MITSUBISHI

三菱可编程控制器

MELSEC Q 系列

QCPU
编程手册
公共指令篇1/2

● 安全注意事项 ●

（使用之前请务必阅读）

在使用 MELSEC-Q 系列可编程控制器之前，应仔细阅读各产品附带的手册以及附带手册中介绍的关联手册，同时在充分注意安全的前提下正确地操作。

请妥善保管产品附带手册以备需要时阅读，并应将本手册交给最终用户。

修订记录

※本手册号在封底的左下角。

印刷日期	※手册编号	修订记录
2009 年 03 月	SH(NA)-080814CHN-A	第一版

日文手册原稿： SH-080804-A

前言

本手册“QCPU 编程手册（公共指令篇）”介绍进行 QCPU 编程时需要的公共指令有关内容。

· 公共指令是指，AJ71QC24、AJ71PT32-S3 等特殊功能模块专用指令、AD57 专用指令、PID 控制专用指令、SFC 专用指令、除 ST 专用指令以外的其它指令。

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

■对象 CPU 模块

CPU 模块	型号
基本型 QCPU	Q00JCPU、Q00CPU、Q01CPU
高性能型 QCPU	Q02CPU、Q02HCPU、Q06HCPU、Q12HCPU、Q25HCPU
过程 CPU	Q02PHCPU、Q06PHCPU、Q12PHCPU、Q25PHCPU
冗余 CPU	Q12PRHCPU、Q25PRHCPU
通用型 QCPU	Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU、Q03UDCPU、Q04UDHCPU、Q06UDHCPU、Q10UDHCPU、Q13UDHCPU、Q20UDHCPU、Q26UDHCPU、Q03UDECPU、Q04UDEHCPU、Q06UDEHCPU、Q10UDEHCPU、Q13UDEHCPU、Q20UDEHCPU、Q26UDEHCPU

目录

公共指令篇 1/2

公共指令篇 2/2

手册体系

在基本手册中介绍基本的规格、功能、使用方法有关内容。
其它手册介绍相应的 CPU 模块及功能。
请根据需要参考下表订购各手册。

“ CPU 模块 ”栏中所示的编号及 CPU 模块的对应如下所示：

编号	CPU 模块
1)	基本型 QCPU
2)	高性能型 QCPU
3)	过程 CPU
4)	冗余 CPU
5)	通用型 QCPU

●： 基本手册　○： 使用相应 CPU 模块 / 功能时参阅。

手册名称 < 手册编号 >	记载内容	CPU 模块				
		1)	2)	3)	4)	5)
■用户手册						
QCPU 用户手册 (硬件设计 / 维护点检篇) <SH-080501CHN>	介绍 CPU 模块、电源模块、基板、扩展电缆及存储卡等的硬件规格及系统维护 · 点检、故障排除、出错代码等。	●	●	●	●	●
QnUCPU 用户手册 (功能解说 / 程序基础篇) <SH-080812CHN>	介绍创建程序所需的功能、编程方法和软元件等。					●
Qn(H)/QnPH/QnPRHCPU 用户手册 (功能解说 / 程序基础篇) <SH-080803>	介绍创建程序所需的功能、编程方法和软元件等。	●	●	●	●	
QCPU 用户手册 (内置以太网板通信篇) <SH-080813CHN>	介绍 CPU 内置以太网板通信的功能有关内容。					○
■编程手册						
QCPU 编程手册 (公共指令篇) <SH-080814CHN>	介绍顺控程序指令、基本指令以及应用指令等的使用方法。	●	●	●	●	●
QCPU(Q 模式)/QnACPU 编程手册 (SFC 篇) <SH-080283CHN>	介绍 MELSAP3 格式的 SFC 程序的创建所需的系统配置、性能规格、功能、编程、调试和出错代码等。	○	○	○	○	○
QCPU(Q 模式) 编程手册 (MELSAP-L 篇) <SH-080076>	介绍 MELSAP-L 格式的 SFC 程序的创建所需的编程方法、规格、功能等。	○	○	○	○	○
QCPU(Q 模式) 编程手册 (结构化文本篇) <SH-080363>	介绍结构化文本语言的编程方法。	○	○	○	○	○
QCPU(Q 模式)/QnACPU 编程手册 (PID 控制指令篇) <SH-080240C>	介绍用于执行 PID 控制的专用指令。	○	○		○	○
QnPHCPU/QnPRHCPU 编程手册 (过程控制指令篇) <SH-080449CHN>	介绍用于执行过程控制的专用指令。			○	○	

关联手册

手册名称 <手册编号>	记载内容
CC-Link IE 控制网络参考手册 <SH-080649>	介绍 CC-Link IE 控制网络的控制网络规格、投行前的设置及步骤、参数设置、编程及故障排除等。
Q 系列 MELSECNET/H 网络系统参考手册（可编程控制器网络篇） <SH-080289C>	介绍 MELSECNET/H 网络系统的可编程控制器网络的规格、投运前的设置及步骤、参数设置、编程及故障排除等。
Q 系列 MELSECNET/H 网络系统参考手册（远程 I/O 网络篇） <SH-080290C>	介绍 MELSECNET/H 网络系统的远程 I/O 网络的规格、投运前的设置及步骤、参数设置、编程及故障排除等。
MELSECNET、MELSECNET/B 数据链接系统参考手册 <IB-68277>	介绍 MELSECNET(II) 和 MELSECNET/B 的概要、规格和各部位名称及设置等。
Q 系列以太网接口模块用户手册（应用篇） <SH-080285C>	介绍以太网模块的电子邮件功能、可编程控制器 CPU 的状态监视、经由 LSECNET/H 或 MELSECNET/10 网络的通信功能、通过数据链接用指令的通信功能、文件传输 (FTP 服务器）等功能。

备忘录

1 概述

本手册介绍进行 QCPU 编程时所需的公共指令有关内容。

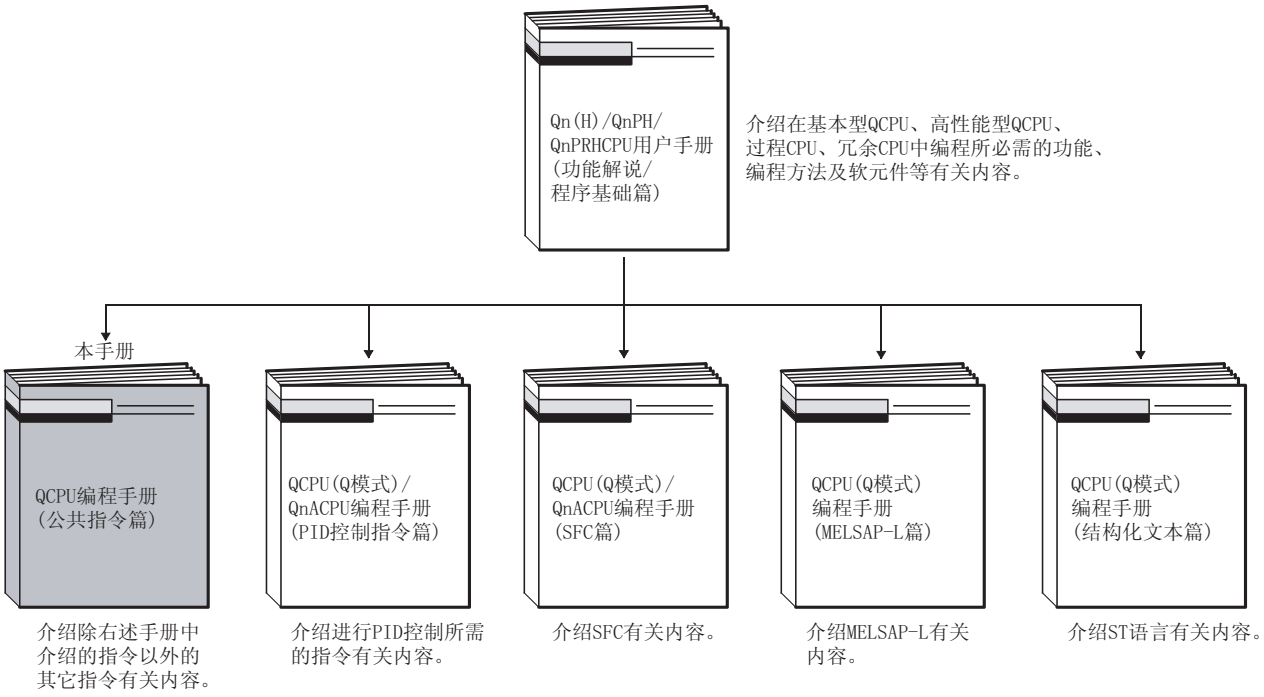
公共指令是除了 AJ71QC24 和 AJ71PT32-S3 等的特殊功能模块用指令、AD57 用指令、PID 控制用指令、SFC 用指令和 ST 用指令之外的其它所有指令。

1.1 相关编程手册

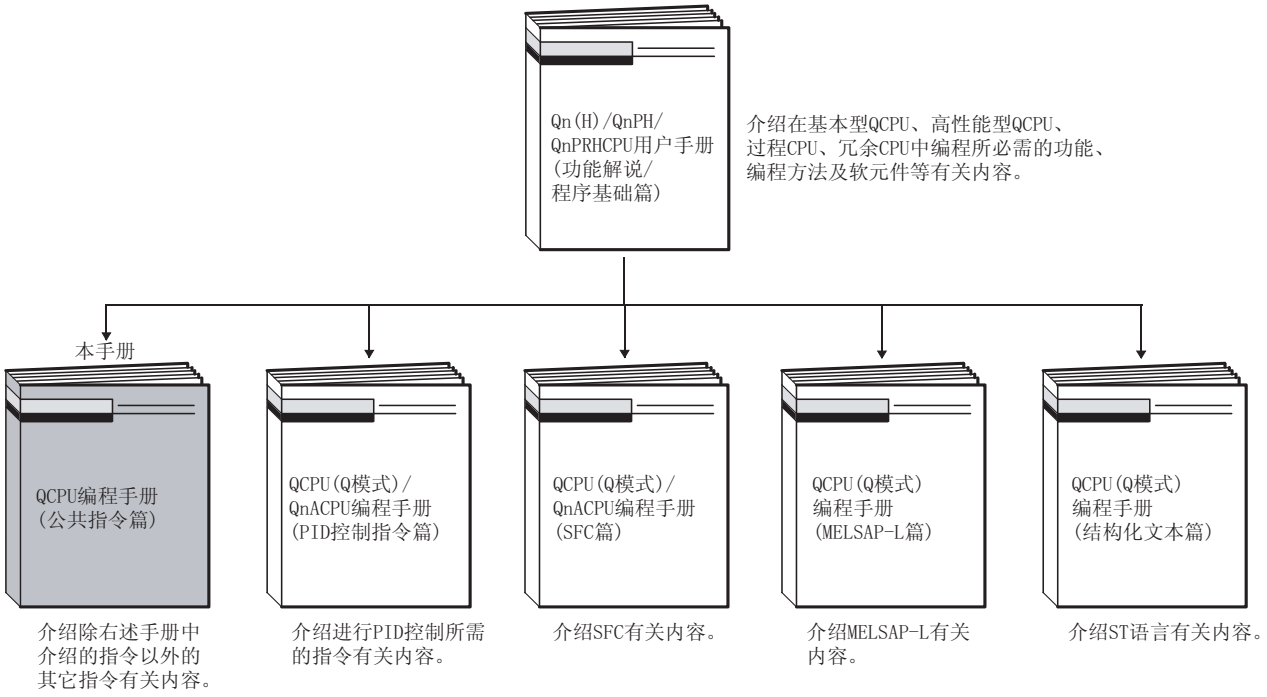
在阅读本手册之前，关于 CPU 模块编程所必需的功能、编程方法、软元件等，请参阅下列手册：

· QnUCPU 用户手册（功能解说 / 程序基础篇）
· Qn(H)/QnPH/QnPRHCPU 编程手册（功能解说 / 程序基础篇）

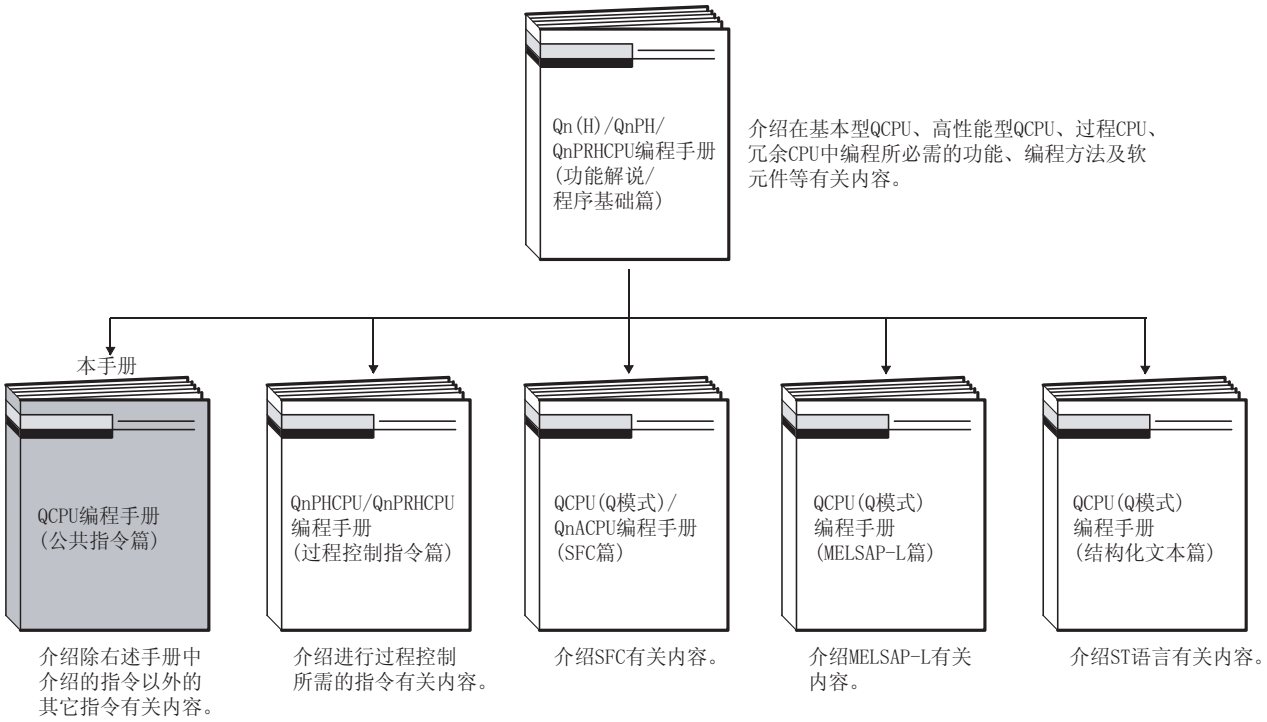
(1) 使用高性能型 QCPU 时



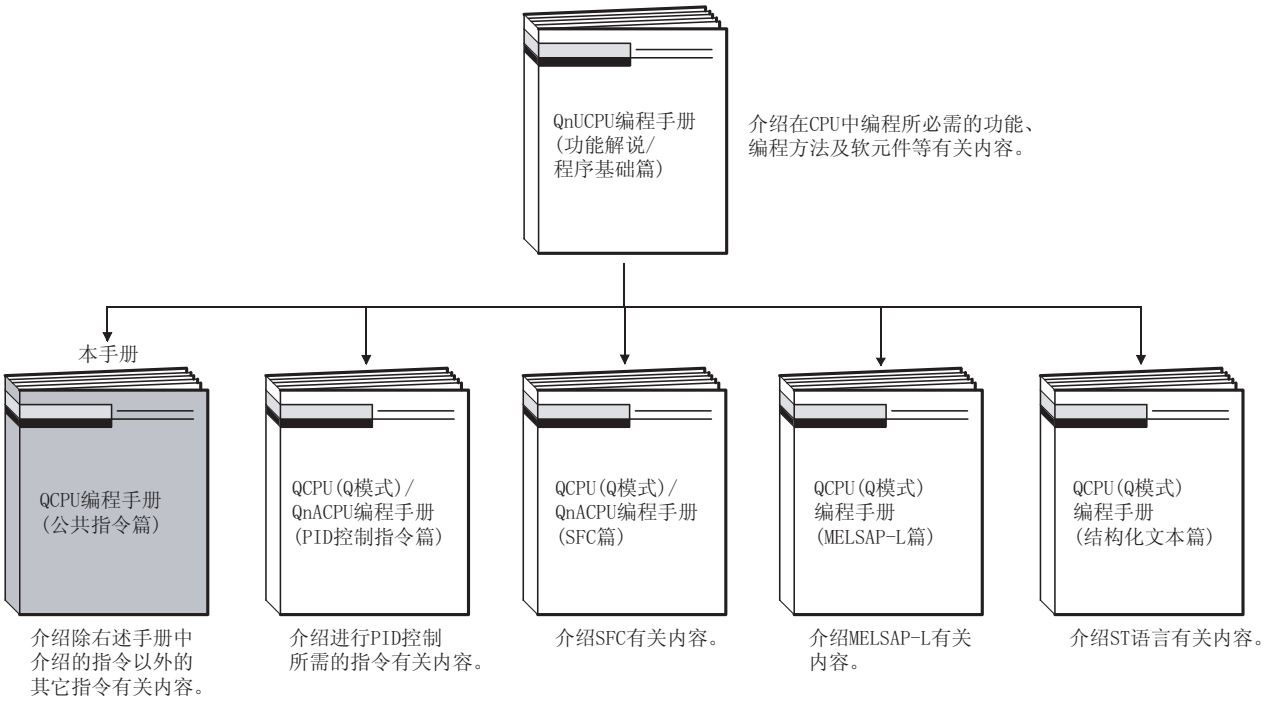
(2) 使用基本型 QCPU 时



(3) 使用过程 CPU、冗余 CPU 时



(4) 使用通用型 QCPU 时



1.2 本手册中使用的总称 / 略称

本手册中除特别注明以外，将使用如下表所示的总称和略称来介绍 Q 系列 CPU 模块的有关内容。

* □表示多个型号及版本等的总称时的可变部分。

总称 / 略称	总称 / 略称的内容
■系列名称	
Q 系列	三菱通用可编程控制器 MELSEC-Q 系列的略称。
■ CPU 模块的类型名称	
CPU 模块	基本型 QCPU、高性能型 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 的总称。
基本型 QCPU	Q00JCPU、Q00CPU 和 Q01CPU 的总称。
高性能型 QCPU	Q02CPU、Q02HCPU、Q06HCPU、Q12HCPU 和 Q25HCPU 的总称。
过程 CPU	Q02PHCPU、Q06PHCPU、Q12PHCPU 和 Q25PHCPU 的总称。
冗余 CPU	Q12PRHCPU 和 Q25PRHCPU 的总称。
通用型 QCPU	Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU、Q03UDCPU、Q04UDHCPU、Q06UDHCPU、Q10UDHCPU、Q13UDHCPU、Q20UDHCPU、Q26UDHCPU、Q03UDECPU、Q04UDEHCPU、Q06UDEHCPU、Q10UDEHCPU、Q13UDEHCPU、Q20UDEHCPU、Q26UDEHCPU 的总称。
■ CPU 模块的型号	
QnCPU	Q00JCPU、Q00CPU、Q01CPU 和 Q02CPU 的总称。
QnHCPU	Q02HCPU、Q06HCPU、Q12HCPU 和 Q25HCPU 的总称。
QnPHCPU	Q02PHCPU、Q06PHCPU、Q12PHCPU、Q25PHCPU 的总称。
QnPRHCPU	Q12PRHCPU 和 Q25PRHCPU 的总称。
QnUCPU	Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU、Q03UDCPU、Q04UDHCPU、Q06UDHCPU、Q10UDHCPU、Q13UDHCPU、Q20UDHCPU、Q26UDHCPU、Q03UDECPU、Q04UDEHCPU、Q06UDEHCPU、Q10UDEHCPU、Q13UDEHCPU、Q20UDEHCPU、Q26UDEHCPU 的总称。
QnU(D)(H)CPU	Q02UCPU、Q03UDCPU、Q04UDHCPU、Q06UDHCPU、Q10UDHCPU、Q13UDHCPU、Q20UDHCPU、Q26UDHCPU 的总称。
QnUD(H)CPU	Q03UDCPU、Q04UDHCPU、Q06UDHCPU、Q10UDHCPU、Q13UDHCPU、Q20UDHCPU、Q26UDHCPU 的总称。
QnUDE(H)CPU	Q03UDECPU、Q04UDEHCPU、Q06UDEHCPU、Q10UDEHCPU、Q13UDEHCPU、Q20UDEHCPU、Q26UDEHCPU 的总称。

（续）

总称 / 略称	总称 / 略称的内容
■其它	
GX Developer	Q 系列兼容 SW □ D5C-GPPW 型 GPP 功能软件包的产品名。 □表示软件版本。 关于可适用于各 CPU 模块的 GX Developer 的版本，请参阅 QCPU 用户手册（硬件设计 / 维护点检篇）的“ 系统配置 ”。
MELSECNET/H	MELSECNET/H 网络系统的略称。
MELSECNET/10	MELSECNET/10 网络系统的略称。
MELSECNET(II、/B)	MELSECNET 和 MELSECNET/B 数据链接系统的略称。
以太网	以太网网络系统的略称。
CC-Link	Control & Communication Link 的略称。
智能功能模块	智能功能模块和特殊功能模块的总称。
智能功能模块软元件	智能功能模块软元件和特殊功能模块软元件的总称。
Q3 □ B	可安装 CPU 模块 (Q00JCPU 除外)、Q 系列电源模块、Q 系列输入输出模块、智能功能模块的 Q33B、Q35B、Q38B、Q312B 型主基板的总称。
Q3 □ SB	可安装基本型 QCPU(Q00JCPU 除外)、高性能型 QCPU、通用型 QCPU、超薄型电源模块、Q 系列输入输出模块、智能功能模块的 Q32SB、Q33SB、Q35SB 型超薄型主基板的总称。
Q3 □ RB	可安装 CPU 模块 (Q00JCPU 除外)、冗余电源模块、Q 系列输入输出模块、智能功能模块的 Q38RB 型电源冗余用主基板的别称。
Q3 □ DB	可安装 CPU 模块 (Q00JCPU 除外)、Q 系列电源模块、Q 系列输入输出模块、智能功能模块的 Q38DB、Q312DB 型多 CPU 间高速主基板的总称。
Q5 □ B	可安装 Q 系列输入输出模块、智能功能模块的 Q52B、Q55B 型扩展基板的总称。
Q6 □ B	可安装 Q 系列电源模块、Q 系列输入输出模块、智能功能模块的 Q63B、Q65B、Q68B、Q612B 型扩展基板的总称。
Q6 □ RB	可安装冗余电源模块、Q 系列输入输出模块、智能功能模块的 Q68RB 型电源冗余用扩展基板的别称。
Q6 □ WRB	可安装冗余电源模块、Q 系列输入输出模块、智能功能模块的 Q65WRB 型冗余扩展基板的别称。
QA1S6 □ B	可安装 AnS 系列电源模块、AnS 系列输入输出模块、特殊功能模块的 QA1S65B、QA1S68B 型扩展基板的总称。
QA6 □ B	可安装 A 系列电源模块、A 系列输入输出模块、特殊功能模块的 QA65B、QA68B 型扩展基板的总称。
A5 □ B	可安装 A 系列输入输出模块、特殊功能模块的无需电源型的 A52B、A55B、A58B 型扩展基板的总称。
A6 □ B	可安装 A 系列输入输出模块、特殊功能模块的 A62B、A65B、A68B 型扩展基板的总称。
QA6ADP	QA6ADP 型 QA 转换适配器模块的略称。
QA6ADP+A5 □ B/A6 □ B	安装了 QA6ADP 的 A 大型扩展基板的略称。

