模块，1 个网络 No. 只有 1 台。
同一网络 No. 安装 2 台以上的网络模块时，起始 I/O 地址号大的网络模块将成为链接直接软元件的写入 / 读出对象。 *注 9.20*
例如，在网络 No. 1 中如图 9.51 所示安装了站号 1 和站号 2 两个网络模块时，站号 2 网络模块将成为链接直接软元件的对象。

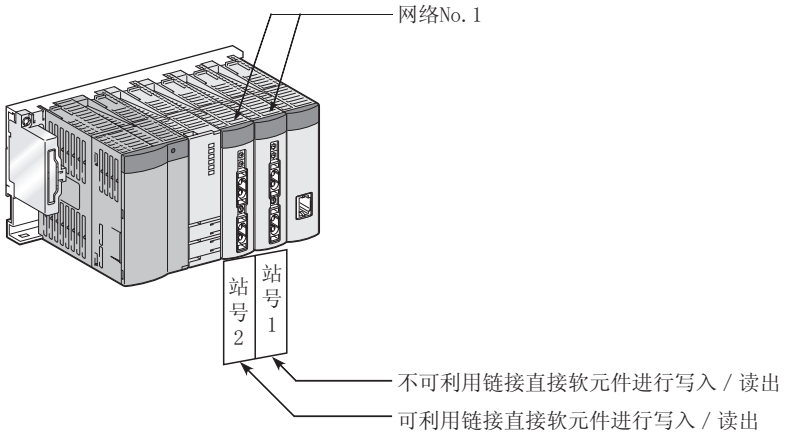


图 9.51 同一网络 No. 安装 2 台以上的网络模块的情况

基本

注 9.20

在基本模式 QCPU 中，不能在同一网络 No. 上安装 2 台以上网络模块。
在基本模式 QCPU 中，要点的内容不适用。

(4) 链接直接软元件与链接刷新的不同点

链接直接软元件与链接刷新的不同点如表 9.11 所示。

表 9.11 链接直接软元件与链接刷新的不同点一览表

项目		链接直接软元件	链接刷新
程序中的表述方法	链接继电器	J□\K4B0~	B0~
	链接寄存器	J□\W0~	W0~
	链接特殊继电器	J□\K4SB0~	SB0~
	链接特殊寄存器	J□\SW0~	SW0~
步数		2 步	1 步
对网络模块的存取范围		J□\□0 ～ 3FFF	刷新参数中设定的范围
存取数据的保证范围		字（16 位）单位	

备 注

1. 关于 MELSECNET/G 网络系统，请参照手册。
   ☞ MELSECNET/G 网络系统参考手册（控制网络篇）
2. 关于 MELSECNET/H 网络系统，请参照以下手册。
   ☞ Q 系列 MELSECNET/H 网络系统参考手册（可编程控制器网络篇）
   ☞ Q 系列 MELSECNET/H 网络系统参考手册（远程 I/O 网络篇）
3. 关于网络参数、公共参数、网络刷新参数，请参照以下手册。
   • 详细说明：
   ☞ MELSECNET/G 网络系统参考手册（控制网络篇）
   ☞ Q 系列 MELSECNET/H 网络系统参考手册（可编程控制器网络篇）
   ☞ Q 系列 MELSECNET/H 网络系统参考手册（远程 I/O 网络篇）
   • 设定方法：
   ☞ GX Developer 操作手册

9.5 模块访问软元件

9.5.1 智能功能模块软元件 (U □ \G □)

(1) 智能功能模块软元件

智能功能模块软元件是指通过 CPU 模块对主基板以及扩展基板上安装的智能功能模块 / 其它特殊功能模块的缓冲存储器进行直接存取的软元件。

(2) 智能功能模块软元件的指定方法与使用示例

(a) 指定方法

智能功能模块软元件是通过智能功能模块 / 特殊功能模块的 I/O 地址号与缓冲存储器地址来指定的。

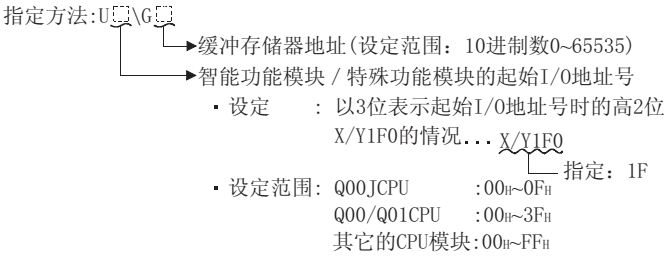


图 9.52 智能功能模块软元件的指定方法

(b) 使用示例

将 I/O 地址号 020 上安装的 Q64AD 型模拟 / 数字转换模块 (X/Y020~02F) 的 CH.1~4 的数字输出值存储到 D0~3 中时，如图 9.53 所示进行指定。

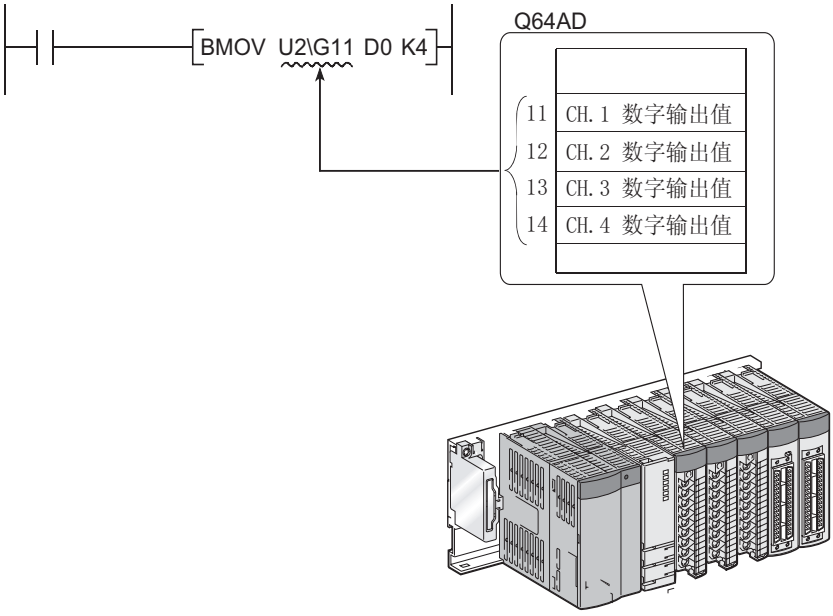


图 9.53 智能功能模块软元件的使用示例

要 点

使用智能功能模块软元件，可对缓冲存储器添加软元件注释。

☞GX Developer 操作手册

(3) 处理速度

智能功能模块软元件的处理速度如下。

- 通过智能功能模块软元件进行写入 / 读出时，比通过 FROM/TO 指令进行写入 / 读出时其处理速度要快一些。（例如，“MOV U2\G11 D0”的情况。）
- 用 1 个指令对智能功能模块 / 特殊功能模块注 9.21 的缓冲存储器进行读出以及其它处理时，请以 FROM/TO 指令的处理速度与指令的处理速度的合计值为粗略估算标准。
（例如，“+ U2\G11 D0 D10”的情况。）

基本 过程 冗余 注 9.21 注 9.21 注 9.21

通用 UD 注 9.21

要 点

在顺控程序内 2 次以上使用智能功能模块软元件对缓冲存储器的数据进行写入 / 读出时，如果使用 FROM/TO 指令在程序的某处进行写入 / 读出，可提高处理速度。

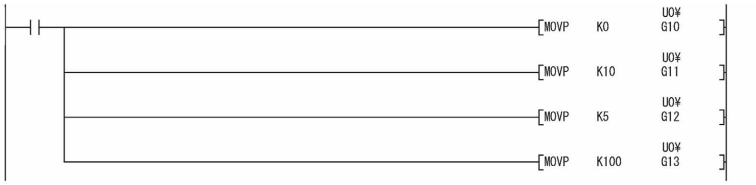


图 9.54 使用多个智能功能模块软元件进行写入的情况

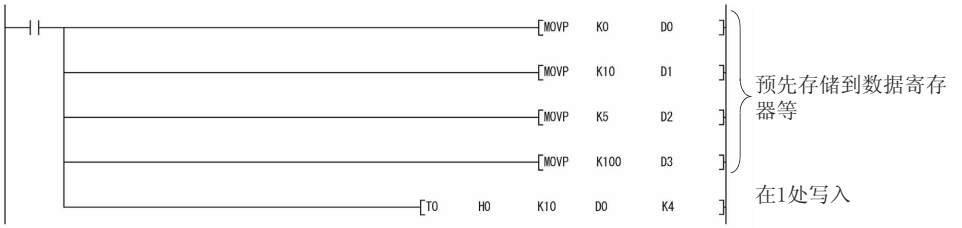


图 9.55 使用 TO 指令在程序的 1 处写入的情况

备 注

1) 关于缓冲存储器的地址和用途，请参照所使用的智能功能模块 / 特殊功能模块注 9.21 的手册。

2) 关于 FROM/TO 指令，请参照以下手册。
☞ QCPU(Q 模式)/QnACPU 编程手册（公共指令篇）

基本 过程 冗余 注 9.21 注 9.21 注 9.21

通用 UD 注 9.21

基本 过程 冗余 注 9.21 注 9.21 注 9.21

通用 UD 注 9.21

在基本模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 中，不能使用 AnS/A 系列对应的特殊功能模块。

冗余
注 9.22

9.5.2 多 CPU 间共享软元件 (U3En\G □)

(1) 多 CPU 间共享软元件

多 CPU 间共享软元件是访问多 CPU 系统的各 CPU 模块的 CPU 共享内存的软元件。

(2) 多 CPU 间共享软元件的特点

多 CPU 间共享软元件的特点如下所示。

- 与至 CPU 共享内存的写入 (S.TO/TO 指令)/ 读出指令 (FROM 指令) 相比，其传送速度更快，可以减少程序的步数。
- 通过使用多 CPU 间共享内存，可以进行位操作。
- 通过对多 CPU 间共享软元件设定软元件注释，可以提高程序的可读性。
- 由于可以将 CPU 共享内存中的信息直接指定为指令变量，因此无需互锁用的软元件等。

(3) 多 CPU 间共享软元件的指定方法

多 CPU 间共享软元件是通过 CPU 模块的 I/O 地址号及 CPU 共享内存地址指定。

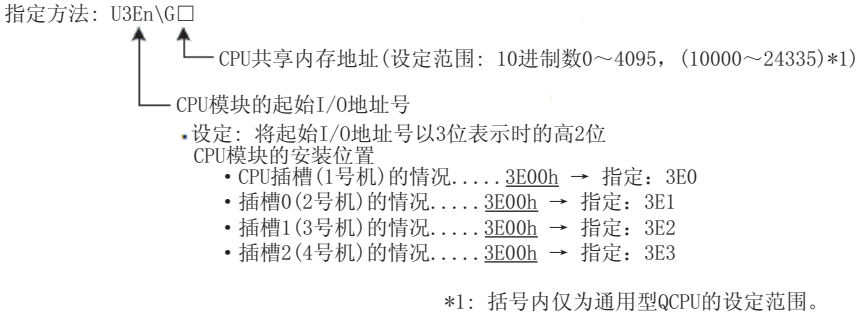


图 9.56 多 CPU 间共享软元件的指定方法

备 注

关于多 CPU 间共享软元件的详细内容，请参照以下手册。

☞QCPU 用户手册 (多 CPU 系统篇)

冗余
注 9.22

在冗余 CPU 中，不能使用多 CPU 间共享软元件。

9.6 变址寄存器 (Z)/通用运算寄存器 (Z)

9.6.1 变址寄存器 (Z)

(1) 变址寄存器

变址寄存器是用于顺控程序中使用的软元件的间接设定（变址修饰）的软元件。

通用 UD 注 9.23

使用 1 个变址寄存器来进行变址修饰。 *注 9.23*

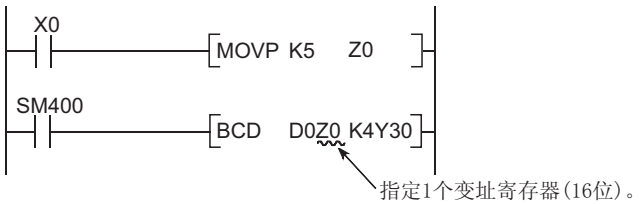


图 9.57 变址寄存器

变址寄存器的个数如下所示。

基本模式 QCPU : Z0 ～ 9(10 个)

高性能 QCPU，

过程 CPU、冗余 CPU : Z0 ～ 15(16 个)

通用型 QCPU : Z0 ～ 19(20 点)

(2) 变址寄存器的位构成

(a) 位构成与写入及读出单位

1 个变址寄存器由 16 位构成，能够以 16 位为单位进行写入及读出。

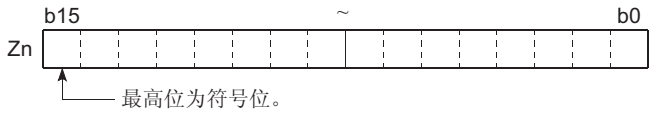


图 9.58 变址寄存器的位构成

要 点

变址寄存器是带符号使用的。

HEX(16 进制）的情况下可存储 0000H~FFFFH，但由于最高位为符号位，所以能够指定的数值范围为 -32768~32767。

通用 UD 注 9.23

在通用型 QCPU 中，对于连号访问方式的文件寄存器 (ZR)，可以进行使用 2 个变址寄存器的 32 位变址修饰。

(b) 以 32 位指令使用变址寄存器

以 32 位指令使用变址寄存器时，以 Zn 与 Zn+1 为处理对象。

在顺控程序中指定的变址寄存器编号 (Zn) 为低 16 位，在顺控程序中指定的变址寄存器编号 +1 的变址据寄存器为高 16 位。

例如用 DMOV 指令指定了 Z2 时，Z2 为低 16 位，Z3 为高 16 位。(32 位构成时的最高变为符号位。)

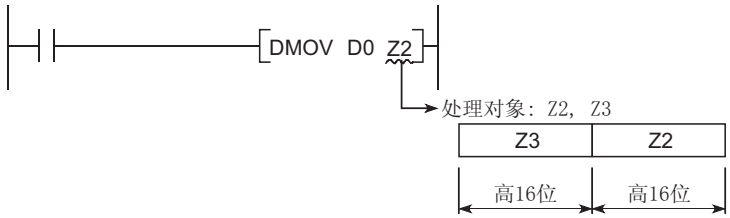


图 9.59 利用 32 位指令进行数据发送以及存储目标

备 注

关于使用了变址寄存器的变址修饰，请参照以下手册。

☞QCPU(Q 模式)/QnACPU 编程手册（公共指令篇）

9.6.2 通用运算寄存器 (Z)

(1) 通用运算寄存器

在通用型 QCPU 中，通过在寄存器运算期间使用变址寄存器，可以提高运算速度。

用于上述情况下的变址寄存器称为通用运算寄存器。

(2) 软元件号

由于通用运算寄存器与变址寄存器为相同的软元件，因此变址修饰时不要与通用运算寄存器的软元件号重复。

要 点

只有通用型 QCPU 才可以提高运算速度 。

在基本模式 QCPU、高性能模式 QCPU、过程 CPU、冗余 CPU 中无此效果。

备 注

关于使用通用运算寄存器时的处理时间，请参照以下手册。

☞ QCPU(Q 模式)/QnACPU 编程手册（公共指令篇）

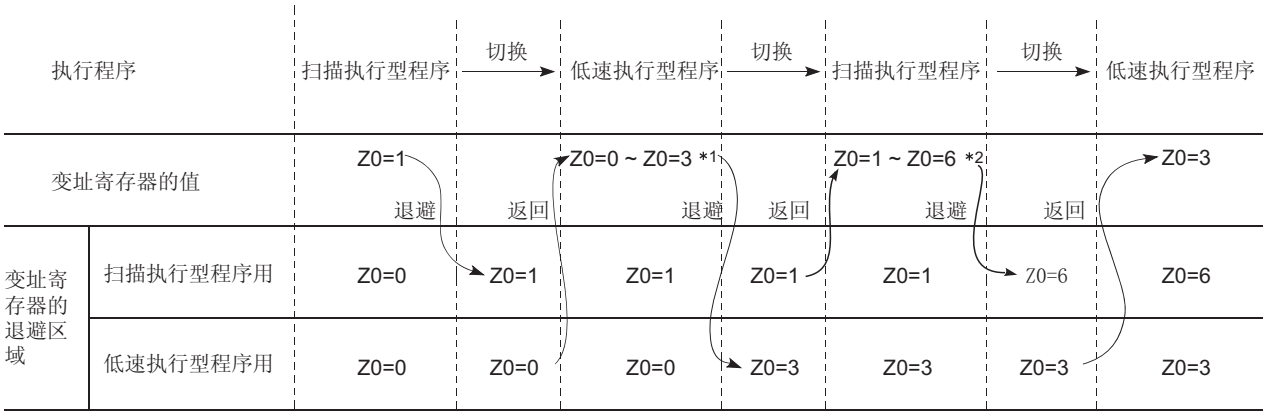
基本 注 9.24
冗余 注 9.24
通用 注 9.24

9.6.3 扫描执行 / 低速执行类型程序切换时的处理

在 CPU 模块中，扫描执行型程序与低速执行型程序切换时，进行变址寄存器 (Z0~15) 的内容的退避（保护）与返回。

(1) 由扫描执行型程序切换为低速执行型程序时
在 CPU 模块中，扫描执行型程序的变址寄存器的值被退避，低速执行型程序的变址寄存器的值将返回。

(2) 由低速执行型切换为扫描执行型程序时
在 CPU 模块中，低速执行型程序的变址寄存器的值被退避，扫描执行型程序的变址寄存器的值将返回。



*1：在低速执行型程序中将Z0变更为3。
*2：在扫描执行型程序中将Z0变更为6。

图 9.60 扫描执行类型程序 / 低速执行类型程序切换时的变址寄存器的退避与返回

(3) 变址寄存器的传递
在扫描执行型程序与低速执行型程序之间进行变址寄存器的传递时，请使用字元件。

基本 注 9.24　冗余 注 9.24　通用 注 9.24

在基本模式 QCPU、冗余 CPU、通用型 QCPU 中，由于不能使用低速执行类型程序，因此在使用基本模式 QCPU、冗余 CPU、通用型 QCPU 时，无需理会本项的内容。

9.6.4 由扫描 / 低速执行型程序切换为中断 / 恒定周期执行型程序时的处理

基本 注 9.25

冗余 注 9.26　通用 UD 注 9.26

在 CPU 模块中，扫描 / 低速执行型程序与中断 / 恒定周期执行型程序之间进行切换时，进行以下处理。

- 变址寄存器的内容的退避（保护）· 返回
- 文件寄存器的块号的退避（保护）· 返回

(1) 退避（保护）· 返回的设定

扫描 / 低速执行型程序与中断 / 恒定周期执行型程序之间进行切换时，对变址寄存器的内容是退避 · 返回还是不退避，可以通过可编程控制器参数的可编程控制器系统设定进行选择设定。

中断程序 / 恒定周期型程序中，不向变址寄存器写入时，如果将 “中断程序 / 恒定周期程序” 设定为 “高速执行”，便可提高切换速度。

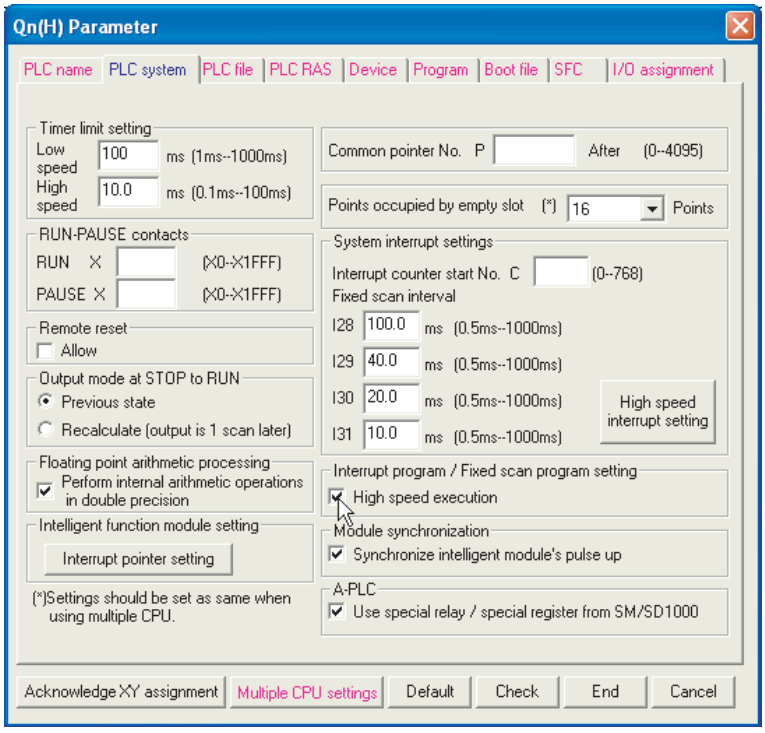


图 9.61 中断程序 / 恒定周期程序设定

基本 注 9.25

在基本模式 QCPU 中，不能使用低速执行型程序以及恒定周期型程序。
本项的 “扫描 / 低速执行型程序”，请换成 “主程序 / 副程序” 未阅读。
“中断程序 / 恒定周期型程序”，请换成 “中断程序” 未阅读。

冗余 注 9.26　通用 UD 注 9.26

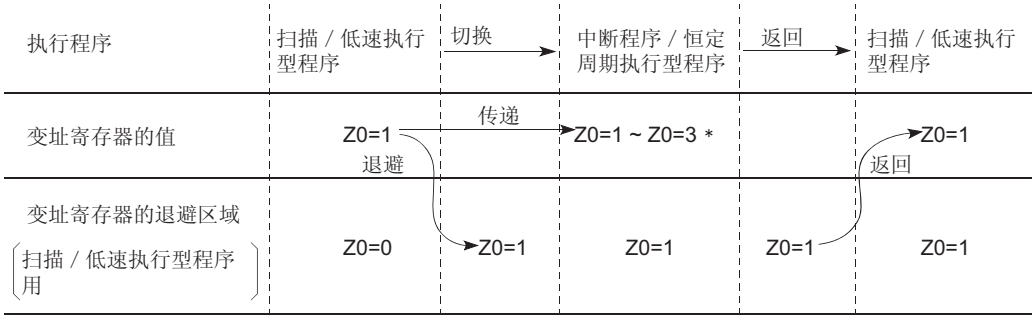
在冗余 CPU、通用型 QCPU 中，不能使用低速执行类型程序。
请在阅读时将本项的 “扫描 / 低速执行类型程序” 替换为 “扫描执行类型程序”。

(2) 变址寄存器的处理

(a) 不选定 “高速执行”时

1) 由扫描 / 低速执行型程序切换为中断 / 恒定周期执行型程序时
在 CPU 模块中，扫描 / 低速执行型程序的变址寄存器的值被退避，并被传递到中断程序 / 恒定周期执行类型程序中。

2) 由中断程序 / 恒定周期型程序切换为扫描 / 低速执行型程序时
CPU 模块使退避的变址寄存器的值返回。



*:中断程序中将Z0变更为3。

图 9.62 变址寄存器的退避（保护）• 返回（未选定 “高速执行”时）

要 点

将变址寄存器的值从中断程序 / 恒定周期型程序传递到扫描 / 低速执行型程序时，请使用字元件。

(b) 选定 “高速执行”时

1) 由扫描 / 低速执行型程序切换为中断 / 恒定周期执行型程序时
在 CPU 模块中，不进行变址寄存器的退避・返回。

2) 中断程序 / 恒定周期程序切换为扫描 / 低速执行型程序时
如果在中断 / 恒定周期执行型程序中向变址寄存器写入数据，则扫描 / 低速执行型程序中使用过的变址寄存器的值将被破坏。

执行程序	扫描 / 低速执行型程序	切换	中断程序 / 恒定周期执行型程序	返回	扫描 / 低速执行型程序
变址寄存器的值	Z0=1	传递	Z0=1 ~ Z0=3 *	传递	Z0=3
变址寄存器的退避区域 〔扫描/低速执行型程序用〕	Z0=0	Z0=1	Z0=0	Z0=0	Z0=0

* 中断程序中将Z0变更为3。

图 9.63 变址寄存器的退避（保护）・返回（选定 “高速执行”时）

在中断 / 恒定周期执行型程序中向变址寄存器写入数据时，请用 ZPUSH 指令 / ZPOP 指令进行变址寄存器的退避・返回。

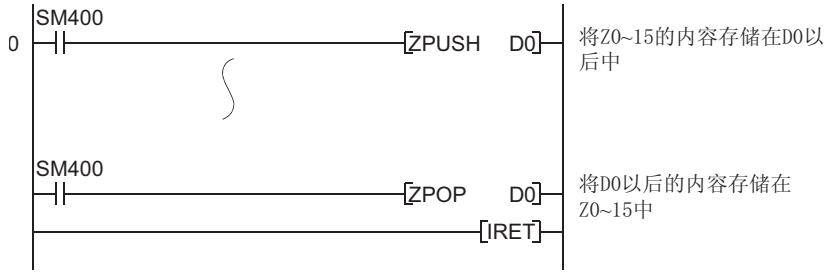


图 9.64 在中断程序 / 恒定周期型程序中向变址寄存器写入数据时

(3) 文件寄存器的块号的处理

(a) 由扫描 / 低速执行型程序切换为中断程序 / 恒定周期执行型程序时
在 CPU 模块中，扫描 / 低速执行型程序的文件寄存器的块号被退避，并被传递到中断程序 / 恒定周期执行类型程序中。

(b) 由中断程序 / 恒定周期型程序切换为扫描 / 低速执行型程序时
CPU 模块使退避的文件寄存器的块号返回。

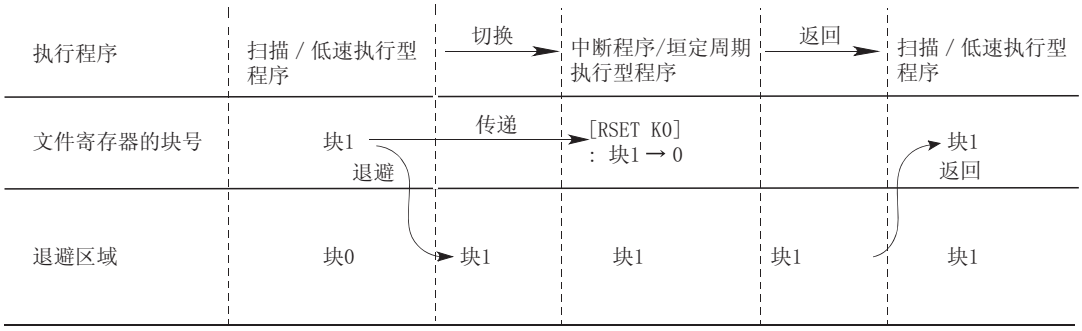


图 9.65 文件寄存器的块号的退避（保护）• 返回

9.7 文件寄存器 (R)

(1) 文件寄存器

文件寄存器是数据寄存器的扩展用软元件。

文件寄存器能够与数据寄存器以相同的速度进行处理。

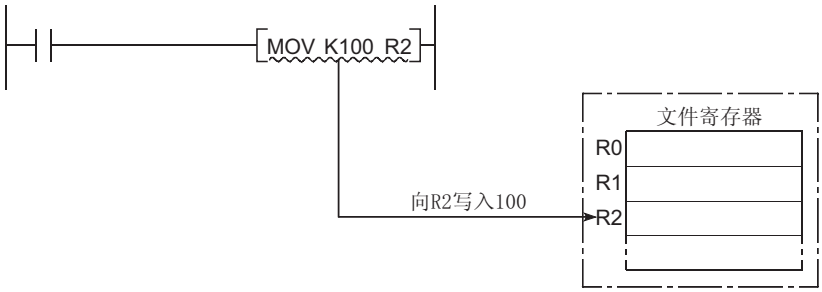


图 9.66 向文件寄存器的写入

(2) 文件寄存器的位构成

(a) 位构成与写入及读出单位

1 个文件寄存器由 16 位构成，能够以 16 位为单位进行写入及读出。

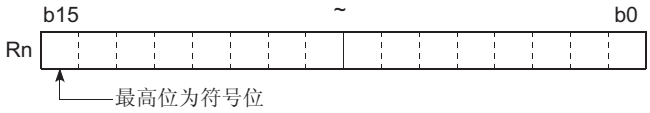


图 9.67 文件寄存器的位构成

(b) 32 位指令中使用文件寄存器

32 位指令中使用文件寄存器时，Rn 与 Rn+1 为处理对象。

通过顺控程序指定的文件寄存器编号 (Rn) 为低 16 位，通过顺控程序指定的文件寄存器编号 +1 的文件寄存器为高 16 位。

例如用 DMOV 指令指定了 R2 时，R2 为低 16 位，R3 为高 16 位。

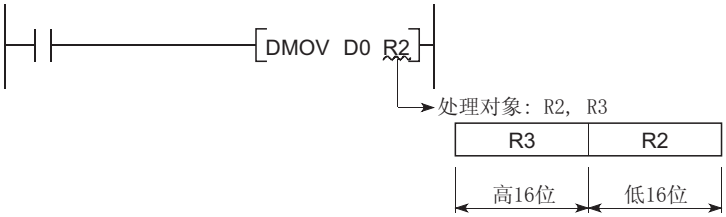


图 9.68 利用 32 位指令进行数据发送及存储目标

2 个文件寄存器中，可存储 -2147483648~2147483647 或者 0_H~$FFFFFFFF_H$ 的数据。

(32 位构成时的最高为符号位。)

(3) 文件寄存器的清除

文件寄存器的内容是由 CPU 模块内置的电池保持的。

即使可编程控制器的电源 OFF 或 CPU 模块复位，文件寄存器的内容仍将保持。(进行锁存清除也不会初始化。)注 9.27

通用 UD 注 9.27

将文件寄存器的内容初始化时，请利用顺控程序或者 GX Developer 进行数据清除操作。

(a) 利用顺控程序进行清除

图 9.69 清除文件寄存器 R0~R999 的示例

(b) 利用 GX Developer 进行清除

利用 GX Developer 进行清除时，请通过 “在线”→“可编程控制器内存清除”选定文件寄存器全部清除，以清除数据。

9.7.1 文件寄存器的存储地点

存储文件寄存器的存储器有 3 种：标准 RAM、SRAM 卡和 Flash 卡。

文件寄存器的存储地点因 CPU 模块而异。

各 CPU 模块的文件寄存器的存储地点如表 9.12 所示。

表 9.12 可以存储文件寄存器的地点

CPU 模块		存储地点
基本模式 QCPU	Q00JCPU	无（禁止使用文件寄存器）
	Q00CPU，Q01CPU	标准 RAM
高性能模式 QCPU，过程 CPU，冗余 CPU 通用型 QCPU		标准 RAM，存储卡 (SRAM 卡，Flash 卡)

通用 UD 注 9.27

在通用型 QCPU 中，可以设定文件寄存器的锁存范围。

应在可编程控制器参数的软元件设定中设定文件寄存器的锁存范围。(☞ 6.3 节 (5))

9.7.2 文件寄存器的容量

(1) 使用标准 RAM 时

标准 RAM 可存储以下点数的文件寄存器。

表 9.13 各 CPU 模块的文件寄存器的容量

CPU 模块		点数 *1*2
基本模式 QCPU	Q00JCPU	禁止使用文件寄存器
	Q00CPU，Q01CPU	64k 点
高性能模式 QCPU	Q02CPU	32k 点
	Q02HCPU，Q06HCPU	64k 点
	Q12HCPU，Q25HCPU	128k 点
过程 CPU	Q12PHCPU，Q25PHCPU	128k 点
冗余 CPU	Q12PRHCPU，Q25PRHCPU	128k 点
通用型 QCPU	Q02UCPU	64k 点
	Q03UDCPU	96k 点
	Q04UDHCPU	128k 点
	Q06UDHCPU	384k 点

* 1: 在基本模式 QCPU 中没有本地软元件，因此文件寄存器中可使用上述点数。
在高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 中，在未使用局部软元件的情况下可使用上述点数。

* 2: 标准 RAM 的容量因 CPU 模块的版本而异，因此可使用的点数也有所不同。(附录 4.1，附录 4.2)

基本

(2) 使用 SRAM 卡时注 9.28

能够扩展的点数、块数根据 CPU 模块的不同而有所不同。

- 使用高性能模式 QCPU、过程 CPU、冗余 CPU 时
  1 个文件最多可存储 1017k 点。
  由于 1 块为 32k 字单位，因此最多可存储 32 块。
- 通用型 QCPU 时
  1 个文件最多可存储 4086k 点。
  由于 1 块为 32k 字单位，因此最多可存储 128 块。

但能够扩展的点数、块数因存储卡中存储的程序、软元件注释等的容量而异。

基本

注 9.28 在基本模式 QCPU 中，不能使用存储卡。

基本

(3) 使用 Flash 卡时注 9.29

能够扩展的点数、块数根据 CPU 模块的不同而有所不同。

- 使用高性能模式 QCPU、过程 CPU、冗余 CPU 时
  1 个文件最多可存储 1018k 点。
  由于 1 块为 32k 字单位，因此最多可存储 32 块。
- 通用型 QCPU 时
  1 个文件最多可存储 2039k 点。
  由于 1 块为 32k 字单位，因此最多可存储 64 块。

但能够扩展的点数、块数因所使用的存储卡容量及存储卡中存储的程序、软元件注释等的容量而异。

备 注

关于能够在 CPU 模块中使用的存储卡，请参照 5.2.6 项。

基本

注 9.29 在基本模式 QCPU 中，不能使用存储卡。

9.7.3 存储对象不同时存储器的存储方法的不同点

基本

文件寄存器的存取方法因各存储器而不同。

注 9.30

表 9.14 不同存储器时文件寄存器的存取方法的不同点

存取方法		标准 RAM	SRAM 卡注 9.30	Flash 卡注 9.30
程序的写入		○	○	○
程序的读取		○	○	×
软元件内存的可编程控制器写入		○	○	○
软元件内存的可编程控制器读取		○	○	×
数据的变更方法	利用 GX Developer 的在线测试操作	○	○	×
	利用 GX Developer 的可编程控制器写入	○	○	×
	利用 GX Developer 的可编程控制器写入（快闪卡 ROM）	×	×	○
	利用串行口通讯模块的批量写入	○	○	×
	来自 GOT900/1000 系列的软元件写入	○	○	×
	来自 GOT900/1000 系列的随机写入指令	○	○	×

备 注

可存储文件寄存器的存储器因 CPU 模块而异。（☞9.7.1 项）

基本

注 9.30

在基本模式 QCPU 中，不能使用存储卡。

基本

注 9.31

9.7.4 文件寄存器的登录步骤

使用文件寄存器时，按照以下步骤将文件寄存器的文件登录到 CPU 模块中。

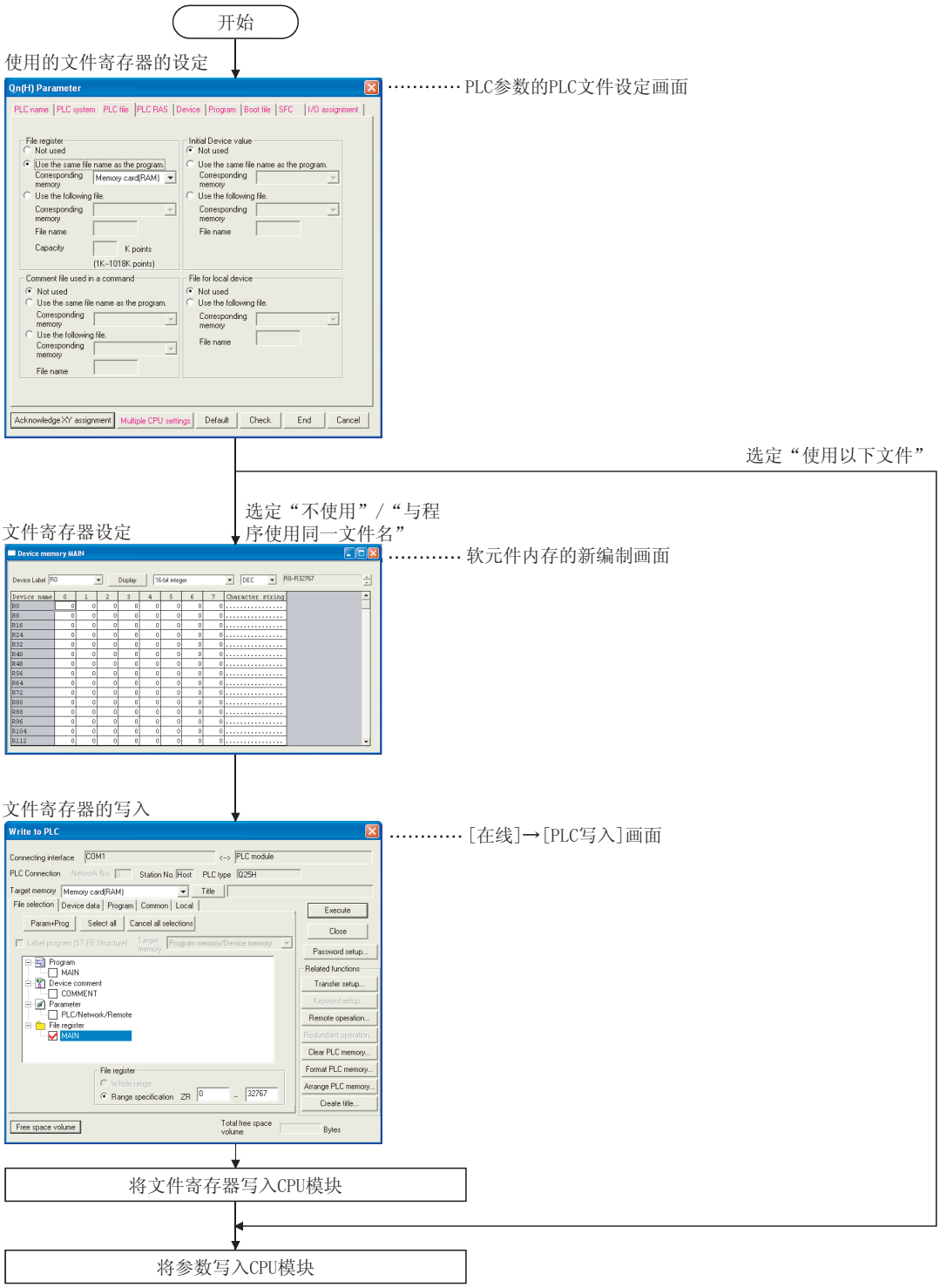


图 9.70 文件寄存器的登录步骤

基本

注 9.31

使用基本模式 QCPU 时，不需要进行文件寄存器的登录。
文件寄存器自动登录到标准 RAM 中。

(1) 使用文件寄存器的设定

在顺控程序中使用标准 RAM 或存储卡内的哪个文件寄存器，是通过可编程控制器参数的可编程控制器文件设定进行设定。

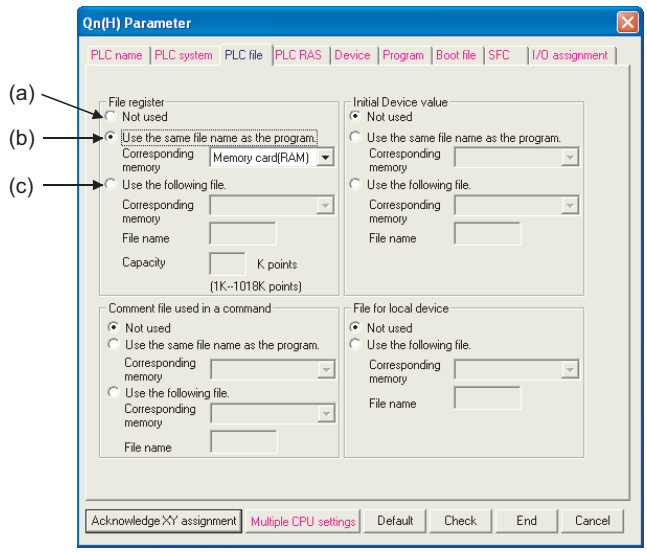


图 9.71 文件寄存器的设定

(a) 不使用

在以下情况下进行设定。

- 不使用文件寄存器时
- 在顺控程序中指定使用的文件寄存器时
  （使用文件寄存器是通过 QDRSET 指令进行指定的。）

(b) 使用与程序相同的文件名

执行与顺控程序文件名相同的文件寄存器时进行设定。

1) 程序切换后的动作

程序切换后，文件寄存器的文件名也将自动切换为与程序相同的名称。

作为只执行文件寄存器的程序中使用的本地软元件，使用起来将非常方便。

2) 文件寄存器点数的设定

能够使用的文件寄存器点数，通过 GX Developer 的 [在线]→[可编程控制器写入] 进行设定。

示例

存在与程序A~C的文件名相同的文件寄存器A~C的情况下，如下所示：

- 执行程序A时 …… 存取文件寄存器A
- 执行程序B时 …… 存取文件寄存器B
- 执行程序C时 ……F存取文件寄存器C

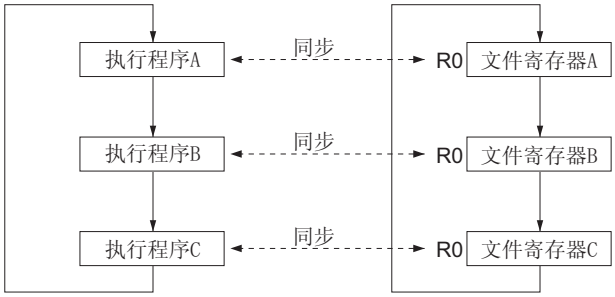


图 9.72 程序切换后的文件寄存器

要 点

有的指令不能指定各程序设定的文件寄存器。

详细内容请参照各指令的编程手册的可使用的软元件。

(c) 使用以下文件

1 个文件寄存器在执行的全程序中通用时进行设定。

设定文件寄存器中使用的 “对象存储器”、“文件名”和 “容量”，并写入 CPU 模块，便可编制参数中指定的文件寄存器的文件。

不设定容量时的情况如下。

- 使用指定驱动器中存储的指定文件名的文件寄存器文件。(容量为存储的文件寄存器文件的容量。)
- 未设定容量时，指定驱动器中不存在指定文件名的文件寄存器文件时，则发生 “参数错误 (3002)”。
- 使用 ATA 卡时，不能将存储卡 (ROM) 设定为对象存储器。(ATA 卡不能存储文件寄存器。)

  将存储卡 (ROM) 设定为对象存储器，并将其写入 CPU 模块，则发生 “参数错误 (3000)”。

(2) 文件寄存器设定

在新建软元件内存时进行指定文件名的文件寄存器的设定。

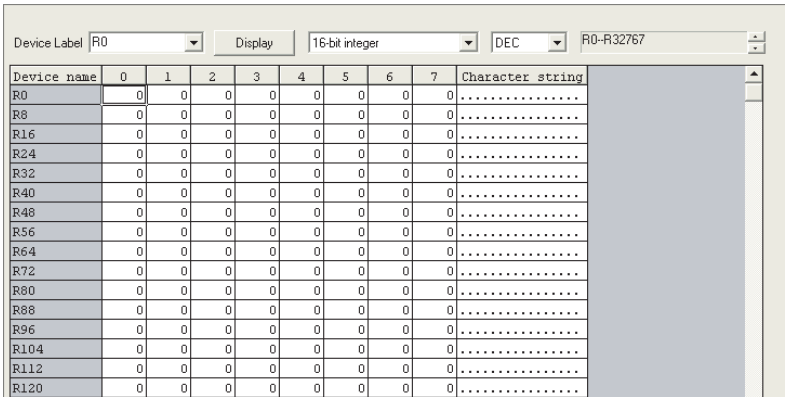


图 9.73 软元件内存画面

(a) 软元件

设定 Rn，显示文件寄存器的一览。

设定 Rn(以上情况下为 R0)，点击按钮 Display 按钮，将显示文件寄存器的一览。

(b) 数据的设定

编制设为文件寄存器的数据。

仅设定文件寄存器的容量时，不需要编制数据。

(3) 将文件寄存器文件登录到 CPU 模块

通过可编程控制器参数的可编程控制器文件设定，选定：

- 不使用 (☞本项 (1)(a))
- 使用与程序相同的文件名 (☞本项 (1)(b))

此时需要将文件寄存器文件登录到 CPU 模块。

登录到 CPU 模块，是通过 GX Developer 的 [在线]→[可编程控制器写入] 进行。

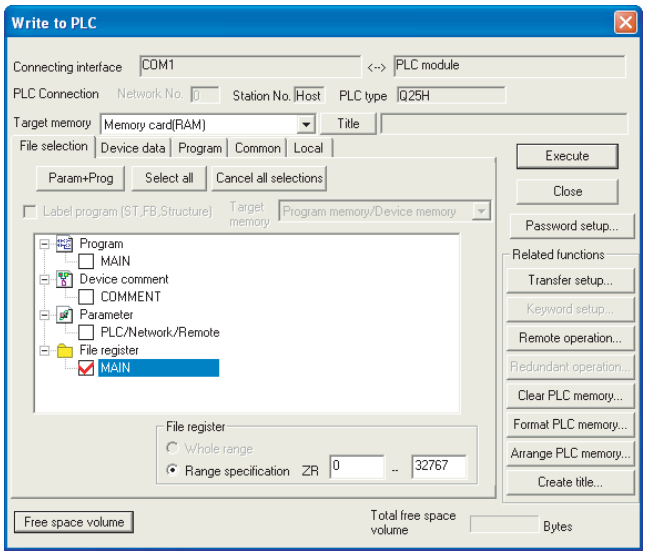


图 9.74 文件寄存器文件的登录

(a) 对象存储器

从标准 RAM、存储卡 (RAM) 和存储卡 (ROM) 中选择要登录文件寄存器的存储器。

使用与程序相同的文件名时，请将文件寄存器登录到在可编程控制器参数的可编程控制器文件设定中指定的存储器中。

(b) 文件寄存器文件的选定

选定文件寄存器的登录存储器后，将显示所设定的文件寄存器的文件名。

选定文件寄存器的文件。

(c) 文件寄存器容量与文件名的设定

设定文件寄存器容量以及写入 CPU 模块的文件名 (可编程控制器侧文件名)。

1) CPU 模块的文件寄存器容量

文件寄存器容量可以以 1 点为单位进行设定。

但作为文件预留 256 点单位的容量。

文件寄存器的设定即使不是从 ZR0 指定时，也将建立从 ZR0 到最终编号的文件。

例如，将文件寄存器的写入范围指定为 ZR1000~1791 时，则建立 ZR0~1791 的文件。

但从 ZR0 到 ZR999 将变为不定的数据，所以写入时请从 ZR0 开始指定。

此外，文件寄存器容量的检查是以 1k 点为单位进行，因此请将文件寄存器容量从 R0 开始以 1k 点为单位进行设定。

(d) 写入 CPU 模块

将指定点数的文件寄存器的文件登录到 CPU 模块的指定存储器。

9.7.5 文件寄存器的指定方法

(1) 块切换方式

块切换方式是指将使用的文件寄存器点数以 32k 点 (R0~32767) 为单位分开指定的方式。

使用了多个块时，请切换为 RSET 指令中使用的块号后进行指定。

各块都是以 R0~32767 进行指定。

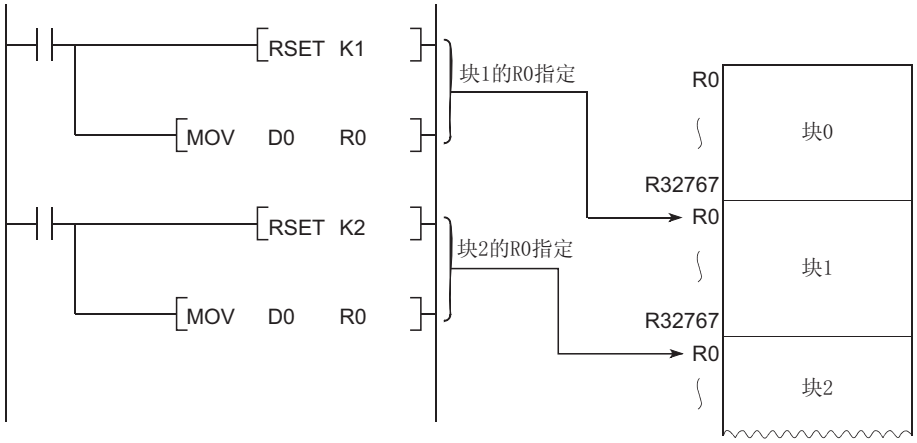


图 9.75 块切换方式

(2) 连号存取方式

连号存取方式是指以连续的软元件编号指定超过 32k 点的文件寄存器的方式。

可将多个块的文件寄存器作为连续的文件寄存器使用。

软元件名称使用 “ZR”。

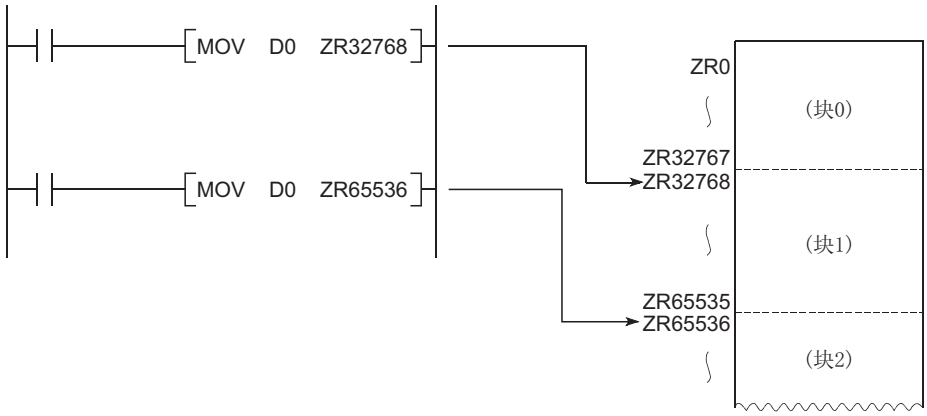


图 9.76 连号存取方式

要 点

可指定的块号以及 ZR 软元件的点数，因文件寄存器的存储地点 (☞9.7.1 项) 或者文件寄存器的容量 (☞9.7.2 项) 而异。

9.7.6 使用文件寄存器时的注意事项

(1) 使用基本模式 QCPU 时

即使向 64k 点以上的文件寄存器编号进行写入 / 读出，也不会导致出错。
但从文件寄存器读出时将存储不定数据，请加以注意。

要 点

基本模式 QCPU 的文件寄存器文件 (MAIN.QDR) 不能删除。
但文件寄存器的内容可删除。(☞9.7 节 (3))

(2) 使用高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 时

(a) 使用未登录 / 已登录的容量以上的文件寄存器编号时

1) 文件寄存器的文件未登录时

a) 高性能模式 QCPU、过程 CPU、冗余 CPU 的情况

文件寄存器的文件未登录到 CPU 模块时，即使进行向文件寄存器的写入 / 读出也不会发生出错。
但从文件寄存器读出时的情况如下所示：

- 在标准 RAM 的情况下，不定数据被存储。
- 在存储卡的情况下，“0H”被存储。

b) 通用型 QCPU 的情况

文件寄存器的文件未登录到 CPU 模块时，如果进行向文件寄存器的写入 / 读出，将发生 OPERATION ERROR(出错代码 : 4101)。

2) 向已登录的容量 (点数) 以上的文件寄存器编号的写入 / 读出

a) 高性能模式 QCPU、过程 CPU、冗余 CPU 的情况

即使向已登录的容量 (点数) 以上的文件寄存器编号进行写入 / 读出时，也不会发生出错。
但从文件寄存器读出时的情况如下所示：

- 在标准 RAM 的情况下，不定数据被存储。
- 在存储卡的情况下，“0H”被存储。

b) 通用型 QCPU 的情况

如果向已登录的容量 (点数) 以上的文件寄存器编号进行写入 / 读出，将发生 OPERATION ERROR(出错代码 : 4101)。

(b) 文件寄存器容量的检查

向文件寄存器进行写入 / 读出时，请先检查文件寄存器的容量，写入 / 读出均应在 CPU 模块中设定的容量 (点数) 以内进行。

1) 文件寄存器容量的确认方法

能够使用的文件寄存器容量可通过文件寄存器容量存储寄存器 (SD647) 进行确认。*1
SD647 中可存储 1k 点单位的文件寄存器容量。

*1 : 如果切换文件寄存器的文件，切换的文件的文件寄存器容量将被存储到 SD647 中。

要 点

以 1k 点分配文件寄存器容量时其余数将被舍去。
为了切实地检查使用范围，请以 1k 点 (1024 点) 为单位进行文件寄存器的设定。

2) 进行检查的时间

- 在使用了文件寄存器的程序中，请从步 0 检查文件寄存器容量。
- 执行文件寄存器的文件切换指令 (QDRSET) 后，请检查文件寄存器容量。
- 执行文件寄存器的块切换指令 (RSET) 前，请先确认切换后的块有 1k 点以上的容量。

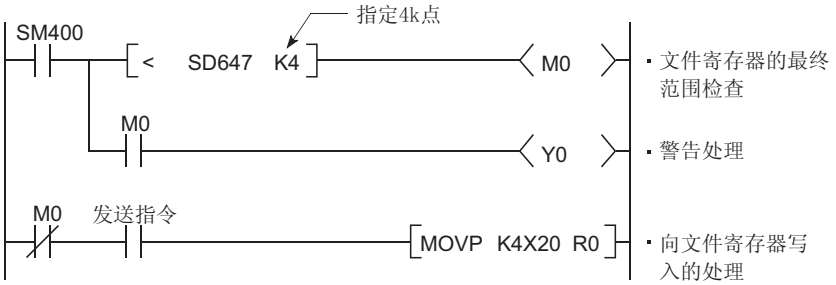
（文件寄存器容量）＞[32k 点×（切换块号）+ 1k 点]

3) 文件寄存器容量的确认步骤

- 明确各顺控程序中使用的文件寄存器的容量。
- 从顺控程序中被设为 SD647 的文件寄存器的全容量开始，检查是否含有使用点数以上的文件寄存器容量。

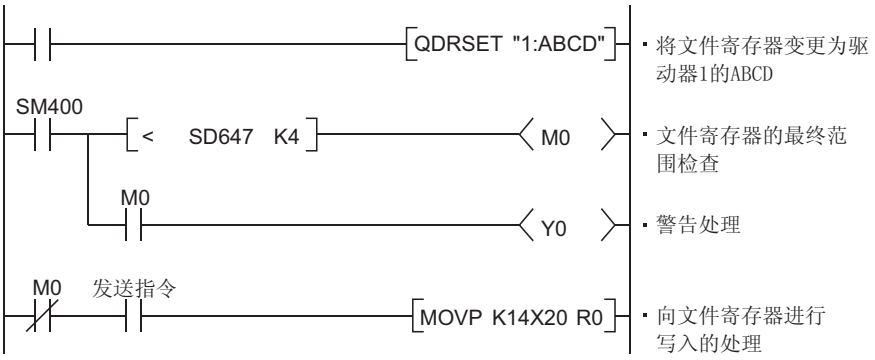
[程序示例 1]

从各程序的起始检查文件寄存器的使用范围时。



[程序示例 2]

执行 QDRSET 指令后检查文件寄存器的使用范围时。



[程序示例 3]

进行块切换时。

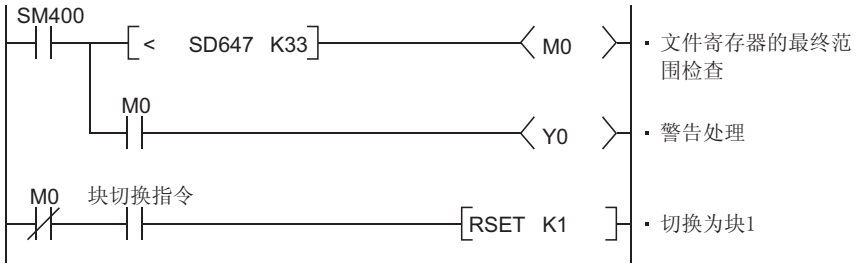


图 9.77 文件寄存器检查的程序示例

(c) 文件寄存器文件的删除

不需要的文件寄存器文件的删除，通过 GX Developer 的 [在线]→[可编程控制器数据删
除] 进行。

基本 过程 冗余
注 9.32 注 9.32 注 9.32
通用
注 9.32

(d) CPU 模块的版本不同时文件寄存器处理时间的不同*注 9.32*

在系列 No. 的高 5 位为 02092 以后的高性能模式 QCPU 中，如果使用标准 RAM 的存取指令通过连号存取方式（ZR□）指定文件寄存器，与 02091 以前的高性能模式 QCPU 相比，其处理时间将延长相当于 1 个指令的时间。（相当于 1 个指令的延长时间分别为，Q02CPU：平均 1.1μs，QnHCPU：平均 0.65μs。）

使用 MOV 指令时的处理时间如表 9.15 所示。

表 9.15 根据文件寄存器的指定方法的处理时间的异同

单位：μs

指令	Q12HCPU		Q02CPU	
	02092 以后	02091 以前	02092 以后	02091 以前
MOV K0 R0	0.11	0.11	0.26	0.26
MOV K0 ZR0	3.55	2.88	7.71	6.64

基本 过程 冗余
注 9.32 注 9.32 注 9.32
通用
注 9.32

在基本模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 中，文件寄存器的处理时间在 CPU 模块的各版本中无差异。

9.8 嵌套结构 (N)

(1) 嵌套结构

嵌套结构是指在主控指令 (MC 指令、MCR 指令) 中使用的，将动作条件以嵌套结构进行编程的软元件。

(2) 在主控指令中的指定方法

在主控指令中，利用梯形图的公共母线的开闭，编写高效的梯形图切换顺控程序的指令。

嵌套结构是从嵌套结构外侧开始以小编号 (N0~N14 的顺序) 进行指定。

关于嵌套的使用方法，请参照以下手册。

☞QCPU(Q 模式)/QnACPU 编程手册 (公共指令篇)

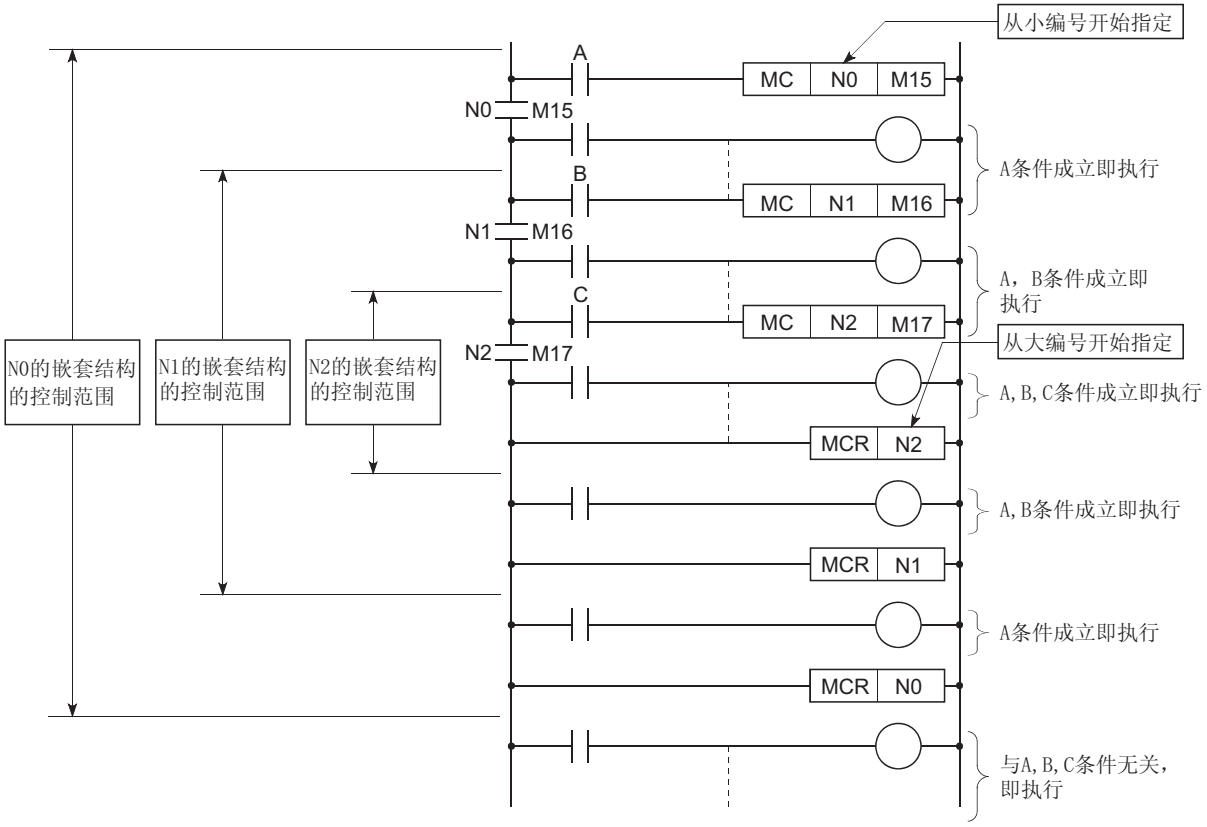


图 9.78 使用了嵌套结构的程序示例

9.9 指针 (P)

(1) 指针

指针是指在跳转指令 (CJ，SCJ，JUMP) 和副函数调用指令 (CALL 等) 中使用的软元件。

(2) 指针的用途

指针的用途有以下几种。

- 跳转指令 (CJ，SCJ，JUMP) 的跳转目标与标签 (跳转目标起始的指定)
- 副函数调用指令 (CALL，CALLP) 的调用目标与标签 (副程序起始的指定)

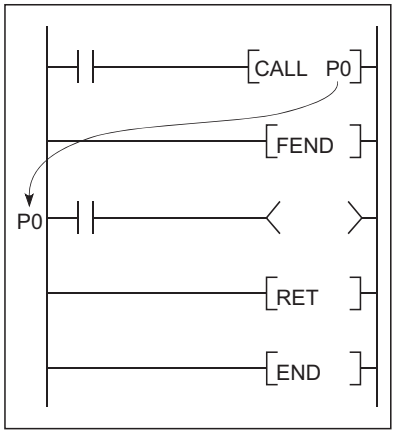


图 9.79 使用了指针的程序

(3) 指针的种类

(a) 基本模式 QCPU 的情况

基本模式 QCPU 不能执行多个程序，不存在局部指针和公共指针的区别。

(b) 高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 的情况

指针有以下 2 种。

- 局部指针 (☞9.9.1 项)
  各程序中独立使用的指针
- 公共指针 (☞9.9.2 项)
  利用副函数调用指令可调用所执行的全部程序的指针。

(4) 能够使用的指针的点数

能够使用的指针的点数因 CPU 模块的不同而不同。

表 9.16 各种 CPU 模块的可使用的指针点数

CPU 模块	点数
基本模式 QCPU	300 点
高性能模式 QCPU 过程 CPU 冗余 CPU 通用型 QCPU	4096 点

备 注

关于跳转指令、副函数调用指令，请参照以下手册。

☞QCPU(Q 模式)/QnACPU 编程手册（公共指令篇）

基本
注 9.33

9.9.1 局部指针

(1) 局部指针

局部指针是用于各程序内的跳转、副程序的调用的指针。

在各程序内可使用同一指针 No.