2 指令一览表

2.1 指令分类

CPU 模块指令的主要类型包括顺控程序指令、基本指令、应用指令、数据链接指令、QCPU 用指令和冗余系统用指令。指令的分类如表 2.1 所示。

表 2.1 指令分类

指令分类		内容	参阅章节
顺控程序指令	触点指令	运算开始、串行连接、并行连接。	第 5 章
	连接指令	梯形图块的连接、运算结果的记忆·读取、运算结果的脉冲化。	
	输出指令	位软元件的输出、脉冲输出、输出取反。	
	移位指令	位软元件的移位	
	主控制指令	主控制	
	结束指令	结束程序	
	其它指令	程序停止、无处理等未列入上述分类中的指令。	
基本指令	比较运算指令	=、>、<等的比较。	第 6 章
	算术运算指令	BIN、BCD 的加法、减法、乘法或除法。	
	BCD 与 BIN 转换指令	将 BCD 转换成 BIN 以及将 BIN 转换成 BCD。	
	数据转移指令	传送指定的数据。	
	程序分支指令	程序跳转	
	程序的执行控制指令	中断程序的允许 / 禁止	
	I/O 刷新指令	部分刷新的执行	
	其它使用方便的指令	用于以下目的的指令： 计数器增加 / 减小、教学定时器、特殊功能定时器、旋转台就近控制等指令。	
应用指令	逻辑运算指令	逻辑和、逻辑积等的逻辑运算。	第 7 章
	旋转指令	指定数据的旋转	
	移位指令	指定数据的移位	
	位处理指令	位设置 / 复位、位测试、位软元件的批量复位。	
	数据处理指令	16 位数据的查找、解码和编码等数据处理。	
	结构化指令	重复运算、子程序调用、梯形图单位的变址修饰。	
	表操作指令	数据表的读 / 写	
	缓冲存储器访问指令	智能功能模块的数据读 / 写	
	显示指令	ASCII 码的打印、字符的 LED 显示等。	
	调试·故障诊断指令	检查、状态获取、采样跟踪、程序跟踪。	
	字符串处理指令	BIN/BCD 与 ASCII 之间的转换、 BIN 与字符串之间的转换、浮点数据与字符串之间的转换、字符串处理等。	
	特殊函数指令	三角函数、角度和弧度之间的转换、指数运算、自然对数、常用对数、方根运算。	
	数据控制指令	上下限控制、死区控制、区域控制、标度。	
	切换指令	文件寄存器的块号切换、文件寄存器 / 注释文件的指定。	
	时钟指令	年、月、日、小时、分钟、秒和星期的读 / 写；时、分、秒→秒的变换。	
	外围设备用指令	至外围设备的输入输出。	
	程序控制用指令	用于切换程序执行条件的指令	
	其它指令	其它未列入上述分类的指令，如看门狗定时器复位指令和定时时钟指令等。	

表 2.1 指令分类（续）

指令分类		内容	参阅章节
数据链接用指令	链接刷新用指令	指定网络的刷新	第 8 章
	路由信息读取 / 写入指令	路由信息的读取 / 写入	
QCPU 用指令	用于 QCPU 的指令	模块信息读取、跟踪设置 / 复位、二进制数据的读取 / 写入、标准 ROM 数据的读取 / 写入、通过存储卡进行的程序安装 / 卸载 / 安装 + 卸载、文件寄存器高速块传送、自站 CPU 共享存储器的写入、其它站 CPU 共享存储器的读取、选择刷新指令和扩展时钟用指令。	第 9 章
冗余系统指令	用于冗余 CPU 的指令	系统切换	第 10 章
多 CPU 高速通信专用指令	多 CPU 间软元件写入 / 读取指令	至其它站 CPU 的软元件写入、从其它站的软元件读取。	第 11 章

2.2 指令一览表的阅读方法

2.3 节至 2.5 节的指令一览表的形式如下所示：

表 2.2 指令一览表的阅读方法

分 类	指令符号	符 号	处理内容	执行条件	基本步数	子集	参阅页面
BIN16位加法减法运算	+	+ S D	• (D)+(S)→(D)		3	●	6-16
	+P	+P S D					
	+	+ S1 S2 D	• (S1)+(S2)→(D)		4	●	6-20
	+P	+P S1 S2 D					

① ② ③ ④ ⑤ ⑥ ⑦ ⑧

说明

1) ..... 将指令按用途进行分类。

2) ..... 表示程序中使用的指令符号。
指令符号是以 16 位指令为基准。
32 位指令、仅在 OFF → ON 的上升沿时执行的指令、实数指令、字符串指令时的情况如下所示：

· 32 位指令 ..... 在指令的起始附加 D。

示例 + → D+
（+：16位指令；D+：32位指令）

· 仅在 OFF → ON 的上升沿时执行的指令 ..... 在指令的末尾附加 P。

示例 + → +P
（+：ON状态下执行的指令；+P：仅在OFF→ON的上升沿时执行的指令）

· 实数指令 ..... 在指令的起始附加 E。

示例 + → E+
（E+：实数指令）

· 字符串指令 ..... 在指令的起始附加 $。

示例 + → $+
（$+：字符串指令）

3) ……表示在梯形图上的符号图。

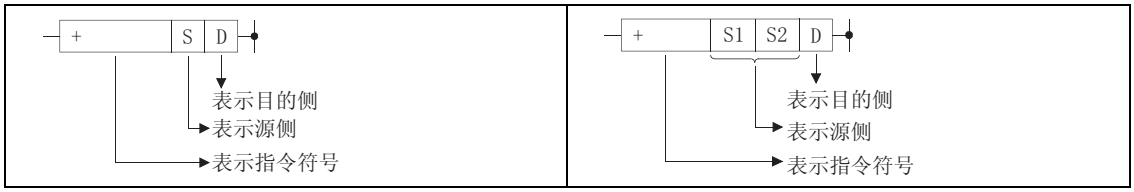


图 2.1 梯形图上的符号图

目的 (destination)..... 表示运算后的数据去向。

源 (source)..... 存储运算前的数据。

4) ……表示各指令的处理内容。

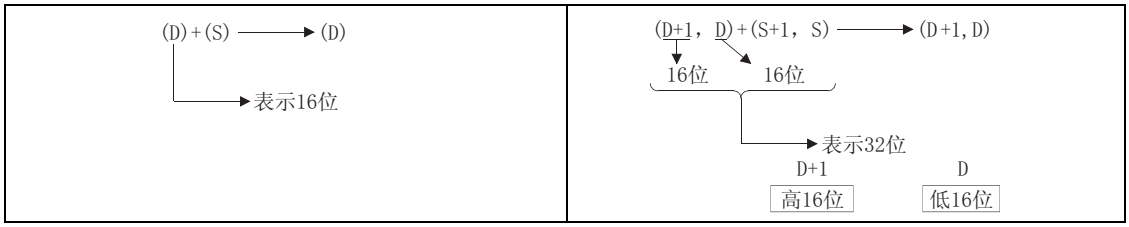
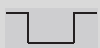


图 2.2 各指令的处理内容

5) ……各指令的执行条件的详细内容如下所示：

符号	执行条件
未记入	是常时执行的指令，与指令的前条件的 ON/OFF 无关，一直执行。 在前条件为 OFF 的情况下，该指令执行 OFF 处理。
	是在 ON 状态下执行的指令，仅在指令的前条件为 ON 的期间执行该指令。前条件为 OFF 时，不执行该指令，不进行处理。
	是在 ON 时仅执行 1 次的指令，仅在指令的前条件的上升沿时 (OFF → ON) 执行指令。 以后即使条件变为 ON 也不执行该指令，不进行处理。
	是在 OFF 状态下执行的指令，仅在指令的前条件为 OFF 的期间执行该指令。前条件为 ON 时，不执行该指令，不进行处理。
	是在 OFF 时仅执行 1 次的指令，仅在指令的前条件的下降沿时 (OFF → ON) 执行指令。 以后即使条件变为 OFF 也不执行该指令，不进行处理。

6)..... 表示各指令的基本步数。

关于步数的内容，请参阅 3.8 节。

7) ……●符号表示包含有可进行子集处理的指令。

关于子集处理的详细内容，请参阅 3.5 节。

8) ……表示说明各指令的页面。

2.3 顺控程序指令

2.3.1 触点指令

表 2.3 触点指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
触点	LD		· 逻辑运算开始 (a 触点逻辑运算开始)		*1	●	5-2
	LDI		· 逻辑否运算开始 (b 触点逻辑运算开始)				
	AND		· 逻辑积 (a 触点串行连接)				
	ANI		· 逻辑积否定 (b 触点串行连接)				
	OR		· 逻辑和 (a 触点并行连接)				
	ORI		· 逻辑和否定 (b 触点并行连接)				
	LDP		· 上升脉冲运算开始		*2	●	5-5
	LDF		· 下降脉冲运算开始				
	ANDP		· 上升脉冲串行连接				
	ANDF		· 下降脉冲串行连接				
	ORP		· 上升脉冲并行连接				
	ORF		· 下降脉冲并行连接				
	LDPI		· 上升脉冲否定运算开始		3*3	●	5-7
	LDFI		· 下降脉冲否定运算开始		3*3		
	ANDPI		· 上升脉冲否定串行连接		4*3		
	ANDFI		· 下降脉冲否定串行连接		4*3		
	ORPI		· 上升脉冲否定并行连接		4*3		
	ORFI		· 下降脉冲否定并行连接		4*3		

*1: 步数根据所使用的软元件而有所不同。

使用软元件	步数
使用内部软元件、文件寄存器 (R0 至 R32767) 时	1
使用直接访问输入 (DX) 时	2
使用除上述以外的软元件时	3

*2: 步数根据所使用的软元件以及 CPU 模块类型而有所不同。

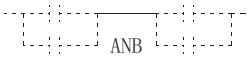
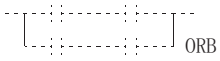
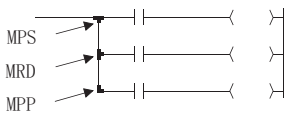
使用软元件	步数
使用内部软元件、文件寄存器 (R0 至 R32767) 时	1
使用直接访问输入 (DX) 时	2
使用除上述以外的软元件时	3

步数根据所使用的软元件而有所不同。

使用软元件	步数
使用内部软元件、文件寄存器 (R0 至 R32767) 时	基本步数
使用连号访问方式文件寄存器 (ZR)、扩展数据寄存器 (D)、扩展链接寄存器 (W)、多 CPU 共享软元件 ((U3En\G10000 ～) 时	基本步数 +1
使用直接访问输入 (DX) 时	基本步数 +1
使用除上述以外的软元件时	基本步数 +2

2.3.2 连接指令

表 2.4 连接指令

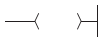
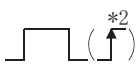
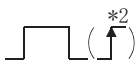
分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
连接	ANB		· 逻辑块之间的 AND(逻辑块之间的串行连接)		1	-	5-10
	ORB		· 逻辑块之间的 OR(逻辑块之间的串行连接)				
	MPS		· 运算结果的记忆		1	-	5-12
	MRD		· 通过 MPS 记忆的运算结果的读取				
	MPP		· 通过 MPS 记忆的运算结果的读取记复位				
	INV		· 运算结果的取反		1	-	5-15
	MEP		· 运算结果上升脉冲化		1	-	5-17
	MEF		· 运算结果下降脉冲化				
	EGP		· 运算结果上升脉冲化 (通过 Vn 记忆)		1	-	5-18
	EGF		· 运算结果下降脉冲化 (通过 Vn 记忆)		*1		

*1: 步数根据所使用的软元件以及 CPU 模块类型而有所不同。

CPU 类型	基本步数
高性能型 QCPU 过程 CPU 冗余 CPU 通用型 QCPU	1
基本型 QCPU	2

2.3.3 输出指令

表 2.5 输出指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
输出	OUT		· 软元件的输出		*1	-	5-20 5-22 5-26 5-28
	SET		· 软元件的设置		*1	-	5-30 5-35
	RST		· 软元件的复位		*1	-	5-32 5-35
	PLS		· 输入信号的上升沿时产生 1 个程序周期的脉冲。		2	-	5-37
	PLF		· 输入信号的下降沿时产生 1 个程序周期的脉冲。				
	FF		· 软元件输出的取反		2	-	5-40
	DELTA		· 直接输出的脉冲化		2	-	5-42
	DELTAP						

*1: 步数根据所使用的软元件而有所不同。
关于步数的内容，请参阅各指令的参阅页面。

*2: 仅在使用报警器 (F) 时变为  。

2.3.4 移位指令

表 2.6 移位指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
移位	SFT		· 软元件的 1 位移动		2	-	5-44
	SFTP						

2.3.5 主控指令

表 2.7 主控指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
主控	MC	MC n D	· 主控开始		2	-	5-46
	MCR	MCR n	· 主控解除		1		

2.3.6 结束指令

表 2.8 结束指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
结束	FEND	FEND	· 结束主程序		1	-	5-50
	END	END	· 结束顺控程序				5-52

2.3.7 其它指令

表 2.9 其它指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
停止	STOP	STOP	· 输入条件成立后，停止顺控程序的运算。 · 如果将 RUN/STOP(键) 开关再次置于 RUN，将执行顺控程序。		1	-	5-54
无处理	NOP	----------	· 无处理 (用于程序的檫除或者空格)		1	-	5-56
	NOPLF	NOPLF	· 无处理 (用于打印输出时的页面更改)				
	PAGE	PAGE n	· 无处理 (将后面的程序作为第 n 页的 0 步开始进行管理)				

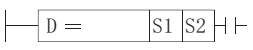
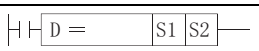
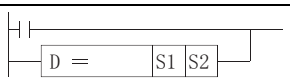
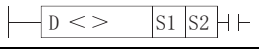
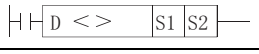
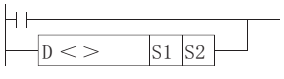
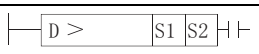
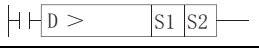
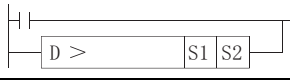
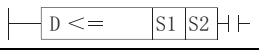
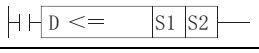
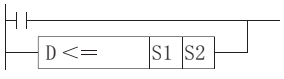
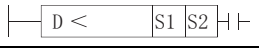
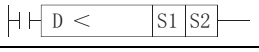
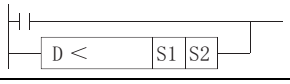
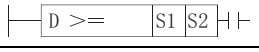
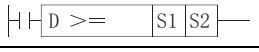
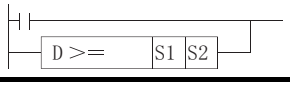
2.4 基本指令

2.4.1 比较运算指令

表 2.10 比较运算指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
BIN16 位数据比较	LD=	┤ = S1 S2 ├	· 当 (S1) = (S2) 时处于导通状态 · 当 (S1) ≠ (S2) 时处于不导通状态		3	●	6-2
	AND=	├ = S1 S2 ─					
	OR=	┤ ├ └ = S1 S2 ┘					
	LD<>	┤ <> S1 S2 ├	· 当 (S1) ≠ (S2) 时处于导通状态 · 当 (S1) = (S2) 时处于不导通状态		3	●	
	AND<>	├ <> S1 S2 ─					
	OR<>	┤ ├ └ <> S1 S2 ┘					
	LD>	┤ > S1 S2 ├	· 当 (S1) > (S2) 时处于导通状态 · 当 (S1) ≦ (S2) 时处于不导通状态		3	●	
	AND>	├ > S1 S2 ─					
	OR>	┤ ├ └ > S1 S2 ┘					
	LD<=	┤ <= S1 S2 ├	· 当 (S1) ≦ (S2) 时处于导通状态 · 当 (S1) > (S2) 时处于不导通状态		3	●	
	AND<=	├ <= S1 S2 ─					
	OR<=	┤ ├ └ <= S1 S2 ┘					
	LD<	┤ < S1 S2 ├	· 当 (S1) < (S2) 时处于导通状态 · 当 (S1) ≧ (S2) 时处于不导通状态		3	●	
	AND<	├ < S1 S2 ─					
	OR<	┤ ├ └ < S1 S2 ┘					
	LD>=	┤ >= S1 S2 ├	· 当 (S1) ≧ (S2) 时处于导通状态 · 当 (S1) < (S2) 时处于不导通状态		3	●	
	AND>=	├ >= S1 S2 ─					
	OR>=	┤ ├ └ >= S1 S2 ┘					

表 2.10 比较运算指令（续）

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
BIN32 位数据比较	LDD=		· 当 (S1+1,S1)＝(S2+1,S2) 时处于导通状态 · 当 (S1+1,S1)≠(S2+1,S2) 时处于不导通状态		*1	●	6-4
	ANDD=						
	ORD=						
	LDD<>		· 当 (S1+1,S1)≠(S2+1,S2) 时处于导通状态 · 当 (S1+1,S1)＝(S2+1,S2) 时处于不导通状态		*1	●	
	ANDD<>						
	ORD<>						
	LDD>		· 当 (S1+1,S1)＞(S2+1,S2) 时处于导通状态 · 当 (S1+1,S1)≦(S2+1,S2) 时处于不导通状态		*1	●	
	ANDD>						
	ORD>						
	LDD<=		· 当 (S1+1,S1)≦(S2+1,S2) 时处于导通状态 · 当 (S1+1,S1)＞(S2+1,S2) 时处于不导通状态		*1	●	
	ANDD<=						
	ORD<=						
	LDD<		· 当 (S1+1,S1)＜(S2+1,S2) 时处于导通状态 · 当 (S1+1,S1)≧(S2+1,S2) 时处于不导通状态		*1	●	
	ANDD<						
	ORD<						
	LDD>=		· 当 (S1+1,S1)≧(S2+1,S2) 时处于导通状态 · 当 (S1+1,S1)＜(S2+1,S2) 时处于不导通状态		*1	●	
	ANDD>=						
	ORD>=						

*1: 步数根据所使用的软元件以及 CPU 类型而有所不同。

CPU 类型	使用软元件	步数
高性能型 QCPU 过程 CPU 冗余 CPU	· 字软元件： 内部软元件（但是文件寄存器 ZR 除外） · 位软元件： 软元件号为 16 的倍数，位数指定为 K8，无变址修饰。 · 常数： 无限制	5 注 1)
	使用除上述以外的软元件时。	3 注 2)
基本型 QCPU 通用型 QCPU	可使用的所有软元件	3 注 2)

注 1) 在高性能型 QCPU、过程 CPU、冗余 CPU 的情况下，步数将会增加，但处理速度将会变快。

注 2) 根据 3.8 节的条件步数有可能会增加。

表 2.10 比较运算指令（续）

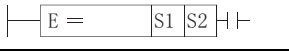
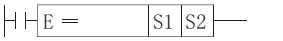
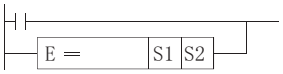
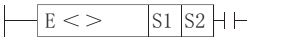
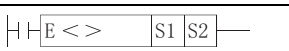
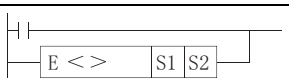
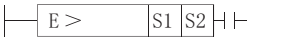
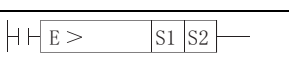
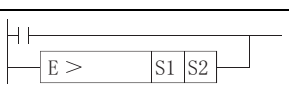
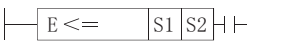
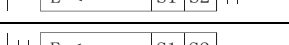
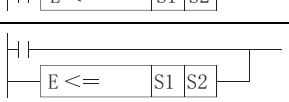
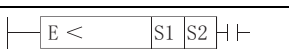
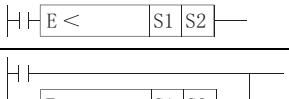
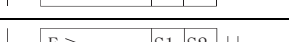
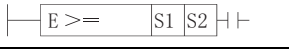
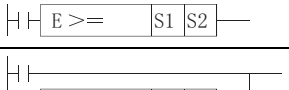
分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
浮点数据比较（单精度）	LDE=		· 当 (S1+1,S1) = (S2+1,S2) 时处于导通状态 · 当 (S1+1,S1) ≠ (S2+1,S2) 时处于不导通状态		3	-	6-6
	ANDE=						
	ORE=						
	LDE<>		· 当 (S1+1,S1) ≠ (S2+1,S2) 时处于导通状态 · 当 (S1+1,S1) = (S2+1,S2) 时处于不导通状态		3	-	
	ANDE<>						
	ORE<>						
	LDE>		· 当 (S1+1,S1) > (S2+1,S2) 时处于导通状态 · 当 (S1+1,S1) ≦ (S2+1,S2) 时处于不导通状态		3	-	
	ANDE>						
	ORE>						
	LDE<=		· 当 (S1+1,S1) ≦ (S2+1,S2) 时处于导通状态 · 当 (S1+1,S1) > (S2+1,S2) 时处于不导通状态		3	-	
	ANDE<=						
	ORE<=						
	LDE<		· 当 (S1+1,S1) < (S2+1,S2) 时处于导通状态 · 当 (S1+1,S1) ≧ (S2+1,S2) 时处于不导通状态		3	-	
	ANDE<						
	ORE<						
	LDE>=		· 当 (S1+1,S1) ≧ (S2+1,S2) 时处于导通状态 · 当 (S1+1,S1) < (S2+1,S2) 时处于不导通状态		3	-	
	ANDE>=						
	ORE>=						