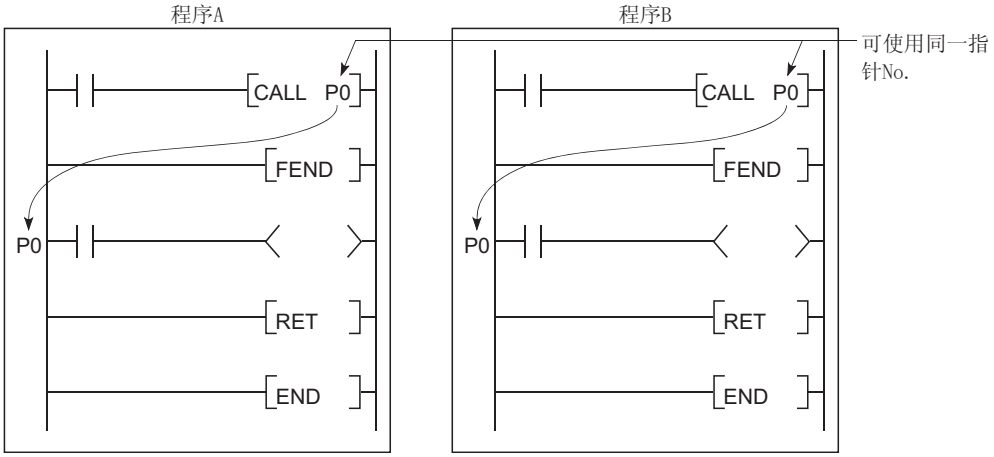


图 9.80 各程序中使用同一指针（局部指针）

基本
注 9.33

在基本模式 QCPU 中，不能执行多个程序，因此不存在局部指针、公共指针的区别。
使用基本模式 QCPU 时，不必在意本项的内容。

(2) 局部指针点数的思路

局部指针，是在程序内存中存储的全部程序中分配之后进行使用。

局部指针点数可从 P0 到所使用的局部指针的最大 No.。（使用点数由 CPU 模块的系统算出。）

例如，仅仅使用 P99 时，也可使用从 P0 到 P99 的 100 点。

在多个程序中使用指针时，在各程序中请从 P0 开始按照顺序使用指针。

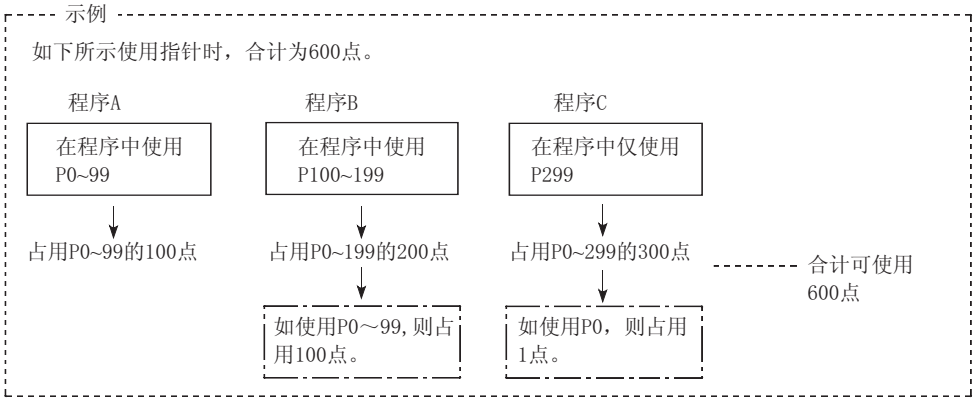


图 9.81 局部指针点数的思路

(3) 局部指针使用时的注意事项

(a) 关于已记载了局部指针的程序

记载了局部指针的程序文件不能进行从其它程序的跳转。

记载了局部指针的程序文件的副程序，通过 ECALL 指令从其它程序调用。

(b) 关于局部指针指针的合计点数

各个程序中使用的指针的合计点数超过 4096 点时，将发生“指针构成错误（出错代码：4020)”。

基本
注 9.34

9.9.2 公共指针

(1) 公共指针

公共指针是指从执行的全部程序中调用副程序的指针。

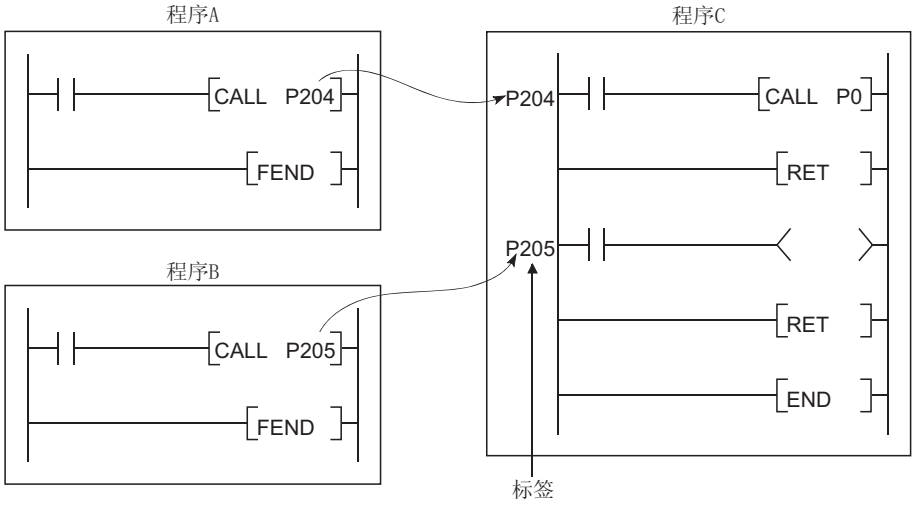


图 9.82 调用其它程序的指针（公共指针）

基本
注 9.34

在基本模式 QCPU 中，不能执行多个程序，因此不存在局部指针、公共指针的区别。
使用基本模式 QCPU 时，不必在意本项的内容。

(2) 公共指针的使用范围

使用公共指针时，通过可编程控制器参数的可编程控制器系统设定来设定公共指针的起始编号。

从所设定的指针编号到 P4095 为止均为公共指针。

但参数中可设定的公共指针的起始编号是局部指针的使用合计点数以后的数值。

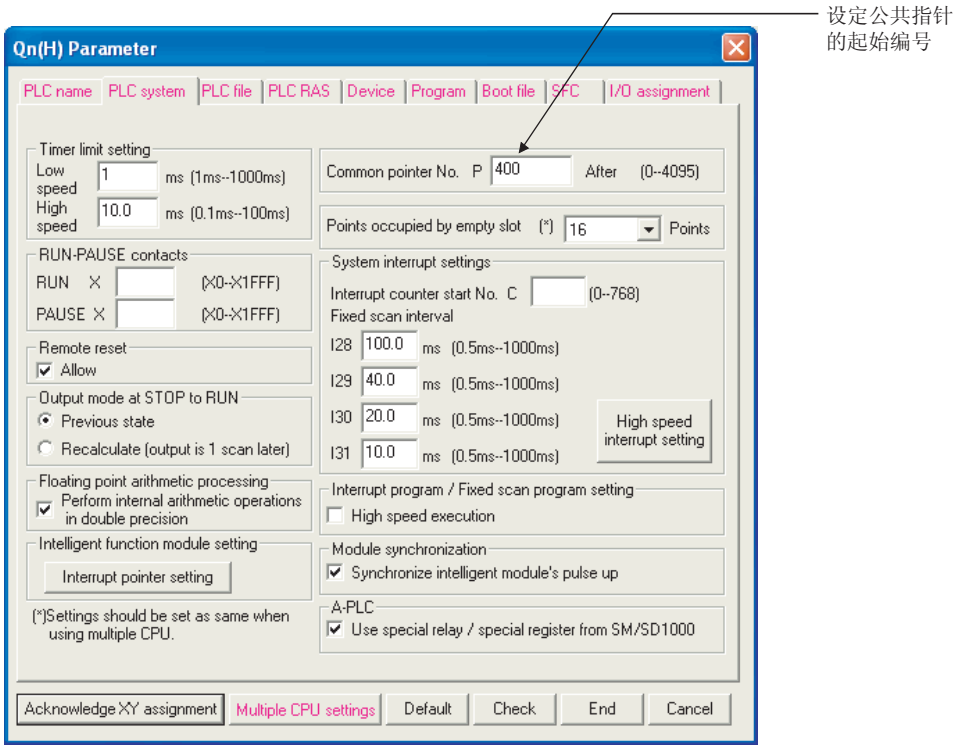


图 9.83 公共指针的设定画面

例如，程序 A 中 100 点、程序 B 中 100 点、程序 C 中 200 点，在局部指针中共使用 400 点时， P400 以后可设为公共指针。

(3) 使用公共指针时的注意事项

(a) 将同一指针 No. 作为标签使用时
同一指针 No. 不能作为标签使用。
如果将同一指针 No. 作为标签使用，则会发生 “指针构成错误（出错代码：4021)”。

(b) 局部指针的合计点数超过公共指针的起始编号时
如果各程序中使用的局部指针的合计点数超过了公共指针的起始编号，则会发生“指针构成错误（出错代码：4020)”。

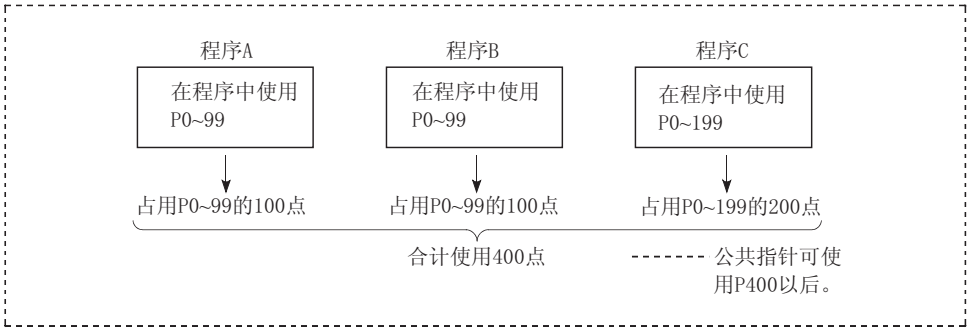


图 9.84 公共指针的使用范围的思路

要 点

不能通过跳转指令跳转到其它程序的公共指针。
公共指针只能通过副函数调用指令来使用。

9.10 中断指针 (I)

(1) 中断指针

中断指针是指在中断程序的起始作为标签使用的软元件。

中断指针可在执行的全部程序中使用。

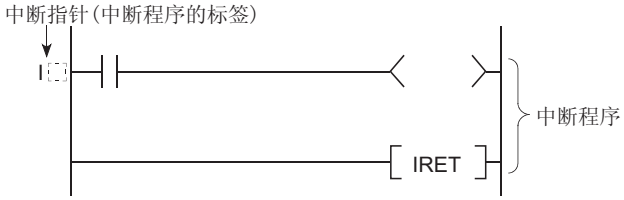


图 9.85 中断指针

(2) 可使用的中断指针的点数

可使用的中断指针的点数如表 9.17 所示。

表 9.17 各 CPU 模块中可使用的中断指针点数

CPU 模块	点数
基本模式 QCPU	128 点 (I0~127)
高性能模式 QCPU 过程 CPU 冗余 CPU 通用型 QCPU	256 点 (I0~255)

高性能
注 9.35

(3) 中断因子

中断指针的中断因子如表 9.18 所示。

表 9.18 中断因子的分类

中断因子	中断指针编号	内容	对应 CPU 模块				
			基本模式 QCPU	高性能模式 QCPU	过程 CPU	冗余 CPU	通用型 QCPU
由中断模块*1 进行的中断 *注 9.35*	I0 ～ 15	来自中断模块*1 的中断输入。	○	○	○	○	○
通过顺控启动发生模块进行的中断	I16 ～ 27	至 CPU 模块的，来自可进行中断启动的与 AnS/A 系列对应的特殊功能模块*9 的中断。	×	○	×	×	×
由内部定时器进行的中断	I28 ～ 31、I49	由 CPU 模块内部定时器进行的恒定周期中断。	△*2	△*3	△*2	△*2	△*2
由于出错而发生的中断*5	I32 ～ 41	由于顺控程序运算的继续运行出错而发生的中断。	×	△*4	△*4	○	○
多 CPU 间同步中断	I45	用于进行与运动控制运算周期同步控制的恒定周期中断。	×	×	×	×	○*6
智能功能模块中断	I50 ～ 255*7	来自智能功能模块*8 的中断。	○	○	○	○	○

○: 可使用　△: 有部分中断指针不能使用　×: 不可使用

*1 : 关于可使用的中断模块，请参照以下手册。
(☞ QCPU 用户手册（硬件设计 / 维护点检篇))
*2 : 中断指针 I49（高速中断功能专用指针）只能在 QnHCPU 中使用。
不能在其它 CPU 模块中使用。
*3 : Q02CPU 中不能使用中断指针 I49（高速中断功能专用指针）。
*4 : 中断指针 I40、I41 只能在冗余 CPU 中使用。
*5 : 可编程控制器参数的可编程控制器 RAS 设定中，关于可设定出错时的运行模式的错误，只有在设定为 “继续运行” 时执行中断。
*6 : 在与多 CPU 间高速主基板 (Q3 □ DB)、多 CPU 间高速通信兼容的 QnUCPU 以及使用了运动控制器的多 CPU 系统中可以使用。在 QnUCPU 的单 CPU 系统中不能使用。
*7 : 基本模式 QCPU 为 I50 ～ 127。
*8 : 相应模块有：串行通信模块、MELSECNET/H 网络模块、以太网接口模块、高速计数器模块等。
有关详细内容请参照各模块的手册。
*9 : 相应模块中有智能通讯模块。
详细内容请参阅各模块的手册。

要 点

使用智能功能模块中断 (☞ 6.23 节）时，需要在可编程控制器参数的可编程控制器系统设定中进行智能功能模块的设定（中断指针设定）。

高性能
注 9.35

在高性能模式 QCPU 中使用中断模块时，请先确认 CPU 模块以及 GX Developer 的版本。
(☞ 附录 4.2)

9.10.1 中断指针编号与中断因子一览

各 CPU 模块的中断指针编号与中断因子的一览如下所示。

(1) 基本模式 QCPU

表 9.19 中断指针编号与中断因子一览表（基本模式 QCPU）

I 编号	中断因子		优先顺序
I0	QI60 的中断	第 1 点	5
I1		第 2 点	6
I2		第 3 点	7
I3		第 4 点	8
I4		第 5 点	9
I5		第 6 点	10
I6		第 7 点	11
I7		第 8 点	12
I8		第 9 点	13
I9		第 10 点	14
I10		第 11 点	15
I11		第 12 点	16
I12		第 13 点	17
I13		第 14 点	18
I14		第 15 点	19
I15		第 16 点	20
I16~27	禁止使用	—	—
I28	内部定时器的中断 *1	100ms	4
I29		40ms	3
I30		20ms	2
I31		10ms	1
I32~49	禁止使用	—	—
I50~127 *2*3	智能功能模块 /QI60 的中断	根据参数设定使用哪个智能功能模块 / 中断模块 (QI60)	21~98

* 1: 内部定时器的时限为缺省值。
通过可编程控制器参数的可编程控制器系统设定能够以 1ms 为单位在 2ms 到 1000ms 之间变更。
* 2: 使用智能功能模块中断时，需要通过可编程控制器参数的可编程控制器系统设定进行智能功能模块设定（中断指针设定）。（智能功能模块的中断请参照 6.23 节。）
* 3: I50~127 的优先顺序为，I50 最高（优先顺序 21），I127 最低（优先顺序 98）。

(2) 高性能模式 QCPU

表 9.20 中断指针编号与中断因子一览表（高性能模式 QCPU）

I 编号	中断因子		优先顺序	I 编号	中断因子		优先顺序
I0	QI60/A1SI61 的中断	第 1 点	220	I32*5	出错中断 *5*6	所有的停止出错	1
I1		第 2 点	221	I33		SINGLE PS DOWN *3 *4	
I2		第 3 点	222	I34		UNIT VERIFY ERR. FUSE BREAK OFF SP. UNIT ERROR MULTI CPU ERROR	
I3		第 4 点	223				
I4		第 5 点	224				
I5		第 6 点	225	I35		OPERATION ERROR SFCP OPE. ERROR SFCP EXE. ERROR EX. POWER OFF	
I6		第 7 点	226				
I7		第 8 点	227				
I8		第 9 点	228				
I9		第 10 点	229	I36		ICM. OPE ERROR FILE OPE. ERROR	
I10		第 11 点	230				
I11		第 12 点	231	I37		空	
I12		第 13 点	232	I38		PRG. TIME OVER	
I13		第 14 点	233	I39		执行 CHK 指令 报警器检测	
I14		第 15 点	234				
I15		第 16 点	235	I40 ～ 48	--	Empty	--
I16	顺控启动发生模块的中断 *1	第 1 点	208	I49	内部定时器的中断	0.2ms ～ 1.0ms*7	*10
I17		第 2 点	209				
I18		第 3 点	210				
I19		第 4 点	211	I50 ～ 255	智能功能模块中断 *8*9	根据参数设定使用哪个智能功能模块	2 ～ 207
I20		第 5 点	212				
I21		第 6 点	213				
I22		第 7 点	214				
I23		第 8 点	215				
I24		第 9 点	216				
I25		第 10 点	217				
I26		第 11 点	218				
I27		第 12 点	219				
I28	内部定时器的中断 *2	100ms	239				
I29		40ms	238				
I30		20ms	237				
I31		10ms	236				

*1: 关于第 1 个～ 12 个，在安装在基板上的顺控启动发生模块内，将靠近高性能模式 QCPU 的模块作为第 1 个按顺序进行分配。

*2: 内部定时器的时限是缺省值。
通过可编程控制器 参数的可编程控制器系统设定能够以 0.5ms 为单位在 0.5ms 到 1000ms 之间变更。

*3: 以序列号的高 5 位为 “07032”或以后的产品为对象。

*4: 多 CPU 系统配置时，所有的 CPU 模块的序列号的高 5 位为 “07032”或以后时，只能在 1 号机的 CPU 模块中使用。

*5: 关于出错中断中的 “I32(所有的停止出错)”，出错时进行 I32 的处理后，高性能模式 QCPU 将停止。

*6: 关于 I32 ～ 48，如果进行电源启动 / 高性能模式 QCPU 的复位操作，将进入禁止执行状态 (DI)。
使用 I32 ～ 48 时，应使用 IMASK 指令进入中断许可状态。
关于 IMASK 指令，请参照以下手册。
☞ QCPU(Q 模式)/QnACPU 编程手册 (公共指令篇)

*7: 内部定时器的时限是在可编程控制器参数的 “可编程控制器系统设定” → “系统中断设定” → “高速中断设定”中设定。
设定范围为 : 在 0.2ms ～ 1.0ms 之间以 0.1ms 为单位设定。

*8: 使用智能功能模块中断时，需要通过可编程控制器参数的可编程控制器系统设定进行智能功能模块设定 (中断指针设定)。(关于智能功能模块的中断，请参照 6.23 节。)

*9: I50 ～ 255 的优先顺序为 :I50 最高 (优先顺序 2)，I255 最低 (优先顺序 207)。

*10: 在可编程控制器参数中设置了 I49 时，不要执行其它中断程序 (10 ～ 48、150 ～ 255) 和恒定周期执行型程序。
如果执行了恒定周期型程序等，I49 的中断程序则不能按照设定的中断周期间隔执行。

(3) 过程 CPU

表 9.21 中断指针编号与中断因子一览表 (过程 CPU)

I 编号	中断因子		优先顺序	I 编号	中断因子		优先顺序
I0	QI60 的中断	第 1 点	208	I32*4	发生出错中断 *4*5	所有的停止出错	1
I1		第 2 点	209	I33		SINGLE PS DOWN *2*3	
I2		第 3 点	210	I34		UNIT VERIFY ERR. FUSE BREAK OFF SP. UNIT ERROR MULTI CPU ERROR	
I3		第 4 点	211				
I4		第 5 点	212				
I5		第 6 点	213	I35		OPERATION ERROR SFCP OPE. ERROR SFCP EXE. ERROR EX.POWER OFF	
I6		第 7 点	214				
I7		第 8 点	215				
I8		第 9 点	216				
I9		第 10 点	217	I36		ICM. OPE ERROR FILE OPE. ERROR	
I10		第 11 点	218				
I11		第 12 点	219	I37		Empty	
I12		第 13 点	220	I38		PRG. TIME OVER	
I13		第 14 点	221	I39		执行 CHK 指令 报警器检测	
I14		第 15 点	222				
I15		第 16 点	223	I40 ～ 49	--	Empty	--
I16	--	Empty	--	I50 ～ 255	智能功能模块中断 *6*7	根据参数设定使用哪个智能功能模块	2 ～ 207
I17							
I18							
I19							
I20							
I21							
I22							
I23							
I24							
I25							
I26							
I27							
I28	内部定时器的中断 *1	100ms	227				
I29		40ms	226				
I30		20ms	225				
I31		10ms	224				

*1: 内部定时器的时限为缺省值。
通过可编程控制器参数的可编程控制器系统设定能够以 0.5ms 为单位在 0.5ms 到 1000ms 之间变更。
*2: 以序列号的高 5 位为 “07032”或以后的产品为对象。
*3: 多 CPU 系统配置时，所有的 CPU 模块的序列号的高 5 位为 “07032”或以后时，只能在 1 号机的 CPU 模块中使用。
*4: 关于出错中断中的 “I32(所有的停止出错)”，出错时进行 I32 的处理后，过程 CPU 将停止。
*5: 关于 I32 ～ 48，如果进行了电源启动 / 过程 CPU 的复位操作，将进入禁止执行状态 (DI)。
使用 I32 ～ 48 时，应使用 IMASK 指令进入中断许可状态。
关于 IMASK 指令，请参照以下手册。
☞ QCPU(Q 模式)/QnACPU 编程手册 (公共指令篇)
*6: 使用智能功能模块中断时，需要通过可编程控制器参数的可编程控制器系统设置进行智能功能模块设定 (中断指针设定)。(关于智能功能模块的中断请参照 6.23 节。)
*7: I50 ～ 255 的优先顺序为，I50 最高 (优先顺序 2)，I255 最低 (优先顺序 207)。

(4) 冗余 CPU

表 9.22 中断指针编号与中断因子一览表（冗余 CPU）

I 编号	中断因子		优先顺序	I 编号	中断因子		优先顺序
I0	QI60 的中断	第 1 点	208	I32*2	发生出错中断 *2*3	所有的停止出错	1
I1		第 2 点	209	I33		SINGLE PS. DOWN	
I2		第 3 点	210	I34		UNIT VERIFY ERR. FUSE BREAK OFF SP. UNIT ERROR	
I3		第 4 点	211				
I4		第 5 点	212				
I5		第 6 点	213	I35		OPERATION ERROR SFCP OPE. ERROR SFCP EXE. ERROR EX.POWER OFF	
I6		第 7 点	214				
I7		第 8 点	215				
I8		第 9 点	216				
I9		第 10 点	217	I36		ICM. OPE ERROR FILE OPE. ERROR	
I10		第 11 点	218				
I11		第 12 点	219	I37		空	
I12		第 13 点	220	I38		PRG. TIME OVER	
I13		第 14 点	221	I39		CHK 指令执行 报警器检测	
I14		第 15 点	222				
I15		第 16 点	223	I40		CAN'T SWITCH	
I16	—	空	—	I41		STANDBY	
I17				I42~49	—	空	—
I18							
I19				I50~255	智能功能模块中断 *4*5	根据参数设定使用哪个智能功能模块	2~207
I20							
I21							
I22							
I23							
I24							
I25							
I26							
I27							
I28	内部定时器的中断 *1	100ms	227				
I29		40ms	226				
I30		20ms	225				
I31		10ms	224				

*1: 内部定时器的时限为缺省值。
通过可编程控制器参数的可编程控制器系统设置能够以 0.5ms 为单位在 0.5ms 到 1000ms 之间变更。
*2: 发生出错中断中的 “I32（停止出错全体）” 为，发生出错时进行 I32 的处理后，冗余 CPU 将停止。
*3: 关于 I32~48，进行电源启动 / 冗余 QCPU 的复位操作，将进入禁止执行状态 (DI)。
使用 I32~48 时，请使用 IMASK 指令进入中断许可状态。
关于 IMASK 指令，请参照以下手册。
☞ QCPU(Q 模式)/QnACPU 编程手册（公共指令篇）
*4: 使用智能功能模块中断时，需要通过可编程控制器参数的可编程控制器系统设定进行智能功能模块设定（中断指针设定）。（智能功能模块的中断请参照 6.23 节。）
*5: I50~255 的优先顺序为，I50 最高（优先顺序 2），I255 最低（优先顺序 207）。

(5) 通用型 QCPU

表 9.23 中断指针编号与中断因子一览表（通用型 QCPU）

I 编号	中断因子		优先顺序	I 编号	中断因子		优先顺序
I0	QI60 的中断	第 1 点	6	I32 ～ 44	--	空闲	--
I1		第 2 点	7				
I2		第 3 点	8				
I3		第 4 点	9				
I4		第 5 点	10				
I5		第 6 点	11				
I6		第 7 点	12				
I7		第 8 点	13				
I8		第 9 点	14				
I9		第 10 点	15				
I10		第 11 点	16				
I11		第 12 点	17				
I12		第 13 点	18				
I13		第 14 点	19				
I14		第 15 点	20				
I15		第 16 点	21	I45*2*4	多 CPU 间同步中断	0.88ms	1
I16～I27	--	空闲	--	I46 ～ 49	--	空闲	--
				I50 ～ 255	智能功能模块中断 *3	通过参数设定使用哪个智能功能模块	22 ～ 227
I28*4	内部定时器的中断 *1	100ms	5				
I29*4		40ms	4				
I30*4		20ms	3				
I31*4		10ms	2				

*1 : 内部定时器的时限显示为缺省值。
通过可编程控制器参数的可编程控制器系统设置能够以 0.5ms 为单位在 0.5ms 到 1000ms 之间变更。

*2 : 在使用了 QnUCPU 的多 CPU 系统中可以使用。

*3 : 在 I50 ～ 255 中 I50 的优先顺序最高（优先顺序 22），I255 的最低（优先顺序 227）。

*4 : 发生中断时，即使程序中不存在中断指针，也不会发生 CAN'T EXECUTE(I)（出错代码： 4220）。

9.11 其它软元件

9.11.1 SFC 块软元件 (BL)

SFC 块软元件是用来检查 SFC 程序的指定块是否有效的软元件。
关于 SFC 块软元件的使用方法，请参照以下手册。
☞QCPU(Q 模式)/QnACPU 编程手册 (SFC 篇)

基本 注 9.36

9.11.2 SFC 转移软元件 (TR)

通用 注 9.36

SFC 转移软元件是用来检查 SFC 程序的指定块的指定转移条件是否被指定为强制转移的软元件。
关于 SFC 转移软元件的使用方法，请参照以下手册。
☞QCPU(Q 模式)/QnACPU 编程手册 (SFC 篇)

9.11.3 网络 No. 指定软元件 (J)

(1) 网络 No. 指定软元件
网络 No. 指定软元件是指利用链接专用指令来指定网络 No. 时使用的软元件。

(2) 网络 No. 指定软元件的指定方法
网络 No. 指定软元件是通过链接专用指令按照图 9.86 所示进行指定。

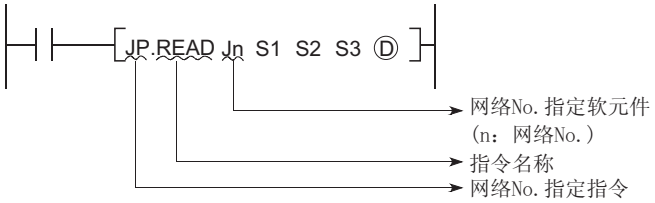


图 9.86 网络 No. 指定软元件的指定方法

备 注

关于链接专用用指令的详细内容，请参照以下手册。
☞MELSECNET/G 网络系统参考手册（控制网络篇）
☞Q 系列 MELSCNET/H 网络系统参考手册（可编程控制器网络篇）

基本 注 9.36　通用 注 9.36

在基本模式 QCPU、通用型 QCPU 中，不能使用 SFC 转移软元件 (TR)。

9.11.4 I/0 号指定软元件 (U)

(1) I/0 号指定软元件

I/0 号指定软元件是通过智能功能模块专用指令来指定 I/0 号时使用的软元件。

(2) I/0 号指定软元件的指定方法

I/0 号指定软元件通过智能功能模块专用的指令，按照图 9.87 所示进行指定。

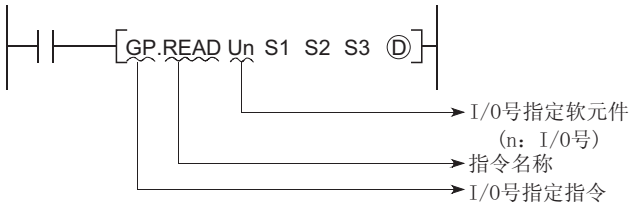


图 9.87 I/0 号指定软元件的指定方法

备 注

关于智能功能模块专用指令的详细内容，请参照所使用的智能功能模块的手册。

9.11.5 宏指令变量软元件 (VD)

(1) 宏指令变量软元件

宏指令变量软元件是在宏指令登录梯形图中使用的软元件。

如果在宏登录梯形图中使用 VD□，在使用宏指令时将转换为指定的软元件。

(2) 宏指令变量软元件的指定方法

宏指令变量软元件的指定方法为，在 GX Developer 的宏登录时作为宏指令登录的梯形图中使用的软元件中，指定作为 VD 的软元件。

顺控程序中使用宏指令时，指定与宏登录梯形图中使用的宏指令变量软元件的从小到大的顺序相对应的软元件。

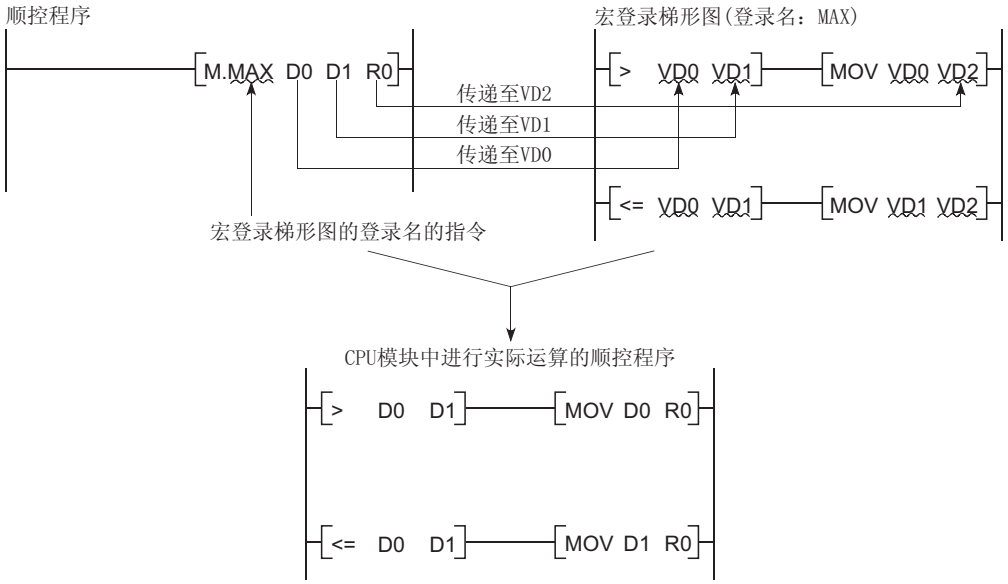


图 9.88 宏指令变量软元件的指定

要 点

1. 宏指令变量软元件可在 1 个宏登录梯形图中使用 VD0~9。
2. GX Developer 中，用读出模式显示程序时，也可以用宏指令形式显示。
   （显示的切换为：[显示]→[宏指令形式显示]。）

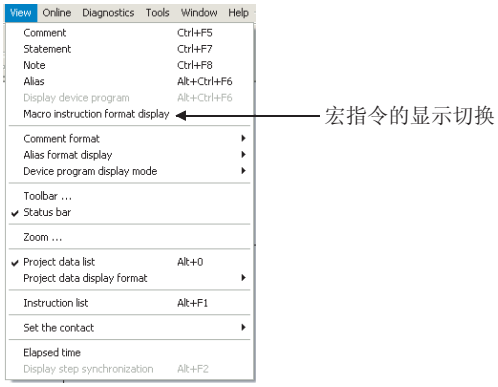


图 9.89 宏指令的显示切换设定

9.12 常数

9.12.1 10 进制常数 (K)

(1) 10 进制常数

10 进制常数是指在顺控程序中指定 10 进制数据的软元件。

在顺控程序中，以 K□（例 :K1234）进行指定。

CPU 模块内部以 2 进制数 (BIN) 储存。(☞3.9.1 项)

(2) 指定范围

10 进制常数的指定范围如下。

- 使用字数据 (16 位) 时 ..... K-32768~32767
- 使用 2 字数据 (32 位) 时 ..... K-2147483648~2147483647

要 点

最高位为符号位。

9.12.2 16 进制常数 (H)

(1) 16 进制常数

16 进制常数是在顺控程序中指定 16 进制数据或 BCD 数据的软元件。

(用 BCD 进行数据指定时，16 进制的各位以 0~9 进行指定。)

顺控程序中，以 H□（例：H1234）进行指定。(☞3.9.2 项)

(2) 指定范围

16 进制常数的设定范围如下所示。

- 使用字数据 (16 位) 时 ..... H0~FFFF
  (BCD 数据则为 H0~9999)
- 使用 2 字数据 (32 位) 时 ..... H0~FFFFFFFF
  (BCD 数据时则为 H0~99999999)

基本 注 9.37

9.12.3 实数 (E)

(1) 实数

实数是指，在顺控程序中指定实数的软元件。

在顺控程序中，以 E□（例 :E1.234）进行指定。（☞3.9.4 项）

图 9.90 实数的指定

(2) 指定范围

(a) 实数的设定范围

实数的设置范围如下所示：

- 单精度浮动小数点数据的情况

  $-2^{-128} < $ 软元件 $\leqq -2^{-126}$ ，0 ，$2^{-126} \leqq$ 软元件 $< 2^{128}$

- 双精度浮动小数点数据的情况 注 9.39

  $-2^{1024} < $ 软元件 $\leqq -2^{-1022}$，0 ，$2^{-1022} \leqq$ 软元件 $< 2^{1024}$

基本 高性能 过程
注 9.38 注 9.38 注 9.38

冗余 注 9.38

(b) 上溢及下溢时的动作

运算时发生了上溢或下溢时的 CPU 模块的动作如表 9.24 所示。

表 9.24 上溢及下溢时的动作

CPU 模块	上溢时的动作	下溢时的动作
基本型 QCPU	发生 4100 错误	发生 4100 错误
高性能模式 QCPU	发生 4100 错误	发生 4100 错误
过程控制 CPU	发生 4100 错误	发生 4100 错误
冗余 CPU	发生 4100 错误	不发生错误变为 0
通用型 QCPU	发生 4141 错误	不发生错误变为 0

基本 注 9.37

在基本型 QCPU 中使用实数运算功能时，应确认 CPU 模块及 GX Developer 的版本（☞附录 4.1）

基本 高性能 过程
注 9.38 注 9.38 注 9.38

冗余 注 9.38

在基本模式 QCPU、高性能模式 QCPU、过程 CPU、冗余 CPU 中，不能使用双精度浮动小数点数据。

(c) 输入了特殊值 *1 时的动作

输入数据以特殊值进行了运算时的动作如下表所示：

表 9.25 输入了特殊值时的动作

CPU 模块	输入了特殊值时的动作
基本型 QCPU	发生 4100 错误
高性能模式 QCPU	发生 4100 错误 *2
过程控制 CPU	发生 4100 错误
冗余 CPU	发生 4100 错误
通用型 QCPU	发生 4140 错误

*1 ： 特殊值是指，“-0”、非规格化数、非数、±∞。
*2 ： 仅在内部运算被设定为单精度时发生出错。

(3) 指定方法

在顺控程序中，可以将实数指定为普通表示及指数表示 2 种方式。

- 普通表示 ..... 按设定的数值直接指定。
  例如将 10.2345 指定为 E10.2345。
- 指数表示 ..... 将设置的数值以（数值）× 10^n 的形式指定。
  例如将 1234 指定为 E1.234+3。*1

*1: E1.234+3 的 +3 表示 10^3。

基本
注 9.39

9.12.4 字符串（“ ”）

(1) 字符串常数

字符串常数是指在顺控程序中指定字符串的软元件。
用“ ”包围的半角字符（例：“ABCD1234”）进行指定。

(2) 可使用的字符

字符串可使用 JIS8 编码。
CPU 模块中需区分大小写。

(3) 指定字符数

字符串的单位为，从指定文字开始到 NUL 编码（00H）为止。
但通过使用了字符串的指令（$MOV 指令等）能够指定的字符串最大为 32 个字符。

基本
注 9.39

在基本模式 QCPU 中，只能在以下指令中使用字符串 :$MOV、STR、DSTR、VAL、DVAL、ESTR、EVAL。

9.13 软元件的方便用法

基本
注 9.40
CPU 模块中执行多个程序时，通过内部用户软元件的局部软元件指定，能够使各个程序独立。注 9.40

基本
注 9.41
9.13.1 全局软元件与局部软元件

在 CPU 模块中，可储存并执行多个程序。
CPU 模块的各软元件分成两类：一类是在多个执行的程序中可共享的全局软元件，另一类是各程序中可作为独立的软元件使用的局部软元件。

(1) 全局软元件
全局软元件是指 CPU 模块中执行的程序可共享使用的软元件。
全局软元件储存在 CPU 模块的软元件内存中，全部程序可使用同一数据。

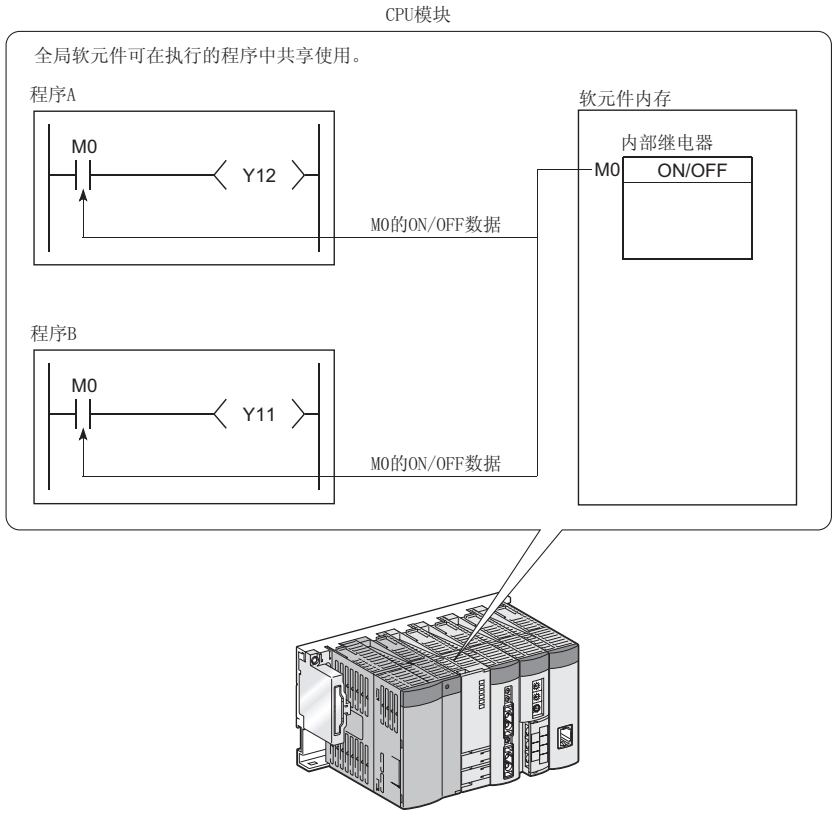


图 9.91 全局软元件的使用

基本
注 9.40
在基本模式 QCPU 中不能执行多个程序，因此，不能使用通过局部软元件指定使各程序独立化的功能。

基本
注 9.41
在基本模式 QCPU 中不能执行多个程序，因此，不存在局部软元件和全局软元件的区别。
使用基本模式 QCPU 时，本项内容不适用。

要 点

1. 未进行局部软元件（☞本项 (2)）设定的软元件全部为全局软元件。
2. 执行多个程序时，需要事先决定全部程序共享的范围或者各程序单独使用的范围（☞本项 (2)）。

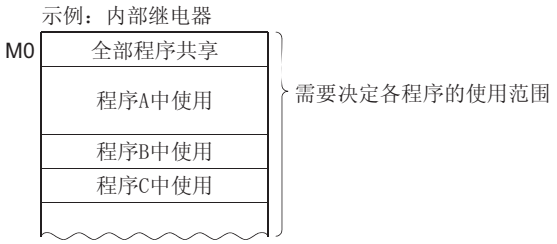


图 9.92 软元件的使用范围的决定

(2) 局部软元件

局部软元件是指在各程序中可独立使用的软元件。
使用局部软元件后，在执行多个独立的程序时，编程时可以无需理会其它程序。
但局部软元件数据只能储存在标准 RAM 和存储卡 (SRAM) 中。

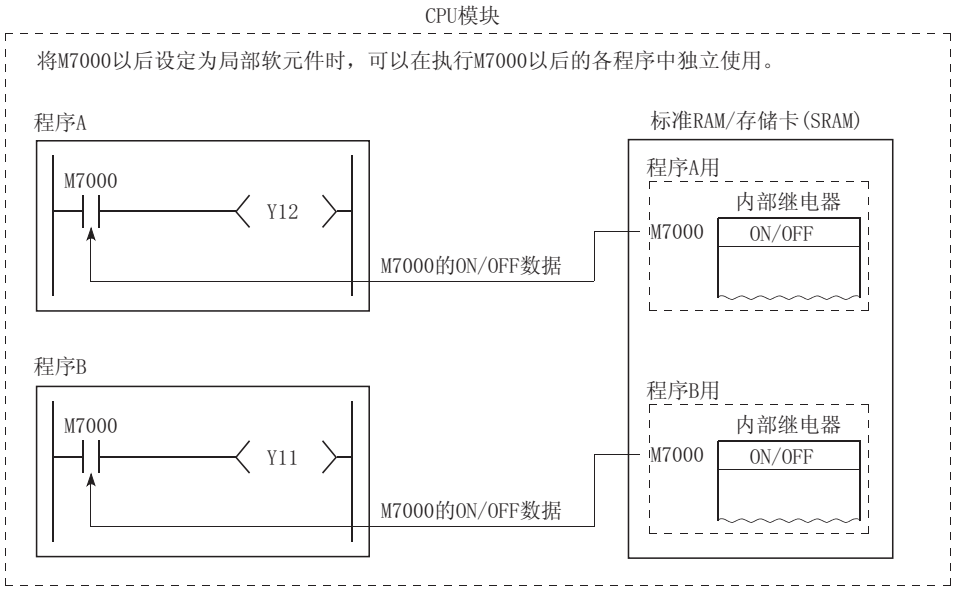


图 9.93 局部软元件的使用

(a) 可作为局部软元件使用的软元件

可作为局部软元件使用的软元件如下。

- 内部继电器 (M)
- 变址继电器 (V)
- 定时器 (T, ST)
- 计数器 (C)
- 数据寄存器 (D)

(b) 局部软元件文件的退避与返回

在使用了局部软元件的程序中，程序执行后，标准 RAM 或者存储卡 (SRAM) 的局部软元件文件的数据与 CPU 模块的软元件内存的数据将进行交换。
因此，扫描时间仅延时数据的交换时间。

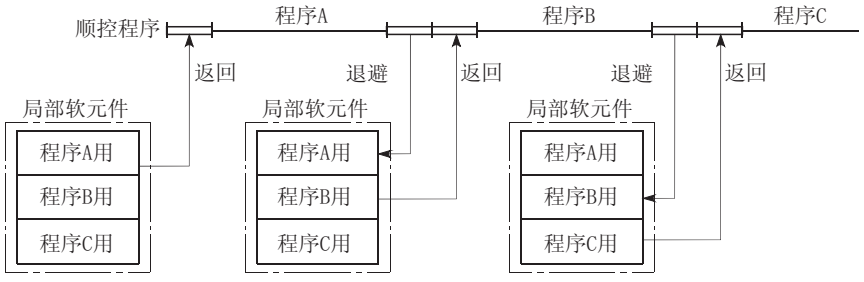


图 9.94 局部软元件文件的退避与返回

要 点

也有的指令不能指定局部软元件。
详细内容请参照各指令的编程手册的可使用的软元件。

备 注

关于局部软元件中可使用的软元件的字数的思路，请参照 9.2 节。

(c) 使用局部软元件时的设定

1) 设定作为局部软元件使用的软元件的范围

使用局部软元件时，通过可编程控制器参数的软元件设定来设定作为局部软元件使用的范围。

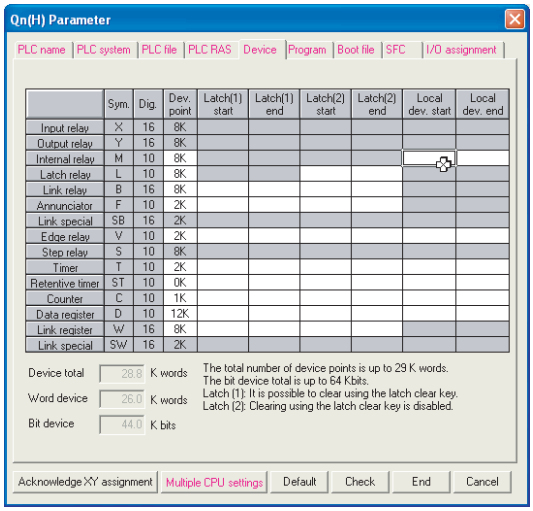


图 9.95 软元件设定

但设定为局部软元件范围的软元件，在全部程序中通用，各程序不能变更其设定范围。

例如将 M0~100 设定为局部软元件时，在使用局部软元件的程序中，M0~100 将成为局部软元件。

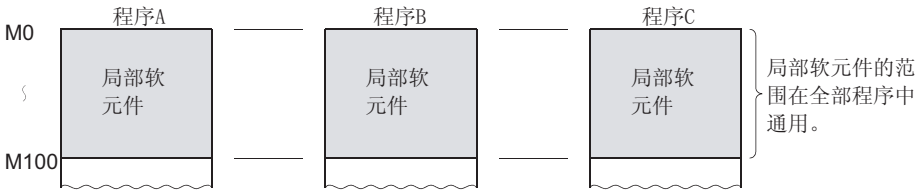


图 9.96 局部软元件的范围

2) 储存局部软元件的驱动器和文件名的设定

设定好作为局部软元件使用的软元件范围后，通过可编程控制器参数的可编程控制器文件设定来设定储存局部软元件用的文件的对象内存和文件名。

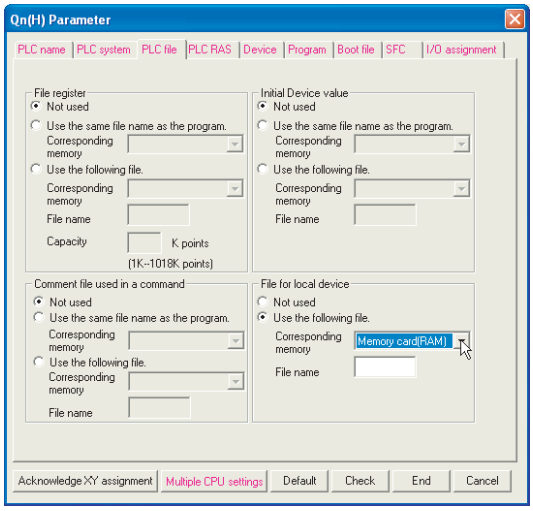


图 9.97 可编程控制器文件设定

3) 设定的内容的写入

将上页 1)、2) 中设定的内容写入 CPU 模块。

通过 GX Developer 的（可编程控制器写入）进行写入。

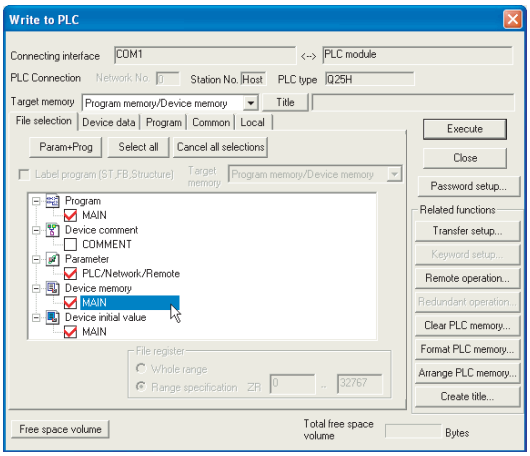


图 9.98 软元件内存的写入

向 CPU 模块写入时，可选择是否使用在可编程控制器参数的可编程控制器文件设定中设定的局部软元件。

如果选择不使用局部软元件，将继续使用以前执行的程序的局部软元件。

存储卡的局部软元件与 CPU 模块的软元件内存不进行交换。

执行程序 A、B、C 时，如果选择为在程序 B 中不使用局部软元件，所使用的局部软元件的情况如下所示：

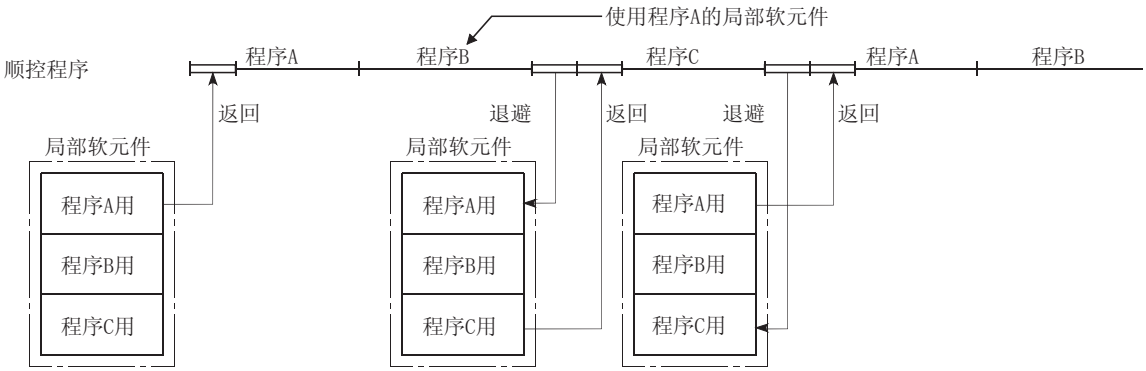


图 9.99 根据设置的不同局部软元件文件的退避与返回的区别

要 点

1. 在将采样追踪文件存储在标准 RAM 中的状态下，如果对标准 RAM 内的局部软元件的容量进行了更改设置，则采样追踪文件将被清除。
   如果希望保留追踪结果，可以按以下步骤将追踪结果保存到个人计算机中。
   - 通过采样追踪执行画面的 [Trace result PLC read(追踪结果可编程控制器读取)] 按钮，将追踪结果读取到个人计算机中。(☞本项 (2))
   - 点击 [Trace result(追踪结果)] 按钮，显示追踪结果。
   - 点击 [Create CSV file(创建 CSV 文件)] 按钮，将追踪结果以 CSV 格式保存。
2. 没有进行局部软元件设定的软元件全部为全局软元件。

(d) 副程序存储目标文件的局部软元件的使用

执行副程序时，可使用副程序的存储目标文件的局部软元件。

副程序的存储目标文件的局部软元件的使用，是通过特殊继电器 (SM776) 的 ON/OFF 进行设定。

表 9.26 利用特殊继电器 (SM776) 的 ON/OFF 来切换局部软元件

SM776	动作
OFF	利用副程序的调用源的文件的局部软元件进行运算。
ON	利用副程序中储存的文件的局部软元件进行运算。

1) SM776:OFF 时的动作

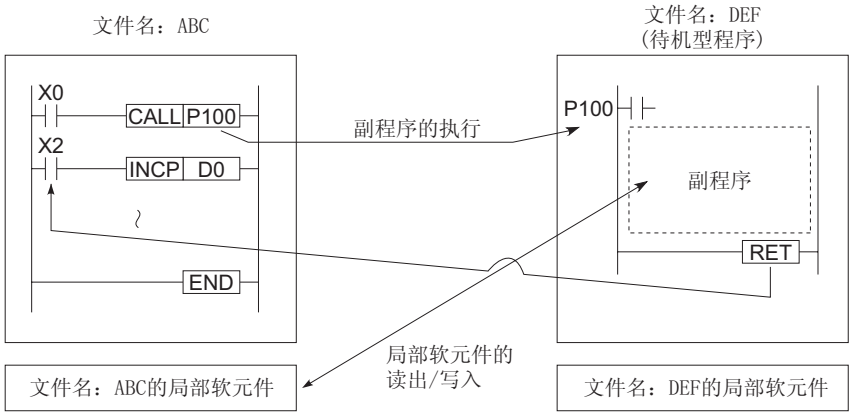


图 9.100 SM776 为 OFF

2) SM776:ON 时的动作

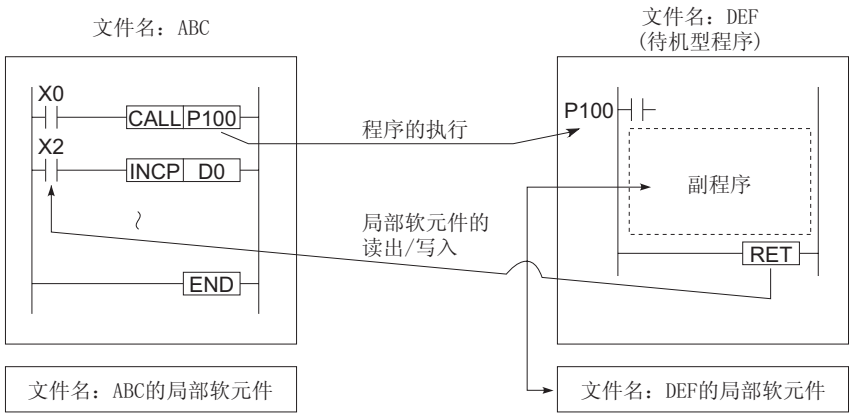


图 9.101 SM776 为 ON 时

3）注意事项

- SM776 为 ON 时，调用副程序时将读出局部软元件数据，RET 指令执行后将进行局部软元件数据的退避。
  因此 SM776 为 ON 时，执行 1 次副程序后扫描时间将延长。
- SM776 的 ON/OFF 设定是以 CPU 模块为单位的。
  不能以文件为单位进行设定。
- 在顺控程序的执行过程中变更了 SM776 的 ON/OFF 时，根据变更后的信息进行控制。

备 注

关于 SM776 的详细内容，请参照本手册的附录 1。

(e) 中断程序 / 恒定周期执行型程序执行时的局部软元件

中断程序 / 恒定周期执行型程序执行时，可使用中断程序 / 恒定周期执行型程序的存储目标文件的局部软元件。

中断程序 / 恒定周期执行型程序的存储目标文件的局部软元件的使用，是通过特殊继电器 SM777 的 ON/OFF 进行设定。

表 9.27 利用特殊继电器 (SM777) 的 ON/OFF 切换局部软元件

SM777	动作
OFF	利用在中断程序 / 恒定周期执行型程序执行之前执行的文件的本地软元件进行运算。
ON	利用中断程序 / 恒定周期执行型程序中储存的文件的本地软元件进行运算。

1) SM777:OFF 时的动作

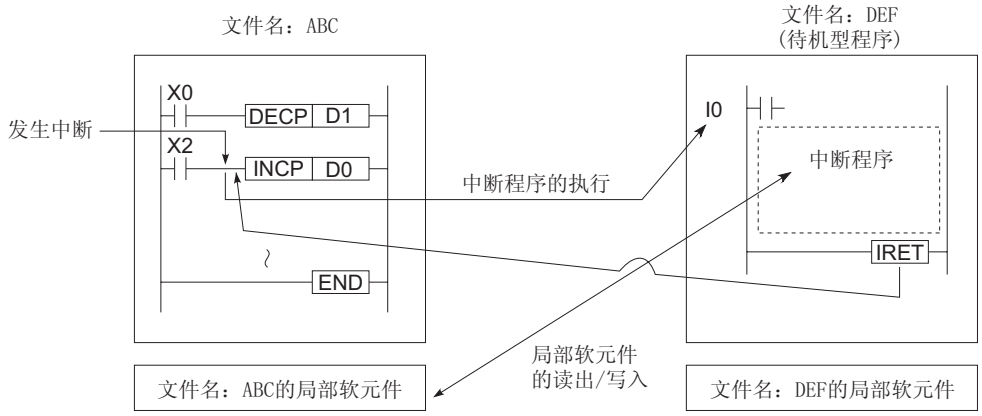


图 9.102 SM777 为 OFF 时

2) SM777:ON 时的动作

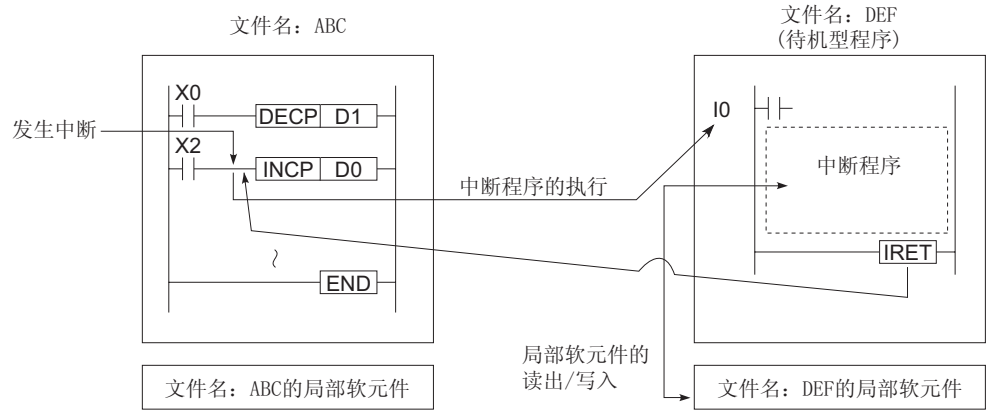


图 9.103 SM777 为 ON 时

3) 注意事项

- SM777 为 ON 时，在中断程序 / 恒定周期执行型程序的执行之前将读出局部软元件数据，IRET 指令执行后将进行局部软元件数据的退避。
  因此 SM777 为 ON 时，执行 1 次中断程序 / 恒定周期执行型程序后扫描时间将延长。
- SM777 的 ON/OFF 设定是以 CPU 模块为单位的。不能以文件为单位进行设定。
- 在顺控程序的执行过程中变更了 SM777 的 ON/OFF 时，根据变更后的信息进行控制。

备 注

关于 SM777 的详细内容，请参照本手册的附录 1。

(f) 局部软元件数据的清除

局部软元件的数据可通过以下任一方法进行清除。

- 可编程控制器的电源 OFF→ON 时或者 CPU 模块复位时
- CPU 模块从 STOP 状态切换为 RUN 状态时

不能由 GX Developer 等清除局部软元件数据。

第 10 章　CPU 模块的处理时间

在本章中将就 CPU 模块的处理时间进行说明。

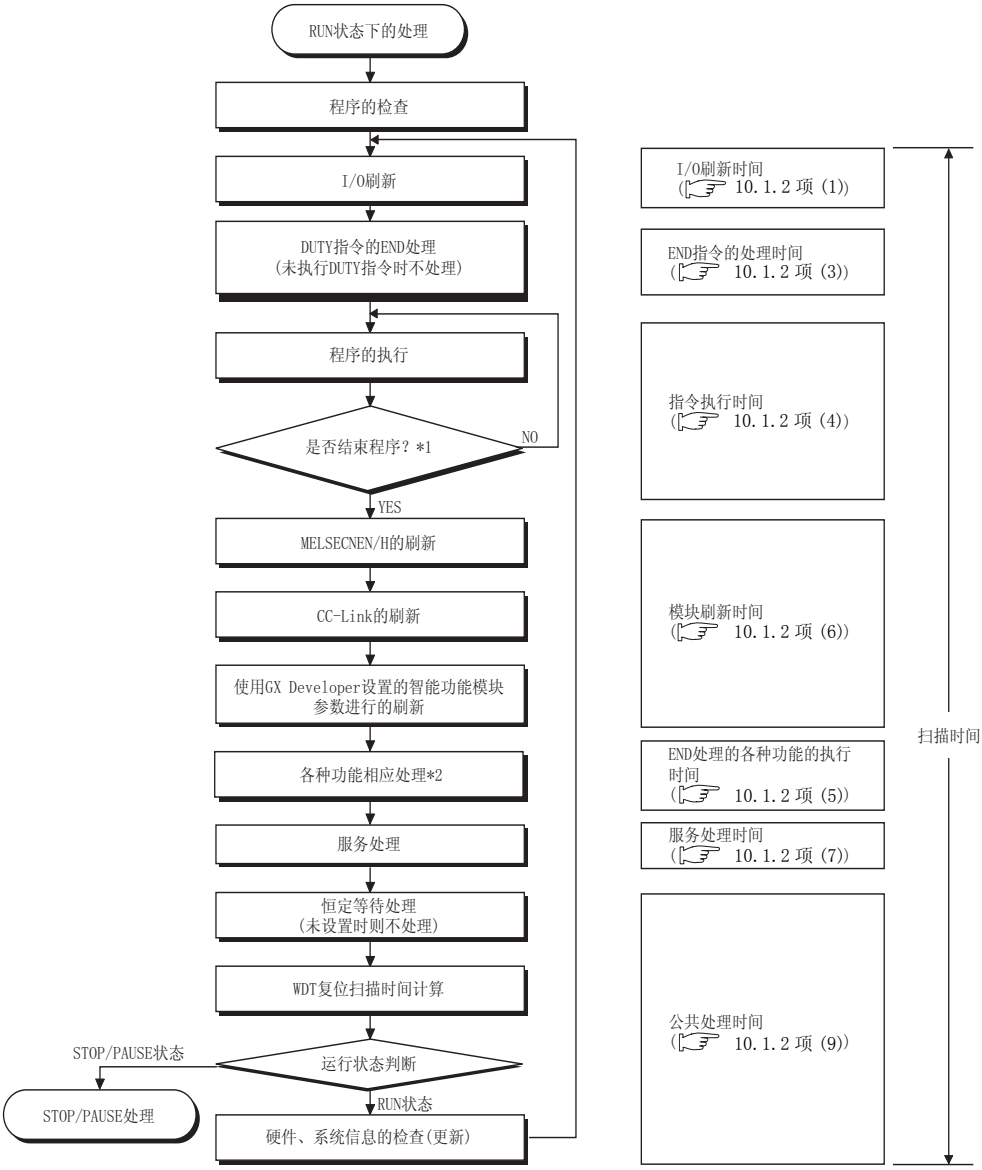
10.1　扫描时间

就 CPU 模块的扫描时间的构成和处理时间进行说明。

10.1.1　扫描时间的构成

说明 CPU 模块的扫描时间的构成。
CPU 模块在 RUN 状态下将循环进行以下处理。

(1) 基本模式 QCPU 的扫描时间的构成

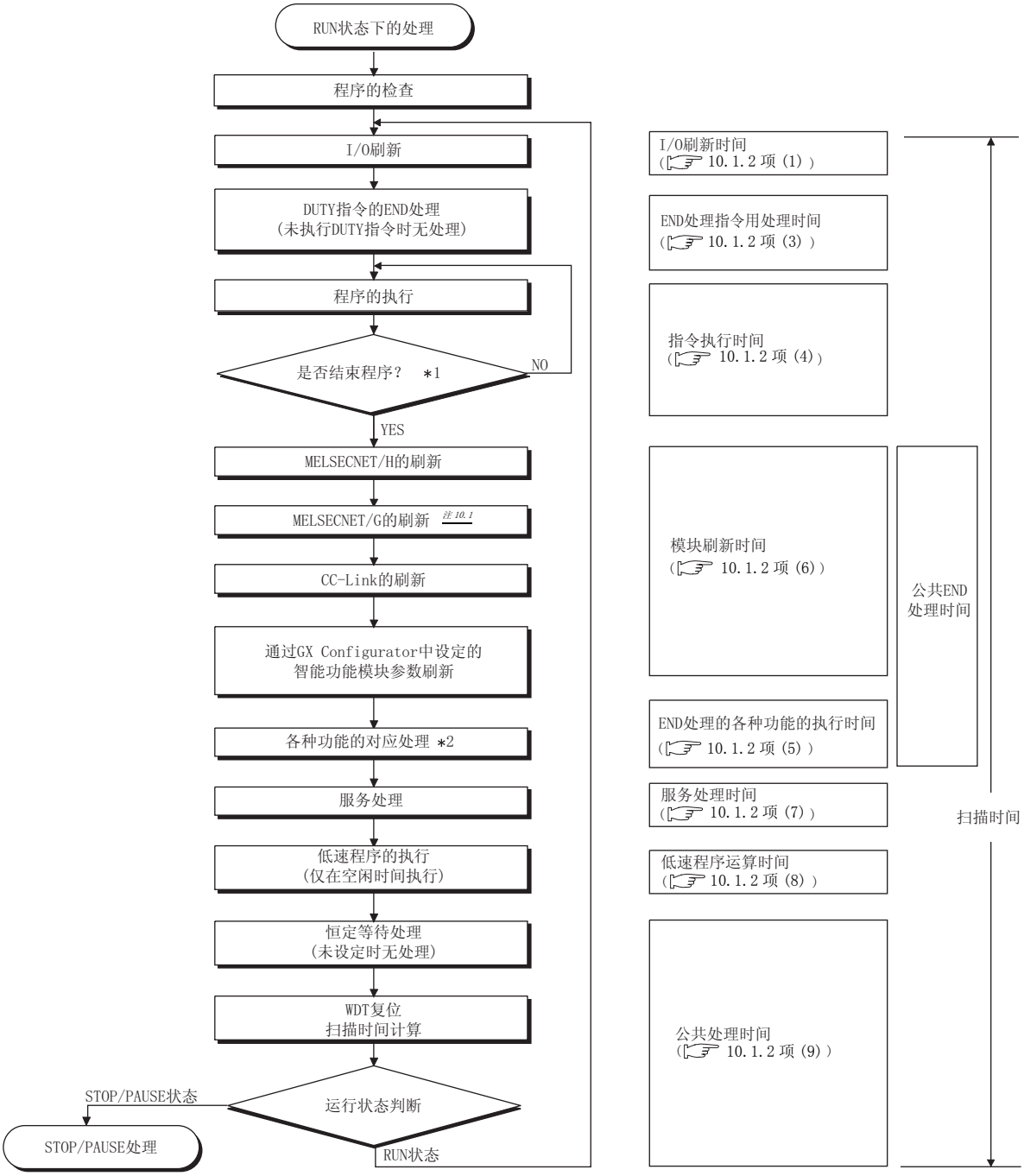


*1:　结束程序是指，表示 END、GOEND、FEND、STOP 指令的执行时机。
*2:　表示日历更新、出错解除。

图 10.1 基本模式 QCPU 的扫描时间的构成

(2) 高性能模式 QCPU 的扫描时间的构成

高性能
注 10.1



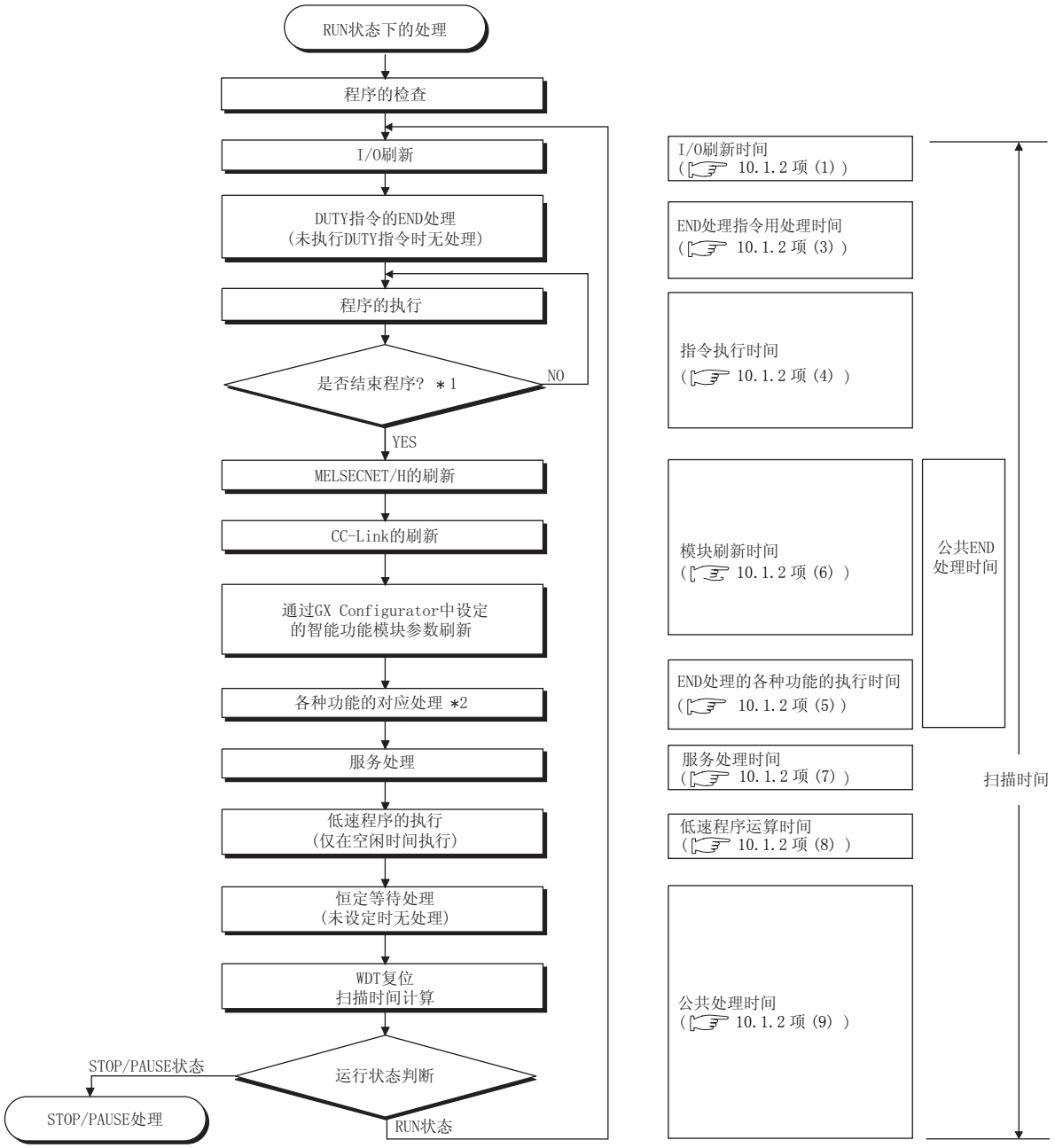
*1 : 结束程序时，将显示 END、GOEND、FEND、STOP 指令的执行时间。
*2 : 表示日历更新、错误解除。

图 10.2 高性能模式 QCPU 的扫描时间的构成

高性能
注 10.1

在高性能模式 QCPU 中使用 MELSECNET/G 时，应确认 CPU 模块及 GX Developer 的版本。
(☞ 附录 4.2)

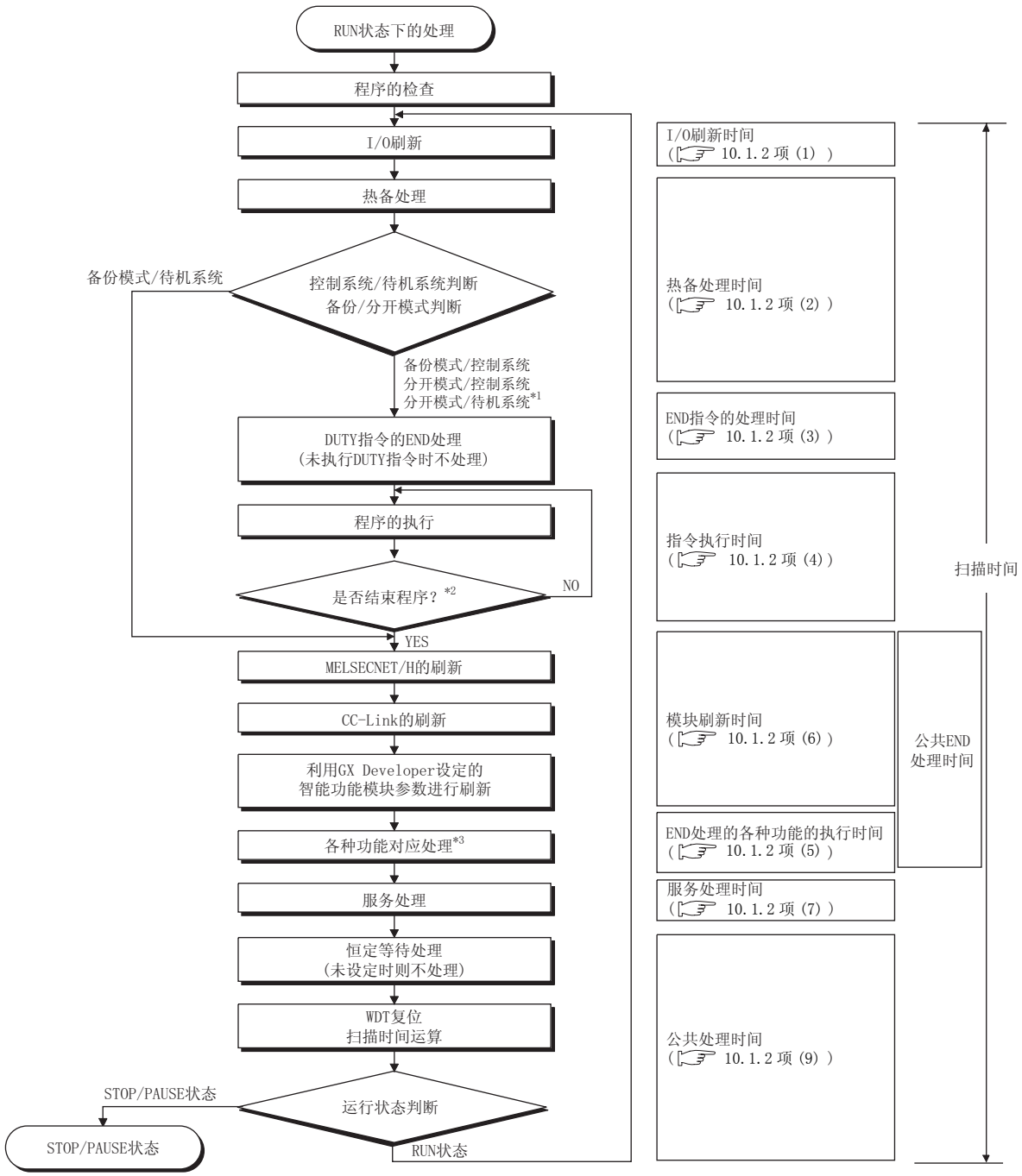
(3) 过程 CPU 的扫描时间的构成



*1 : 结束程序时，将显示 END、GOEND、FEND、STOP 指令的执行时间。
*2 : 表示日历更新、错误解除。

图 10.3 过程 CPU 的扫描时间的构成

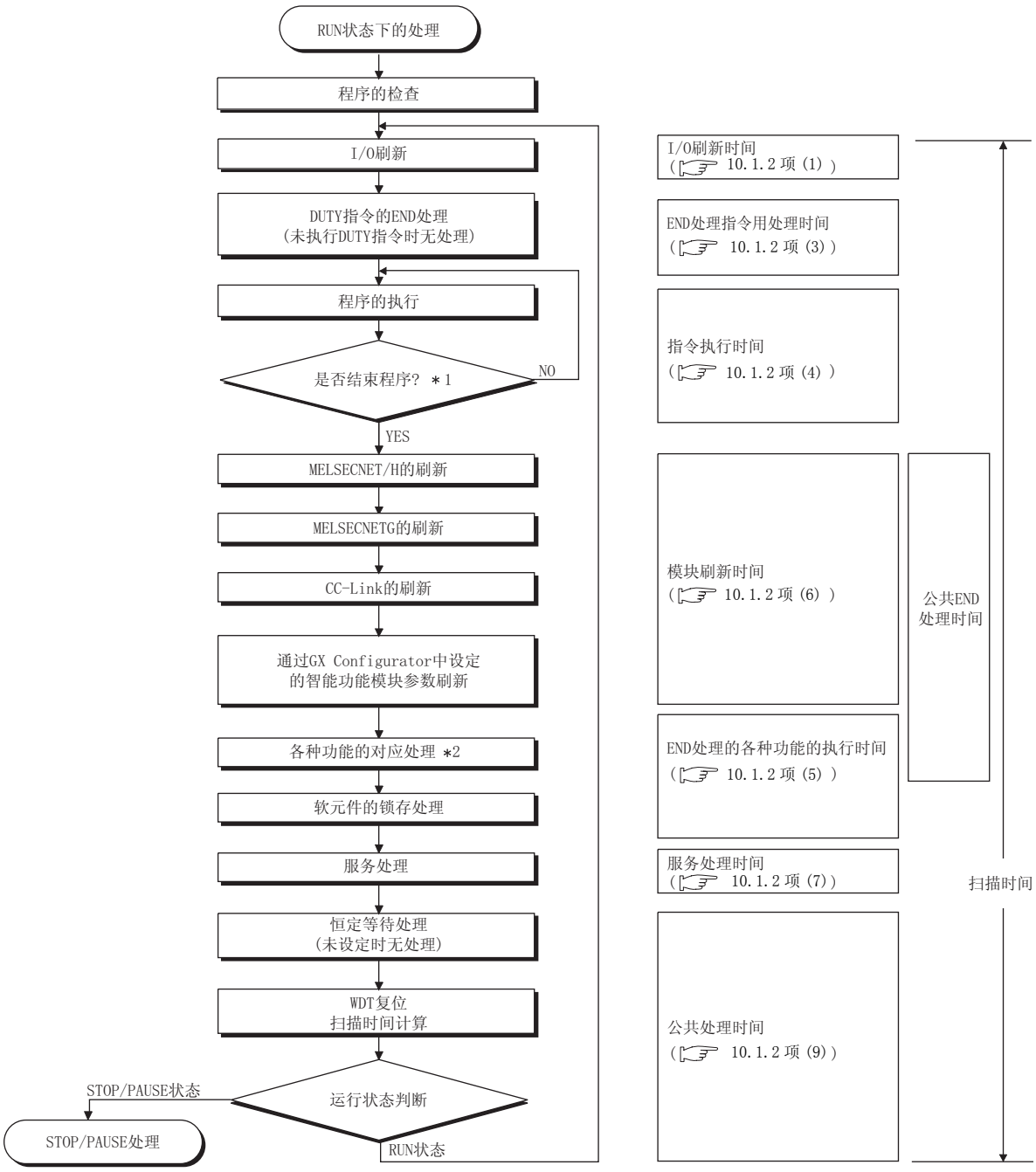
(4) 冗余 CPU 的扫描时间的构成



*1: 从备份模式变更为独立模式后，待机系统 (RUN LED 闪烁) 不执行程序。
*2: 结束程序时，将显示 END, GOEND, FEND, STOP 指令的执行时间。
*3: 表示日历更新、程序内存检查处理和错误解除。