表 2.10 比较运算指令（续）

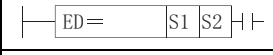
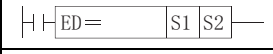
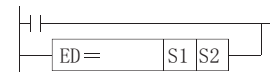
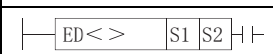
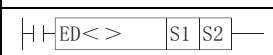
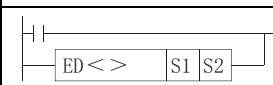
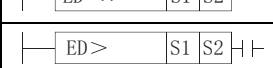
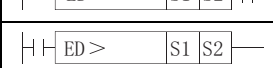
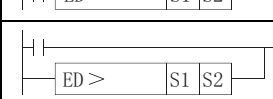
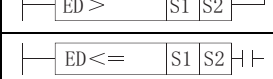
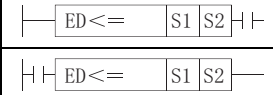
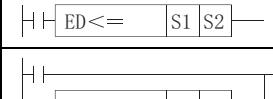
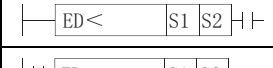
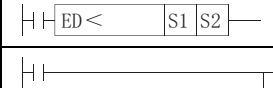
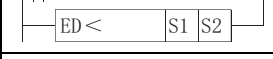
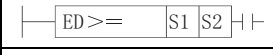
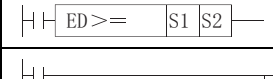
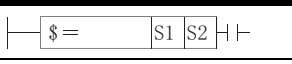
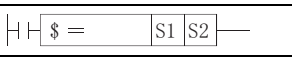
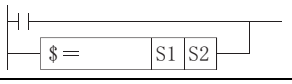
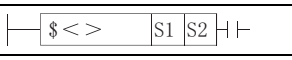
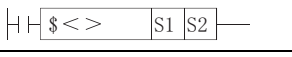
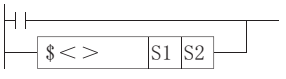
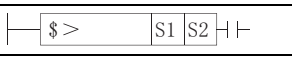
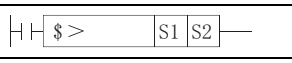
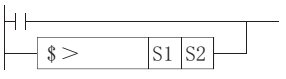
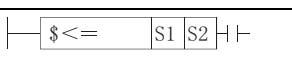
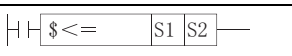
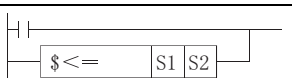
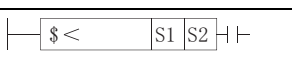
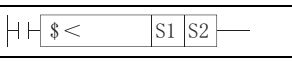
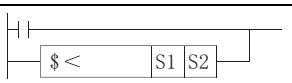
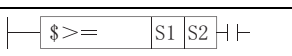
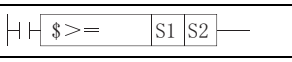
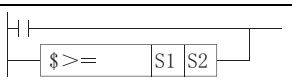
分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
浮点数据比较（双精度）	LDED=		· 当 (S1+3,S1+2,S1+1,S1)＝(S2+3,S2+2,S2+1,S2) 时处于导通状态 · 当 (S1+3,S1+2,S1+1,S1)≠(S2+3,S2+2,S2+1,S2) 时处于不导通状态		3	-	6-8
	ANDED=						
	ORED=						
	LDED<>		· 当 (S1+3,S1+2,S1+1,S1)≠(S2+3,S2+2,S2+1,S2) 时处于导通状态 · 当 (S1+3,S1+2,S1+1,S1)＝(S2+3,S2+2,S2+1,S2) 时处于不导通状态		3	-	
	ANDED<>						
	ORED<>						
	LDED>		· 当 (S1+3,S1+2,S1+1,S1)＞(S2+3,S2+2,S2+1,S2) 时处于导通状态 · 当 (S1+3,S1+2,S1+1,S1)≦(S2+3,S2+2,S2+1,S2) 时处于不导通状态		3	-	
	ANDED>						
	ORED>						
	LDED<=		· 当 (S1+3,S1+2,S1+1,S1)≦(S2+3,S2+2,S2+1,S2) 时处于导通状态 · 当 (S1+3,S1+2,S1+1,S1)＞(S2+3,S2+2,S2+1,S2) 时处于不导通状态		3	-	
	ANDED<=						
	ORED<=						
	LDED<		· 当 (S1+3,S1+2,S1+1,S1)＜(S2+3,S2+2,S2+1,S2) 时处于导通状态 · 当 (S1+3,S1+2,S1+1,S1)≧(S2+3,S2+2,S2+1,S2) 时处于不导通状态		3	-	
	ANDED<						
	ORED<						
	LDED>=		· 当 (S1+3,S1+2,S1+1,S1)≧(S2+3,S2+2,S2+1,S2) 时处于导通状态 · 当 (S1+3,S1+2,S1+1,S1)＜(S2+3,S2+2,S2+1,S2) 时处于不导通状态		3	-	
	ANDED>=						
	ORED>=						

表 2.10 比较运算指令（续）

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
字符串数据比较	LD$=		· 对字符串 S1 和字符串 S2 进行逐字比较。*2 · 当（字符串 S1)=(字符串 S2) 时处于导通状态 · 当（字符串 S1）≠（字符串 S2）时处于不导通状态		3	-	6-11
	AND$=						
	OR$=						
	LD$<>		· 对字符串 S1 和字符串 S2 进行逐字比较。*2 · 当（字符串 S1）≠（字符串 S2) 时处于导通状态 · 当（字符串 S1)=(字符串 S2）时处于不导通状态		3	-	
	AND$<>						
	OR$<>						
	LD$>		· 对字符串 S1 和字符串 S2 进行逐字比较。*2 · 当（字符串 S1）＞（字符串 S2) 时处于导通状态 · 当（字符串 S1）≤（字符串 S2）时处于不导通状态		3	-	
	AND$>						
	OR$>						
	LD$<=		· 对字符串 S1 和字符串 S2 进行逐字比较。*2 · 当（字符串 S1）≤（字符串 S2) 时处于导通状态 · 当（字符串 S1）＞（字符串 S2）时处于不导通状态		3	-	
	AND$<=						
	OR$<=						
	LD$<		· 对字符串 S1 和字符串 S2 进行逐字比较。*2 · 当（字符串 S1）＜（字符串 S2) 时处于导通状态 · 当（字符串 S1）≥（字符串 S2）时处于不导通状态		3	-	
	AND$<						
	OR$<						
	LD$>=		· 对字符串 S1 和字符串 S2 进行逐字比较。*2 · 当（字符串 S1）≥（字符串 S2) 时处于导通状态 · 当（字符串 S1）＜（字符串 S2）时处于不导通状态		3	-	
	AND$>=						
	OR$>=						

*2: 进行字符串比较时的比较条件如下所示：

· 一致的条件：字符串中的所有字符都一致时。

· 较大字符串的条件：如果字符串不同，取字符码较大的字符串。如果字符串的长度不同，取较长的字符串。

· 较小字符串的条件：如果字符串不同，取字符码较小的字符串。如果字符串的长度不同，取较短的字符串。

表 2.10 比较运算指令（续）

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
BIN16 位块数据比较	BKCMP=	BKCMP＝ S1 S2 D n	· 将从 S1 开始的 n 点数据与从 S2 开始的 n 点数据以 BIN16 位数据进行比较，并且将比较结果存储到从 (D) 中指定的位软元件开始的 n 点中。		5	-	6-15
	BKCMP<>	BKCMP＜＞ S1 S2 D n					
	BKCMP>	BKCMP＞ S1 S2 D n					
	BKCMP<=	BKCMP＜＝ S1 S2 D n					
	BKCMP<	BKCMP＜ S1 S2 D n					
	BKCMP>=	BKCMP＞＝ S1 S2 D n					
	BKCMP=P	BKCMP＝P S1 S2 D n					
	BKCMP<>P	BKCMP＜＞P S1 S2 D n					
	BKCMP>P	BKCMP＞P S1 S2 D n					
	BKCMP<=P	BKCMP＜＝P S1 S2 D n					
	BKCMP<P	BKCMP＜P S1 S2 D n					
	BKCMP>=P	BKCMP＞＝P S1 S2 D n					
BIN32 位块数据比较	DBKCMP=	DBKCMP＝ S1 S2 D n	· 将从 S1 中指定的软元件开始的 n 点 BIN 32 位数据或者常数与从 S2 中指定的软元件开始的 n 点 BIN 32 位数据进行比较，并且将运算结果存储到 (D) 中指定的软元件的后面。		5	-	6-18
	DBKCMP<>	DBKCMP＜＞ S1 S2 D n					
	DBKCMP>	DBKCMP＞ S1 S2 D n					
	DBKCMP<=	DBKCMP＜＝ S1 S2 D n					
	DBKCMP<	DBKCMP＜ S1 S2 D n					
	DBKCMP>=	DBKCMP＞＝ S1 S2 D n					
	DBKCMP=P	DBKCMP＝P S1 S2 D n					
	DBKCMP<>P	DBKCMP＜＞P S1 S2 D n					
	DBKCMP>P	DBKCMP＞P S1 S2 D n					
	DBKCMP<=P	DBKCMP＜＝P S1 S2 D n					
	DBKCMP<P	DBKCMP＜P S1 S2 D n					
	DBKCMP>=P	DBKCMP＞＝P S1 S2 D n					

2.4.2 算术运算指令

表 2.11 算术运算指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
BIN16 位加减运算	+	+ S D	· (D)+(S)→(D)		3	●	6-22
	+P	+P S D					
	+	+ S1 S2 D	· (S1)+(S2)→(D)		4	●	6-24
	+P	+P S1 S2 D					
	-	— S D	· (D)－(S)→(D)		3	●	6-22
	-P	—P S D					
	-	— S1 S2 D	· (S1)－(S2)→(D)		4	●	6-24
	-P	—P S1 S2 D					
BIN32 位加减运算	D+	D+ S D	· (D+1,D)+(S+1,S)→(D+1,D)		*1	●	6-26
	D+P	D+P S D					
	D+	D+ S1 S2 D	· (S1+1,S1)+(S2+1,S2)→(D+1,D)		*2	●	6-28
	D+P	D+P S1 S2 D					
	D-	D — S D	· (D+1,D)－(S+1,S)→(D+1,D)		*1	●	6-26
	D-P	D —P S D					
	D-	D — S1 S2 D	· (S1+1,S1)－(S2+1,S2)→(D+1,D)		*2	●	6-28
	D-P	D —P S1 S2 D					
BIN16 位乘除运算	*	* S1 S2 D	· (S1) × (S2)→(D+1,D)		*3	●	6-30
	*P	*P S1 S2 D					
	/	/ S1 S2 D	· (S1) ÷ (S2)→商 (D), 余数 (D+1)		4 *4	●	
	/P	/P S1 S2 D					
BIN32 位乘除运算	D*	D * S1 S2 D	· (S1+1,S1) × (S2+1,S2) →(D+3,D+2,D+1,D)		4 *4	●	6-32
	D*P	D *P S1 S2 D					
	D/	D/ S1 S2 D	· (S1+1,S1) ÷ (S2+1,S2) →商 (D+1,D), 余数 (D+3,D+2)		4 *4	●	
	D/P	D/P S1 S2 D					

*1: 步数根据所使用的软元件以及 CPU 类型而有所不同。

CPU 类型	使用软元件	步数
高性能型 QCPU 过程 CPU 冗余 CPU	· 字软元件： 内部软元件（但是文件寄存器 ZR 除外） · 位软元件： 软元件号为 16 的倍数，位数指定为 K8，无变址修饰。 · 常数： 无限制	5 注 1)
	使用除上述以外的软元件时。	3 注 2)
基本型 QCPU 通用型 QCPU	可使用的所有软元件	3 注 2)

注 1) 在高性能型 QCPU、过程 CPU、冗余 CPU 的情况下，步数将会增加，但处理速度将会变快。
注 2) 根据 3.8 节的条件步数有可能会增加。

*2: 步数根据所使用的软元件以及 CPU 类型而有所不同。

CPU 类型	使用软元件	步数
高性能型 QCPU 过程 CPU 冗余 CPU	· 字软元件： 内部软元件（但是文件寄存器 ZR 除外） · 位软元件： 软元件号为 16 的倍数，位数指定为 K8，无变址修饰。 · 常数： 无限制	6 注 1)
	使用除上述以外的软元件时。	4 注 2)
基本型 QCPU	可使用的所有软元件	4 注 2)
通用型 QCPU		3 注 2)

注 1) 在高性能型 QCPU、过程 CPU、冗余 CPU 的情况下，步数将会增加，但处理速度将会变快。
注 2) 根据 3.8 节的条件步数有可能会增加。

*3: 步数根据所使用的软元件以及 CPU 类型而有所不同。

CPU 类型	使用软元件	步数
QCPU	· 字软元件： 内部软元件（但是文件寄存器 ZR 除外） · 位软元件： 软元件号为 16 的倍数，位数指定为 K8，无变址修饰。 · 常数： 无限制	3
	使用除上述以外的软元件时。	4 注 1)

注 1) 根据 3.8 节的条件步数有可能会增加。

*4: 只有在通用型 QCPU 时基本步数为 3。

表 2.11 算术运算指令（续）

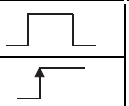
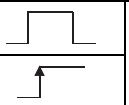
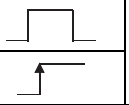
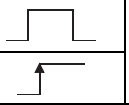
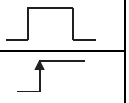
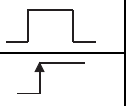
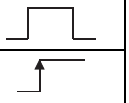
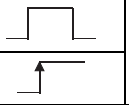
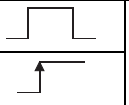
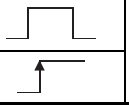
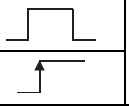
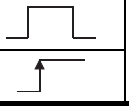
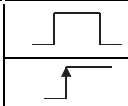
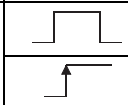
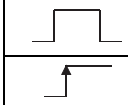
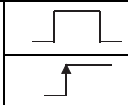
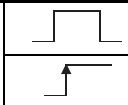
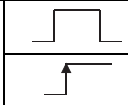
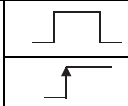
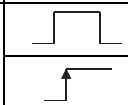
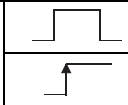
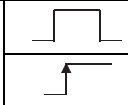
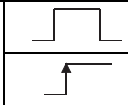
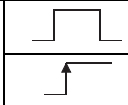
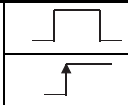
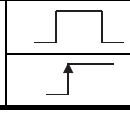
分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
BCD4 位数加减运算	B+	B+ S D	· (D)+(S)→(D)		3	●	6-34
	B+P	B+P S D					
	B+	B+ S1 S2 D	· (S1)+(S2)→(D)		4	-	6-36
	B+P	B+P S1 S2 D					
	B-	B－ S D	· (D)－(S)→(D)		3	●	6-34
	B-P	B－P S D					
	B-	B－ S1 S2 D	· (S1)－(S2)→(D)		4	-	6-36
	B-P	B－P S1 S2 D					
BCD8 位数加减运算	DB+	DB+ S D	· (D+1,D)+(S+1,S)→(D+1,D)		3	-	6-38
	DB+P	DB+P S D					
	DB+	DB+ S1 S2 D	· (S1+1,S1)+(S2+1,S2)→(D+1,D)		4	-	6-40
	DB+P	DB+P S1 S2 D					
	DB-	DB － S D	· (D+1,D)－(S+1,S)→(D+1,D)		3	-	6-38
	DB-P	DB －P S D					
	DB-	DB － S1 S2 D	· (S1+1,S1)－(S2+1,S2)→(D+1,D)		4	-	6-40
	DB-P	DB －P S1 S2 D					
BCD4 位数乘除运算	B*	B＊ S1 S2 D	· (S1) × (S2)→(D+1,D)		4	●	6-42
	B*P	B＊P S1 S2 D					
	B/	B/ S1 S2 D	· (S1) ÷ (S2)→商 (D), 余数 (D+1)		4	●	
	B/P	B/P S1 S2 D					
BCD8 位数乘除运算	DB*	DB ＊ S1 S2 D	· (S1+1,S1) × (S2+1,S2) →(D+3,D+2,D+1,D)		4	-	6-44
	DB*P	DB ＊P S1 S2 D					
	DB/	DB/ S1 S2 D	· (S1+1,S1) ÷ (S2+1,S2) →商 (D+1,D), 余数 (D+3,D+2)		4	●	
	DB/P	DB/P S1 S2 D					

表 2.11 算术运算指令（续）

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
浮点数据加减运算（单精度）	E+	E+ S D	· (D+1,D)+(S+1,S)→(D+1,D)		3	● *6	6-46
	E+P	E+P S D					
	E+	E+ S1 S2 D	· (S1+1,S1)+(S2+1,S2)→(D+1,D)		4 *5	● *6	6-48
	E+P	E+P S1 S2 D					
	E-	E− S D	· (D+1,D)−(S+1,S)→(D+1,D)		3	● *6	6-46
	E-P	E−P S D					
	E-	E− S1 S2 D	· (S1+1,S1)−(S2+1,S2)→(D+1,D)		4 *5	● *6	6-48
	E-P	E−P S1 S2 D					
浮点数据加减运算（双精度）	ED+	ED+ S D	· (D+3,D+2,D+1,D)+(S+3,S+2,S+1,S) →(D+3,D+2,D+1,D)		3	●	6-50
	ED+P	ED+P S D					
	ED+	ED+ S1 S2 D	· (S1+3,S1+2,S1+1,S1)+ (S2+3,S2+2,S2+1,S2)→ (D+3,D+2,D+1,D)		4	●	6-52
	ED+P	ED+P S1 S2 D					
	ED-	ED− S D	· (D+3,D+2,D+1,D)−(S+3,S+2,S+1,S) →(D+3,D+2,D+1,D)		3	●	6-50
	ED-P	ED−P S D					
	ED-	ED− S1 S2 D	· (S1+3,S1+2,S1+1,S1)− (S2+3,S2+2,S2+1,S2)→ (D+3,D+2,D+1,D)		4	●	6-52
	ED-P	ED−P S1 S2 D					
浮点数据乘除运算（单精度）	E*	E＊ S1 S2 D	· (S1+1,S1) × (S2+1,S2)→(D+1,D)		3	● *6	6-54
	E*P	E＊P S1 S2 D					
	E/	E/ S1 S2 D	· (S1+1,S1) ÷ (S2+1,S2)→商 (D+1,D)		4	● *6	
	E/P	E/P S1 S2 D					
浮点数据乘除运算（双精度）	ED*	ED＊ S1 S2 D	· (S1+3,S1+2,S1+1,S1) × (S2+3,S2+2,S2+1,S2)→ (D+3,D+2,D+1,D)		4	●	6-56
	ED*P	ED＊P S1 S2 D					
	ED/	ED/ S1 S2 D	· (S1+3,S1+2,S1+1,S1) ÷ (S2+3,S2+2,S2+1,S2)→ 商 (D+3,D+2,D+1,D)		4	●	
	ED/P	ED/P S1 S2 D					
BIN16 位数据块加减运算	BK+	BK+ S1 S2 D n	· 将从 (S1) 开始的 n 点 BIN16 位数据与从 (S2) 开始的 n 点数据进行批量加法。		5	-	6-58
	BK+P	BK+P S1 S2 D n					
	BK-	BK − S1 S2 D n	· 将从 (S1) 开始的 n 点 BIN16 位数据与从 (S2) 开始的 n 点数据进行批量减法。		5	-	
	BK-P	BK −P S1 S2 D n					

*5: 只有在使用通用型 QCPU 时，基本步数为 3。
*6: 只有在使用通用型 QCPU 时，子集才会有效。

表 2.11 算术运算指令（续）

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
BIN32 位数据块加减运算	DBK+	DBK+ S1 S2 D n	· 将从 (S1) 中指定的软元件开始的 n 点 BIN32 位数据或常数与从 (S2) 中指定的软元件开始的 n 点 BIN32 位数据进行加法运算后，将结果存储到 (D) 中指定的软元件的后面。		5	-	6-61
	DBK+P	DBK+P S1 S2 D n					
	DBK-	DBK − S1 S2 D n	· 将从 (S1) 中指定的软元件开始的 n 点 BIN32 位数据或常数与从 (S2) 中指定的软元件开始的 n 点 BIN32 位数据进行减法运算后，将结果存储到 (D) 中指定的软元件的后面。		5	-	
	DBK-P	DBK −P S1 S2 D n					
字符串数据合并	$+	$+ S D	· 将 (S) 中指定的字符串与 (D) 中指定的字符串相连接，并存储到 (D) 的后面。		3	-	6-64
	$+P	$+P S D					
	$+	$+ S1 S2 D	· 将 (S2) 中指定的字符串与 (S1) 中指定的字符串相连接，并存储到 (D) 的后面。		4	-	6-66
	$+P	$+P S1 S2 D					
BIN 数据递增	INC	INC D	· (D)+1→(D)		2	●	6-68
	INCP	INCP D					
	DINC	DINC D	· (D+1,D)+1→(D+1,D)		*7	●	6-70
	DINCP	DINCP D					
	DEC	DEC D	· (D)−1→(D)		2	●	6-68
	DECP	DECP D					
	DDEC	DDEC D	· (D+1,D)−1→(D+1,D)		*7	●	6-70
	DDECP	DDECP D					

*7: 步数根据所使用的软元件以及 CPU 类型而有所不同。

CPU 类型	使用软元件	步数
高性能型 QCPU 过程 CPU 冗余 CPU	· 字软元件： 内部软元件（但是文件寄存器 ZR 除外） · 位软元件： 软元件号为 16 的倍数，位数指定为 K8，无变址修饰。 · 常数： 无限制	3 注 1)
	使用除上述以外的软元件时。	2 注 2)
基本型 QCPU 通用型 QCPU	可使用的所有软元件	2 注 2)

注 1) 在高性能型 QCPU、过程 CPU、冗余 CPU 的情况下，步数将会增加，但处理速度将会变快。

注 2) 根据 3.8 节的条件步数有可能会增加。

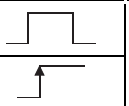
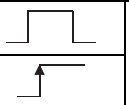
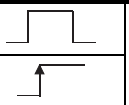
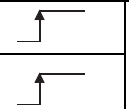
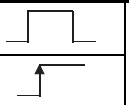
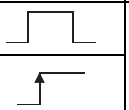
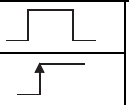
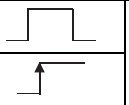
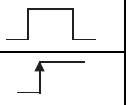
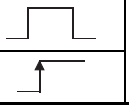
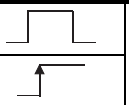
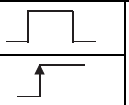
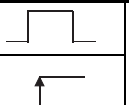
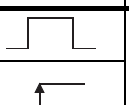
2.4.3 数据转换指令

表 2.12 数据转换指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
BCD 转换	BCD	BCD S D	(S) 转换为BCD (D) BIN(0～9999)		3 *1	●	6-72
	BCDP	BCDP S D					
	DBCD	DBCD S D	(S+1, S) 转换为BCD (D+1, D) BIN(0～99999999)		3 *1	●	
	DBCDP	DBCDP S D					
BIN 转换	BIN	BIN S D	(S) BIN转换 (D) BCD(0～9999)		3 *1	●	6-74
	BINP	BINP S D					
	DBIN	DBIN S D	(S+1, S) 转换为BIN (D+1, D) BCD(0～99999999)		3 *1	●	
	DBINP	DBINP S D					
BIN ↓ 浮点数转换（单精度）	FLT	FLT S D	(S) 转换为实数 (D+1, D) BIN(－32768～32767)		3 *1	● *2	6-77
	FLTP	FLTP S D					
	DFLT	DFLT S D	(S+1, S) 转换为实数 (D+1, D) BIN (-2147483648～2147483647)		3 *1	● *2	
	DFLTP	DFLTP S D					
BIN ↓ 浮点数转换（双精度）	FLTD	FLTD S D	(S) 转换为实数 (D+3, D+2, D+1, D) BIN(－32768～32767)		4	●	6-79
	FLTDP	FLTDP S D					
	DFLTD	DFLTD S D	(S+1, S) 转换为实数 (D+3, D+2, D+1, D) BIN (-2147483648～2147483647)		4	●	
	DFLTDP	DFLTDP S D					
浮点数（单精度）↓ BIN 转换	INT	INT S D	(S+1, S) 转换为BIN (D) 实数(-32768～32767)		3 *1	● *2	6-81
	INTP	INTP S D					
	DINT	DINT S D	(S+1, S) 转换为BIN (D+1, D) 实数(-2147483648～2147483647)		3 *1	● *2	
	DINTP	DINTP S D					
浮点数（双精度）↓ BIN 转换	INTD	INTD S D	(S+3, S+2, S+1, S) 转换为BIN (D) 实数(－32768～32767)		3	●	6-83
	INTDP	INTDP S D					
	DINTD	DINTD S D	(S+3, S+2, S+1, S) 转换为BIN (D+1, D) 实数(-2147483648～2147483647)		3	●	
	DINTDP	DINTDP S D					