图 10.4 冗余 CPU 的扫描时间的构成

(5) 通用型 QCPU 的扫描时间的构成



*1 : 结束程序时，将显示 END、GOEND、FEND、STOP 指令的执行时间。
*2 : 表示日历更新、错误解除。

图 10.5 通用型 QCPU 的扫描时间的构成

10.1.2 扫描时间的相关因素的处理时间

扫描时间是指 10.1.1 项中所示的处理以及执行时间的合计。
在本项中将就 10.1.1 项中所示的处理以及执行时间的计算方法进行说明。

(1) I/O 刷新时间

I/O 刷新时间是指主基板、扩展基板上安装的以下模块的 I/O 数据的刷新时间。

- 输入模块
- 输出模块
- 智能功能模块（特殊功能模块）注 10.2

基本 过程 冗余
注 10.2 注 10.2 注 10.2

通用
注 10.2

■ 计算方法

I/O 刷新时间的计算公式如下。

(I/O 刷新时间)=(输入点数 /16) × N1+(输出点数 /16) × N2

关于 N1，N2，请参照表 10.1。

表 10.1 I/O 刷新时间

CPU 模块	N1					N2				
	Q3□B、Q3□SB、Q3□RB Q3□DB	Q5□B、Q6□B、Q6□RB	Q6□WRB	QA1S6 □B	QA6□B	Q3□B、Q3□SB、Q3□RB Q3□DB	Q5□B、Q6□B、Q6□RB	Q6□WRB	QA1S6□B	QA6□B
Q00JCPU	2.05μs	2.95μs	----	----	----	1.25μs	2.20μs	----	----	----
Q00CPU	2.00μs	2.75μs	----	----	----	1.20μs	2.05μs	----	----	----
Q01CPU	1.95μs	2.70μs	----	----	----	1.15s	2.00μs	----	----	----
Q02CPU	2.2μs	2.9μs	----	4.3μs	4.3μs	1.3μs	2.1μs	----	3.5μs	3.5μs
Q02HCPU Q06HCPU Q12HCPU Q25HCPU	1.7μs	2.4μs	----	3.7μs	3.7μs	1.3μs	2.1μs	----	3.5μs	3.5μs
Q12PHCPU Q25PHCPU	1.7μs	2.4μs	----	----	----	1.3μs	2.1μs	----	----	----
Q12PRHCPU Q25PRHCPU	1.7μs	2.4μs	2.4μs	----	----	1.3μs	2.1μs	2.1μs	----	----
Q02UCPU	1.5μs	2.3μs	----	----	----	1.0μs	1.8μs	----	----	----
Q03UDCPU、Q04UDHCPU、Q06UDHCPU	1.5μs	2.3μs	----	----	----	1.0μs	1.8μs	----	----	----

备 注

可使用的基板因 CPU 模块而异。
☞ QCPU 用户手册（硬件设计 / 维护点检篇）

基本 过程 冗余
注 10.2 注 10.2 注 10.2

通用
注 10.2

在基本模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 中，不能使用 AnS/A 系列对应的特殊功能模块。

(2) 热备处理时间

热备处理时间是指使用冗余 CPU 的热备功能时所需的时间。

关于热备处理时间，请参照以下手册。

☞ QnPRHCPU 用户手册（冗余 CPU 篇）

(3) END 指令处理的指令用处理时间

(a) END 处理及 DUTY 指令的执行

在执行 DUTY 指令期间，CPU 模块根据 END 处理的次数（即 DUTY 指令指定的扫描次数）对用户动态块（SM420 ～ 424、SM430 ～ 434注 10.3）进行 ON/OFF。在这种情况下，END 处理时间随着指定的 DUTY 指令的次数而变化。

基本 注 10.3

表 10.2 DUTY 指令的 END 处理时间

CPU 模块	END 处理时间	
	设定数为 1 时	设定数为 5 时
Q00JCPU	0.15ms	0.21ms
Q00CPU	0.14ms	0.19ms
Q01CPU	0.12ms	0.16ms
Q02CPU	0.02ms	0.02ms
Q02HCPU，Q06HCPU， Q12HCPU，Q25HCPU， Q12PHCPU，Q25PHCPU， Q12PRHCPU，Q25PRHCPU	0.01ms	0.01ms
Q02UCPU	0.0190ms	0.0200ms
Q03UDCPU	0.0043ms	0.0046ms
Q04UDHCPU、Q06UDHCPU	0.0041ms	0.0045ms

(4) 指令执行时间

指令执行时间是指在 CPU 模块中执行的程序中使用的各指令的处理时间的合计。

关于各指令的处理时间，请参照以下手册。

☞ QCPU(Q 模式)/QnACPU 编程手册（公共指令篇）

基本 注 10.3

在基本模式 QCPU 中，不能使用 SM430~434。

基本
注 10.4

(a) 中断程序 / 恒定周期执行型程序执行时的起始时间注 10.4

中断程序 / 恒定周期执行型程序具有起始时间。
此外，中断程序有启动前起始时间和程序结束起始时间。

指令执行时间计算时，请在上述指令执行时间的基础上加上如下所示的起始时间。

表 10.3 中断程序的启动前的起始时间 (B1)

CPU 模块	恒定周期中断 (I28 ～ 31) 处理		多 CPU 间同步中断 (I45)		来自于 QI60(I0 ～ 15) 的中断处理 *1/ 来自于智能功能模块的中断 (I50 ～ 127) 处理	
	无高速启动	有高速启动	无高速启动	有高速启动	无高速启动	有高速启动
Q00JCPU	175μs	150μs	----	----	350μs	325μs
Q00CPU	145μs	125μs	----	----	285μs	265μs
Q01CPU	135μs	120μs	----	----	270μs	255μs
Q02CPU	190μs	85μs	----	----	205μs	100μs
Q02HCPU、Q06HCPU、Q12HCPU、Q25HCPU、Q12PHCPU、Q25PHCPU	85μs	40μs	----	----	90μs	45μs
Q12PRHCPU、Q25PRHCPU(未连接扩展基板时)	85μs	40μs	----	----	90μs	45μs
Q12PRHCPU、Q25PRHCPU(连接了扩展基板时)	85μs	40μs	----	----	1090μs*2	1045μs*2
Q02UCPU	50μs	19μs	----	----	60μs	31μs
Q03UDCPU	47μs	17μs	46μs	16μs	54μs	22μs
Q04UDHCPU、Q06UDHCPU	46μs	16μs	44μs	14μs	52μs	22μs

*1: 是指主基板的插槽 0 上安装有 QI60 时的数值。
*2: 连接扩展基板时不能使用 QI60。表内的数值为进行来自于智能功能模块的中断处理时的系统开销时间。

基本
注 10.4

在基本模式 QCPU 中不能使用恒定周期执行型程序，因此无需考虑恒定周期执行型程序的起始时间。

表 10.4 中断程序的结束起始时间 (B2)

CPU 模块	无高速启动	有高速启动
Q00JCPU	175μs	150μs
Q00CPU	145μs	125μs
Q01CPU	135μs	120μs
Q02CPU	180μs	75μs
Q02HCPU，Q06HCPU，Q12HCPU，Q25HCPU，Q12PHCPU，Q25PHCPU，Q12PRHCPU，Q25PRHCPU	80μs	35μs
Q02UCPU	26μs	7μs
Q03UDCPU	26μs	7μs
Q04UDHCPU、Q06UDHCPU	26μs	7μs

基本

注 10.5

表 10.5 恒定周期执行型程序的起始时间*注 10.5*

CPU 模块	无高速启动	有高速启动
Q02CPU	380μs	230μs
Q02HCPU，Q06HCPU，Q12HCPU，Q25HCPU，Q12PHCPU，Q25PHCPU，Q12PRHCPU，Q25PRHCPU	165μs	100μs
Q02UCPU	73μs	25μs
Q03UDCPU	73μs	24μs
Q04UDHCPU、Q06UDHCPU	73μs	23μs

基本

注 10.5

在基本模式 QCPU 中不能使用用恒定周期执行型程序，因此无需考虑恒定周期执行类型程序的系统开销时间。

基本
注 10.6

1）可使用中断程序内的局部软元件时的总消耗时间 注 10.6

将 SM777（中断程序中的局部软元件的允许 / 禁止设定）设置为 ON，将中断程序内的局部软元件设定为可用状态时，表 10.6、表 10.7 中所示的时间被加到表 10.4 和表 10.5 中的总消耗时间中去。

表 10.6 使用标准 RAM 内的局部软元件文件时

CPU 模块	中断程序启动前（表 10.3）的起始时间的加算值	中断程序结束（表 10.4）的起始时间的加算值
Q02CPU	0.35 × N+0.05 ms	0.35 × N+0.05 ms
Q02HCPU、Q06HCPU、Q12HCPU、Q25HCPU，Q12PHCPU、Q25PHCPU，Q12PRHCPU、Q25PRHCPU	0.15 × N+0.03 ms	0.15 × N+0.03 ms
Q02UCPU	0.23 × N+0.15 ms	0.23 × N+0.02 ms
Q03UDCPU	0.23 × N+0.10 ms	0.23 × N+0.02 ms
Q04UDHCPU、Q06UDHCPU	0.10 × N+0.10 ms	0.10 × N+0.02 ms

N：局部软元件点数（单位：k 字）

表 10.7 使用 SRAM 卡内的局部软元件文件时

CPU 模块	中断程序启动前（表 10.3）的起始时间的加算值	中断程序结束（表 10.4）的起始时间的加算值
Q02CPU	1.15 × N+0.30 ms	1.15 × N+0.30 ms
Q02HCPU，Q06HCPU，Q12HCPU，Q25HCPU，Q12PHCPU，Q25PHCPU，Q12PRHCPU，Q25PRHCPU	0.85 × N+0.15 ms	0.85 × N+0.15 ms
Q02UCPU	0.41 × N+0.27 ms	0.41 × N+0.02 ms
Q03UDCPU	0.41 × N+0.18 ms	0.41 × N+0.02 ms
Q04UDHCPU、Q06UDHCPU	0.40 × N+0.18 ms	0.40 × N+0.02 ms

N：局部软元件点数（单位：k 字）

基本
注 10.6

在基本模式 QCPU 中不能执行多个程序，因此不存在全局软元件和局部软元件（☞ 9.13.1 项）的区别。
无需考虑使用局部软元件时的起始时间。

(5) END 处理的各种功能的执行时间

END 处理的各种功能的执行时间是指日历更新、程序内存检查处理、错误解除所需时间的合计。

(a) 日历更新处理时间

是指时钟数据设置请求 (SM210 从 OFF 变为 ON)，或者时钟数据读出请求 (SM213 为 ON) 发出后，通过 END 处理变更 / 读出时钟数据所需的时间。

表 10.8 日历更新处理时间

CPU 模块	END 处理时间	
	请求时钟数据设置时	有时钟数据读出请求
Q00JCPU	1.25 ms	0.04 ms
Q00CPU	0.99 ms	0.03 ms
Q01CPU	0.98 ms	0.02 ms
Q02CPU	0.26 ms	0.01 ms
Q02HCPU、Q06HCPU、Q12HCPU、Q25HCPU、Q12PHCPU、Q25PHCPU、Q12PRHCPU、Q25PRHCPU	0.11 ms	0.005 ms
Q02UCPU	0.044 ms	0.017 ms
Q03UDCPU、Q04UDHCPU、Q06UDHCPU	0.011 ms	0.004 ms

(b) 错误解除处理

是指 SM50(错误解除) 启动 (从 OFF 变为 ON) 时，SD50 中储存的继续运行错误的错误解除时间。

表 10.9 错误解除处理时间

CPU 模块	公共处理时间	
	报警器	其它错误
Q00JCPU	2.1 ms	2.0 ms
Q00CPU	1.75 ms	1.7 ms
Q01CPU	1.45 ms	1.35 ms
Q02CPU	1.15 ms	0.41 ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU	0.84 ms	0.21 ms
Q02UCPU	0.180 ms	0.175 ms
Q03UDCPU	0.068 ms	0.062 ms
Q04UDHCPU、Q06UDHCPU	0.065 ms	0.062 ms

基本 高性能
注 10.7 注 10.7

(c) 存储器检查处理时间 *注 10.7* *注 10.8*

是在 GX Developer 的可编程控制器参数中设为进行程序内存检查时的时间。
存储器检查所需时间

通用
UD
注 10.8

$$= \frac{\text{1扫描周期的检查容量(步)}^{*1}}{1024} \times 3.5\text{ms}$$

*1： 表示可编程控制器参数的可编程控制器 RAS 设定 (2) 中所设定的步。(☞ 8.1.2 项 (5))

基本 高性能
注 10.7 注 10.7

在基本模式 QCPU、高性能模式 QCPU 中，不能设定存储器检查，因此不需要考虑本处理时间。

通用
UD
注 10.8

在通用型 QCPU 中，由于不花费存储器检查的处理时间，因此不需要考虑本处理时间。

(6) 模块刷新时间

模块刷新时间是指 MELSECNET/H 以及 CC-Link 等的刷新时间的合计。

注 10.9

注 10.10 注 10.10 注 10.10

(a) MELSECNET/G 的刷新 *注 10.9* *注 10.10*

是指在 MELSECNET/G 网络模块的链接软元件与 CPU 模块的软元件之间进行数据刷新的时间。

关于 MELSECNET/G 的刷新时间，请参照以下手册。

☞ MELSECNET/G 网络系统参考手册（控制网络篇）

(b) MELSECNET/H 的刷新

是指 MELSECNET/H 网络模块的链接软元件与 CPU 模块的软元件之间刷新数据的时间。

关于 MELSECNET/H 的刷新时间，请参照以下手册。

☞ Q 系列 MELSCNET/H 网络系统参考手册（可编程控制器网络篇）

☞ Q 系列 MELSCNET/H 网络系统参考手册（远程 I/O 网络篇）

(c) CC-Link 的自动刷新

是指 CC-Link 的主 / 本地模块与 CPU 模块之间刷新数据的时间。

关于 CC-Link 的自动刷新时间，请参照以下手册。

☞ CC-Link 系统主 / 本地模块用户手册（详细篇）

注 10.9

在高性能模式 QCPU 中使用 MELSECNET/G 时，请确认 CPU 模块以及 GX Developer 的版本。
(☞ 附录 4.2)

注 10.10 注 10.10 注 10.10

在基本模式 QCPU、过程 CPU、冗余 CPU 中，不能使用 MELSECNET/G。

(d) 智能功能模块的自动刷新

是指智能功能模块的缓冲存储器与 CPU 模块的软元件之间刷新数据的时间。
自动刷新设定是通过智能功能模块通过实用程序包 (GX Configurator) 进行。

■ 计算方法

智能自动刷新时间的计算公式如下。

(刷新时间)=KN1+KN2 × (刷新点数)

KN1,KN2 采用表 10.10 中的数值。

表 10.10 主基板中安装有智能模块时

CPU 模块	KN1	KN2
Q00JCPU	115μs	55μs
Q00CPU	91μs	46μs
Q01CPU	85μs	41μs
Q02CPU	53μs	13μs
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU	27μs	6μs
Q02UCPU	23μs	6μs
Q03UDCPU	6μ	5μs
Q04UDHCPU、Q06UDHCPU	4μs	5μs

表 10.11 扩展基板中安装有智能模块时

CPU 模块	KN1	KN2
Q00JCPU	120μs	56μs
Q00CPU	92μs	48μs
Q01CPU	86μs	43μs
Q02CPU	61μs	15μs
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU	29μs	8μs
Q02UCPU	45μs	7μs
Q03UDCPU	7μ	6μs
Q04UDHCPU、Q06UDHCPU	5μs	6μs

(示例)

相对于模拟数字转换模块 (Q64AD) 的自动刷新点数为 4 点时 (安装在 Q01CPU 的主基板上)

0.249(ms)=0.085+0.041 × 4

(7) 服务处理时间

服务处理是指与 GX Developer 以及外围设备的通讯处理。

(a) 基本模式 QCPU 的情况

通过 GX Developer 进行监视时的处理时间如表 10.12 所示。

表 10.12 GX Developer 的监视处理时间

功能	与自站 CPU 模块的 RS-232 连接时			与其它站点连接时 *4		
	Q00JCPU	Q00CPU	Q01CPU	Q00JCPU	Q00CPU	Q01CPU
程序的可编程控制器读出 *1	1.6ms	1.3ms	1.2ms	2.3ms	1.9ms	1.8ms
软元件监视 *2	1.2ms	1.0ms	0.9ms	2.4ms	2.0ms	1.9ms
RUN 中写入 *3	1.0ms	1.0ms	1.0ms	1.9ms	1.6ms	1.5ms

*1: 从程序内存读出 8k 步的程序时的时间。
*2: 登录监视设定为 32 点时的时间。
*3: 添加了 100 步的梯形图时的时间。
*4: 经由 MELSECNET/H,Ethernet,CC-Link、串行口通讯模块进行存取时的时间。

- 与串行口通讯模块、Ethernet 模块的通讯

  与串行口通讯模块以及 Etnernet 模块的通讯所需时间。
  关于与各模块的通讯时间，请参照以下手册。
  ☞ Q 系列 MELSEC 通讯协议参考手册

(b) 高性能模式 QCPU、过程 CPU、冗余 CPU 的情况

通过 GX Developer 进行监视的处理时间如表 10.13、表 10.14 所示。

表 10.13 软元件登录监视中设定为数据寄存器 32 点时

CPU 模块	处理时间
Q02CPU	0.017ms
Q02HCPU，Q06HCPU，Q12HCPU，Q25HCPU， Q12PHCPU，Q25PHCPU，Q12PRHCPU，Q25PRHCPU	0.011ms

表 10.14 设定了监视条件时

CPU 模块	处理时间	
	指定步一致	指定软元件一致
Q02CPU	0.05ms	0.01ms
Q02HCPU，Q06HCPU，Q12HCPU，Q25HCPU， Q12PHCPU，Q25PHCPU，Q12PRHCPU，Q25PRHCPU	0.03ms	0.01ms

(c) 通用型 QCPU 的情况

通过 GX Developer 进行监视、程序读取时的处理时间如表 10.15 所示。

表 10.15 软元件登录监视、程序读取时的处理时间

CPU 模块	处理时间	
	软元件登录监视 （数据寄存器 32 点）	程序读取 （10k 步）
Q02CPU	0.65ms	1.60ms
Q03UDCPU	0.35ms	1.10ms
Q04UDHCPU、Q06UDHCPU	0.35ms	1.10ms

基本 冗余 通用
注10.11 注10.11 注 10.11

(8) 低速程序运算时间*注 10.11*

低速程序运算时间是指在 CPU 模块中执行的低速执行型程序中使用的各指令的处理时间的合计。

关于处理时间，与本项 (4) 相同。

基本 冗余 通用
注 10.11 注 10.11 注 10.11

在基本模式 QCPU、冗余 CPU、通用型 QCPU 中不能执行低速执行型程序，因此无需考虑低速程序运算时间。

(9) 公共处理时间

系统处理的 CPU 模块的公共处理时间。

公共处理时间为表 10.16 中的数值。

表 10.16 公共处理时间

CPU 模块	公共处理时间
Q00JCPU	0.66ms
Q00CPU	0.60ms
Q01CPU	0.52ms
Q02CPU	0.40ms
Q02HCPU、Q06HCPU、Q12HCPU、Q25HCPU、Q12PHCPU、Q25PHCPU	0.16ms
Q12PRHCPU、Q25PRHCPU	0.5ms
Q02UCPU	0.17ms
Q03UDCPU	0.12ms
Q04UDHCPU、Q06UDHCPU	0.09ms

(10) 多 CPU 间高速通信处理时间

多 CPU 间高速通信处理时间是指，使用多 CPU 间高速通信功能时多 CPU 间的数据传送所需时间。

关于多 CPU 间高速通信处理时间，请参照以下手册。

☞ QCPU 用户手册（多 CPU 系统篇）

基本 高性能 过程
注 10.12 注 10.12 注 10.12
冗余
注 10.12

(11)软元件数据的锁存处理时间*注 10.12*

在可编程控制器参数的软元件设定中，进行了锁存范围的设定的情况下 *1*2，扫描时间将延迟如下所示的时间。

表中的 N1、N2、N3 如下所示：

- N1：锁存指定的软元件类型数
  （锁存范围 (1) 与锁存范围 (2) 被作为其它的软元件类型计数。）
- N2：锁存指定的位软元件的点数
- N3：锁存指定的字软元件的点数

表 10.17 锁存处理的延迟时间

CPU 模块	处理时间
Q02UCPU	(4.0 × N1) + (0.12 × (N2 ÷ 16 + N3)) μs
Q03UDCPU	(3.0 × N1) + (0.12 × (N2 ÷ 16 + N3)) μs
Q04UDHCPU、Q06UDHCPU	(3.0 × N1) + (0.05 × (N2 ÷ 16 + N3)) μs

*1 ： 设定定时器 (T)、累计定时器 (ST)、计数器 (C) 的锁存范围时，每 1 点将占用字软元件 1 点及位软元件 2 点。

*2 ： 包括进行了锁存继电器 (L) 的点数设定时。

要 点

为了缩短扫描时间，应尽可能设定较少的锁存点数。

通过以下替代方法可以减少锁存点数。

- 将锁存数据移至文件寄存器中。
- 将更新频率较少的软元件数据使用 SP.DEVST 指令存储到标准 ROM 中。（标准 ROM 中存储的软元件数据可以通过 S(P).DEVLD 指令读取）

☞ QCPU(Q 模式)/QnACPU 编程手册（公共指令篇）

基本 高性能 过程
注 10.12 注 10.12 注 10.12
冗余
注 10.12

在基本模式 QCPU、高性能模式 QCPU、过程 CPU、冗余 CPU 中，由于不执行用于锁存的处理，因此无需理会本项 (11) 的内容。

10.1.3 扫描时间延长的原因

本项中的功能或操作即为 CPU 模块的扫描时间延时的原因。
使用本项所示的功能或操作时，请在 10.1.2 项中算出的数值的基础上加上本项所示的数值。

基本 注10.13

(1) 采样追踪*注 10.13*

如果执行采样追踪（☞ 6.14 节），将产生表 10.18 所示的处理时间。

表 10.18 设定了位软元件中内部继电器 50 点，字元件中数据寄存器 50 点作为采样追踪数据时的处理时间

CPU 模块		处理时间
标准 RAM	Q02CPU	0.16ms
	Q02HCPU、Q06HCPU、Q12HCPU、Q25HCPU、Q12PHCPU、Q25PHCPU、Q12PRHCPU、Q25PRHCPU	0.06ms
	Q02UCPU	0.15ms
	Q03UDCPU	0.07ms
	Q04UDHCPU、Q06UDHCPU	0.06ms
SRAM 卡	Q02CPU	0.24ms
	Q02HCPU、Q06HCPU、Q12HCPU、Q25HCPU、Q12PHCPU、Q25PHCPU、Q12PRHCPU、Q25PRHCPU	0.12ms
	Q02UCPU	0.16ms
	Q03UDCPU	0.08ms
	Q04UDHCPU、Q06UDHCPU	0.06ms

基本 注 10.13

在基本模式 QCPU 中不能使用采样追踪，因此无需考虑采样追踪的处理时间。

基本
注10.14

(2) 局部软元件的使用*注 10.14*

如果使用局部软元件，将产生表 10.19 所示的处理时间。

表 10.19 局部软元件设定：10k 点时的处理时间

CPU 模块		处理时间 *1
标准 RAM	Q02CPU	3.52 × n ms
	Q02HCPU、Q06HCPU、Q12HCPU、Q25HCPU、Q12PHCPU、Q25PHCPU、Q12PRHCPU、Q25PRHCPU	1.54 × n ms
	Q02UCPU	2.54 × n ms
	Q03UDCPU	2.40 × n ms
	Q04UDHCPU、Q06UDHCPU	1.25 × n ms
SRAM 卡	Q02CPU	10.73 × n ms
	Q02HCPU、Q06HCPU、Q12HCPU、Q25HCPU、Q12PHCPU、Q25PHCPU、Q12PRHCPU、Q25PRHCPU	8.16 × n ms
	Q02UCPU	4.39 × n ms
	Q03UDCPU	4.30 × n ms
	Q04UDHCPU、Q06UDHCPU	4.18 × n ms

*1: n 为程序文件数。

基本
注 10.14

在基本模式 QCPU 中不能使用局部软元件，因此无需考虑本页所记载的处理时间。

(a) 副程序内的局部软元件处于可用状态时的扫描时间延时

SM776(CALL 时的局部软元件的允许 / 禁止设定)ON，设定副程序内的局部软元件处于可用状态时，副程序的 1 次 CALL 的扫描时间延时如表 10.20、表 10.21 所示。

表 10.20 使用标准 RAM 内的局部软元件文件时

CPU 模块	同一文件内的副函数 调用时的扫描时间延长时间	别的文件内的副函数 调用时的扫描时间延长时间
Q02CPU	0.35 × N+0.05 ms	0.70 × N+0.10 ms
Q02HCPU，Q06HCPU，Q12HCPU， Q25HCPU，Q12PHCPU， Q25PHCPU，Q12PRHCPU，Q25PRHCPU	0.15 × N+0.03 ms	0.30 × N+0.05 ms
Q02UCPU	0.00 ms	0.46 × N+0.17 ms
Q03UDCPU	0.00 ms	0.46 × N+0.12 ms
Q04UDHCPU、Q06UDHCPU	0.00 ms	0.20 × N+0.12 ms

N：局部软元件点数（单位：k 字）

表 10.21 使用 SRAM 卡内的局部软元件文件时

CPU 模块	同一文件内的副函数 调用时的扫描时间延长时间	别的文件内的副函数 调用时的扫描时间延长时间
Q02CPU	1.15 × N+0.30	2.30 × N+0.60
Q02HCPU，Q06HCPU，Q12HCPU， Q25HCPU，Q12PHCPU，Q25PHCPU， Q12PRHCPU，Q25PRHCPU	0.85 × N+0.15	1.65 × N+0.30
Q02UCPU	0.00 ms	0.82 × N+0.29 ms
Q03UDCPU	0.00 ms	0.82 × N+0.20 ms
Q04UDHCPU、Q06UDHCPU	0.00 ms	0.80 × N+0.20 ms

N：局部软元件点数（单位：k 字）

基本 注 10.15

(3) 多个程序的执行*注 10.15*

是指在 CPU 模块中执行多个程序时，各程序执行时的起始时间。
如果执行多个程序，将产生表 10.22 所示的处理时间。

表 10.22 执行多个程序时的起始时间

CPU 模块	处理时间
Q02CPU	0.08×n ms*1
Q02HCPU，Q06HCPU，Q12HCPU，Q25HCPU，Q12PHCPU，Q25PHCPU，Q12PRHCPU，Q25PRHCPU	0.03×n ms*1
Q02UCPU	0.04×n ms*1
Q03UDCPU	0.02×n ms*1
Q04UDHCPU、Q06UDHCPU	0.02×n ms*1

*1: n 为程序文件数。

基本 注 10.15

(4) 存储卡的装卸*注 10.15*

如果进行存储卡的装卸，将产生表 10.23 所示的处理时间。
本处理时间仅在存储卡装卸时的 1 个扫描周期产生。

表 10.23 存储卡装卸的处理时间

CPU 模块	处理时间	
	插入存储卡时	拔出存储卡时
Q02CPU	0.16ms	0.10ms
Q02HCPU，Q06HCPU，Q12HCPU，Q25HCPU，Q12PHCPU，Q25PHCPU，Q12PRHCPU，Q25PRHCPU	0.08ms	0.04ms
Q02UCPU	0.7ms	0.2ms
Q03UDCPU、Q04UDHCPU、Q06UDHCPU	0.6ms	0.1ms

基本 注 10.15

在基本模式 QCPU 中不能执行多个程序，也不能使用存储卡，因此无需考虑本页中记载的处理时间。

(5) 文件寄存器的使用

使用文件寄存器时，在参数设定中选择了 “使用与程序相同的文件”的情况下，将发生表 10.24 中所示的处理时间。

选择了 “使用下述文件”的情况下，不发生扫描时间的延迟。

表 10.24 使用文件寄存器时的处理时间

CPU 模块		处理时间
标准 RAM	Q02CPU	1.03ms
	Q02HCPU、Q06HCPU、Q12HCPU、Q25HCPU、Q12PHCPU、Q25PHCPU、Q12PRHCPU、Q25PRHCPU	0.41ms
	Q02UCPU	0.082ms
	Q03UDCPU	0.043ms
	Q04UDHCPU、Q06UDHCPU	0.041ms
SRAM 卡	Q02CPU	1.14×n ms*1
	Q02HCPU、Q06HCPU、Q12HCPU、Q25HCPU、Q12PHCPU、Q25PHCPU、Q12PRHCPU、Q25PRHCPU	0.50×n ms*1
	Q02UCPU	0.11×n ms*1
	Q03UDCPU	0.06×n ms*1
	Q04UDHCPU、Q06UDHCPU	0.06×n ms*1

*1: n 为程序文件数。

(6) RUN 中写入的执行

如果执行 RUN 中写入，将产生以下处理时间。

(a) 梯形图模式中的 RUN 中写入

RUN 中写入时如果重新设定 RUN 中写入用步，扫描时间仅延长表 10.25 所示的数值。

表 10.25 重新设定 RUN 中写入用步时的延时

CPU 模块型号	运行中写入用预留容量（步）	
	无变更	重新设定
Q00JCPU	最大 2.1ms	最大 30ms
Q00CPU	最大 1.7ms	最大 26ms
Q01CPU	最大 1.7ms	最大 36ms
Q02CPU	最大 4ms （最大 10ms*1）	最大 30ms
Q02HCPU、Q06HCPU、Q12HCPU、Q25HCPU、Q12PHCPU、Q25PHCPU、Q12PRHCPU、Q25PRHCPU	最大 2ms （最大 4ms*1）	最大 90ms
Q02UCPU	最大 1.2ms	最大 1.2ms
Q03UDCPU	最大 1.0ms	最大 1.0ms
Q04UDHCPU、Q06UDHCPU	最大 0.7ms	最大 0.7ms

*1: 表示运行中写入的写入范围内存在有局部软元件时的值。

基本
注10.16

(b) 文件的运行中写入时 *注 10.16*

表 10.26 文件的运行中写入时的延迟时间

CPU 模块型号	能确保程序内存中有可用空间时	能确保存储卡中有可用空间时（ATA 卡除外）
Q02CPU	最大 90ms （最大 200ms*2）	最大 150ms
Q02HCPU、Q06HCPU、Q12HCPU、Q25HCPU、Q12PHCPU、Q25PHCPU、Q12PRHCPU、Q25PRHCPU	最大 350ms （最大 650ms*2）	最大 650ms
Q02UCPU	最大 5ms	*3
Q03UDCPU	最大 4ms	*3
Q04UDHCPU、Q06UDHCPU	最大 4ms	*3

*2 ： 表示 SFC 程序时的值。
*3 ： 对于通用型 QCPU，与程序内存、存储卡的可用空间无关，可以进行文件的运行中写入。

使用 ATA 卡时，30k 步将延长 1.25s 的扫描时间。

基本
注 10.16

在基本模式 QCPU 中，由于不能执行文件的运行中写入，因此无需考虑文件的运行中写入时的延迟时间。

冗余 注10.17

(7) 接收组外的输出状态*注 10.17*

构成多 CPU 系统时，在 GX Developer 的多 CPU 设定中设为接收组外的输出状态时，扫描时间将延长。

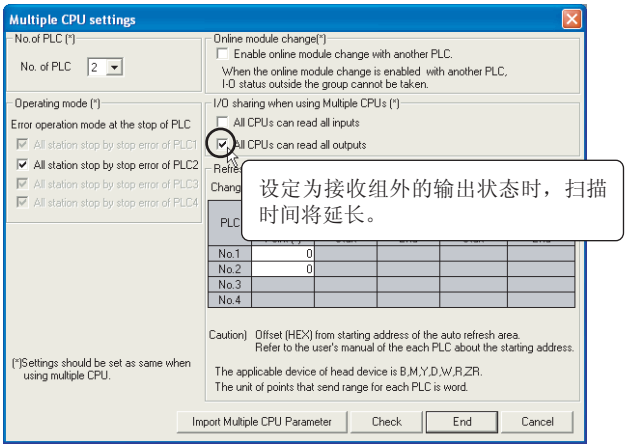


图 10.6 多 CPU 设定画面

(8) 仅在使用基本模式 QCPU 时扫描时间延长的功能

使用基本模式 QCPU 时，在使用以下功能时扫描时间将延长。

- 系统监视
- 电池检查
- 保险丝断裂检查
- 模块校验
- 一般数据处理

扩展基板 1 段合计装有 12 个智能功能模块，执行系统监视后的扫描时间延长。

表 10.27 使用系统监视时扫描时间的延长（共装有 12 个智能功能模块时）

CPU 模块型号	扫描时间延长
Q00JCPU	0.036ms
Q00CPU	0.015ms
Q01CPU	0.011ms

基本 注10.18 高性能 注10.18

(9) 存储器检查功能的执行*注 10.18*

如果在 CPU 模块中执行存储器检查功能，扫描时间将延长相当于存储器检查功能的处理时间。关于存储器检查功能的处理时间，请参阅 6.28 节。

冗余 注 10.17

在冗余 CPU 中不能使用多 CPU 系统，因此无需考虑组外的输出状态的接收处理时间。

基本 注 10.18 高性能 注 10.18

由于基本模式 QCPU、高性能模式 QCPU 中没有存储器检查功能，因此无需考虑存储器检查功能的处理时间。

基本 通用
注10.19 注10.19

(10)扫描时间测定 *注 10.19*

如果通过 GX Developer 执行 CPU 模块的扫描时间测定（☞6.13 节），将发生如表表 10.28 所示的处理时间。

表 10.28 通过 GX Developer 执行 CPU 模块的扫描时间测定时的处理时间

CPU 模块	处理时间
Q02CPU	52.0+39.5 ×分支指令数μs *1
Q02HCPU、Q06HCPU、Q12HCPU、Q25HCPU Q12PHCPU、Q25PHCPU Q12PRHCPU、Q25PRHCPU	23.0+17.0 ×分支指令数μs *1

*1: 分支指令数是指，从扫描测定的开始位置起至结束位置为止的区间内执行的下述指针分支指令、副程序调用指令的数目：
- 指针分支指令 ：CJ、SCJ、JMP
- 副程序调用指令 ：CALL(P)、FCALL(P)、ECALL(P)、EFCALL(P)、XCALL(P)、RET

基本 高性能 过程
注 10.20 注 10.20 注 10.20

冗余
注 10.20

(11)程序内存批量传送 *注 10.20*

如果通过 GX Developer 执行程序内存批量传送，将发生表 10.29 所示的处理时间。

表 10.29 传送至程序高速缓冲存储器时的扫描时间的延迟时间

CPU 模块	处理时间
Q02UCPU	4.2ms
Q03UDCPU	4.0ms
Q04UDHCPU、Q06UDHCPU	3.8ms

基本 通用
注 10.19 注 10.19

由于在基本型 QCPU、通用型 QCPU 中不能使用扫描时间测定，因此无需考虑扫描时间测定的处理时间。

基本 高性能 过程
注 10.20 注 10.20 注 10.20

冗余
注 10.20

在基本模式 QCPU、高性能模式 QCPU、过程 CPU、冗余 CPU 中，由于不使用程序高速缓冲存储器，因此无需考虑从程序高速缓冲存储器传送至程序内存时所需的处理时间。

10.1.4 通过变更设定可缩短扫描时间的因素

如本项所示通过变更 GX Developer 的可编程控制器参数的设定，可缩短扫描时间。

基本 冗余 通用
注10.21 注10.21 注10.21

(1) A 系列 CPU 兼容设定 (☞ 8.1.2 项 (2)) *注 10.21*

可编程控制器系统设定中，通过设为 “不使用 SM1000, SD1000 以后的特殊继电器 / 特殊寄存器”（设定画面的检查除外），可缩短表 10.30 所示的处理时间的扫描时间。

表 10.30 根据 A 系列 CPU 兼容设定的处理时间的异同

CPU 模块型号	处理时间
Q02CPU	0.07ms
Q02HCPU，Q06HCPU，Q12HCPU，Q25HCPU，Q12PHCPU，Q25PHCPU	0.03ms

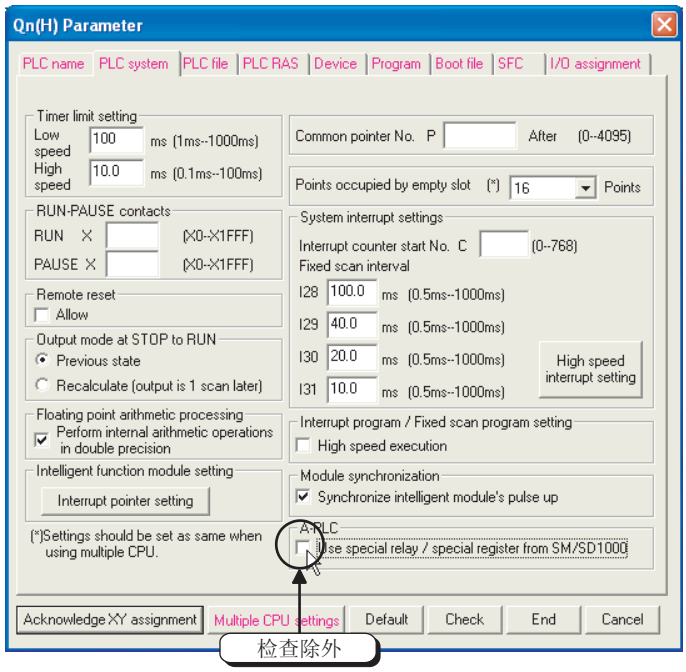


图 10.7 A 系列 CPU 兼容设置

要 点

设定为 “不使用 SM1000, SD1000 以上的特殊继电器 / 特殊寄存器”（设定画面的检查除外）时，请按以下所示将 A 系列兼容特殊继电器转换为 Q 系列专用特殊继电器。

- A 系列兼容特殊继电器 (SM1000~1299)
  →Q 系列专用特殊继电器 (SM0~999)
- A 系列兼容特殊继电器 (SD1000~1299)
  →Q 系列专用特殊继电器 (SD0~999)

基本 冗余 通用
注 10.21 注 10.21 注 10.21

在基本模式 QCPU、冗余 CPU、通用型 QCPU 中不能使用 A 系列 CPU 兼容设定。

基本 过程 冗余
注10.22 注10.22 注10.22

通用
UD
注10.22

(2) 浮动小数点运算处理 (☞ 8.1.2 项 (2)) *注 10.22*

如果在可编程控制器系统设定中设为 “不进行双精度内部运算处理”（设定画面的检查除外），使用了浮动小数点的指令的运算处理所需时间将缩短。

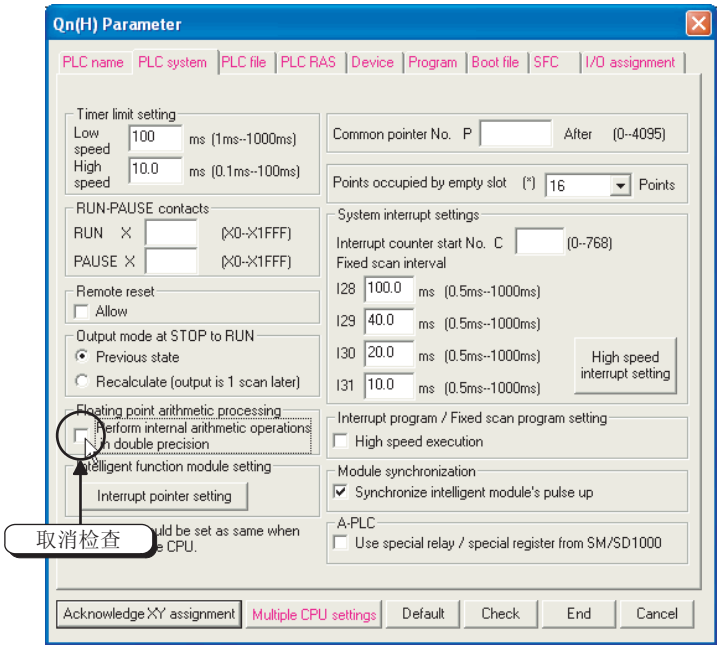


图 10.8 浮动小数点运算处理画面

备 注

关于使用了浮动小数点的指令的运算处理时间，请参照以下手册。

☞ QCPU(Q 模式)/QnACPU 编程手册（公共指令篇）

基本 过程 冗余
注 10.22 注 10.22 注 10.22

通用
UD
注 10.22

在基本模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 中，不能使用浮动小数点运算处理的设定。

基本 通用
注10.23 注10.23

(3) 文件使用方法设定 (☞ 8.1.2 项 (7)) *注 10.23*

在不使用文件寄存器、软元件初始值或者软元件注释的程序中，通过文件使用方法设定设为 “不使用”，便可缩短程序的起始时间。

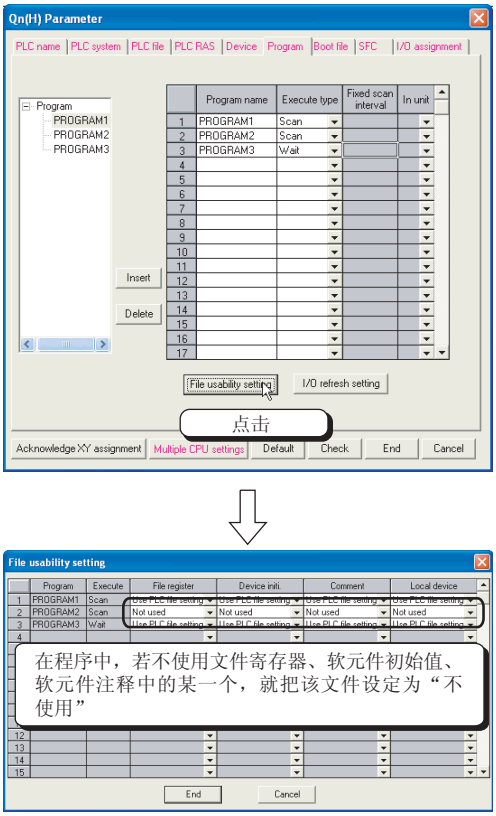


图 10.9 文件使用方法设定画面

要 点

仅在可编程控制器文件设定中设为 “与程序使用同一文件名”时本设定才能有效缩短时间。

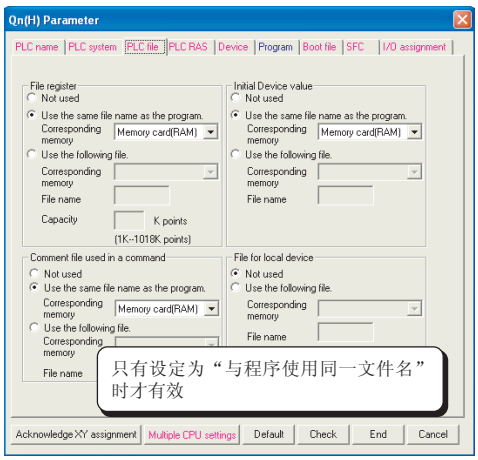


图 10.10 可编程控制器文件设定

基本 通用
注 10.23 注 10.23

在基本模式 QCPU、通用型 QCPU 中，不能使用文件使用方法设定。

10.2 其它处理时间

就 10.1 节中未说明的其它处理时间进行说明。

(1) 恒定扫描的精度

恒定扫描的精度如表 10.31 所示。

表 10.31 恒定扫描的精度

CPU 模块	无监视，无用户中断	有监视，无用户中断	无监视，有用户中断	有监视，有用户中断
Q00JCPU	0.20ms	0.90ms	中断程序的执行时间（参照 10.1.2 项 (4)(a)）	以下的合计时间 1) 左栏中的“有监视，无用户中断”所示的时间 2) 中断程序的执行时间的合计时间
Q00CPU	0.12ms	0.60ms		
Q01CPU	0.10ms	0.50ms		
Q02CPU	0.02ms			
Q02HCPU、Q06HCPU、Q12HCPU、Q25HCPU、Q12PHCPU、Q25PHCPU、Q12PRHCPU,Q25PRHCPU、Q02UCPU、Q03UDCPU、Q04UDHCPU、Q06UDHCPU	0.01ms			

有监视：连接 GX Developer，处于监视状态或利用串行口通讯功能与外围设备进行通讯的状态。
无监视：表示未处于利用 GX Developer 或串行口通讯功能与外围设备进行通讯的状态。

第 11 章　将程序写入 CPU 模块的步骤

以下说明将 GX Developer 编写的程序写入 CPU 模块的步骤。
关于 CPU 模块的启动步骤，本手册中没有记载。
关于 CPU 模块的启动步骤，请参照以下手册。
☞ QCPU 用户手册（硬件设计 / 维护点检篇）
☞ QCPU 用户手册（多 CPU 系统篇）
☞ QnPRHCPU 用户手册（冗余系统篇）

11.1　基本模式 QCPU

11.1.1　编写程序时的研究事项

在基本模式 QCPU 中编写程序时，需要预先确定程序容量、使用软元件点数等。

(1) 程序容量的研究
研究使用的 CPU 中可执行的程序容量以内，是否能储存程序与参数。
(☞ 5.4.3 项 (1))
各 CPU 模块中可执行的程序容量如表 11.1 所示。

表 11.1 基本模式 QCPU 的程序容量

CPU 模块	程序容量
Q00JCPU	8k 步 (32k 字节)
Q00CPU	8k 步 (32k 字节)
Q01CPU	14k 步 (56k 字节)

(2) 软元件的用途及点数的设定
研究在程序中使用的软元件的用途与点数。(☞ 第 9 章)

(3) 软元件初始值的设定
设定基本模式 QCPU 的软元件内存以及智能功能模块的缓冲存储器的初始值所需的数据。(☞ 6.26 节)

(4) 引导运行的讨论
进行引导运行时，进行可编程控制器参数的引导文件设定。
(☞ 5.1.5 项，11.1.4 项)

11.1.2 硬件检查

写入编写好的程序之前，需要进行硬件检查。

步骤如下，▭ 表示 GX Developer 的操作项目，▭ 表示基本模式 QCPU 的操作项目。

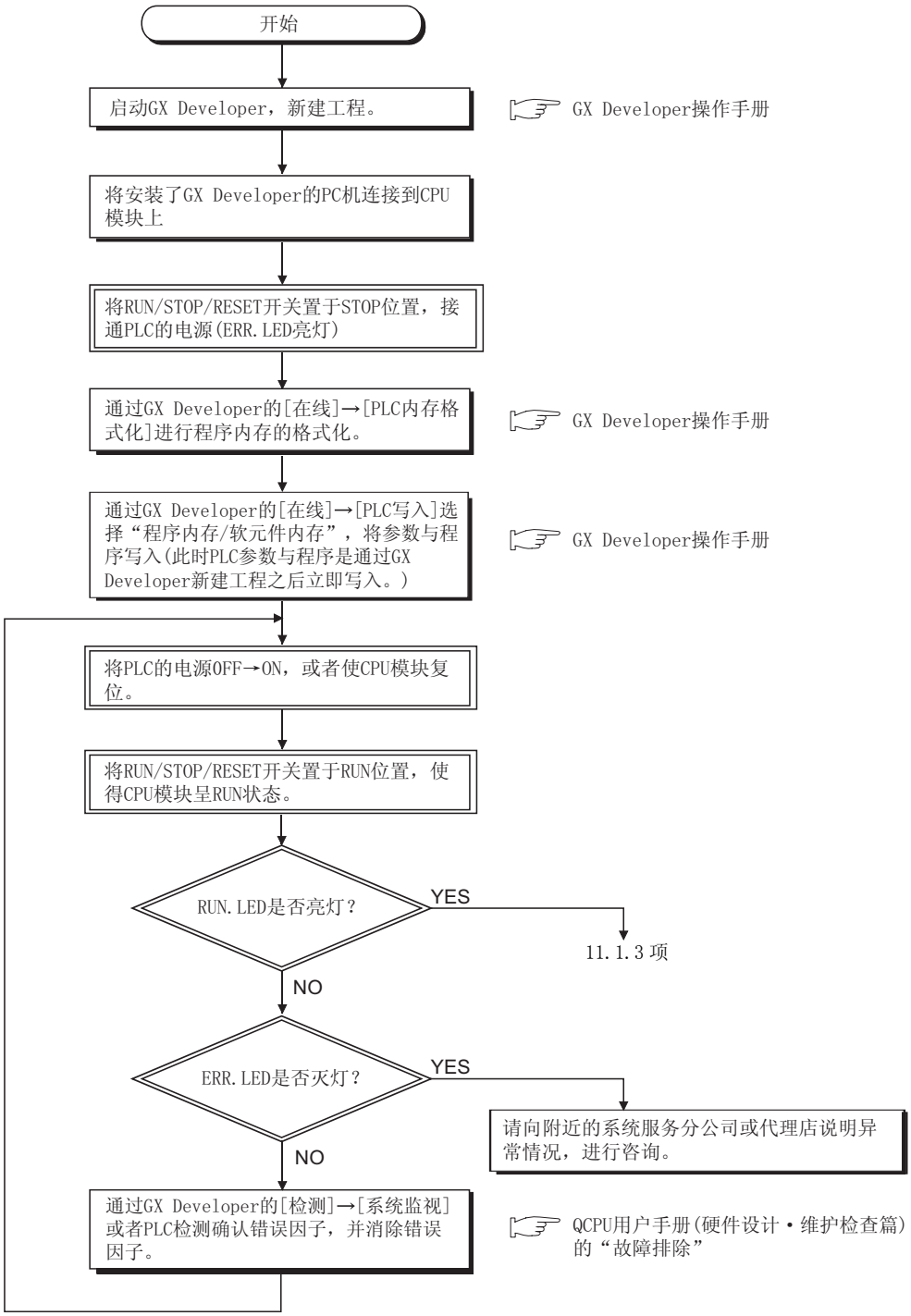


图 11.1 硬件检查的流程

备 注

关于 CPU 模块的设定和安装步骤，请参照以下手册。

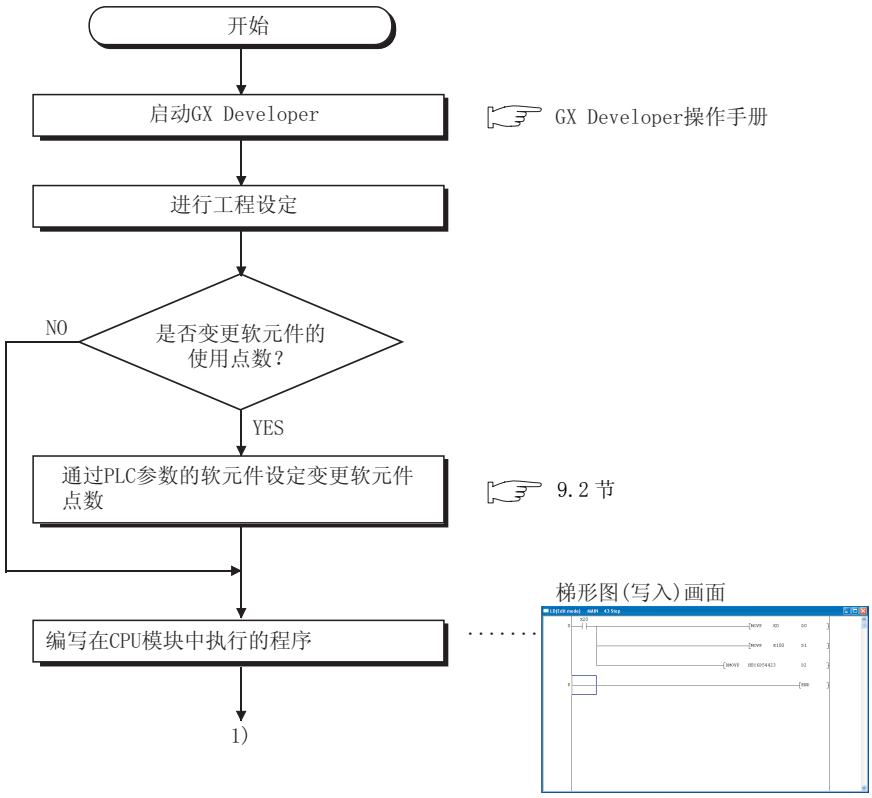
☞ QCPU 用户手册（硬件设计 / 维护点检篇）

11.1.3 将程序写入 CPU 模块的步骤

下面说明的是，将用 GX Developer 编写的参数和程序写入基本模式 QCPU 的步骤。

在本项中说明的是，将程序写入程序内存（☞ 5.1.2 项）的步骤。
将程序储存在标准 ROM 中，进行引导运行时，在实施了本项步骤之后再实施 11.1.4 项中的步骤。

步骤如下，▭ 表示 GX Developer 的操作项目，▭ 表示基本模式 QCPU 的操作项目。



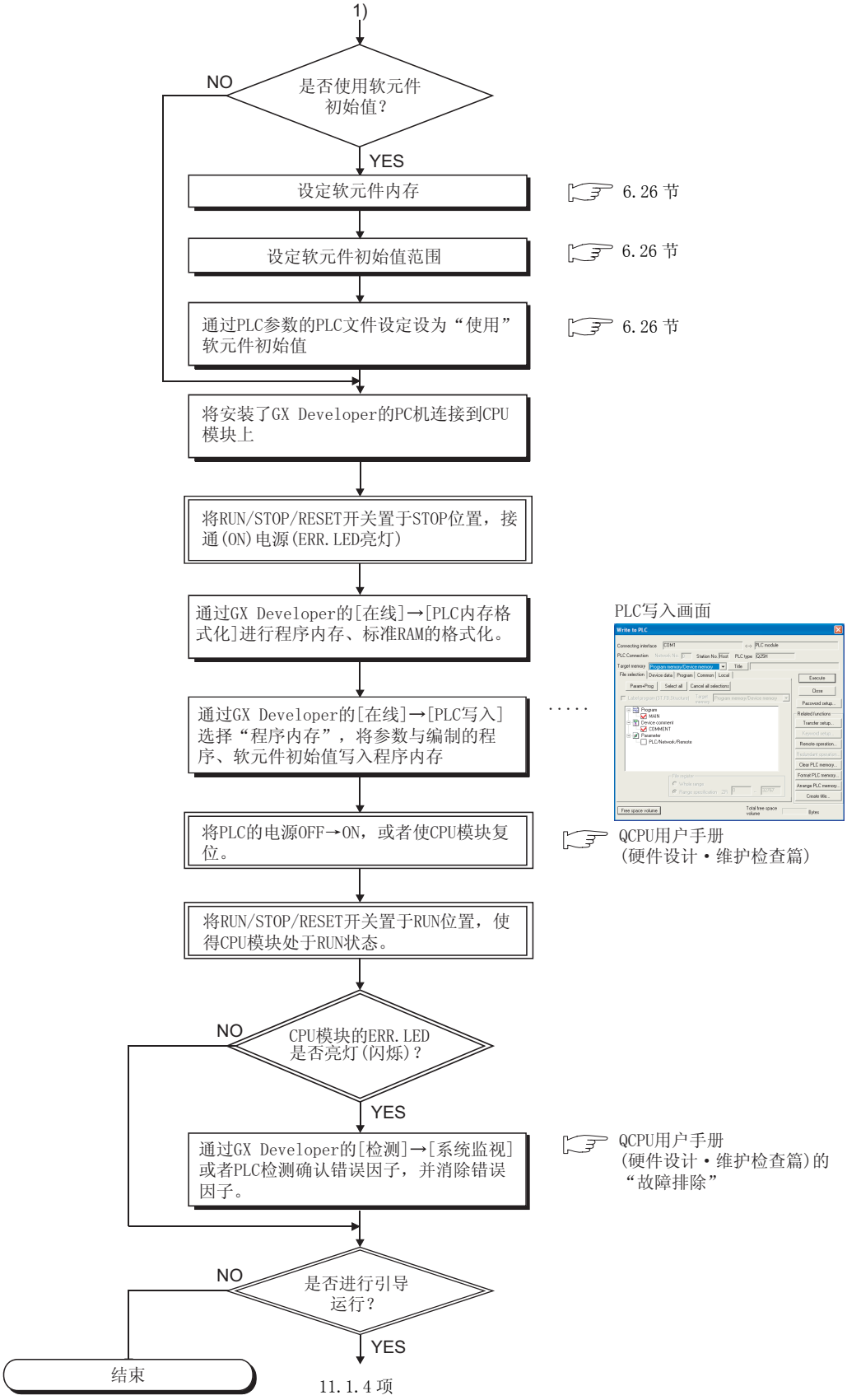


图 11.2 程序的写入步骤

11.1.4 引导运行的步骤

以下将说明引导运行的步骤。

步骤如下，□ 表示 GX Developer 的操作项目，▭ 表示基本模式 QCPU 的操作项目。

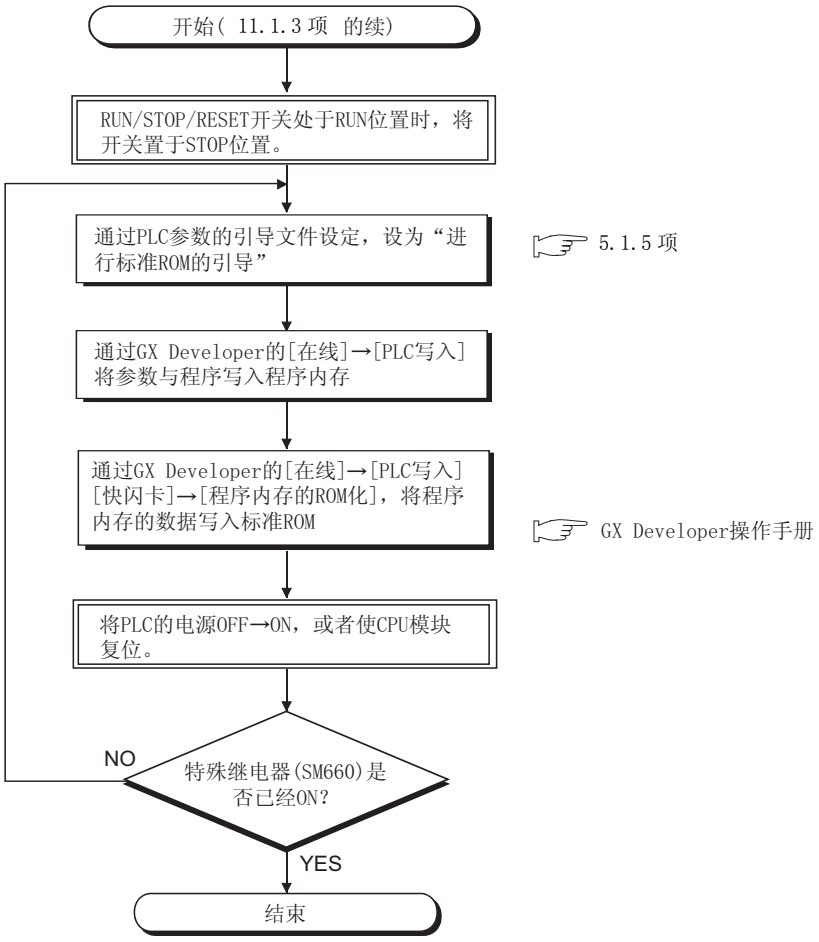


图 11.3 引导运行的流程

11.2 高性能模式 QCPU、过程 CPU 、冗余 CPU、通用型 QCPU

11.2.1 编写程序时的研究事项

用 CPU 模块编写程序时，需要事先确定在各程序中使用的程序容量、使用软元件点数和文件名等。

(1) 程序容量的研究

研究一下在使用 CPU 模块中可执行的程序容量以内，是否能储存程序和参数。(☞ 5.4.3 项 (2))

在各 CPU 模块中可执行的程序容量如表 11.2 所示。

表 11.2 高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 的程序容量

CPU 模块	程序容量
Q02CPU、Q02HCPU	28k 步 (112k 字节)
Q06HCPU	60k 步 (240k 字节)
Q12HCPU、Q12PHCPU、Q12PRHCPU	124k 步 (496k 字节)
Q25HCPU、Q25PHCPU、Q25PRHCPU	252k 步 (1008k 字节)
Q02UCPU	20k 步 (80k 字节)
Q03UDCPU	30k 步 (120k 字节)
Q04UDHCPU	40k 步 (160k 字节)
Q06UDHCPU	60k 步 (240k 字节)

可设定是否将参数储存在程序内存 / 标准 ROM/ 存储卡中。

只有程序需要上述程序容量时，请将参数储存在标准 ROM/ 存储卡中。

(2) 确定使程序结构化的单位

在高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 中，可改变文件名储存多个程序。

编写多个程序时，需确定使程序结构化的单位 (各工程、功能类别)。

(3) 所编写的程序的执行条件的设定

执行多个程序时，需设定各程序的执行条件。(☞ 3.3.6 项)

文件名与执行条件未设定时，不能执行程序。

(4) 使用软元件的用途与点数的设定

研究在程序中使用的软元件的用途与点数。(☞ 第 9 章)

(5) 软元件初始值的设定

设定高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 的软元件内存以及智能功能模块的缓冲存储器的初始值所需的数据。(☞ 6.26 节)

(6) 引导运行的研究

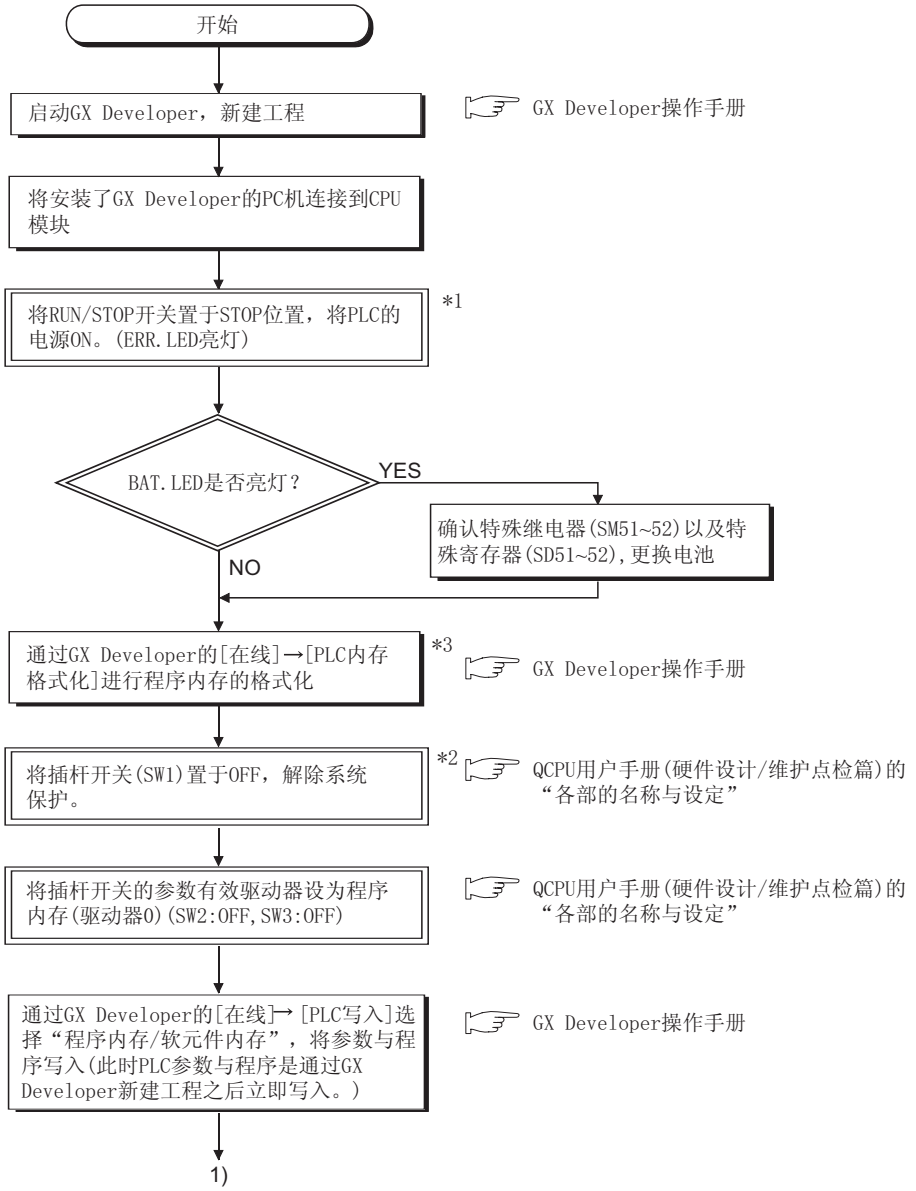
进行引导运行时，进行可编程控制器参数的引导文件设定。

(☞ 5.2.9 项，11.2.5 项)

11.2.2 硬件检查

写入编写好的程序之前，需进行硬件检查。

在以下步骤中，▭表示 GX Developer 的操作项目，▭表示高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 的操作项目。



*1 ：在通用型 QCPU 中，为 RUN/STOP/RESET 开关。

*2 ：在通用型 QCPU 中，无需通过插杆开关进行参数有效驱动器的设定。(☞ 5.2.11 项)

*3 ：在通用型 QCPU 中，不能通过插杆开关进行系统保护的解除。

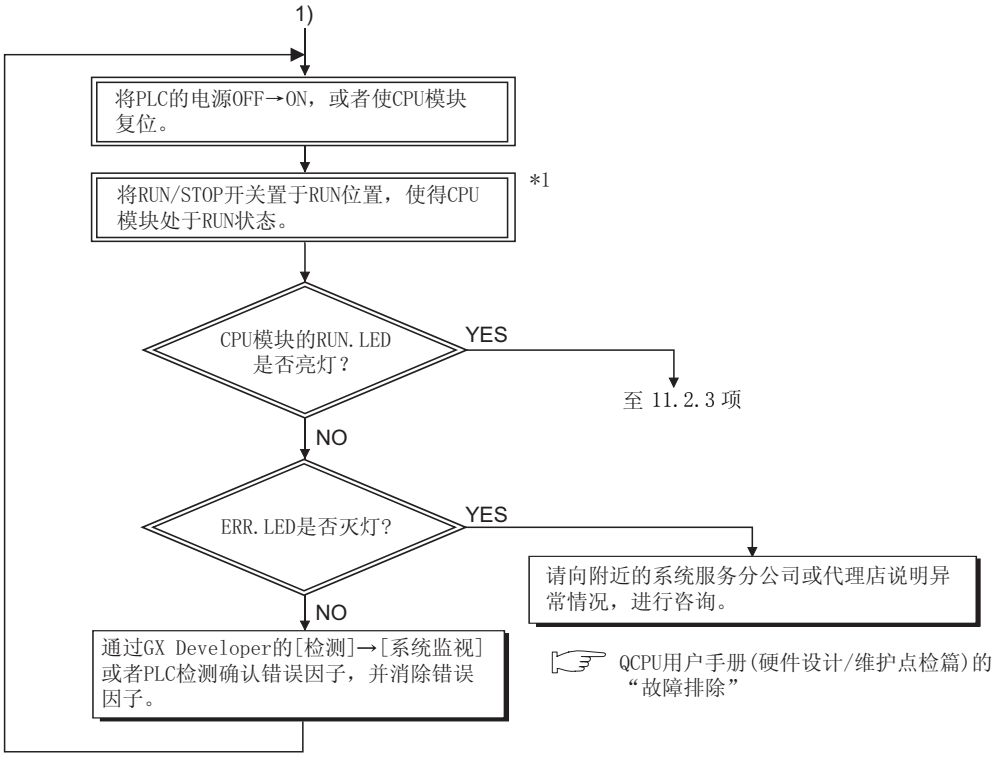


图 11.4 硬件检查的流程

*1: 在通用型 QCPU 中，为 RUN/STOP/RESET 开关。

备 注

关于 CPU 模块的设定、安装步骤，请参照以下手册。

☞ QCPU 用户手册（硬件设计 / 维护点检篇）

11.2.3 写入 1 个程序的步骤

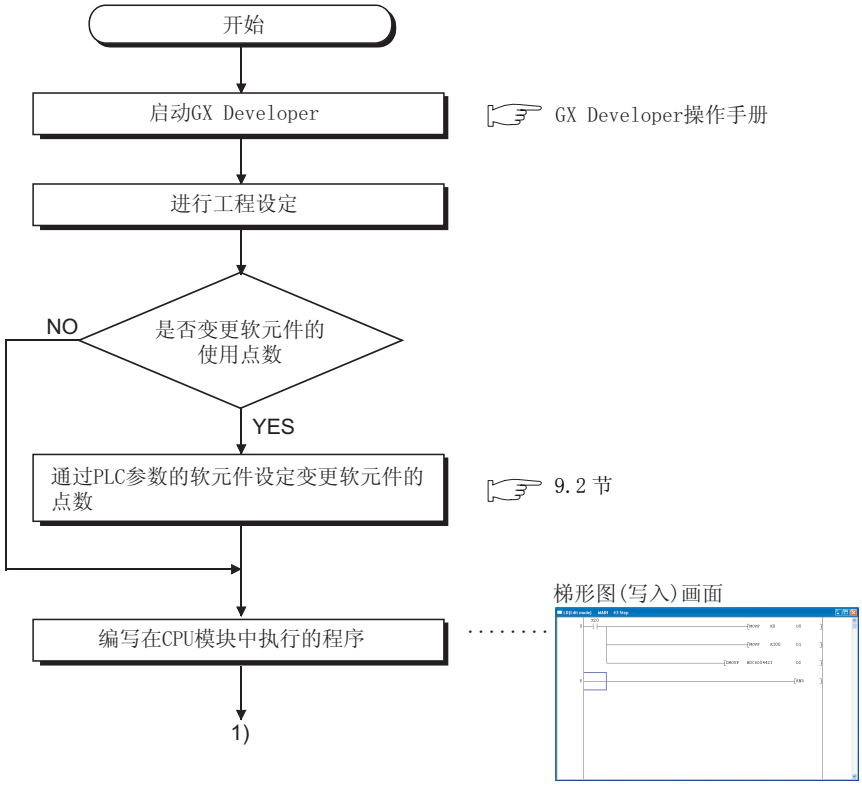
下面说明的是，将通过 GX Developer 创建的参数、程序写入到高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 时的步骤。

在本项中说明的是将程序写入程序内存（☞ 5.2.2 项）的步骤。

将程序储存到标准 ROM 注 11.1 及存储卡中，进行引导运行时，请在实施本项的步骤之后实施 11.2.5 项中的步骤。

通用 UD 注 11.1

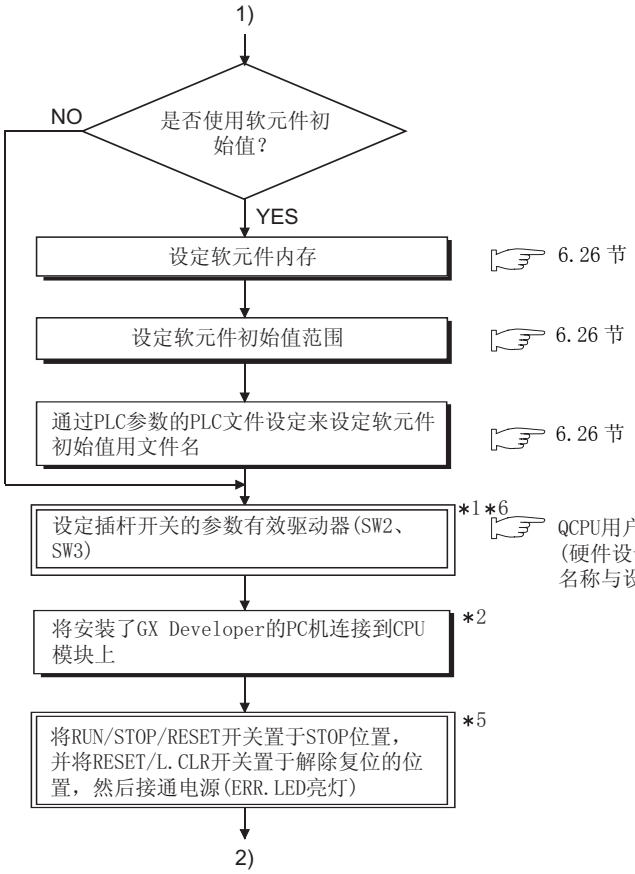
在以下步骤中，▭ 表示 GX Developer 的操作项目，▭ 表示高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 的操作项目。



通用 UD 注 11.1

在通用型 QCPU 中，不能进行从标准 ROM 至程序内存的引导。

（☞ 5.2.3 项）

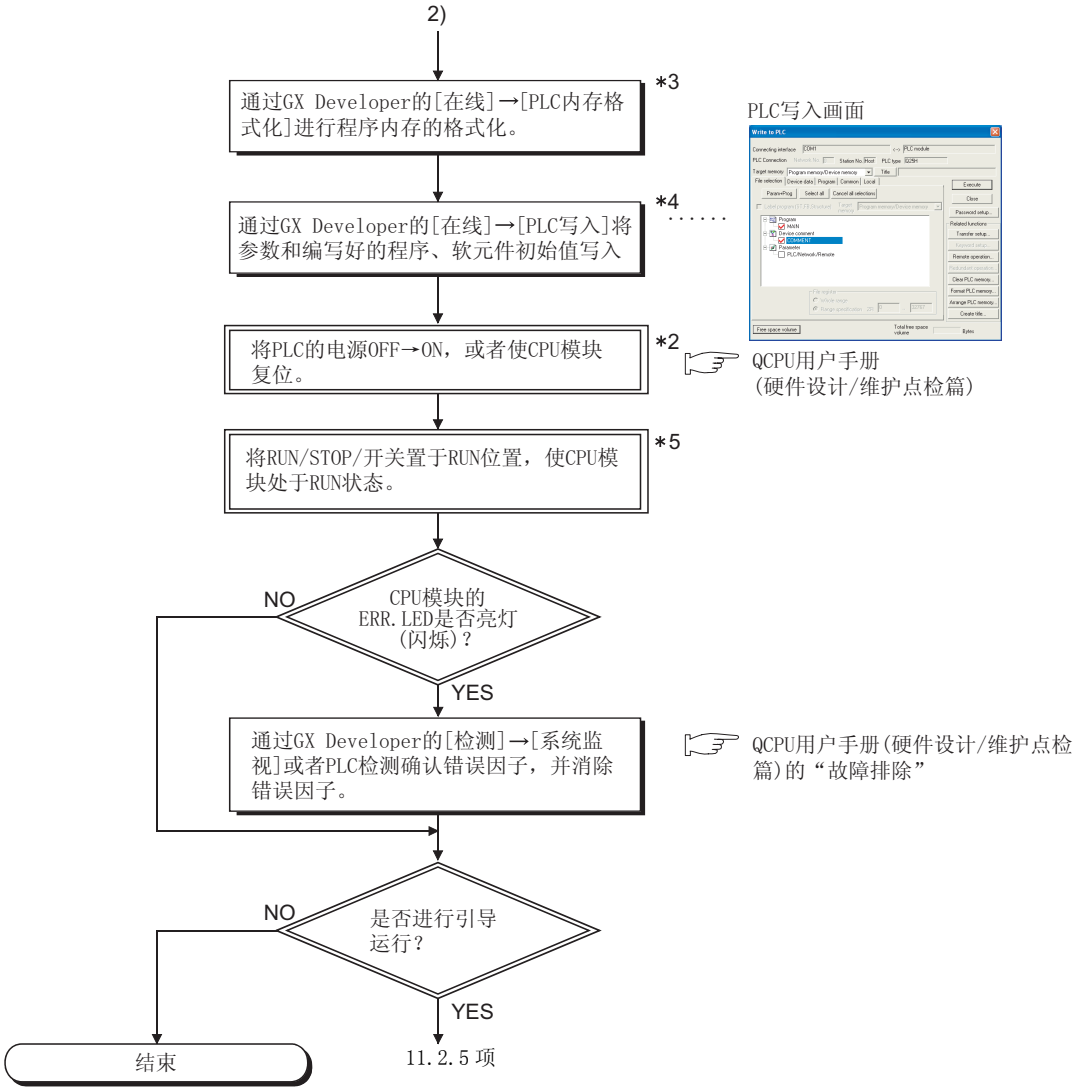


*1: 插杆开关的参数有效驱动器的缺省值为程序内存（驱动器 0）。(SW2:OFF，SW3:ON)
将参数储存在标准 ROM 或存储卡中时，请变更插杆开关的参数有效驱动器。

*2: 使用冗余 CPU 时的步骤，请参照以下手册。
(☞ QnPRHCPU 用户手册（冗余系统篇）)

*5: 在通用型 QCPU 中，为 RUN/STOP/RESET 开关。

*6: 在通用型 QCPU 中，无需通过插杆开关进行参数有效驱动器的设定。(☞ 5.2.11 项)



*2: 使用冗余 CPU 时的步骤，请参照以下手册。
QnPRHCPU 用户手册（冗余系统篇）

*3: 将文件寄存器和软元件初始值储存在标准 RAM 或存储卡 (Flash 卡除外）时，请将标准 RAM 或存储卡进行格式化。

*4: 请将各数据写入以下存储器。
- 程序 ..... 程序内存
- 参数 ..... 设定为插杆开关的参数有效驱动器的存储器
- 软元件初始值 ... 可编程控制器参数的可编程控制器文件设定所指定的存储器

*5: 在通用型 QCPU 中，为 RUN/STOP/RESET 开关。

图 11.5 写入 1 个程序的流程

要 点

在通用型 QCPU 中，在使用了 MELSECNET/H 的系统中由 1 号机进行管理的情况下，可以经由 MELSECNET/H 与发生了 MISSING PARA. 的通用型 QCPU 进行通信。（但是，仅在 1 号网络的情况下）

11.2.4 多个程序的写入步骤

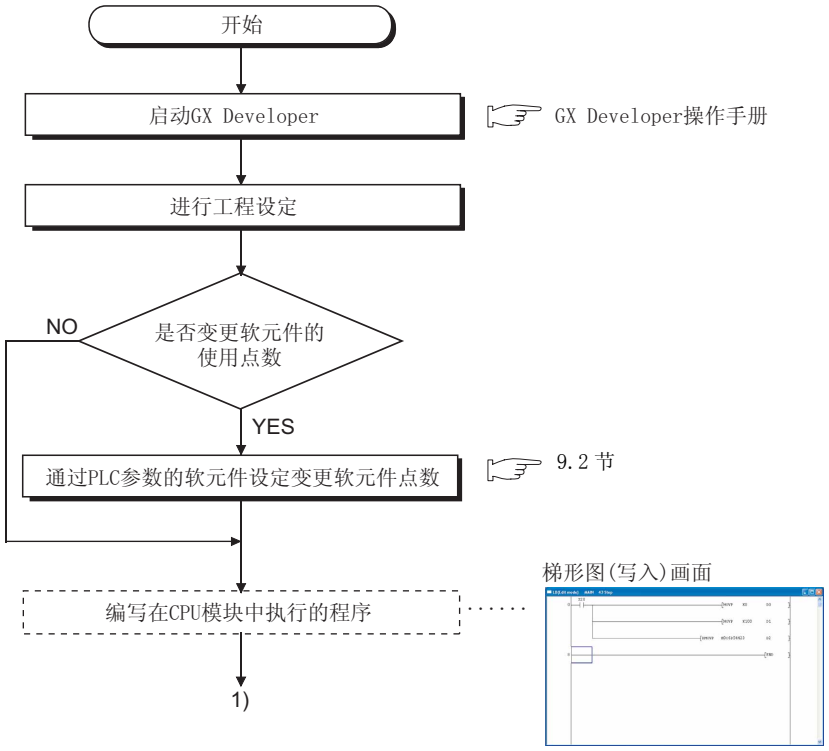
下面说明的是，将通过 GX Developer 创建的参数和多个程序写入到高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 中的步骤。

在本项中说明的是，将程序写入程序内存（☞ 5.2.2 项）的步骤。

通用 UD 注 11.2

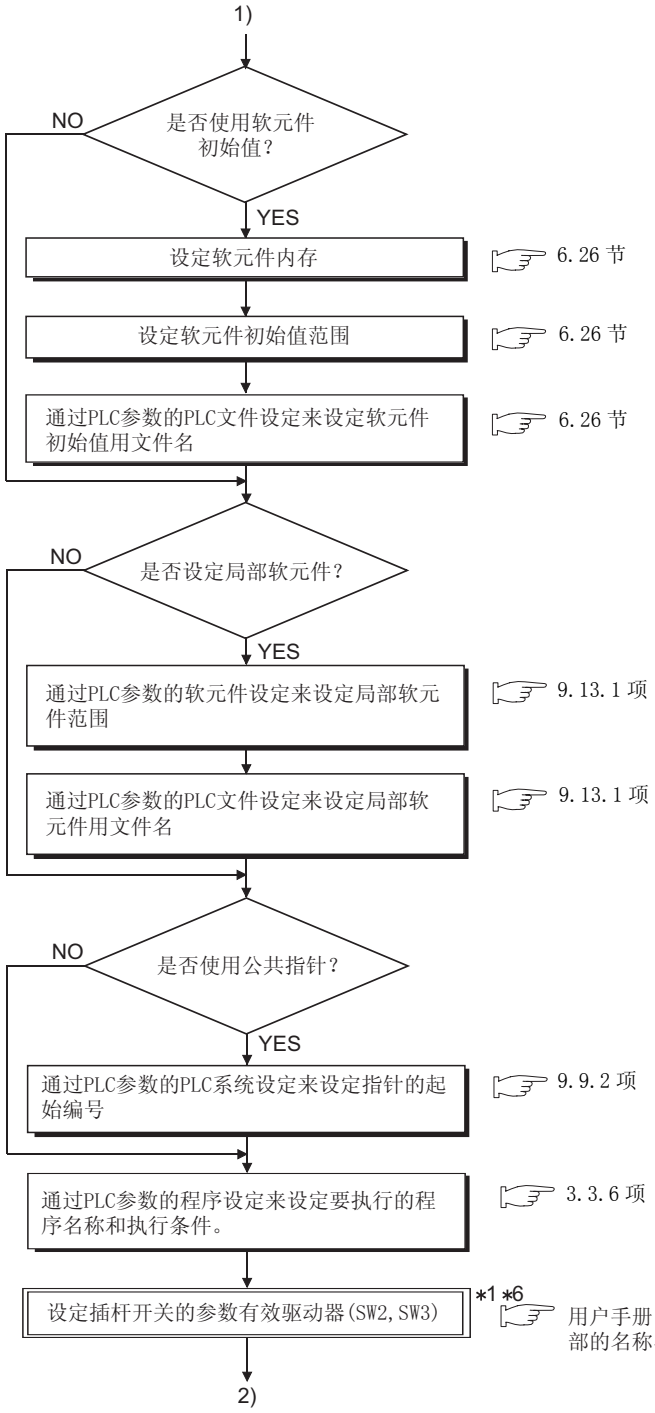
将程序储存到标准 ROM 注 11.2 和存储卡中进行引导运行时，应在本项步骤实施之后再实施 11.2.5 项中的步骤。

在下列步骤中，▭ 表示 GX Developer 的操作项目，▭ 表示高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 的操作项目。



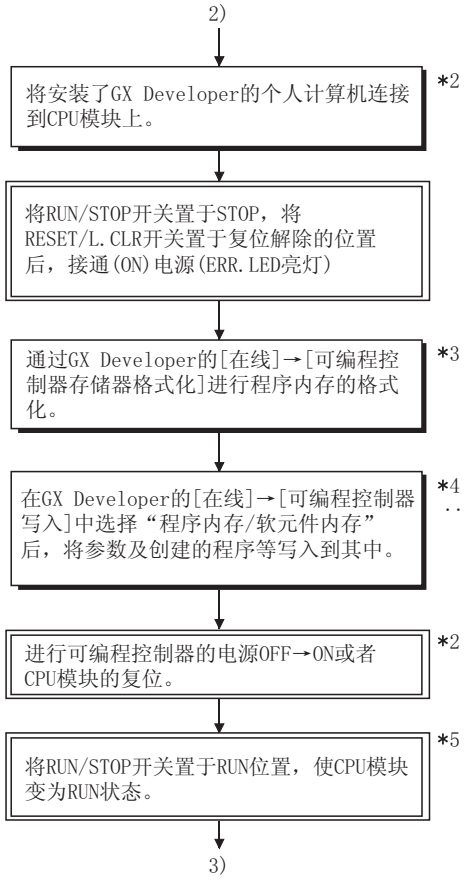
通用 UD 注 11.2

在通用型 QCPU 中，不能执行从标准 ROM 向程序内存的引导。
（☞ 5.2.3 项）

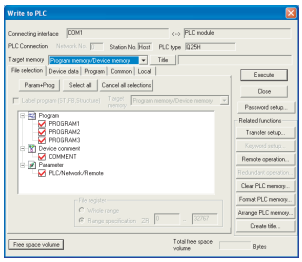


*1: 插杆开关的参数有效驱动器的缺省值为程序内存（驱动器 0）。(SW2:OFF，SW3:ON)
将参数储存在标准 ROM 或存储卡中时，请变更插杆开关的参数有效驱动器。

*6: 在通用型 QCPU 中，不需要通过插杆开关进行参数有效驱动设置。(☞ 5.2.11 项)



可编程控制器画面



*2 ： 关于使用冗余 CPU 时的步骤，请参照以下手册。

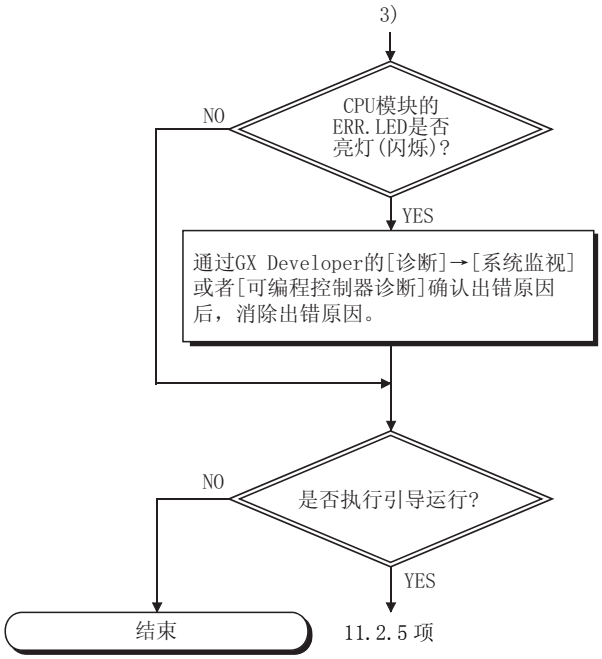
☞ QnPRHCPU 用户手册（冗余系统篇）

*3: 将文件寄存器和软元件初始值存储到标准 RAM 或存储卡（Flash 卡除外）中时，请将标准 RAM 或存储卡进行格式化。

*4: 请将各数据写入以下存储器。

- 程序..... 程序内存
- 参数..... 设定为插杆开关的参数有效驱动器的存储器
- 软元件初始值... 可编程控制器参数的可编程控制器文件设定中指定的存储器

*5: 在通用型 QCPU 中，为 RUN/STOP/RESET 开关。



☞ QCPU用户手册(硬件设计/维护点检篇)
的“故障排除”

图 11.6 进行多个程序写入的流程

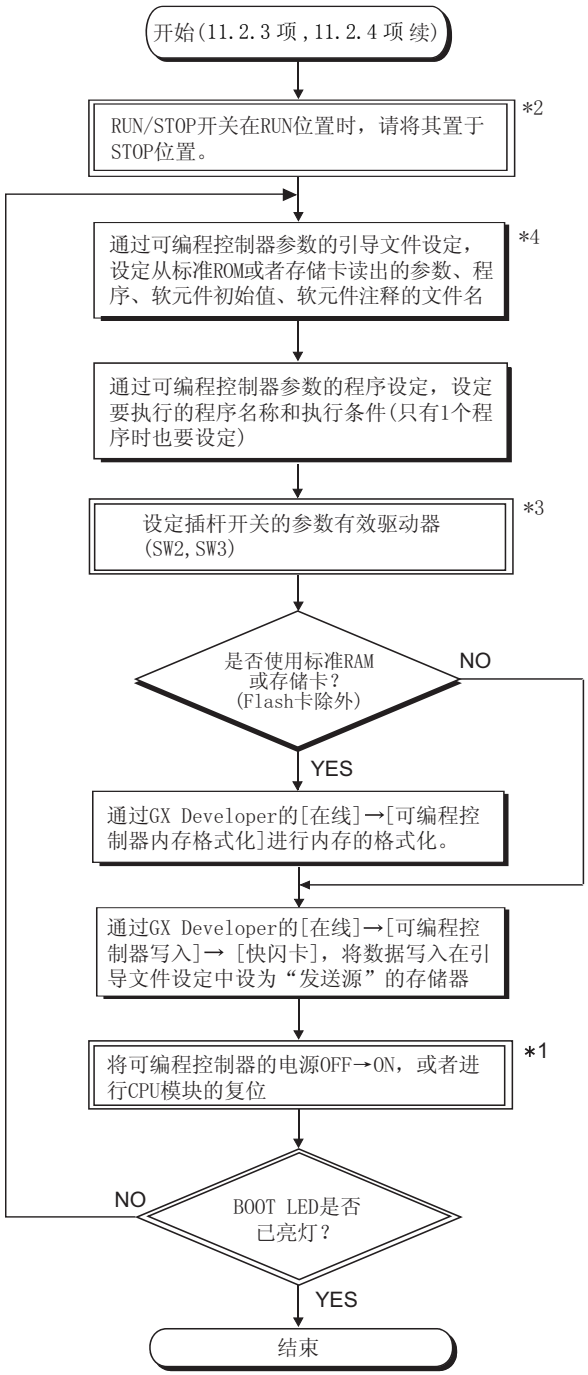
要 点

在通用型 QCPU 中，在使用了 MELSECNET/H 的系统中由 1 号机进行管理的情况下，可以经由 MELSECNET/H 与发生了 MISSING PARA. 的通用型 QCPU 进行通信。（但是，仅在 1 号网络的情况下）

11.2.5 引导运行的步骤

下面将说明引导运行的步骤。

步骤如下，▭ 表示 GX Developer 的操作项目，▭ 表示高性能模式 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 的操作项目。



*1: 关于使用冗余 CPU 时的步骤，请参照以下手册。
QnPRHCPU 用户手册（冗余系统篇）

*2: 在通用型 QCPU 中，为 RUN/STOP/RESET 开关。

*3: 在通用型 QCPU 中，无需通过插杆开关进行参数有效驱动器的设定。(5.2.11 项)

*4: 在通用型 QCPU 中，不能执行从标准 ROM 向程序内存的引导。(5.2.3 项)

图 11.7 引导运行的流程

备忘录

附录

附录 1 特殊继电器列表

特殊继电器，SM，是可编程控制器内用于固定应用的内部继电器。
由于这个原因，顺控程序不能以和使用普通内部继电器相同的方式使用特殊继电器。
但是，它们可以按照需要被变为 ON 或者 OFF 以控制 CPU 模块和远程 I/O 模块。
附表 .1 中的标题的含义如下所示。

附表 .1 特殊继电器的说明列表

条目	条目的功能
号码	•表示特殊寄存器号
名称	•表示特殊寄存器的名称
含义	•表示特殊寄存器的内容
解释	•更详细地讨论特殊寄存器的内容
设置方式 （当被设定时）	•表示继电器是由系统设定还是由用户设定；如果是由系统，当执行设置时。 〈设置方式〉 S ：由系统设定 U ：用户设定（顺控程序或者来自 GX Developer 的测试操作） S/U ：系统和用户都可以设定 〈当被设定时〉 只用于指示由系统设定寄存器 每个 END ：在每个 END 处理过程中进行设定 初始化 ：只在初始化处理（当电源变为 ON，或者从 STOP 变为 RUN 时）过程中进行设定 状态改变 ：只在有状态变化时才进行设定 出错 ：在出错发生时进行设定 指令执行 ：当指令被执行时设定 请求 ：只在有用户请求（通过 SM，等软元件。）时才进行设定 系统切换 ：在执行系统切换时进行设定。
相应的 ACPU M9□□□	•表示 ACPU 中相应的特殊继电器 (M9□□□) （当内容改变时，特殊继电器代表 M9□□□格式改变。不兼容 Q00J/Q00/Q01 和 QnPRH。） •新增表示特殊继电器是新近增加到 QnACPU 或者 Q 系列 CPU 模块中。
相应的 CPU	表示相应的 CPU 模块型号。 ○ ：表示所有的 QnACPU 和 QCPU。 QCPU ：表示所有的 Q 系列 CPU 模块。 Q00J/Q00/Q01 ：表示基本型 QCPU。 Qn(H) ：表示高性能型 QCPU。 QnPH ：表示过程控制 CPU。 QnPRH ：表示冗余 CPU。 QnU ：表示通用型 QCPU。 QnA ：表示 QnA 系列和 Q2ASCPU 系列。 Rem ：表示 MELSECNET/H 远程 I/O 模块。 每个 CPU 模块的型号 ：表示相关的特殊 CPU 模块。（例如：Q4AR，Q3A）

对于下列条目的详细信息，请参考下面的手册：
- 网络 ☞相应网络模块的手册
- SFC ☞QCPU(Q 模式)/QnACPU 编程手册 (SFC)

要 点

(1) SD1200 到 SD1255 用于 QnACPU。
这些继电器在 QCPU 中是空闲的。

(1) 诊断信息

附表 .2 特殊继电器

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU M9□□□	相应的 CPU
SM0	检测出错	OFF ：没有出错 ON ：出错	• 如果检测结果为出错则变为 ON。 （包括当报警器为 ON 和 CHK 指令检测到出错时） • 如果条件自此以后恢复为正常，仍保持为 ON。	S（出错）	新增	QnA Qn(H) QnPH QnPRH Rem
			• 如果检测结果为出错则变为 ON。 （包括报警器为 ON） • 如果自此以后条件恢复为正常，仍保持为 ON。	S（出错）	新增	Q00J/Q00/Q01 QnU
SM1	自检测出错	OFF ：没有自检测出错 ON ：自检测	• 如果检测结果为出错则变为 ON。 （不包括当报警器为 ON 或者 CHK 指令检测到出错时） • 如果自此以后条件恢复为正常，则保持为 ON。	S（出错）	M9008	QnA Qn(H) QnPH QnPRH Rem
			• 如果检测结果为出错则变为 ON。 （不包括当报警器为 ON 时） • 如果自此以后条件恢复为正常，则保持为 ON。	S 出错）	新增	Q00J/Q00/Q01 QnU
SM5	出错公共信息	OFF ：没有出错公共信息 ON ：出错公共信息	• 当 SM0 是 ON 时，如果有出错公共信息则为 ON	S（出错）	新增	○ Rem
SM16	出错个别信息	OFF ：没有出错个别信息 ON ：出错个别信息	• 当 SM0 是 ON 时，如果有出错个别信息则为 ON	S（出错）	新增	
SM50	出错复位	OFF→ON：出错复位	• 指导出错复位操作	U	新增	
SM51	电池电量不足锁存	OFF ：正常 ON ：出错	• 如果 CPU 模块或者存储卡的电池电压下降到低于额定值，则为 ON。 • 如果此后电池电压恢复为正常，保持为 ON。 • 和 BAT.ALARM/BAT.LED 同步。	S（出错）	M9007	QnA Qn(H) QnPH QnPRH QnU
			• 如果 CPU 模块或者存储卡上的电池电压下降到低于额定值，则为 ON。 • 如果此后电池电压恢复为正常，保持为 ON。 • 和 BAT.LED 同步	S（出错））	新增	Q00J/Q00/Q01
SM52	电池电量不足	OFF ：正常 ON ：出错	• 和 SM51 相同，但是当电池电压恢复为正常后，接着变为 OFF。	S（出错）	M9006	○ QCPU QnA
SM53	AC/DC DOWN 检测	OFF ：没有检测到 AC/DC DOWN ON ：检测到 AC/DC DOWN	• 如果在 AC 电源模块的过程中发生 20ms 以内的瞬间掉电，变为 ON。 当电压被切换到 OFF，然后再切换为 ON 时复位。	S（出错）	M9005	
			• 如果在使用 DC 电源模块的过程中发生 10ms 以内的瞬间掉电，变为 ON。 当电源被切换到 OFF，然后再切换为 ON 时复位。			
			• 如果在使用 DC 电源模块的过程中发生 1ms 以内的瞬间掉电，变为 ON。 当电源切换到 OFF，然后再切换为 ON 时复位。			
SM54	MINI 链接出错	OFF ：正常 ON ：出错	• 即使只在安装的 AJ71PT32(S3) 模块中的一个上检测到 MINI(S3) 链接出错，也变为 ON。 • 如果自此以后条件恢复为正常，则保持为 ON。	S（出错）	M9004	

附表 .2 特殊继电器

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU M9□□□	相应的 CPU
SM56	操作出错	OFF ：正常 ON ：操作出错	• 当产生操作出错时，变为 ON。 • 如果自此以后情况恢复为正常，仍保持为 ON。	S（出错）	M9011	○
SM60	保险丝熔断检测	OFF ：正常 ON ：模块有熔断的保险丝	• 如果至少有一个保险丝熔断的输出模块存在，变为 ON。 • 如果自此以后情况恢复为正常，仍保持为 ON。 • 即使对远程 I/O 站输出模块，也进行保险丝熔断状态检查。	S（出错）	M9000	○ Rem
SM61	I/O 模块验证出错	OFF ：正常 ON ：出错	• 如果 I/O 模块和上电时注册的状态不同，变为 ON。 • 如果自此以后情况恢复为正常，则保持为 ON。 • 对远程 I/O 站模块也进行 I/O 模块校验。	S（出错）	M9002	
SM62	报警器检测	OFF ：没有检测到 ON ：检测到	• 如果即使只有一个报警器 F 变为 ON，也变为 ON。	S（电源接通或者）	M9009	○
SM80	CHK 检测	OFF ：没有检测到 ON ：检测到	• 如果 CHK 指令检测到出错，变为 ON。 • 即使自此以后情况恢复为正常，保持为 ON。	S（电源接通或者）	新增	QnA Qn(H) QnPH QnPRH
SM90	用于步转移的看门狗定时器的启动 （只在存在 SFC 程序时才被激活）	OFF ：不启动（看门狗定时器复位） ON ：启动（看门狗定时器被启动）	对应 SD90	• 当步转移看门狗定时器的测量开始时，变为 ON。 • 当它变为 OFF 时，复位步转移看门狗定时器。	U	M9108
SM91			对应 SD91			M9109
SM92			对应 SD92			M9110
SM93			对应 SD93			M9111
SM94			对应 SD94			M9112
SM95			对应 SD95			M9113
SM96			对应 SD96			M9114
SM97			对应 SD97			新增
SM98			对应 SD98			新增
SM99			对应 SD99			新增
SM100	串行通讯功能使用标志	OFF ：没有使用串行通讯功能。 ON ：使用了串行通讯功能。	• 在串行通讯设置参数中存储是否使用串行通讯功能的设置	S（电源接通或者复位）	新增	Q00J/Q00/Q01
SM101	通讯协议状态标志	OFF ：GX Developer ON ：MC 协议通讯软元件	• 存储通过 RS-232 接口是通讯的软元件是 GXDeveloper 还是 MC 协议通讯软元件	S(RS232 通讯）		
SM110	协议出错	OFF ：已清除 ON ：异常	• 当在串行通讯功能中，使用了异常协议进行通讯时，变为 ON。 • 如果自此以后情况恢复为正常，仍保持为 ON。	S（出错）		
SM111	通讯状态	OFF ：正常 ON ：异常	• 当用于进行通讯的模式和在串行通讯功能中设置的模式不一样时，变为 ON。 • 如果自此以后情况恢复为正常，仍保持为 ON。	S（出错）		
SM112	出错信息清除	ON ：已清除	• 当存储在 SM110、SM111、SD110 和 SD111 中的出错代码被清除时，变为 ON。（当从 OFF 变为 ON 时触发）	U		
SM113	溢出出错	OFF ：正常 ON ：异常	• 当在串行通讯出错中发生溢出出错时，变为 ON。	S（出错）		
SM114	奇偶性出错	OFF ：正常 ON ：异常	• 当在串行通讯出错中发生奇偶性出错时，变为 ON。	S（出错）		
SM115	结构出错	OFF ：正常 ON ：异常	• 当在串行通讯出错中发生结构出错时，变为 ON。	S（出错）		

(2) 系统信息

附表 .3 特殊继电器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的CPU ACPU M9□□□	相应的 CPU
SM202	LED OFF 命令	OFF→ON : LED OFF	• 当此继电器从 OFF 变为 ON 时，对应 SD202 上的单个位的 LED 关闭	U	新增	QnA Qn(H) QnPH QnPRH QnU
SM203	STOP 触点	STOP 状态	• 在 STOP 状态变为 ON	S（状态改变）	M9042	○
SM204	PAUSE 触点	PAUSE 状态	• 在 PAUSE 状态变为 ON	S（状态改变）	M9041	
SM205	STEP-RUN 触点	STEP-RUN 状态	• 在 STEP-RUN 状态变为 ON	S（状态改变）	M9054	QnA
SM206	PAUSE 许可线圈	OFF :PAUSE 禁止 ON :PAUSE 许可	• 当 PAUSE 触点变为 ON 时，如果此继电器是 ON，则 PAUSE 状态被输入	U	M9040	○
	软元件测试请求接收状态	OFF : 软元件测试还没有执行 ON : 软元件测试执行	• 当在 GX Developer 上执行软元件测试模式时，变为 ON。	S（请求）	新增	Rem
SM210	时钟数据设置请求	OFF : 忽略 ON : 设置请求	• 当此继电器从 OFF 变为 ON 时，在扫描周期有变化的 END 指令执行之后，正被从 SD210 存储到 SD213 的时钟数据将被写到 CPU 模块中。	U	M9025	○
SM211	时钟数据出错	OFF : 没有出错 ON : 出错	• 当时钟数据 (SD210 到 SD213) 值产生出错时为 ON，如果没有检测到出错则为 OFF。	S（请求）	M9026	
SM212	时钟数据显示	OFF : 忽略 ON : 显示	• 在 CPU 模块前面的 LED 显示器上，以月、日、小时、分、秒的格式显示时钟数据。（只对 Q3ACPU 和 Q4ACPU 有效）	U	M9027	Q3A Q4A Q4AR
SM213	时钟数据显示	OFF : 忽略 ON : 读取请求	• 当此继电器是 ON 时，时钟数据以 BCD 值读到 SD210 到 SD213 中。	U	M9028	○ Rem
SM220	1 号 CPU 准备完成	OFF :1 号 CPU 准备未完成 ON :1 号 CPU 准备完成	电源接通或复位操作时，可以从其它号 CPU 访问 1 号 CPU 时，变为 ON。SM220 在多 CPU 间同步设置中被设置为非同步时，作为 1 号 CPU 访问的互锁。	S（状态改变）	新增	QnU
SM221	2 号 CPU 准备完成	OFF :2 号 CPU 准备未完成 ON :2 号 CPU 准备完成	电源接通或复位操作时，可以从其它号 CPU 访问 2 号 CPU 时，变为 ON。SM221 在多 CPU 间同步设置中被设置为非同步时，作为 2 号 CPU 访问的互锁。			
SM222	3 号 CPU 准备完成	OFF :3 号 CPU 准备未完成 ON :3 号 CPU 准备完成	电源接通或复位操作时，可以从其它号 CPU 访问 3 号 CPU 时，变为 ON。SM222 在多 CPU 间同步设置中被设置为非同步时，作为 3 号 CPU 访问的互锁。			
SM223	4 号 CPU 准备完成	OFF :4 号 CPU 准备未完成 ON :4 号 CPU 准备完成	电源接通或复位操作时，可以从其它号 CPU 访问 4 号 CPU 时，变为 ON。SM223 在多 CPU 间同步设置中被设置为非同步时，作为 4 号 CPU 访问的互锁。			QnU*5
SM235	运行中模块更换标志	OFF : 当前没有进行运行中模块更换 ON : 运行中模块更换正在进行	在运行中模块更换正在进行时变为 ON。（用于本站 CPU）	S（当运行中模块更换结束时）	新增	QnPH
SM236	运行中模块更换标志	OFF : 运行中模块更换未结束 ON : 运行中模块更换结束	• 运行中模块更换结束后变为 ON 一个扫描周期。 • 此触点只能被扫描程序所使用。（用于本站 CPU）	S（当运行中模块更换结束时）	新增	
SM240	1 号 CPU 出错标志	OFF :1 号 CPU 复位被取消 ON :1 号 CPU 复位	• 当 1 号 CPU 的复位被取消时，变为 OFF。 • 当 1 号 CPU 被复位时，变为 ON。（包括从基板上卸载可编程控制器的情况）。 其他可编程控制器也进入复位状态。	S（状态改变）	新增	Q00/Q01*1 Qn(H)*1 QnPH QnU
SM241	2 号 CPU 出错标志	OFF :2 号 CPU 复位被取消 ON :2 号 CPU 复位	• 当 2 号 CPU 的复位被取消时，变为 OFF。 • 当 2 号 CPU 被复位时，变为 ON。（包括从基板上卸载可编程控制器的情况）。 其他可编程控制器处于 “MULTI CPU DOWN” 状态（出错代码：7000）。			
SM242	3 号 CPU 出错标志	OFF :3 号 CPU 复位被取消 ON :3 号 CPU 复位	• 当 3 号 CPU 的复位被取消时，变为 OFF。 • 当 3 号 CPU 被复位时，变为 ON。（包括从基板上卸载可编程控制器的情况）。 其他可编程控制器处于 “MULTI CPU DOWN” 状态（出错代码 :7000）。			
SM243	4 号 CPU 出错标志	OFF :4 号 CPU 复位被取消 ON :4 号 CPU 复位	• 当 4 号 CPU 复位被取消时变为 OFF。 • 当 4 号 CPU 是复位时，变为 ON（包括从基板上卸载可编程控制器的情况）。 其他可编程控制器处于 “MULTI CPU DOWN” 状态（出错代码 :7000）。			Qn(H)*1 QnPH QnU*5
SM244	1 号 CPU 出错标志	OFF :1 号 CPU 正常 ON :1CPU 处于停止出错状态	• 当 1 号 CPU 正常时，变为 OFF（包括连续出错）。 • 当 1 号 CPU 处于停止出错状态时，变为 ON。			Q00/Q01*1 Qn(H)*1 QnPH QnU
SM245	2 号 CPU 出错标志	OFF :2 号 CPU 正常 ON :2 号 CPU 处于停止出错状态	• 当 2 CPU 号正常时，变为 OFF（包括连续出错）。 • 当 2 号 CPU 处于停止出错状态，变为 ON。			
SM246	3 号 CPU 出错标志	OFF :3 号 CPU 正常 ON :3 号 CPU 处于停止出错状态	• 当 3 号 CPU 正常时，变为 OFF（包括连续出错）。 • 当 3 号 CPU 处于停止出错状态，变为 ON。			

*5: 除 Q02UCPU 外的通用型 QCPU。

附表 .3 特殊继电器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU M9□□□	相应的 CPU
SM247	4 号 CPU 出错标志	OFF :4 号 CPU 正常 ON :4 号 CPU 处于停止出错状态	• 当 4 号 CPU 正常时，变为 OFF（包括连续出错）。 • 当 4 号 CPU 号处于停止出错状态，变为 ON	S（状态改变）	新增	Qn(H)*1 QnPH QnU*5
SM250	读有负载的最大 I/O 号	OFF ：忽略 ON ：读	• 当此继电器从 OFF 变为 ON 时，有负载的最大 I/O 号被读到 SD250。	U	新增	QnA Qn(H) QnPH QnPRH Rem
SM251	I/O 更换标志	OFF ：无更换 ON ：更换	• 在将要被更换的 I/O 模块的起始 I/O 号设定到 SD251 之后，通过打开此 继电器，可以在运行中更换 I/O 模块（电源接通）。（对每个设置只能更换一个模块。） • 在程序或者用于在 RUN 过程中进行 I/O 模块更换的外围设备的测试模式中，或者在用于在 STOP 时进行 I/O 更换的外围设备的测试模式，将此继电器变为 ON。 • 在 I/O 模块更换完成之前，不要执行 RUN/STOP 模式改变。	U	M9094	Q2A(S1) Q3A Q4A Q4AR
SM252	I/O 更换 OK	OFF ：更换禁止 ON ：更换允许	• 当 I/O 更换 OK 时，变为 ON。	S（结束）	新增	
SM254	所有站刷新命令	OFF ：刷新到达的站 ON ：刷新所有站	• 对批量刷新有效（也对低速循环有效） • 在 MELSECNET/H 中指定是只接收到达的站，还是接收所有的从站。	U	新增	Qn(H) QnPH QnPRH
			• 在 MELSECNET/G 中指定是只接收到达的站，还是接收所有的从站。			Qn(H)*2
			• 对批量刷新有效（也对低速循环有效） • 在 MELSECNET/H 中指定是只接收到达的站，还是接收所有的从站。			QnU
SM255	MELSECNET/10, MELSECNET/H 模块 1 信息	OFF ：运行网络 ON ：待机网络	• 为待机网络变为 ON（如果 没有指定是活动还是待机，则假设是活动。）	S（初始化）	新增	QnA Qn(H) QnPH QnPRH
SM256		OFF ：读 ON ：不读	• 用于从链接到 CPU 模块的刷新（B、W 等），表示是否从链接模块读。	U	新增	
SM257		OFF ：写 ON ：不写	• 用于从 CPU 模块到链接的刷新（B、W 等），指定是否写入链接模块。	U	新增	
SM260	MELSECNET/10, MELSECNET/H 模块 2 信息	OFF ：运行网络 ON ：待机网络	• 为待机网络变为 ON （如果没有是活动还是待机，假设是活动。）	S（初始化）	新增	
SM261		OFF ：读 ON ：不读	• 用于从链接到 CPU 模块的 刷新（B、W 等），表示是否从链接模块读。	U	新增	
SM262		OFF ：写 ON ：不写	• 用于从 CPU 模块到链接的刷新（B、W 等），指定是否写到链接模块。	U	新增	
SM265	MELSECNET/10, MELSECNET/H 模块 3 信息	OFF ：运行网络 ON ：待机网络	• 为待机网络变为 ON （如果没有指定是活动还是待机，假定为活动。）	S（初始化）	新增	
SM266		OFF ：读 ON ：不读	• 用于从链接到 CPU 模块的刷新（B、W 等），表示是否从链接模块读。	U	新增	
SM267		OFF ：写 ON ：不写	• 用于从 CPU 模块到链接的刷新（B、W 等），指定是否到写到链接模块。	U	新增	
SM270	MELSECNET/10, MELSECNET/H 模块 4 信息	OFF ：运行网络 ON ：待机网络	• 为待机网络变为 ON （如果没有指定是活动还是待机，假定为活动。）	S（初始化）	新增	
SM271		OFF ：读 ON ：不读	• 用于从链接到 CPU 模块的刷新（B、W 等），表示是否从链接模块读。	U	新增	
SM272		OFF ：正常 ON ：出错	• 用于从 CPU 模块到链接的刷新（B、W 等），指定是否写到链接模块。	U	新增	
SM280	CC-Link 出错	OFF ：无延迟 ON ：有延迟	• 当在任何一个安装的 CC-Link 模块中检测到 CC-Link 出错时，变为 ON。当正常操作恢复时，变为 OFF。	S（状态改变）	新增	Qn(H) QnPH QnPRH Rem
			• 当在任何一个安装的 CC-Link 模块中检测到 CC-Link 出错时，变为 ON。如果条件恢复为正常，仍保持为 ON。	S（出错）	新增	QnA
SM315	通讯预留时间延迟许可 / 禁止标志	OFF ：无延迟 ON ：有延迟	• 当为通讯处理预留的时间被设定到 SD315 中时，此标志被激活。 • 以将 END 处理延迟在 SD315 中为执行通讯处理设定的时间，变为 ON。 （扫描时间将增加在 SD315 中设定的时间。） • 当没有通讯处理时，变为 OFF，以便于不延迟在 SD315 中设定的时间的情况下执行 END 处理。（默认是 OFF）	U	新增	Q00J/Q00/Q01

*1: 适用于功能版本为 B 或者更高的 CPU。
*2: 以序列号的高 5 位为 “09012” 以后的 CPU 为对象。
*5: 除 Q02UCPU 外的通用型 QCPU。

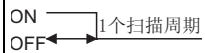
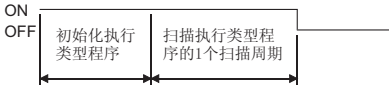
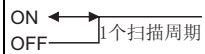
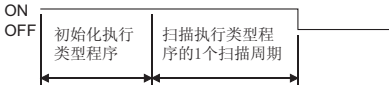
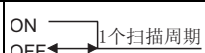
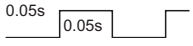
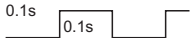
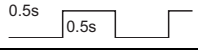
附表 .3 特殊继电器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU M9□□□	相应的 CPU
SM320	存在 / 不存在 SFC 程序	OFF :SFC 程序不存在 ON :SFC 程序存在	• 当注册了 SFC 程序时，变为 ON。 • 当没有注册 SFC 程序时，变为 OFF。	S（初始化）	M9100	QnA Q00J/Q00/Q01*1 Qn(H) QnPH QnPRH QnU
SM321	启动 / 停止 SFC 程序	OFF :SFC 程序没有执行（停止） ON :SFC 程序执行（启动）	• 初始值被设定为和 SM320 相同的值。（自动变为 ON ，如果 SFC 程序存在。） • 将此继电器从 ON 变为 OFF 以停止程序执行。 • 将此继电器从 OFF 变为 ON 以恢复程序执行。	S（初始化）/U	M9101 形式在变化	
SM322	SFC 程序启动状态	OFF ：初始化启动 ON ：重启	• 在可编程控制器参数对话框的 SFC 设置中的 SFC 程序启动模式被设定为初始值。 在初始化启动时：OFF 在连续启动时 : ON	S（初始化）/U	M9102 形式在变化	
SM323	SFC 程序启动状态	OFF ：连续转移无效 ON ：连续转移有效	在没有设定 SFC 数据软元件的 “Continuous transition bit” 地方设定存在 / 不存在块的连续转移。	U	M9103	
SM324	连续转移防止标志	OFF ：当转移被执行时 ON ：当没有转移时	• 在连续转移模式中运行或者在连续转移过程中为 OFF，而当连续转移没有执行时为 ON。 • 在没有转移模式中运行时为常 ON。	S（指令执行）	M9104	
				S（状态改变）	新增	
SM325	块停止时的输出模式	OFF : OFF ON ：保持	选择在块停止时刻活动步的线圈输出是否被保持。 • 作为初始值，当线圈输出为 OFF 时参数中块停止时的输出模式是 OFF，当线圈输出被保持时为 ON。 • 当此继电器是 OFF 时，所有线圈输出变为 OFF。 • 当此继电器是 ON 时，线圈输出被保持。	S（初始化）/U	M9196	
SM326	SFC 软元件清除模式	OFF ：清除软元件 ON ：保持软元件	当 SFC 程序存在时，选择当处于停止状态的 CPU 在顺控程序或者 SFC 程序被修改之后重新运行时软元件的状态。	U	新增	
SM327	执行结束步时的输出	OFF ：保持步的输出为 OFF（清除） ON ：保持步的输出保持	• 该继电器 OFF 时，选择在从 STOP - 程序写入 -RUN 的切换时刻软元件的状态。（所有软元件，步继电器除外）	S（初始化）/U	新增	QnA Qn(H) QnPH QnPRH QnU
				U		Q00J/Q00/Q01*1
SM328	当到达结束步时清除处理的模式	OFF ：清除处理执行。 ON ：清除处理是不执行。	当结束步到达时，如果在块中除正被保持的步以外没有活动步存在，选择是否执行清除处理? • 当此继电器变为 OFF 时，所有的活动步被强制结束，以结束此块。 • 当此继电器是 ON 时，执行的块继续执行。 • 当结束步到达时，如果除正被保持的步以外没有活动步存在，则正被保持的步被结束以结束此块。	U	新增	Q00J/Q00/Q01*1 QnU
SM330	低速执行类型程序的运行模式	OFF ：异步模式 ON ：同步模式	选择低速执行类型程序是以同步模式执行，还是以异步模式执行。 • 异步模式（此继电器变为 OFF。）在此模式中，低速执行类型程序的操作在剩余的时间内继续执行。 • 同步模式（此继电器变为 ON。） 在此模式中，低速执行类型程序的操作不再继续执行，如果有剩余的时间，操作将从下一个扫描周期继续执行。	U	新增	QnA Qn(H) QnPH
SM331	正常 SFC 程序执行状态	OFF ：未执行 ON ：正在执行	• 表示是否有正常的 SFC 程序正在执行。 • 被用作 SFC 控制指令执行互锁。	S（状态改变）	新增	Qn(H)*3 QnPH*4 QnPRH
SM332	程序执行管理 SFC 程序执行状态	OFF ：未执行 ON ：正在执行	• 表示是否有程序执行管理 SFC 程序正在执行。 • 被用作 SFC 控制指令执行互锁。			
SM390	访问执行标志	ON 表示智能功能模块访问的结束	• 执行的智能功能模块 访问指令在存储之前的状态。（当重新执行智能功能模块访问指令时，此数据被重写。） • 被用户在程序中用作结束位。	S（状态改变）	新增	Qn(H) QnPH QnPRH
SM391	GINT 指令执行结束标志	OFF ：未执行 ON ：执行结束	表示 S(P),GINT 指令的执行状态。 • 指令执行前为 OFF。 • 指令执行结束后为 ON。	S（指令执行）	新增	QnU

*1: 适用于功能版本为 B 或者更高的 CPU。
*3: 以序列号的高 5 位为 “04122” 以后的 CPU 为对象。
*4: 以序列号的高 5 位为 “07032” 以后的 CPU 为对象。

(3) 系统时钟 / 计数器

附表 .4 特殊继电器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU M9□□□	相应的 CPU
SM400	常 ON		• 正常是 ON	S（每个 END 处理）	M9036	○
SM401	常 OFF		• 正常是 OFF	S（每个 END 处理）	M9037	
SM402	在 RUN 之后，只 ON1 个扫描周期		• 在 RUN 之后，只 ON1 个扫描周期。 • 此连接只能用于扫描执行类型程序。 • 当使用初始化执行类型程序时，在 RUN 之后的第一个扫描周期中，此继电器在扫描执行类型程序的 END 处理中变为 OFF。 	S（每个 END 处理）	M9038	QnA Qn(H) QnPH QnPRH QnU
			• 在 RUN 之后，只 ON1 个扫描周期。	S（每个 END 处理）	新增	Q00J/Q00/Q01
SM403	在 RUN 之后，只 OFF1 个扫描周期		• 在 RUN 之后，只 OFF1 个扫描周期。 • 此连接只能用于扫描周期执行类型程序。 • 当使用了初始化执行类型程序时，在 RUN 之后第一个扫描周期中，扫描执行类型程序的 END 处理时此继电器变为 OFF。 	S（每个 END 处理）	M9039	QnA Qn(H) QnPH QnPRH QnU
			• 在 RUN 之后，只 OFF1 个扫描周期。	S（每个 END 处理）	新增	Q00J/Q00/Q01
SM404	在 RUN 之后低速执行类型程序只 ON1 个扫描周期		• 在 RUN 之后，只 ON1 个扫描周期。 • 此连接只能用于低速执行类型程序。	S（每个 END 处理）	新增	QnA Qn(H) QnPH
SM405	在 RUN 之后，低速执行类型程序只 OFF1 个扫描周期		• 在 RUN 之后，只 OFF1 个扫描周期。 • 此连接只能用于低速执行类型程序。	S（状态改变）	新增	
SM409	0.01 秒时钟		• 以 5 毫秒为周期，在 ON 和 OFF 之间重复改变。 • 当可编程控制器电源变为 OFF 或者执行 CPU 模块复位时，从 OFF 开始启动。 （注意，在程序执行过程中，当指定的时间结束时，ON-OFF 状态改变。）	S（状态改变）	新增	Qn(H) QnPH QnPRH QnU
SM410	0.1 秒时钟		• 以指定的时间间隔，在 ON 和 OFF 之间重复改变。 • 当可编程控制器电源变为 OFF 或者执行 CPU 模块复位时，从 OFF 开始启动。 （注意，在程序执行过程中，当指定的时间结束时，ON-OFF 状态改变。）	S（状态改变）	M9030	○
SM411	0.2 秒时钟				M9031	
SM412	1 秒时钟				M9032	
SM413	2 秒时钟				M9033	
SM414	2n 秒时钟		• 此继电器按照在 SD414 中指定的时间间隔（单位：s），在 ON 和 OFF 两个状态交替改变。 • 当可编程控制器电源变为 OFF 或者执行 CPU 模块复位时，从 OFF 开始启动。 （注意，在程序执行过程中，当指定的时间结束时，ON-OFF 状态改变。）	S（状态改变）	M9034 形式	
SM415	2n(ms) 时钟		• 此继电器按照 SD415 中指定的时间间隔（单位：ms），在 ON 和 OFF 两个状态交替改变。 • 当可编程控制器电源变为 OFF 或者执行 CPU 模块复位时，从 OFF 开始启动。 （注意，在程序执行过程中，当指定的时间结束时，ON-OFF 状态改变。）	S（状态改变）	新增	Qn(H) QnPH QnPRH QnU

附表 .4 特殊继电器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU M9□□□	相应的 CPU
SM420	0 号用户定时时钟		• 继电器以固定扫描间隔重复 ON/OFF 切换。 • 当可编程控制器电源变为 ON 或者执行 CPU 模块复位时，从 OFF 开始启动。 （对于冗余 CPU，此继电器在系统切换之后一直为 OFF。） • ON/OFF 间隔是用 DUTY 指令设定的 DUTY | n1 | n2 | SM420 n1：ON 扫描周期间隔 n2：OFF 扫描周期间隔	S（每个 END 处理）	M9020	○
SM421	1 号用户定时时钟				M9021	
SM422	2 号用户定时时钟				M9022	
SM423	3 号用户定时时钟				M9023	
SM424	4 号用户定时时钟	n2个扫描周期 n2个扫描周期 n1个扫描周期			M9024	
SM430	5 号用户定时时钟		• 用于和 SM420 到 SM424 低速程序一起使用	S（每个 END 处理）	新增	QnA Qn(H) QnPH
SM431	6 号用户定时时钟					
SM432	7 号用户定时时钟					
SM433	8 号用户定时时钟					
SM434	9 号用户定时时钟					

(4) 扫描周期信息

附表 .5 特殊继电器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU M9□□□	相应的 CPU
SM510	低速程序执行标志	OFF ：结束或者未执行 ON ：正在执行。	• 当执行低速执行类型程序时，变为 ON。	S（每个 END 处理）	新增	QnA Qn(H) QnPH
SM551	读模块工作间隔	OFF ：忽略 ON ：读	• 当此继电器从 OFF 变为 ON 时，SD550 指定的模块工作间隔被读到 SD551 到 SD552。	U	新增	QnA Qn(H) QnPH QnPRH Rem

(5) I/O 刷新

附表 .6 特殊继电器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU M9□□□	相应的 CPU
SM580	程序到程序 I/O 刷新	OFF ：未刷新 ON ：刷新	• 当此特殊继电器变为 ON 时，在第一个程序执行之后执行 I/O 刷新，然后执行第二个程序。 当顺控程序和 SFC 程序将要被执行时，先执行顺控程序，再执行 I/O 刷新，最后执行 SFC 程序。	U	新增	Q00J/Q00/Q01*1

*1：适用于功能版本为 B 或者更高的 CPU。

(6) 存储卡

附表 .7 特殊继电器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU M9□□□	相应的 CPU
SM600	存储卡 (A) 可用标志	OFF : 不可用 ON : 使用允许	• 当存储卡 (A) 准备好供用户使用时，变为 ON	S（状态改变）	新增	QnA Qn(H) QnPH QnPRH QnU
SM601	存储卡 (A) 保护标志	OFF : 无保护 ON : 保护	• 当存储卡 (A) 保护开关为 ON，变为 ON	S（状态改变）	新增	
SM602	驱动器 1 标志	OFF : 无驱动器 1 ON : 驱动器 1 存在	• 当安装的存储卡 (A) 是 RAM 时，变为 ON	S（状态改变）	新增	
SM603	驱动器 2 标志	OFF : 无驱动器 2 ON : 驱动器 2 存在	• 当安装的存储卡 (A) 是 ROM 时，变为 ON	S（状态改变）	新增	
SM604	存储卡 (A) 使用标志	OFF : 未使用 ON : 使用中	• 当安装的存储卡 (A) 是 ROM 时，变为 ON	S（状态改变）	新增	
SM605	存储卡 (A) 卸载 / 插入禁止标志	OFF : 卸载 / 插入允许 ON : 卸载 / 插入禁止	• 当存储卡 (A) 不能插入或者卸载时，变为 ON	U	新增	
SM609	存储卡卸载 / 插入允许标志	OFF : 卸载 / 插入禁止 ON : 卸载 / 插入允许	• 由用户设定为 ON，以允许卸载 / 插入存储卡。 • 在存储卡卸载之后，由系统设定为 OFF。 • 只有当 SM604 和 SM605 为 OFF 时，才能使用此触点。	S/U	新增	Qn(H) QnPH QnPRH QnU
SM620	驱动器 3/4 可用标志	OFF : 不可用 ON : 允许使用	• 常 ON	S（初始化）	新增	QCPU
	存储卡 B 可用标志	OFF : 不可用 ON : 允许使用	• 当存储卡 B 准备好供用户使用时，变为 ON	S（初始化）	新增	Q2A(S1) Q3A Q4A Q4AR
SM621	驱动器 3/4 保护标志	OFF : 无保护 ON : 保护	• 常 OFF	S（初始化）	新增	QCPU
	存储卡 B 保护标志	OFF : 无保护 ON : 保护	• 当存储卡 B 保护开关是 ON 时，变为 ON	S（初始化）	新增	Q2A(S1) Q3A Q4A Q4AR
SM622	驱动器 3 标志	OFF : 无驱动器 3 ON : 驱动器 3 存在	• 常 ON	S（初始化）	新增	QCPU
			• 当驱动器 3（卡 2RAM 区）为当前状态时，变为 ON	S（初始化）	新增	Q2A(S1) Q3A Q4A Q4AR
SM623	驱动器 4 标志	OFF : 无驱动器 4 ON : 驱动器 4 存在	• 常 ON	S（初始化）	新增	QCPU
			• 当驱动器 4（卡 2ROM 区）为当前状态时，变为 ON	S（初始化）	新增	Q2A(S1) Q3A Q4A Q4AR
SM624	存储卡 B 使用标志	OFF : 未使用 ON : 使用中	• 变为 ON，当驱动器 3（标准 RAM）或者驱动器 4（标准 ROM）内文件被使用时。	S（状态改变）	新增	Q00J/Q00/Q01*1 Qn(H) QnPH QnPRH
			• 当存储卡 B 被使用时，变为 ON	S（状态改变）	新增	Q2A(S1) Q3A Q4A Q4AR
	驱动器 3/4 使用标志	OFF : 未使用 ON : 使用中	• 变为 ON，当驱动器 3（标准 RAM）或者驱动器 4（标准 ROM）内文件被使用时。	S（状态改变）	新增	QnU
SM625	存储卡 B 卸载 / 插入禁止标志	OFF : 卸载 / 插入允许 ON : 卸载 / 插入禁止	• 当存储卡 B 不能插入或者卸载时，变为 ON	U	新增	Q2A(S1) Q3A Q4A Q4AR

附表 .7 特殊继电器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的CPU ACPU M9□□□	相应的 CPU
SM640	文件寄存器使用	OFF ：文件寄存器未使用 ON ：文件寄存器使用中	• 当文件寄存器被使用时，变为 ON	S（状态改变）	新增	○
SM650	注释使用	OFF ：文件寄存器未使用 ON ：文件寄存器使用中	• 当注释文件被使用时，变为 ON	S（状态改变）	新增	QnA Qn(H) QnPH QnPRH QnU
SM660	引导操作	OFF ：内部内存执行 ON ：引导操作进行中	• 当引导操作正在进行时，变为 ON • 如果引导指示开关是 OFF 时，变为 OFF	S（状态改变）	新增	QnA Qn(H) QnPH QnPRH
		OFF ：程序内存执行 ON ：引导操作进行中	• 当引导操作正在进行时，变为 ON	S（状态改变）	新增	Q00J/Q00/Q01 QnU
SM671	锁存数据备份到标准 ROM 结束标志	OFF ：未结束 ON ：结束	• 锁存数据备份到标准 ROM 结束时，变为 ON。 • 锁存数据备份到标准 ROM 的执行时间，存储在 SD672 以后。	S（状态改变）	新增	QnU
SM672	存储卡 A 文件寄存器访问范围标志	OFF ：在访问范围内 ON ：超出访问范围	• 当访问是在存储卡 A 的文件寄存器 R 的范围以外的区域进行时（在 END 处理中设定。），变为 ON • 在用户程序中复位	S/U	新增	QnA Qn(H) QnPH QnPRH
SM673	存储卡 B 文件寄存器访问范围标志	OFF ：在访问范围内 ON ：超出访问范围	• 当访问是在存储卡 B 的文件寄存器 R 的范围以外的区域进行时（在 END 处理中设定。），变为 ON • 在用户程序中复位	S/U	新增	Q2A(S1) Q3A Q4A Q4AR
SM675	锁存数据备份到标准 ROM 结束出错	OFF ：未出错 ON ：出错	• 正常备份锁存数据时，不能备份数据到标准 ROM 中，变为 ON。 • 正常备份锁存数据时，数据备份到标准 ROM 中，变为 OFF。	S	新增	QnU
SM676	恢复反复执行指定	OFF ：未指定 ON ：指定	• SM676 为 ON 时，如果执行锁存数据备份，则每次下次电源 OFF → ON 时恢复数据。 • 删除备份的锁存数据，或每次电源 OFF → ON 时恢复数据，直到锁存数据备份再次执行。	U	新增	QnU
SM680	程序内存写入出错	OFF ：写入出错 ON ：未执行写入 / 正常	• 如果程序内存（闪存 ROM）写入时检测到出错，变为 ON。 写入指示时为 OFF。	S（写入时）	新增	QnU
SM681	程序内存写入标志	OFF ：写入中 ON ：未执行写入	• 正在进行程序内存（闪存 ROM）写入时，变为 ON。写入结束时变为 OFF。	S（写入时）	新增	QnU
SM682	程序内存改写次数出错标志	OFF ：改写次数为 100,000 以后 ON ：改写次数不到 100,000	• 当程序内存（闪存 ROM）改写次数达到 100,000 时，变为 ON。	S（写入时）	新增	QnU
SM685	标准 ROM 写入出错	OFF ：写入出错 ON ：未写入 / 正常	• 如果标准 ROM 写入时检测到出错，变为 ON。 • 写入指示时为 OFF。	S（写入时）	新增	QnU
SM686	标准 ROM 写入标志	OFF ：写入中 ON ：未执行写入	• 正在进行标准 ROM 写入时，变为 ON。写入结束时变为 OFF。	S（写入时）	新增	QnU
SM687	标准 ROM 改写次数出错标志	OFF ：改写次数为 100,000 以后 ON ：改写次数不到 100,000	• 当标准 ROM 改写次数达到 100,000 时，变为 ON。（必需更换 CPU 模块。）	S（写入时）	新增	QnU

(7) 指令相关特殊继电器

附表 .8 特殊继电器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU M9□□□	相应的 CPU
SM700	进位标志	OFF ：进位 OFF ON ：进位 ON	• 用在应用指令中的进位标志	S（指令执行）	M9012	○
SM701	输出字符数切换	切换输出的字符数、输出模式	• 通过 PR、PRC、BINDA、DBINDA、BINHA、DBINHA、BCDDA、DBCDDA 或 COMRD 指令使用。 • 详细内容请参阅 QCPU(Q 模式)/QnACPU 编程手册（公共指令篇）。	U	M9049	QnA Qn(H) QnPH QnPRH QnU
SM702	查找方法	OFF ：查找下一个 ON ：二分法查找	• 指定查找指令使用的方法。 • 必须排列数据，以用于二分法查找。	U	新增	○
SM703	排序类型	OFF ：升序 ON ：降序	• 排序指令用于指定以升序存储数据还是以降序存储数据。	U	新增	
SM704	块比较	OFF ：没有找到匹配项 ON ：所有的匹配项	• 当 BKCMP 指令的所有的数据条件都满足时，变为 ON	S（指令执行）	新增	
SM707	实数指令处理类型选择	OFF ：速度优先 ON ：精度优先	• 当 SM707 是 OFF 时，实数指令以高速执行。 • 当它是 ON 时，实数指令以高精度执行。	U	新增	Q4AR

附表 .8 特殊继电器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU M9□□□	相应的 CPU
SM710	CHK 指令优先等级标志	OFF ：条件优先级 ON ：模式优先级	• 当 OFF 时，保持初始设置。 • 当 ON 时，更新 CHK 优先级。	S（指令执行）	新增	QnA Qn(H) QnPH QnPRH
SM711	分开传送状态	OFF ：分开处理之外 ON ：在分开处理过程中	• 在 AD57(S1) 的处理中，当屏幕被拆分以用于传送时变为 ON，当拆分处理结束时变为 OFF	S（指令执行）	M9065	QnA
SM712	传送处理选择	OFF ：批量处理 ON ：分开处理	• 在 AD57(S1) 的处理中，当防水屏幕被分开以用于传送时变为 ON。	S（指令执行）	M9066	
SM714	通讯请求注册区域 BUSY 信号	OFF ：允许到远程端子模块的通讯请求 ON ：禁止到远程端子模块的通讯请求	• 用于确定是否可以执行连接到 AJ71PT32-S3 的远程端子模块的通讯请求。	S（指令执行）	M9081	
SM715	EI 标志	OFF ：在 DI 过程中 ON ：在 EI 过程中	• 当 EI 指令正被执行时。	S（指令执行）	新增	○
SM720	注释读结束标志	OFF ：注释读未结束 ON ：注释读结束	• 只在 COMRD 或者 PRC 指令的处理结束时，接通一个扫描周期。	S（状态改变）	新增	Qn(H) QnPH
			• 只在 COMRD 指令的处理结束时，接通一个扫描周期。			Qn(H) QnPH QnPRH
SM721	文件正被访问	OFF ：文件未被访问 ON ：文件正被访问	• 当文件正被 S.FWRITE、S.FREAD、COMRD、PRC 或者 LEDC 指令访问时，切换为 ON。	S（状态改变）	新增	Qn(H) QnPH
			• 当文件正被 S.FWRITE、S.FREAD、COMRD 或者 LEDC 指令访问时，切换为 ON。			Qn(H) QnPH QnPRH
			• 当文件正被 S.FWRITE、S.FREAD、COMRD 或者 SP.DEVST 指令访问时，切换为 ON。			QnU
SM722	BIN/DBIN 指令出错禁止标志	OFF ：执行出错检测 ON ：未执行出错检测	• 当为 BIN 或者 DBIN 指令抑制了 “OPERATION ERROR” 时，变为 ON。	U	新增	QCPU
SM730	用于 CC-Link 通讯请求登录区的 BUSY 信号	OFF ：与智能软元件站进行通讯的请求被允许 ON ：与智能软元件站进行通讯的请求被禁止	• 用于判断是允许还是禁止与连接了 CC-Link 模块的智能软元件站进行通讯的请求。	S（执行指令时）	新增	QnA
SM734	XCLL 指令执行条件指定	OFF ：执行条件的上升沿时不执行 ON ：执行条件的上升沿时执行	• OFF 的情况下执行条件的上升沿时不执行 XCLL 指令。 • ON 的情况下执行条件的上升沿时执行 XCLL 指令。	U	新增	Qn(H)*1
SM735	SFC 注释读取指令执行中标志	OFF ：未执行 SFC 注释读取指令 ON ：正在执行 SFC 注释读取指令	• 正在执行 SFC 步注释读取指令 (S(P).SFCSCOMR)、SFC 转移条件注释读取指令 (S(P).SFCTCOMR) 时 ON。	S（状态变化）	新增	Qn(H)*2 QnPH*3 QnPRH*3
SM736	PKEY 指令执行进行中标志	OFF ：指令未执行 ON ：指令执行	• 当 PKEY 指令正在被执行时，变为 ON。 当 CR 是输入或者当输入字符串达到 32 个字符时，变为 OFF。	S（指令执行）	新增	QnA
SM737	用于 PKEY 指令的键盘输入接收标志	OFF ：键盘输入接收允许 ON ：键盘输入接收禁止	• 当键盘输入正在进行时，变为 ON。 当键盘输入已经存储在 CPU 中时，变为 OFF。	S（指令执行）	新增	
SM738	MSG 指令接收标志	OFF ：指令未执行 ON ：指令执行	• 当执行 MSG 指令时，变为 ON。	S（指令执行）	新增	Qn(H) QnPRH

*1: 以序列号的高 5 位为 “06082” 或者更高的 CPU。

附表 .8 特殊继电器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU M9□□□	相应的 CPU
SM774	PID 无冲击处理（用于全微分）	OFF ：匹配 ON ：不匹配	• 指定在手动模式中，设定值 (SV) 是否和过程值 (PV) 匹配。	U	新增	QnA Q00J/Q00/Q01*4 Qn(H) QnPRH QnU
SM775	在 COM 指令执行过程中，选择刷新处理	OFF ：执行链接刷新 ON ：不执行链接刷新	• 在执行 COM 指令时，在只与 CPU 模块进行通讯的情况下，选择是否执行链接刷新处理。	U	新增	QnA Q00J/Q00/Q01 Qn(H) QnPH
		OFF ：执行所有的刷新处理 ON ：执行在 SD778 中设定的刷新	• 当执行 COM 指令时，选择是执行所有的刷新处理还是执行 SD778 设定的刷新处理。	U	新增	Q00J/Q00/Q01*4 Qn(H)*5 QnPH*3 QnPRH
SM776	在 CALL 指令执行时允许 / 禁止局部软元件	OFF ：局部软元件禁止 ON ：局部软元件允许	• 设定 CALL 指令执行时调用的子程序的局部软元件是否有效。	U	新增	QnA Qn(H) QnPH QnPRH QnU
SM777	在中断程序中允许 / 禁止局部软元件	OFF ：局部软元件禁止 ON ：局部软元件允许	• 设定在执行中断程序时局部软元件是否有效。	U	新增	
SM780	CC-Link 专用指令可执行	OFF ：CC-Link 专用指令可执行 ON ：CC-Link 专用指令不可执行	• 当可以同时执行的 CC-Link 专用指令数达到 32 时。切换为 ON，当指令数小于 32 时，切换为 OFF。	U	新增	QnA
SM794	PID 无冲击处理（用于非全微分	OFF ：匹配 ON ：不匹配	• 指定在手动模式中，设定值 (SV) 是否和过程值 (PV) 相匹配。	U	新增	Q00J/Q00/Q01*4 Qn(H)*6 QnPRH QnU