*1: 只有在使用通用型 QCPU 时，基本步数为 2。
*2: 只有在使用通用型 QCPU 时，子集才会有效。

表 2.12 数据转换指令（续）

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
BIN16 位 ↕ 32 位转换	DBL	DBL S D	• (S) 转换 (D+1, D) BIN(-32768～32767)		3	-	6-85
	DBLP	DBLP S D					
	WORD	WORD S D	• (S+1, S) 转换 (D) BIN(-32768～32767)		3	-	6-86
	WORDP	WORDP S D					
BIN ↓ 格雷码转换	GRY	GRY S D	• (S) 转换为格雷码 (D) BIN(-32768～32767)		3	-	6-87
	GRYP	GRYP S D					
	DGRY	DGRY S D	• (S+1, S) 转换为格雷码 (D+1, D) BIN(-2147483648～2147483647)		3	-	
	DGRYP	DGRYP S D					
格雷码 ↓ BIN 转换	GBIN	GBIN S D	• (S) 转换为BIN数据 (D) 格雷码(-32768～32767)		3	-	6-89
	GBINP	GBINP S D					
	DGBIN	DGBIN S D	• (S+1, S) 转换为BIN数据 (D+1, D) 格雷码(-2147483648～2147483647)		3	-	
	DGBINP	DGBINP S D					
2 的补码	NEG	NEG D	• (D) (D) BIN数据		2	-	6-91
	NEGP	NEGP D					
	DNEG	DNEG D	• (D+1, D) (D+1, D) BIN数据		2	-	
	DNEGP	DNEGP D					
	ENEG	ENEG D	• (D+1, D) (D+1, D) 实数数据		2	-	6-93
	ENEGP	ENEGP D					
	EDNEG	EDNEG D	• (D+3, D+2, D+1, D) (D+3, D+2, D+1, D) 实数数据		3	-	6-94
	EDNEGP	EDNEGP D					
块转换	BKBCD	BKBCD S D n	· 将从 (S) 开始的 n 点的 BIN 数据批量转换为 BCD 数据，并存储到 (D) 的后面。		4	-	6-95
	BKBCDP	BKBCDP S D n					
	BKBIN	BKBIN S D n	· 将从 (S) 开始的 n 点的 BCD 数据批量转换为 BIN 数据，并存储到 (D) 的后面。		4	-	6-97
	BKBINP	BKBINP S D n					
浮点数单精度 ↓ 双精度	ECON	ECON S D	• (S+1, S) 转换为双精度 (D+3, D+2, D+1, D) 32位浮点型实数		3	-	6-99
	ECONP	ECONP S D					
浮点数双精度 ↓ 单精度	EDCON	EDCON S D	• (S+3, S+2, S+1, S) 转换为单精度 (D+1, D) 64位浮点型实数		3	-	6-100
	EDCONP	EDCONP S D					

2.4.4 数据传送指令

表 2.13 数据传送指令表

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
16 位数据传送	MOV	MOV S D	• (S) → (D		*3	●	6-102
	MOVP	MOVP S D			*1		
32 位数据传送	DMOV	DMOV S D	• (S+1,S) → (D+1,D)		*2	●	
	DMOVP	DMOVP S D					
浮点数据传送（单精度）	EMOV	EMOV S D	• (S+1, S) → (D+1, D) 实数数据		*2	●	6-104
	EMOVP	EMOVP S D					
浮点数据传送（双精度）	EDMOV	EDMOV S D	• (S+3, S+2, S+1, S) → (D+3, D+2, D+1, D) 实数数据		2	●	6-106
	EDMOVP	EDMOVP S D					
字符串数据传送	$MOV	$MOV S D	· 将 (S) 中指定的字符串传送到 (D) 中指定的软元件的后面。		3	-	6-108
	$MOVP	$MOVP S D					
16 位数据否定传送	CML	CML S D	• $\overline{(S)}$ → (D)		*1	●	6-111
	CMLP	CMLP S D					
32 位数据否定传送	DCML	DCML S D	• $\overline{(S+1,S)}$ → (D+1,D)		*2	●	
	DCMLP	DCMLP S D					
块传送	BMOV	BMOV S D n	(S) → (D), n		4	●	6-114
	BMOVP	BMOVP S D n					
同一 16 位数据块传送	FMOV	FMOV S D n	(S) → (D), n		4	●	6-117
	FMOVP	FMOVP S D n					
同一 32 位数据块传送	DFMOV	DFMOV S D n	(S+1, S) → (D+1, D), n		4	●	6-120
	DFMOVP	DFMOVP S D n					
16 位数据转换	XCH	XCH D1 D2	• (D1) ↔ (D2)		3	●	6-122
	XCHP	XCHP D1 D2					
32 位数据转换	DXCH	DXCH D1 D2	• (D1+1, D1) ↔ (D2+1, D2)		3	●	
	DXCHP	DXCHP D1 D2					
块数据转换	BXCH	BXCH S D n	(S) ↔ (D), n		4	-	6-124
	BXCHP	BXCHP S D n					
高低字节转换	SWAP	SWAP D	(S) b15 ～ b8 b7 ～ b0 [8位][8位] (D) b15 ～ b8 b7 ～ b0 [8位][8位]		3	-	6-126
	SWAPP	SWAPP D					

*1: 步数根据所使用的软元件以及 CPU 类型而有所不同。

CPU 类型	使用软元件	步数
QCPU	· 字软元件： 内部软元件（但是文件寄存器 ZR 除外） · 位软元件： 软元件号为 16 的倍数，位数指定为 K4，无变址修饰。 · 常数： 无限制	2
	使用除上述以外的软元件时。	3 注 1)

注 1) 根据 3.8 节的条件步数有可能会增加。

*2: 步数根据所使用的软元件以及 CPU 类型而有所不同。

CPU 类型	使用软元件	步数
高性能型 QCPU 过程 CPU 冗余 CPU	· 字软元件： 内部软元件（但是文件寄存器 ZR 除外） · 位软元件： 软元件号为 16 的倍数，位数指定为 K8，无变址修饰。 · 常数： 无限制	3
	使用除上述以外的软元件时。	3 注 1)
基本型 QCPU	· 字软元件： 内部软元件（但是文件寄存器 ZR 除外） · 位软元件： 软元件号为 16 的倍数，位数指定为 K8，无变址修饰。 · 常数： 无限制（使用上述软元件 + 常数时，步数变为 3。）	2
	使用除上述以外的软元件时。	3 注 1)
通用型 QCPU	可使用的所有软元件	2 注 1)

注 1) 根据 3.8 节的条件步数有可能会增加。

*3: 步数根据所使用的软元件以及 CPU 类型而有所不同。

CPU 类型	使用软元件	步数
QCPU	· 字软元件： 内部软元件（但是文件寄存器 ZR 除外） · 位软元件： 软元件号为 16 的倍数，位数指定为 K4，无变址修饰。 · 常数： 无限制	2
	使用除上述以外的软元件时。	3 注 1)

注 1) 根据 3.8 节的条件步数有可能会增加。

2.4.5 程序分支指令

表 2.14 程序分支指令表

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
跳转	CJ	CJ Pn	· 当输入条件成立时跳转到 Pn。		2	●	6-127
	SCJ	SCJ Pn	· 从输入条件成立后的下一个扫描跳转到 Pn。		2	●	
	JMP	JMP Pn	· 无条件地跳转到 Pn。		2	●	
	GOEND	GOEND	· 当输入条件成立时，跳转到 END 指令。		1	-	6-130

2.4.6 程序执行控制指令

表 2.15 程序执行控制指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
中断禁止	DI	DI	· 禁止执行中断程序。		1	-	6-131
中断允许	EI	EI	· 解除中断程序的执行禁止。		1	-	
中断禁止允许设置	IMASK	IMASK S	· 对各中断程序进行中断禁止 / 允许设置。		2	-	
返回	IRET	IRET	· 从中断程序返回至顺控程序。		1	-	6-137

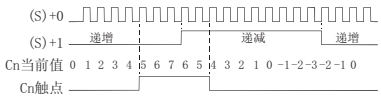
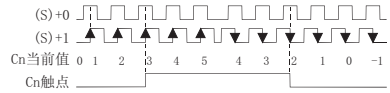
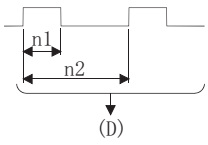
2.4.7 I/O 刷新指令

表 2.16 I/O 刷新指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
I/O 刷新	RFS	RFS S n	· 在扫描过程中对相应的输入输出部分进行刷新。		3	-	6-139
	RFSP	RFSP S n					

2.4.8 其它使用方便的指令

表 2.17 其它使用方便的指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
递增 / 递减计数器	UDCNT1	UDCNT1 S D n			4	-	6-141
	UDCNT2	UDCNT2 S D n			4	-	6-144
教学定时器	TTMR	TTMR D n	• (TTMR为ON的时间)×n ⟶ (D) n=0:1, n=1:10, n=2:100		3	-	6-147
特殊定时器	STMR	STMR S n D	· 根据 STMR 指令的输入条件的 ON/OFF 状态，对从 (D) 中指定的位软元件开始的 4 点执行以下动作： (D)+0：OFF 延迟定时器输出 (D)+1：OFF 后单次触发定时器输出 (D)+2：ON 后单次触发定时器输出 (D)+3：ON 延迟定时器 +OFF 延迟定时器		3	-	6-149
就近控制	ROTC	ROTC S n1 n2 D	· 在 n1 分割的旋转台上，从停止位置按照最短路径旋转到 (S+1) 中指定的位置。		5	-	6-152
斜坡信号	RAMP	RAMP n1 n2 D1 n3 D2	· 将 D1 中指定的软元件数据通过 n3 次扫描使其从 n1 变为 n2。		6	-	6-154
脉冲密度	SPD	SPD S n D	· 将 (S) 中指定的软元件的脉冲输入在 n 中指定的时间内计数，并将计数结果存储到 (D) 中指定的软元件中。		4	-	6-157
脉冲输出	PLSY	PLSY n1 n2 D	• (n1) Hz ⟶ (D) n2次输出		4	-	6-159
脉冲宽度调制	PWM	PWM n1 n2 D			4	-	6-161
矩阵输入	MTR	MTR S D1 D2 n	· 从 (S) 中指定的软元件开始，依次读取 16 点× n 列的数据，并将其存储到 (D2) 中指定的软元件后面。		5	-	6-163

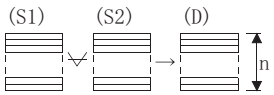
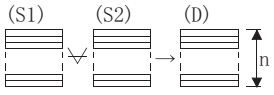
2.5 应用指令

2.5.1 逻辑运算指令

表 2.18 逻辑运算指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
逻辑积	WAND	WAND S D	• (D) ∧ (S) → (D)		3	●	7-3
	WANDP	WANDP S D					
	WAND	WAND S1 S2 D	• (S1) ∧ (S2) → (D)		4 *1	●	7-5
	WANDP	WANDP S1 S2 D					
	DAND	DAND S D	• (D+1, D) ∧ (S+1, S) → (D+1, D)		*2	●	7-3
	DANDP	DANDP S D					
	DAND	DAND S1 S2 D	• (S1+1, S1) ∧ (S2+1, S2) → (D+1, D)		*3	●	7-5
	DANDP	DANDP S1 S2 D					
	BKAND	BKAND S1 S2 D n	(S1) ∧ (S2) → (D) n		5	-	7-8
	BKANDP	BKANDP S1 S2 D n					
逻辑和	WOR	WOR S D	• (D) ∨ (S) → (D)		3	●	7-10
	WORP	WORP S D					
	WOR	WOR S1 S2 D	• (S1) ∨ (S2) → (D)		4 *1	●	7-12
	WORP	WORP S1 S2 D					
	DOR	DOR S D	• (D+1, D) ∨ (S+1, S) → (D+1, D)		*2	●	7-10
	DORP	DORP S D					
	DOR	DOR S1 S2 D	• (S1+1, S1) ∨ (S2+1, S2) → (D+1, D)		*3	●	7-12
	DORP	DORP S1 S2 D					
	BKOR	BKOR S1 S2 D n	(S1) ∨ (S2) → (D) n		5	-	7-14
	BKORP	BKORP S1 S2 D n					
排他逻辑和	WXOR	WXOR S D	• (D) ⊻ (S) → (D)		3	●	7-16
	WXORP	WXORP S D					
	WXOR	WXOR S1 S2 D	• (S1) ⊻ (S2) → (D)		4 *1	●	7-18
	WXORP	WXORP S1 S2 D					
	DXOR	DXOR S D	• (D+1, D) ⊻ (S+1, S) → (D+1, D)		*2	●	7-16
	DXORP	DXORP S D					

表 2.18 逻辑运算指令（续）

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
排他逻辑和	DXOR	DXOR S1 S2 D	• (S1+1, S1) ⩒ (S2+1, S2) → (D+1, D)		*3	●	7-18
	DXORP	DXORP S1 S2 D					
	BKXOR	BKXOR S1 S2 D n			5	-	7-20
	BKXORP	BKXORP S1 S2 D n					
否定排他逻辑和	WXNR	WXNR S D	• $\overline{(D) \veebar (S)} \rightarrow (D)$		3	●	7-22
	WXNRP	WXNRP S D					
	WXNR	WXNR S1 S2 D	• $\overline{(S1) \veebar (S2)} \rightarrow (D)$		4 *1	●	7-25
	WXNRP	WXNRP S1 S2 D					
	DXNR	DXNR S D	• $\overline{(D+1, D) \veebar (S+1, S)} \rightarrow (D+1, D)$		*2	●	7-22
	DXNRP	DXNRP S D					
	DXNR	DXNR S1 S2 D	• $\overline{(S1+1, S1) \veebar (S2+1, S2)} \rightarrow (D+1, D)$		*3	●	7-25
	DXNRP	DXNRP S1 S2 D					
	BKXNR	BKXNR S1 S2 D n			5	-	7-27
	BKXNRP	BKXNRP S1 S2 D n					

*1: 只有在使用通用型 QCPU 时，基本步数为 3。
*2: 步数根据所使用的软元件以及 CPU 模块类型而有所不同。

CPU 类型	使用软元件	步数
高性能型 QCPU 过程 CPU 冗余 CPU	· 字软元件： 内部软元件（但是文件寄存器 ZR 除外） · 位软元件： 软元件号为 16 的倍数，位数指定为 K8，无变址修饰。 · 常数： 无限制	5 注 1)
	使用除上述以外的软元件时。	3 注 2)
基本型 QCPU 通用型 QCPU	可使用的所有软元件	3 注 2)

注 1) 在高性能型 QCPU、过程 CPU、冗余 CPU 的情况下，步数将会增加，但处理速度将会变快。
注 2) 根据 3.8 节的条件步数有可能会增加。

*3: 步数根据所使用的软元件以及 CPU 模块类型而有所不同。

CPU 类型	使用软元件	步数
高性能型 QCPU 过程 CPU 冗余 CPU	· 字软元件： 内部软元件（但是文件寄存器 ZR 除外） · 位软元件： 软元件号为 16 的倍数，位数指定为 K8，无变址修饰。 · 常数： 无限制	6 注 1)
	使用除上述以外的软元件时。	4 注 2)
基本型 QCPU	可使用的所有软元件	4 注 2)
通用型 QCPU		3 注 2)

注 1) 在高性能型 QCPU、过程 CPU、冗余 CPU 的情况下，步数将会增加，但处理速度将会变快。
注 2) 根据 3.8 节的条件步数有可能会增加。

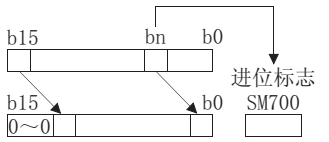
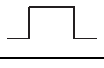
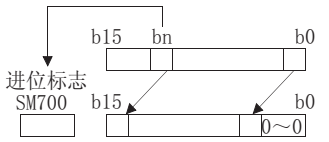
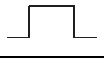
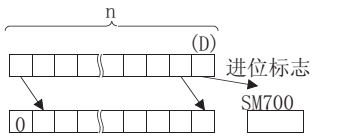
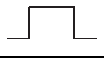
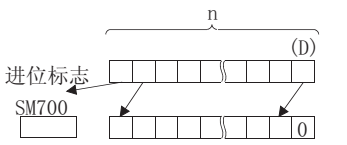
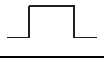
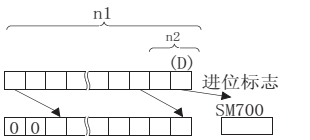
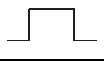
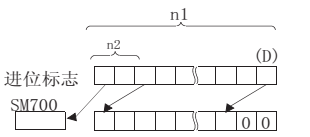
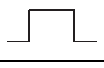
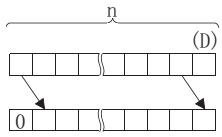
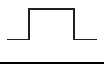
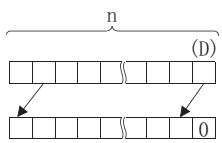
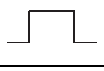
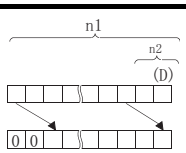
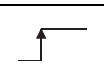
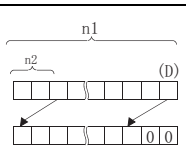
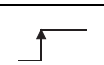
2.5.2 旋转指令

表 2.19 旋转指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
右旋转	ROR	ROR D n	b15 (D) b0 SM700 向右旋转n位 进位标志		3	●	7-29
	RORP	RORP D n					
	RCR	RCR D n	b15 (D) b0 SM700 向右旋转n位 进位标志		3	●	
	RCRP	RCRP D n					
左旋转	ROL	ROL D n	SM700 b15 (D) b0 进位标志 向左旋转n位		3	●	7-32
	ROLP	ROLP D n					
	RCL	RCL D n	SM700 b15 (D) b0 进位标志 向左旋转n位		3	●	
	RCLP	RCLP D n					
右旋转	DROR	DROR D n	(D+1) (D) b31 ～ b16 b15 ～ b0 SM700 向右旋转n位 进位标志		3	●	7-35
	DRORP	DRORP D n					
	DRCR	DRCR D n	(D+1) (D) b31 ～ b16 b15 ～ b0 SM700 向右旋转n位 进位标志		3	●	
	DRCRP	DRCRP D n					
左旋转	DROL	DROL D n	SM700 (D+1) (D) b31 ～ b16 b15 ～ b0 进位标志 向左旋转n位		3	●	7-37
	DROLP	DROLP D n					
	DRCL	DRCL D n	SM700 (D+1) (D) b31 ～ b16 b15 ～ b0 进位标志 向左旋转n位		3	●	
	DRCLP	DRCLP D n					

2.5.3 移位指令

表 2.20 移位指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
16 位数据的 n 位移位	SFR	SFR D n			3	●	7-39
	SFRP	SFRP D n					
	SFL	SFL D n			3	●	
	SFLP	SFLP D n					
n 位数据的 1 位移位	BSFR	BSFR D n			3	-	7-42
	BSFRP	BSFRP D n					
	BSFL	BSFL D n			3	-	
	BSFLP	BSFLP D n					
n 位数据的 n 位移位	SFTBR	SFTBR D n1 n2			4	-	7-47
	SFTBRP	SFTRP D n1 n2					
	SFTBL	SFTBL D n1 n2			4	-	
	SFTBLP	SFTBLP D n1 n2					
n 字数据的 1 字移位	DSFR	DSFR D n			3	●	7-47
	DSFRP	DSFRP D n					
	DSFL	DSFL D n			3	●	
	DSFLP	DSFLP D n					
n 字数据的 n 字移位	SFTWR	SFTWR D n1 n2			4	-	7-49
	SFTWRP	SFTWRP D n1 n2					
	SFTWL	SFTWL D n1 n2			4	-	
	SFTWLP	SFTWLP D n1 n2					

2.5.4 位处理指令

表 2.21 位处理指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
位设置 / 复位	BSET	BSET D n	(D) b15 bn b0 1		3	●	7-52
	BSETP	BSETP D n					
	BRST	BRST D n	(D) b15 bn b0 0		3	●	
	BRSTP	BRSTP D n					
位测试	TEST	TEST S1 S2 D	(S1) b15 ～ b0 (D) (S2) 中指定的位		4	-	7-54
	TESTP	TESTP S1 S2 D					
	DTEST	DTEST S1 S2 D	(S1) b31 ～ b0 (D) (S2) 中指定的位		4	-	
	DTESTP	DTESTP S1 S2 D					
位软元件批量复位	BKRST	BKRST D n	(D) ON OFF ON ON 复位 (D) OFF OFF OFF OFF n		3	-	7-57
	BKRSTP	BKRSTP D n					

2.5.5 数据处理指令

表 2.22 数据处理指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
数据查找	SER	SER S1 S2 D n	(S1) (S2) n (D) :匹配号 (D+1):匹配个数		5	-	7-59
	SERP	SERP S1 S2 D n					
	DSER	DSER S1 S2 D n	32位 (S1) (S2) n (D) :匹配号 (D+1):匹配个数		5	-	
	DSERP	DSERP S1 S2 D n					
位检查	SUM	SUM S D	(S) b15 b0 (D):1的个数		3	●	7-62
	SUMP	SUMP S D					
	DSUM	DSUM S D	(S+1) (S) (D):1的个数		3	●	
	DSUMP	DSUMP S D					
解码	DECO	DECO S D n	8→256解码 (S) 解码 (D) n 2^n位		4	-	7-64
	DECOP	DECOP S D n					
编码	ENCO	ENCO S D n	256→8编码 (S) 2^n位 编码 (D) n		4	-	7-66
	ENCOP	ENCOP S D n					

表 2.22 数据处理指令（续）

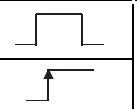
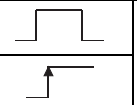
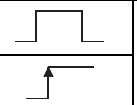
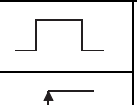
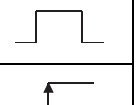
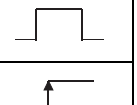
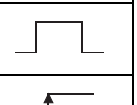
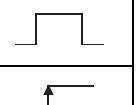
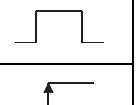
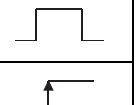
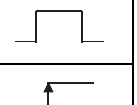
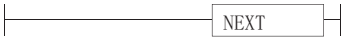
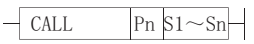
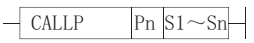
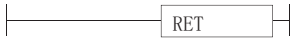
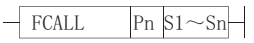
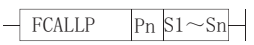
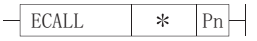
分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
7 段解码	SEG	SEG S D	b3～b0 (S) □□ (D) □ 7SEG		3	●	7-68
	SEGP	SEGP S D					
分离 · 合并	DIS	DIS S D n	· 将 (S) 中指定的 16 位数据以 4 位为单位进行分离后，存储到从 (D) 开始的 n 点的低 4 位中。(n ≦ 4)		4	-	7-70
	DISP	DISP S D n					
	UNI	UNI S D n	· 对 (S) 中指定的软元件开始的 n 点的低 4 位数据进行合并后，存储到 (D) 中指定的软元件中。(n ≦ 4)		4	-	7-72
	UNIP	UNIP S D n					
	NDIS	NDIS S1 D S2	· 将 (S1) 中指定的软元件后面的数据按 (S2) 中指定的位进行分离后，依次存储到 (D) 中指定的软元件后面。		4	-	7-74
	NDISP	NDISP S1 D S2					
	NUNI	NUNI S1 D S2	· 将 (S1) 中指定的软元件后面的数据与 (S2) 后面指定的各个位进行合并后，依次存储到 (D) 中指定的软元件后面。				
	NUNIP	NUNIP S1 D S2					
	WTOB	WTOB S D n	· 将 (S) 中指定的软元件开始的 n 点 16 位数据以 8 位为单位进行分解后，依次存储到 (D) 中指定的软元件后面。		4	-	7-78
	WTOBP	WTOBP S D n					
	BTOW	BTOW S D n	· 将 (S) 中指定的软元件开始的 n 点 16 位数据的低 8 位合并为 16 位后，依次存储到 (D) 中指定的软元件后面。				
	BTOWP	BTOWP S D n					
搜索	MAX	MAX S D n	· 将 (S) 中指定的软元件开始的 n 点的数据以 16 位为单位进行搜索，将最大值存储到 (D) 中指定的软元件中。		4	-	7-82
	MAXP	MAXP S D n					
	MIN	MIN S D n	· 将 (S) 中指定的软元件开始的 n 点的数据以 16 位为单位进行搜索，将最小值存储到 (D) 中指定的软元件中。				7-84
	MINP	MINP S D n					
	DMAX	DMAX S D n	· 将 (S) 中指定的软元件开始的 2 × n 点的数据以 32 位为单位进行搜索，将最大值存储到 (D) 中指定的软元件中。		4	-	7-82
	DMAXP	DMAXP S D n					
	DMIN	DMIN S D n	· 将 (S) 中指定的软元件开始的 2 × n 点的数据以 32 位为单位进行搜索，将最小值存储到 (D) 中指定的软元件中。				7-84
	DMINP	DMINP S D n					

表 2.22 数据处理指令（续）

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
排序	SORT	SORT S1 n S2 D1 D2 • S2：1次执行的比较数 • D1：排序结束后变为ON的软元件 • D2：系统用	· 将 (S1) 中指定的软元件开始的 n 点的数据以 16 位为单位进行排序。[需要 n × (n-1)/2 次扫描]		6	-	7-86
	DSORT	DSORT S1 n S2 D1 D2 • S2：1次执行的比较数 • D1：排序结束后变为ON的软元件 • D2：系统用	· 将 (S1) 中指定的软元件开始的 2 × n 点的数据以 32 位为单位进行排序。[需要 n × (n-1)/2 次扫描]				
合计值计算	WSUM	WSUM S D n	· 将 (S) 中指定的软元件开始的 n 点的 16 位 BIN 数据全部进行加法运算后，存储到 (D) 中指定的软元件中。		4	-	7-90
	WSUMP	WSUMP S D n					
	DWSUM	DWSUM S D n	· 将 (S) 中指定的软元件开始的 n 点的 32 位 BIN 数据全部进行加法运算后，存储到 (D) 中指定的软元件中。				7-92
	DWSUMP	DWSUMP S D n					
平均值计算	MEAN	MEAN S D n	· 对 (S) 中指定的软元件开始的 n 点的 (以 16 位为单位) 进行平均值计算后，存储到 (D) 中指定的软元件中。		4	-	7-94
	MEANP	MEANP S D n					
	DMEAN	DMEAN S D n	· 对 (S) 中指定的软元件开始的 n 点的 (以 32 位为单位) 进行平均值计算后，存储到 (D) 中指定的软元件中。				
	DMEANP	DMEANP S D n					

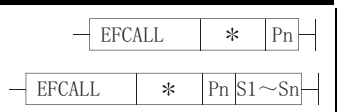
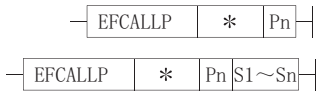
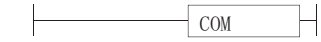
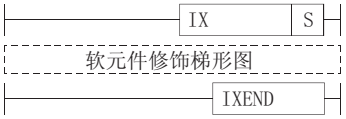
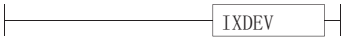
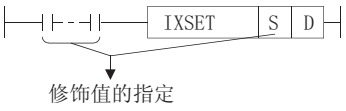
2.5.6 结构化指令

表 2.23 结构化指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
重复	FOR		· 在 FOR ～ NEXT 之间执行 n 次。		2	-	7-96
	NEXT				1	-	
	BREAK		· 强制结束 FOR ～ NEXT 之间的执行后，跳转到指针 Pn 处。		3	-	7-99
	BREAKP						
子程序调用	CALL	 	· 当输入条件成立时执行子程序 Pn。 (S1 ～ Sn 是至子程序的变量。n ≤ 5)		*1 2 + n	● *3	7-101
	CALLP	 					
	RET		· 从子程序处返回。		1	-	7-106
	FCALL	 	· 当输入条件不成立时不执行子程序 Pn。 (S1 ～ Sn 是至子程序的变量。n ≤ 5)		*1 2 + n	-	7-107
	FCALLP	 					
	ECALL	  *:文件名	· 当输入条件成立时执行指定程序的子程序 Pn。 (S1 ～ Sn 是至子程序的变量。n ≤ 5)		*2 3 + n	-	7-111
	ECALLP	  *:文件名					

*1: n 表示至子程序的变量数。
*2: n 表示至子程序的变量数与程序名的合计。
程序名的步数变为“ 程序中的字符数÷ 2 ”步（小数点以下被进位）”
*3: 只有在使用通用型 QCPU 时，子集才会有效。

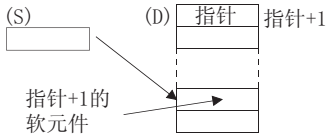
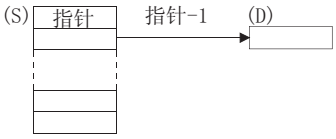
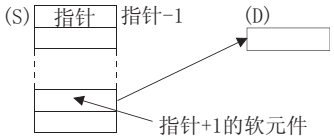
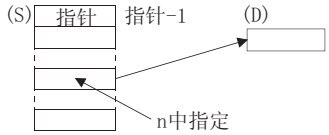
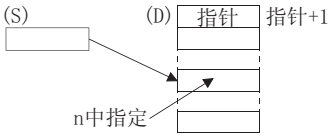
表 2.23 结构化指令（续）

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
子程序调用	EFCALL		· 当输入条件不成立时不执行指定程序的子程序 Pn。 (S1 ～ Sn 是至子程序的变量。n ≤ 5)		*2 3 + n	-	7-116
	EFCALLP						
	XCALL		· 当输入条件成立时执行子程序 Pn。 · 当输入条件不成立时不执行子程序 Pn。 (S1 ～ Sn 是至子程序的变量。n ≤ 5)		*1 2 + n	-	7-120
	COM		· 执行链接刷新、一般数据处理。		1	-	7-125
固定变址修饰	IX		· 对软元件修饰梯形图中使用的各软元件进行变址修饰。		2	-	7-128
	IXEND				1	-	
	IXDEV		· 将 IX ～ IXEND 之间用于进行变址修饰的修饰值存储到 D 中指定的软元件的后面。		1	-	7-132
	IXSET				3	-	

*1: n 表示至子程序的变量数。
*2: n 表示至子程序的变量数与程序名的合计。
程序名的步数变为“ 程序中的字符数÷ 2 ”步（小数点以下被进位）”

2.5.7 数据表操作指令

表 2.24 数据表操作指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
数据表处理	FIFW	FIFW S D			3	-	7-135
	FIFWP	FIFWP S D					
	FIFR	FIFR S D			3	-	7-137
	FIFRP	FIFRP S D					
	FPOP	FPOP S D			3	-	7-139
	FPOPP	FPOPP S D					
	FDEL	FDEL S D n			4	-	7-141
	FDELP	FDELP S D n					
	FINS	FINS S D n			4	-	7-141
	FINSP	FINSP S D n					

2.5.8 缓冲存储器访问指令

表 2.25 缓冲存储器访问指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
数据读取	FROM	FROM n1 n2 D n3	· 从智能功能模块以 16 位为单位进行数据读取。		5	-	7-144
	FROMP	FROMP n1 n2 D n3					
	DFRO	DFRO n1 n2 D n3	· 从智能功能模块以 32 位为单位进行数据读取。		5	-	
	DFROP	DFROP n1 n2 D n3					
数据写入	TO	TO n1 n2 S n3	· 从智能功能模块以 16 位为单位进行数据写入。		5	-	7-147
	TOP	TOP n1 n2 S n3					
	DTO	DTO n1 n2 S n3	· 从智能功能模块以 32 位为单位进行数据写入。		5	-	
	DTOP	DTOP n1 n2 S n3					

2.5.9 显示指令

表 2.26 显示指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
ASCII 打印	PR	＊SM701为OFF时 PR S D	· 从 (S) 中指定的软元件开始，将 8 点 (16 个字符) 的 ASCII 码输出到输出模块中。		3	-	7-150
	PR	＊SM701为ON时 PR S D	· 将从 (S) 中指定的软元件开始至 00H 为止的 ASCII 码输出到输出模块中。				
	PRC	PRC S D	· 将 (S) 中指定的软元件的注释转换为 ASCII 码后，输出到输出模块中。				7-153
复位	LEDR	LEDR	· 进行报警器的复位以及 LED 显示器的显示复位。		1	-	7-156

2.5.10 调试 · 故障诊断指令

表 2.27 调试 · 故障诊断指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
检查	CHKST	CHKST	· 当 CHKST 执行时执行 CHK 指令。 · 当 CHKST 未执行时，跳转至 CHK 指令的下一步。		1	-	7-159
	CHK	CHK 检查条件	· 正常时→SM80:OFF，SD80:0 · 异常时→SM80:ON，SD80: 故障号				
	CHKCIR	CHKCIR	· 通过 CHK 指令检查的梯形图模式的变更开始。		1	-	7-163
	CHKEND	CHKEND	· 通过 CHK 指令检查的梯形图模式的变更结束。				

2.5.11 字符串处理指令

表 2.28 字符串处理指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
BIN ↓ 10 进制 ASCII	BINDA	BINDA S D	· 将 (S) 中指定的 1 字 BIN 值转换为 5 位数 10 进制 ASCII 值后，存储到 (D) 中指定的字软元件中。		3	-	7-167
	BINDAP	BINDAP S D					
	DBINDA	DBINDA S D	· 将 (S) 中指定的 2 字 BIN 值转换为 10 位数 10 进制 ASCII 值后，存储到 (D) 中指定的字软元件中。		3	-	
	DBINDAP	DBINDAP S D					
BIN ↓ 16 进制 ASCII	BINHA	BINHA S D	· 将 (S) 中指定的 1 字 BIN 值转换为 4 位数 16 进制 ASCII 值后，存储到 (D) 中指定的字软元件中。		3	-	7-170
	BINHAP	BINHAP S D					
	DBINHA	DBINHA S D	· 将 (S) 中指定的 2 字 BIN 值转换为 8 位数 16 进制 ASCII 值后，存储到 (D) 中指定的字软元件中。		3	-	
	DBINHAP	DBINHAP S D					
BCD ↓ 10 进制 ASCII	BCDDA	BCDDA S D	· 将 (S) 中指定的 1 字 BCD 值转换为 4 位数 10 进制 ASCII 值后，存储到 (D) 中指定的字软元件号后面。		3	-	7-173
	BCDDAP	BCDDAP S D					
	DBCDDA	DBCDDA S D	· 将 (S) 中指定的 2 字 BCD 值转换为 8 位数 10 进制 ASCII 值后，存储到 (D) 中指定的字软元件号后面。		3	-	
	DBCDDAP	DBCDDAP S D					
10 进制 ASCII ↓ BIN	DABIN	DABIN S D	· 将 (S) 中指定的 5 位数 10 进制 ASCII 值转换为 1 字 BIN 值后，存储到 (D) 中指定的字软元件号后面。		3	-	7-176
	DABINP	DABINP S D					
	DDABIN	DDABIN S D	· 将 (S) 中指定的 10 位数 10 进制 ASCII 值转换为 2 字 BIN 值后，存储到 (D) 中指定的字软元件号后面。		3	-	
	DDABINP	DDABINP S D					
16 进制 ASCII ↓ BIN	HABIN	HABIN S D	· 将 (S) 中指定的 4 位数 16 进制 ASCII 值转换为 1 字 BIN 值后，存储到 (D) 中指定的字软元件号后面。		3	-	7-179
	HABINP	HABINP S D					
	DHABIN	DHABIN S D	· 将 (S) 中指定的 8 位数 16 进制 ASCII 值转换为 2 字 BIN 值后，存储到 (D) 中指定的字软元件号后面。		3	-	
	DHABINP	DHABINP S D					

表 2.28 字符串处理指令（续）

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
10 进制 ASCII ↓ BCD	DABCD	DABCD S D	· 将 (S) 中指定的 4 位数 10 进制 ASCII 值转换为 1 字 BCD 值后，存储到 (D) 中指定的字软元件号后面。		3	-	7-182
	DABCDP	DABCDP S D					
	DDABCD	DDABCD S D	· 将 (S) 中指定的 8 位数 10 进制 ASCII 值转换为 2 字 BCD 值后，存储到 (D) 中指定的字软元件号后面。		3	-	
	DDABCDP	DDABCDP S D					
软元件注释的读取	COMRD	COMRD S D	· 将 (S) 中指定的软元件的注释数据存储到 (D) 中指定的软元件中。		3	-	7-185
	COMRDP	COMRDP S D					
字符串的长度检测	LEN	LEN S D	· 将 (S) 中指定的软元件中存储的字符串数据的长度（字符数）存储到 (D) 中指定的软元件中。		3	-	7-188
	LENP	LENP S D					
BIN ↓ 10 进制字符串	STR	STR S1 S2 D	· 将 (S2) 中指定的 1 字 BIN 值转换为 (S1) 中指定的总位数和小数部分位数的 10 进制字符串后，存储到 (D) 中指定的软元件中。		4	-	7-190
	STRP	STRP S1 S2 D					
	DSTR	DSTR S1 S2 D	· 将 (S2) 中指定的 2 字 BIN 值转换为 (S1) 中指定的总位数和小数部分位数的 10 进制字符串后，存储到 (D) 中指定的软元件中。		4	-	
	DSTRP	DSTRP S1 S2 D					
10 进制字符串 ↓ BIN	VAL	VAL S D1 D2	· 将 (S) 中指定的包含有小数点的字符串转换为 1 字 BIN 值及小数部分位数后，存储到 (D1)、(D2) 中指定的软元件中。		4	-	7-196
	VALP	VALP S D1 D2					
	DVAL	DVAL S D1 D2	· 将 (S) 中指定的包含有小数点的字符串转换为 2 字 BIN 值及小数部分位数后，存储到 (D1)、(D2) 中指定的软元件中。		4	-	
	DVALP	DVALP S D1 D2					
浮点数 ↓ 字符串	ESTR	ESTR S1 S2 D	· 将 (S) 中指定的 32 位浮点数据转换为字符串后，存储到 (D) 中指定的软元件中。		4	-	7-200
	ESTRP	ESTRP S1 S2 D					
字符串 ↓ 浮点数	EVAL	EVAL S D	· 将 (S) 中指定的字符串转换为 32 位浮点数据后，存储到 (D) 中指定的软元件中。		3	-	7-206
	EVALP	EVALP S D					
16 进制 BIN ↓ ASCII	ASC	ASC S D n	· 将 (S) 中指定的软元件号后面的 1 字 BIN 值转换为 16 进制 ASCII 后，以 n 中指定的字符数存储到 (D) 中指定的字软元件号后面。		4	-	7-210
	ASCP	ASCP S D n					
ASCII ↓ 16 进制 BIN	HEX	HEX S D n	· 将 (S) 中指定的字软元件后面的 16 进制 ASCII 数据以 n 中指定的字符数转换为 BIN 值后，存储到 (D) 中指定的软元件号后面。		4	-	7-212
	HEXP	HEXP S D n					

表 2.28 字符串处理指令（续）

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
字符串处理	RIGHT	RIGHT S D n	· 将 (S) 中指定的字符串的最后字符开始的 n 个字符存储到 (D) 中指定的软元件中。		4	-	7-214
	RIGHTP	RIGHTP S D n					
	LEFT	LEFT S D n	· 将 (S) 中指定的字符串的起始字符开始的 n 个字符存储到 (D) 中指定的软元件中。				
	LEFTP	LEFTP S D n					
	MIDR	MIDR S1 D S2	· 在 (S1) 中指定的字符串中，将从 (S2) 中指定位置开始的指定数量的字符存储到 (D) 中指定的软元件中。		4	-	7-217
	MIDRP	MIDRP S1 D S2					
	MIDW	MIDW S1 D S2	· 将从 (S1) 中指定的字符串开始的指定数量的字符，存储到 (D) 中指定的字符串的 (S2) 中指定的位置处。				
	MIDWP	MIDWP S1 D S2					
	INSTR	INSTR S1 S2 D n	· 从 (S2) 中指定的字符串的第 n 个字符开始搜索 (S1) 中指定的字符串，并将匹配的位置存储到 (D) 中。		5	-	7-221
	INSTRP	INSTRP S1 S2 D n					
	STRINS	STRINS S D n	· 将 (S) 中指定的字符串数据插入到 (D) 中指定的字符串数据的从起始开始的第 (n) 个字符（插入位置）处。		4	–	7-227
	STRINSP	STRINSP S D n					
	STRDEL	STRDEL D n1 n2	· 在 (D) 中指定的字符串数据中，从起始算起的第 (n1) 个字符（删除开始位置）开始，删除 (n2) 中指定的字符数。		4	–	7-229
	STRDELP	STRDELP D n1 n2					
浮点数 ↓ BCD 分解	EMOD	EMOD S1 S2 D	· 将 (S1) 中指定的 32 位浮点数据按照 (S2) 中指定的小数部分的位数转换为 BCD 数据后，存储到 (D) 中指定软元件中。		4	-	7-227
	EMODP	EMODP S1 S2 D					
BCD ↓ 浮点数	EREXP	EREXP S1 S2 D	· 将 (S1) 中指定的 BCD 数据按照 (S2) 中指定的小数部分的位数转换为 32 位浮点数据后，存储到 (D) 中指定的软元件中。		4	-	7-229
	EREXPP	EREXPP S1 S2 D					

2.5.12 特殊函数指令

表 2.29 特殊函数指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
三角函数（浮点数单精度）	SIN	SIN S D	· Sin (S+1 , S) ⟶ (D+1 , D)		3	-	7-231
	SINP	SINP S D					
	COS	COS S D	·Cos (S+1 , S) ⟶ (D+1 , D)		3	-	7-235
	COSP	COSP S D					
	TAN	TAN S D	·Tan(S+1 , S) ⟶ (D+1 , D)		3	-	7-239
	TANP	TANP S D					
	ASIN	ASIN S D	·Sin^{-1} (S+1 , S) ⟶ (D+1 , D)		3	-	7-243
	ASINP	ASINP S D					
	ACOS	ACOS S D	·Cos^{-1} (S+1 , S) ⟶ (D+1 , D)		3	-	7-247
	ACOSP	ACOSP S D					
	ATAN	ATAN S D	·Tan^{-1} (S+1 , S) ⟶ (D+1 , D)		3	-	7-251
	ATANP	ATANP S D					
三角函数（浮点数双精度）	SIND	SIND S D	·Sin (S+3, S+2, S+1, S) ⟶ (D+3, D+2, D+1, D)		3	-	7-233
	SINDP	SINDP S D					
	COSD	COSD S D	·Cos (S+3, S+2, S+1, S) ⟶ (D+3, D+2, D+1, D)		3	-	7-237
	COSDP	COSDP S D					
	TAND	TAND S D	·Tan (S+3, S+2, S+1, S) ⟶ (D+3, D+2, D+1, D)		3	-	7-241
	TANDP	TANDP S D					
	ASIND	ASIND S D	·Sin^{-1} (S+3, S+2, S+1, S) ⟶ (D+3, D+2, D+1, D)		3	-	7-245
	ASINDP	ASINDP S D					
	ACOSD	ACOSD S D	·Cos^{-1} (S+3, S+2, S+1, S) ⟶ (D+3, D+2, D+1, D)		3	-	7-249
	ACOSDP	ACOSDP S D					
	ATAND	ATAND S D	·Tan^{-1} (S+3, S+2, S+1, S) ⟶ (D+3, D+2, D+1, D)		3	-	7-253
	ATANDP	ATANDP S D					

表 2.29 特殊函数指令（续）

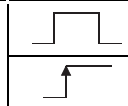
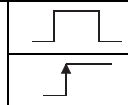
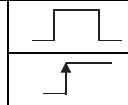
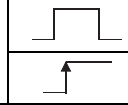
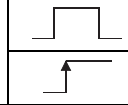
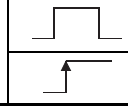
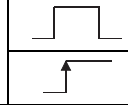
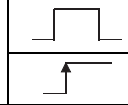
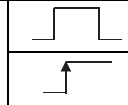
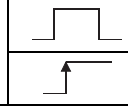
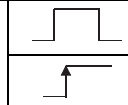
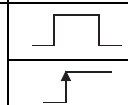
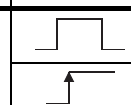
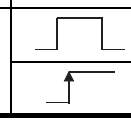
分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
度↕弧度转换	RAD	RAD S D	• (S+1, S) ⟶ (D+1, D) 度 → 弧度转换		3	-	7-255
	RADP	RADP S D					
	RADD	RADD S D	• (S+3, S+2, S+1, S) ⟶ (D+3, D+2, D+1, D) 度 → 弧度转换		3	-	7-257
	RADDP	RADDP S D					
	DEG	DEG S D	• (S+1, S) ⟶ (D+1, D) 弧度 → 度转换		3	-	7-259
	DEGP	DEGP S D					
	DEGD	DEGD S D	• (S+3, S+2, S+1, S) ⟶ (D+3, D+2, D+1, D) 弧度 → 度转换		3	-	7-261
	DEGDP	DEGDP S D					
平方根	SQR	SQR S D	$\cdot\sqrt{(S+1, S)}$ ⟶ (D+1 , D)		3	-	7-267
	SQRP	SQRP S D					
	SQRD	SQRD S D	$\cdot\sqrt{(S+3, S+2, S+1, S)}$ ⟶ (D+3, D+2, D+1, D)		3	-	7-269
	SQRDP	SQRDP S D					
指数运算	EXP	EXP S D	$\cdot e^{(S+1, S)}$ ⟶ (D+1 , D)		3	-	7-271
	EXPP	EXPP S D					
	EXPD	EXPD S D	$\cdot e^{(S+3, S+2, S+1, S)}$ ⟶ (D+3, D+2, D+1, D)		3	-	7-274
	EXPDP	EXPDP S D					
自然对数	LOG	LOG S D	$\cdot Log_e$ (S+1 , S) ⟶ (D+1 , D)		3	-	7-276
	LOGP	LOGP S D					
	LOGD	LOGD S D	$\cdot Log_e$ (S+3, S+2, S+1, S) ⟶ (D+3, D+2, D+1, D)		3	-	7-278
	LOGDP	LOGDP S D					
幂	POW	POW S1 S2 D	$\cdot$ (S1+1, S1) $^{(S2+1, S2)}$ ⟶ (D+1, D)		4	–	7-263
	POWP	POWP S1 S2 D					
	POWD	POWD S1 S2 D	$\cdot$ (S1+3, S1+2, S1+1, S1) $^{(S2+3, S2+2, S2+1, S2)}$ ⟶ (D+3, D+2, D+1, D)		4	–	7-265
	POWDP	POWDP S1 S2 D					
常用对数	LOG10	LOG10 S D	$\cdot$ log10 (S+1, S) ⟶ (D+1, D)		3	–	7-280
	LOG10P	LOG10P S D					
	LOG10D	LOG10D S D	$\cdot$ log10 (S+3, S+2S+1, S) ⟶ (D+3, D+2, D+1, D)		3	–	7-282
	LOG10DP	LOG10DP S D					

表 2.29 特殊函数指令（续）

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
随机数产生	RND	RND D	· 产生一个随机数 (0 ～ 32767) 后，存储到 (D) 中指定的软元件中。		2	-	7-284
	RNDP	RNDP D					
随机数系列变更	SRND	SRND D	· 根据存储在 (S) 中指定的软元件中的 16 位 BIN 数据的内容，对随机数系列进行变更。				
	SRNDP	SRNDP D					
平方根	BSQR	BSQR S D	• $\sqrt{(S)}$ → (D)+0 整数部分；+1 小数部分		3	-	7-286
	BSQRP	BSQRP S D					
	BDSQR	BDSQR S D	• $\sqrt{(S+1,\ S)}$ → (D)+0 整数部分；+1 小数部分		3	-	
	BDSQRP	BDSQRP S D					
三角函数	BSIN	BSIN S D	• Sin(S) → (D)+0 符号；+1 整数部分；+2 小数部分		3	-	7-289
	BSINP	BSINP S D					
	BCOS	BCOS S D	• Cos(S) → (D)+0 符号；+1 整数部分；+2 小数部分		3	-	7-291
	BCOSP	BCOSP S D					
	BTAN	BTAN S D	• Tan(S) → (D)+0 符号；+1 整数部分；+2 小数部分		3	-	7-293
	BTANP	BTANP S D					
	BASIN	BASIN S D	• $Sin^{-1}(S)$ → (D)+0 符号；+1 整数部分；+2 小数部分		3	-	7-295
	BASINP	BASINP S D					
	BACOS	BACOS S D	• $Cos^{-1}(S)$ → (D)+0 符号；+1 整数部分；+2 小数部分		3	-	7-297
	BACOSP	BACOSP S D					
	BATAN	BATAN S D	• $Tan^{-1}(S)$ → (D)+0 符号；+1 整数部分；+2 小数部分		3	-	7-299
	BATANP	BATANP S D					