*1: 以序列号的高 5 位为 “06082”以后的 CPU 为对象。
*2: 以序列号的高 5 位为 “07012”以后的 CPU 为对象。
*3: 以序列号的高 5 位为 “07032”以后的 CPU 为对象。
*4: 适用于功能版本 B 以后的 CPU。
*5: 以序列号的高 5 位为 “04012”以后的 CPU 为对象。
*6: 以序列号的高 5 位为 “05032”以后的 CPU 为对象。

附表 .8 特殊继电器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU M9□□□	相应的 CPU
SM796	多 CPU 高速总线的专用指令使用块信息 (1 号 CPU)	OFF ：固定块 ON ：SD796 设置的块不能固定	• 多 CPU 高速总线（目标 CPU=1 号 CPU）的专用指令使用的专用指令传送区剩余块数少于 SD796 指定的块数时，变为 ON。 指令执行时为 ON。当 END 处理存在空余区时，为 OFF。	S（指令 /END 处理执行时）	新增	QnU*7
SM797	多 CPU 高速总线的专用指令使用块信息 (2 号 CPU)	OFF ：固定块 ON ：SD797 设置的块不能固定	• 多 CPU 高速总线（目标 CPU=2 号 CPU）的专用指令使用的专用指令传送区剩余块数少于 SD797 指定的块数时，变为 ON。 指令执行时为 ON。当 END 处理存在空余区时，为 OFF。	S（指令 /END 处理执行时）	新增	QnU*7
SM798	多 CPU 高速总线的专用指令使用块信息 (3 号 CPU)	OFF ：固定块 ON ：SD798 设置的块不能固定	• 多 CPU 高速总线（目标 CPU=3 号 CPU）的专用指令使用的专用指令传送区剩余块数少于 SD798 指定的块数时，变为 ON。 指令执行时为 ON。当 END 处理存在空余区时，为 OFF。	S（指令 /END 处理执行时）	新增	QnU*7
SM799	多 CPU 高速总线的专用指令使用块信息 (4 号 CPU)	OFF ：固定块 ON ：SD799 设置的块不能固定	• 多 CPU 高速总线（目标 CPU=4 号 CPU）的专用指令使用的专用指令传送区剩余块数少于 SD799 指定的块数时，变为 ON。 指令执行时为 ON。当 END 处理存在空余区时，为 OFF。	S（指令 /END 处理执行时）	新增	QnU*7

*7: 除 Q02UCPU 外的通用型 QCPU。

(8) 调试

附表 .9 特殊继电器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU M9□□□	相应的 CPU
SM800	跟踪准备	OFF ：未准备好 ON ：已准备好	• 当跟踪准备完成时，切换为 ON	S（状态改变）	新增	Qn(H) QnPH QnPRH QnU
	采样跟踪准备		• 当采样跟踪准备好时，变为 ON	S（状态改变）	新增	QnA
SM801	跟踪启动	OFF ：中断 ON ：启动	• 当此继电器切换为 ON 时，跟踪被启动。 • 当此继电器切换为 OFF 时，跟踪中断。 （所有的相关特殊 M 切换为 OFF。）	U	M9047	Qn(H) QnPH QnPRH QnU
	采样跟踪启动		• 采样跟踪启动，当此继电器变为 ON 时 • 当此继电器变为 OFF 时（相关特殊 M 都为 OFF），中断	U	M9047	QnA
SM802	跟踪执行进行中	OFF ：中断 ON ：启动	• 在跟踪执行过程中，切换为 ON。	S（状态改变）	M9046	Qn(H) QnPH QnPRH QnU
	采样跟踪执行进行中		• 在采样跟踪执行过程中，变为 ON	S（状态改变）	M9046	QnA
SM803	跟踪触发器	OFF→ON: 启动	• 当此继电器从 OFF 切换到 ON 时（等同于 TRACE 指令执行状态），跟踪被触发器。	U	M9044	Qn(H) QnPH QnPRH QnU
	采样跟踪触发器		• 当此从 OFF 变为 ON 时，（等同于 STRA 指令执行状态）采样跟踪触发器变为 ON。	U	M9044	QnA
SM804	在跟踪触发器之后	OFF ：不在触发器之后 ON ：在触发器之后	• 在跟踪被触发器之后，切换为 ON。	S（状态改变）	新增	Qn(H) QnPH QnPRH QnU
	在采样跟踪触发器之后		• 在采样跟踪触发器之后，变为 ON。	S（状态改变）	新增	QnA

附表 .9 特殊继电器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU M9□□□	相应的 CPU
SM805	跟踪结束	OFF ：未结束 ON ：结束	• 跟踪结束时，切换为 ON	S（状态改变）	M9043	Qn(H) QnPH QnPRH QnU
	采样跟踪结束		• 采样跟踪结束时，变为 ON	S（状态改变）	M9043	QnA
SM806	状态锁存准备	OFF ：未准备好 ON ：准备好	• 当状态锁存是准备好，变为 ON	S（状态改变）	新增	
SM807	状态锁存命令	OFF→ON：锁存	• 运行状态锁存命令	U	新增	
SM808	状态锁存结束	OFF ：锁存未结束 ON ：锁存结束	• 当状态锁存结束时，变为 ON	S（状态改变）	M9055	
SM809	状态锁存清除	OFF→ON：清除	• 允许下一个状态锁存	U	新增	
SM810	程序跟踪准备	OFF ：未准备好 ON ：准备好	• 当程序跟踪是准备好，变为 ON	S（状态改变）	新增	
SM811	启动程序跟踪	OFF ：中断 ON ：启动	• 当此继电器变为 ON 时，程序跟踪启动 • 当此继电器变为 OFF 时（相关特殊 M 都为 OFF），中断	S（状态改变）	新增	
SM812	在程序跟执行进行中	OFF ：中断 ON ：启动	• 当程序跟踪执行正在进行时，变为 ON	U	新增	
SM813	程序跟踪触发器	OFF→ON：启动	• 当此继电器从 OFF 变为 ON 时（等同于 PTRA 指令执行状态），程序跟踪触发器变为 ON	S（状态改变）	新增	
SM814	程序跟踪触发器	OFF ：不在触发之后 ON ：在触发之后	• 在程序跟踪触发器之后，变为 ON	S（状态改变）	新增	
SM815	程序跟踪结束	OFF ：未结束 ON ：结束	• 在程序跟踪结束时，变为 ON	S（状态改变）	新增	
SM820	步跟踪准备	OFF ：未准备好 ON ：准备好	• 在程序跟踪注册准备好时，变为 ON	S（状态改变）	新增	QnA
SM821	步跟踪启动	OFF ：中断 ON ：启动	• 选择是启动还是中断步跟踪的执行。 • 当此继电器变为 ON，步跟踪被启动 • 当 OFF 时中断（相关特殊 M 都为 OFF）	U	M9182 格式改变	
SM822	步跟踪执行进行中	OFF ：中断 ON ：启动	• 当步跟踪执行正在进行时，变为 ON • 在结束或者挂起时，变为 OFF	S（状态改变）	M9181	
SM823	步跟踪触发之后	OFF ：不在触发之后 ON ：在第一个触发之后	• 只要在正被执行的步跟踪内有 1 块被触发时，变为 ON • 当步跟踪开始时，变为 OFF	S（状态改变）	新增	
SM824	步跟踪触发后	OFF ：有触发未成立的块 ON ：所有块均触发成立	• 如果正在执行步跟踪的所有的块都触发成立，则变为 ON。 • 当步跟踪开始时，变为 OFF。	S（状态改变）	新增	
SM825	步跟踪结束	OFF ：未结束 ON ：结束	• 在步跟踪结束时，变为 ON • 当步跟踪开始时，变为 OFF	S（状态改变）	M9180	
SM826	跟踪出错	OFF ：正常 ON ：出错	• 如果出错发生在跟踪执行过程中，切换为 ON	S（状态改变）	新增	Qn(H) QnPH QnPRH QnU
	采样跟踪出错		• 如果出错发生在采样跟踪执行过程中，变为 ON	S（状态改变）	新增	QnA
SM827	状态锁存出错	OFF ：正常 ON ：出错	• 如果出错发生在状态锁存的执行过程中，变为 ON	S（状态改变）	新增	
SM828	程序跟踪出错	OFF ：正常 ON ：出错	• 如果出错发生在程序跟踪的执行过程中，变为 ON	S（状态改变）	新增	
SM829	跟踪设置强制登录指定	OFF ：允许强制登录 ON ：不允许强制登录	• SM829 为 ON 后，通过 GX Developer 登录采样跟踪设置，即使跟踪条件或触发条件成立的情况下，也可以将采样跟踪设置设置到 QnUCPU 中。	U	新增	QnU

(9) 锁存区

表附 .10 特殊继电器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU M9□□□	相应的 CPU
SM900	电源关断文件	OFF ：无电源关断文件 ON ：电源关断文件存在	• 如果电源关断在访问过程中有文件存在，变为 ON。	S（状态改变）/U	新增	QnA
SM910	RKEY 注册标志	OFF ：键盘输入未注册 ON ：键盘输入注册	• 注册键盘输入时，变为 ON。 • 如果键盘输入未注册时，OFF。	S（指令执行）	新增	

(10)A → Q/QnA 转换的对应

ACPU 的特殊继电器 M9000 ～ M9255 在通过 A → Q/QnA 转换进行了转换之后，所对应的特殊继电器为 SM1000 ～ SM1255。

（但是，基本型 QCPU、冗余 CPU 和通用型 QCPU 不支持 A → Q/QnA 转换。）

这些特殊继电器都是由系统设定的，用户程序不能设定它们。

要使用用户程序将它们变为 ON/OFF，请将程序中的特殊继电器更换为 QCPU/QnACPU 的相应继电器。

但是，SM1084 和 SM1200 到 SM1255（对应转换前的 M9084 和 M9200 到 M9255）中的一些继电器不能由用户程序变为 ON/OFF，即使在转换之前它们可以由用户程序变为 ON/OFF 的话。关于 ACPU 特殊继电器的详细信息，见单独 CPU 的用户手册，和 MELSECNET 或者 MELSECNET/B 数据链接系统参考手册。

要 点

当转换后的特殊继电器用于高性能型 QCPU 和过程控制 CPU 时，处理时间可能会变长。当转换后的特殊继电器未被使用时，在 GX Developer 参数的 PC 系统设置内，取消对 “A-series CPU compatibility setting”的选择。

备 注

下面是对 “用于改进的特殊继电器”栏中，特殊继电器的附加解释。

1. 当提供了用于改进的特殊继电器时，软元件号应该改为提供的 QCPU/QnACPU 特殊继电器。
2. 当提供了□时，转换后的特殊继电器可以用于此软元件号。
3. 当提供了☒时，此软元件号不能和 QCPU/QnACPU 一起工作。

在高性能模式 QCPU 以及过程 CPU 中使用转换后的特殊继电器时，需要耗费处理时间。在不使用转换后的特殊继电器时，应在 GX Developer 的可编程控制器参数的可编程控制器系统设置中将 “A 系列 CPU 兼容设置”的勾选取消。

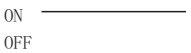
附表 .11 特殊继电器列表

ACPU 特殊继电器	转换后的特殊继电器	用于改进的特殊继电器	名称	含义	详细信息	相应的 CPU
M9000	SM1000	－	保险丝熔断	OFF ：正常 ON ：模块有熔断的保险丝	• 当有一个或者多个保险丝已熔断的输出模块时，接通。 • 如果自此以后情况恢复为正常，仍保持为 ON。 • 对远程 I/O 站，也检查输出模块的保险丝情况。	QnA Qn(H) QnPH
M9002	SM1002	－	I/O 模块验证出错	OFF ：正常 ON ：出错	• 如果 I/O 模块的状态和电源上电时输入的状态不同，接通。 • 如果自此以后情况恢复为正常，保持为 ON。 • 对远程 I/O 站模块，也进行 I/O 模块校验。 • 只有当特殊寄存器 SD1116 到 SD1123 被复位时，复位才被允许。	
M9004	SM1004	－	NIMI 链接出错	OFF ：正常 ON ：出错	• 即使只在安装的 AJ71PT32(S3) 模块中的一个检测到 MINI(S3) 链接出错，变为 ON。 • 如果自此以后情况恢复为正常，仍保持为 ON。	QnA

附表 .11 特殊继电器列表

ACPU 特殊继电器	转换后的特殊继电器	用于改进的特殊继电器	名称	含义	详细信息	相应的 CPU
M9005	SM1005	–	AC DOWN 检测	OFF : AC DOWN 未检测到 ON : AC DOWN 检测到	• 如果在使用 AC 电源模块的过程中发生 20ms 以内的瞬间掉电，变为 ON。 • 当电源被切换到 OFF，然后再切换为 ON 时，复位。	QnA Qn(H) QnPH
					• 如果在使用 DC 电源模块的过程中发生 10ms 以内的瞬间掉电，变为 ON。 • 当电源是被切换到 OFF，然后再切换为 ON 时，复位。	Qn(H) QnPH
					• 如果在使用 DC 电源模块的过程中发生 1ms 以内的瞬间掉电，变为 ON。 • 当电源是被切换到 OFF，然后再切换为 ON 时，复位。	QnA
M9006	SM1006	–	电池电量不足	OFF : 正常 ON : 电池电量不足	• 当电池电压下降到或者低于指定值时，变为 ON。 • 当电池电压恢复为正常，变为 OFF。	QnA Qn(H) QnPH
M9007	SM1007	–	电池电量不足锁存	OFF : 正常 ON : 电池电量不足	• 当电池电压下降到或者低于指定值时，变为 ON。 • 如果电池电压恢复为正常，仍保持为 ON。	
M9008	SM1008	SM1	自检测出错	OFF : 没有出错 ON : 出错	• 当作为自检测的结果发现出错时，接通。	
M9009	SM1009	SM62	报警器检测	OFF : 无 F 号检测到 ON : 有 F 号检测到	• 当 SETF 指令的的 OUTF 被执行时，接通。 • 当 SD1124 数据是零时，切换为关断。	
M9011	SM1011	SM56	操作出错标志	OFF : 没有出错 ON : 出错	• 当操作出错发生在应用指令的执行过程中时，接通。 • 如果自此以后情况恢复为正常，仍保持为 ON。	
M9012	SM1012	SM700	进位标志	OFF : 进位 OFF ON : 进位 ON	• 用在应用指令中的进位标志。	
M9016	SM1016	×	数据内存清除标志	OFF : 忽略 ON : 输出清除	• 当 SM1016 是 ON 时，从计算机等设备上以远程运行模式清除包括锁存范围（特殊继电器和特殊寄存器以外的其他软元件）在内的数据内存。	
M9017	SM1017	×	数据内存清除标志	OFF : 忽略 ON : 输出清除	• 当 SM1017 是 ON 时，从计算机等设备上以远程运行模式清除未锁存的数据内存（特殊继电器和特殊寄存器以外的其他软元件）。	
M9020	SM1020	–	0 号用户定时时钟	n2个扫描周期 n2个扫描周期 n1个扫描周期	• 在预先定义的扫描周期的间隔上重复 ON/OFF 的继电器。 • 当电源接通或者执行复位时，时钟从 OFF 启动。 用 DUTY 指令设定 ON/OFF 的间隔。 DUTY n1 n2 SM1020 n1: ON 扫描周期间隔 n2: OFF 扫描周期间隔	
M9021	SM1021	–	1 号用户定时时钟			
M9022	SM1022	–	2 号用户定时时钟			
M9023	SM1023	–	3 号用户定时时钟			
M9024	SM1024	–	4 号用户定时时钟			
M9025	SM1025	–	时钟数据设置请求	OFF : 忽略 ON : 设置请求存在	• SM1025 从 OFF 变为 ON 的扫描周期中，在 END 指令执行之后，将存储在 SD1025 到 SD1028 中的时钟数据写到 CPU 模块。	
M9026	SM1026	–	时钟数据出错	OFF : 没有出错 ON : 出错	• 由时钟数据 (SD1025 到 SD1028) 出错接通	
M9027	SM1027	–	时钟数据显示	OFF : 忽略 ON : 显示	• 从 SD1025 到 SD1028 中读时钟数据，月、日、小时、分钟和秒显示在 CPU 模块前面的 LED 显示器上。	Q3A Q4A Q4AR
M9028	SM1028	–	时钟数据读请求	OFF : 忽略 ON : 读取请求	• 当 SD1028 是 ON 时，以 BCD 格式读时钟数据到 SD1025 到 SD1028 中。	QnA Qn(H) QnPH
M9029	SM1029	×	数据通讯请求的批量处理	OFF : 批量处理未执行 ON : 批量处理执行	• SM1029 继电器接通，在此扫描周期的 END 处理中，使用顺控程序去处理在此扫描周期中接收到的所有数据通讯请求。 • 数据通讯请求的批量处理可以在运行过程中接通和关断。 • 默认是 OFF（按照数据通讯请求的接收顺序，每个 END 处理一次执行一个）。	
M9030	SM1030	–	0.1 秒时钟	0.05s 0.05s	• 生成 0.1 秒，0.2 秒，1 秒和 2 秒时钟。 • 不是在每个扫描周期都接通或者关断，但是即使在扫描周期过程中，如果相应时间已经用完，也会接通或者关断。 • 当可编程控制器电源接通或者执行 CPU 模块复位时，从 OFF 启动。	
M9031	SM1031	–	0.1 秒时钟	0.1s 0.1s		
M9032	SM1032	–	1 秒时钟	0.5s 0.5s		
M9033	SM1033	–	2 秒时钟	1s 1s		

附表 .11 特殊继电器列表

ACPU 特殊继电器	转换后的特殊继电器	用于改进的特殊继电器	名称	含义	详细信息	相应的 CPU
M9034	SM1034	–	2n 分钟时钟 (1 分钟时钟)*		• 按照在 SD414 钟指定的秒数，在 ON 和 OFF 之间交替。(默认 :n=30) • 不是在每个扫描周期都接通或者关断，但是即使在在扫描周期过程中，如果相应时间已经用完，也会接通或者关断。 • 当可编程控制器电源接通或者执行 CPU 模块复位时，以 OFF 启动。	
M9036	SM1036	–	常 ON		• 被用作顺控程序中初始化和应用指令的虚拟触点。 • SM1038 和 SM1037 被接通和关断，而不管 CPU 模块前面的钥匙开关的位置如何。如果钥匙开关处于停止位置 ，则切换为关断。如果钥匙开关不在停止位置 S，M1038 只接通一个扫描周期，而 SM1039 只关断一个扫描周期。	
M9037	SM1037	–	常 OFF			
M9038	SM1038	–	只 ON1 个扫描周期在 RUN 之后			
M9039	SM1039	–	RUN 标志 (在 RUN 之后，只 OFF1 个扫描周期)			
M9040	SM1040	SM206	PAUSE 状态触点	OFF : PAUSE 禁止 ON : PAUSE 允许	• 当 RUN 钥匙开关在 PAUSE 位置或者 PAUSE 触点已接通，并且如果 SM204 是 ON，则 PAUSE 模式被设定，并且 SM206 接通。	
M9041	SM1041	SM204	USE 状态触点	OFF : PAUSE 无效 ON : PAUSE 有效		
M9042	SM1042	SM203	STOP 状态触点	OFF : STOP 无效 ON : STOP 有效	• 当 RUN 钥匙开关或者 RUN/STOP 开关位于停止位置，接通。	
M9043	SM1043	SM805	采样跟踪结束	OFF : 采样跟踪进行中 ON : 采样跟踪结束	• 在 STRA 指令执行之后，采样跟踪执行了参数预设的次数时接通。 复位当 STRAR 指令执行时。	QnA Qn(H) QnPH
M9044	SM1044	SM803	采样跟踪	OFF→ON STRA 和执行相同 ON→OFF STRAR 和执行相同	• 接通 / 关断 SM803 可以执行 STRA STRA/STRAR 指令。 (SM803 是由外围设备强制接通 / 关断的。) 当从 OFF 切换到 ON: STRA 指令 当从 ON 切换到 OFF: STRAR 指令 存储在 SD1044 中的值被用作采样跟踪的条件。 在扫描时，时间 → 时间 (10ms 为单位)	
M9045	SM1045	×	看门狗定时器 (WDT) 复位	OFF : 不复位 WDT ON : 复位 WDT	• SM1015 继电器接通以复位 WDT，当 ZCOM 指令和数据通讯请求批量处理被执行 (当扫描时间超出 200ms 时被使用)。	
M9046	SM1046	SM802	采样跟踪	OFF : 跟踪未进行 ON : 跟踪进行中	• 在采样跟踪过程中切换为 ON。	
M9047	SM1047	SM801	采样跟踪准备	OFF : 采样跟踪中断 ON : 采样跟踪启动	• 采样跟踪未执行，除非 SM801 变为 ON。 当 SM801 变为 OFF 时，采样跟踪被中断。	
M9049	SM1049	SM701	切换字符输出数	OFF : 输出直到碰到 NULL 代码 ON : 16 字符输出	• 当 SM701 是 OFF 时，输出字符直到碰到 NULL(00H) 代码。 • 当 SM701 是 ON 时，输出 16 字符的 ASCII 代码。	
M9051	SM1051	×	CHG 指令执行禁止	OFF : 允许 ON : 禁止	• 切换为 ON 以禁止 CHG 指令。 • 切换为 ON，当程序发送被请求时。 当发送结束时自动切换到 OFF。	
M9052	SM1052	×	SEG 指令开关	OFF : 7SEG 段显示 ON : I/O 部分刷新	• 当 SM1052 是 ON 时，SEG 指令以 I/O 部分刷新指令执行。 当 SM1052 是 OFF 时，SEG 指令以 7-SEG 显示指令执行。	
M9054	SM1054	SM205	STEP RUN 标志	OFF : SETP RUN 无效 ON : SETP RUN 有效	• 当 RUN 钥匙开关处于 SETPRUN 位置，切换为 ON。	QnA
M9055	SM1055	SM808	状态锁存结束标志	OFF : 未结束 ON : 结束	• 当状态锁存结束时，接通。 • 由复位指令关闭。	

*: 1 分钟时钟表示 ACPU 的特殊继电器 (M9034) 的名称。

附表 .11 特殊继电器列表

ACPU 特殊继电器	转换后的特殊继电器	用于改进的特殊继电器	名称	含义	详细信息	相应的 CPU
M9056	SM1056	×	主端 P、I 设置请求	OFF ：P、I 设置正被请求以外的情况 ON ：P、I 设置正被请求	• 在 RUN 过程中，当其他程序的发送（例如，当主程序正在运行时的子程序）结束之后提供 P、I 设置请求。当 P、I 设置结束后自动被切换到 OFF。	QnA Qn(H) QnPH
M9057	SM1057	×	从端 P、I 设置请求	OFF ：P、I 设置正被请求以外的情况 ON ：P、I 设置正被请求		
M9058	SM1058	×	主端 P、I 设置结束	在 P、I 设置结束时立即为 ON	• 当 P、I 设置结束时立即变为 ON，然后重新变为 OFF。	
M9059	SM1059	×	子程序 P、I 设置结束	在 P、I 设置结束时立即为 ON		
M9060	SM1060	×	子程序 2P、I 设置请求	OFF ：P、I 设置正被请求之外的情况 ON ：P、I 设置正被请求	• 在 RUN 过程中，当其他程序的发送（例如，当主程序正在运行时的子程序）结束之后提供 P、I 设置请求。当 P、I 设置结束后自动被切换到 OFF。	
M9061	SM1061	×	子程序 3P、I 设置请求	OFF ：P、I 设置正被请求之外的情况 ON ：P、I 设置正被请求		
M9065	SM1065	SM711	分开发送状态	OFF ：分开处理没有进行 ON ：在分开处理过程中	• 当通过分开处理执行到 AD57(S1)/AD58 的防水屏幕发送时接通，在分开处理结束时关断	QnA
M9066	SM1066	SM712	发送处理切换	OFF ：批量处理 ON ：分开处理	• 当到 AD57(S1)/AD58 的防水屏幕发送是通过分开处理执行时接通。	
M9070	SM1070	×	A8UPU/A8PUJ 需要的查找时间 *	OFF ：读取时间没有缩短 ON ：读取时间缩短	• 变为 ON 以缩短在 A8UPU/A8PUJ 中的查找时间。（在这种情形中，扫描时间增加 10 %.）	QnA Qn(H) QnPH
M9081	SM1081	SM714	通讯请求注册区域 BUSY 信号	OFF ：通讯请求注册区域中的空余空间 ON ：在通讯请求注册区域中无空余空间	• 与连接到 AJ71PT32-S3、A2C 或者 A52G 的远程端子模块通讯允许 / 禁止的指示。	QnA
M9084	SM1084	×	出错检查	OFF ：出错检查执行 ON ：没有出错检查	它设定当 END 指令被处理时是否执行下面的出错检查（设定 END 指令处理时间）。 • 检查保险丝是否断开。 • 检查电池 • I/O 模块的验证检查	QnA Qn(H) QnPH
M9091	SM1091	×	指令出错标志	OFF ：没有出错 ON ：出错	• 当操作出错的详细因素被存储到 SD1091 中时，变为 ON。 • 如果自此以后情况恢复为正常，保持为 ON。	
M9094	SM1094	SM251	I/O 更换标志	OFF ：更换 ON ：无更换	• 在将要被更换的 I/O 模块的起始 I/O 号码被设定到 SD251 之后，当 SM251 变为 ON 时，I/O 模块可以在运行（电源接通）中更换。（一个设置只允许更换一个模块。） • 在程序或者外围设备测试模式中切换为接通，以便在 CPU 为 RUN 过程中更换模块。在外围设备测试模式中切换到接通以便在 CPU 停止过程中更换模块。 • 在 I/O 模块更换结束之前，不能改变 RUN/STOP 模式。	QnA
M9100	SM1100	SM320	SFC 程序存在 / 不存在	OFF ：SFC 程序被使用 ON ：SFC 程序停止	• 如果 SFC 程序被注册，接通。 • 如果没有注册，关断。	QnA Qn(H) QnPH
M9101	SM1101	SM321	启动 / 停止 SFC 程序	OFF ：SFC 程序停止 ON ：SFC 程序启动	• SM320 中的值被设置为初始值。（当 SFC 程序存在时，继电器自动变为 ON。） • 当此继电器从 ON 变为 OFF 时，SFC 程序的执行停止。 • 当此继电器从 OFF 变为 ON 时，恢复 SFC 程序的执行。	
M9102	SM1102	SM322	SFC 程序启动程序	OFF ：初始化启动 ON ：继续运行	• 可编程控制器参数对话框的 SFC 设置中的 SFC 程序启动模式被设定为初始值。 在初始化启动时：OFF 在连续启动：ONx	

*: A8UPU/A8PUJ 不能用于 QCPU/QnACPU。

附表 .11 特殊继电器列表

ACPU 特殊继电器	转换后的特殊继电器	用于改进的特殊继电器	名称	含义	详细信息	相应的 CPU
M9103	SM1103	SM323	存在 / 不存在连续转移	OFF : 连续转移无效 ON : 连续转移有效	• 设定在没有设定 SFC 信息软元件的 “continuous transition bit” 的地方是否执行块的连续转移。	QnA Qn(H) QnPH
M9104	SM1104	SM324	连续转移挂起标志	OFF : 当转移结束时 ON : 当没有转移时	• 在运行在连续转移模式或者在连续转移过程中为 OFF，当连续转移没有执行时为 ON。 • 常 ON，当运行在无连续转移模式中时。	
M9108	SM1108	SM90	步转移警戒狗定时器启动（等同于 SD90）	OFF : 警戒狗定时器复位 ON : 警戒狗定时器复位启动	• 当步转移警戒狗定时器的测量启动时，变为 ON。 将此继电器置为 OFF，以复位步转移警戒狗定时器。	
M9109	SM1109	SM91	步转移警戒狗定时器启动（等同于 SD91）			
M9110	SM1110	SM92	步转移警戒狗定时器启动（等同于 SD92）			
M9111	SM1111	SM93	步转移警戒狗定时器启动（等同于 SD93）			
M9112	SM1112	SM94	步转移警戒狗定时器启动（等同于 SD94）			
M9113	SM1113	SM95	步转移警戒狗定时器启动（等同于 SD95）			
M9114	SM1114	SM96	步转移警戒狗定时器启动（等同于 SD96）			
M9180	SM1180	SM825	活动步采样跟踪结束标志	OFF : 跟踪启动 ON : 跟踪结束	• 当所有指定块的采样跟踪结束时设置，当采样跟踪启动时复位。	QnA
M9181	SM1181	SM822	活动步采样跟踪执行标志	OFF : 没有正在执行跟踪 ON : 跟踪执行执行中	• 当采样跟踪正在执行时置位。当采样跟踪结束或者挂起时复位。	
M9182	SM1182	SM821	活动步采样跟踪允许	OFF : 跟踪禁止 / 挂起 ON : 跟踪允许	• 选择采样跟踪执行允许 / 禁止。 ON : 允许采样跟踪执行。 OFF : 禁止采样跟踪执行。如果在采样跟踪执行过程中关断，跟踪被挂起。	
M9196	SM1196	SM325	操作输出块停止	OFF : 线圈输出 OFF ON : 线圈输出 ON	• 选择执行块停止时的操作输出。 ON : 通过使用在块停止时正被执行的步的操作输出，保持正被使用的线圈的 ON/OFF 状态。 OFF : 所有的线圈输出关断。(SET 指令的操作输出被保留，而不管 M9196 的 ON/OFF 状态如何。)	QnA Qn(H) QnPH
M9197	SM1197	×	在保险丝熔断和 I/O 验证出错显示之间切换	SM1197 / SM1198 / 要显示的 I/O 号 OFF / OFF / X/Y0 到 7F0 ON / OFF / X/Y800 到 FF0 OFF / ON / X/Y1000 到 17F0 ON / ON / X/Y1800 到 1FF0	依据 SM1197 和 SM1198 的 ON/OFF 的组合，在保险丝熔断模块存储寄存器 (SD1100 到 SD1107) 和 I/O 模块验证出错存储寄存器 (SD1116 到 SD1123) 中的 I/O 号之间切换。	
M9198	SM1198	×				
M9199	SM1199	×	运行中采样跟踪 / 状态锁存的数据恢复	OFF : 数据恢复禁止 ON : 数据恢复允许	• 当执行采样跟踪 / 状态锁存时，在重启时恢复存储在 CPU 模块中的设置数据。 • SM1199 应该为 ON 以便重新执行。（当重新从外围设备写数据时不需要。）	

附表 .11 特殊继电器列表

ACPU 特殊继电器	转换后的特殊继电器	用于改进的特殊继电器	名称	含义	详细信息	相应的 CPU
M9200	SM1200	–	ZNRD 指令 (LRDP 指令用于 ACPU) 接收 (用于主站)	OFF : 未接收 ON : 接收	• 取决于是否接收到 ZNRD(字软元件读) 指令。 • 在程序用作 ZNRD 指令的互锁。 • 使用 RST 指令去复位。	QnA
M9201	SM1201	–	ZNRD 指令 (LRDP 指令用于 ACPU) 结束 (用于主站)	OFF : 未结束 ON : 结束	• 取决于 ZNRD(字软元件读) 指令执行是否结束。 • 在 ZNRD 指令结束之后，用作复位 M9200 和 M9201 的条件触点。 • 使用 RST 指令去复位。	
M9202	SM1202	–	ZNWR 指令 (LWTP 指令用于 ACPU) 接收 (用于主站)	OFF : 未接收 ON : 接收	• 取决于 ZNWR(字软元件写) 指令是否已经被接收。 • 在程序中用作用于 ZNWR 指令的互锁。 • 使用 RST 指令去复位。	
M9203	SM1203	–	ZNWR 指令 (LWTP 指令用于 ACPU) 结束 (用于主站)	OFF : 未接收 ON : 结束	• 取决于 ZNWR(字软元件写) 指令执行是否结束。 • 在 ZNWR 指令结束之后，用作复位 M9202 和 M9203 的条件触点。 • 使用 RST 指令去复位。	
M9204	SM1204	–	ZNRD 指令 (LRDP 指令用于 ACPU) 接收 (用于本地站)	OFF : 未接收 ON : 结束	• 接通表示 ZNRD 指令在本地站结束。	
M9205	SM1205	–	ZNWR 指令 (LWTP 指令用于 ACPU) 接收 (用于本地站)	OFF : 未结束 ON : 结束	• 接通表示 ZNWR 指令在本地站结束。	
M9206	SM1206	–	主机站链接参数出错	OFF : 正常 ON : 异常	• 取决于主机的链接参数设置是否有效。	
M9207	SM1207	–	链接参数检查结果	OFF : 匹配 ON : 不匹配	• 取决于两级系统中第二级主站的链接参数设置是否和三级系统中第三级主站的链接参数设置相匹配。(只对三级系统中的主站有效。)	
M9208	SM1208	–	设置主站 B 和 W 传送范围 (只用于较低级链接主站)	OFF : 发送到第 2 级和第 3 级 ON : 只发送到第 2 级	• 取决于高一级链接主站 (本站) 控制的 B 和 W 数据是否被发送到低一级链接本地站 (第三级站)。 当 SM1208 是 OFF ..... B 和 W 的本站被发送到第三级站。 当 SM1208 是 ON ..... 和 W 的本站不发送到第三级站。	
M9209	SM1209	–	链接参数检查命令 (只用于低一级链接主站)	OFF : 执行检查功能 ON : 检查不执行	• 设为 ON，以便不匹配高一级和低一级链接的 B 和 W。 • 当 SM1209 是 ON，不检查高一级和低一级链接的链接参数。 • 当 SM1209 是 OFF，检查高一级和低一级链接的链接参数。	
M9210	SM1210	–	链接卡出错 (用于主站)	OFF : 正常 ON : 异常	控制是否执行取决于链接卡硬件是否有故障。	
M9211	SM1211	–	链接模块出错 (用于本地站使用)	OFF : 正常 ON : 异常	控制是否执行取决于链接卡硬件是否有故障。	
M9224	SM1224	–	链接状态	OFF : 运行中 ON : 离线，站到站的测试，或者自环路回送测试	取决于主站是运行中还是离线，或者是处于站到站的测试或自环路回送测试模式中。	
M9225	SM1225	–	正向环路出错	OFF : 正常 ON : 异常	取决于正向环路电缆的出错条件。	
M9226	SM1226	–	反向环路出错	OFF : 正常 ON : 异常	取决于反向环路电缆的出错条件。	
M9227	SM1227	–	环路测试状态	OFF : 未执行 ON : 正向或者反向环路测试执行中	取决于主站是执行正向环路测试，还是反向环路测试。	
M9232	SM1232	–	本地站操作状态	OFF : RUN 或者 STEP RUN 状态 ON : STOP 或者 PAUSE 状态	依据本地站处于 STOP 或者 PAUSE 模式，来执行控制。	
M9233	SM1233	–	本地站出错检测状态	OFF : 没有出错 ON : 出错检测	取决于本地站是否在其他站中检测到出错。	

附表 .11 特殊继电器列表

ACPU 特殊继电器	转换后的特殊继电器	用于改进的特殊继电器	名称	含义	详细信息	相应的 CPU
M9235	SM1235	–	本地站，远程 I/O 站参数出错检测状态	OFF ：没有出错 ON ：出错检测	取决于本地或者远程 I/O 站是否在主站检测到任何链接参数出错。	QnA
M9236	SM1236	–	本地站，远程 I/O 站初始化通讯状态	OFF ：无通讯 ON ：通讯进行中	取决于本地或者远程 I/O 站和主站初始化通讯的结果。（参数通讯等。）	
M9237	SM1237	–	本地站，远程 I/O 站出错	OFF ：正常 ON ：异常	取决于本地或者远程 I/O 站的出错条件。	
M9238	SM1238	–	本地站， 远程 I/O 站正向或者反向环路出错	OFF ：正常 ON ：异常	取决于本地或者远程 I/O 站的正向和反向环路电缆的出错条件。	
M9240	SM1240	–	链接状态	OFF ：运行中 ON ：离线，站到站测试，或者自环路回送测试	取决于本地站是运行中还是处于离线，或者是处于站到站测试或自环路回送测试模式中。	
M9241	SM1241	–	正向环路电缆出错	OFF ：正常 ON ：异常	取决于正向环路电缆的出错条件。	
M9242	SM1242	–	反向环路电缆出错	OFF ：正常 ON ：异常	取决于反向环路电缆的出错条件。	
M9243	SM1243	–	环路回送实现	OFF ：环路回送未执行 ON ：环路回送实现	取决于本地站上是否正在发生环路回送。	
M9246	SM1246	–	数据未接收	OFF ：接收 ON ：未接收	取决于数据是否已经从主站中接收到。	
M9247	SM1247	–	数据未接收	OFF ：接收 ON ：未接收	取决于在三级系统，第三级站是否已经从它的主站接收到数据。	
M9250	SM1250	–	参数未接收	OFF ：接收 ON ：未接收	取决于是否已经从主站接收到链接参数。	
M9251	SM1251	–	链接继电器	OFF ：正常 ON ：异常	取决于本地站的数据链接情况。	
M9252	SM1252	–	环路测试状态	OFF ：未执行 ON ：正向或者反向环路测试执行进行中	取决于本地站是否正在执行正向或者反向环路测试。	
M9253	SM1253	–	主站操作状态	OFF ：RUN 或者 STEP RUN 状态 ON ：STOP 或者 PAUSE 状态	控制是否执行，取决于主站是处于 STOP 模式还是处于 PAUSE 模式。	
M9254	SM1254	–	本站以外的本地站操作状态	OFF ：RUN 或者 STEP RUN 状态 ON ：STOP 或者 PAUSE 状态	控制是否执行，取决于本站以外的主站是处于 STOP 模式还是处于 PAUSE 模式。	
M9255	SM1255	–	本站以外的本地站出错	OFF ：正常 ON ：异常	取决于本站以外的本地站是否处于出错状态。	

(11)过程控制控制指令

附表 .12 特殊继电器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU M9□□□	相应的 CPU
SM1500	保持模式	OFF ：不保持 ON ：保持	• 指定当 S.IN 指令范围检查发生超限时是否保持输出值。	U	新增	Q4AR QnPH QnPRH
SM1501	保持模式	OFF ：不保持 ON ：保持	• 指定当 S.OUT 指令范围检查发生超限时是否保持输出值。	U	新增	

(12)对于冗余系统（本站系统 CPU 信息 *1）

SM1510 到 SM1599 只对冗余系统有效。
对单独系统，所有的继电器都为 OFF。.

附表 .13 特殊继电器列表

号码	名称	含义	解释	设置方式（当被设定时	相应的 ACPU M9 □□□	相应的 CPU
SM1510	运行模式	OFF ：冗余系统备用模式，独立系统 ON ：冗余系统独立模式	• 当运行模式是冗余系统分开时，变为 ON。	S（每次 END）	新增	Q4AR QnPRH
SM1511	电源接通时的启动模式	OFF ：系统固定模式 ON ：先前控制系统锁存模式	• 当当电源变为 ON 时冗余系统的启动模式是先前控制系统锁存模式时，变为 ON。	S（初始化）	新增	Q4AR
SM1512	CPU 启动时的启动模式	OFF ：初始化启动 ON ：热启动	• 当冗余系统被启动时，CPU 模块的运行模式是热启动时，变为 ON。	S（初始化）	新增	
SM1511	系统 A 识别标志	• 区分系统 A 和系统 B。 • 如果热备电缆断开，系统识别被保持在先前的状态。 （内嵌表格见下）		S（初始化）	新增	QnPRH
SM1512	系统 B 识别标志					
SM1513	CPU 启动时操作状态的	OFF ：初始化启动 ON ：热启动	• 当冗余系统实际启动时，CPU 模块的运行模式是热启动时，变为 ON。	S（初始化）	新增	Q4AR
	调试模式状态标志	OFF ：不处于调试模式 ON ：调试模式	• 当冗余系统运行模式被设定到调试模式，变为 ON。	S（初始化）	新增	QnPRH
SM1514	CPU 模块更换时的运行模式	OFF ：初始化启动 ON ：热启动	• 当为冗余系统进行 CPU 模块运行切换，操作是热启动时，变为 ON。	S（初始化）	新增	Q4AR
SM1515	输出保持模式	OFF ：输出复位 ON ：输出保持	• 当在停止出错过程中输出模式是输出保持时，变为 ON。	S（每次 END）	新增	
SM1516	操作系统状态	OFF ：控制系统 ON ：待机系统	• 当 CPU 模块运行系统状态是待机系统时，变为 ON。	S（状态改变）	新增	
SM1515	控制系统判断标志	• 表示运行系统状态。 • 如果在冗余系统运行过程中，跟踪电缆断开，此标志状态不变。 （内嵌表格见下）		S（状态改变）	新增	QnPRH
SM1516	待机系统判断标志					

	系统A	系统B	当TRK.CABLEERR.发生时（出错代码：6210）发生（未知）
SM1511	ON	OFF	OFF
SM1512	OFF	ON	OFF

	控制系统	待机系统	当TRK.CABLE ERR.发生时（出错代码：6210）发生（未知）
SM1515	ON	OFF	OFF
SM1516	OFF	ON	OFF

*1： 本站 CPU 模块的信息被存储。

附表 .13 特殊继电器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU M9 □□□	相应的 CPU
SM1517	CPU 模块启动状态	OFF ：电源接通启动 ON ：操作系统开关启动	• 当 CPU 模块是操作系统开关启动时，变为 ON。 • 使用用户程序复位。	S（状态改变）/U	新增	Q4AR
			• 当 CPU 模块是由系统切换启动（从待机系统切换到控制系统），变为 ON。当待机系统是通过电源接通启动而被切换到控制系统，保持为 OFF。	S（状态改变）	新增	QnPRH
SM1518	跟踪执行模式	OFF ：批运行模式 ON ：进位模式	• 如果跟踪内存在 END 处正被使用，当此继电器变为 OFF，跟踪的启动被延迟，直到可执行。 • 如果跟踪内存在 END 处正被使用，当此继电器是变为 ON 时，启动的跟踪程序执行，直到下一次 END。	U	新增	Q4AR
	待机系统到控制系统切换状态标志	ON ┐ 1个扫描周期 OFF ◄┘►	• 只要从待机系统切换到控制系统（只 ON1 个扫描周期）发生，变为 ON。 • 此状态标志只能用于扫描周期执行类型程序。	S（每次 END）	新增	QnPRH
SM1519	先前控制系统识别标志	ON ┐ 1个扫描周期 OFF ◄┘►	• 上次运行的控制系统是系统 B，如果电源上电，或者两个系统一起复位，则在 RUN 之后，只有系统 A 侧 ON1 个扫描周期。	S（每次 END）	新增	
SM1520	数据跟踪传送触发规格	OFF ：无触发器 ON ：由触发器	SM1520 块 1 SM1521 块 2 SM1522 块 3 SM1523 块 4 SM1524 块 5 SM1525 块 6 SM1526 块 7 SM1527 块 8 SM1528 块 9 SM1529 块 10 SM1530 块 11 SM1531 块 12 SM1532 块 13 SM1533 块 14 SM1534 块 15 SM1535 块 16 SM1536 块 17 SM1537 块 18 SM1538 块 19 SM1539 块 20 SM1540 块 21 SM1541 块 22 SM1542 块 23 SM1543 块 24 SM1544 块 25 SM1545 块 26 SM1520 块 27 SM1521 块 28 SM1522 块 29 〈在 Q4AR 中〉 • 当用数据跟踪指令 S.TRUCK 发送数据时，目标块被指定为触发器。 〈在 QnPRH 中〉 • 当数据发送是基于冗余参数对话框的跟踪设置时，目标块被指定为触发器。 • 当在跟踪设置中允许“Auto Tracking block No.1”时，SM1520 由系统在电源接通 /STOP 到 RUN 时将其变为 ON。在其他情况中，SM1520 到 SM1583 由用户变为 ON。	〈在 Q4AR 中〉 U 〈在 QnPRH 中〉 S（初始化）/U	新增	Q4AR QnPRH
SM1521						
SM1522						
SM1523						
SM1524						
SM1525						
SM1526						
SM1527						
SM1528						
SM1529						
SM1530						
SM1531						
SM1532						
SM1533						
SM1534						
SM1535						
SM1536						
SM1537						
SM1538						
SM1539						
SM1540						
SM1541						
SM1542						
SM1543						
SM1544						
SM1545						
SM1546						
SM1547						
SM1548						

附表 .13 特殊继电器列表

号码	名称	含义	解释			设置方式（当被设定时）	相应的 ACPU M9 □□□	相应的 CPU
SM1549	数据跟踪传送链接规格	OFF ：无触发器 ON ：有触发器	SM1549	块 30	〈在 Q4AR 中〉 当数据是数据跟踪指令 S.TRUCK 发送时，目标块被指定为触发器。 〈在 QnPRH 中〉 • 当数据是基于冗余参数对话框的跟踪设置发送时，目标块被指定为触发器。 • 当在跟踪设置中允许“AutoTrackingblock No.1”时，SM1520 由系统在电源接通 /STOP 到 RUN 时将其变为 ON。在其他情况中，SM1520 到 SM1583 由用户变为 ON。	〈在 Q4AR 中〉 U 〈在 QnPRH 中〉 S（初始化）/U	新增	Q4AR QnPRH
SM1550			SM1550	块 31				
SM1551			SM1551	块 32				
SM1552			SM1552	块 33				
SM1553			SM1553	块 34				
SM1554			SM1554	块 35				
SM1555			SM1555	块 36				
SM1556			SM1556	块 37				
SM1557			SM1557	块 38				
SM1558			SM1558	块 39				
SM1559			SM1559	块 40				
SM1560			SM1560	块 41				
SM1561			SM1561	块 42				
SM1562			SM1562	块 43				
SM1563			SM1563	块 44				
SM1564			SM1564	块 45				
SM1565			SM1565	块 46				
SM1566			SM1566	块 47				
SM1567			SM1567	块 48				
SM1568			SM1568	块 49				
SM1569			SM1569	块 50				
SM1570			SM1570	块 51				
SM1571			SM1571	块 52				
SM1572			SM1572	块 53				
SM1573			SM1573	块 54				
SM1574			SM1574	块 55				
SM1575			SM1575	块 56				
SM1576			SM1576	块 57				
SM1577			SM1577	块 58				
SM1578			SM1578	块 59				
SM1579			SM1579	块 60				
SM1580			SM1580	块 61				
SM1581			SM1581	块 62				
SM1582			SM1582	块 63				
SM1583			SM1583	块 64				
SM1590	来自网络模块的切换状态	OFF ：正常 ON ：切换不成功	• 如果网络模块检测到网络故障并发送切换请求到本站 CPU 模块，当切换不能正常执行，变为 ON。			S（出错发生）	新增	Q4AR
	来自网络模块的系统切换允许 / 禁止标志	OFF ：系统切换请求发送模块不存在 ON ：存在有系统切换请求发送模块	• 当从网络模块上发送系统切换请求时，变为 ON。发送系统切换的模块号可以通过 SD1590 确认。 • SD1590 的各个位全部 OFF 时该继电器将 OFF。			S（每次 END）		QnPRH
SM1591	在系统切换时，待机系统出错检测禁止标志	ON ：在系统切换时新待机系统没有检测到出错 ON ：在系统切换时新待机系统检测到出错	当 SD1590 的所有位都为 OFF 时，变为 OFF。如果在系统切换过程中新待机站检测到 6210：STANDBY，此标志用于确认。 此可以应用到下列切换方法： • 从 GX Developer 上进行系统切换 • 使用专用指令进行系统切换 • 由智能功能模块进行系统切换			U	新增	
SM1592	允许 / 禁止用户系统切换	OFF ：禁止用户系统切换 ON ：允许用户系统切换	• 此标志允许用户从 GX Developer 或者通过专用指令 (SP.CONTSW) 执行系统切换。			U	新增	
SM1593	待机系统 CPU 的扩展基板的访问设置	OFF ：出错 ON ：无处理	独立模式中，待机系统 CPU 访问扩展基板上安装的智能功能模块缓冲存储区时的操作设置。 OFF ：待机系统 CPU 访问扩展基板上安装的智能功能模块缓冲存储区时，返回“OPERATION ERROR”（出错代码：4112）。 ON ：待机系统 CPU 访问扩展基板上安装的智能功能模块缓冲存储区时，无处理。			U	新增	QnPRH*2
SM1595	复制内存到其他系统启动标志	OFF ：启动内存复制 ON ：无内存复制被启动	• 当 SM1595 从 OFF 变为 ON，从控制系统到待机系统的内存复制启动。注意，当 SM1595 从 OFF 变为 ON，内存复制并不启动，如果复制目标的 I/O 号（待机系统 CPU 模块：3D1H）没有存储到 SD1595。					QnPRH
SM1596	复制内存到其他系统状态标志	OFF ：内存复制未执行 ON ：内存复制执行	• 当内存被复制到其他系统，变为 ON。 • 当内存复制执行已经结束时，变为 OFF。			S（复制启动 / 完成）		

附表 .13 特殊继电器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU M9 □□□	相应的 CPU
SM1597	内存复制到其他系统结束标志	OFF ：内存复制未结束 ON ：内存结束	• 到其他系统的内存复制一结束就变为 ON。	S（结束）/U	新增	QnPRH
SM1598	内存复制过程中标准 ROM 的复制内容	OFF ：复制标准 ROM 数据 ON ：标准 ROM 数据未被复制	• 如果由用户设定为 ON，当执行内存复制时，标准 ROM 的数据并不被复制到其他系统。	U		

*2: 以序列号的高 5 位为 “09012”以后的 CPU 为对象。

(13)用于冗余系统（其他系统 CPU 信息 *1)

SM1600 到 SM1650 只对 CPU 冗余系统备用模式有效，因此不能在独立模式中刷新它们。备用模式或者独立模式对。SM4651 到 SM1699 都有效。对单独系统，SM1600 到 SM1699 都为关断。

附表 .14 特殊继电器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的本站 SM □□ *2	相应的 CPU
SM1600	检测出错	OFF ：无出错 ON ：有出错	• 如果在检测结果中发生出错，则接通。（包括外部检测） • 保持为接通，即使此后恢复为正常。	S（每次 END）	SM0	Q4AR
	其他系统出错标志	OFF ：无出错 ON ：有出错	• 当在冗余系统中发生出错时，接通。出错检查（接通 SD1600 的单个位。） • 当无出错存在时，关断。	S（每次 END）	-	QnPRH
SM1601	自检测出错	OFF ：无自检测出错 ON ：有自检测出错	• 当在自检测结果中发生出错时，接通。 • 即使此后恢复为正常，保持为接通。	S（每次 END）	SM1	Q4AR
SM1605	出错公共信息	OFF ：无出错公共信息 ON ：有出错公共信息	• 当有出错公共信息时，并且 SM1600 接通，接通。	S（每次 END）	SM5	
SM1610	其它系统的检测出错	OFF ：无出错 ON ：有出错	• 当发生检测出错时，接通。（包括报警器为 ON 时的出错检测，和 CHK 指令检测到的出错。） • 对应其他系统的 SM0 的状态。	S（每次 END）	SM0	QnPRH
SM1611	其它系统的自检测出错。	OFF ：无自检测出错发生 ON ：有自检测出错发生	• 当发生自检测出错时，接通。（不包括报警器为 ON 时的出错检测，和 CHK 指令检测到的出错。） • 对应其他系统的 SM1 的状态。	S（每次 END）	SM1	
SM1615	其它系统的公共出错信息	OFF ：无公共出错信息存在 ON ：有公共出错信息存在	• 当其他系统中有公共出错信息时，接通。 • 对应其他系统的 SM5 的状态。	S（每次 END）	SM5	
SM1616	出错个别信息	OFF ：无出错个别信息 ON ：有出错个别信息	• 当有出错个别信息时，并且 SM1600 接通，接通。	S（每次 END）	SM16	Q4AR
SM1626	其它系统的出错个别信息	OFF ：无个别出错信息存在 ON ：有个别出错信息存在	• 其他系统中有个别出错信息时，接通。 • 对应其他系统的 SM16 的状态 。	S（每次 END）	SM16	QnPRH
SM1649	待机系统取消出错标志	OFF 到 ON: 待机系统的取消出错	通过将此继电器从 OFF 变为 ON，可以将待机系统 CPU 模块中发生的连续出错取消掉。 使用 SD1649 指定要被取消的出错的出错代码。	U	-	
SM1653	STOP 触点	STOP 状态	• 当处于 STOP 状态时，接通。	S（每次 END）	SM203	Q4AR
SM1654	PAUSE 触点	PAUSE 状态	• 当处于 PAUSE 状态时，接通。	S（每次 END）	SM204	
SM1655	STEP-RUN 触点	STEP-RUN 状态	• 当处于 STEP-RUN 状态时，接通。	S（每次 END）	SM205	

*1 存储其他系统 CPU 检测信息和系统信息。
*2 这给出了用于本站系统 CPU 的特殊继电器 (SM□□)。

(14)用于冗余系统（跟踪）

备用模式和独立模式对 SM1700 到 SM1799 都有效。

对单独系统都为 OFF。

附表 .15 特殊继电器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU M9 □□□	相应的 CPU
SM1700	跟踪执行标志	OFF ：不可以执行 ON ：可以执行	• 当跟踪可以正常执行时，变为 ON。	S（状态改变）	新增	Q4AR
	传送结束标志	OFF ：传送未结束 ON ：传送结束	• 一旦块 1 到块 64 的传送结束，变为 ON 一个扫描周期。	S（状态改变）	新增	QnPRH
SM1709	在运行中程序更改冗余跟踪过程中手动切换禁止 / 允许设置	ON ：手动切换（禁止被删除） OFF ：手动切换禁止	(1) 将此继电器从 OFF 变为 N，以允许在运行中程序更改冗余跟踪过程中进行手动切换。 在手动切换禁止状态被删除之后，系统自动关闭 SM1709。 (2) 在运行中程序更换冗余跟踪过程中，发生了下列情况中的一种，则执行系统切换，而不管此继电器的状态如何。 •电源关断、复位、硬件故障、CPU 停止出错。 (3) 在下列状态种，系统切换禁止状态也可以被此继电器删除。 •多个块运行中程序更改冗余跟踪执行状态 •文件批量运行中程序更改冗余跟踪执行状态	S（当执行时）/U		
SM1710	在运行中程序更改过程中发送跟踪数据允许标志	OFF ：无软元件跟踪 ON ：发送软元件存储区	(1) 设定在运行中程序更改冗余跟踪过程中，是否执行下列数据的跟踪。 •软元件存储区 （包括 SM/SD，它们将自动执行跟踪） •PIDINIT 信息、S.PIDINIT 信息、SFC 信息 (2) SM1710 也可以用于设定当多个程序块的运行中更改或者批量的文件正在被执行以确保两个系统的一致性时，是否执行跟踪。 (3) 此 SM 也被跟踪数据从控制系统 CPU 模块发送到待机系统 CPU 模块。	U		

附表 .15 特殊继电器列表

号码	名称	含义	解释			设置方式（当被设定时）	相应的 ACPU M9 □□□	相应的 CPU
SM1712	传送触发器结束标志	OFF ：传送未结束 ON ：传送结束	SM1712	块 1	〈在 Q4AR 中〉 只在相应的数据传送结束时的一个扫描周期中变为 ON。 〈在 QnPRH 中〉 只在相应的块传送结束时的一个扫描周期中变为 ON。	S（状态改变）	新增	Q4AR QnPRH
SM1713			SM1713	块 2				
SM1714			SM1714	块 3				
SM1715			SM1715	块 4				
SM1716			SM1716	块 5				
SM1717			SM1717	块 6				
SM1718			SM1718	块 7				
SM1719			SM1719	块 8				
SM1720			SM1720	块 9				
SM1721			SM1721	块 10				
SM1722			SM1722	块 11				
SM1723			SM1723	块 12				
SM1724			SM1724	块 13				
SM1725			SM1725	块 14				
SM1726			SM1726	块 15				
SM1727			SM1727	块 16				
SM1728			SM1728	块 17				
SM1729			SM1729	块 18				
SM1730			SM1730	块 19				
SM1731			SM1731	块 20				
SM1732			SM1732	块 21				
SM1733			SM1733	块 22				
SM1734			SM1734	块 23				
SM1735			SM1735	块 24				
SM1736			SM1736	块 25				
SM1737			SM1737	块 26				
SM1738			SM1738	块 27				
SM1739			SM1739	块 28				
SM1740			SM1740	块 29				
SM1741			SM1741	块 30				
SM1742			SM1742	块 31				
SM1743			SM1743	块 32				
SM1744			SM1744	块 33				
SM1745			SM1745	块 34				
SM1746			SM1746	块 35				

附表 .15 特殊继电器列表

号码	名称	含义	解释			设置方式（当被设定时）	相应的 ACPU M9 □□□	相应的 CPU
SM1747	传送触发器结束标志	OFF ：传送未结束 ON ：传送结束	SM1747	块 36	<在 Q4AR 中> 只在相应的数据传送结束时的一个扫描周期中变为 ON。 <在 QnPRH 中> 只在相应的块传送结束时的一个扫描周期中变为 ON。	S（状态改变）	新增	Q4AR QnPRH
SM1748			SM1748	块 37				
SM1749			SM1749	块 38				
SM1750			SM1750	块 39				
SM1751			SM1751	块 40				
SM1752			SM1752	块 41				
SM1753			SM1753	块 42				
SM1754			SM1754	块 43				
SM1755			SM1755	块 44				
SM1756			SM1756	块 45				
SM1757			SM1757	块 46				
SM1758			SM1758	块 47				
SM1759			SM1759	块 48				
SM1760			SM1760	块 49				
SM1761			SM1761	块 50				
SM1762			SM1762	块 51				
SM1763			SM1763	块 52				
SM1764			SM1764	块 53				
SM1765			SM1765	块 54				
SM1766			SM1766	块 55				
SM1767			SM1767	块 56				
SM1768			SM1768	块 57				
SM1769			SM1769	块 58				
SM1770			SM1770	块 59				
SM1771			SM1771	块 60				
SM1772			SM1772	块 61				
SM1773			SM1773	块 62				
SM1774			SM1774	块 63				
SM1775			SM1775	块 64				

(15)余电源模块信息

附表 .16 特殊继电器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU M9 □□□	相应的 CPU
SM1780	电源 OFF 检测标志	OFF ：无输入电源为 OFF 状态的冗余电源模块 ON ：有输入电源为 OFF 状态的冗余电源模块	• 当一个或以上输入电源为 OFF 状态的冗余电源模块被检测到时，变为 ON。 • 如果 SD1780 的某个位为 ON 时，变为 ON。 • 如果 SD1780 的所有位都为 OFF 时，本继电器也 OFF。 • 当主基板不是冗余主基板（Q38RB）时，本继电器将变为 OFF。 0：执行与外围设备的通信 1：不执行与外围设备的通信	S（每次 END）	新增	Qn(H)*2 QnPH*2 QnPRH Rem
SM1781	电源模块故障检测标志	OFF ：没有检测到有故障的冗余电源模块 ON ：检测到有故障的冗余电源模块	• 当检测到一个或者更多有故障的冗余电源模块，变为 ON。 • 如果 SD1781 的某个位为 ON，本继电器将 ON。 • 如果 SD1781 的所有位都为 OFF，本继电器也将 OFF。 • 当主基板不是冗余主基板（Q38RB），本继电器变为 OFF。 0：执行与外围设备的通信 1：不执行与外围设备的通信	S（每次 END）		
SM1782	电源模块 1*1 用瞬间掉电检测标志	OFF ：未检测出瞬间掉电故障 ON ：检测出瞬间掉电	• 当到电源 1 或者 2 的输入电源的瞬间掉电被检测到一次或多次，变为 ON。在变为 ON 之后，如果电源从瞬间掉电中恢复，也将保持为 ON 状态。 • 当 CPU 模块启动时，将电源 1、电源 2 的标志（SM1782、SM1783）变为 OFF。 • 在其中一个冗余电源模块的输入电源为 OFF 时，输入电源为 OFF 的冗余电源模块相对应的标志将变为 OFF。 • 当主基板不是冗余主基板（Q38RB）时，本继电器变为 OFF。 0：执行与外围设备的通信 1：不执行与外围设备的通信	S（每次 END）		
SM1783	电源模块 2*1 用瞬间掉电检测标志					

*1：“电源 1”表示安装在冗余基板（Q38RB/Q68RB/Q65WRB）的 POWER1 插槽上的冗余电源模块（Q64RP）。
“电源 2”表示安装在冗余基板（Q38RB/Q68RB/Q65WRB）的 POWER2 插槽上的冗余电源模块（Q64RP）。

*2：以序列号的高 5 位为“04012”以后的 CPU 为对象。
多 CPU 系统配置中，以序列号的高 5 位为“07032”以后的 CPU 为对象。

附录 2 特殊寄存器列表

特殊寄存器，SD，是可编程控制器内用于固定应用的内部寄存器。
由于这个原因，在顺控程序不能以和使用普通内部寄存器相同的方式使用这些特殊寄存器。
但是，它们可以按照需要被写入数据以控制 CPU 模块和远程 I/O 模块。
如果没有相反的特殊指定，存储在特殊寄存器中的数据是以 BIN 值存储的。

后面附表 .17 中的标题的含义如下所示。

附表 .17 对于下列条目的详细信息，请参考下面的手册

条目	条目的功能
号码	• 表示特殊寄存器号
名称	• 表示特殊寄存器的名称
含义	• 表示特殊寄存器的内容
解释	• 更详细地讨论特殊寄存器的内容
设置方式 （当被设定时）	• 如果是由系统执行设置时，表示继电器是由系统设定还是由用户设定； 〈设置方式〉 S : 由系统设定 U : 用户设定（顺控程序或者来自 GX Developer 的测试操作） S/U : 系统和用户都可以设定 〈当被设定时〉 只用于指示由系统设定寄存器 每次 END : 在每个 END 处理过程中进行设定 初始化 : 只在初始化处理（当电源变为 ON，或者从 STOP 变为 RUN 时）过程中进行设定 状态改变 : 只在有状态变化时才进行设定 出错 : 在出错发生时进行设定 指令执行 : 当指令被执行时设定 请求 : 只在有用户请求（通过 SM，等软元件。）时才进行设定 系统切换 : 在执行系统切换时进行设定。
相应的 ACPU M9□□□	• 表示 ACPU 中相应的特殊继电器 （当内容改变时，特殊继电器代表 D9□□□格式改变。不兼容 Q00J/Q00/Q01 和 QnPRH。） • 新增表示特殊继电器是新近增加到 QnACPU 或者 Q 系列 CPU 模块中。
相应的 CPU	表示相应的 CPU 模块型号。 ○ : 表示所有的 QnACPU 和 QCPU。 QCPU : 表示所有的 Q 系列 CPU 模块。 Q00J/Q00/Q01 : 表示基本型 QCPU。 Qn(H) : 表示高性能型 QCPU。 QnPH : 表示过程控制 CPU。 QnPRH : 表示冗余 CPU。 QnU : 表示通用型 QCPU。 QnA : 表示 QnA 系列和 Q2ASCPU 系列。 Rem : 表示 MELSECNET/H 远程 I/O 模块。 每个 CPU 模块的型号 : 表示相关的特殊 CPU 模块。（例如：Q4AR、Q3A）

对于下列条目的详细信息，请参考下面的手册：
• 网络 ☞相应网络模块的手册
• SFC ☞QCPU(Q 模式)/QnACPU 编程手册 (SFC)

要 点

SD1200 ～ 1255 用于 QnACPU。
在 QCPU 中将成为 “空闲”。

(1) 检测信息

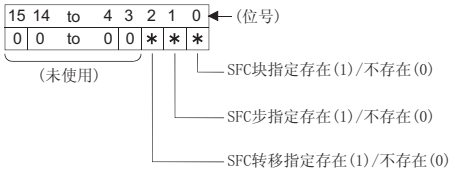
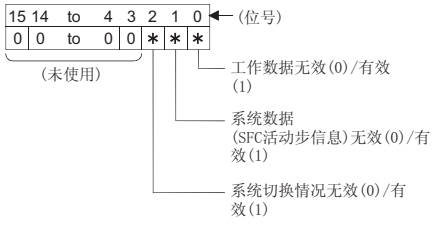
附表 .18 特殊寄存器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9 □□□	相应的 CPU
SD0	检测出错	检测出错代码	• 检测发现的出错的出错代码，存储为 BIN 数据。 • 内容和最新的故障历史信息相同。	S（出错）	D9008 格式改变	○ Rem
SD1	用于检测出错发生的时间	用于检测出错发生的时间	• SD0 数据被更新时的年（后两位数）和月被以 2 位 BCD 代码存储。 b15 到 b8 b7 到 b0 年(0到99) | 月(1到12) (实例)十月，1995 H9510	S（出错）	新增	
SD2			• SD0 被更新时的日期和小时以 2 位 BCD 代码存储。 b15 到 b8 b7 到 b0 日期(1到31) | 小时(0到23) (实例)25日，上午10点 H2510			
SD3			• SD0 数据被更新时的分钟和秒以 2 位 BCD 代码存储。 b15 到 b8 b7 到 b0 分钟(0到59) | 秒(0到59) (实例)35分48秒 H3548			
SD4	出错信息类别	出错信息类别代码	用于表示正被存储到公共信息区域（SD5 到 SD15）和个别信息区域（SD16 到 SD26）中的是存储哪种类型的信息的类别代码被存储在这里。 类别代码用于判断被存储的出错信息类型。 b15 到 b8 b7 到 b0 单个信息类别代码 | 公共信息类别代码 • 公共信息类别代码存储下列代码： 0：没有出错 1：基板 / 模块号 / 可编程控制器号 / 基板号* 2：文件名称 / 驱动器名称 3：时间（设定值） 4：程序出错位置 5：系统切换原因（只用于 Q4ARCPU 和冗余 CPU） 6：跟踪容量超限出错的原因（冗余 CPU 特有） 7：电源号（冗余 CPU 特有） 8：跟踪传送数据分类（冗余 CPU 特有） *：对于包含基本型 QCPU、高性能型 QCPU、过程控制 CPU 和通用型 QCPU 的多 CPU 系统，模块号或者 CPU 号是根据发生出错存储。（参考相应号码被存储的出错代码。） 1 号可编程控制器：1，2 号可编程控制器：2，3 号可编程控制器：3，4 号可编程控制器：4 • 个别信息类别代码存储下列代码： 0 ：没有出错 1 ：（空置） 2 ：文件名称 / 驱动器名称 3 ：时间（实际测量值） 4 ：程序出错位置 5 ：参数号 6 ：报警器号 7 ：CHK 指令故障号（这不能用于基本型 QCPU 和通用型 QCPU） 8 ：系统切换故障的原因（冗余 CPU 特有） 12：文件诊断信息（通用型 QCPU 特有） 13：参数号 /CPU 号（通用型 QCPU 特有）	S（出错）	新增	

附表 .18 特殊寄存器列表

<table>
<tr><th>号码</th><th>名称</th><th>含义</th><th>解释</th><th>设置方式
（当被设定时）</th><th>相应的
ACPU
D9 □□□</th><th>相应的 CPU</th></tr>
<tr><td>SD5
SD6
SD7
SD8
SD9
SD10
SD11
SD12
SD13
SD14
SD15</td><td>出错公共信息</td><td>出错公共信息</td><td>• 对应出错代码 (SD0) 的公共信息存储在这里。
• 下列十种类型的信息被存储在这里：
• 出错公共信息 类型可以通过 SD4 中的 “公共信息类别代码”来判断。
（存储在 SD4 中的 “公共信息 类别代码”的值对应下面的 1) 到 8)、12)、13) 。）
1) 槽号
<table><tr><th>号码</th><th>含义</th></tr><tr><td>SD5</td><td>槽号/PLC 号/基板 *1、*2、*3、*4</td></tr><tr><td>SD6</td><td>I/O号 *5</td></tr><tr><td>SD7
SD8
SD9
SD10
SD11
SD12
SD13
SD14
SD15</td><td>(空置)</td></tr></table>*1: 对于包含基本型 QCPU、高性能型 QCPU、过程控制 CPU 和通用型 QCPU 的多 CPU 系统，依据发生的出错存储槽号或者 CPU 号。
在多 CPU 系统中的插槽 0，是最右端的 CPU 模块右边的插槽。
（参考已存储的号码对应的出错代码。）
1 号 CPU：1，2 号 CPU：2，3 号 CPU：3，4 号 CPU：4
*2: 如果在 MELSECNET/H 远程 I/O 站中安装的模块上发生保险丝熔断或者 I/O 验证出错，网络号被存储到高 8 位，站号被存储到低 8 位。
使用 I/O 号去检查发生保险丝 熔断 或者 I/O 验证出错的模块。
*3: 当为可以安装模块的最后一个插槽以后的插槽中的模块执行指令时，255 被存储到基本型 QCPU 上的 SD5 中。
*4: 基板号和槽号的定义
〈基板号〉
此值用于确认安装了 CPU 模块的基板。 下列给出了基板号的的定义。
<table><tr><th>基板号</th><th>定义</th></tr><tr><td>0</td><td>表示安装了CPU模块的主基板。</td></tr><tr><td>1 到 7</td><td>表示扩展基板。通过扩展基板上的级号设置连接器进行的级号设置是基板号。
当级号设置是扩展1: 基板号 = 1
当级号设置是扩展7: 基板号 = 7</td></tr></table>冗余 CPU 的基板号一直是 “0”，因为它不能连接到扩展基板。
〈槽号〉
此值用于确认每个基板的插槽和安装在此插槽上的模块。
•主基板的 I/O 槽 0(CPU 插槽右侧的插槽）被定义为 “槽号 = 0”的插槽。
•按照主基板和扩展基板 1 到 7 的顺序，槽号码被连续分配到基板的槽号上。
•当基板的插槽在可编程控制器 参数对话框的 I/O 分配设置中设定之后，槽号只分配到已设定的槽号上。
*5: 当 0FFFFH 被存储到 SD6 (I/O 号）中时，由于可编程控制器参数对话框的 I/O 分配设置中有重叠的 I/O 号，所以 I/O 号不能被确认。 因此，使用 SD5 确认出错位置。
2) 文件名称 / 驱动器名称
<table><tr><th>号码</th><th colspan="2">含义</th></tr><tr><td>SD5</td><td colspan="2">驱动器</td></tr><tr><td>SD6
SD7
SD8
SD9</td><td colspan="2">文件名称
(ASCII代码：8字符</td></tr><tr><td>SD10</td><td>扩展 *6</td><td>2EH(.)</td></tr><tr><td>SD11</td><td colspan="2">(ASCII代码：3字符)</td></tr><tr><td>SD12
SD13
SD14
SD15</td><td colspan="2">(空置)</td></tr></table>(实例)文件名称
ABCDEFGH. IJK
b15到 b8 b7到 b0
<table><tr><td>42H(B)</td><td>41H(A)</td></tr><tr><td>44H(D)</td><td>43H(C)</td></tr><tr><td>46H(F)</td><td>45H(E)</td></tr><tr><td>48H(H)</td><td>47H(G)</td></tr><tr><td>49H(I)</td><td>2EH(.)</td></tr><tr><td>4BH(K)</td><td>4AH(J)</td></tr></table></td><td>S(出错）</td><td>新增</td><td>○
Rem</td></tr>
</table>

附表 .18 特殊寄存器列表

<table>
<tr><th>号码</th><th>名称</th><th>含义</th><th>解释</th><th>设置方式
（当被设定时）</th><th>相应的
ACPU
D9 □□□</th><th>相应的 CPU</th></tr>
<tr><td>SD5
SD6
SD7
SD8
SD9
SD10</td><td rowspan="2">出错公共信息</td><td rowspan="2">出错公共信息</td><td>3) 时间（设定值）
<table><tr><th>号码</th><th>含义</th></tr><tr><td>SD5</td><td>时间：1 μs为单位(0到999 μs)</td></tr><tr><td>SD6</td><td>时间：1ms为单位(0到65535ms)</td></tr><tr><td>SD7
SD8
SD9
SD10
SD11
SD12
SD13
SD14
SD15</td><td>(空置)</td></tr></table>4) 程序出错位置
<table><tr><th>号码</th><th>含义</th></tr><tr><td>SD5
SD6
SD7
SD8</td><td>文件名称
(ASCII代码：8字符)</td></tr><tr><td>SD9</td><td>扩展 *6 | 2EH(.)</td></tr><tr><td>SD10</td><td>(ASCII代码：3字符)</td></tr><tr><td>SD11</td><td>模式 *7</td></tr><tr><td>SD12</td><td>块号</td></tr><tr><td>SD13</td><td>步号/转移号</td></tr><tr><td>SD14</td><td>顺控步号(L)</td></tr><tr><td>SD15</td><td>顺控步号(H)</td></tr></table>*7 ： 模式数据的内容
 
</td><td>S(出错)</td><td>新增</td><td>○
Rem</td></tr>
<tr><td>SD11
SD12
SD13
SD14
SD15</td><td>5) 系统切换的原因
<table><tr><th>号码</th><th>含义</th></tr><tr><td>SD5</td><td>系统切换情况
(0：自动切换/1：手动切换)</td></tr><tr><td>SD6</td><td>系统切换方向
(0：自动切换/1：手动切换
1：控制系统到待机系统)</td></tr><tr><td>SD7</td><td>跟踪标志 *8</td></tr><tr><td>SD8
SD9
SD10
SD11
SD12
SD13
SD14
SD15</td><td>(空置)</td></tr></table>*8 ：跟踪标志内容
表示跟踪数据是否有效。
 
</td><td>S(出错)</td><td>新增</td><td>Q4AR</td></tr>
</table>

附表 .18 特殊寄存器列表

号码	名称	含义	解释	设置方式（当被设定时	相应的 ACPU D9 □□□	相应的 CPU
SD5 SD6 SD7 SD8 SD9 SD10 SD11 SD12	出错公共信息	出错公共信息	5) 系统切换的原因 SD5：系统切换情况 *13 SD6：控制系统切换指令参数 SD7～SD15：（空置） *13：系统切换原因的细节 0 ：无系统切换情况(默认) 1 ：电源OFF、复位、硬件故障、看门狗出错 2 ：停止出错(看门狗出错除外) 3 ：网络模块的系统切换请求 16 ：控制系统切换指令 17 ：来自GX Developer的控制系统切换请求 6) 跟踪容量超限出错的原因 当可以被跟踪的数据数 (100k) 被超过时，用相应特殊继电器的位模式来指示块号。 SD5：b15=1(SM1535)(块16)，b8=1(SM1528)(块9)，b0=1(SM1520)(块1) SD6：b0=0 SD7：b0=0 SD8：b15=1(SM1583)(块64)，b0=1(SM1568)(块49) SD9：b0=0 SD15：b0=0	S(出错)	新增	QnPRH
SD13 SD14 SD15			7) 电源号 SD5：基板号 SD6：电源号 SD7～SD15：（空置） 1：电源1异常 2：电源2异常 “电源模块1”：安装在冗余基板(Q38RB、68RB)的POWER1插槽上的冗余电源模块 “电源模块2”：安装在冗余基板(Q38RB、68RB)的POWER2插槽上的冗余电源模块	S(出错)	新增	Qn(H)*1 QnPH*1 QnPRH

*1: 以序列号的高 5 位为 “07032” 以后的 CPU 为对象。

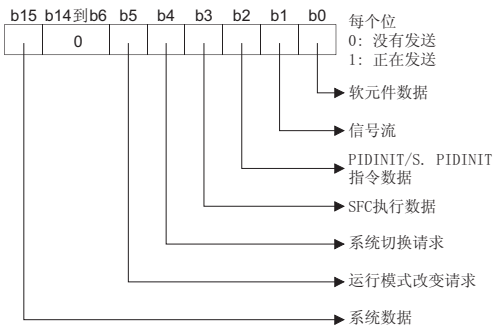
附表 . 18 特殊寄存器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9 □□□	相应的 CPU
SD5						
SD6						
SD7						
SD8						
SD9						
SD10	出错公共信息	出错公共信息		S（出错）	新增	QnPRH
SD11						
SD12						
SD13						
SD14						
SD15						

8) 跟踪传送数据块
在跟踪过程中存储数据块。

号码	含义
SD5	数据类型 *15
SD6	（空置）
SD7	
SD8	
SD9	
SD10	
SD11	
SD12	
SD13	
SD14	
SD15	

*15：数据分类的详细信息



附表 . 18 特殊寄存器列表

<table>
<tr><th>号码</th><th>名称</th><th>含义</th><th>解释</th><th>设置方式
（当被设定时）</th><th>相应的
ACPU
D9 □□□</th><th>相应的 CPU</th></tr>
<tr><td>SD16</td><td rowspan="11">出错公共信息</td><td rowspan="11">出错公共信息</td><td rowspan="11">• 对应出错代码 (SD0) 的单个信息存储在这里。
• 存储了下列八种不同类型的信息。
• 出错个别信息类型可以通过 SD4 中的 “个别信息类别代码” 来判断。
（存储在 SD4 中的 “个别信息类别代码” 的值对应下列的 1) 到 8)。）
1) （空置）
2) 文件名 / 驱动器名
(实例)文件名 = ABCDEFGH. IJK
b15到 b8 b7 到 b0
42H(B) 41H(A)
44H(D) 43H(C)
46H(F) 45H(E)
48H(H) 47H(G)
49H(I) 2EH(.)
4BH(K) 4AH(J)
号码 含义
SD16 驱动器
SD17 SD18 SD19 SD20 文件名 (ASCII码：8个字符)
SD21 扩展 *6 2EH(.)
SD22 (ASCII码：3个字符)
SD23 SD24 SD25 SD26 (空置)
3) 时间（实际测量值）
号码 含义
SD16 时间：1μs为单位(0到999μs)
SD17 时间：1ms为单位(0到65535ms)
SD18 SD19 SD20 SD21 SD22 SD23 SD24 SD25 SD26 (空置)
4) 程序出错位置
号码 含义
SD16 SD17 SD18 SD19 文件名 (ASCII码：8个字符)
SD20 扩展 *6 2EH(.)
SD21 (ASCII码：3个字符)
SD22 模式 *7
SD23 块号
SD24 步号/转换号
SD25 顺控步号(L)
SD26 顺控步号(H)
*7 ： 模式数据的内容
15 14 到 4 3 2 1 0 ← (位号)
0 0 到 0 0 * * *
(未使用)
SFC块指定 存在(1)/不存在(0)
SFC步指定 存在(1)/不存在(0)
SFC转换指定 存在(1)/不存在(0)
5) 参数数目 6) 报警器数/ 7) CHK指令故障号
号码 含义
SD16 参数数目 *16
SD17 SD18 SD19 SD20 SD21 SD22 SD23 SD24 SD25 SD26 (空置)
号码 含义
SD16 数目
SD17 SD18 SD19 SD20 SD21 SD22 SD23 SD24 SD25 SD26 (空置)
*16: 对于参数数目的详细信息，参考 QCPU 用户手册（功能解说 / 程序基础篇）或者使用的 QnACPU 的用户手册。</td><td rowspan="11">S（出错）</td><td rowspan="11">新增</td><td rowspan="11">○
Rem</td></tr>
<tr><td>SD17</td></tr>
<tr><td>SD18</td></tr>
<tr><td>SD19</td></tr>
<tr><td>SD20</td></tr>
<tr><td>SD21</td></tr>
<tr><td>SD22</td></tr>
<tr><td>SD23</td></tr>
<tr><td>SD24</td></tr>
<tr><td>SD25</td></tr>
<tr><td>SD26</td></tr>
</table>

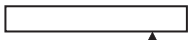
备 注

*6 ： 扩展方面的附表 .19。

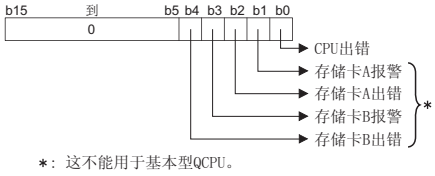
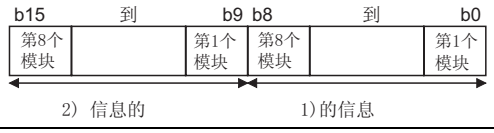
附表 .19 扩展名

SDn	SDn+1		扩展名称	文件类型
高 8 位	低 8 位	高 8 位		
51H	50H	41H	QPA	参数
51H	50H	47H	QPG	• 顺控程序 • SFC 程序
51H	43H	44H	QCD	软元件注释
51H	44H	49H	QDI	软元件初始化值
51H	44H	52H	QDR	文件寄存器
51H	44H	53H	QDS	仿真数据 (用于 QnA)
51H	44H	4CH	QDL	局部软元件 (基本型 QCPU 以外的其他 CPU)
51H	54H	44H	QTD	采样跟踪数据 (基本型 QCPU 以外的其他 CPU)
51H	54H	4CH	QTL	状态锁存数据 (用于 QnA)
51H	54H	50H	QTP	程序跟踪数据 (用于 QnA)
51H	54H	52H	QTR	SFC 跟踪文件 (用于 QnA)
51H	46H	44H	QFD	故障历史数据 (基本型 QCPU 和通用型 QCPU 以外的其他 CPU)
51H	46H	44H	QST	SP.DEVST/S.DEVLD 指令文件 (用于通用型 QCPU)

附表 .18 特殊寄存器列表

<table>
<tr><th>号码</th><th>名称</th><th>含义</th><th>解释</th><th>设置方式
（当被设定时）</th><th>相应的
ACPU
D9 □□□</th><th>相应的 CPU</th></tr>
<tr><td rowspan="2">SD26</td><td rowspan="2">出错公共信息</td><td rowspan="2">出错公共信息</td><td>8) 系统切换故障的原因
<table><tr><th>号码</th><th>含义</th></tr><tr><td>SD16</td><td>系统切换禁止情况 *14</td></tr><tr><td>SD17
SD18
SD19
SD20
SD21
SD22
SD23
SD24
SD25
SD26</td><td>（空置）</td></tr></table>*14：系统切换故障详细的原因
 
0 ：正常切换结束(默认)
1 ：热备电缆故障(电缆移动，电缆故障，内部电路故障，硬件故障)
2 ：在待机系统中发生硬件故障，电源OFF，复位或者看门狗出错
3 ：在控制系统中发生硬件故障，电源OFF，复位或者看门狗出错
4 ：准备跟踪
5 ：时间超限
6 ：待机系统处于停止出错(看门狗出错除外)
7 ：两个系统的运行状态不同(只在备用模式中)
8 ：在过程中内存从控制系统到待机系统
9 ：程序在线更改
10 ：待机系统网络模块检测到出错
11 ：系统切换正在执行
12 ：运行中模块更换正在进行</td><td rowspan="2">S（出错）</td><td rowspan="2">新增</td><td>QnPRH</td></tr>
<tr><td>12) 文件诊断信息
<table><tr><td>SD16</td><td>故障信息(H)</td><td>驱动器号(L)</td></tr><tr><td>SD17
SD18
SD19
SD20</td><td colspan="2">文件名
(ASCII码：8个字符)</td></tr><tr><td>SD21
SD22</td><td>扩展 *6</td><td>2EH(.)</td></tr><tr><td></td><td colspan="2">(ASCII码：3个字符)</td></tr><tr><td>SD23
SD24</td><td colspan="2">故障信息2
(读取CRC值)</td></tr><tr><td>SD25
SD26</td><td colspan="2">故障信息3
(计算CRC值)</td></tr></table>13) 参数号 /CPU 号
<table><tr><th>号码</th><th>含义</th></tr><tr><td>SD16</td><td>参数号 *16</td></tr><tr><td>SD17</td><td>CPU号(1到4)</td></tr><tr><td>SD18
SD19
SD20
SD21
SD22
SD23
SD24
SD25
SD26</td><td>（空置）</td></tr></table></td><td>QnU</td></tr>
</table>

附表 .18 特殊寄存器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD50	出错复位	执行出错复位的出错号	• 存储执行出错复位的出错号	U	新增	○ Rem
SD51	电池电量不足锁存	位模式指示哪里发生电池电压下降	• 当电池电压下降时，所有的对应位变为 1(ON)。 • 随后，即使在电池电压恢复到正常之后，它们保持为 1(ON)。  *：这不能用于基本型QCPU。 • 在报警中，在为电池电量低指定的时间以内，数据可以被保持。 • 此出错表示电池放电的结束。 • 当使用高性能型 QCPU、过程控制 CPU、冗余 CPU 或者通用型 QCPU 时，存储卡 B 是标准的，因此相应位一直保持为 OFF。	S（出错）	新增	○
SD52	电池电量不足	位模式指示哪里发生电池电压下降	• 和上面 SD51 的配置相同 • 检到报警 (ON) 后，通过错误检测 (ON) 使报警 OFF。（仅用于通用型 QCPU） • 当电池电压恢复为正常，变为 0(OFF)。 • 当使用高性能型 QCPU、过程控制 CPU、冗余 CPU 或者通用型 QCPU 时，存储卡 B 是标准的，因此相应位一直保持为 OFF。	S（出错）	新增	
SD53	AC/DC DOWN 检测	AC/DC DOWN 检测次数	• 在 CPU 模块的运行过程中，每当输入电压下降到或者低于额定值的 85%(AC 电源）/65%(DC 电源）时，此值增加 1 并以 BIN 码存储。	S（出错）	D9005	○ Rem
SD54	MINI 链接出错	出错检测状态	1) 当安装的 MINI(-S3) 的 X(n+0)/X(n+20)，X(n+6)/X(n+26)，X(n+7)/X(n+27) 和 X(n+8)/X(n+28) 中的任何一个变为 ON 时，相应站的位变为 1(ON)。 2) 当安装的 MINI(-S3) 和 CPU 模块之间不能通讯时，变为 1(ON)。 	S（出错）	D9004 格式改变	QnA
SD60	保险丝熔断的模块号	保险丝熔断的模块号	• 存储在这里的值是最小的有保险丝熔断的模块的站 I/O 号。	S（出错）	D9000	○ Rem
SD61	I/O 模块验证出错号	I/O 模块验证出错号	• 发生 I/O 模块校验号出错的模块的最小 I/O 号。	S（出错）	D9002	

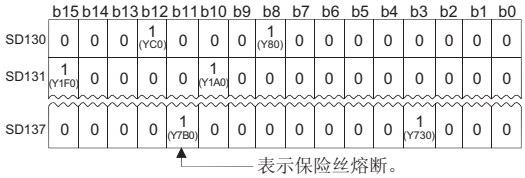
附表 . 18 特殊寄存器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD62	报警器号	报警器号	• 检测到的第一个报警器 号 (F 号) 被存储在这里。	S(指令执行)	D9009	
SD63	报警器数	报警器数	• 存储查找到的报警器数。	S(指令执行)	D9124	
SD64	检测到的报警器号码表	报警器检测号	当 F 由于 OUT F 或 SET F，而变为 ON 时，依次变为 ON 的 F 号被注册到 SD64 到 SD79 中。 由 RST F 变为 OFF 的 F 号，被从 SD64-SD79 中删除，存储在删除的 F 号之后的 F 号被移位到前面的寄存器中。 执行 LEDR 指令 ，将 SD64 到 SD79 的 内容向前移动一位。 (这也可以通过使用 Q3A/Q4ACPU 的 INDICATOR RESET 开关来进行。) 在检测到 16 个报警器之后，检测到的第 17 个不会被存储到 SD64 到 SD79 中。 SET SET SET RST SET SET SET SET SET SET SET F50 F25 F99 F25 F15 F70 F65 F38 F110 F151 F210 LEDR SD62 | 0 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 99 | (检测到的号码) SD63 | 0 | 1 | 2 | 3 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 8 | (检测到的报警器数) SD64 | 0 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 99 | SD65 | 0 | 0 | 25 | 25 | 99 | 99 | 99 | 99 | 99 | 99 | 99 | 99 | 15 | SD66 | 0 | 0 | 0 | 99 | 0 | 15 | 15 | 15 | 15 | 15 | 15 | 15 | 70 | SD67 | 0 | 0 | 0 | 0 | 0 | 0 | 70 | 70 | 70 | 70 | 70 | 70 | 65 | SD68 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 65 | 65 | 65 | 65 | 65 | 38 | SD69 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 38 | 38 | 38 | 38 | 110 | SD70 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 110 | 110 | 110 | 151 | SD71 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 151 | 151 | 210 | SD72 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 210 | 0 | SD73 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SD74 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SD75 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SD76 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SD77 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SD78 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SD79 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | (检测到的号码)	S(指令执行)	D9125 D9126 D9127 D9128 D9129 D9130 D9131 D9132 新增 新增 新增 新增 新增 新增 新增 新增 新增	○
SD80	CHK 号	CHK 号	• 由 CHK 指令检测到的出错代码以 BCD 代码存储。	S(指令执行)	新增	QnA Qn(H) QnPH QnPRH

附表 .18 特殊寄存器列表

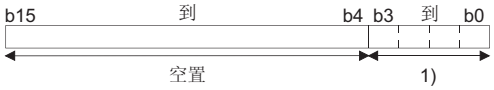
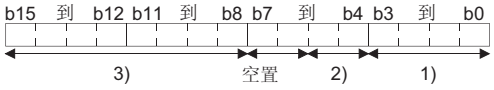
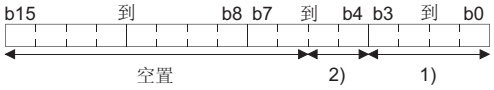
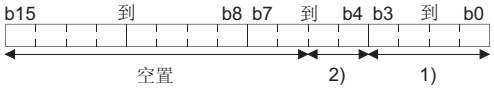
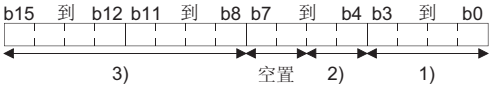
号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD90	步转移看门狗定时器设定值（只有当 SFC（只有当 SFC	用于定时器设定值和时间溢出出错的 F 号	对应 SM90	U	D9108	QnA Qn(H) QnPH QnPRH
SD91			对应 SM91		D9109	
SD92			对应 SM92		D9110	
SD93			对应 SM93		D9111	
SD94			对应 SM94		D9112	
SD95			对应 SM95		D9113	
SD96			对应 SM96		D9114	
SD97			对应 SM97		新增	
SD98			对应 SM98		新增	
SD99			对应 SM99		新增	
			• 设定当步转移看门狗定时器设置或者看门狗定时器时限出错发生时，将变为 ON 的报警器号（F 号）。 b15 到 b8 b7 到 b0 F号设置（0 到 255） 定时器时间限设置（1 到 255s：（1s 为单位）） • 在活动步中将 SM90 到 SM99 中的任何一个变为 ON 以启动定时器，并且如果在定时器时间限以内没有满足到下一个相应步的转移条件，则设定的报警器（F）变为 ON。			
SD100	传送速度存储区域通讯	存储在串行通讯设置中指定的传送速度。	96 : 9.6kbps, 192 : 19.2kbps, 384 : 38.4kbps, 576 : 57.6kbps, 1152 : 115.2kbps	S（电源接通或者复位）	新增	Q00/Q01
SD101	设置存储区域	存储在串行通讯设置中指定的通讯设置。	b15 到 b6 b5 b4 b3 到 b0 * 运行中程序修改设置 0：否 1：允许 * 总数检查是/否 0：是 1：否 *：由于数据由系统使用，所以未定义。	S（电源接通或者复位）	新增	
SD102	传送时间等待存储区域	存储在串行通讯设置中指定的传送等待时间。	0 : 无等待时间 1 到 FH : 等待时间（单位：10ms） 默认为 0。	S（电源接通或者复位）	新增	
SD105	CH1 传送速度设置（RS-232）	存储当采用 GX Developer 时预设的传送速度。	96 : 9600bps, 192 : 19.2kbps, 384 : 38.4kbps, 576 : 57.6kbps, 1152 : 115.2kbps *：RS-232 以外的连接，保持 RS-232 连接时数据不变。 （未连接时，默认值为 1152。）	S	新增	Qn(H) QnPH QnPRH QnU Rem
SD110	数据发送结果存储区域	当使用串行通讯功能时，存储数据发送结果。	数据发送时的出错代码被存储。	S（出错）	新增	Q00/Q01
SD111	数据接收结果存储区域	当使用串行通讯功能时，存储数据接收结果。	存储数据接收时的出错代码。	S（出错）		
SD119	电池寿命延长原因	电池寿命延长原因	存储使电池寿命延长功能有效的原因。 SD119 不为 0 时，电池寿命延长功能有效。 0：无原因 1：原因 b15 到 b2 b1 b0 b0：CPU开关设置 b1：正在通过锁存数据备份功能进行到标准ROM的备份	S（状态改变）	新增	QnU

附表 . 18 特殊寄存器列表

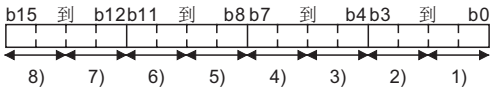
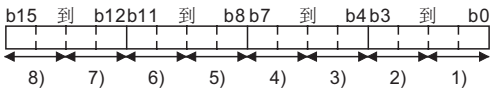
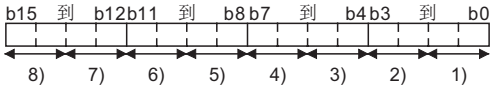
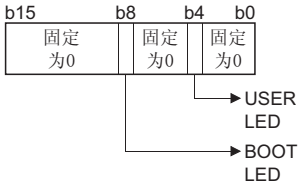
号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD130	保险丝熔断的模块	以 16 点为单位的位模式表示保险丝熔断的模块 0：无保险丝熔断 1：有保险丝熔断存在	• 保险丝熔断的输出模块的号码以位模式（以 16 点为单位）输入。 （如果模块号是由参数设定的，则参数设定的号码被存储。）  • 不清除，即使将熔断的保险丝更换为一个新保险丝。 此标志由出错复位操作清除	S（出错）	新增	Q00J/Q00/Q01
SD131						
SD132						
SD133						
SD134						
SD135						
SD136						
SD137						
SD150	I/O 模块验证出错	位模式，以 16 点为单位，表示验证出错的模块。 0：无 I/O 验证出错 1：I/O 验证出错存在	• 当 I/O 模块，其数据和电源接通时输入的数据不同，被检测到时，I/O 模块号（以 16 点为单位）被输入到位模式中。（当执行参数设置时，在参数中预设 I/O 模块号设置。）  • 不清除，即使将熔断的保险丝更换为新的保险丝。 此标志由出错复位操作清除。			
SD152						
SD153						
SD154						
SD155						
SD156						
SD157						

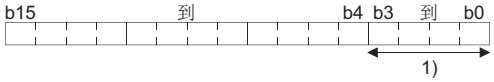
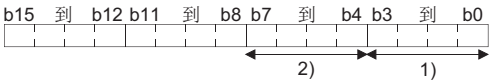
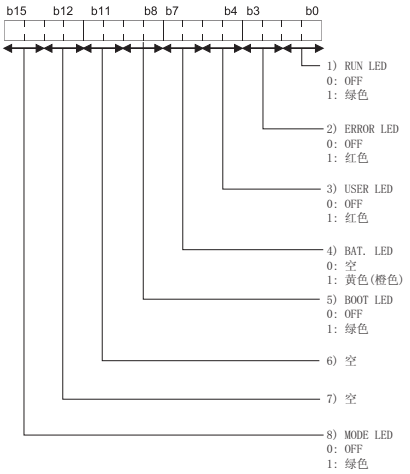
(2) 系统信息

附表 .20 特殊寄存器列表

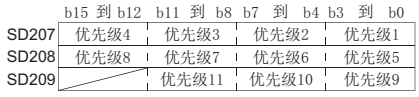
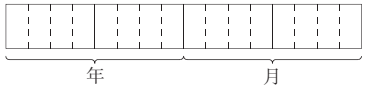
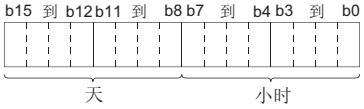
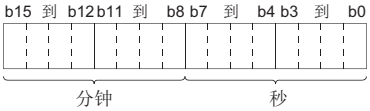
号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD200	开关的状态	CPU 开关的状态	• 远程 I/O 模块的开关状态被以下列格式存储。  1) 远程 I/O 模块开关状态常 1：停止	S（经常）	新增	Rem
			• CPU 的开关状态被以下列格式存储。  1)：CPU开关状态 — 0：RUN；1：停止；2：L.CLR 2)：存储卡开关 — 常OFF 3)：DIP开关 — b8到b12对应系统设置开关1的SW1到SW5。0：OFF，1：ON。b13到b15空置。	S（每个 END 处理）	新增	Qn(H) QnPH QnPRH
			• CPU 的开关状态被以下列格式存储。  1)：CPU开关状态 — 0：RUN；1：停止 2)：内存卡开关常OFF — 常OFF	S（每个 END 处理）	新增	Q00J/Q00/Q01
			• CPU 的开关状态被以下列格式存储。  1)：CPU开关状态 — 0：RUN；1：停止 2)：内存卡开关常OFF — 常OFF	S（RUN/STOP/RESET 开关改变时）	新增	QnU
			• CPU 的开关状态被以下列格式存储。  1)：CPU开关状态 — 0：RUN；1：停止；2：L.CLR 2)：存储卡开关 — b4对应到存储卡A，b5对应存储卡B；0：OFF，1：ON 3)：DIP开关 — b8到b12c对应系统设置开关1的SW1到SW5。b14和b15分别对应系统设置开关2的SW1和SW2。0：OFF，1：ON	S（每个 END 处理）	新增	QnA

附表 . 20 特殊寄存器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD201	LED 状态	CPU-LED 的状态	• 下列位模式用于存储 CPU 模块上 LED 的状态： • 0 是熄灭，1 是点亮，2 是闪烁。  1)：RUN　5)：BOOT 2)：ERR.　6)：空　空模式位模式 3)：USER　7)：空　0：OFF 1：绿色 4)：BAT.　8)：MODE　2：橙色 （基本型 QCPU 不包括 3）到 8）。）	S（状态改变）	新增	QCPU
			• 下列位模式用于存储 CPU 模块上 LED 的状态： • 0 是熄灭，1 是点亮，2 是闪烁。  1)：RUN　5)：BOOT 2)：ERROR　6)：空 3)：USER　7)：空 4)：BAT.　8)：MODE	S（状态改变）	新增	QnU
			• 下列位模式用于存储 CPU 模块上 LED 的状态： • 0 是熄灭，1 是点亮，2 是闪烁  1)：RUN　5)BOOT 2)：ERROR　6)：CARD A（存储卡 A） 3)：USER　7)：CARD B（存储卡 B） 4)：BAT.ALARM　8)：空	S（状态改变）	新增	QnA
SD202	LED 熄灭命令	熄灭的 LED 的位模式	•　存储熄灭的 LED 的位模式 （只允许 USER 和 BOOT） • 1：熄灭，0：不熄灭	U	新增	
			• 使用此寄存器指定要被熄灭的 LED，将 SM202 从 OFF 变为 ON 以熄灭指定的 LED。 可以指定 USER 和 BOOT 为将要熄灭的 LED。 • 以下列位模式指定将要熄灭的 LED。 （1：熄灭，0：不熄灭）  	U	新增	Qn(H) QnPH QnPRH QnU

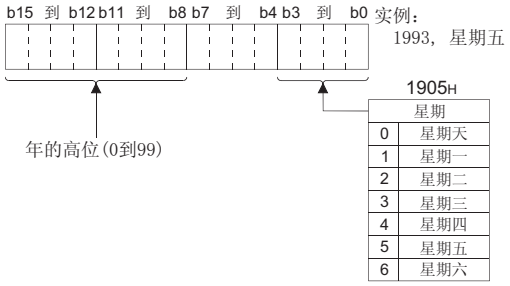
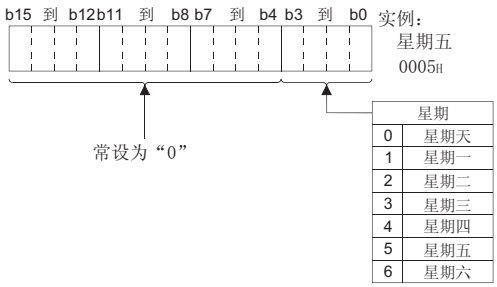
号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
			• 远程 I/O 模块的运行状态被以在下列格式存储。  1) 远程 I/O 模块 运行状态 2：停止	S（经常）	新增	Rem
SD203	CPU 的运行状态	CPU 的运行状态	• CPU 运行状态按照下图所示进行存储：  1)：CPU的运行状态 0：RUN 1：STEP-RUN(仅用于QnACPU) 2：STOP 3：PAUSE 2)：STOP/PAUSE的原因 0：在远程操作程序中来自RUN/STOP开关的指令(对于基本型QCPU，是“RUN/STOP/RESET开关”) 1：远程触点 2：GX Developer/串行通讯等 3：内部程序指令 注释：优先级是先来先执行 4：出错	S（每个 END 处理）	D9015 格式改变	○
SD204	LED 显示色彩	CPU-LED 显示色彩	• SD201 的 1) 到 8) 表示 LED 状态时，LED 显示色彩。 	S（状态改变）	新增	QnU

附表 .20 特殊寄存器列表

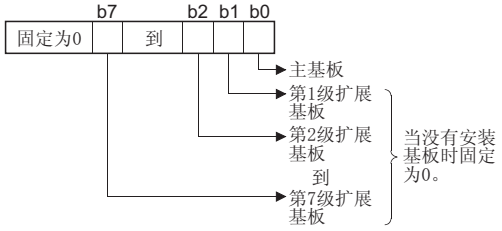
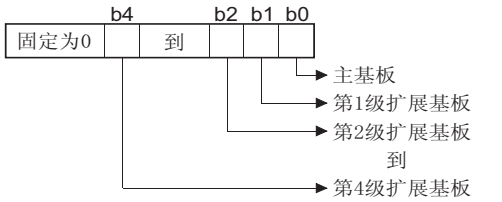
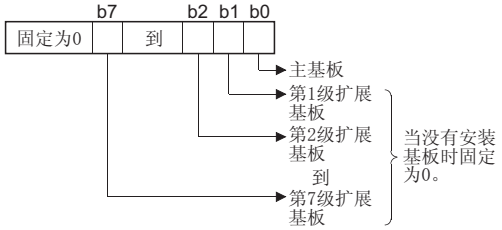
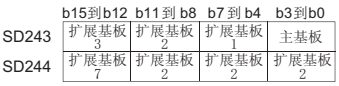
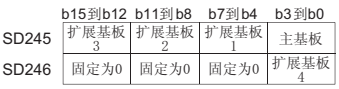
号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD206	软元件测试执行类型	0：测试还没有执行 1：在 X 软元件测试过程中 2：在 Y 软元件测试过程中 3：在 X/Y 软元件测试过程中	• 当软元件测试模式是在 GX Developer 上执行时，进行设置。	S（请求）	新增	Rem
SD207	LED 显示优先级	优先级 1 到 4	• 当产生出错时，LED 显示（闪烁）对应出错号设置优先级。 （基本型 QCPU 只支持报警器（出错条目号 7）。 • 在通用型 QCPU 中，设置发生异常时的各优先顺序相应出错 LED 的执行 / 不执行。 • 用于优先级的设置区域如下所示：  （当使用冗余CPU时，优先级11有效。） 默认值 SD207 = 4321H(0000H 对于基本型 QCPU) SD208 = 8765H(0700H 对于基本型 QCPU) (0765H 对于冗余 CPU) SD209 = 00A9H(0000H 对于基本型 QCPU) (0B09H 对于冗余 CPU) • 如果 “0” 被设定，则无显示。 • 在基本型 QCPU 中，如果 “7” 被设定到优先级 1 到 11 中的某一个，则当报警器变为 ON 时 ERR. LED 变为 ON。 • 在基本型 QCPU 中，如果 “7” 没有被设定到优先级 1 到 11 中的某一个，则当报警器变为 ON 时 ERR. LED 不变为 ON。 但是，即使设定了 “0”，信息有关的 CPU 模块操作停止（包括参数设置）出错将会被 LED 无条件地显示。	U	D9038	QnA Q00J/Q00/Q01*9 Qn(H) QnPH QnPRH QnU
SD208		优先级 5 到 8			D9039 格式改变	
SD209		优先级 9 到 11			新增	
SD210	时钟数据	时钟数据（年，月）	• 年（后两位数）和月按如下所示，以 BCD 码存储：  实例： 7月，1993 9307H 年 月	S（请求）/U	D9025	○ Rem
SD211	时钟数据	时钟数据（日期，小时）	• 日期和小时按如下所示，以 BCD 码存储：  实例： 31日，上午10 3110H 天 小时		D9026	
SD212	时钟数据	时钟数据（分钟，秒）	• 分钟和秒（在小时之后）按如下所示，以 BCD 码存储：  实例： 35分钟，48秒。 3548H 分钟 秒		D9027	

*9：适用于功能版本为 B 或者更高的 CPU。

附表 . 20 特殊寄存器列表

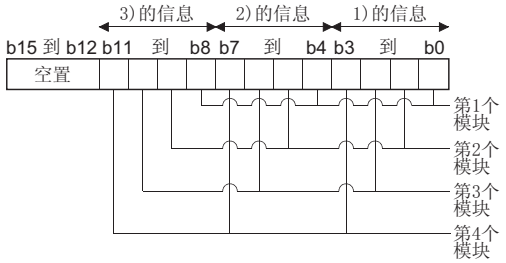
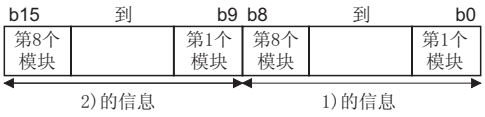
<table>
<tr><th>号码</th><th>名称</th><th>含义</th><th>解释</th><th>设置方式
（当被设定时）</th><th>相应的
ACPU
D9□□□</th><th>相应的 CPU</th></tr>
<tr><td rowspan="2">SD213</td><td rowspan="2">时钟数据</td><td>时钟数据（年的前位，星期）</td><td>• 按如下所示，以 BCD 码存储年（前两位）和星期。
 
</td><td rowspan="2">S（请求）/U</td><td rowspan="2">D9028</td><td>QCPU
Rem</td></tr>
<tr><td>时钟数据（星期）</td><td>• 按如下所示，星期是以 BCD 码存储：
 
</td><td>QnA</td></tr>
<tr><td>SD220
SD221
SD222
SD223
SD224
SD225
SD226
SD227</td><td>LED 显示数据</td><td>LED 显示数据</td><td>• LED 显示存储在这里的 ASCII 数据（16 字符）。
（在基本型 QCPU 中，寄存器存储出错发生（包括报警器 ON）时的消息（16 字符的 ASCII 数据）。
<table><tr><th></th><th>b15 到 b8 b7 到 b0</th></tr><tr><td>SD220</td><td>右起第15个字符</td><td>右起第16个字符</td></tr><tr><td>SD221</td><td>右起第13个字符</td><td>右起第14个字符</td></tr><tr><td>SD222</td><td>右起第11个字符</td><td>右起第12个字符</td></tr><tr><td>SD223</td><td>右起第9个字符</td><td>右起第10个字符</td></tr><tr><td>SD224</td><td>右起第7个字符</td><td>右起第8个字符</td></tr><tr><td>SD225</td><td>右起第5个字符</td><td>右起第6个字符</td></tr><tr><td>SD226</td><td>右起第3个字符</td><td>右起第4个字符</td></tr><tr><td>SD227</td><td>右起第1个字符</td><td>右起第2个字符</td></tr></table>• 在基本型 QCPU 和通用型 QCPU 中，CHK 指令执行时的显示软元件数据不被存储。</td><td>S（当改变时）</td><td>新增</td><td>○</td></tr>
<tr><td>SD235</td><td>直到运行中模块更换正在执行的基板</td><td>运行中模块更换正在执行的基板的起始 I/O 号 /10H</td><td>• 10H 被加到运行中模块更换正在执行的模块的起始 I/O 号的值上。</td><td>S（在运行中模块更换过程中）</td><td>新增</td><td>QnPH
QnPRH</td></tr>
<tr><td>SD240</td><td>基板模式</td><td>0：自动模式
1：详细模式</td><td>• 存储基板模式。</td><td>S（初始化）</td><td>新增</td><td rowspan="2">QCPU
Rem</td></tr>
<tr><td>SD241</td><td>扩展基板数</td><td>0：只有主基板
1 到 7：扩展基板数</td><td>• 存储安装的最大扩展基板数。</td><td>S（初始化）</td><td>新增</td></tr>
</table>

附表 .20 特殊寄存器列表

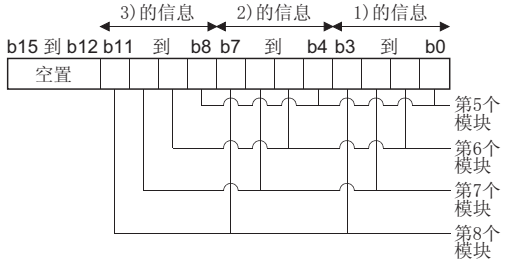
号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD242	A/Q 基板差别	基板类型差别 0：安装了 QA**B（A 模式） 1：安装了 Q**B（Q 模式）		S（初始化）	新增	Qn(H) QnPH QnPRH Rem
	安装的 Q 基板存在 / 不存在	基板类型差别 0：没有安装基板 1：Q**B 安装了		S（初始化）	新增	Q00J/Q00/Q01 Rem
	安装的 Q 基板存在 / 不存在	基板类型差别 0：没有安装基板 1：Q**B 安装了		S（初始化）	新增	QnU
SD243 SD244	基板插槽数	基板插槽数	 • 如上所示，每个区域存储安装的插槽数。	S（初始化）	新增	Qn(H) QnPH QnPRH QnU Rem
SD243 SD244	基板插槽数（操作状态）	基板插槽数	 • 如上所示，每个区域存储安装的插槽数。 （当进行参数设置时设定插槽数）	S（初始化）	新增	Q00J/Q00/Q01 Rem
SD245 SD246	基板插槽数（安装状态）	基板插槽数	 • 如上所示，每个区域存储基板上安装的模块的插槽数 （安装的基板的实际插槽数）。	S（初始化）	新增	Q00J/Q00/Q01*9
SD250	安装的最大 I/O	安装的最大 I/O 号	• 当 SM250 从 OFF 变为 ON 时，以 BIN 值存储安装的模块的最后 I/O 号的高 2 位数加 1 后的值。	S（请求 END）	新增	Qn(H) QnPH QnPRH Rem
			• 以 BIN 值存储安装的模块的最后 I/O 号的高 2 位数加 1 后的值。	S（初始化）	新增	Q00J/Q00/Q01 QnU Rem
SD251	用于更换的起始 I/O 号	用于模块更换的起始 I/O 号	• 存储在运行状态（电源接通）中被卸载 / 更换的 I/O 模块的起始 I/O 号的高两位数。（默认值：100H）。	U	D9094	Q2A(S1) Q3A Q4A Q4AR
SD253	RS422 传送速度	RS422 传送速度	• 存储 RS422 的传送速度。 0 ： 9600bps 1 ： 19.2kbps 2 ： 38.4kbps	S（当改变时）	新增	QnA

*9: 适用于功能版本为 B 或者更高的 CPU。

附表 .20 特殊寄存器列表

号码	名称	含义		解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD254	MELSECNET/10. MELSECNET/H 信息	安装的模块数		• 表示安装的 MELSECNET/10 模块或者 MELSECNET/H 模块的数目。	S（初始化）	新增	○
SD255		来自第 1 个模块的信息	I/O 号	• 表示安装的 MELSECNET/10 模块或者 MELSECNET/H 模块的 I/O 号			
SD256			网络号	• 表示安装的 MELSECNET/10 模块或者 MELSECNET/H 模块的网络号			
SD257			组号站号	• 表示安装的 MELSECNET/10 模块或者 MELSECNET/H 模块的组号			
SD258			站号	• 表示安装的 MELSECNET/10 模块或者 MELSECNET/H 模块的站号			
SD259			待机信息	• 在待机站中，待机站的模块数被存储。(1 到 4)			QnA Qn(H) QnPH QnPRH QnU
SD260 到 SD264		来自第 2 个模块的信息		• 配置和第一个模块的相同。			
SD265 到 SD269		来自第 3 个模块的信息		• 配置和第一个模块的相同			QnA Qn(H) QnPH QnPRH QnU*10
SD270 到 SD274		来自第 4 个模块的信息		• 配置和第一个模块的相同。			
SD280	CC-Link 出错	出错检测状态		1) 当安装的 CC-Link 模块的 Xn0 变为 ON 时，相应站的位信息变为 1(ON)。 2) 当安装的 CC-Link 模块的 Xn1 或者 XnF 变为 OFF，相应站的位信息变为 1(ON)。 3) 变为 1，(ON) 当安装的 CC-Link 模块和 CPU 模块之间不能通讯时。  上面的第 n 号模块是安装起始 I/O 号码排序的。 （但是，没有进行参数设置的 I/O 号没有考虑在内。）	S（出错）	新增	Qn(H) QnPH QnPRH Rem
				• 当安装的 CC-Link 模块的 Xn0 变为 ON 是，相应站的位信息变为 1(ON)。 • 当安装的 CC-Link 模块的 Xn1 或者 XnF 变为 OFF 时，相应站的位信息变为 1(ON)。 • 变为 1(ON)，当安装的 CC-Link 模块和 CPU 模块不能通讯时。 	S（出错）	新增	QnA

*10: 除 Q02UCPU 外的通用型 QCPU。

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD281	CC-Link 出错	出错检测状态	1) 当安装的 CC-Link 模块的 Xn0 变为 ON 时，相应站的位信息变为 1(ON)。 2) 当安装的 CC-Link 模块的 Xn1 或者 XnF 变为 OFF，相应站的位信息变为 1(ON)。 3) 当安装的 CC-Link 模块和 CPU 模块之间不能通讯时变为 1(ON)。  上面的第 n 号模块是按照起始 I/O 号码排序的。 （但是，没有进行参数设置的 I/O 号没有考虑在内。）	S（出错）	新增	Qn(H)*14 QnPH*14
SD290	软元件分配（和参数内容相同）	分配给 X 的点数	• 存储当前为 X 软元件分配的点数	S（初始化）	新增	○ Rem
SD291		分配给 Y 的点数	• 存储当前为 Y 软元件分配的点数			
SD292		分配给 M 的点数	• 存储当前为 M 软元件分配的点数			
SD293		分配给 L 的点数	• 存储当前为 L 软元件分配的点数			○
SD294		分配给 B 的点数	• 存储当前为 B 软元件分配的点数			○ Rem
SD295		分配给 F 的点数	• 存储当前为 F 软元件分配的点数			○
SD296		分配给 SB 的点数	• 存储当前为 SB 软元件分配的点数			○ Rem

*14: 以序列号的高 5 位为 “08032”以后的 CPU 为对象。

附表 .20 特殊寄存器列表

号码	名称	含义		解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD297	软元件分配（和参数内容相同）	分配给 V 的点数		• 存储当前为 V 软元件分配的点数	S（初始化）	新增	○
SD298		分配给 S 的点数		• 存储当前为 S 软元件分配的点数			
SD299		分配给 T 的点数		• 存储当前为 T 软元件分配的点数			
SD300		分配给 ST 的点数		• 存储当前为 ST 软元件分配的点数			
SD301		分配给 C 的点数		• 存储当前为 C 软元件分配的点数			
SD302		分配给 D 的点数		• 存储当前为 D 软元件分配的点数			○ Rem
SD303		分配给 W 的点数		• 存储当前为 W 软元件分配的点数			
SD304		分配给 SW 的点数		• 存储当前为 SW 软元件分配的点数			
SD305	软元件分配（变址寄存器）	16 位修饰 Z 分配点数		• 存储以 16 位范围修饰的变址寄存器 (Z) 的点数。（通过参数的 ZR 软元件的变址修饰设置）	S（初始化）	新增	QnU
SD315	为通讯处理预留的时间	为通讯处理预留的时间		• 为与 GX Developer 或者其他单元进行的通讯处理预留指定的时间。 • 指定的值越大，与其他软元件 (GXD eveloper，串行通讯单元）通讯的响应时间就越短。 • 如果指定的值超出了下面的范围，则按没有设置处理。 • 设置范围：1 到 100ms • 扫描时间将延长指定的时间。	U	新增	QCPU Rem
SD340	以太网信息	安装的模块数		• 表示安装的以太网模块数。	S（初始化）	新增	
SD341		来自第 1 个模块的信息	I/O 号	• 表示安装的以太网模块的 I/O 号			
SD342			网络号	• 表示安装的以太网模块的网络号			
SD343			组号	• 表示安装的以太网模块的组号			
SD344			站号	• 表示安装的以太网模块的站号			
SD345 到 SD346			空置	• 空置 （对于 QCPU，第一个模块的以太网 IP 地址被存储在缓冲存储区内。）			Qn(H) QnPH QnPRH QnU Rem
SD347			空置	• 空置 （对于 QCPU，用 ERRORRD 指令读取第一个模块的以太网出错代码。）			
SD348 到 SD354	以太网信息	来自第 2 个模块的信息		• 配置和第一个模块的相同。	S（初始化）	新增	
SD355 到 SD361		来自第 3 个模块的信息		• 配置和第一个模块的相同。			Qn(H) QnPH QnPRH QnU*10 Rem
SD362 到 SD368		来自第 4 个模块的信息		• 配置和第一个模块的相同。			

*10: 除 Q02UCPU 外的通用型 QCPU。

附表 .20 特殊寄存器列表

号码	名称	含义		解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD340	以太网信息	安装的模块数		• 表示安装的以太网模块数。	S（初始化）	新增	QnA
SD341		来自第 1 个模块的信息	I/O 号	• 表示安装的以太网模块的 I/O 号			
SD342			网络号	• 表示安装的以太网模块的网络号			
SD343			组号	• 表示安装的以太网模块的组号			
SD344			站号	• 表示安装的以太网模块的站号			
SD345 到 SD346			IP 地址	• 表示安装的以太网模块的 IP 地址			
SD347			出错代码	• 表示安装的以太网模块的出错代码			
SD348 到 SD354		来自第 2 个模块的信息		• 配置和第一个模块的相同。			
SD355 到 SD361		来自第 3 个模块的信息		• 配置和第一个模块的相同。			
SD362 到 SD368		来自第 4 个模块的信息		• 配置和第一个模块的相同。			
SD380	以太网指令接收状态	第 1 个模块的指令接收状态		b15 到 b8 b7 b6 b5 b4 b3 b2 b1 b0 0 未使用 通道1的指令接收状态 通道2的指令接收状态 通道3的指令接收状态 通道4的指令接收状态 通道5的指令接收状态 通道6的指令接收状态 通道7的指令接收状态 通道8的指令接收状态 ON：接收(通道使用时。) OFF：未接收(通道是未使用。)	S（初始化）	新增	QnA
					S（指令执行）		QnPRH
SD381	以太网指令接收状态	第 2 个模块的指令接收状态		• 配置和第一个模块的相同。			
SD382		第 3 个模块的指令接收状态		• 配置和第一个模块的相同。			
SD383		第 4 个模块的指令接收状态		• 配置和第一个模块的相同。			
SD392	软件版本	内部系统软件版本		• 以 ASCII 码存储内部系统软件版本。 高字节 | 低字节 存储在低字节 存储在高字节 对于版本“A”，举例来说，“41H”被存储。 注释：内部系统软件版本可能和打印在外壳上的版本符号所指示的版本不同。	S（初始化）	D9060	QnA

附表 .20 特殊寄存器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD393	多 CPU 系统信息	多 CPU 数	• 组成多可编程控制器系统的 CPU 模块数被存储。(1 到 3，空置的也包括）	S（初始化）	新增	Q00/Q01*9 QnU
SD394		CPU 安装信息	• 1 号 CPU 到 3 号 CPU 的 CPU 模块类型以及是否安装 CPU 模块，这些信息被存储。 b15 到 b12 b11 到 b8 b7 到 b4 b3 到 b0 SD394 空置(0) | 3号CPU | 2号CPU | 1号CPU CPU模块安装或者未安装 0：未安装 1：安装 CPU模块类型 0：PLC CPU 1：运动CPU 2：PC CPU			Q00/Q01*9
SD395		多 CPU 号	• 在多 CPU 系统配置中，主站 CPU 的 CPU 号被存储。 1 号 CPU：1，2 号 CPU：2，3 号 CPU：3，4 号 CPU：4	S（初始化）	新增	Q00/Q01*9 Qn(H)*9 QnPH QnU
SD396		1 号 CPU 运行状态	在多 CPU 系统配置中，本站 CPU 的 CPU 号被存储。 （在 SD393 中指示的多可编程控制器的号码信息被存储。） b15 b14 到 b8 b7 到 b4 b3 到 b0 空置 | 分类 | 运动状态 安装状态 0：未安装 1：安装 0 ：正常 1 ：监视故障 2 ：媒介故障 3 ：重大故障 FH：复位 0：RUN 2：STOP 3：PAUSE 4：初始化 FH：复位	S（初始化 / END 处理出错发生时）	新增	Q00/Q01*9 QnU
SD397		2 号 CPU 运行状态				
SD398		3 号 CPU 运行状态				
SD399		4 号 CPU 运行状态				QnU

*9：适用于功能版本为 B 或者更高的 CPU。

(3) 系统时钟 / 计数器

附表 .21 特殊寄存器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD412	1 秒计数器	以 1 秒为单位计数	• 下列可编程控制器 CPU 模块 RUN 时，每秒增加 1 • 计数在 0 到 32767 到 -32768 到 0 之间重复	S（状态改变）	D9022	○
SD414	2n 秒时钟设置	2n 秒时钟单位	• 存储 2n 秒时钟的值 n（默认是 30） • 设置可以在 1 到 32767 之间进行	U	新增	
SD415	2nms 时钟设置	2nms 时钟单位	• 存储 2nms 时钟的值 n（默认是 30） • 设置可以在 1 到 32767 之间进行	U	新增	Qn(H) QnPH QnPRH QnU
SD420	扫描计数器	每个扫描周期的计数值	• 在 CPU 模块被设定为 RUN 之后，每个扫描执行增加 1。 （初始化执行类型程序中，扫描周期不计数。） • 计数在 0 到 32767 到 -32768 到 0 之间重复	S（每个 END 处理）	新增	QnA Qn(H) QnPH QnPRH QnU
			• 在 CPU 模块被设定为 RUN 之后，每个扫描执行增加 1。 • 计数在 0 到 32767 到 -32768 到 0 之间重复	S（每个 END 处理）	新增	Q00J/Q00/Q01
SD430	低速扫描计数器	在每个扫描周期中的计数值	• 在 CPU 模块被设定为 RUN 之后，每个扫描执行增加 1。 • 计数在 0 到 32767 到 -32768 到 0 之间重复 • 只用于低速执行类型程序	S（每个 END 处理）	新增	QnA Qn(H) QnPH

(4) 扫描周期信息

附表 .22 特殊寄存器列表

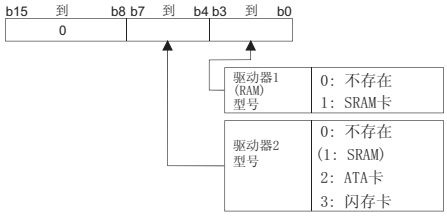
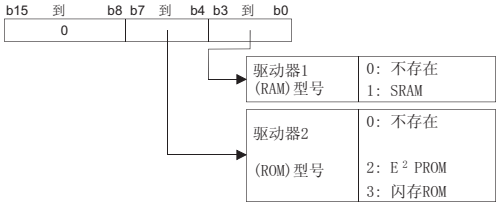
号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD500	执行程序号	正在执行的程序号	• 当前正在执行的程序的程序号被以 BIN 值存储。	S（状态改变）	新增	QnA Qn(H) QnPH QnPRH QnU
SD510	低速执行类型程序号	正在执行的低速执行类型程序号	• 当前正在执行的低速执行类型程序的程序号被以 BIN 值存储。 • 只有当 SM510 是 ON 时才允许。	S（每个 END 处理）	新增	QnA Qn(H) QnPH
SD520	当前扫描周期时间	当前扫描周期时间（以 1ms 为单位）	• 当前扫描时间被存储到 SD520 和 SD521 中。 （测量是以 100μs 为单位。（对于通用型 QCPU，以 1μs 为单位。）） SD520：存储 ms 的值。（存储范围：0 到 65535） SD521：存储μs 的值。（存储范围：0 到 900（对于通用型 QCPU，存储范围：0 到 999）） （实例）当当前扫描时间是 23.6ms 时，下列值被存储。 SD520 = 23 SD521 = 600	S（每个 END 处理）	D9017 格式改变	QnA Q00J/Q00/Q01 Qn(H) QnPH QnPRH QnU
SD521		当前扫描周期时间（以 100μs 为单位）		S（每个 END 处理）	新增	
SD522	初始化扫描周期时间	初始化扫描周期时间（以 1ms 为单位）	• 存储初始化执行类型程序的扫描时间到 SD522 和 SD523。 （测量是以 100μs 为单位。（对于通用型 QCPU，以 1μs 为单位。）） SD522：存储 ms 的地方。（存储范围：0 到 65535） SD523：存储μs 的地方。（存储范围：0 到 900（对于通用型 QCPU，存储范围：0 到 999））	S（每个 END 处理）	新增	QnA Qn(H) QnPH QnPRH QnU
SD523		初始化扫描时间（以 100μs 为单位）				
SD524	最小扫描周期时间	最小时间（以 1ms 单位）	• 存储扫描时间的最小值到 SD524 和 SD525 中，初始化执行类型程序除外（测量是以 100μs 为单位。） （测量是以 100μs 为单位。（对于通用型 QCPU，以 1μs 为单位。）） SD524：存储 ms 的地方。（存储范围：0 到 65535） SD525：存储μs 的地方。（存储范围：0 到 900（对于通用型 QCPU，存储范围：0 到 999））	S（每个 END 处理）	D9018 格式改变	
SD525		最小扫描周期时间（以 100μs 为单位）		S（每个 END 处理）	新增	
SD526	最大扫描周期时间	最大扫描周期时间（以 1ms 为单位）	• 存储扫描时间的最大值到 D526 和 SD527 中，初始化执行类型程序除外。 （测量是以 100μs 为单位。（对于通用型 QCPU，以 1μs 为单位。）） SD526：存储 ms 的地方。（存储范围：0 到 65535） SD527：存储μs 的地方。（存储范围：0 到 900（对于通用型 QCPU，存储范围：0 到 999））	S（每个 END 处理）	D9019 格式改变	
SD527		最大扫描周期时间（以 100us 为单位）			新增	
SD528	低速执行类型程序的当前扫描周期时间	当前扫描时间（以 1ms 为为单位）	• 存储低速执行类型程序的当前扫描时间到 SD528 和 SD529。 （测量是以 100μs 为单位。） SD528：存储 ms 的地方。（存储范围：0 到 65535） SD529：存储μs 的地方。（存储范围：0 到 900）	S（每个 END 处理）	新增	QnA Qn(H) QnPH
SD529		当前扫描时间（以 100μs 为单位）				
SD532	低速执行类型程序的最小扫描周期时间	最小扫描周期时间（以 1ms 为单位）	• 存储低速执行类型程序的扫描时间的最小值到 SD532 和 SD533。 （测量是以 100μs 为单位。） SD532：存储 ms 的地方。（存储范围：0 到 65535） SD533：存储μs 的地方。（存储范围：0 到 900）	S（每个 END 处理	新增	
SD533		最小扫描周期时间（以 100μs 为单位）				
SD534	低速执行类型程序的最大扫描周期时间	最大扫描周期时间（以 1ms 为单位）	• 存储低速执行类型程序的扫描时间的最大值到 SD534 和 SD535，不包括第一个扫描周期的时间。 （测量是以 100μs 为单位。） SD534：存储 ms 的地方。（存储范围：0 到 65535） SD535：存储μs 的地方。（存储范围：0 到 900）	S（每个 END 处理）	新增	
SD535		最大扫描周期时间（以 100μus 为单位）				
SD540	END 处理时间	END 处理时间（以 1ms 为单位）	• 存储从扫描周期执行类型程序结束到下一个扫描周期的开始之间的时间到 SD540 和 SD541。 （测量是以 100μs 为单位。（对于通用型 QCPU，以 1μs 为单位。）） SD540：存储 ms 的地方。（存储范围：0 到 65535） SD541：存储μs 的地方。（存储范围：0 到 900（对于通用型 QCPU，存储范围：0 到 999））	S（每个 END 处理）	新增	QnA Qn(H) QnPH QnPRH QnU
SD541		END 处理时间（以 100μs 为单位）				
SD524	最小扫描周期时间	最小扫描周期时间（以 1ms 为单位）	• 存储扫描时间的最小值到 SD524 和 SD525。 （测量是以 100μs 为单位。） SD524：存储 ms 的地方。（存储范围：0 到 65535） SD525：存储μs 的地方。（存储范围：0 到 900）	S（每个 END 处理）	新增	Q00J/Q00/Q01
SD525		最小扫描周期时间（以 100μs 为单位）				
SD526	最大扫描周期时间	最大扫描周期时间（以 1ms 为单位）	• 存储扫描时间的最大值到 SD526 和 SD527。 （测量是以 100μs 为单位。） SD526：存储 ms 的地方。（存储范围：0 到 65535） SD527：存储μs 的地方。（存储范围：0 到 900）	S（每个 END 处理）		
SD527		最大扫描周期时间（以 100μs 为单位）				

附表 .22 特殊寄存器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD540	END 处理时间	END 处理时间（以 1ms 为单位）	• 存储从当前扫描周期程序结束到下一个扫描周期启动之间的时间到 SD540 和 SD541。 （测量是以 100μs 为单位。） SD540：存储 ms 的地方。（存储范围：0 到 65535） SD541：存储μs 的地方。（存储范围：0 到 900）	S（每个 END 处理）	新增	Q00J/Q00/Q01
SD541		END 处理时间（以 100μs 为单位）				
SD542	恒定扫描周期等待时间	恒定扫描周期等待时间（以 1ms 为单位）	• 存储恒定扫描设置的等待时间到 SD542 和 SD543。 （测量是以 100μs 为单位。（对于通用型 QCPU，以 1μs 为单位。）） SD542：存储 ms 的地方。（存储范围：0 到 65535） SD543：存储μs 的地方。（存储范围：0 到 900（对于通用型 QCPU，存储范围：0 到 999））	S（每个 END 处理）	新增	○
SD543		恒定扫描周期等待时间（以 100μs 为单位）				
SD544	低速执行类型程序的累积执行时间	低速执行类型程序的累积执行时间（以 1ms 为单位）	• 存储低速执行类型程序的累积执行时间到 SD544 和 SD545。 （测量是以 100μs 为单位。） SD544：存储 ms 的地方。（存储范围：0 到 65535） SD545：存储μs 的地方。（存储范围：0 到 900） • 在一个低速扫描周期结束之后清除到 0。	S（每个 END 处理）	新增	QnA Qn(H) QnPH
SD545		低速执行类型程序的累积执行时间（以 100μs 为单位）				
SD546	低速执行类型程序的执行时间	低速执行类型程序的执行时间（以 1ms 为单位）	• 在 1 个扫描周期内低速执行类型程序存储执行时间的 SD546 和 SD547。 （测量是以 100μs 为单位。） SD546：存储 ms 的地方。（存储范围：0 到 65535） SD547：存储μs 的地方。（存储范围：0 到 900） • 在每个扫描周期存储。	S（每个 END 处理）	新增	
SD547		低速执行类型程序的执行时间（以 100μs 为单位）				
SD548	扫描执行类型程序执行时间	扫描执行类型程序执行时间（以 1ms 为单位）	• 在 1 个扫描周期内存储扫描执行类型程序执行时间到 SD548 和 SD549。 （测量是以 100μs 为单位。） SD548：存储 ms 的地方。（存储范围：0 到 65535） SD549：存储μs 的地方。（存储范围：0 到 900） • 在每个扫描周期存储。	S（每个 END 处理）	新增	QnA Qn(H) QnPH QnPRH
SD549		扫描执行类型程序执行时间（以 100μs 为单位）				
SD548	扫描程序执行时间	扫描程序执行时间（以 1ms 为单位）	• 在 1 个扫描内存储扫描程序执行时间到 SD548 和 SD549。 （测量是以 100μs 为单位。（对于通用型 QCPU，以 1μs 为单位。）） SD548：存储 ms 的地方。（存储范围：0 到 65535） SD549：存储μs 的地方。（存储范围：0 到 900（对于通用型 QCPU，存储范围：0 到 999）） • 在每个扫描周期存储。	S（每个 END 处理）	新增	Q00J/Q00/Q01 QnU
SD549		扫描程序执行时间（以 100μs 为单位）				
SD550	服务间隔测量模块	基板 / 模块号	• 为测量服务间隔的模块设定 I/O 号	U	新增	QnA Qn(H) QnPH QnPRH
SD551	服务间隔时间	模块服务间隔（以 1ms 为单位）	• 当 SM551 变为 ON 时，将在 SD550 中指定的模块的服务间隔存储到 SD551 和 SD552。 （测量是以 100μs 为单位。） SD551：存储 ms 的地方。（存储范围：0 到 65535） SD552：存储μs 的地方。（存储范围：0 到 900）	S（请求）	新增	
SD552		模块服务间隔（以 100μs 为单位）				

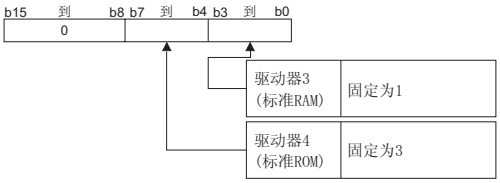
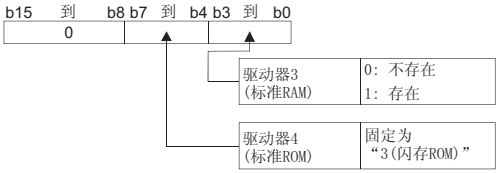
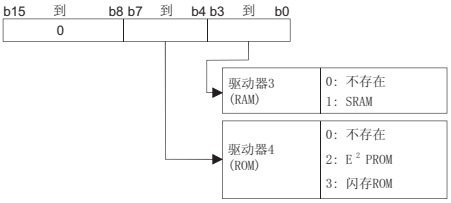
(5) 驱动器信息

附表 .23 特殊寄存器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD600	存储卡型号	存储卡型号	• 表示安装的存储卡型号 	S（初始化和卡卸载）	新增	Qn(H) QnPH QnPRH QnU
	存储卡 A 型号	存储卡 A 型号	• 表示安装的存储卡 A 型号 	S（初始化和卡卸载）	新增	QnA
SD602	驱动器 1(RAM 存储卡）容量	驱动器 1 容量	• 驱动器 1 容量被以 1k 字节为单位存储。 （存储格式化后的空余容量）	S（初始化和卡卸载）	新增	QnA Qn(H) QnPH QnPRH QnU
SD603	驱动器 2（ ROM 存储卡）容量	驱动器 2 的容量	• 驱动器 2 容量被以 1k 字节为单位存储。*1 （存储格式化后的空余容量）	S（初始化和卡卸载）	新增	Qn(H) QnPH QnPRH QnU
			• 驱动器 2 容量被以 1k 字节为单位存储。	S（初始化和卡卸载）	新增	QnA
SD604	存储卡使用情况	存储卡使用情况	• 存储卡 (A) 的使用情况以位模式存储。 （当 ON 时为使用中） • 这些位模式的意义如下所示： b0 ： 引导操作(QBT) b1 ： 参数(QPA) b2 ： 软元件注释(QCD) b3 ： 软元件初始化值(QDI) b4 ： 文件寄存器R(QDR) b5 ： 采样跟踪(QTD) b6 ： 未使用 b7 ： 未使用 b8 ： 未使用 b9 ： CPU故障历史(QFD) b10 ： 未使用 b11 ： 局部软元件(QDL) b12 ： 未使用 b13 ： 未使用 b14 ： 未使用 b15 ： 未使用	S（状态改变）	新增	Qn(H) QnPH QnPRH
	存储卡 A 使用情况	存储卡 A 使用情况	• 存储卡 A 的使用情况以位模式存储。 （当 ON 为使用中） • 这些位模式的意义如下所示： b0 ： 引导操作(QBT) b1 ： 参数(QPA) b2 ： 软元件注释(QCD) b3 ： 软元件初始化值(QDI) b4 ： 文件R(QDR) b5 ： 采样跟踪(QTD) b6 ： 状态锁存(QTL) b7 ： 程序跟踪(QTP) b8 ： 仿真数据(QDS) b9 ： CPU故障历史(QFD) b10 ： SFC跟踪(QTR) b11 ： 局部软元件(QDL) b12 ： 未使用 b13 ： 未使用 b14 ： 未使用 b15 ： 未使用	S（状态改变）	新增	QnA

*1: 使用了 Q2MEM-8MBA 时，根据高性能型 QCPU 的序列号和 ATA 卡的生产管理编号的不同，特殊寄存器 SD603 中存储的值也不相同。
详细内容请参阅 QCPU 用户手册（硬件设计 / 维护点检篇）。

附表 . 23 特殊寄存器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD604	存储卡使用情况	存储卡使用情况	• 存储卡 (A) 的使用情况以位模式存储。 （当 ON 时为使用中） • 这些位模式的意义如下所示： b0 : 引导操作(QBT)*1 / b8 : 未使用 b1 : 参数(QPA) / b9 : 未使用 b2 : 软元件注释(QCD) / b10 : 未使用 b3 : 软元件初始化值(QDI)*2 / b11 : 局部软元件(QDL) b4 : 文件寄存器R(QDR) / b12 : 未使用 b5 : 采样跟踪(QTD) / b13 : 未使用 b6 : 未使用 / b14 : 未使用 b7 : 未使用 / b15 : 未使用 *1: 引导开始时 ON，结束时 OFF。 *2: 软元件初始值反映开始时 ON，结束时 OFF。	S（状态改变）	新增	QnU
SD620	驱动器 3/4 型号	驱动器 3/4 型号	• 表示驱动器 3/4 型号。 	S（初始化）	新增	Qn(H) QnPH QnPRH QnU
			• 表示驱动器 3/4 型号。 	S（初始化）	新增	Q00J/Q00/Q01
	存储卡 B 型号	存储卡 B 型号	• 表示安装的存储卡 B 型号 	S（初始化 / 卡安装和卸载）	新增	Q2A(S1) Q3A Q4A Q4AR
SD622	驱动器 3（标准 RAM）容量	驱动器 3(RAM) 容量	• 驱动器 3 容量以 1k 字节为单位存储。 （存储格式化后的空余容量）	S（初始化）	新增	Qn(H) QnPH QnPRH QnU
				S（初始化 / 卡安装和卸载）	新增	Q2A(S1) Q3A Q4A Q4AR
SD623	驱动器 4（标准 ROM）容量	驱动器 4 容量	• 驱动器 4 容量以 1k 字节为单位存储。 （存储格式化后的空余容量）	S（初始化）	新增	Qn(H) QnPH QnPRH QnU
				S（初始化 / 卡安装和卸载）	新增	Q2A(S1) Q3A Q4A Q4AR

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD624	驱动器 3/4 使用情况	驱动器 3/4 使用情况	• 驱动器 3/4 的使用情况以位模式存储。（当 ON 时为使用中） • 这些位模式的意义如下所示： b0 : 引导操作(QBT)　b8 : 未使用 b1 : 参数(QPA)　b9 : CPU故障历史(QFD) b2 : 软元件注释(QCD)　b10 : SFC跟踪(QTS) b3 : 软元件初始化值(QDI)　b11 : 局部软元件(QDL) b4 : 文件R(QDR)　b12 : 未使用 b5 : 采样跟踪(QTD)　b13 : 未使用 b6 : 未使用　b14 : 未使用 b7 : 未使用　b15 : 未使用	S（状态改变）	新增	Qn(H) QnPH QnPRH
	存储卡 B 使用情况	存储卡 B 使用情况	• 存储卡 B 的使用情况以位模式存储。（当 ON 时为使用中） • 这些位模式的意义如下所示： b0 : 引导操作(QBT)　b8 : 仿真数据(QDS) b1 : 参数(QPA)　b9 : CPU故障历史(QFD) b2 : 软元件注释(QCD)　b10 : SFC跟踪(QTS) b3 : 软元件初始化值(QDI)　b11 : 局部软元件(QDL) b4 : 文件R(QDR)　b12 : 局部软元件(QDL) b5 : 采样跟踪(QTD)　b13 : 未使用 b6 : 状态锁存(QTL)　b14 : 未使用 b7 : 程序跟踪(QTP)　b15 : 未使用	S（状态改变）	新增	Q2A(S1) Q3A Q4A Q4AR
	驱动器 3/4 使用情况	驱动器 3/4 使用情况	• 驱动器 3/4 的使用情况以位模式存储。（当 ON 时为使用中） • 这些位模式的意义如下所示： b0 : 未使用　b8 : 未使用 b1 : 参数(QPA)　b9 : 未使用 b2 : 软元件注释(QCD)　b10 : 未使用 b3 : 软元件初始化值(QDI)*1　b11 : 局部软元件(QDL) b4 : 文件寄存器R(QDR)　b12 : 未使用 b5 : 采样跟踪(QTD)　b13 : 未使用 b6 : 未使用　b14 : 未使用 b7 : 未使用　b15 : 未使用 *1: 引导开始时 ON，结束时 OFF。	S（状态改变）	新增	QnU