2.5.13 数据控制指令

表 2.30 数据控制指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
上下限控制	LIMIT	LIMIT S1 S2 S3 D	· (S3)＜(S1) 时 ..... 将 (S1) 的值存储到 (D) 中 · (S1)≦(S3)≦(S2) 时 ..... 将 (S3) 的值存储到 (D) 中 · (S2)＜(S3) 时 ..... 将 (S2) 的值存储到 (D) 中	⌐¬	5	-	7-301
	LIMITP	LIMITP S1 S2 S3 D		↑			
	DLIMIT	DLIMIT S1 S2 S3 D	· (S3+1,S3)＜(S1+1,S1) 时……将 (S1+1，S1) 的值存储到 ((D)+1，(D)) 中 · (S1+1,S1)≦(S3+1,S3)＜(S2+1,S2) 时……将 (S3+1，S3) 的值存储到 ((D)+1，(D)) 中 · (S2,S2+1)＜(S3,S3+1) 时……将 (S2+1，S2) 的值存储到 ((D)+1，(D)) 中	⌐¬	5	-	
	DLIMITP	DLIMITP S1 S2 S3 D		↑			
死区控制	BAND	BAND S1 S2 S3 D	· (S1)≦(S3)≦(S2) 时…0→(D) · (S3)＜(S1) 时……(S3)－(S1)→(D) · (S2)＜(S3) 时……(S3)－(S2)→(D)	⌐¬	5	-	7-304
	BANDP	BANDP S1 S2 S3 D		↑			
	DBAND	DBAND S1 S2 S3 D	· (S1+1,S1)≦(S3+1,S3)≦(S2+1,S2) 时 …0→((D)+1,(D)) · (S3+1,S3)＜(S1+1,S1) 时… (S3+1,S3)－(S1+1,S1)→((D)+1,(D)) · (S2+1,S2)＜(S3+1,S3) 时… (S3+1,S3)－(S2+1,S2)→((D)+1,(D))	⌐¬	5	-	
	DBANDP	DBANDP S1 S2 S3 D		↑			
区域控制	ZONE	ZONE S1 S2 S3 D	· (S3)＝0 时…0→(D) · (S3)＞0 时…(S3)+(S2)→(D) · (S3)＜0 时…(S3)－(S1)→(D)	⌐¬	5	-	7-307
	ZONEP	ZONEP S1 S2 S3 D		↑			
	DZONE	DZONE S1 S2 S3 D	· (S3+1,S3)＝0 时…0→((D)+1,(D)) · (S3+1,S3)＞0 时… (S3+1,S3)+(S2+1,S2)→((D)+1,(D)) · (S3+1,S3)＜0 时… (S3+1,S3)+(S1+1,S1)→((D)+1,(D))	⌐¬	5	-	
	DZONEP	DZONEP S1 S2 S3 D		↑			

表 2.30 数据控制指令（续）

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
点坐标数据	SCL	SCL S1 S2 D	· 对 (S2) 中指定的标度用转换数据 (16 位数据单位) 通过 (S1) 中指定的输入值进行标度，将运算结果存储到 (D) 中指定的软元件中。 标度转换是基于存储在 (S2) 中指定的软元件后面的标度用转换数据进行的。		4	-	7-310
	SCLP	SCLP S1 S2 D					
	DSCL	DSCL S1 S2 D	· 对 (S2) 中指定的标度用转换数据 (32 位数据单位) 通过 (S1) 中指定的输入值进行标度，将运算结果存储到 (D) 中指定的软元件中。 标度转换是基于存储在 (S2) 中指定的软元件后面的标度用转换数据进行的。		4	-	
	DSCLP	DSCLP S1 S2 D					
X/Y 坐标数据	SCL2	SCL2 S1 S2 D	· 对 (S2) 中指定的标度用转换数据 (16 位数据单位) 通过 (S1) 中指定的输入值进行标度，将运算结果存储到 (D) 中指定的软元件中。 标度转换是基于存储在 (S2) 中指定的软元件后面的标度用转换数据进行的。		4	-	7-314
	SCL2P	SCL2P S1 S2 D					
	DSCL2	DSCL2 S1 S2 D	· 对 (S2) 中指定的标度用转换数据 (32 位数据单位) 通过 (S1) 中指定的输入值进行标度，将运算结果存储到 (D) 中指定的软元件中。 标度转换是基于存储在 (S2) 中指定的软元件后面的标度用转换数据进行的。		4	-	
	DSCL2P	DSCL2P S1 S2 D					

2.5.14 切换指令

表 2.31 切换指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
块号切换	RSET	RSET S	· 将扩展文件寄存器的块号变更为 (S) 中指定的编号。		2	-	7-317
	RSETP	RSETP S					
文件设置	QDRSET	QDRSET 文件名	· 对作为文件寄存器使用的文件名进行设置。		*1 2 + n	-	7-319
	QDRSETP	QDRSETP 文件名					
	QCDSET	QCDSET 文件名	· 对作为注释文件使用的文件名进行设置。		*1 2 + n	-	7-322
	QCDSETP	QCDSETP 文件名					

*1：n 表示“ 文件名的字符数÷ 2 ”步。(小数点以下被进位)

2.5.15 时钟指令

表 2.32 时钟指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
时钟数据的读取 / 写入	DATERD	DATERD D	• (时钟因子) →(D)+0 年 +1 月 +2 日 +3 时 +4 分 +5 秒 +6 星期		2	-	7-324
	DATERDP	DATERDP D					
	DATEWR	DATEWR S	• (D)+0 年 →(时钟因子) +1 月 +2 日 +3 时 +4 分 +5 秒 +6 星期		2	-	7-326
	DATEWRP	DATEWRP S					
时钟数据的加减运算	DATE+	DATE+ S1 S2 D	(S1) 时 分 秒 + (S2) 时 分 秒 → (D) 时 分 秒		4	-	7-328
	DATE+P	DATE+P S1 S2 D					
	DATE-	DATE- S1 S2 D	(S1) 时 分 秒 − (S2) 时 分 秒 → (D) 时 分 秒		4	-	7-330
	DATE-P	DATE-P S1 S2 D					
时钟数据的转换	SECOND	SECOND S D	(S) 时 分 秒 → (D) 秒(低位) 秒(高位)		3	-	7-332
	SECONDP	SECONDP S D					
	HOUR	HOUR S D	(S) 秒(低位) 秒(高位) → (D) 时 分 秒		3	-	7-334
	HOURP	HOURP S D					

表 2.32 时钟指令（续）

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
日期比较	LDDT=	DT = S1 S2 n	S1 年 S1+1 月 S1+2 日 = S2 年 S2+1 月 S2+2 日 → 比较运算结果		4	-	7-336
	ANDDT=	DT = S1 S2 n					
	ORDT=	DT = S1 S2 n					
	LDDT<>	DT <> S1 S2 n	S1 年 S1+1 月 S1+2 日 <> S2 年 S2+1 月 S2+2 日 → 比较运算结果		4	-	
	ANDDT<>	DT <> S1 S2 n					
	ORDT<>	DT <> S1 S2 n					
	LDDT<	DT < S1 S2 n	S1 年 S1+1 月 S1+2 日 < S2 年 S2+1 月 S2+2 日 → 比较运算结果		4	-	
	ANDDT<	DT < S1 S2 n					
	ORDT<	DT < S1 S2 n					
	LDDT<=	DT <= S1 S2 n	S1 年 S1+1 月 S1+2 日 <= S2 年 S2+1 月 S2+2 日 → 比较运算结果		4	–	
	ANDDT<=	DT <= S1 S2 n					
	ORDT<=	DT <= S1 S2 n					
	LDDT>	DT > S1 S2 n	S1 年 S1+1 月 S1+2 日 > S2 年 S2+1 月 S2+2 日 → 比较运算结果		4	–	
	ANDDT>	DT > S1 S2 n					
	ORDT>	DT > S1 S2 n					
	LDDT>=	DT >= S1 S2 n	S1 年 S1+1 月 S1+2 日 >= S2 年 S2+1 月 S2+2 日 → 比较运算结果		4	–	
	ANDDT>=	DT >= S1 S2 n					
	ORDT>=	DT >= S1 S2 n					

表 2.32 时钟指令（续）

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
时钟比较	LDTM=	TM = S1 S2 n	(S1) 时 / (S1)+1 分 / (S1)+2 秒 = (S2) 时 / (S2)+1 分 / (S2)+2 秒 → 比较运算结果		4	-	7-341
	ANDTM=	TM = S1 S2 n					
	ORTM=	TM = S1 S2 n					
	LDTM<>	TM <> S1 S2 n	(S1) 时 / (S1)+1 分 / (S1)+2 秒 <> (S2) 时 / (S2)+1 分 / (S2)+2 秒 → 比较运算结果		4	–	
	ANDTM<>	TM <> S1 S2 n					
	ORTM<>	TM <> S1 S2 n					
	LDTM<	TM < S1 S2 n	(S1) 时 / (S1)+1 分 / (S1)+2 秒 < (S2) 时 / (S2)+1 分 / (S2)+2 秒 → 比较运算结果		4	-	
	ANDTM<	TM < S1 S2 n					
	ORTM<	TM < S1 S2 n					
	LDTM<=	TM <= S1 S2 n	(S1) 时 / (S1)+1 分 / (S1)+2 秒 <= (S2) 时 / (S2)+1 分 / (S2)+2 秒 → 比较运算结果		4	–	
	ANDTM<=	TM <= S1 S2 n					
	ORTM<=	TM <= S1 S2 n					
	LDTM>	TM > S1 S2 n	(S1) 时 / (S1)+1 分 / (S1)+2 秒 > (S2) 时 / (S2)+1 分 / (S2)+2 秒 → 比较运算结果		4	–	
	ANDTM>	TM > S1 S2 n					
	ORTM>	TM > S1 S2 n					
	LDTM>=	TM >= S1 S2 n	(S1) 时 / (S1)+1 分 / (S1)+2 秒 >= (S2) 时 / (S2)+1 分 / (S2)+2 秒 → 比较运算结果		4	–	
	ANDTM>=	TM >= S1 S2 n					
	ORTM>=	TM >= S1 S2 n					

2.5.16 程序控制指令

表 2.34 程序控制指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
程序控制指令	PSTOP	PSTOP 文件名	· 将指定的程序设置为待机类型。		*1 2 + n	-	7-347
	PSTOPP	PSTOPP 文件名					
	POFF	POFF 文件名	· 将指定程序的 OUT 指令的线圈置于 OFF 后设置为待机类型。		*1 2 + n	-	7-349
	POFFP	POFFP 文件名					
	PSCAN	PSCAN 文件名	· 将指定程序作为扫描执行型进行登录。		*1 2 + n	-	7-351
	PSCANP	PSCANP 文件名					
	PLOW	PLOW 文件名	· 将指定程序作为低速执行型进行登录。		*1 2 + n	-	7-353
	PLOWP	PLOWP 文件名					
	LDPCHK	PCHK 文件名	· 指定文件名的程序处于执行状态时变为导通状态。 · 指定文件名的程序处于未执行状态时变为不导通状态。		*1 2 + n	-	7-355
	ANDPCHK	PCHK 文件名					
	ORPCHK	PCHK 文件名					

*1：n 表示“ 文件名的字符数÷ 2 ”步。（小数点以下被进位）

2.5.17 其它指令

表 2.35 其它指令

分类	指令符号	符号	处理内容	基本步数	参阅页面	指令符号	符号
WDT 复位	WDT	WDT	· 在顺控程序中对看门狗定时器进行复位。		1	-	7-357
	WDTP	WDTP					
定时时钟	DUTY	DUTY n1 n2 D	(D) n1扫描 n2扫描 SM420～SM424. SM430～SM434		4	-	7-359
时间检查	TIMCHK	TIMCHK S1 S2 D	· 对输入条件的 ON 进行计测，如果连续 ON 时间超过所设置的时间，则将 (D) 中指定的软元件置于 ON。		4	-	7-361
以 1 字节为单位进行直接读取 / 写入	ZRRDB	ZRRDB n D	0 低8位 1 高8位 2 低8位 3 高8位 n 8位 ZR0 ZR1 (D)		3	-	7-362
	ZRRDBP	ZRRDBP n D					
	ZRWRB	ZRWRB n S	(S) 0 低8位 1 高8位 2 低8位 3 高8位 n 8位 ZR0 ZR1		3	-	7-364
	ZRWRBP	ZRWRBP n S					
	ADRSET	ADRSET S D	(S) → (D) 指定软元件的间接地址 软元件名		3	-	7-366
	ADRSETP	ADRSETP S D					
通过键盘进行的数字键入	KEY	KEY S n D1 D2	· 对 (S) 中指定的输入模块的 8 点的 ASCII 数据进行读取，转换为 16 进制数值后存储到 (D1) 中指定的软元件号后面。		5	-	7-367
变址寄存器的批量保存	ZPUSH	ZPUSH D	· 将变址寄存器的内容保存到 (D) 中指定的软元件后面。		2	-	7-371
	ZPUSHP	ZPUSHP D					
变址寄存器的批量恢复	ZPOP	ZPOP D	· 将保存在 (D) 中指定的软元件后面的数据读取到变址寄存器中。				
	ZPOPP	ZPOPP D					

2.5.18 数据链接用指令

表 2.36 数据连接用指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
Q 系列网络刷新	S.ZCOM	S. ZCOM Jn	进行指定网络的刷新处理。		5	-	8-2
	SP.ZCOM	SP. ZCOM Jn					
	S.ZCOM	S. ZCOM Un					
	SP.ZCOM	SP. ZCOM Un					
路由信息的读取	S.RTREAD	S. RTREAD n D	对路由参数中设置的数据进行读取。		7	-	8-130 8-132
	SP.RTREAD	SP. RTREAD n D					
	Z.RTREAD	Z. RTREAD n D					
	ZP.RTREAD	ZP. RTREAD n D					
路由信息的登录	S.RTWRITE	S. RTWRITE n S	将路由数据写入到路由参数中指定的区域。		8	-	8-134 8-136
	SP.RTWRITE	SP. RTWRITE n S					
	Z.RTWRITE	Z. RTWRITE n S					
	ZP.RTWRITE	ZP. RTWRITE n S					

2.5.19 QCPU 指令

表 2.37 QCPU 指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
模块信息读取	UNIRD	UNIRD n1 D n2	· 从 (n) 中指定的起始 I/O 号开始，读取 (n2) 中指定点数的模块信息后，存储在 (D) 中指定的软元件后面。	⌐¬	4	-	9-2
	UNIRDP	UNIRDP n1 D n2		↑			
跟踪设置	TRACE	TRACE	· SM800、SM801、SM802 为 ON 时按设置的次数，将通过外围设备设置的跟踪数据存储到采样跟踪用文件中。	↑	1	-	9-6
跟踪复位	TRACER	TRACER	· 对通过 TRACE 指令设置的数据进行复位。	↑	1	-	
向指定文件写入数据	SP.FWRITE	SP.FWRITE U0 S0 D0 S1 S2 D1	· 对指定的文件进行数据写入。	↑	11	-	9-8
从指定文件读取数据	SP.FREAD	SP.FREAD U0 S0 D0 S1 S2 D1	· 对指定的文件进行数据读取。	↑	11	-	9-17
向标准 ROM 写入数据	SP.DEVST	SP.DEVST n1 S n2 D	· 对标准 ROM 的软元件数据存储用文件进行数据写入。	↑	9	-	9-29
从标准 ROM 读取数据	S.DEVLD	S.DEVLD n1 D n2	· 对标准 ROM 的软元件数据存储用文件进行数据读取。	⌐¬	8	-	9-31
	SP.DEVLD	SP.DEVLD n1 D n2		↑			
从存储器进行程序装载	PLOADP	PLOADP S D	· 将存储在存储卡、标准存储器（除驱动器 0 以外）中的程序传送到驱动器 0 中，并置于待机状态。	↑	3	-	9-33
从程序存储器中卸载程序	PUNLOADP	PUNLOADP S D	· 将存储在标准存储器（驱动器 0) 中的待机程序从存储器中删除。	↑	3	-	9-36
装载 + 卸载	PSWAPP	PSWAPP S1 S2 D	· 将存储在 (S1) 中指定的标准存储器（驱动器 0) 中的待机程序从存储器中删除，将存储在 (S2) 中指定的存储卡、标准存储器（除驱动器 0 以外）中的程序传送到驱动器 0 中，并置于待机状态。	↑	4	-	9-38
文件寄存器高速块传送	RBMOV	RBMOV S D n	· 将从 (S) 中指定的软元件开始的 n 点的 16 位数据批量传送到 (D) 中指定的软元件开始的 n 点区域内。	⌐¬	4	-	9-41
	RBMOVP	RBMOVP S D n		↑			

表 2.37 QCPU 指令（续）

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
自站 CPU 共享存储器写入	S.TO	S.TO n1 n2 n3 n4 D	· 将自站的软元件写入到自站 CPU 模块的 CPU 共享存储器中。		5	-	9-47
	SP.TO	SP.TO n1 n2 n3 n4 D					
	TO	TO n1 n2 S n3	· 将自站的软元件写入到自站 CPU 模块的 CPU 共享存储器中。		5	-	9-50
	TOP	TOP n1 n2 S n3					
	DTO	DTO n1 n2 S n3	· 将自站的软元件以 32 位为单位写入到自站 CPU 模块的 CPU 共享存储器中。		5	-	
	DTOP	DTOP n1 n2 S n3					
其它站 CPU 共享存储器读取	FROM	FROM n1 n2 D n3	· 从其它站 CPU 模块的 CPU 共享存储器中将软元件读取到自站 CPU 中。		5	-	9-55
	FROMP	FROMP n1 n2 D n3					
	DFRO	DFRO n1 n2 D n3	· 从其它站 CPU 模块的 CPU 共享存储器中将软元件以 32 位为单位读取到自站 CPU 中。		5	-	
	DFROP	DFROP n1 n2 D n3					
CPU 共享存储器自动刷新	COM	COM	· 进行智能功能模块的自动刷新、一般数据及 CPU 共享存储器的自动刷新。		1	-	9-61
扩展时钟数据的读取	S.DATERD	S.DATERD D	• (时钟因子) → (D)+0 年, +1 月, +2 日, +3 时, +4 分, +5 秒, +6 星期, +7 1/1000秒		6	-	9-67
	SP.DATERD	SP.DATERD D					
扩展时钟数据的加减运算	S.DATE+	S.DATE+ S1 S2 D	(S1) 时 分 秒 — 1/1000秒 + (S2) 时 分 秒 — 1/1000秒 → (D) 时 分 秒 — 1/1000秒		8	-	9-70
	SP.DATE+	SP.DATE+ S1 S2 D					
	S.DATE-	S.DATE− S1 S2 D	(S1) 时 分 秒 — 1/1000秒 − (S2) 时 分 秒 — 1/1000秒 → (D) 时 分 秒 — 1/1000秒		8	-	9-73
	SP.DATE-	SP.DATE− S1 S2 D					

2.5.20 冗余系统指令（用于冗余 CPU）

表 2.39 冗余系统指令（用于冗余 CPU）

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
系统切换	SP.CONTSW	SP. CONTSW S D	在执行了 SP.CONTSW 指令的扫描的 END 处理时，进行控制系统与待机系统的切换。		8	-	10-2

2.5.21 多 CPU 高速通信专用指令

表 2.40 多 CPU 高速通信专用指令

分类	指令符号	符号	处理内容	执行条件	基本步数	子集	参阅页面
向其它站 CPU 写入软元件	D.DDWR	D. DDWR n S1 S2 D1 D2	多 CPU 系统配置时，将自站 CPU 中指定的软元件(S2)后面的数据，按(S1)+1 中指定的写入数据点数存储到其它站 CPU(n1) 的指定软元件(D1)的后面。		10	-	11-11
	DP.DDWR	DP. DDWR n S1 S2 D1 D2			10	-	
从其它站 CPU 读取软元件	D.DDRD	D. DDRD n S1 S2 D1 D2	多 CPU 系统配置时，将其它站 CPU(n1) 的指定软元件(D1)的后面的数据，按(S1)+1 中指定的读取数据点数存储到自站 CPU 中指定的软元件(S2)的后面。		10	-	11-15
	DP.DDRD	DP. DDRD n S1 S2 D1 D2			10	-	

3 指令构成

3.1 指令构成

多数的 CPU 模块指令可分为指令部分和软元件部分。

指令部分和软元件部分的用途如下所示：

· 指令部分 ..... 表示该指令的功能。
· 软元件部分..... 表示指令中使用的数据。

软元件部分被分为源数据、目标数据和软元件数。

(1) 源数据 (S)

(a) 源数据是用于运算的数据。

(b) 根据各指令中指定的软元件，其形式如下所示：

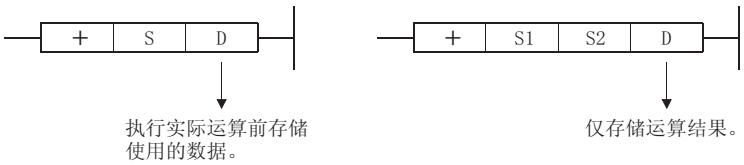
· 常数 .....指定运算中使用的数值。
是在创建程序时进行设置，因此在程序执行过程中无法变更。
将常数作为可变数据使用时，应进行变址修饰。

· 位软元件、字软元件 .....指定存储运算中使用的数据的软元件。
数据必须存储在指定的软元件当中，直到运算开始执行。
在程序执行过程中，通过变更指定软元件中存储的数据，可以对该指令中使用的数据进行更改。

(2) 目标数据 (D)

(a) 目标数据中存储运算后的数据。
但是，根据指令情况，有时运算前目标数据中也需要存储用于运算的数据。

例 BIN16 位数据的加法运算时

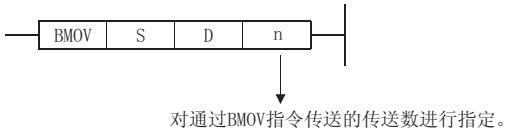


(b) 目标数据中必须设置用于数据存储的软元件。

(3) 软元件数 / 传送数 (n)