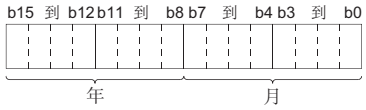
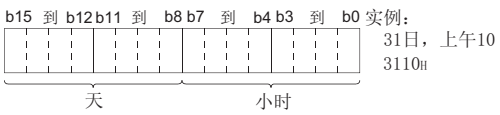
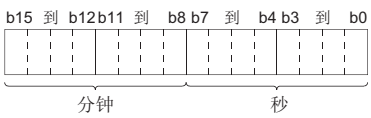
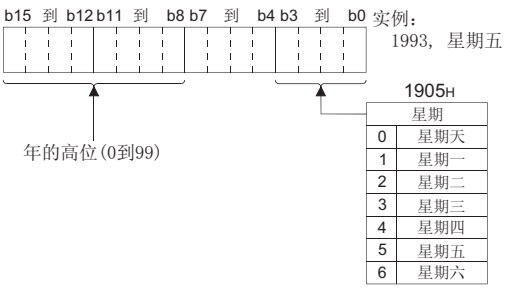
附表 .23 特殊寄存器列表

<table>
<tr><th>号码</th><th>名称</th><th>含义</th><th>解释</th><th>设置方式
（当被设定时）</th><th>相应的
ACPU
D9□□□</th><th>相应的 CPU</th></tr>
<tr><td>SD622</td><td>驱动器 3（标准 RAM）容量</td><td>驱动器 3 容量</td><td>• 驱动器 3 容量以 1k 字节为单位存储。</td><td>S（初始化）</td><td rowspan="3">新增</td><td rowspan="3">Q00J/Q00/Q01</td></tr>
<tr><td>SD623</td><td>驱动器 4（标准 ROM）容量</td><td>驱动器 4 容量</td><td>• 驱动器 4 容量以 1k 字节为单位存储。</td><td>S（初始化）</td></tr>
<tr><td>SD624</td><td>驱动器 3 使用情况</td><td>驱动器 3 使用情况</td><td>• 驱动器 3 使用情况以位模式存储。
b15 到 b5 b4 到 b0
0 | 0 0 0 0 0
文件寄存器(R)
0：未使用
1：使用中</td><td>S（状态改变）</td></tr>
<tr><td>SD640</td><td>文件寄存器驱动器</td><td>驱动器号：</td><td>• 存储文件寄存器正在使用的驱动器号</td><td>S（状态改变）
*10</td><td>新增</td><td>○</td></tr>
<tr><td>SD641</td><td rowspan="6">文件寄存器文件名称</td><td rowspan="6">文件寄存器文件名称</td><td rowspan="2">• 以 ASCII 代码存储在参数上选择的文件寄存器文件名称（带扩展）或者 QDRSET 指令使用的文件寄存器文件名称。
b15 到 b8 b7 到 b0
SD641 第2个字符 第1个字符
SD642 第4个字符 第3个字符
SD643 第6个字符 第5个字符
SD644 第8个字符 第7个字符
SD645 扩展的第1个字符 2EH(.)
SD646 扩展的第3个字符 扩展的第2个字符</td><td rowspan="2">S（状态改变）</td><td rowspan="2">新增</td><td rowspan="2">QnA
Qn(H)
QnPH
QnPRH
QnU</td></tr>
<tr><td>SD642</td></tr>
<tr><td>SD643</td><td rowspan="2">• 以 ASCII 代码存储在参数上选择的文件寄存器文件名称（MAIN.QDR）。
b15 到 b8 b7 到 b0
SD641 第2个字符(A) 第1个字符(M)
SD642 第4个字符(N) 第3个字符(I)
SD643 第6个字符() 第5个字符()
SD644 第8个字符() 第7个字符()
SD645 扩展的第1个字符(Q) 2EH(.)
SD646 扩展的第3个字符(R) 扩展的第2个字符(D)</td><td rowspan="2">S（初始化）</td><td rowspan="2">新增</td><td rowspan="2">Q00J/Q00/Q01</td></tr>
<tr><td>SD644</td></tr>
<tr><td>SD645</td></tr>
<tr><td>SD646</td></tr>
<tr><td rowspan="2">SD647</td><td rowspan="2">文件寄存器容量</td><td rowspan="2">文件寄存器容量</td><td rowspan="2">• 以 1k 字为单位存储当前选择的文件寄存器的数据容量。</td><td>S（状态改变）</td><td rowspan="2">新增</td><td>QnA
Qn(H)
QnPH
QnPRH
QnU</td></tr>
<tr><td>S（初始化）</td><td>Q00J/Q00/Q01</td></tr>
<tr><td>SD648</td><td>文件寄存器块号</td><td>文件寄存器块号</td><td>• 存储当前选择的文件寄存器块号。</td><td>S（状态改变）
*10</td><td>D9035</td><td>○</td></tr>
<tr><td>SD650</td><td>注释驱动器号</td><td>注释驱动器号</td><td>• 存储在参数上选择的注释驱动器号或者 QCDSET 指令使用的注释驱动器号。</td><td>S（状态改变）</td><td>新增</td><td rowspan="8">QnA
Qn(H)
QnPH
QnPRH
QnU</td></tr>
<tr><td>SD651</td><td rowspan="6">注释文件名称</td><td rowspan="6">注释文件名称</td><td rowspan="6">• 以 ASCII 代码存储在参数上选择的文件寄存器文件名称（带扩展）或者 QCDSET 指令使用的文件寄存器文件名称。
b15 到 b8 b7 到 b0
SD651 第2个字符 第1个字符
SD652 第4个字符 第3个字符
SD653 第6个字符 第5个字符
SD654 第8个字符 第7个字符
SD655 扩展的第1个字符 2EH(.)
SD656 扩展的第3个字符 扩展的第2个字符</td><td rowspan="6">S（状态改变）</td><td rowspan="6">新增</td></tr>
<tr><td>SD652</td></tr>
<tr><td>SD653</td></tr>
<tr><td>SD654</td></tr>
<tr><td>SD655</td></tr>
<tr><td>SD656</td></tr>
</table>

*10：在基本型 QCPU 中，在参数执行之后，在 SEOP 到 RUN 或者 RSET 指令执行时设定数据。

附表 .23 特殊寄存器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD660	引导操作指定文件	引导指定文件驱动器号	• 存储正被存储的引导启动指定文件 (*.QBT) 所在的驱动器号。	S（初始化）	新增	QnA Qn(H) QnPH QnPRH
SD661 SD662 SD663 SD664 SD665 SD666		引导指定文件的文件名称	• 存储引导启动指定文件的文件名 (*.QBT)。 b15 到 b8 b7 到 b0 SD661 第2个字符 第1个字符 SD662 第4个字符 第3个字符 SD663 第6个字符 第5个字符 SD664 第8个字符 第7个字符 SD665 扩展的第1个字符 2EH(.) SD666 扩展的第3个字符 扩展的第2个字符	S（初始化）	新增	
SD670	参数有效驱动器信息	参数有效驱动号	• 存储有效的参数存储目标驱动器的信息。 0：驱动器 0（程序内存） 1：驱动器 1(SRAM 卡） 2：驱动器 2（闪存卡 /ATA 卡） 4：驱动器 4（标准 ROM）	S（初始化）	新增	QnU
SD671	锁存数据备份功能的状态	状态显示	显示锁存数据备份功能的状态。 状态 / 有无备份数据 / 下一次以后的电源 OFF→ON时的还原动作 0 无备份数据 — 无 — 不进行还原。 1 还原准备完毕 — 有 — 仅在下一次的电源OFF→ON时进行还原。 2 还原执行完毕 — 有 — 不执行还原。 3 备份执行等待 — 有 — 不执行还原。 4 反复还原执行准备完毕 — 有 — 每次电源OFF→ON时执行还原。 • “2：还原执行完毕”为还原执行之后的状态。 “3：备份执行等待”为“2：还原执行完毕”时电源 OFF → ON 后的状态。	S（状态改变）	新增	QnU
SD672	备份信息	备份时间（年，月）	• 以 BCD 码的 2 位存储备份的年（后两位数）和月： b15 到 b12 b11 到 b8 b7 到 b4 b3 到 b0 实例： 7月，1993 9307H 年 月	S（写入时）	新增	QnU
SD673		备份时间（日期，小时）	• 以 BCD 码的 2 位存储备份的日期和小时： b15 到 b12 b11 到 b8 b7 到 b4 b3 到 b0 实例： 31日，上午10 3110H 天 小时			
SD674		备份时间（分钟，秒）	• 以 BCD 码的 2 位存储备份的分钟和秒： b15 到 b12 b11 到 b8 b7 到 b4 b3 到 b0 实例： 35分钟，48秒。 3548H 分钟 秒			
SD675		备份时间（年的前位，星期）	• 以 BCD 码的 2 位存储备份的年（前两位）和星期。 b15 到 b12 b11 到 b8 b7 到 b4 b3 到 b0 实例： 1993，星期五 1905H 年的高位(0到99) 星期 0 星期天 1 星期一 2 星期二 3 星期三 4 星期四 5 星期五 6 星期六			

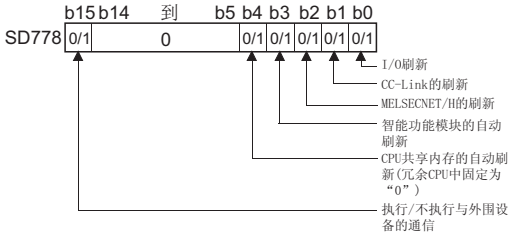
号码	名称	含义	解释	设置方式（当被设定时）	相应的ACPU D9□□□	相应的 CPU
SD676	备份数据还原信息	还原时间（年，月）	• 以 BCD 码的 2 位存储还原的年（后两位数）和月： 	S（初始化）	新增	QnU
SD677		还原时间（日期，小时）	• 以 BCD 码的 2 位存储还原的日期和小时： 			
SD678		还原时间（分钟，秒）	• 以 BCD 码的 2 位存储还原的分钟和秒： 			
SD679		还原时间（年的前位，星期）	• 以 BCD 码的 2 位存储还原的年（前两位）和星期。 			
SD681	程序内存写入（传输）状况	写入（传输）状况显示 (%)	• 以百分比显示至程序内存（闪存 ROM）的写入（传输）状况。在有 (0 ～ 100%) 写入指示的时点设置为 “0”。	S（写入时）	新增	QnU
SD682	程序内存写入次数指标	至目前为止的写入次数指标	• 以 32 位的 BIN 值存储至目前为止的程序内存（闪存 ROM）写入操作次数的指标值。 指标值超过了 10 万次时将发生 “FLASH ROM ERROR”（出错代码：1610）。（指标值超过 10 万次后依然计数。） 注）写入次数并不等于指标值。（通过系统执行闪存 ROM 的写入寿命延长功能，使得约 2 次写入操作只增加指标值 1。）	S（写入时）	新增	QnU
SD683						
SD686	标准 ROM 写入（传输）状况	写入（传输）状况显示 (%)	• 以百分比显示至标准 ROM（闪存 ROM）的写入（传输）状况。在有 (0 ～ 100%) 写入指示的时点设置为 “0”。	S（写入时）	新增	QnU
SD687	标准 ROM 写入次数指标	至目前为止的写入次数指标	• 以 32 位的 BIN 值存储至目前为止的标准 ROM（闪存 ROM）写入操作次数的指标值。 指标值超过了 10 万次时将发生 “FLASH ROM ERROR”（出错代码：1610）。（指标值超过 10 万次后依然计数。） 注）写入次数并不等于指标值。（通过系统执行闪存 ROM 的写入寿命延长功能，使得上一次 “计数到” 后的合计写入容量达到约 1M 字节后增加指标值 1。）	S（写入时）	新增	QnU
SD695	至标准 ROM 的写入指令执行次数指定	指定至标准 ROM 的写入指令执行次数	• 指定 1 日内的标准 ROM 写入指令（SP.DEVST）的最多执行次数。 • 标准 ROM 写入指令的执行次数超过了 SD 中设置的次数时，将发生 “OPERATION ERROR”（出错代码：4113）。 • SD695 的设置范围为 0 ～ 32767。设置了 0 或者超出范围的值时，执行标准 ROM 写入指令时将发生 “OPERATION ERROR”（出错代码：4113）。	U	新增	QnU

(6) 指令相关寄存器

附表 .24 特殊寄存器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD705 SD706	掩码模式	掩码模式	• 在块操作过程中，将 SM705 变为 ON 以便可以使用存储在 SD705（如果正在使用双字，则是 SD705 和 SD706）上的掩码模式，用掩码的值在块中所有的数据上进行操作。	U	新增	Q00J/Q00/Q01 Qn(H) QnPH QnPRH QnA
SD714	空通讯请求注册区域的号	0 到 32	• 存储用于连接到 MELSECNET/MINI-S3 的远程端子模块的通讯请求区域中空块的号。	S（在执行过程中）	D9081	QnA
SD715 SD716 SD717	IMASK 指令掩码模式	掩码模式	• 使用 IMASK 指令掩码的模式以下列方式存储： b15 到 b1 b0 SD715 I15 到 I1 I0 SD716 I31 到 I17 I16 SD717 I47 到 I33 I32	S（在执行过程中）	新增	○
SD718 SD719	累加器	累加器	• 用作在 A 系列程序中使用的累加器的替代。	S/U	新增	
SD720	PLOADP 指令的程序号指定	PLOADP 指令的程序号指定	当 PLOADP 指令被指定时，存储 PLOADP 指令装载的程序的程序号。 指定范围：1 到 124	U	新增	Qn(H) QnPH
SD730	CC-Link 通讯请求的空注册区域号	0 到 32	• 存储和连接到 A(1S)J61QBT61 的智能软元件站进行通讯的请求的空注册区域号。	S（在执行过程中）	新增	QnA
SD736	PKEY 输入	PKEY 输入	• 用 PKEY 指令暂时存储键盘数据输入的特殊寄存器。	S（在执行过程中）	新增	

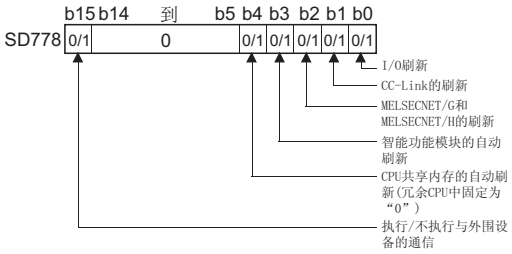
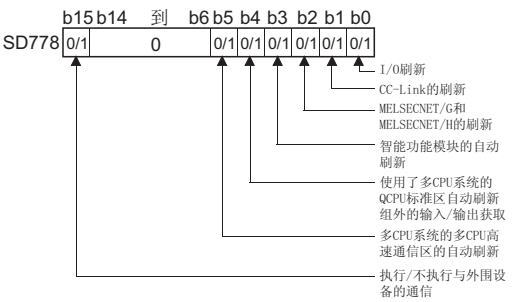
附表 .24 特殊寄存器列表

<table>
<tr><th>号码</th><th>名称</th><th>含义</th><th>解释</th><th>设置方式
（当被设定时）</th><th>相应的
ACPU
D9□□□</th><th>相应的 CPU</th></tr>
<tr><td>SD738
SD739
SD740
SD741
SD742
SD743
SD744
SD745
SD746
SD747
SD748
SD749
SD750
SD751
SD752
SD753
SD754
SD755
SD756
SD757
SD758
SD759
SD760
SD761
SD762
SD763
SD764
SD765
SD766
SD767
SD768
SD769</td><td>消息存储</td><td>消息存储</td><td>• 存储 MSG 指令指定的消息。
<table><tr><th></th><th>b15 到 b8</th><th>b7 到 b0</th></tr><tr><td>SD738</td><td>第2个字符</td><td>第1个字符</td></tr><tr><td>SD739</td><td>第4个字符</td><td>第3个字符</td></tr><tr><td>SD740</td><td>第6个字符</td><td>第5个字符</td></tr><tr><td>SD741</td><td>第8个字符</td><td>第7个字符</td></tr><tr><td>SD742</td><td>第10个字符</td><td>第9个字符</td></tr><tr><td>SD743</td><td>第12个字符</td><td>第11个字符</td></tr><tr><td>SD744</td><td>第14个字符</td><td>第13个字符</td></tr><tr><td>SD745</td><td>第16个字符</td><td>第15个字符</td></tr><tr><td>SD746</td><td>第18个字符</td><td>第17个字符</td></tr><tr><td>SD747</td><td>第20个字符</td><td>第19个字符</td></tr><tr><td>SD748</td><td>第22个字符</td><td>第21个字符</td></tr><tr><td>SD749</td><td>第24个字符</td><td>第23个字符</td></tr><tr><td>SD750</td><td>第26个字符</td><td>第25个字符</td></tr><tr><td>SD751</td><td>第28个字符</td><td>第27个字符</td></tr><tr><td>SD752</td><td>第30个字符</td><td>第29个字符</td></tr><tr><td>SD753</td><td>第32个字符</td><td>第31个字符</td></tr><tr><td>SD754</td><td>第34个字符</td><td>第33个字符</td></tr><tr><td>SD755</td><td>第36个字符</td><td>第35个字符</td></tr><tr><td>SD756</td><td>第38个字符</td><td>第37个字符</td></tr><tr><td>SD757</td><td>第40个字符</td><td>第39个字符</td></tr><tr><td>SD758</td><td>第42个字符</td><td>第41个字符</td></tr><tr><td>SD759</td><td>第44个字符</td><td>第43个字符</td></tr><tr><td>SD760</td><td>第46个字符</td><td>第45个字符</td></tr><tr><td>SD761</td><td>第48个字符</td><td>第47个字符</td></tr><tr><td>SD762</td><td>第50个字符</td><td>第49个字符</td></tr><tr><td>SD763</td><td>第52个字符</td><td>第51个字符</td></tr><tr><td>SD764</td><td>第54个字符</td><td>第53个字符</td></tr><tr><td>SD765</td><td>第56个字符</td><td>第55个字符</td></tr><tr><td>SD766</td><td>第58个字符</td><td>第57个字符</td></tr><tr><td>SD767</td><td>第60个字符</td><td>第59个字符</td></tr><tr><td>SD768</td><td>第62个字符</td><td>第61个字符</td></tr><tr><td>SD769</td><td>第64个字符</td><td>第63个字符</td></tr></table></td><td>S（在执行过程中）</td><td>新增</td><td>QnA</td></tr>
<tr><td>SD774
到
SD775</td><td>PID 限制设置
（用于完整微分）</td><td>0: 有限制
1: 无限制</td><td>• 按照如下所示，为每个 PID 环路指定限制:
<table><tr><th></th><th>b15</th><th></th><th>b1</th><th>b0</th></tr><tr><td>SD774</td><td>环路16</td><td>到</td><td>环路2</td><td>环路1</td></tr><tr><td>SD775</td><td>环路32</td><td>到</td><td>环路18</td><td>环路17</td></tr></table></td><td>U</td><td>新增</td><td>Qn(H)
QnPRH
QnU</td></tr>
<tr><td>SD774</td><td>PID 限制设置
（用于完整微分）</td><td>0: 有限制
1: 无限制</td><td>• 按照如下所示，为每个 PID 环路指定限制:
<table><tr><th></th><th>b15 到 b8</th><th>b7 到 b1</th><th>b0</th></tr><tr><td>SD774</td><td></td><td>环路8 到 环路2</td><td>环路1</td></tr></table></td><td>U</td><td>新增</td><td>Q00J/Q00/Q01*9</td></tr>
<tr><td>SD778</td><td>执行 COM 指令时的刷新处理选择</td><td>b0 ～ b14:
0: 不刷新
1: 刷新
b15 位
0: 执行与 CPU 模块的通讯
1: 不执行与 CPU 模块的通讯</td><td>• 选择执行 COM 指令时，是否刷新数据。
• 当 SM775 变为 ON 时，SD778 的指定有效。
 

• 在下述情况下，根据 COM 指令在多 CPU 之间执行刷新。
来自于其它号 CPU 的接收动作 :SD778 的 b4(CPU 共享内存的自动刷新) 为 1 时
来自于自站 CPU 的接收动作 :SD778 的 b15(执行 / 不执行与外围设备的通信) 为 0 时</td><td>U</td><td>新增</td><td>Q00J/Q00/Q01*9
Qn(H)*11
QnPH*12
QnPRH</td></tr>
</table>

*9 : 以功能版本 B 以后的 CPU 为对象。
*11: 以序列号的高 5 位为 “04012” 以后的 CPU 为对象
*12: 以序列号的高 5 位为 “07032” 以后的 CPU 为对象

附表 .24 特殊寄存器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD778	执行 COM 指令时的刷新处理选择	b0 ～ b14: 0: 不刷新 1: 刷新 b15 位 0: 执行与 CPU 模块的通讯 1: 不执行与 CPU 模块的通讯	• 选择执行 COM 指令时，是否刷新数据。 • 当 SM775 变为 ON 时，SD778 的指定有效。  • 在下述情况下，根据 COM 指令在多 CPU 之间执行刷新。 来自于其它号 CPU 的接收动作 :SD778 的 b4(CPU 共享内存的自动刷新）为 1 时 来自于自站 CPU 的接收动作 :SD778 的 b15（执行 / 不执行与外围设备的通信）为 0 时 • SD778 的 b2(MELSECNET/G 和 MELSECNET/H 的刷新）为 1 时，MELSECNET/G 和 MELSECNET/H 都将刷新。因此刷新点数较多时，COM 指令的处理时间将会延长。	U	新增	Qn(H)*13
			• 选择执行 COM 指令时，是否刷新数据。 • 当 SM775 变为 ON 时，SD778 的指定有效。 	U	新增	QnU
SD780	同时执行的 CC-Link 专用指令的剩余数目	0 到 32	• 存储同时执行的 CC-Link 专用指令的剩余数目。	U	新增	QnA
SD781 到 SD793	IMASK 指令的掩码模式	掩码模式	• 按照如下所示，存储 IMASK 指令掩码的掩码模式: b15 b1 b0 SD781 I63 到 I49 I48 SD782 I79 到 I65 I64 到 SD793 I255 到 I241 I240	S（在执行过程中）	新增	Qn(H) QnPH QnPRH QnU
SD781 到 SD785	IMASK 指令的掩码模式	掩码模式	• 如下所示，存储由 IMASK 指令掩码的掩码模式: b15 b1 b0 SD781 I63 到 I49 I48 SD782 I79 到 I65 I64 到 到 SD785 I127 到 I113 I112	S（在执行过程中）	新增	Q00J/Q00/Q01
SD794 到 SD795	PID 限制设置（用于不完整微分）	0: 限制设定 1: 限制未设定	• 按如下所示，指定每个 PID 环路的限制。 b15 b1 b0 SD794 环路16 到 环路2 环路1 SD795 环路32 到 环路18 环路17	U	新增	Qn(H)*13 QnPRH QnU
SD794	PID 限制设置（用于不精确微分）	0: 有限制 1: 无限制	• 按如下所示，指定每个 PID 环路的限制。 b15 到 b8 b7 b1 b0 SD794 环路8 到 环路2 环路1	U	新增	Q00J/Q00/Q01*9

*9 : 适用于功能版本为 B 或者更高的 CPU。
*13: 以序列号的高 5 位为 “09012” 以后的 CPU 为对象

号码	名称	含义	解释	设置方式（当被设定时）	相应的ACPU D9□□□	相应的 CPU
SD796	多 CPU 间高速总线对应专用指令最大使用块数设置（1 号 CPU 用）	专用指令最大使用块数范围 1～9（默认：2） * 设置了 1～9 以外的值时按 9 执行动作。	• 指定多 CPU 间高速总线对应专用指令（对象 CPU=1 号 CPU）的最多使用块数。对 1 号 CPU 执行多 CPU 间通信专用指令时，专用指令传输区的空余块数未达到本寄存器的设置值的情况下，使 SM796 为 ON。作为多 CPU 间通信专用指令的连续执行用互锁信号使用。	U(RUN 后 1 个扫描时）	新增	QnU*14
SD797	多 CPU 间高速总线对应专用指令最大使用块数设置（2 号 CPU 用）	专用指令最大使用块数范围 1～9（默认：2） * 设置了 1～9 以外的值时按 9 执行动作。	• 指定多 CPU 间高速总线对应专用指令（对象 CPU=2 号 CPU）的最多使用块数。对 2 号 CPU 执行多 CPU 间通信专用指令时，专用指令传输区的空余块数未达到本寄存器的设置值的情况下，使 SM797 为 ON。作为多 CPU 间通信专用指令的连续执行用互锁信号使用。	U(RUN 后 1 个扫描时）	新增	QnU*14
SD798	多 CPU 间高速总线对应专用指令最大使用块数设置（2 号 CPU 用）	专用指令最大使用块数范围 1～9（默认：2） * 设置了 1～9 以外的值时按 9 执行动作。	• 指定多 CPU 间高速总线对应专用指令（对象 CPU=3 号 CPU）的最多使用块数。对 3 号 CPU 执行多 CPU 间通信专用指令时，专用指令传输区的空余块数未达到本寄存器的设置值的情况下，使 SM798 为 ON。作为多 CPU 间通信专用指令的连续执行用互锁信号使用。	U(RUN 后 1 个扫描时）	新增	QnU*14
SD799	多 CPU 间高速总线对应专用指令最大使用块数设置（2 号 CPU 用）	专用指令最大使用块数范围 1～9（默认：2） * 设置了 1～9 以外的值时按 9 执行动作。	• 指定多 CPU 间高速总线对应专用指令（对象 CPU=4 号 CPU）的最多使用块数。对 4 号 CPU 执行多 CPU 间通信专用指令时，专用指令传输区的空余块数未达到本寄存器的设置值的情况下，使 SM799 为 ON。作为多 CPU 间通信专用指令的连续执行用互锁信号使用。	U(RUN 后 1 个扫描时）	新增	QnU*14

*14：除 Q02UCPU 外的通用型 QCPU。

(7) 调试

附表 .25 特殊寄存器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD806 SD807 SD808 SD809 SD810 SD811	状态锁存文件名称	状态锁存文件名称	• 以 ASCII 码存储从执行状态锁存的开始的文件名称（带扩展）。 b15 到 b8 b7 到 b0 SD806 第2个字符 第1个字符 SD807 第4个字符 第3个字符 SD808 第6个字符 第5个字符 SD809 第8个字符 第7个字符 SD810 扩展的第1个字符 2EH(.) SD811 扩展的第3个字符 扩展的第2个字符	S（在执行过程中）	新增	QnA
SD812 SD813 SD814 SD815 SD816	状态锁存步	状态锁存步	• 存储从执行状态锁存的点开始的步号。 SD812 模式* SD813 块号 SD814 步号/转移号 SD815 顺控步号(L) SD816 顺控步号(H) *: 模式数据的内容 15 14 到 4 3 2 1 0 ← (位号) 0 0 到 0 0 * * * (未使用) SFC块指定存在(1)/不存在(0) SFC步指定存在(1)/不存在(0) SFC移指定存在(1)/不存在(0)	S（在执行过程中）	D9055 格式改变	

(8) 锁存区域

附表 .26 特殊寄存器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD900	电源中断的驱动器	电源中断的驱动器	• 存储驱动器号，如果在电源掉电过程中有文件正被访问。	S（状态改变）	新增	
SD901 SD902 SD903 SD904 SD905 SD906	电源掉电过程中的活动文件名	电源掉电过程中的活动文件名	• 以 ASCII 码存储文件名（带扩展），如果在电源掉电过程中有文件正被访问。 b15 到 b8 b7 到 b0 SD901 第2个字符 第1个字符 SD902 第4个字符 第3个字符 SD903 第6个字符 第5个字符 SD904 第8个字符 第7个字符 SD905 扩展的第1个字符 2EH(.) SD906 扩展的第3个字符 扩展的第2个字符	S（状态改变）	新增	
SD910 SD911 SD912 SD913 SD914 SD915 SD916 SD917 SD918 SD919 SD920 SD921 SD922 SD923 SD924 SD925	RKEY 输入	RKEY 输入	• 按顺序存储输入的 PU 键码。 b15 到 b8 b7 到 b0 SD910 第2个字符 第1个字符 SD911 第4个字符 第3个字符 SD912 第6个字符 第5个字符 SD913 第8个字符 第7个字符 SD914 第10个字符 第9个字符 SD915 第12个字符 第11个字符 SD916 第14个字符 第13个字符 SD917 第16个字符 第15个字符 SD918 第18个字符 第17个字符 SD919 第20个字符 第19个字符 SD920 第22个字符 第21个字符 SD921 第24个字符 第23个字符 SD922 第26个字符 第25个字符 SD923 第28个字符 第27个字符 SD924 第30个字符 第29个字符 SD925 第32个字符 第31个字符	S（状态改变）	新增	QnA

(9) 冗余 CPU 信息（主站系统 CPU 信息*1）

附表 .27 特殊寄存器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD952	从控制系统到待机系统内存复制的历史记录	从控制系统到待机系统内存复制的最新状态	存储最后一次执行的从控制系统到待机系统的内存复制的结束状态。 1) 在从控制系统到待机系统的内存复制正常结束 / 异常结束时，存储和存储到 SD1596 中的值一样的值。 2) 为电源故障做备份，此特殊寄存器保持最后一次执行从控制系统到待机系统内存复制的状态。 3) 由锁存清除操作清除到 0。	S（状态改变）	新增	QnPRH

*1: 主站系统 CPU 信息被存储。

(10)A → Q/QnA 转换

在 A → Q/QnA 转换之后，ACPU 特殊寄存器 D9000 ～ D9255 对应于 Q/QnA 特殊寄存器 SD1000 ～ SD1255。

(但是，基本型 QCPU、冗余 CPU 和通用型 QCPU 不支持 A → Q/QnA 转换。)

这些特殊继电器都是由系统设定的，用户程序不能设定它们。

要由用户程序设定数据，修改程序以使用 Q/QnACPU 特殊寄存器。

但是，SD1200 到 SD1255(对应转换前的 D9200 到 9255) 中的一些寄存器不能由用户程序设定，如果在转换之前它们可以由用户程序设定的话。

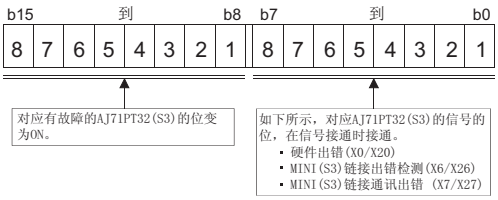
对于 ACPU 特殊寄存器的详细信息，请参考相应的 CPU 的用户手册，和 MELSECNET 或者 MELSECNET/B 数据链接系统参考手册。

备 注

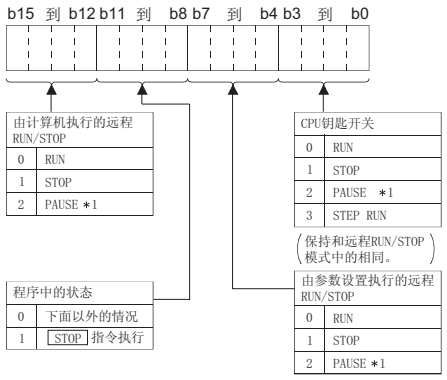
“用于改进的特殊寄存器”一栏的附加解释

1. 对于指定了用于改进的特殊寄存器的软元件号，将其修改为用于 QCPU/QnACPU 的特殊寄存器。
2. 对于指定了[-]的软元件号，可以使用转换之后的特殊寄存器。
3. 指定了[X]的软元件号不能用于 QCPU/QnACPU。

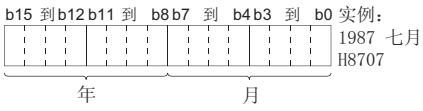
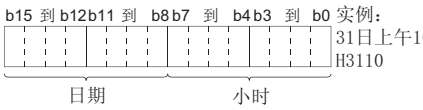
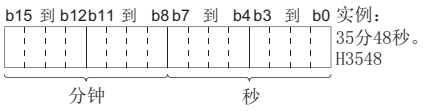
附表 .27 特殊寄存器列表

<table>
<tr><th>ACPU 特殊转换</th><th>转换后的特殊寄存器</th><th>用于改进的特殊寄存器</th><th>名称</th><th>含义</th><th>详细信息</th><th>相应的 CPU</th></tr>
<tr><td>D9000</td><td>SD1000</td><td>－</td><td>保险丝熔断</td><td>保险丝熔断的模块的号码</td><td>• 当保险丝熔断的模块被检测到时，检测到的模块的最小号码的第一个 I/O 号被存储到十六进制数中。
（实例：当 Y50 到 6F 输出模块的保险丝熔断时，“50”被存储到十六进制数中）
要使用外围设备监视此号码，以十六进制执行给定的监视操作。
（当 SD1100 到 SD1107 的所有内容都被复位到 0 时，清除。）
• 对远程 I/O 站的输出模块，也执行保险丝熔断检查。</td><td rowspan="2">QnA
Qn(H)
QnPH</td></tr>
<tr><td>D9001</td><td>SD1001</td><td>－</td><td>保险丝熔断</td><td>保险丝熔断模块的数目</td><td>• 存储当发生保险丝熔断时对应设置开关号的模块号或者基板槽号。
<table><tr><th colspan="2">用于A0J2的I/O模块</th><th colspan="2">扩展基板</th></tr><tr><th>设置开关</th><th>存储的数据</th><th>基板槽号</th><th>存储的数据</th></tr><tr><td>0</td><td>0</td><td>0</td><td>4</td></tr><tr><td>1</td><td>1</td><td>1</td><td>5</td></tr><tr><td>2</td><td>2</td><td>2</td><td>6</td></tr><tr><td>3</td><td>3</td><td>3</td><td>7</td></tr><tr><td>4</td><td>4</td><td colspan="2" rowspan="4"></td></tr><tr><td>5</td><td>5</td></tr><tr><td>6</td><td>6</td></tr><tr><td>7</td><td>7</td></tr></table>• 对于远程 I/O 站，（模块 I/O 号 /10H)+1 值被存储。</td></tr>
<tr><td>D9002</td><td>SD1002</td><td>－</td><td>I/O 模块验证出错</td><td>验证出错的 I/O 模块号</td><td>• 如果其数据不同于输入的数据的 I/O 模块，在电源变为 ON 时被检测到，在检测到的基板中，最低号码的基板的第一个 I/O 号被存储成十六进制数据。（存储方法和 SD1000 的相同。）
要用外围设备监视此号码，以十六进制执行给定的监视操作。
（清除，当 SD1116 到 SD1123 的所有内容都复位到 0 时。）
• 对远程 I/O 端子的模块也执行 I/O 模块验证检查。</td><td>QnA
Qn(H)
QnPH</td></tr>
<tr><td>D9004</td><td>SD1004</td><td>－</td><td>MINI 连接主站模块出错</td><td>在参数上设定的存储设置状态</td><td>• 在安装的 AJ71PT32(S3) 上检测到的 MINI(S3) 链接的出错状态被存储。
 
</td><td>QnA</td></tr>
<tr><td rowspan="3">D9005</td><td rowspan="3">SD1005</td><td rowspan="3">－</td><td rowspan="3">AC DOWN 计数器</td><td rowspan="3">AC DOWN 发生的次数</td><td>• 当使用 AC 电源模块时，在发生 20ms 以内的瞬间掉电时加 1。
（此值以 BIN 码存储。）当电源从 OFF 切换到 ON，它被复位。</td><td>QnA
Qn(H)
QnPH</td></tr>
<tr><td>• 当使用 DC 电源模块时，在发生 10ms 以内的瞬间掉电时加 1。
（此值以 BIN 码存储。）当电源从 OFF 切换到 ON，它被复位。</td><td>Qn(H)
QnPH</td></tr>
<tr><td>• 当使用 DC 电源模块时，在发生 1ms 以内的瞬间掉电时加 1。
（此值以 BIN 码存储。）当电源从 OFF 切换到 ON，它被复位。</td><td>QnA</td></tr>
<tr><td>D9008</td><td>SD1008</td><td>SD0</td><td>自检测出错</td><td>自检测出错号</td><td>• 当在自检测结果中发现出错时，出错号以 BIN 码存储。</td><td>QnA
Qn(H)
QnPH</td></tr>
</table>

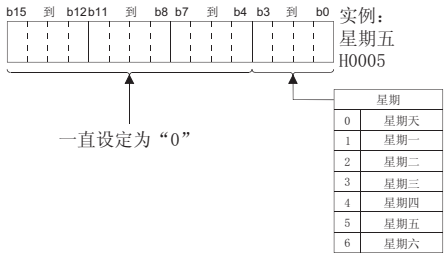
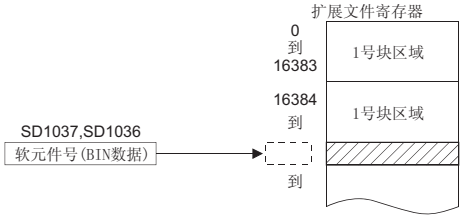
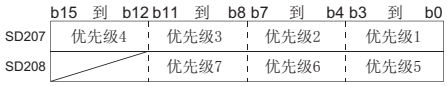
附表 . 27 特殊寄存器列表

ACPU 特殊转换	转换后的特殊寄存器	用于改进的特殊寄存器	名称	含义	详细信息	相应的 CPU
D9009	SD1009	SD62	报警器检测	发生外部故障的 F 号	• 当 F0 到 2047 中的一个被 OUT F 或者 SET F 变为 ON 时，在变为 ON 的 F 号中最早被检测到的 F 号，被以 BIN 码存储。 • SD62 可以用 RST F 或者 LEDR 指令清除。 如果还其检测到他 F 号，则清除 SD62 将使下一个号码被存储到 SD62。	Qn(H) QnPH Q2AS Q2A
					• 当 F0 到 2047 中的一个由 OUT F 或者 SET F 变为 ON 时，在变为 ON 的 F 号中最早被检测到的 F 号，被以 BIN 码存储 。 • SD62 可以通过执行 RST F 或者 LEDR 指令来清除，或者通过将 CPU 模块前面的 INDICATORRESET 开关移动到 ON 位置来清除。 如果还其检测到他 F 号，则清除 SD62 将使下一个号码被存储到 SD62。	Q3A Q4A Q4AR
D9010	SD1010	×	出错步	发生运行出错的步号。	• 当在执行应用指令过程中发生操作出错时，发生出错的步号以 BIN 码存储。 因此，每当发生操作出错时，SD1010 的内容被更新。	Qn(H) QnPH
D9011	SD1011	×	出错步	发生操作出错的步号	• 当在执行的应用指令过程中发生操作出错时，发生出错的步号以 BIN 码存储。由于步号是在 SM1011 从 OFF 变到 ON 时被存储到 SD1011 中的，SD1011 的数据并不被更新，除非用户程序清除了 SM1011。	
D9014	SD1014	×	I/O 控制模式	I/O 控制模式号	• I/O 控制模式设置以下列号中的一个返回： 0：输入和输出都处于直接模式 1：输入是刷新模式，输出是直接模式 3：输入和输出都处于刷新模式	
D9015	SD1015	SD203	CPU 的运行状态	CPU 的运行状态	• 如下所示的 CPU 的运行状态被存储到 SD203。  *1：当 CPU 模块处于 RUN 模式，并且 SM1040 是关断时，CPU 模块保持在 RUN 模式，如果它被改变到 PAUSE 模式的话。	QnA Qn(H) QnPH

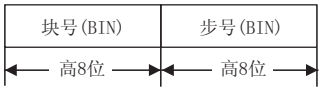
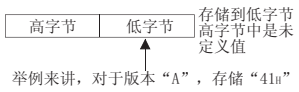
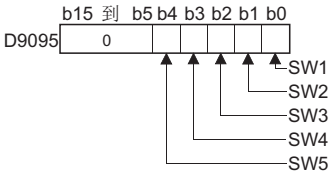
附表 .27 特殊继电器列表

ACPU 特殊转换	转换后的特殊寄存器	用于改进的特殊寄存器	名称	含义	详细信息	相应的 CPU
D9016	SD1016	×	程序号	0：主程序 (ROM) 1：主程序 (RAM) 2：子程序 1(RAM) 3：子程序 2(RAM) 4：子程序 3(RAM) 5：子程序 1(ROM) 6：子程序 2(ROM) 7：子程序 3(ROM) 8：主程序 (E^2PROM) 9：子程序 1(E^2PROM) A：子程序 2(E^2PROM) B：子程序 3(E^2PROM)	• 表示当前哪个顺控程序正在运行。0 到 B 之间的一个值被存储在 BIN 码。	QnA Qn(H) QnPH
D9017	SD1017	SD520	扫描周期时间	最小扫描周期时间 (10ms 为单位)	• 如果扫描周期时间小于 SD520 中的值，则在每个 END 新值被存储。 也就是说，扫描周期时间的最小值被以 BIN 码存储到 SD520。	
D9018	SD1018	SD524	扫描周期时间	扫描周期时间 (10ms 为单位)	• 在每个 END 中，扫描周期时间被以 BIN 码存储，并一直被重写。	
D9019	SD1019	SD526	扫描周期时间	最大扫描周期时间 (10ms 为单位)	• 如果扫描周期时间大于 SD526 中的值，则在每个 END 新值被存储。 也就是说，扫描周期时间的最大值被以 BIN 码存储到 SD526。	
D9020	SD1020	×	恒定扫描周期	恒定扫描周期时间 (由用户以 10ms 为单位进行设定)	• 设置连续程序启动间隔为 10ms 的倍数。 0 ：无设置 1 到 200 ：设置。程序以 (设定值) × 10 ms 的间隔执行。	
D9021	SD1021	–	扫描周期时间	扫描周期时间 (1ms 为单位)	• 在每个 END 中，扫描周期时间被以 BIN 码存储，并一直被重写。	
D9022	SD1022	SD412	1 秒计数器	以 1s 为单位计数。	• 当 PCCPU 开始运行时，它开始每秒计数 1。 • 从 0 开始计数，一直加到 32767，然后到 -32768，再重新加到 0。 计数一直重复此过程。	
D9025	SD1025	–	时钟数据	时钟数据 (年，月)	• 以 BCD 码存储年 (低 2 位数字) 和月。  实例： 1987 七月 H8707	
D9026	SD1026	–	时钟数据	时钟数据 (日，小时)	• 以 BCD 码存储日期和小时。  实例： 31日上午10点。 H3110	
D9027	SD1027	–	时钟数据	时钟数据 (分钟，秒)	• 以 BCD 码存储分钟和秒。  实例： 35分48秒。 H3548	

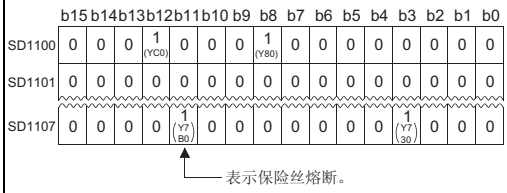
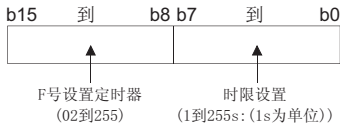
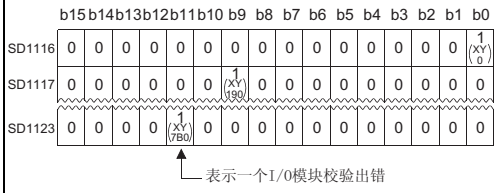
附表 . 27 特殊继电器列表

ACPU 特殊转换	转换后的特殊寄存器	转换后的特殊寄存器	名称	含义	详细信息	相应的 CPU
D9028	SD1028	–	时钟数据	时钟数据（星期）	• 以 BCD 码存储星期。 	QnA Qn(H) QnPH
D9035	SD1035	SD648	扩展文件寄存器	使用的块号	• 以 BCD 码存储正被使用的扩展文件寄存器的块号。	
D9036	SD1036	×	用于软元件号的指定的扩展文件寄存器	来自扩展文件寄存器的单个软元件被直接访问时的软元件号	• 以 BIN 数据，用 SD1036 和 SD1037 中的 2 个字，为用于直接读和写的扩展文件寄存器指定软元件号。 从 1 号块的 R0 开始，使用连续的号码去指定软元件号。 	
D9037	SD1037	×				
D9038	SD1038	SD207	LED 显示优先级	优先级 1 到 4	• 设置 ERRORLED 的优先级，此 LED 点亮（或者闪烁），以出错代码号表示出错。 • 优先级设置区域的配置如下所示。  • 详细信息，请参考可用 CPU 的用户手册和编程手册（基础篇）。	
D9039	SD1039	SD208		优先级 5 到 7		
D9044	SD1044	×	用于采样跟踪	采样跟踪过程中的步或者时间	• 用外围设备变为 ON/OFF。 当执行 STRA 或者 STRAR 时，存储在 SD1044 中的值被用作采样跟踪条件。 在扫描过程中 --------0 在定时间 ---------- 时间（10ms 为单位） 此值以 BIN 码存储到 SD1044 中。	
D9049	SD1049	×	用于 SFC 的工作区域	扩展文件寄存器的块号	• 以二进制值存储扩展文件寄存器的块号，此寄存器用作用于执行 SFC 程序的工作区域。 • 如果使用了 16K 字节或者更小的空置区域（不能是 1 号扩展文件寄存器），或者如果 SM320 是 OFF，则存储 “0”。	
D9050	SD1050	×	SFC 程序出错号	SFC 程序产生的出错代码	• 以 BIN 码存储 SFC 程序中发生的出错的出错代码。 0 : 没有出错 80: SFC 程序参数出错 81: SFC 代码出错 82: 超过了同时执行的步数 83: 块启动出错 84: SFC 程序运行出错	
D9051	SD1051	×	出错块	出错发生的块号	• 以 BIN 码存储 SFC 程序中发生出错的块的块号。 在出错 83 中，起始块号被存储。	
D9052	SD1052	×	出错步	出错发生的步号	• 以 BIN 值存储 SFC 程序中发生出错代码 84 的步号。 • 当出错代码 80、81 或者 82 发生时，存储 “0”。 • 当出错代码 83 发生时，存储块起始步号。	

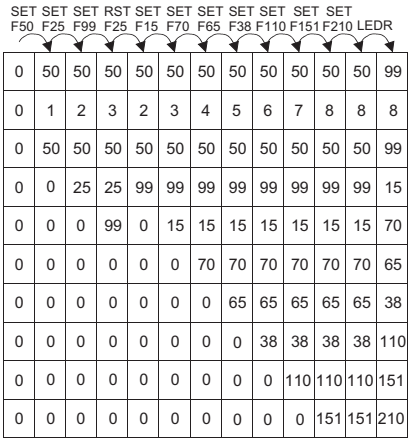
附表 .27 特殊寄存器列表

ACPU 特殊转换	转换后的特殊寄存器	用于改进的特殊寄存器	名称	含义	详细信息	相应的 CPU
D9053	SD1053	×	出错转移	出错发生处的转移条件号	• 以 BIN 值存储 SFC 程序中发生出错代码 84 的地方的转移条件号。 当出错代码 80、81、82 或者 83 发生，存储 “0”。	QnA Qn(H) QnPH
D9054	SD1054	×	出错顺控程序	出错发生处的顺控程序步号	• 以 BIN 码存储 SFC 程序中发生出错 84 的发送条件和操作输出的顺控程序步号。	
D9055	SD1055	SD812	状态锁存执行步号	状态锁存步	• 当执行状态锁存时，存储步号。 • 如果在主顺控程序中执行状态锁存，以二进制值存储步号。 • 如果在 SFC 程序执行状态锁存，存储块号和步号。 	
D9060	SD1060	SD392	软件版本	内部软件的软件版本	• 以 ASCII 码存储内部系统的软件版本。  举例来讲，对于版本 “A”，存储 “41H” 注释 :初始化系统的软件版本可能和印刷在外壳上的版本信息指示的版本不同。	QnA
D9072	SD1072	×	可编程控制器通讯检查	串行通讯模块数据检查	• 在串行通讯模块的自环路回送测试中，串行通讯模块自动写 / 读数据以进行通讯检查。	QnA Qn(H) QnPH
D9081	SD1081	SD714	通讯请求注册区域中的空余块数	通讯请求注册区域中的空余块数	• 存储到连接到 MELSECNET/MINI-S3 主基板、A2CCPU 或者 A52GCPU 的远程端子模块的通讯请求注册区域中的空余块数。	QnA
D9085	SD1085	×	用于设置时间检查值的寄存器	1 s 到 65535 s	• 为 MELSECNET/10 设置数据链接指令 (ZNRD，ZNWR) 的时间检查时间。 • 设置范围 : 1s 到 65535s(1 到 65535) • 设置单位 : 1 s • 默认值 : 10s(如果设置了 0，则应用默认值 10s)	QnA Qn(H) QnPH
D9090	SD1090	×	特殊功能模块数	特殊功能模块数	• 对于详细信息，请参考每个微型计算机程序软件包的手册。	
D9091	SD1091	×	详细出错代码	自检测详细出错代码	• 存储指令出错原因的详细代码。	
D9094	SD1094	SD251	要更换的 I/O 模块的起始 I/O 号	要更换的 I/O 模块的起始 I/O 号	• 以 BIN 值存储将运行 (电源接通) 中卸载 / 安装的 I/O 模块起始 I/O 号码的头两位数。 实例) 输入模块 X2F0 → H2F	
D9095	SD1095	SD200	DIP 开关信息	DIP 开关信息	• 以下列格式存储 CPU 模块的 DIP 开关信息。 0: OFF 1: ON 	Qn(H) QnPH

附表 . 27 特殊寄存器列表

ACPU 特殊转换	转换后的特殊寄存器	用于改进的特殊寄存器	名称	含义	详细信息	相应的 CPU
D9100	SD1100	–	保险丝熔断的模块	以 16 点为单位的位模式，表示保险丝熔断的模块	• 保险丝熔断的输出模块号（以 16 点为单位）以位模式输入。（当执行参数设置时，预设输出模块号。）  • 对远程 I/O 站的输出模块也执行保险丝熔断检查。 （如果正常状态被存储，则不执行清除。因此，需要由用户程序执行清除。）	
D9101	SD1101					
D9102	SD1102					
D9103	SD1103					
D9104	SD1104					
D9105	SD1105					
D9106	SD1106					
D9107	SD1107					
D9108	SD1108	–	步发送看门狗定时器设置	定时器设置值和时间用完时的报警器号	• 设定步转移看门狗定时器的设置值和当看门狗定时器时间用完时将接通的报警器号 (F 号)。  • (通过将 SM1108 到 SM1114 中的任何一个接通，看门狗定时器启动。如果对应定时器的步后面的发送条件在设定的时间内不能建立，则设置的报警器 (F) 将接通。)	QnA Qn(H) QnPH
D9109	SD1109					
D9110	SD1110					
D9111	SD1111					
D9112	SD1112					
D9113	SD1113					
D9114	SD1114					
D9116	SD1116	–	I/O 模块验证出错	以 16 点为单位的位模式，表示验证出错的模块。	• 当其数据和电源接通时输入的数据不同的 I/O 模块被检测到时，I/O 模块号（以 16 点为单位）被以位模式输入。（当执行参数设置时，在参数中预设 I/O 模块号。）  • I/O 模块验证检查也可以对远程 I/O 站模块执行。 （如果正常状态被存储，则不执行清除。因此，需要由用户程序执行清除。）	
D9117	SD1117					
D9118	SD1118					
D9119	SD1119					
D9120	SD1120					
D9121	SD1121					
D9122	SD1122					
D9123	SD1123					
D9124	SD1124	SD63	报警器检测数量	报警器检测数量	• 当 F0 到 255（对于 AuA 和 AnU 是 F0 到 2047）中的一个由 SET F 变为 ON 时，1 加到 SD63 中的值上。当 RST F 或者 LEDR 指令执行时，从 SD63 中的值减 1。（如果 CPU 模块有 INDICATORRESET 开关，则按下此开关可以执行相同的处理。） • 由 SET F 变为 ON 的数量被以 BIN 码存储到 SD63 中。SD63 的值最大是 8。	

附表 .27 特殊寄存器列表

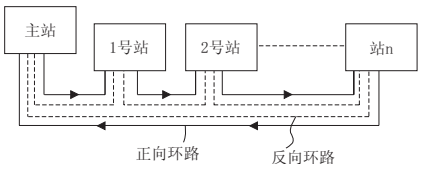
<table>
<tr><th>ACPU 特殊转换</th><th>转换后的特殊寄存器</th><th>用于改进的特殊寄存器</th><th>名称</th><th>含义</th><th>详细信息</th><th>相应的 CPU</th></tr>
<tr><td>D9125</td><td>SD1125</td><td>SD64</td><td rowspan="9">报警器检测号</td><td rowspan="9">报警器检测号</td><td rowspan="9">• 当 F0 到 2047 中的一个被 SET F 变为 ON 时，按顺序变为 ON 的报警器号（F 号）被注册到 D9125 到 D9132 中。
• 由 RST F 关断的 F 号被从 D9125 到 D9132 中的某一个中擦除，而存储在被擦除的 F 号之后的 F 号被移位到先前的寄存器中。
通过执行 LEDR 指令，SD64 到 SD71 的内容向前移动一位。（对于 A3N 和 A3HCPU，可以通过使用 CPU 模块前面的 INDICATORRESET 开关执行此操作。）
当检测到 8 个报警器时，即使检测到了第 9 个，也不会被存储到 SD64 到 SD71 中。
 
</td><td rowspan="9">QnA
Qn(H)
QnPH</td></tr>
<tr><td>D9126</td><td>SD1126</td><td>SD65</td></tr>
<tr><td>D9127</td><td>SD1127</td><td>SD66</td></tr>
<tr><td>D9128</td><td>SD1128</td><td>SD67</td></tr>
<tr><td>D9129</td><td>SD1129</td><td>SD68</td></tr>
<tr><td>D9130</td><td>SD1130</td><td>SD69</td></tr>
<tr><td>D9131</td><td>SD1131</td><td>SD70</td></tr>
<tr><td>D9132</td><td>SD1132</td><td>SD71</td></tr>
</table>

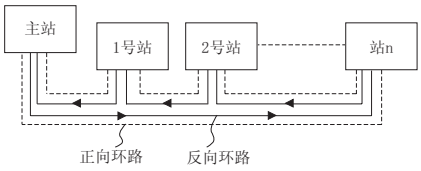
(11)QnA 专用特殊寄存器列表

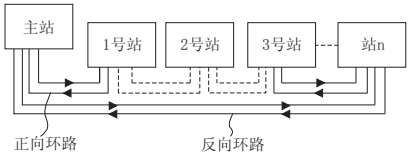
附表 .28 特殊寄存器列表

ACPU 特殊转换	转换后的特殊寄存器	用于改进的特殊寄存器	名称	含义	详细信息	相应的 CPU
D9200	SD1200	-	ZNRD(LRDP 用于 ACPU) 处理结果	0 : 正常结束 2 : ZNRD 指令设置故障 3 : 相关站上的出错 4 : 相关站 ZNRD 执行禁止	存储 ZNRD(字软元件读) 指令的执行结果 • ZNRD 指令设置故障 : ..... ZNRD 指令常量，源和 / 或者目标的故障设置。 • 相应站出错 : ..... 其中的一个站没有通讯。 • ZNRD 不能在相应站中执行 : 指定的站是远程 I/O 站。	QnA
D9201	SD1201	-	ZNWR(LWTP 用于 ACPU) 处理结果	0 : 正常结束 2 : ZNWR 指令设置故障 3 : 相关站上的出错 4 : 相关站 ZNWR 执行禁止	存储 ZNWR(字软元件读) 指令的执行结果 • ZNWR 指令设置故障 : ..... ZNWR 指令常量，源和 / 或者目标的故障设置。 • 相应站出错 : ..... 其中的一个站设有通讯。 • ZNWR 不能在相应站中执行 : .... 指定的站是远程 I/O 站。	
D9202	SD1202	-	本地站链接类型	存储 1 号到 16 号的条件	存储从站是否对应 MELSECNET 或者 MELSECNETII。 • 对应 MELSECNETII 站的位变为 “1”。 • 对应 MELSECNET 站或者没有连接的位变为 “0”。 • 如果本地站在运行过程中停止，则停止之前的内容被保持。 SD1224 到 SD1227 和 SD1228 到 SD1231 的内容被执行逻辑或操作。 如果相应位是 “0”，则上面的特殊寄存器的相应位变为有效。 • 如果自身 (主) 站停止，则停止之前的内容也会被保持。	
D9203	SD1203	-		存储 17 号到 32 号的条件		
D9204	SD1204	-	链接状态	0 : 正向环路，在数据链接中 1 : 反向环路，在数据链接中 2 : 正向 / 反向实现环路回送 3 : 只在正向实现环路回送 4 : 只在反向实现环路回送 5 : 数据链接禁止	存储数据链接的当前路径状态。 • 正向环路中的数据链接 • 反向环路中的数据链接 • 正向 / 反向实现环路回送	

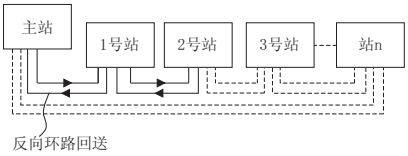
软元件号	位															
	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
SD1202	L16	L15	L14	L13	L12	L11	L10	L9	L8	L7	L6	L5	L4	L3	L2	L1
SD1203	L32	L31	L30	L29	L28	L27	L26	L25	L24	L23	L22	L21	L20	L19	L18	L17
SD1241	L48	L47	L46	L45	L44	L43	L42	L41	L40	L39	L38	L37	L36	L35	L34	L33
SD1242	L64	L63	L62	L61	L60	L59	L58	L57	L56	L55	L54	L53	L52	L51	L50	L49

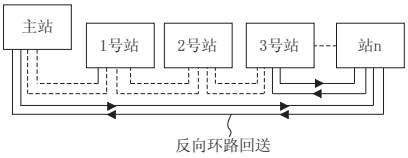
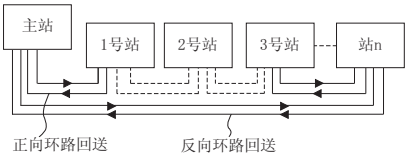






附表 .28 特殊寄存器列表

<table>
<tr><th>ACPU 特殊转换</th><th>ACPU 特殊转换</th><th>用于改进的特殊寄存器</th><th>名称</th><th>含义</th><th>详细信息</th><th>相应的 CPU</th></tr>
<tr><td>D9204</td><td>SD1204</td><td>–</td><td>链接状态</td><td>0 ： 正向环路， 在数据链接中
1 ： 反向环路， 在数据链接中
2 ： 正向 / 反向实现环路回送
3 ： 只在正向实现环路回送
4 ： 只在反向实现环路回送
5 ： 数据链接禁止</td><td>• 只在正向环路实现环路回送
 

• 只在反向环路实现环路回送
 
</td><td rowspan="11">QnA</td></tr>
<tr><td>D9205</td><td>SD1205</td><td>–</td><td>实现环路回送的站</td><td>实现正向环路回送的站</td><td rowspan="2">存储正在执行环路回送的本地或者远程 I/O 站号
 

在上面的实例中，1 被存储到 D9205，3 到 D9206。
如果数据链接恢复为正常状态（正向环路中的数据链接），D9205 和 D9206 中的值保持为 1 和 3。因此要将它们变为 “0”，使用顺控程序或者执行复位操作。</td></tr>
<tr><td>D9206</td><td>SD1206</td><td>–</td><td>实现环路回送的站</td><td>实现反向环路回送的站</td></tr>
<tr><td>D9210</td><td>SD1210</td><td>–</td><td>重试次数</td><td>存储 1 号到 16 号的条件</td><td>存储由于传送出错而引起的重试次数。
在最大值 “FFFFH” 处，计数停止。
要恢复到值 “0”，执行复位操作。</td></tr>
<tr><td>D9211</td><td>SD1211</td><td>–</td><td>环路选择次数</td><td>存储 1 号到 16 号的条件</td><td>存储环路电缆被切换到反向环路或者环路回送的次数。
在最大值 “FFFFH” 处，计数停止。
要恢复到值 “0”，执行复位操作。</td></tr>
<tr><td>D9212</td><td>SD1212</td><td>–</td><td>本地站运行状态</td><td>存储 1 号到 16 号的条件</td><td rowspan="4">存储处于 STOP 或者 PAUSE 模式的本地站号。
<table><tr><th rowspan="2">软元件号</th><th colspan="16">位</th></tr><tr><th>b15</th><th>b14</th><th>b13</th><th>b12</th><th>b11</th><th>b10</th><th>b9</th><th>b8</th><th>b7</th><th>b6</th><th>b5</th><th>b4</th><th>b3</th><th>b2</th><th>b1</th><th>b0</th></tr><tr><td>SD1212</td><td>L16</td><td>L15</td><td>L14</td><td>L13</td><td>L12</td><td>L11</td><td>L10</td><td>L9</td><td>L8</td><td>L7</td><td>L6</td><td>L5</td><td>L4</td><td>L3</td><td>L2</td><td>L1</td></tr><tr><td>SD1213</td><td>L32</td><td>L31</td><td>L30</td><td>L29</td><td>L28</td><td>L27</td><td>L26</td><td>L25</td><td>L24</td><td>L23</td><td>L22</td><td>L21</td><td>L20</td><td>L19</td><td>L18</td><td>L17</td></tr><tr><td>SD1214</td><td>L48</td><td>L47</td><td>L46</td><td>L45</td><td>L44</td><td>L43</td><td>L42</td><td>L41</td><td>L40</td><td>L39</td><td>L38</td><td>L37</td><td>L36</td><td>L35</td><td>L34</td><td>L33</td></tr><tr><td>SD1215</td><td>L64</td><td>L63</td><td>L62</td><td>L61</td><td>L60</td><td>L59</td><td>L58</td><td>L57</td><td>L56</td><td>L55</td><td>L54</td><td>L53</td><td>L52</td><td>L51</td><td>L50</td><td>L49</td></tr></table>
当本地站被切换到 STOP 或者 PAUSE 模式时，寄存器中对应此站号的位变为 “1”。
实例：当站 7 切换到 STOP 模式，SD1212 中的 b6 变为 “1”，当 SD1212 被监视时，它的值是 “(40H)”。</td></tr>
<tr><td>D9213</td><td>SD1213</td><td>–</td><td>本地站运行状态</td><td>存储到 17 号 到 32 号的条件</td></tr>
<tr><td>D9214</td><td>SD1214</td><td>–</td><td>本地站运行状态</td><td>存储 33 号到 48 号的条件</td></tr>
<tr><td>D9215</td><td>SD1215</td><td>–</td><td>本地站运行状态</td><td>存储 49 号到 64 号的条件</td></tr>
<tr><td>D9216</td><td>SD1216</td><td>–</td><td>本地站出错检测状态</td><td>存储 1 号到 16 号的条件</td><td rowspan="4">存储出错的本地站号。
<table><tr><th rowspan="2">软元件号</th><th colspan="16">位</th></tr><tr><th>b15</th><th>b14</th><th>b13</th><th>b12</th><th>b11</th><th>b10</th><th>b9</th><th>b8</th><th>b7</th><th>b6</th><th>b5</th><th>b4</th><th>b3</th><th>b2</th><th>b1</th><th>b0</th></tr><tr><td>SD1216</td><td>L16</td><td>L15</td><td>L14</td><td>L13</td><td>L12</td><td>L11</td><td>L10</td><td>L9</td><td>L8</td><td>L7</td><td>L6</td><td>L5</td><td>L4</td><td>L3</td><td>L2</td><td>L1</td></tr><tr><td>SD1217</td><td>L32</td><td>L31</td><td>L30</td><td>L29</td><td>L28</td><td>L27</td><td>L26</td><td>L25</td><td>L24</td><td>L23</td><td>L22</td><td>L21</td><td>L20</td><td>L19</td><td>L18</td><td>L17</td></tr><tr><td>SD1218</td><td>L48</td><td>L47</td><td>L46</td><td>L45</td><td>L44</td><td>L43</td><td>L42</td><td>L41</td><td>L40</td><td>L39</td><td>L38</td><td>L37</td><td>L36</td><td>L35</td><td>L34</td><td>L33</td></tr><tr><td>SD1219</td><td>L64</td><td>L63</td><td>L62</td><td>L61</td><td>L60</td><td>L59</td><td>L58</td><td>L57</td><td>L56</td><td>L55</td><td>L54</td><td>L53</td><td>L52</td><td>L51</td><td>L50</td><td>L49</td></tr></table>
如果本地站检测到出错，对应站号的位变为 “1”。
实例：当站 6 和 12 检测到出错，SD1216 中的 b5 和 b11 变为 “1”，当 SD1216 被监视时，它的值是 “2080(820H)”。</td></tr>
<tr><td>D9217</td><td>SD1217</td><td>–</td><td>本地站出错检测状态</td><td>存储 17 号到 32 号的条件</td></tr>
<tr><td>D9218</td><td>SD1218</td><td>–</td><td>本地站出错检测状态</td><td>存储 33 号到 48 号的条件</td></tr>
<tr><td>D9219</td><td>SD1219</td><td>–</td><td>本地站出错检测状态</td><td>存储 49 号到 64 号的条件</td></tr>
</table>

附表 .28 特殊寄存器列表

ACPU 特殊转换	转换后的特殊寄存器	用于改进的特殊寄存器	名称	含义	详细信息	相应的 CPU
D9220	SD1220	-	本地站参数不一致：远程 I/0 站 I/0 分配出错	存储 1 号到 16 号的条件	存储包含不匹配的参数的本地站号或者设置了不正确 I/0 分配的远程站号。 SD1220: L16 L15 L14 L13 L12 L11 L10 L9 L8 L7 L6 L5 L4 L3 L2 L1 SD1221: L32 L31 L30 L29 L28 L27 L26 L25 L24 L23 L22 L21 L20 L19 L18 L17 SD1222: L48 L47 L46 L45 L44 L43 L42 L41 L40 L39 L38 L37 L36 L35 L34 L33 SD1223: L64 L63 L62 L61 L60 L59 L58 L57 L56 L55 L54 L53 L52 L51 L50 L49 如果作为三级系统的主站的本地站检测参数出错或者远程 I/0 站 I/0 分配异常，对应本地站或者远程 I/0 站的站号的软元件号的位变为“1”。 实例：当本地站 5 和远程 I/0 站 14 检测到出错，SD1220 中的 b4 和 b13 变为“1”，当 SD1220 被监视时，它的值是“8208(2010H)”。	QnA
D9221	SD1221	-	本地站参数不一致：远程 I/0 站 I/0 分配出错	存储 17 号到 32 号的条件		
D9222	SD1222	-	本地站参数不一致：远程 I/0 站 I/0 分配出错	存储 33 号到 48 号的条件		
D9223	SD1223	-	本地站参数不一致：远程 I/0 站 I/0 分配出错	存储 49 号到 64 号的条件		
D9224	SD1224	-	本地站和远程 I/0 站初始化通讯进行中	存储 1 号到 16 号的条件	存储正在和它们的相关主站通讯初始化数据的本地或者远程站的站号。 SD1224: L/R16 L/R15 L/R14 L/R13 L/R12 L/R11 L/R10 L/R9 L/R8 L/R7 L/R6 L/R5 L/R4 L/R3 L/R2 L/R1 SD1225: L/R32 L/R31 L/R30 L/R29 L/R28 L/R27 L/R26 L/R25 L/R24 L/R23 L/R22 L/R21 L/R20 L/R19 L/R18 L/R17 SD1226: L/R48 L/R47 L/R46 L/R45 L/R44 L/R43 L/R42 L/R41 L/R40 L/R39 L/R38 L/R37 L/R36 L/R35 L/R34 L/R33 SD1227: L/R64 L/R63 L/R62 L/R61 L/R60 L/R59 L/R58 L/R57 L/R56 L/R55 L/R54 L/R53 L/R52 L/R51 L/R50 L/R49 和当前正在通讯初始化设置的站号对应的位变为“1”。 实例：当站 23 和 45 正在通讯时，SD1225 的 b6 和 SD1226 的 b12 变为“1”，当 SD1225 被监视时，它的值是“64(40H)”，当 SD1226 被监视时，它的值是“4096(1000H)”。	
D9225	SD1225	-	本地站和远程 I/0 站初始化通讯进行中	存储 17 号到 32 号的条件		
D9226	SD1226	-	本地站和远程 I/0 站初始化通讯进行中	存储 33 号到 48 号的条件		
D9227	SD1227	-	本地站和远程 I/0 站初始化通讯进行中	存储 49 号到 64 号的条件		
D9228	SD1228	-	本地站和远程 I/0 站出错	存储 1 号到 16 号的条件	存储出错的本地或者远程站号。 SD1228: L/R16 L/R15 L/R14 L/R13 L/R12 L/R11 L/R10 L/R9 L/R8 L/R7 L/R6 L/R5 L/R4 L/R3 L/R2 L/R1 SD1229: L/R32 L/R31 L/R30 L/R29 L/R28 L/R27 L/R26 L/R25 L/R24 L/R23 L/R22 L/R21 L/R20 L/R19 L/R18 L/R17 SD1230: L/R48 L/R47 L/R46 L/R45 L/R44 L/R43 L/R42 L/R41 L/R40 L/R39 L/R38 L/R37 L/R36 L/R35 L/R34 L/R33 SD1231: L/R64 L/R63 L/R62 L/R61 L/R60 L/R59 L/R58 L/R57 L/R56 L/R55 L/R54 L/R53 L/R52 L/R51 L/R50 L/R49 对应出错站号的位变为“1”。 实例：当本地站 3 和远程 I/0 站 14 出错时，SD1228 的 b2 和 b13 变为“1”，当 SD1228 被监视时，它的值是“8196(2004H)”。	
D9229	SD1229	-	本地站和远程 I/0 站出错	存储 17 号到 32 号的条件		
D9230	SD1230	-	本地站和远程 I/0 站出错	存储 33 号到 48 号的条件		
D9231	SD1231	-	本地站和远程 I/0 站出错	存储 49 号到 64 号的条件		