(a) 在软元件数 / 传送数中，对使用多个软元件的指令中所使用的软元件数 / 传送数进行指定。

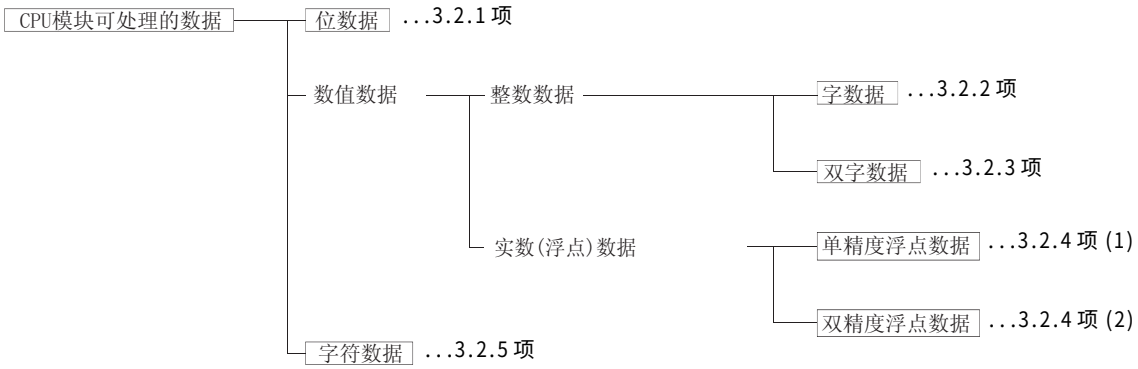
使用块传送指令时



(b) 软元件数 / 传送数的设置范围为 0 ～ 32767。
但是，软元件数 / 传送数为 0 时，该指令将被视为无效。

3.2 数据的指定方法

以下 6 种数据可以用于 CPU 模块指令：



3.2.1 使用位数据时

位数据是触点或线圈等以 1 位为单位处理的数据。

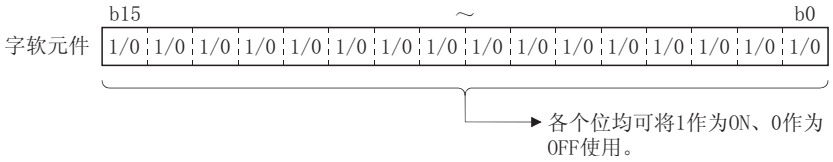
“ 位软元件 ”及“ 位指定字软元件 ”可以被当作位数据使用。

(1) 使用位软元件时

位软元件以 1 点为单位进行指定。

(2) 使用字软元件时

(a) 字软元件通过指定位号，可以将指定位号的 1/0 作为位数据使用。

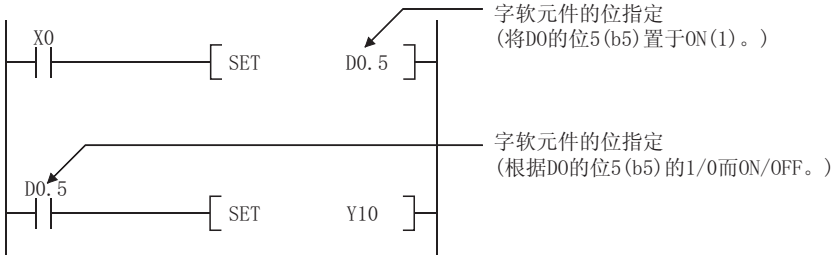


(b) 字软元件的位指定是通过指定“ **字软元件** . **位号** ”来完成的。

(位号指定是以 16 进制数进行)。

例如 :D0 的位 5(b5) 指定为 D0.5，D0 的位 10(b10) 指定为 D0.A。

但是，对于定时器 (T)、累计定时器 (ST)、计数器 (C) 或变址寄存器 (Z)，不能进行位指定 (例： 不能指定为 Z0.0)。



3.2.2 使用字 (16 位) 数据时

字数据是基本指令和应用指令中使用的 16 位数值数据。

以下两种形式的字数据可以在 CPU 模块中使用：

· 10 进制常数.....K-32768 ～ K32767
· 16 进制常数.....H0000 ～ HFFFF

字软元件和进行了位数指定的位软元件可以作为字数据使用。

但是，对于直接访问输入 (DX) 和直接访问输出 (DY)，不能通过位数指定进行字数据指定。(关于直接访问输入和直接访问输出，请参阅 CPU 用户手册 (功能解说 / 程序基础篇)。)

(1) 使用位软元件时

(a) 通过位数指定，位软元件就可以处理字数据。

位数据的位数指定是通过指定 “ 位数 位软元件的起始号 ” 来完成的。
位数指定以 4 点 (4 位) 为单位，可在 K1 ～ K4 的范围内指定。
(对于链接直接软元件，指定是通过 “ J 网络号 \ 位数 位软元件的起始号 ” 来完成的。例如将网络号 2 指定为 X100 ～ X10F 时，变为 J2\K4X100)。
例如，将位数指定为 X0 时，点数指定的情况如下所示：

· K1X0 ..... X0 ～ X3 的 4 点被指定
· K2X0 ..... X0 ～ X7 的 8 点被指定
· K3X0 ..... X0 ～ XB 的 12 点被指定
· K4X0 ..... X0 ～ XF 的 16 点被指定

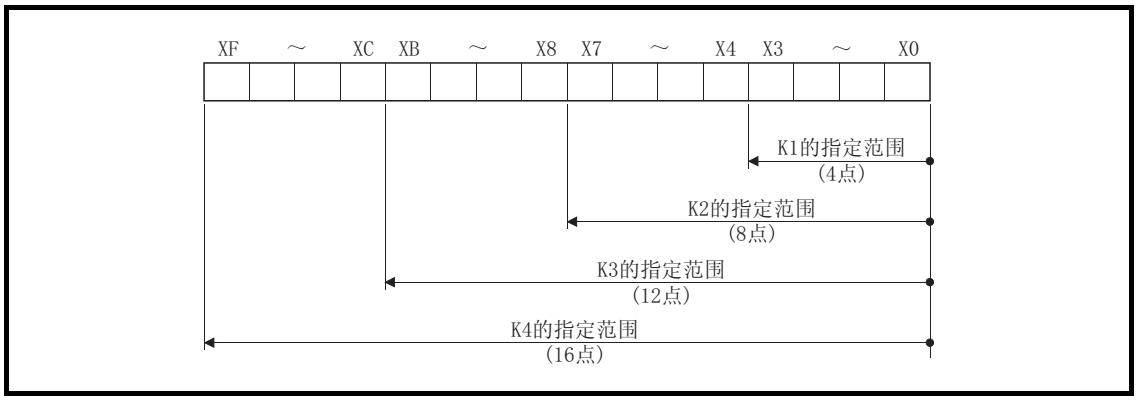


图 3.1 16 位指令时的位数指定设定范围

(b) 在源数据 (S) 中已进行了位数指定时，可作为源数据处理的数值如表 3.1 所示。

表 3.1 作为位数指定处理的数值一览表

指定位数	16 位指令时
K1(4 点)	0 ～ 15
K2(8 点)	0 ～ 255
K3(12 点)	0 ～ 4095
K4(16 点)	-32768 ～ 32767

(c) 目标为字软元件时

对于目标侧的字软元件，在源数据中已进行了数位指定的位后面的位状态将被存储为 0。

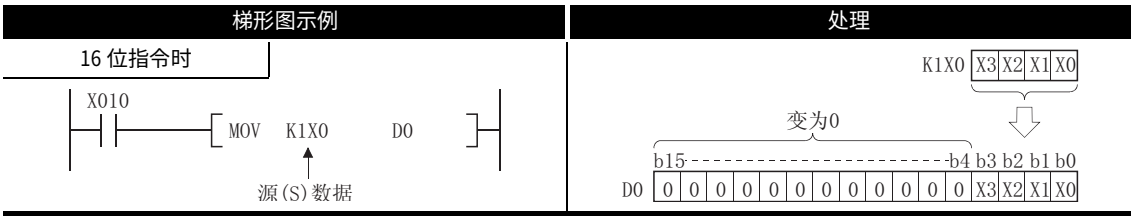


图 3.2 梯形图示例及处理内容

(d) 在目标 (D) 中已存在有位数指定时，则指定的点数将被作为目标使用。

进行了位数指定的点数后面的位软元件不发生变化。

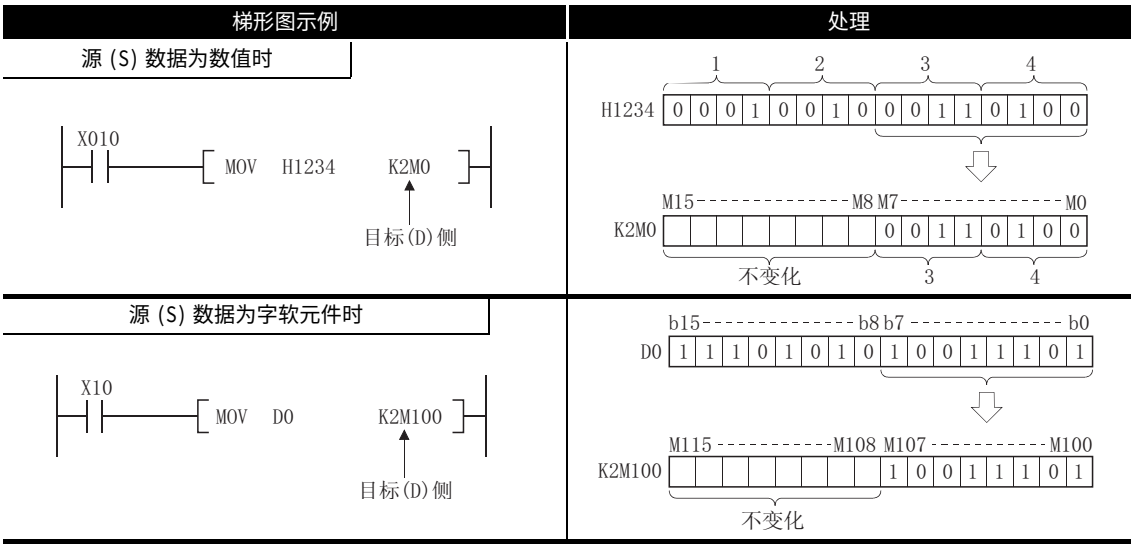
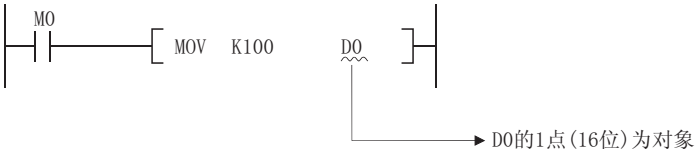


图 3.3 梯形图示例及处理内容

(2) 使用字软元件时

字软元件是以 1 点 (16 位) 为单位进行指定。



要 点

1. 进行位数指定处理时，位软元件的起始软元件号可以为任意数值。
2. 直接访问输入输出 (DX,DY) 不能进行位数指定。

3.2.3 使用双字数据 (32 位) 时

双字数据是基本指令和应用指令中使用的 32 位数值数据。

CPU 模块可处理的双字数据有以下 2 种：

· 10 进制常数 ..... K-2147483648 ～ K2147483647

· 16 进制常数 ..... H00000000 ～ HFFFFFFFF

字软元件以及进行了位数指定的位软元件可以当作双字数据使用。

但是，对于直接访问输入 (DX) 和直接访问输出 (DY)，不能通过位数指定进行双字数据指定。

(1) 使用位软元件时

(a) 位软元件通过位数指定可以处理双字数据。

位软元件的位数指定是通过指定“ 位数 位软元件的起始号 ”来完成的。

(对于链接直接软元件，指定是通过“ J 网络号 \ 位数 位软元件的起始号 ”来完成的。)

将网络号 2 指定为 X100 ～ X11F 时，变为 J2\K8X100)。

位数指定以 4 点 (4 位) 为单位，可在 K1 ～ K8 的范围内指定。

例如，将位数指定为 K0 时，点数指定的情况如下所示：

· K1X0 .....X0 ～ X3 的 4 点被指定

· K2X0 .....X0 ～ X7 的 8 点被指定

· K3X0 .....X0 ～ XB 的 12 点被指定

· K4X0 .....X0 ～ XF 的 16 点被指定

· K5X0 ..... X0 ～ X13 的 20 点被指定

· K6X0 ..... X0 ～ X17 的 24 点被指定

· K7X0 ..... X0 ～ X1B 的 28 点被指定

· K8X0 ..... X0 ～ X1F 的 32 点被指定

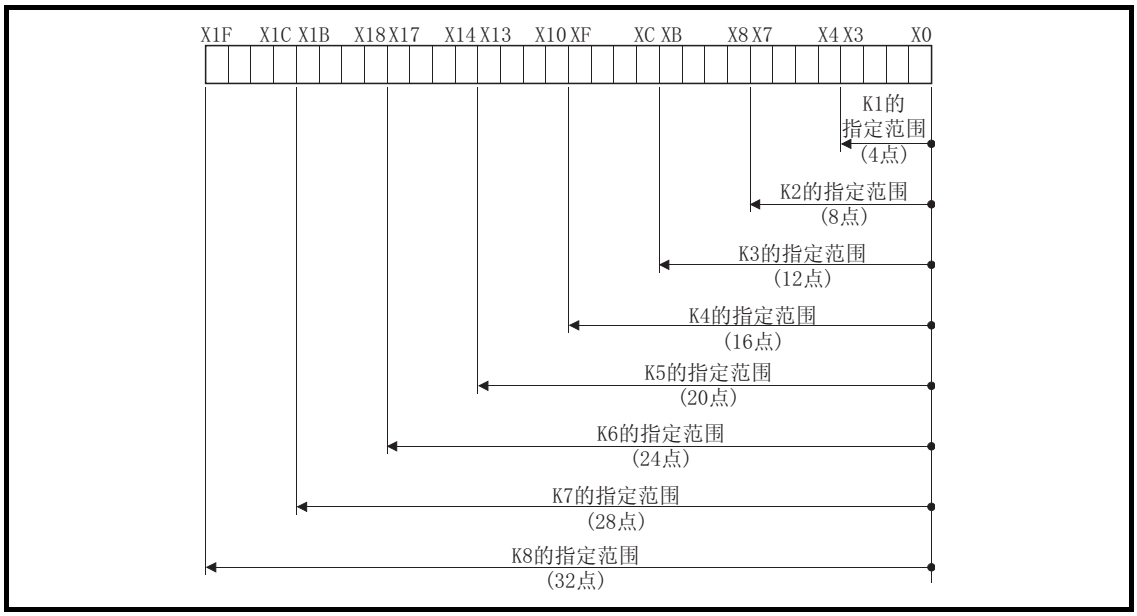


图 3.4 32 位指令位指定设置范围

(b) 在源数据 (S) 中已存在有位数指定时，可作为源数据处理的数值如表 3.2 所示。

表 3.2 作为位数指定处理的数值一览表

指定位数	32 位指令时	指定位数	32 位指令时
K1(4 点)	0 ～ 15	K5(20 点)	0 ～ 1048575
K2(8 点)	0 ～ 255	K6(24 点)	0 ～ 16777215
K3(12 点)	0 ～ 4095	K7(28 点)	0 ～ 268435455
K4(16 点)	0 ～ 65535	K8(32 点)	－2147483648 ～ 2147483647

(c) 目标为字软元件时

对于目标侧的字软元件，在源数据中已进行了数位指定的位后面的位状态将被存储为0。

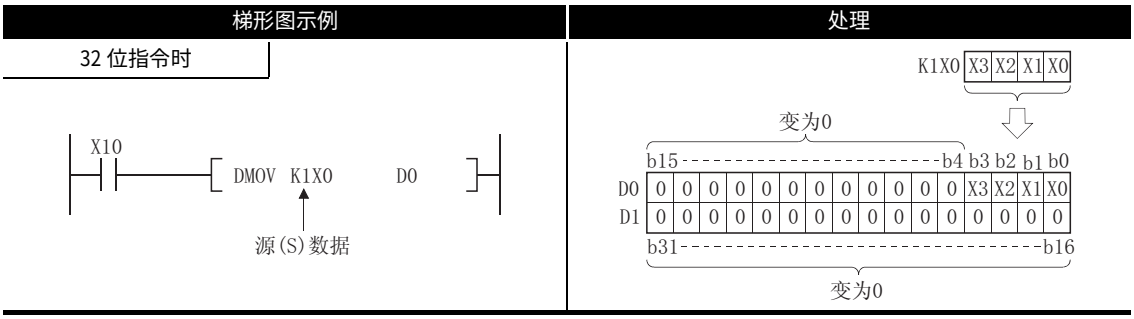


图 3.5 梯形图示例及处理内容

(d) 在目标 (D) 中已存在有位数指定时，则指定的点数将被作为目标使用。

进行了位数指定的点数后面的位软元件不发生变化。

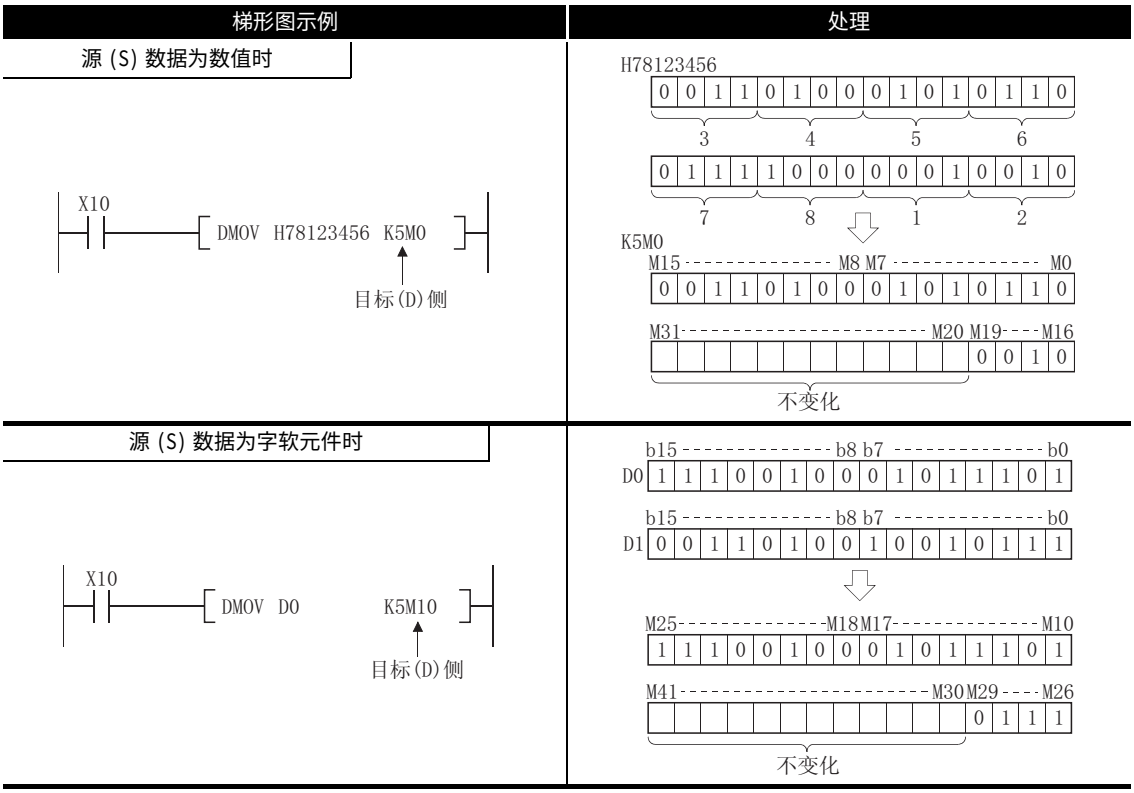


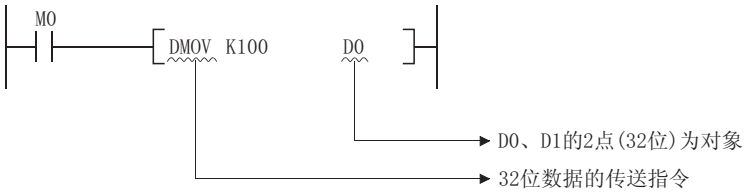
图 3.6 梯形图示例及处理内容

要 点

1. 进行位数指定处理时，位软元件的起始软元件号可以为任意数值。
2. 直接访问输入输出 (DX,DY) 不能进行位数指定。

(2) 使用字软元件时

字软元件是被指定为在低 16 位中使用的软元件。
在 32 位指令中，使用（指定软元件号）及（指定软元件号 +1)。



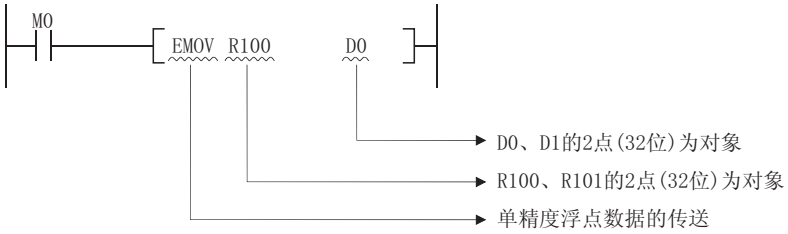
3.2.4 使用实数数据时

实数数据是用于基本指令和应用指令的浮点数据。

只有字软元件能够存储实数数据。

(1) 单精度浮点数据

在处理单精度浮点数据的指令中，指定低 16 位中使用的软元件。
单精度浮点数据存储在（指定软元件号）及（指定软元件号 +1) 的 32 位中。



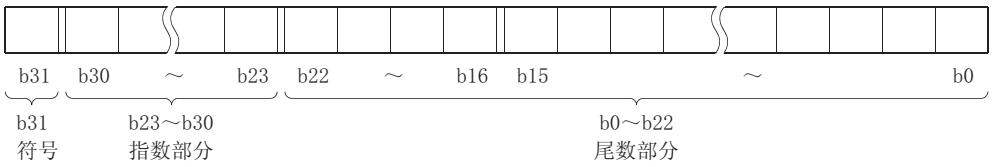
备注

在顺控程序中，浮点数据通过 E□指定。

单精度浮点数据使用 2 个字软元件并以下列方式表达：

[符号] 1.[尾数部分] × 2 [指数部分]

单精度浮点数据内部表示时的位构成及含义如下：



· 符号　通过 b13 表示符号。
0: 正
1: 负

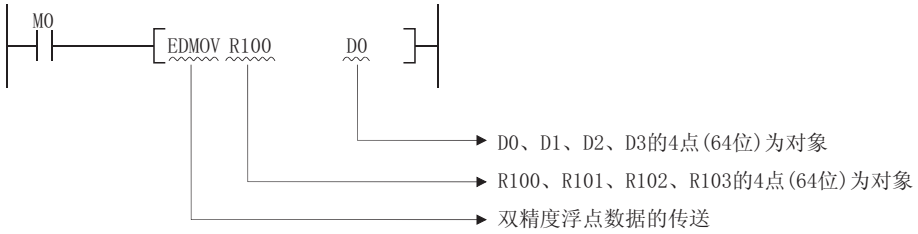
· 指数部分　通过 b23 ～ b30 表示 2^n 的 n。
根据 b23 ～ b30 的 BIN 值，n 的值如下所示：

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

· 尾数部分　通过 b0 ～ b22 的 23 位表示，在 2 进制数中 1.XXXXXX…表示为 XXXXXX…的值。

(2) 双精度浮点数据

在处理双精度浮点数据的指令中，指定低 16 位中使用的软元件。
双精度浮点数据存储在（指定软元件号）～（指定软元件号 +3）的 64 位中。



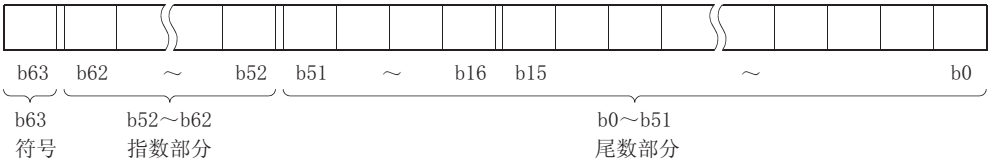
备注

在顺控程序中，浮点数据通过 E□指定。

双精度浮点数据使用 4 个字软元件并以下列方式表达：

[符号] 1.[尾数部分] × 2 [指数部分]