附表 .28 特殊寄存器列表

<table>
<tr><th>ACPU 特殊转换</th><th>转换后的特殊寄存器</th><th>用于改进的特殊寄存器</th><th>名称</th><th>含义</th><th>详细信息</th><th>相应的 CPU</th></tr>
<tr><td>D9232</td><td>SD1232</td><td>–</td><td>本地站和远程 I/0 站环路出错</td><td>存储 1 号到 8 号的条件</td><td rowspan="8">存储发生正向或者反向环路出错的本地或者远程站号
<table><tr><th rowspan="2">软元件号</th><th colspan="16">位</th></tr><tr><th>b15</th><th>b14</th><th>b13</th><th>b12</th><th>b11</th><th>b10</th><th>b9</th><th>b8</th><th>b7</th><th>b6</th><th>b5</th><th>b4</th><th>b3</th><th>b2</th><th>b1</th><th>b0</th></tr><tr><td>SD1232</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td></tr><tr><td></td><td colspan="2">L/R8</td><td colspan="2">L/R7</td><td colspan="2">L/R6</td><td colspan="2">L/R5</td><td colspan="2">L/R4</td><td colspan="2">L/R3</td><td colspan="2">L/R2</td><td colspan="2">L/R1</td></tr><tr><td>SD1233</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td></tr><tr><td></td><td colspan="2">L/R16</td><td colspan="2">L/R15</td><td colspan="2">L/R14</td><td colspan="2">L/R13</td><td colspan="2">L/R12</td><td colspan="2">L/R11</td><td colspan="2">L/R10</td><td colspan="2">L/R9</td></tr><tr><td>SD1234</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td></tr><tr><td></td><td colspan="2">L/R24</td><td colspan="2">L/R23</td><td colspan="2">L/R22</td><td colspan="2">L/R21</td><td colspan="2">L/R20</td><td colspan="2">L/R19</td><td colspan="2">L/R18</td><td colspan="2">L/R17</td></tr><tr><td>SD1235</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td></tr><tr><td></td><td colspan="2">L/R32</td><td colspan="2">L/R31</td><td colspan="2">L/R30</td><td colspan="2">L/R29</td><td colspan="2">L/R28</td><td colspan="2">L/R27</td><td colspan="2">L/R26</td><td colspan="2">L/R25</td></tr><tr><td>SD1236</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td></tr><tr><td></td><td colspan="2">L/R40</td><td colspan="2">L/R39</td><td colspan="2">L/R38</td><td colspan="2">L/R37</td><td colspan="2">L/R36</td><td colspan="2">L/R35</td><td colspan="2">L/R34</td><td colspan="2">L/R33</td></tr><tr><td>SD1237</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td></tr><tr><td></td><td colspan="2">L/R48</td><td colspan="2">L/R47</td><td colspan="2">L/R46</td><td colspan="2">L/R45</td><td colspan="2">L/R44</td><td colspan="2">L/R43</td><td colspan="2">L/R42</td><td colspan="2">L/R41</td></tr><tr><td>SD1238</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td></tr><tr><td></td><td colspan="2">L/R56</td><td colspan="2">L/R55</td><td colspan="2">L/R54</td><td colspan="2">L/R53</td><td colspan="2">L/R52</td><td colspan="2">L/R51</td><td colspan="2">L/R50</td><td colspan="2">L/R49</td></tr><tr><td>SD1239</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td></tr><tr><td></td><td colspan="2">L/R64</td><td colspan="2">L/R63</td><td colspan="2">L/R62</td><td colspan="2">L/R61</td><td colspan="2">L/R60</td><td colspan="2">L/R59</td><td colspan="2">L/R58</td><td colspan="2">L/R57</td></tr></table></td><td rowspan="8"></td></tr>
<tr><td>D9233</td><td>SD1233</td><td>–</td><td>本地站和远程 I/0 站环路出错</td><td>存储 9 号到 16 号的条件</td></tr>
<tr><td>D9234</td><td>SD1234</td><td>–</td><td>本地站和远程 I/0 站环路出错</td><td>存储 17 号到 24 号的条件</td></tr>
<tr><td>D9235</td><td>SD1235</td><td>–</td><td>本地站和远程 I/0 站环路出错</td><td>存储 25 号到 32 号的条件</td></tr>
<tr><td>D9236</td><td>SD1236</td><td>–</td><td>存储 25 号到 32 号的条件</td><td>存储 25 号到 32 号的条件</td></tr>
<tr><td>D9237</td><td>SD1237</td><td>–</td><td>本地站和远程 I/0 站环路出错</td><td>存储 41 号到 48 号的条件</td></tr>
<tr><td>D9238</td><td>SD1238</td><td>–</td><td>本地站和远程 I/0 站环路出错</td><td>存储 49 号到 56 号的条件</td></tr>
<tr><td>D9239</td><td>SD1239</td><td>–</td><td>本地站和远程 I/0 站环路出错</td><td>存储 57 号到 64 号的条件</td></tr>
<tr><td>D9240</td><td>SD1240</td><td>–</td><td>通讯出错检测到的次数</td><td>存储接收到的出错的累积值</td><td>存储下列传送出错检测到的次数：
CRC，OVER，AB. IF
计数最大到 FFFFH.
要恢复值到 “0”，执行复位操作。</td><td rowspan="3">QnA</td></tr>
<tr><td>D9241</td><td>SD1241</td><td rowspan="2">–</td><td rowspan="2">本地站链接类型</td><td>存储 33 号到 48 号的条件</td><td>存储从站是否对应 MELSECNET 或者 MELSECNET II。
• 对应 MELSECNETII 站的位变为 “1。”
• 对应 MELSECNET 站或者未连接的位变为 “0。”
<table><tr><th rowspan="2">软元件号</th><th colspan="16">位</th></tr><tr><th>b15</th><th>b14</th><th>b13</th><th>b12</th><th>b11</th><th>b10</th><th>b9</th><th>b8</th><th>b7</th><th>b6</th><th>b5</th><th>b4</th><th>b3</th><th>b2</th><th>b1</th><th>b0</th></tr><tr><td>SD1202</td><td>L16</td><td>L15</td><td>L14</td><td>L13</td><td>L12</td><td>L11</td><td>L10</td><td>L9</td><td>L8</td><td>L7</td><td>L6</td><td>L5</td><td>L4</td><td>L3</td><td>L2</td><td>L1</td></tr><tr><td>SD1203</td><td>L32</td><td>L31</td><td>L30</td><td>L29</td><td>L28</td><td>L27</td><td>L26</td><td>L25</td><td>L24</td><td>L23</td><td>L22</td><td>L21</td><td>L20</td><td>L19</td><td>L18</td><td>L17</td></tr><tr><td>SD1241</td><td>L48</td><td>L47</td><td>L46</td><td>L45</td><td>L44</td><td>L43</td><td>L42</td><td>L41</td><td>L40</td><td>L39</td><td>L38</td><td>L37</td><td>L36</td><td>L35</td><td>L34</td><td>L33</td></tr><tr><td>SD1242</td><td>L64</td><td>L63</td><td>L62</td><td>L61</td><td>L60</td><td>L59</td><td>L58</td><td>L57</td><td>L56</td><td>L55</td><td>L54</td><td>L53</td><td>L52</td><td>L51</td><td>L50</td><td>L49</td></tr></table></td></tr>
<tr><td>D9242</td><td>SD1242</td><td>存储站号 (0 到 64)</td><td>• 如果本地站在运行过程中停止，停止前的内容不保持。
SD1224 到 SD1227 和 SD1228 到 SD1231 的内容被执行逻辑或操作。
如果相应位是 “0”，上面的特殊寄存器的相应位变为有效。
• 如果自身（主）站停止，停止前的内容也被保持。</td></tr>
<tr><td>D9243</td><td>SD1243</td><td>–</td><td>用于本站的站号信息</td><td>存储站号 (0 到 64)</td><td>允许本地站去确认它的自身站号</td><td></td></tr>
<tr><td>D9244</td><td>SD1244</td><td>–</td><td>存储从站数</td><td>存储从站数</td><td>表示一个环路中的从站数。</td><td></td></tr>
<tr><td>D9245</td><td>SD1245</td><td>–</td><td>接受出错检测到的次数</td><td>接受出错检测到的次数</td><td>存储下列传送出错检测到的次数：CRC，OVER，AB.IF
计数最大到 FFFFH。要恢复此值到 “0”，执行复位操作。</td><td></td></tr>
</table>

附表 . 28 特殊寄存器列表

ACPU 特殊转换	转换后的特殊寄存器	用于改进的特殊寄存器	名称	含义	详细信息	相应的 CPU
D9248	SD1248	–	本地站运算状态	存储 1 号到 16 号的条件	存储处于 STOP 或者 PAUSE 模式的本地站号。 软元件号 | 位 b15 b14 b13 b12 b11 b10 b9 b8 b7 b6 b5 b4 b3 b2 b1 b0 SD1248 L16 L15 L14 L13 L12 L11 L10 L9 L8 L7 L6 L5 L4 L3 L2 L1 SD1249 L32 L31 L30 L29 L28 L27 L26 L25 L24 L23 L22 L21 L20 L19 L18 L17 SD1250 L48 L47 L46 L45 L44 L43 L42 L41 L40 L39 L38 L37 L36 L35 L34 L33 SD1251 L64 L63 L62 L61 L60 L59 L58 L57 L56 L55 L54 L53 L52 L51 L50 L49 对应处于 STOP 或者 PAUSE 模式的站号的位变为 “1”。 实例：当本地站 7 和 15 处于 STOP 模式，SD1248 的 b6 和 b14 变为 “1”，当 SD1248 被监视时，它的值是 “16448(4040H)”。	QnA
D9249	SD1249	–	本地站运算状态	存储 17 号到 32 号的条件		
D9250	SD1250	–	本地站运算状态	存储 33 号到 48 号的条件		
D9251	SD1251	–	本地站运算状态	存储 49 号到 64 号的条件		
D9252	SD1252	–	本地站运算状态	存储 1 号到 16 号的条件	存储出错的本地或者远程站号。 软元件号 | Bit b15 b14 b13 b12 b11 b10 b9 b8 b7 b6 b5 b4 b3 b2 b1 b0 SD1252 L16 L15 L14 L13 L12 L11 L10 L9 L8 L7 L6 L5 L4 L3 L2 L1 SD1253 L32 L31 L30 L29 L28 L27 L26 L25 L24 L23 L22 L21 L20 L19 L18 L17 SD1254 L48 L47 L46 L45 L44 L43 L42 L41 L40 L39 L38 L37 L36 L35 L34 L33 SD1255 L64 L63 L62 L61 L60 L59 L58 L57 L56 L55 L54 L53 L52 L51 L50 L49 对应出错站号的位变为 “1”。 实例：当本地站 12 出错时，SD1252 的 b11 变为 “1”，当 SD1252 被监视时，它的值是 “2048(800H)”。	
D9253	SD1253	–	本地站异常状态	存储 17 号到 32 号的条件		
D9254	SD1254	–	本地站异常状态	存储 33 号到 48 号的条件		
D9255	SD1255	–	本地站	存储 49 号到 64 号的条件		

(12) 保险丝熔断的模块

附表 . 29 特殊寄存器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD1300	保险丝熔断的模块	以 16 点为单位的位模式，表示保险丝熔断的模块 0 ： 无保险丝熔断 1 ： 保险丝熔断存在	• 保险丝熔断的输出模块的号码以位模式（以 16 点为单位）输入。 （如果已经用参数设置了模块号，参数设置号被存储。） • 也检测远程站的输出模块的保险丝熔断条件 b15 b14 b13 b12 b11 b10 b9 b8 b7 b6 b5 b4 b3 b2 b1 b0 SD1300 0 0 0 1(YC0) 0 0 0 1(Y80) 0 0 0 0 0 0 0 0 SD1301 1(Y1F0) 0 0 0 0 1(Y1A0) 0 0 0 0 0 0 0 0 0 0 SD1331 0 0 0 0 1(Y1F B0) 0 0 0 0 0 0 0 1(Y1F 30) 0 0 0 表示保险丝熔断。 • 即使保险丝熔断被更换为新的也不清除。 此标志由出错复位操作清除。	S（出错）	D9100	QnA Qn(H) QnPH QnPRH QnU Rem
SD1301					D9101	
SD1302					D9102	
SD1303					D9103	
SD1304					D9104	
SD1305					D9105	
SD1306					D9106	
SD1307					D9107	
SD1308					新增	
SD1309 到 SD1330					新增	
SD1331					新增	

(13)I/O 模块验证

附表 .30 特殊寄存器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD1400	I/O 模块验证错误	以 16 点为单位的位模式，表示有验证出错的模块。 0 ： 无 I/O 验证出错 1 ： 有 I/O 验证出错	• 当 I/O 模块信息和系统上电时注册的信息不同的 I/O 模块被检测到时，这些 I/O 模块的号码被以位模式输入。 （如果 I/O 号码是由参数设定的，则参数设定的号码被存储。） • 还检测 I/O 模块信息。 b15 b14 b13 b12 b11 b10 b9 b8 b7 b6 b5 b4 b3 b2 b1 b0 SD1400 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 (X Y 0) SD1401 0 0 0 0 0 0 1 (X Y 190) 0 0 0 0 0 0 0 0 0 SD1431 0 1 (X Y 1FE0) 0 0 0 0 0 0 0 0 0 0 0 0 0 0 表示一个I/O模块验证出错 • 即使熔断的保险丝被更换为一个新的，也不清除。此标志由出错复位操作清除。	S（出错）	D9116	QnA Qn(H) QnPH QnPRH QnU Rem
SD1401					D9117	
SD1402					D9118	
SD1403					D9119	
SD1404					D9120	
SD1405					D9121	
SD1406					D9122	
SD1407					D9123	
SD1408					新增	
SD1409 到 SD1430					新增	
SD1431					新增	

(14)过程控制指令

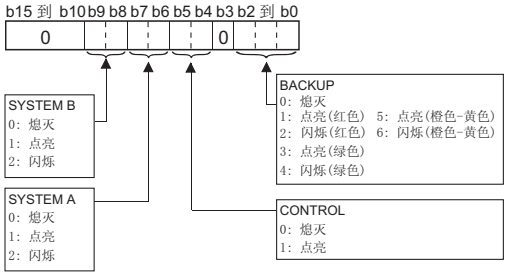
附表 .31 特殊寄存器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD1500 SD1501	基本周期	基本周期时间	• 以浮点数设置用于过程控制指令的基本周期（1 秒为单位） 浮点数 = SD1501 | SD1500	U	新增	Q4AR QnPH
SD1502	过程控制指令详细出错代码	过程控制指令详细出错代码	• 给出了发生在过程控制指令中的出错的详细出错内容	S（出错）	新增	
SD1503	过程控制指令产生出错的位置	过程控制指令产生出错的位置	• 给出了发生在过程控制指令中的出错处理块	S（出错）	新增	
SD1506 SD1507	虚拟软元件	虚拟软元件	• 用于使用过程控制指令去指定虚拟软元件	U	新增	QnPH QnPRH

(15)用于冗余系统（主站系统 CPU 信息 *1）

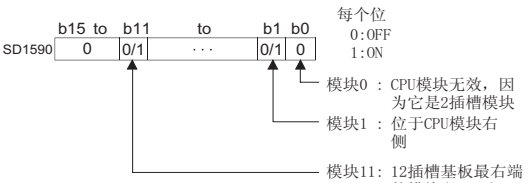
SD1510 到 SD1599 只对冗余系统有效。
对于单独系统，它们都被设定为 0。.

附表 .32 特殊寄存器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD1512	在 CPU 模块启动过程中的运行模式	热启动切换电源输出时间	• 给出了当 CPU 模块启动时，系统在运行模式中自动从热启动切换到初始化启动的电源输出时间 (S)。	S（初始化）	新增	Q4AR
SD1585	冗余系统 LED 状态	4 种 LED 状态 • BACKUP • CONTROL • SYSTEM A • SYSTEM B	用于 BACKUP、CONTRL、SYSTEMA、SYSTEMB 的 LED 状态以下列格式存储：  	S（状态改变）	新增	QnPRH
SD1588	系统切换的原因	发生在本站的系统切换的原因	存储本站系统上系统切换的原因。 下列对应系统切换方法的值被存储： 初始化为 0，当电源是被关断，然后接通或者 RESET 开关被设定到 RESET 位置，然后回到中间位置。 0：初始值（控制系统没有被切换） 1：电源关断、复位、H/W 故障，WDT 出错 2：CPU 停止出错（不包括 WDT） 3：来自网络模块的系统切换请求 16：系统切换专用指令 17：来自 GX Developer 的系统切换请求	S（当条件发生时）	○	

*1：主站 CPU 模块的信息被存储。

附表 .32 特殊寄存器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD1589	系统切换故障条件的原因	系统切换故障的原因	• 存储系统切换故障的原因。 0：系统切换正常（默认） 1：热备电缆没有连接，热备电缆出错，FPGA 电路故障。 2：H/W 故障，电源关断，复位，待机系统的 WDT 出错 3：H/W 故障，电源关断，复位，待机系统的 WDT 出错 4：跟踪数据发送初始化 5：通讯超时 6：待机系统的严重出错（不包括 WDT 出错） 7：两个系统之间有差异 （只有后备模式能检测到） 8：在从控制系统到待机系统的内存复制过程中 9：在运行中程序更改过程中 10：待机系统上，处于智能功能模块故障的检测过程中 11：系统切换正在执行 • 当本站系统电源上电时，复位到 “0”。 • 只要系统切换成功，复位到 “0”。	S（当系统被切换时）	○	QnPRH
SD1590	切换请求网络号	请求源网络号	• 当 SM1590 变为 ON 时，存储请求源工作网络号。	S（出错）	新增	Q4AR
	请求系统切换的网络模块起始地址	请求系统切换的网络模块起始地址	• 存储发起系统切换请求的网络模块的起始地址。 • 在网络出错被用户复位之后，自动由系统关断。  • 请参考 SD1690，它存储了其他系统上相应网络模块的起始地址。	S（出错 / 状态改变）	新增	QnPRH
SD1595	内存复制目标 I/O 号码	内存复制目标 I/O 号码	• 在 SM1595 从 OFF → ON 之前存储内存复制目标 I/O 号（待机系统 CPU 模块：3D1H）。	U	新增	
SD1596	内存复制状态	内存复制状态	• 存储内存复制功能的执行结果。 0 ：内存复制成功结束 4241H ：待机系统电源关断 4242H ：热备电缆断开或者损坏 4247H ：内存复制功能正在执行 4248H ：不支持内存复制目标 I/O 号码	S（状态改变）	新增	

(16)用于冗余系统(其他系统 CPU 信息 *1)

SD1600 到 SD1659 只对冗余系统的备份模式有效，当处于独立模式时刷新不能执行。
SD1651 到 SD1699 对备用模式和独立模式都有效。当处于单独系统中时，SD1600 到 SD1699 都为 0。

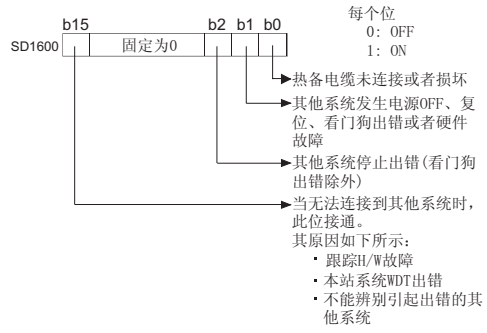
附表 .33 特殊寄存器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU SD□□*2	相应的 CPU
SD1600	检测出错	检测出错号	• 以 BIN 码存储在其他系统 CPU 模块检测过程中发生的出错的出错号。 • 存储最新发生的出错。	S(每次 END)	SD0	Q4AR
SD1601	检测出错发生时间	检测出错发生时间	• SD1600 存储更新的日期和时间。 • 存储 BCD 码的两位数中的一个。 • 对于存储状态参考 SD1 到 SD3。 (SD1→SD1601，SD2→SD1602，SD3→SD1603)	S(每次 END)	SD1 到 SD3	
SD1602						
SD1603						
SD1604	出错信息分类	出错信息分类	• 存储出错注释信息 / 个别信息分类代码。 • 对于存储状态参考 SD4。	S(每次 END)	SD4	
SD1605	出错公共信息	出错公共信息	• 存储出错代码的公共信息。 • 对于存储状态参考 SD5 到 SD15。 (SD5→SD1605，SD6→SD1606，SD7→SD1607，SD8→SD1608，SD9→SD1609，SD10→SD1610，SD11→SD1611，SD12→SD1612，SD13→SD1613，SD14→SD1614，SD15→SD1615)	S(每次 END)	SD5 到 SD15	
SD1606						
SD1607						
SD1608						
SD1609						
SD1610						
SD1611						
SD1612						
SD1613						
SD1614						
SD1615						
SD1616	出错个别信息	出错个别信息	• 存储出错代码的个别信息。 • 对于存储状态参考 SD16 到 SD26。 (SD16→SD1616，SD17→SD1617，SD18→SD1618，SD19→SD1619，SD20→SD1620，SD21→SD1621，SD22→SD1622，SD23→SD1623，SD24→SD1624，SD25→SD1625，SD26→SD1626)	S(每次 END)	SD16 到 SD26	
SD1617						
SD1618						
SD1619						
SD1620						
SD1621						
SD1622						
SD1623						
SD1624						
SD1625						
SD1626						
SD1650	开关状态	CPU 模块开关状态	• 存储 CPU 模块的开关状态。 • 对于存储状态参考 SD200。 (SD1650→SD200)	S(每次 END)	SD200	
SD1651	LED 状态	CPU 模块的 LED 状态	• 存储 CPU 模块的 LED 状态。 • 熄灭为 0，点亮为 1，闪烁为 2。 • 对于存储状态参考 SD201。 (SD1651→SD201)	S(每次 END)	SD201	
SD1653	CPU 模块运行状态	CPU 模块运行状态	• 存储 CPU 模块的运行状态。 • 对于存储状态参考 SD203。 (SD1653→SD203)	S(每次 END)	SD203	

*1 ： 存储其他系统 CPU 模块的检测信息和系统信息。

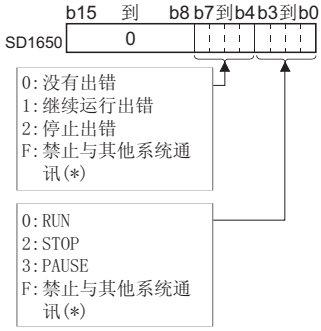
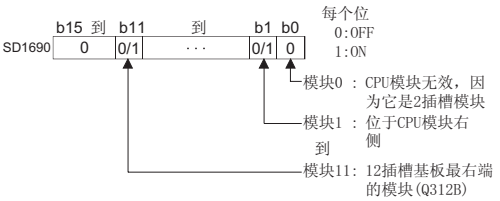
*2 ： 表示用于主站系统 CPU 模块的特殊寄存器 (SD□□)。

附表 .33 特殊寄存器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU SD□ □*2	相应的 CPU
SD1600	系统出错信息	系统出错信息	• 如果对冗余系统的出错检查检测到出错，下面所示的相应位变为 ON。当出错被清除之后，此位变为 OFF。  • 如果 b0、b1、b2 和 b15 中如何一个是 ON，则其他位都为 OFF。 • 在调试模式中，b0、b1、b2 和 b15 都为 OFF。	S(每个 END)	-	QnPRH
SD1601	系统切换结果	系统切换结果	存储引起系统切换的原因。 • 当系统切换发生时，将系统切换的原因存储到两个系统的 SD1601 中。 • 在电源从 OFF 到 ON/ 从复位到重新置位时，初始化到 0。 • 下列给出的值被存储到此寄存器中。 0: 初始化值 (系统切换还未发生) 1: 电源关断，复位，H/W 故障，WDT 出错，(*) 2: CPU 停止出错 (WDT 除外) 3: 网络模块发出的系统切换请求 16: 系统切换专用指令 17: 来自 GX Developer 的系统切换请求 *: 当系统切换是由控制系统的电源 OFF/ 复位引起时，“1” 不被存储到新待机系统的 SD1601。	S(当系统被切换时)		
SD1602	系统切换专用指令参数	系统切换专用指令参数	• 存储用于系统切换专用指令 SP.CONTSW 的参数。 (用于 SP.CONTSW 的参数 (SD1602) 在两个系统 A 和 B 中都存储。) • 只有当 “16” 被存储到 SD1601 中时，SD1602 才有效。 • 只要系统切换指令 SP.CONTSW 被激活，SD1602 就被更新。	S(当系统被切换时)		
SD1610	其他系统检测出错	检测出错代码	• 出错值以 BIN 码存储。 • 存储其他系统 CPU 模块的 SD0	S(每个 END)	SD0	
SD1611 SD1612 SD1613	其他系统检测出错发生时间	检测出错发生时间	• 存储和存储在 SD1610 中的出错代码相对应的检测出错发生时的日期和时间。 • 数据格式和 SD1 到 SD3 相同。 • 也存储此值到 SD1 到 SD3。	S(每个 END)	SD1 到 SD3	
SD1614	其他系统出错信息类别	出错信息类别代码	• 存储对应出错注释信息 / 个别信息代码的类别代码。 • 数据格式和 SD4 的相同。 • 也存储此值到 SD4 中。	S(每个 END)	SD4	
SD1615 到 SD1625	其他系统出错公共信息	出错公共信息	• 存储和存储在此系统 CPU 中的出错代码的相对应的公共信息。 • 数据格式和 SD5 到 SD15 的相同。 • 也存储此值到 SD5 到 SD15 中。	S(每个 END)	SD5 到 SD15	
SD1626 到 SD1636	其他系统出错个别信息	出错个别信息	• 存储和存储在此系统 CPU 中的出错代码相对应的个别信息。 • 数据格式和 SD16 到 SD26 的相同。 • 也存储此值到 SD16 到 SD26 中。	S(每个 END)	SD16 到 SD26	
SD1649	待机系统上出错清除	要被清除的出错的出错代码	• 存储通过清除待机系统出错而被清除的出错的出错代码。 • 存储要被清除的出错的出错代码到此寄存器，将 SM1649 从 OFF 变到 ON 以清除待机系统出错。 • 当存储到此寄存器时，出错代码的最低位 (1 位置) 中的值被忽略。 (通过存储 4100 到此寄存器以及复位出错，出错 4100 到 4109 可以被清除。)	S(每个 END)	-	

*2: 表示用于主站系统 CPU 模块的特殊寄存器 (SD□ □)。

附表 .33 特殊寄存器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU SD□ □*2	相应的 CPU
SD1650	其他系统运行信息	其他系统运行信息	以下列格式存储另一个系统 CPU 模块的运行信息。 当发生通讯出错，或者当处于调试模式时，存储 “00FFH”。  *: 禁止与其他系统通讯，调试模式 注释：通讯出错是由下列原因引起： • 当电源被关断，或者当其他系统被复位。 • 在系统或者 B 中发生 H/W 出错。 • WDT 出错发生。 • 热备电缆未连接。 • 热备电缆断开或者损坏。	S（每个 END）	–	QnPRH
SD1690	请求在主站（控制）系统上进行系统切换的网络模块的起始地址	请求在主站（控制）系统上进行系统切换的网络模块的起始地址	• 存储发出系统切换请求的网络模块的起始地址，使用下列格式。 • 在网络出错被用户复位之后自动由系统关断。  • 参考存储相应的主站系统上网络模块的起始地址的 SD1590。	S（每个 END）		

*2 : 表示用于主站系统 CPU 的特殊寄存器 (SD□ □)。

(17)用于冗余系统（交换）

SD1700 到 SD1779 只对冗余系统有效。

对于单独系统，这些寄存器都为 0。.

附表 .34 特殊寄存器列表

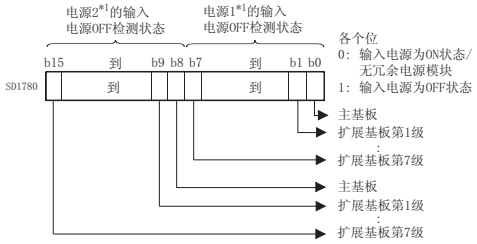
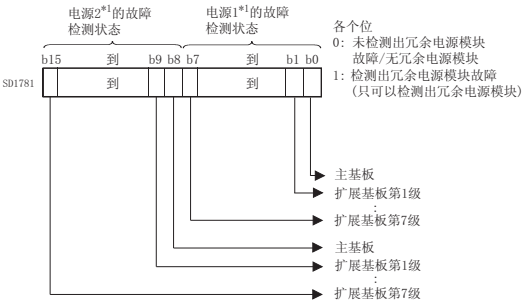
号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□	相应的 CPU
SD1700	交换出错检测计数	交换出错检测计数	• 将交换出错检测计数值 +1。	S（出错）		Q4AR QnPRH
SD1710	运行中程序改变的等待时间（待机系统）	运行中程序改变的等待时间（待机系统）	• 以秒为单位设定，从用于冗余功能的运行中程序更改完成控制系统 CPU 模块的运行中程序更改开始，到待机系统 CPU 模块的运行中程序更改启动为止，待机系统 CPU 模块的等待时间。 • 如果控制系统 CPU 模块的运行中程序更换结束之后，在预设的时间以内没有运行中程序更改请求发送到待机系统 CPU 模块，则两个系统 CPU 模块认为这是用于冗余的运行中程序更改故障。在这种情形中，系统 CPU 模块恢复在运行中程序改变过程中被挂起的，系统 A&B 之间的一致性检查。而且，控制系统 CPU 模块被设定为接收新的用于冗余的运行中程序更改请求。 • 当两个系统上电时，将 90 秒设定到 SD1710 中，作为默认值。 • 设定此值在 90 到 3600 秒范围以内。当设置是 0 到 89 秒时，认为 90 秒用于操作。如果设置超出允许的范围，则认为 0 到 3600 秒以外的时间用于操作。 • 在用于冗余的多个块的运行中更换的和批量文件的运行中更换的过程中，依照 SD1710 中的设置，检查用于待机系统 CPU 模块的运行中程序更改启动的等待时间。	U/S（初始化）	新增	QnPRH

*2 ： 表示用于本站系统 CPU 的特殊寄存器 (SD□□)。

(18)冗余电源模块信息

SD1780 到 SD1789 只对冗余电源系统有效。对于单个电源系统，这些位都为 0。.

附表 .35 特殊寄存器列表

号码	名称	含义	解释	设置方式（当被设定时）	相应的 ACPU D9□□□	相应的 CPU
SD1780	电源 OFF 检测状态	电源 OFF 检测状态	• 以下列位模式存储输入电源 OFF 状态时的冗余电源模块 (Q64RP) 的状态。 • 当主基板不是冗余电源主基板 (Q38RB) 时，本寄存器存储 0。  • 多 CPU 系统配置时，状态只被存储到 1 号机的 CPU 模块中。	S（每次 END）	新增	Qn(H)*2 QnPH*2 QnPRH Rem
SD1781	电源模块故障检测状态	电源模块故障检测状态	• 以下列位模式存储冗余电源模块 (Q64RP) 的故障检测状态。（当检测出冗余电源模块有故障之后，有故障的冗余电源模块的输入电源被切换到 OFF 时，相应位被清除为 0。） • 当主基板不是冗余电源主基板 (Q38RB) 时，此寄存器存储 0。  • 多 CPU 系统配置时，状态只被存储到 1 号机的 CPU 模块中。	S（每次 END）	新增	
SD1782	用于电源模块 1*1 的瞬间掉电检测计数器	用于电源 1 的瞬间掉电检测计数	• 对电源 1/ 电源 2 的瞬间掉电的次数进行计数。 • 监视安装在冗余电源主基板 (Q38RB) 上的电源 1/ 电源 2 的状态，并对瞬间掉电的次数进行计数。 • 不监视安装在冗余扩展基板上的电源 1/ 电源 2 的状态。 • 当 CPU 模块启动时，电源 1/ 电源 2 的计数器被清除为 0。 • 如果到两个冗余电源模块中的一个的输入电源被切换为 OFF，和输入电源被切换为 OFF 的冗余电源模块相对应的计数器被清除为 0。 • 每次检测到电源 1/ 电源 2 的瞬间掉电时，计数器加 1。（0 ～ 65535：当计数超出 65535 时，从 0 开始重新计数。） • 当主基板不是冗余电源主基板 (Q38RB) 时此寄存器存储 0。 • 多 CPU 系统配置时，状态只被存储到 1 号机的 CPU 模块中。	S（每次 END）	新增	
SD1783	用于电源模块 2*1 的瞬间掉电检测计数器	用于电源 1 的瞬间掉电检测计数		S（每次 END）	新增	

*1: “电源 1”表示安装在冗余基板 (Q38RB/68RB/Q65WRB) 的 POWER1 插槽中的冗余电源模块 (Q64RP)。
“电源 2”表示安装在冗余基板 (Q38RB/68RB/Q65WRB) 的 POWER2 插槽中的冗余电源模块 (Q64RP)。

*2: 以序列号的高 5 位为 “04012”以后的 CPU 为对象。
多 CPU 系统配置中，以序列号的高 5 位为 “07032”以后的 CPU 为对象。

附录 3 参数号一览

在参数设置中发生了错误时，参数号被存储到特殊寄存器 (SD16 ～ 26) 中。
以下为参数号与参数设置位置的对应表。

附表 .36 参数号一览

项目		参数号	参阅章节
标识		0000H	8.1.1 项 (1)
注释		0001H	8.1.2 项 (1)
I/O 分配	类型	0400H	4.7 节 8.1.1 项 (8) 8.1.2 项 (10)
	型号		
	点数		
	起始 XY(起始 I/O 号)		
基本设置	基板型号	0401H	4.4 节 8.1.1 项 (8) 8.1.2 项 (10)
	电源模块型号		
	扩展电缆型号		
	插槽数		
详细设置	出错时的输出模式	0403H	6.8 节 8.1.1 项 (8) 8.1.2 项 (10)
	I/O 响应时间	0405H	6.7 节 8.1.1 项 (8) 8.1.2 项 (10)
	控制 CPU	0406H	8.1.1 项 (8) 8.1.2 项 (10) QCPU 用户手册 (多 CPU 系统篇)
开关设置		0407H	6.10 节 8.1.1 项 (8) 8.1.2 项 (10)
组号		05mnH	8.3 节 (2)
		0AmnH	8.3 节 (1)
冗余参数		0D00H	8.2 节 QCPU 用户手册 (冗余系统篇)

(转下页)

附表.36 参数号一览（续）

项目		参数号	参阅章节
CPU 台数		0E00H	8.1.1 项 (11)
动作模式		0E01H	8.1.2 项 (12)
组外的 I/O 设置	接收组外的输入状态	0E04H	QCPU 用户手册（多 CPU 系统篇）
	接收组外的输出状态		
定时器时限设置	低速	1000H	8.1.1 项 (2) 8.1.2 项 (2) 9.2.10 项
	高速		
RUN-PAUSE 触点	RUN	1001H	6.6.1 项 8.1.1 项 (2) 8.1.2 项 (2)
	PAUSE		6.6.2 项 8.1.1 项 (2) 8.1.2 项 (2)
远程复位		1002H	6.6.3 项 8.1.1 项 (2) 8.1.2 项 (2)
STOP → RUN 时的输出模式		1003H	6.4 节 8.1.1 项 (2) 8.1.2 项 (2)
浮点运算处理		1004H	3.9.4 项 8.1.2 项 (2)
公共指针号		1005H	3.9.2 项 8.1.2 项 (2)
空插槽点数		1007H	4.6.1 项 (5) 8.1.1 项 (2) 8.1.2 项 (2)
中断程序 / 恒定周期程序设置		1008H	3.1.3 项 8.1.1 项 (2) 8.1.2 项 (2)
系统中断设置	中断计数器起始号		8.1.1 项 (2) 8.1.2 项 (2) 9.2.11 项 (4)
	恒定周期间隔 (n:28 ～ 31)		8.1.1 项 (2) 8.1.2 项 (2) 9.10 节

（转下页）

附表.36 参数号一览（续）

项目			参数号	参阅章节
智能功能模块设置（中断指针设置）			100A$_H$	9.10 节 8.1.2 项 (2)
模块同步设置			100C$_H$	8.1.1 项 (2) 8.1.2 项 (2)
A 系列 CPU 兼容设置			100D$_H$	8.1.2 项 (2) 9.3.2 项 9.3.3 项
使用串行口通讯功能			100E$_H$	6.24 节 8.1.1 项 (9)
发送速度				
总数确认				
总数等待时间				
运行中写入设置				
系统中断设置	高速中断设置	X 输入	100F$_H$	3.1.3 项 3.3.5 项 8.1.2 项 (2)
		Y 输出	1010$_H$	
		缓冲读取	1011$_H$	
		缓冲写入	1012$_H$	
	服务处理设置		1013$_H$	9.6 节
锁存数据备份操作有效触点			1014$_H$	6.29 节
文件寄存器			1100$_H$	8.1.2 项 (3) 9.7 节
指令中使用的注释文件			1101$_H$	8.1.2 项 (3)
软元件初始值			1102$_H$	6.26 节 8.1.1 项 (3) 8.1.2 项 (3)
局部软元件用文件			1103$_H$	9.13.1 项 8.1.2 项 (3)
锁存数据备份操作时传送至标准 ROM			1104$_H$	6.29 节
SP.DEVST/S.DEVLD 指令中使用的文件			1105$_H$	6.30 节

（转下页）

附表.36 参数号一览（续）

项目		参数号	参阅章节
软元件点数		2000H	8.1.1 项 (5) 8.1.2 项 (6) 9.1 节 9.2 节
锁存 (1) 起始 / 最终		2001H	3.7 节 6.3 节 8.1.1 项 (5) 8.1.2 项 (6)
锁存 (2) 起始 / 最终		2002H	
局部软元件起始 / 最终		2003H	9.13.1 项 8.1.2 项 (6)
WDT（看门狗定时器）设置	WDT 设置	3000H	8.2 节 8.1.1 项 (4) 8.1.2 项 (4)
	初始执行监视时间		3.3.1 项 8.1.2 项 (4)
	低速执行监视时间		3.3.3 项 8.1.2 项 (4)
出错检查	执行电池检查	3001H	6.17 节 8.1.1 项 (4) 8.1.2 项 (4)
	执行保险丝熔断检查		
	执行模块校验		
出错时的运行模式	运算错误	3002H	6.17 节 8.1.1 项 (4) 8.1.2 项 (4)
	扩展指令错误		
	保险丝熔断		
	执行模块校验		
	智能模块程序执行出错		
	文件访问出错		
	存储卡操作出错		
	外部电源供给 OFF		

（转下页）

附表 .36 参数号一览（续）

项目		参数号	参阅章节
恒定扫描		3003H	6.2 节 8.1.1 项 (4) 8.1.2 项 (4)
故障历史记录		3005H	6.18 节 8.1.2 项 (4)
低速程序执行时间		3006H	3.3.3 项 8.1.2 项 (4)
存储器检查	执行程序内存检查	3008H	8.1.2 项 (5)
详细设置	H/W 出错时 CPU 动作模式	4004H	6.9 节 8.1.1 项 (8)
MELSECNET/H 个数设置		5000H	8.3 节 (2)
其它站存取时的有效模块		5001H	
链接间传输参数		5002H	
路由参数		5003H	
起始 I/O 号		5NM0H	
网络号			
总站数			
模式		5NM0H	
刷新参数		5NM1H	
公共参数		5NM2H	
站固有参数		5NM3H	
子站用参数		5NM5H	
公共参数 2		5NMAH	
站固有参数 2		5NMBH	
中断设置			

（转下页）

附表.36 参数号一览（续）

项目		参数号	参阅章节
程序设置		7000H	3.3.6 项
引导选项	程序内存清除	7000H	5.2.8 项 5.2.9 项 8.1.1 项 (6) 8.1.2 项 (7) 8.1.2 项 (8)
	存储卡→标准 ROM 全部数据自动写入		
引导文件设置			
SFC 程序启动模式		8002H	8.1.1 项 (7) 8.1.2 项 (9)
启动条件		8003H	
块停止时的输出模式		8005H	
以太网个数设置		9000H	8.3 节 (3)
起始 I/O 号		9N00H	
网络号			
组号			
站号			
动作设置			
初始设置		9N01H	
开放设置		9N02H	
路由器中继参数		9N03H	
站号←→ IP 相关信息		9N05H	
FTP 参数		9N06H	
电子邮件设置		9N07H	
	通知设置	9N08H	
中断设置		9N09H	
路由参数		9N04H	
MELSECNET/G 个数设置		A000H	8.3 节 (1)
链接间传送（数据链接间传送参数）		A002H	
路由参数		A003H	
起始 I/O No.		ANM0H	
网络 No.			
链接总（从）站数			
模式		ANM0H	
刷新参数		ANM1H	
公共参数		ANM2H	
站固有参数		ANM3H	

（转下页）

附表.36 参数号一览（续）

项目	参数号	参阅章节
模块数量设置	C000H	8.3 节 (4)
远程输入 (RX) 刷新软元件	CNM1H	
远程输出 (RY) 刷新软元件		
远程寄存器 (RWr) 刷新软元件		
远程寄存器 (RWw) 刷新软元件		
版本 2 远程输入 (RX) 刷新软元件		
版本 2 远程输出 (RY) 刷新软元件		
版本 2 远程寄存器 (RWr) 刷新软元件		
版本 2 远程寄存器 (RWw) 刷新软元件		
特殊继电器 (SB) 刷新软元件		
特殊寄存器 (SW) 刷新软元件		
起始 I/O No.	CNM2H	
动作设置		
总连接个数		
重试次数		
自动恢复个数		
待机主站编号		
CPU 宕机指定		
扫描模式指定		
延迟时间设置		
站信息设置		
远程设备站		
初始设置		
中断设置		

（转下页）

附表.36 参数号一览（续）

项目		参数号	参阅章节
电源 ON 时的设置		D001H	8.2 节 (1) QnPRHCPU 用户手册 （冗余系统篇）
待机系统监视设置			
调试模式设置			
备份模式设置			
热备传输模式设置		D002H	8.2 节 (2) 参阅 QnPRHCPU 用户手册（冗余系统篇）的“5.5.3 热备发送设置数据”
热备软元件设置		D003H	
上升沿 / 下降沿执行指令用历史记录（信号流）			
软元件详细设置			
	热备块号		
	自动传输热备块号 1（自动启动 SM1520）		
	软元件范围设置		
	文件寄存器文件设定		
组设置		D004H	8.3 节 (2) 8.3 节 (3)
冗余设置		D5**H	8.3 节 (2)
		D9**H	8.3 节 (3)
刷新设置		E002H E003H	8.1.1 项 (11) 8.1.2 项 (12) QCPU 用户手册 （多 CPU 系统篇）
在线模块更换设置		E006H	8.1.2 项 (12) QCPU 用户手册 （多 CPU 系统篇）
刷新参数详细软元件指定		E007H	
多 CPU 间高速通信功能各 CPU 发送范围设置		E008H	
多 CPU 间高速通信功能自动刷新设置		E009H	
多 CPU 间高速通信功能自动刷新软元件设置		E00AH	
多 CPU 间同步启动设置		E00BH	
本机 No. 设置		E00CH	

（转下页）

附录 4 比较

Q 系列 CPU 模块通过版本升级来添加功能、变更规格等。
能够在 CPU 模块中使用的功能、规格因功能版本 / 系列 No. 而异。

附录 4.1 基本模式 QCPU 的功能升级

(1) 规格比较

附表 .37 规格比较

规格		CPU 模块序列号的高 5 位	
		功能版本 A	功能版本 B
		“04121”或以前	“04122”或以后
标准 RAM 容量	Q00JCPU	×	
	Q00CPU	64k 字节	128k 字节
	Q01CPU	64k 字节	128k 字节
CPU 共享内存	Q00JCPU	×	
	Q00CPU	×	○
	Q01CPU	×	○

○：可使用 / 对应，×：禁止使用 / 不对应

(2) 功能比较及 GX Developer 的版本与追加功能的对应

附表 .38 追加功能及 GX Developer 的版本与追加功能的对应

追加功能	对应功能版本	对应序列号	对应 GX Developer
功能块 （☞ GX Developer 操作手册（功能块篇））	A	04121 以前	版本 8.00A 以后
结构化文本 (ST) 语言 （☞ QCPU(Q 模式）编程手册（结构化文本篇））			
MELSAP3 （☞ QCPU(Q 模式)QnACPU 编程手册 (SFC 篇)）	B	04122 以后	
PID 运算功能 *1 （☞ QCPU(Q 模式)QnACPU 编程手册 (PID 控制指令篇)）			
实数运算功能 *1（☞ 9.12.3 项）			
智能功能模块事件中断 （☞ 6.23 节）			
软元件初始值自动设定功能 （☞ 6.26 节）			
远程口令设定功能 （☞ 6.19.2 项）			
电子邮件参数 （☞ 电子邮件功能对应模块的手册）			
使用了指针的 RUN 中写入（☞ 6.15.2 项）			
文件寄存器 R 的大容量化 *2（☞ 9.7 节）			–
多 CPU 系统对应 （☞ QCPU 用户手册（多 CPU 系统篇）） *2			版本 8.00A 以后
多块 RUN 中写入 （☞ 5.1.2 项 (3)(b)，5.2.2 项 (3)(b)）			
CC-Link 远程网络新增模式对应 （☞ CC-Link 系统主站 / 本地站用户手册（详细篇））		06112 以后	版本 8.03D 以后

–：与 GX Developer 无关　*：不兼容 Q00JCPU

*1：通过版本 7 或者更低版本的 GX Developer 读取装载在 GX Developer 版本 8 上的 CPU 指令时，它将被 GX Developer 处理为“指令代码出错”。

*2：不对应于 Q00JCPU。

(3) 基本模式 QCPU 的不同点

附表 .39 基本模式 QCPU 的不同点

项目			Q00JCPU	Q00CPU	Q01CPU
CPU 模块			CPU 模块、电源模块、主基板 (5 插槽) 一体机	CPU 模块单体	
主基板 / 小型主基板			不要	要	
扩展基板			可连接 但使用小型主基板时不能连接		
扩展段数			最大 2 段	最大 4 段 (但使用小型主基板时不能连接)	
可安装 I/O 模块数			16 个	24 个	
电源模块					
	主基板		不要	要	
	小型主基板		不要	要	
	扩展基板	Q52B、Q55B	不要		
		Q63B、Q65B、Q68B、Q68RB、Q612B	要		
扩展电缆			QC05B、QC06B、QC12B、QC30B、QC50B、QC100B		
存储卡接口			无		
外部接口		RS-232	有 (发送速度 :9.6kbps、19.2kbps、38.4kbps、57.6kbps、115.2kbps)		
		USB	无		
处理速度 (顺控程序指令)		LD X0	0.20μs	0.16μs	0.10μs
		MOV D0 D1	0.70μs	0.56μs	0.35μs
程序容量 *1			8k 步 (32k 字节)	8k 步 (32k 字节)	14k 步 (56k 字节)
内存容量	程序内存		58k 字节	94k 字节	
	标准 RAM		--------	128k 字节 *2	
	标准 ROM		58k 字节	94k 字节	
	CPU 共享内存 *3		无	1k 字节 (用户自由区 320 字)	
标准 ROM 的写入次数			最多 10 万次		
软元件内存容量			在 16.4k 字的范围内可变更软元件点数		
I/O 软元件点数 (含远程 I/O)			2048 点		
I/O 点数			256 点	1024 点	
文件寄存器			无	有	
串行口通讯功能			无	有 (使用 CPU 模块的 RS-232 接口)	

*1: 程序容量的 1 步为 4 字节。
*2: 在功能版本 A 中为 64k 字节。
*3: 功能版本 B 中添加的内存。
CPU 共享内存不被锁存。
可编程控制器的电源 OFF → ON 或者进行 CPU 模块的复位操作时，CPU 共享内存将被清除。

附录 4.2 高性能模式 QCPU 的功能升级

(1) 规格比较

附表 .40 规格比较

规格		CPU 模块序列号的高 5 位				
		功能版本 A		功能版本 B		
		02091 或以前	02092 或以后	“02112” 或以后	03051 或以后	04012 或以后
标准 RAM 容量	Q02CPU	64k 字节				
	Q02HCPU	64k 字节				128k 字节
	Q06HCPU	64k 字节				128k 字节
	Q12HCPU	64k 字节	256k 字节			
	Q25HCPU	64k 字节	256k 字节			
CPU 共享内存		×	×	○	○	○
SRAM 卡的电池寿命延长		×	×	×	×	○
2M 字节的 SRAM 卡对应		×	×	×	×	○

○：可使用 / 对应；×：禁止使用 / 不对应

(2) 功能比较及根据 GX Developer 的版本能否使用新增功能

附表.41 功能比较及根据 GX Developer 的版本能否使用新增功能

新增功能	对应功能版本	对应序列号	对应 GX Developer
自动写入到标准 ROM 的功能 (☞ 5.2.8 项)	A	02092 或以后	版本 6 或以后
外部 I/O 强制 ON/OFF 功能 (☞ 6.11.3 项)			
远程口令设定功能 (☞ 6.19.2 项)			
MELSECNET/H 远程 I/O 网络对应 (☞ 4.6.2 项)			
中断模块对应 (☞ 9.10 节)			
编程模块对应 (☞ QCPU 用户手册 (硬件设计 / 维护点检篇))			—
多 CPU 系统对应 (☞ QCPU 用户手册 (多 CPU 系统篇))	B	02122 或以后	版本 7 或以后
多 CPU 系统的 PC CPU 模块对应 (☞ QCPU 用户手册 (多 CPU 系统篇))		03051 或以后	版本 7.10L 或以后
高速中断功能 (☞ 6.22 节)		04012 或以后	版本 8 或以后
专用指令的模块指定的变址修饰对应 (☞ 可使用专用指令的智能功能模块的手册)			—
COM 指令的刷新项目的选择 (☞ 程序手册公共指令篇)			—
SFC 程序文件的运行中写入 (☞ 6.12.2 项)		04122 或以后	版本 8 或以后
文件存储器容量更改 (☞ 5.4.4 项)			
CC-Link 远程网络新增模式对应 (☞ CC-Link 系统主站 / 本地站模块用户手册 (详细篇))		05032 或以后	版本 8.03D 或以后
不完全微分 PID 运算功能 (☞ QCPU(Q 模式)/QnACPU 编程手册 (PID 控制指令篇))			
浮点比较指令的高速化			—
SFC 激活步注释的读取对应 (☞ QCPU(Q 模式)/QnACPU 编程手册 (SFC 篇))		07012 或以后	—

-: 与 GX Developer 无关的功能

附表 .41 功能比较及根据 GX Developer 的版本能否使用新增功能（续）

新增功能	对应功能版本	对应序列号	对应 GX Developer
电源冗余系统的出错检测（☞ 6.20 节）	B	07032 或以后	版本 8.23Z 或以后
1/1000 秒单位的时钟数据对应（☞ 6.5 节）			—
采样追踪文件的标准 RAM 存储（☞ 6.14 节）			版本 8.23Z 或以后
多 CPU 系统时的刷新软元件的个别设置（☞ QCPU 用户手册（多 CPU 系统篇））			
运行中写入时的下降沿指令的执行 / 不执行选择（☞ 6.12.3 项）		07092 或以后	版本 8.27D 或以后
CC-Link 循环数据站单位块保证功能对应（☞ CC-Link 系统主站 / 本地站模块用户手册（详细篇））		08032 或以后	版本 8.32J 或以后
8 个 CC-Link 参数设置对应（☞ CC-Link 系统主站 / 本地站模块用户手册（详细篇））			
MELSECNET/G 网络对应（☞ MELSECNET/G 网络系统参考手册（控制网络篇））		09012 或以后	版本 8.45X 或以后
ATA 卡变更对应（☞ QCPU 用户手册（硬件设计 / 维护点检篇））			

-: 与 GX Developer 无关的功能

附录 4.3 过程控制 CPU 的功能升级

(1) 功能比较及根据 GX Developer 的版本能否使用新增功能

附表 .42 功能比较及根据 GX Developer 的版本能否使用新增功能

新增功能	对应功能版本	对应序列号	对应 GX Developer
专用指令的模块指定的变址修饰对应 (☞ 可使用专用指令的智能功能模块的手册)	C	07032 或以后	--
COM 指令的刷新项目的选择 (☞ QCPU(Q 模式)/QnACPU 编程手册 (公共指令篇))			
SFC 程序文件的运行中写入 (☞ 6.12.2 项)			版本 8 (版本8.22Y或以前)
文件的容量单位 (☞ 5.4.4 项)			
CC-Link 远程网络新增模式对应 (☞ CC-Link 系统主站 / 本地站模块用户手册 (详细篇))			
存储器检查功能 (☞ 6.28 节)			版本 8.23Z 或以后
SFC 激活步注释的读取对应 (☞ QCPU(Q 模式)/QnACPU 编程手册 (SFC 篇))			--
电源冗余系统的出错检测 (☞ 6.20 节)			版本 8.23Z 或以后
1/1000 秒单位的时钟数据对应 (☞ 6.5 节)			--
采样追踪文件的标准 RAM 存储 (☞ 6.14 节)			版本 8.23Z 或以后
多 CPU 系统时的刷新软元件的个别设置 (☞ QCPU 用户手册 (多 CPU 系统篇))			
运行中写入时的下降沿指令的执行 / 不执行选择 (☞ 6.12.3 项)		07092 或以后	版本 8.27D 或以后
CC-Link 循环数据站单位块保证功能对应 (☞ CC-Link 系统主站 / 本地站模块用户手册 (详细篇))		08032 或以后	版本 8.32J 或以后
8 个 CC-Link 参数设置对应 (☞ CC-Link 系统主站 / 本地站模块用户手册 (详细篇))			

-: 与 GX Developer 无关的功能

附录 4.4 冗余 CPU 的功能升级

(1) 新增功能及根据 GX Developer 的版本能否使用新增功能

附表 .43 新增功能及根据 GX Developer 的版本能否使用新增功能

新增功能	对应功能版本	对应序列号	对应 GX Developer
SFC 激活步注释的读取对应 (☞ QCPU(Q 模式)/QnACPU 编程手册 (SFC 篇))	D	07032 以后	—
1/1000 秒单位的时钟数据对应 (☞ 6.5 节)			
采样追踪文件的标准 RAM 存储 (☞ 6.14 节)			版本 8.23Z 以后
运行中写入时的下降沿指令的执行 / 不执行选择 (☞ 6.12.3 项)		07092 以后	版本 8.27D 以后
扩展基板对应 (☞ QnPRHCPU 用户手册 (冗余系统篇))		09012 以后	版本 8.45X 以后

-: 与 GX Developer 无关的功能

附录 5 从 Qn(H)CPU 转换为 QnUCPU 时的限制及替代方法

(1) 限制事项一览表

从 Qn(H)CPU 转换为 QnUCPU 时的限制事项如附表 .44 所示。

附表 .44 从 Qn(H)CPU 转换为 QnUCPU 时的限制及替代方法

项目		限制内容	替代方法	参照
系统构成	可连接产品	不能使用功能版本 A 的 CC-Link 主站 / 本地站模块。	使用功能版本 B 或以后的 CC-Link 主站 / 本地站模块。(基本模式的限制也同样适用于此。)	-----
		不能 AnS/A 系列的模块。	使用 Q 系列的模块。	-----
		不能连接 AnS/A 系列的模块安装用扩展基板。	连接 Q 系列的模块安装用扩展基板。	-----
	工具连接	不能原样不变地使用以前的软件包 (GX Developer 等)。	使用 QnUCPU 对应的软件包。	QCPU 用户手册 (硬件设计 / 维护点检篇)
程序	语言・指令	不能使用附表 .45 中所示的指令。	-----	附录 5(2)
	浮动小数点运算	不能内部双精度运算。	用以下指令替代。 ・浮动小数点运算指令 (双精度) ・单精度→双精度转换指令 ・双精度→单精度转换指令	-----
	程序执行类型变更	不能通过远程操作变更程序的执行类型。	-----	-----
	低速程序	不能使用低速程序。	使用其它的执行类型 (扫描、恒定周期) 的程序。	-----
	中断程序	不能使用 0.2ms ～ 1ms 间隔的高速中断功能的中断指针 (I49)。	-----	-----
		不能使用中断计数器。	通过程序一览监视确认中断程序执行次数。	-----
	局部软元件	不能进行各个程序的局部软元件设置。(不能进行文件的使用方法设置。)	-----	-----
	软元件初始值	不能进行各个程序的软元件初始值设置。(不能进行文件的使用方法设置。)	-----	-----

(接下一页)

附表 . 44 从 Qn(H)CPU 转换为 QnUCPU 时的限制及替代方法（续）

项目		限制内容	替代方法	参照
程序	文件寄存器	不能进行各个程序的文件寄存器设置。（不能进行文件的使用方法设置。）	-----	-----
	注释文件	不能进行各个程序的注释文件设置。（不能进行文件的使用方法设置。）	-----	-----
	锁存设置	即使对软元件进行锁存，也不能锁存电源 OFF 及复位之前的软元件数据。	将要锁存的数据移至文件寄存器。	6.3 节
		如果对软元件进行锁存，扫描时间将延迟。		
	QCDSET 指令	SM721（文件访问中）为 ON 时变为无处理。	-----	-----
	SM/SD	SM/SD 的部分规格将被变更。	-----	-----
		不能使用 A 兼容 SM/SD(1000 ～ 1199)。	转换为兼容的 Q 系列的 SM/SD。	-----
驱动文件操作	引导	从标准 ROM、存储卡引导的方法将被变更。	按 QnUCPU 的参数有效驱动的思路动作。	5.2.11 项
		存储卡→标准 ROM 全部数据自动写入功能的方法将被变更。	通过传送到程序内存（快闪 ROM）中，可以用常规运行进行替代。	-----
	文件使用方法设置	不能设置文件使用方法。	-----	-----
与模块的通信	I/O 刷新	不能对各个程序的 I/O 刷新进行设置。	通过 COM 指令及 REF 指令替代。	-----
与外部设备的通信	MC 协议	不能对 QnUCPU 进行使用了 MC 协议的 A 兼容 1C/1E 帧的通信。	-----	-----
故障历史记录	故障历史记录	不能将故障历史记录存储到存储卡内的文件中。	将可存储部分全部存储到内存中。（无需存储卡）	6.18.2 项
调试功能	扫描时间测定	不能使用扫描时间测定功能。	-----	-----
	监视	不能进行梯形图监视时的监视条件设置。	-----	-----
	测试	不能执行外部 I/O 的强制 ON/OFF 功能。	-----	-----
开关	RUN/STOP 开关	用于变更 RUN、STOP 的开关变为 RUN/STOP/RESET 开关，操作方法发生了变化。	-----	-----
	RESET/L.CLR 开关	用于变更 RESET/L.CLR 的开关变为 RUN/STOP/RESET 开关，操作方法发生了变化。	-----	-----
	系统保护	没有系统保护开关功能。	-----	-----
	参数有效驱动器	参数有效驱动器的思路发生了变化。	自动检测存储参数文件的驱动器。	5.2.11 项
其它功能	LED 控制功能	不能进行 LED 显示优先顺序指定。	-----	-----

(2) 在通用型 QCPU 中不能使用的指令

在通用型 QCPU 中不能使用的指令如附表 .45 所示。

附表 .45 通用型 QCPU 中不能使用的指令

指令符号	指令名称	替代指令
IX	固定变址修饰指令	-----
IXEND		
IXDEV		
IXSET		
PR	ASCII 码打印指令	-----
PRC	注释打印指令	-----
CHKST	特定格式故障确认指令	-----
CHK		
CHKCIR	检查指令的检查格式变更指令	-----
CHKEND		
PLOW	程序低速执行登录指令	-----
LDPCHK	程序执行状态确认指令	-----
ANDPCHK		
ORPCHK		
LD/LDI/AND/ANI/OR/ORI	强制转移确认指令 (SFC 控制指令)	-----
S(P).SFCSCOMR	SFC 步注释读取指令	-----
S(P).SFCTCOMR	SFC 转移条件注释的读取指令	-----
S(P).DDWR *1	至他号机 CPU 模块的软元件写入	D(P).DDWR *2
S(P).DDRD *1	来自于他号机 CPU 模块的软元件读取	D(P).DDRD *2
S(P).SFCS *1	运动 SFC 程序的启动请求	D(P).SFCS *2
S(P).SVST *1	伺服程序的启动请求	D(P).SVST *2
S(P).CHGA *1	停止轴 / 同步编码器 / 凸轮轴的当前值变更	D(P).CHGA *2
S(P).CHGV *1	定位中以及 JOG 运行中的轴速度变更	D(P).CHGV *2
S(P).CHGT *1	实时模式时，运行 / 停止中的转矩限制值变更	D(P).CHGT *2
S(P).GINT *1	他号机 CPU 的中断程序的启动请求	D(P).GINT *2
KEY	数字键输入指令	-----
PLOADP	从存储卡的程序安装	-----
PUNLOADP	从存储卡的程序卸载	-----
PSWAPP	安装 + 卸载	-----

*1: 在 Q02UCPU 中可以使用。
在 Q03UDCPU、Q04UDHCPU、Q06UDHCPU 中，请使用各替代命令。
*2: 在 Q02UCPU 中不能使用。

附录 6 电池运输时的注意事项

含有锂的电池在运输时需要根据运输限制做出适当的处理。

(1) 限制对象机型

Q 系列 CPU 模块（含存储卡）中使用的电池如附表 .46 所示进行分类。

附表 .46 运输限制对象机型一览表

产品名称	型号	产品形态	运输处理
电池	Q8BAT	锂电池单体（电池组）	危险物
电池	Q8BAT-SET	锂电池单体（电池组）+ Q8BAT 连接电缆	
电池	Q7BAT	锂电池单体	
电池	Q7BAT-SET	锂电池单体 + 电池盒	
电池	Q6BAT	锂电池单体	非危险物
SRAM 卡用电池	Q2MEM-BAT Q3MEM-BAT	纽扣锂电池单体	
存储卡	Q2MEM-1MBS Q2MEM-2MBS	附带纽扣锂电池 (Q2MEM-BAT)	
	Q3MEM-4MBS Q3MEM-8MBS	附带纽扣锂电池 (Q3MEM-BAT)	
	Q3MEM-4MBS-SET Q3MEM-8MBS-SET	附带纽扣锂电池 (Q3MEM-BAT)+ 存储卡保护盖	

(2) 运输时的处理

交货时本公司已经根据运输规定进行了装箱，客户重新装箱或开箱之后运输时，请严格遵守 IATA Dangerous Goods Regulations(IATA 危险物规则书），IMDG Code（国际海上危险物运输规程）以及各国的运输规定。

详细内容请与您利用的运输方进行确认。

附录 7 软元件点数分配表

(1) 基本模式 QCPU 用

附表 .47 软元件点数分配表（基本模式 QCPU 用）

软元件名称	符号	进制	软元件点数 *1*2		限制确认	
			点数	编号	容量（字）*3	位点数 *2
输入继电器	X	16	2k(2048) 点	X0000~07FF	÷16 128 字	×1 2048 点
输出继电器	Y	16	2k(2048) 点	Y0000~07FF	÷16 128 字	×1 2048 点
内部继电器	M	10	k( ) 点	M0~	÷16 字	×1 点
锁存继电器	L	10	k( ) 点	L0~	÷16 字	×1 点
链接继电器	B	16	k( ) 点	B0000~	÷16 字	×1 点
报警器	F	10	k( ) 点	F0~	÷16 字	×1 点
链接特殊继电器	SB	16	1k(1024) 点	SB0000~03FF	÷16 64 字	×1 1024 点
变址继电器	V	10	k( ) 点	V0~	÷16 字	×1 点
步进继电器	S	10	2k(2048) 点	S0~2047	÷16 128 字	×1 2048 点
定时器	T	10	k( ) 点	T0~	$\times\frac{18}{16}$ 字	×2 点
累计定时器	ST	10	k( ) 点	ST0~	$\times\frac{18}{16}$ 字	×2 点
计数器	C	10	k( ) 点	C0~	$\times\frac{18}{16}$ 字	×2 点
数据寄存器	D	10	k( ) 点	D0~	×1 字	—
链接寄存器	W	16	k( ) 点	W0000~	×1 字	—
链接特殊寄存器	SW	16	1k(1024) 点	SW0000~03FF	×1 1024 字	—
软元件合计					字 (16704 字以下)	点

*1 ：□的点数在系统中为固定值。（禁止变更）
*2 ：1 个软元件的最大点数为 32k 点。
*3 ：该处是填写软元件点数乘以（或除以）容量（字）栏内数字所得数值。

(2) 高性能模式 QCPU，过程 CPU，冗余 CPYU 用

附表 .48 软元件点数分配表（高性能模式 QCPU，过程 CPU，冗余 CPYU 用）

软元件名称	符号	进制	软元件点数 *1*2		限制检查	
			点数	编号	容量（字）*3	位点数 *2
输入继电器	X	16	8k(8192) 点	X0000~1FFF	÷16 512 字	×1 8192 点
输出继电器	Y	16	8k(8192) 点	Y0000~1FFF	÷16 512 字	×1 8192 点
内部继电器	M	10	k( ) 点	M0~	÷16 字	×1 点
锁存继电器	L	10	k( ) 点	L0~	÷16 字	×1 点
链接继电器	B	16	k( ) 点	B0000~	÷16 字	×1 点
报警器	F	10	k( ) 点	F0~	÷16 字	×1 点
链接特殊继电器	SB	16	2k(2048) 点	SB0000~07FF	÷16 128 字	×1 2048 点
变址继电器	V	10	k( ) 点	V0~	÷16 字	×1 点
步进继电器	S	10	8k(8192) 点	S0~8191	÷16 512 字	×1 8192 点
定时器	T	10	k( ) 点	T0~	$\times\frac{18}{16}$ 字	×2 点
累计定时器	ST	10	k( ) 点	ST0~	$\times\frac{18}{16}$ 字	×2 点
计数器	C	10	k( ) 点	C0~	$\times\frac{18}{16}$ 字	×2 点
数据寄存器	D	10	k( ) 点	D0~	×1 字	—
链接寄存器	W	16	k( ) 点	W0000~	×1 字	—
链接特殊寄存器	SW	16	2k(2048) 点	SW0000~07FF	×1 2048 字	—
软元件合计					字 (29696 字以下)	点 (65536 点以下)

*1 ：□的点数是系统中固定的。（不能变更）
*2 ：1 个软元件的最大点数为 32k 点。
*3 ：记载软元件点数乘以（或除以）容量（字）栏中的数字后得出的数值。

(3) 通用型 QCPU 用

附表 .49 软元件点数分配表（通用型 QCPU 用）

软元件名称	符号	进制	软元件点数 *1*2		限制确认	
			点数	编号	容量（字）*3	位点数 *2
输入继电器	X	16	8k(8192) 点	X0000 ～ 1FFF	÷16 512 字	× 1 8192 点
输出继电器	Y	16	8k(8192) 点	Y0000 ～ 1FFF	÷16 512 字	× 1 8192 点
内部继电器	M	10	k( ) 点	M0 ～	÷16 字	× 1 点
锁存继电器	L	10	k( ) 点	L0 ～	÷16 字	× 1 点
链接继电器	B	16	k( ) 点	B0000 ～	÷16 字	× 1 点
报警器	F	10	k( ) 点	F0 ～	÷16 字	× 1 点
链接特殊继电器	SB	16	k( ) 点	SB0000 ～ 07FF	÷16 字	× 1 点
变址继电器	V	10	k( ) 点	V0 ～	÷16 字	× 1 点
步进继电器	S	10	8k(8192) 点	S0 ～ 8191	÷16 512 字	× 1 8192 点
定时器	T	10	k( ) 点	T0 ～	$\times\frac{18}{16}$ 字	× 2 点
累计定时器	ST	10	k( ) 点	ST0 ～	$\times\frac{18}{16}$ 字	× 2 点
计数器	C	10	k( ) 点	C0 ～	$\times\frac{18}{16}$ 字	× 2 点
数据寄存器	D	10	k( ) 点	D0 ～	× 1 字	--
链接寄存器	W	16	k( ) 点	W0000 ～	× 1 字	--
链接特殊寄存器	SW	16	k( ) 点	SW0000 ～ 07FF	× 1 字	--
软元件合计					字 (29696 字以下)	点 (65536 点以下)

*1 ：□的点数是系统中固定的。（不能变更）
*2 ：1 个软元件的最大点数为 32k 点。
*3 ：记载软元件点数乘以（或除以）容量（字）栏中的数字后得出的数值。

索引

关于质保

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责任的故障或缺陷（以下称 “故障”），则经销商或三菱服务公司将负责免费维修。
注意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货到目的地日的一年内。
注意产品从三菱生产并出货之后，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. 因不适当存储或搬运、用户粗心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技术公告等方式予以通告。
(2) 产品停产后，将不再提供产品（包括维修零件）。

3. 海外的服务

在海外，维修由三菱在当地的海外 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，对于任何非三菱责任的原因而导致的损失、机会损失、因三菱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。

6. 产品应用

(1) 在使用三菱 MELSEC 通用可编程控制器时，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保险功能。
(2) 三菱通用可编程控制器是以一般工业用途等为对象设计和制造的。因此，可编程控制器的应用不包括那些会影响公共利益的应用，如核电厂和其它由独立供电公司经营的电厂以及需要特殊质量保证的应用如铁路公司或用于公用设施目的的应用。
另外，可编程控制器的应用不包括航空、医疗应用、焚化和燃烧设备、载人设备、娱乐及休闲设施、安全装置等与人的生命财产密切相关以及在安全和控制系统方面需要特别高的可靠性时的应用。
然而，对于这些应用，假如用户咨询当地三菱代表机构，提供有特殊要求方案的大纲并提供满足特殊环境的所有细节及用户自主要求，则可以进行一些应用。

谨此

QCPU

用户手册(功能解说/程序基础篇)

技术服务热线:
800-828-9910
服务时间: **9:00~12:00**
13:00~17:00(节假日除外)

三菱电机自动化(上海)有限公司

地址: 上海市黄浦区新昌路80号智富广场4楼
邮编: 200003
电话: 021-61200808 传真: 021-61212444
网址: www.mitsubishielectric-automation.cn

书号	SH(NA)-080503CHN-C(0710)STC
印号	STC-QCPU-F-UM(0710)