双精度浮点数据内部表示时的位构成及含义如下：



· 符号　通过 b63 表示符号。
0：正
1：负

· 指数部分　通过 b52 ～ b62 表示 2^n 的 n。
根据 b52 ～ b62 的 BIN 值，n 的值如下所示：

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

· 尾数部分　通过 b0 ～ b51 的 52 位表示，在 2 进制数中 1.XXXXXX…表示为 XXXXXX…的值。

要 点

1. 通过外围设备的监视功能可以监视 CPU 模块的浮点数据。
2. 在浮点数据表示 0 时，以下范围全部变为 0。
   (a) 单精度浮点数据时： b0 ～ b31
   (a) 双精度浮点数据时： b0 ～ b63
3. 浮点数据的设置范围如下所示。*1
   (a) 单精度浮点数据时
   -2^{128} ＜软元件≤ -2^{-126}，0 ，2^{-126} ≤软元件＜ 2^{128}
   (a) 双精度浮点数据时
   -2^{1024} ＜软元件≤ -2^{-1022}，0 ，2^{-1022} ≤软元件＜ 2^{1024}
4. 不要在浮点数据中指定－0(只有浮点型实数的最高位为 1 时)。(如果以－0 进行浮点运算，将发生运算错误)。
   在以双精度进行浮点运算的内部运算的 CPU 模块中指定了 -0 时，在 CPU 模块内部将会把 -0 转换为 0 后再执行浮点运算，因此不会发生运算错误。
   在以单精度进行浮点运算的内部运算的 CPU 模块中指定了 -0 时，因为处理速度优先，在浮动点运算时原样不变地使用 -0，因此将发生运算错误。
   (a) 当指定了 -0 时，下列 CPU 模块中不会发生运算错误。
   · 内部运算设置为双精度的高性能型 QCPU *(浮点运算的内部运算默认为双精度。)
   (b) 当指定了 -0 时，下列 CPU 模块中将发生运算错误。
   · 基本型 QCPU *3
   · 内部运算设置为单精度的高性能型 QCPU *2
   · 过程 CPU
   · 冗余 CPU
   · 通用型 QCPU

*1: 关于实数超出范围和输入了特殊值时的动作情况，请参阅所使用的 CPU 模块的用户手册 (功能解说 / 程序基础篇)。

*2: 浮点运算的内部运算的单精度与双精度之间的切换是在可编程控制器参数的可编程控制器系统设置中进行。关于浮点运算的单精度及双精度有关内容，请参阅所使用的 CPU 模块的用户手册 (功能解说 / 程序基础篇)。

*3: 在序列号的前五位为 “ 04122 或以后 ” 的基本型 QCPU 中，能够执行浮点运算。

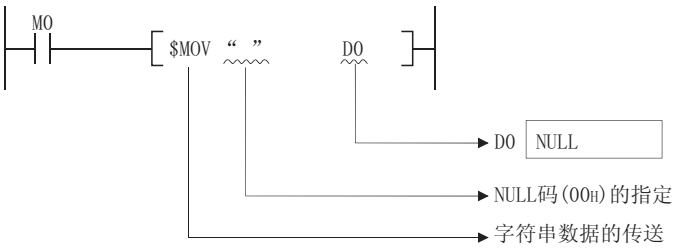
3.2.5 使用字符串数据时

字符串数据是基本指令和应用指令中使用的字符数据。

它包含从指定字符起至表示字符串末尾的 NULL 码 (00H) 为止的所有数据。

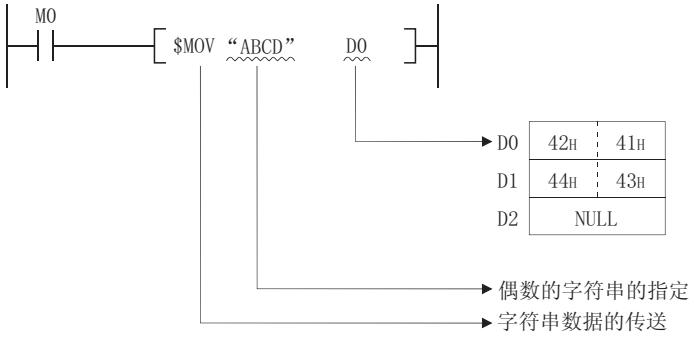
(1) 当指定字符为 NULL 码时

使用 1 个字来存储 NULL 码。



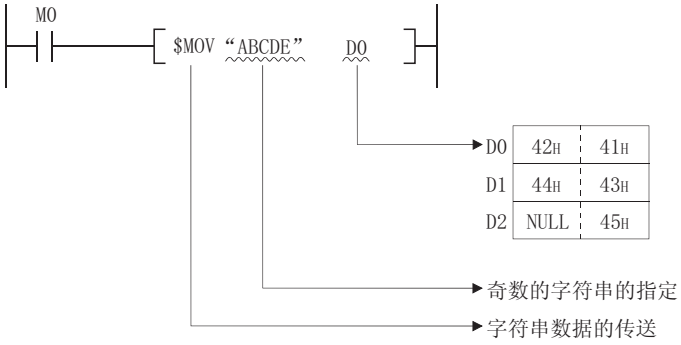
(2) 当字符数是偶数时

使用 (字符数 /2+1) 个字存储字符串及 NULL 码。
例如，如果将 " ABCD " 传送至 D0 ～，则字符串 (ABCD) 将被存储到 D0 及 D1 中，NULL 码将被存储到 D2 中。(NULL 码将被存储到最后的 1 个字中。)



(3) 当字符数是奇数时

使用 (字符数 /2) 个字 (小数部分进位) 存储字符串及 NULL 码。
例如，如果将 " ABCDE " 传送到 D0 ～，则字符串 (ABCDE) 及 NULL 码将被存储到 D0 ～ D2 中。(NULL 码将被存储到最后 1 个字的高 8 位处。)



3.3 变址修饰

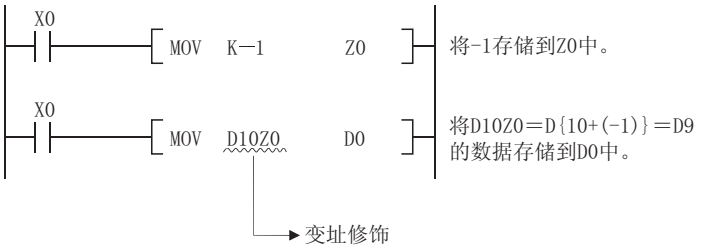
(1) 变址修饰的概要

(a) 变址修饰是通过使用变址寄存器进行的间接设置。
在顺控程序中使用变址修饰时，使用的软元件将变为 (直接指定的软元件号)+(变址寄存器的内容)。
例如，指定了 D2Z2 时，如果 Z2 的内容为 3，则 D(2+3)=D5，D5 成为指定的软元件。

(b) 只有在使用通用型 QCPU 时，才可以使用 16 位变址寄存器和 32 位变址寄存器进行变址修饰。

(2) 通过 16 位变址寄存器进行的变址修饰

(a) 在 16 位范围内进行变址修饰时
各变址寄存器均可在 -32768 至 32767 的范围内进行设置。
变址修饰按照以下方式执行：



(b) 可进行变址修饰的软元件

除去下面所列的限制以外，变址修饰可以应用于触点、线圈、基本指令和应用指令中使用的软元件。

1) 不能进行变址修饰的软元件

软元件	内容
K、H	32 位常数
E	浮点数据
$	字符串数据
□.□	字软元件的位指定
FX、FY、FD	功能软元件
P	作为标签的指针
I	作为标签的中断指针
Z	变址寄存器
S	步进继电器
TR	SFC 传送软元件 *1
BL	SFC 块软元件 *1

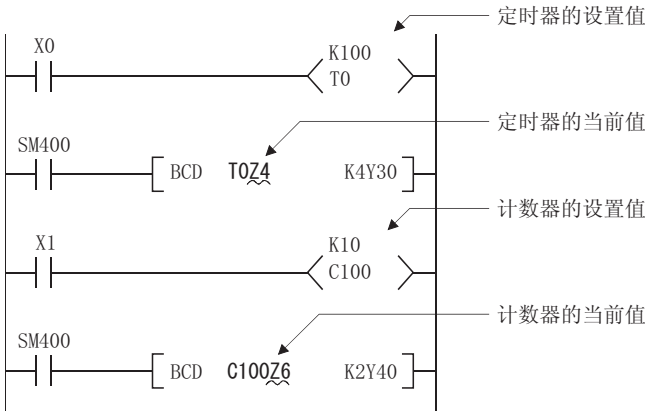
*1: SFC 传送软元件和 SFC 块软元件是供 SFC 使用的软元件。
关于如何使用这些软元件，请参阅以下手册：
・QCPU(Q 模式)/QnACPU 编程手册 (SFC 篇)

2）对使用变址寄存器有限制的软元件

软元件	内容	使用示例
T	· 只有 Z0 和 Z1 可以用作定时器的触点和线圈。	T0Z0 K100 T1Z1
C	· 只有 Z0 和 Z1 可以用作计数器的触点和线圈。	C0Z1 K100 C1Z0

备注

对于定时器和计数器的当前值，变址寄存器编号的使用无限制。



(c) 进行了变址修饰时及实际处理软元件的情况如下所示：
（当 Z0=20，Z1=-5 时）

梯形图示例	实际处理软元件
X0 MOV K20 Z0 MOV K－5 Z1 X1 MOV K2X50Z0 K1M38Z1	X1 MOV K2X64 K1M33 说明 K2X50Z0 · · · · · K2X(50+14)=K2X64 将K20转换为16进制数 K1M38Z1 · · · · · K1M(38-5)=K1M33
X0 MOV K20 Z0 MOV K－5 Z1 X1 MOV D0Z0 K3Y12FZ1	X1 MOV D20 K3Y12A 说明 D0Z0 · · · · · D(0+20)=D20 K3Y12FZ1 · · · · · K3Y(12F－5)=K3Y12 16进制数

图 3.7 梯形图示例及实际处理软元件

(3) 通过 32 位变址寄存器进行的变址修饰 (只对于通用型 QCPU)

通过 32 位进行变址修饰时的变址寄存器的指定方法可以从以下 2 种中选择：

· 指定 32 位变址修饰中使用的变址寄存器的范围。

· 通过 “ ZZ ” 表示指定 32 位变址修饰。

要 点

通过 “ ZZ ” 表示的 32 位变址修饰只能在版本为 8.68W 以后的 GX Developer 中使用，且只能用于以下 CPU 模块：

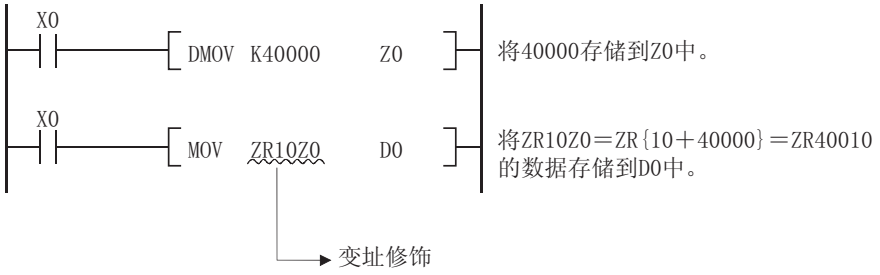
· 序列号的前 5 位数为 “ 10042 ” 以后的 QnU(D)(H)CPU

· QnUDE(H)CPU

(a) 对使用 32 位变址修饰的变址寄存器的范围进行指定时

1) 各变址修饰寄存器的设置范围为 -2147483648 ～ 2147483647。

变址修饰的情况如下所示：



2) 指定方法

通过 32 位变址寄存器进行变址修饰时，在 GX Developer 的可编程控制器参数的软元件设置中，中指定使用的变址寄存器的起始号。

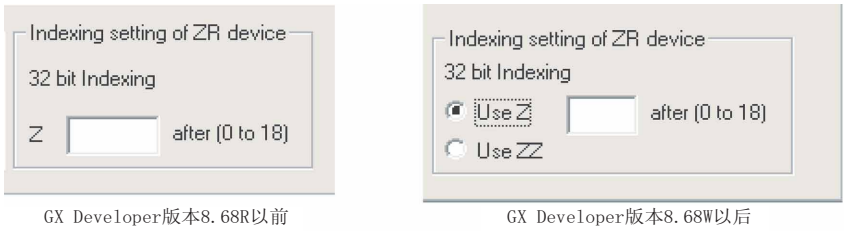


图 3.8 ZR 软元件的变址修饰设置参数的设置画面

要 点

当在可编程控制器参数的软元件设置中，将使用的变址寄存器的起始号进行了更改时，不要只修改参数或者进行可编程控制器写入。必须是与程序一道进行可编程控制器写入。

如果进行强制写入，将会发生 CAN'T EXE. PRG 错误。 (出错代码 :2500)

3）可进行变址修饰的软元件

变址修饰只能使用如下所示的软元件：

软元件	内容
ZR	连号访问方式文件寄存器
D	扩展数据寄存器
W	扩展链接寄存器

4）变址寄存器的使用范围

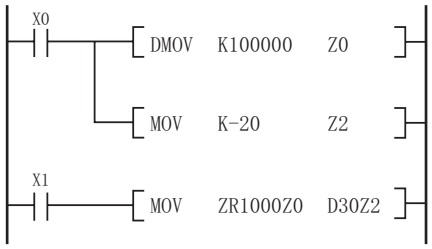
下表列出了通过 32 位变址寄存器进行变址修饰时的变址寄存器可用范围。
在通过 32 位变址寄存器进行变址修饰时，使用指定的变址寄存器 (Zn) 和紧接着的下一个变址寄存器 (Zn+1)，因此请注意防止所使用的变址寄存器重叠。

设置值	使用的变址寄存器	设置值	使用的变址寄存器
Z0	Z0，Z1	Z10	Z10，Z11
Z1	Z1，Z2	Z11	Z11，Z12
Z2	Z2，Z3	Z12	Z12，Z13
Z3	Z3，Z4	Z13	Z13，Z14
Z4	Z4，Z5	Z14	Z14，Z15
Z5	Z5，Z6	Z15	Z15，Z16
Z6	Z6，Z7	Z16	Z16，Z17
Z7	Z7，Z8	Z17	Z17，Z18
Z8	Z8，Z9	Z18	Z18，Z19
Z9	Z9，Z10	Z19	禁止使用

5）进行了变址修饰时及实际的处理软元件如下所示：

（当 Z0(32 位)=100000，Z2(16 位)=-20 时）

梯形图示例



实际的处理软元件

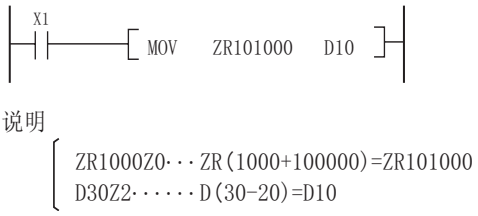
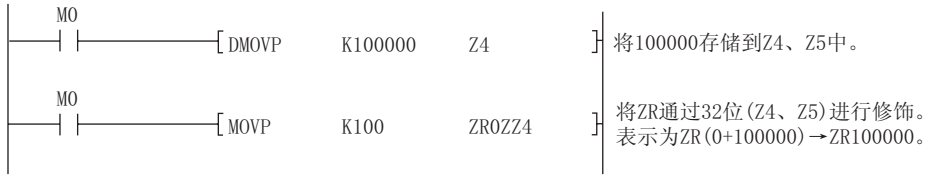


图 3.9 梯形图示例及实际的处理软元件

(b) 通过“ ZZ ”表示指定 32 位变址修饰时

1）通过表示为“ ZR0ZZ4 ”的“ ZZ ”表示对变址修饰进行指定，可以通过任意的变址寄存器进行 32 位变址修饰指定。

通过“ ZZ ”表示的 32 位变址修饰的示例如下：



2）指定方法

通过“ ZZ ”表示进行 32 位变址修饰时，在 GX Developer 的可编程控制器参数的“ ZR 软元件的变址修饰设置 ”中设置“ 使用 ZZ ”。

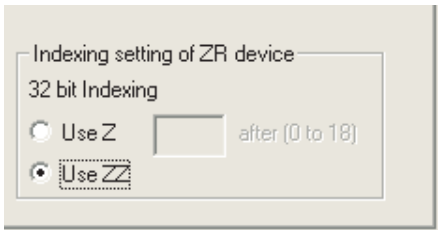


图 3.10 ZR 软元件的变址修饰设置参数的设置画面

3）可进行变址修饰的软元件

变址修饰只能使用如下所示的软元件：

软元件	内容
ZR	连号访问方式文件寄存器
D	扩展数据寄存器
W	扩展链接寄存器

4）变址寄存器的使用范围

下表列出了通过“ ZZ ”表示进行 32 位变址修饰时的变址寄存器可用范围。

指定通过“ ZZ ”表示进行 32 位变址修饰时，以 ZRmZZn 的形式进行指定。

通过指定 ZRmZZn，将 ZRm 的软元件号以 Zn，Zn+1 的 32 位值进行修饰。

“ ZZ ”表示 *1	使用的变址寄存器	“ ZZ ”表示 *1	使用的变址寄存器
□ZZ0	Z0，Z1	□ZZ10	Z10，Z11
□ZZ1	Z1，Z2	□ZZ11	Z11，Z12
□ZZ2	Z2，Z3	□ZZ12	Z12，Z13
□ZZ3	Z3，Z4	□ZZ13	Z13，Z14
□ZZ4	Z4，Z5	□ZZ14	Z14，Z15
□ZZ5	Z5，Z6	□ZZ15	Z15，Z16
□ZZ6	Z6，Z7	□ZZ16	Z16，Z17
□ZZ7	Z7，Z8	□ZZ17	Z17，Z18
□ZZ8	Z8，Z9	□ZZ18	Z18，Z19
□ZZ9	Z9，Z10	□ZZ19	禁止使用

*1:□表示修饰对象的软元件名 (ZR、D.W)。

5) 通过 “ ZZ ” 表示进行了变址修饰时及实际的处理软元件如下所示：

(当 Z0(32 位)=100000，Z2(16 位)=-20 时)

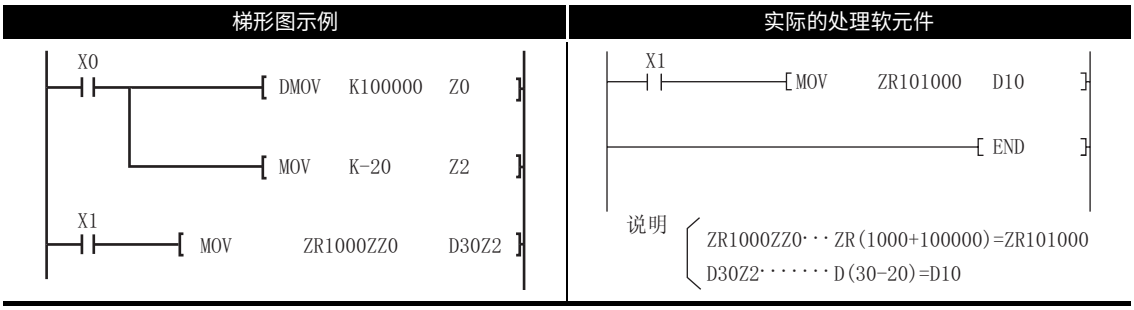


图 3.11 梯形图示例及实际的处理软元件

6) 可使用 “ ZZ ” 表示的功能

在下表所示的 GX Developer 的功能中，可以使用通过 “ ZZ ” 表示进行 32 位变址修饰指定。

编号	功能名称 · 说明
1	通过程序中的指令进行软元件指定
2	软元件登录监视
3	软元件测试
4	带执行条件的软元件测试
5	监视条件设置
6	采样跟踪 (跟踪点 (软元件指定)、跟踪对象软元件)

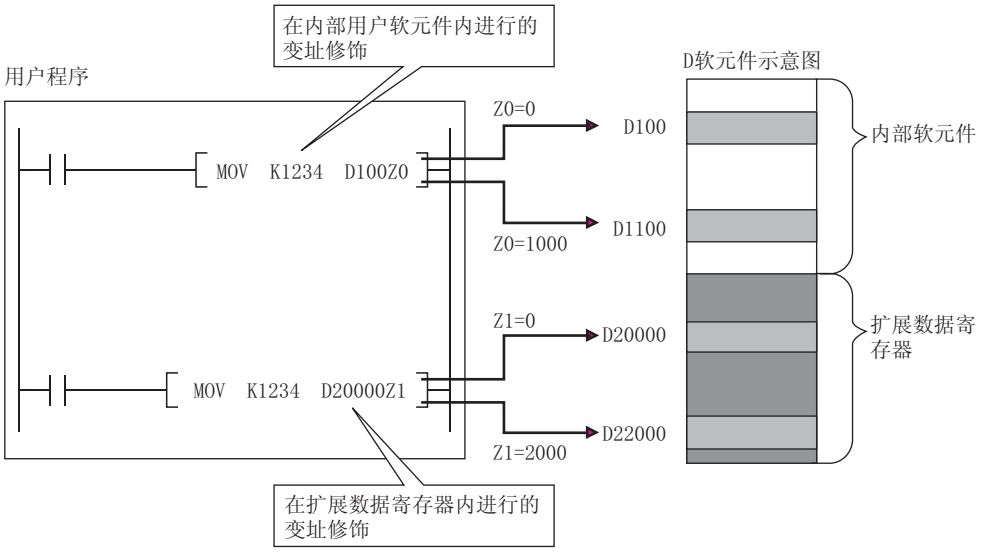
要 点

不能将单个的 ZZn 作为软元件处理，例如 “ DMOV K100000 ZZ0 ”。 为了通过 “ ZZ ” 表示进行 32 位变址修饰指定，对变址寄存器进行值的设置时，应对 Zn(Z0 ～ Z19) 进行设置。

在 GX Developer 的各功能中，不能输入单个的 ZZn。

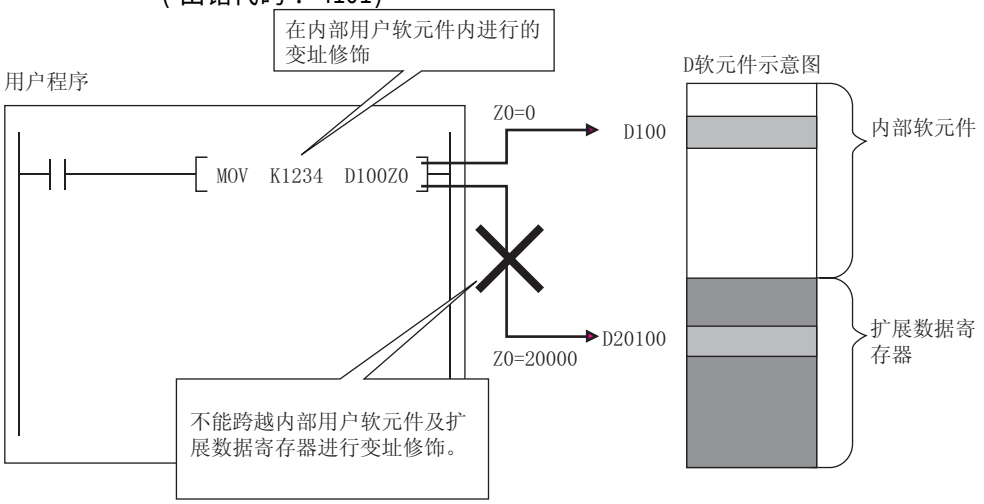
(4) 通过扩展数据寄存器 (D)、扩展链接寄存器 (W) 进行变址修饰时 (通用型 QCPU)

与通过内部用户软元件的数据寄存器 (D)、扩展链接寄存器 (W) 进行的变址修饰相同，可以在扩展数据寄存器 (D)、扩展链接寄存器 (W) 的范围内通过变址修饰进行软元件指定。



1) 跨越内部用户软元件及扩展数据寄存器 (D)、扩展链接寄存器 (W) 的变址修饰

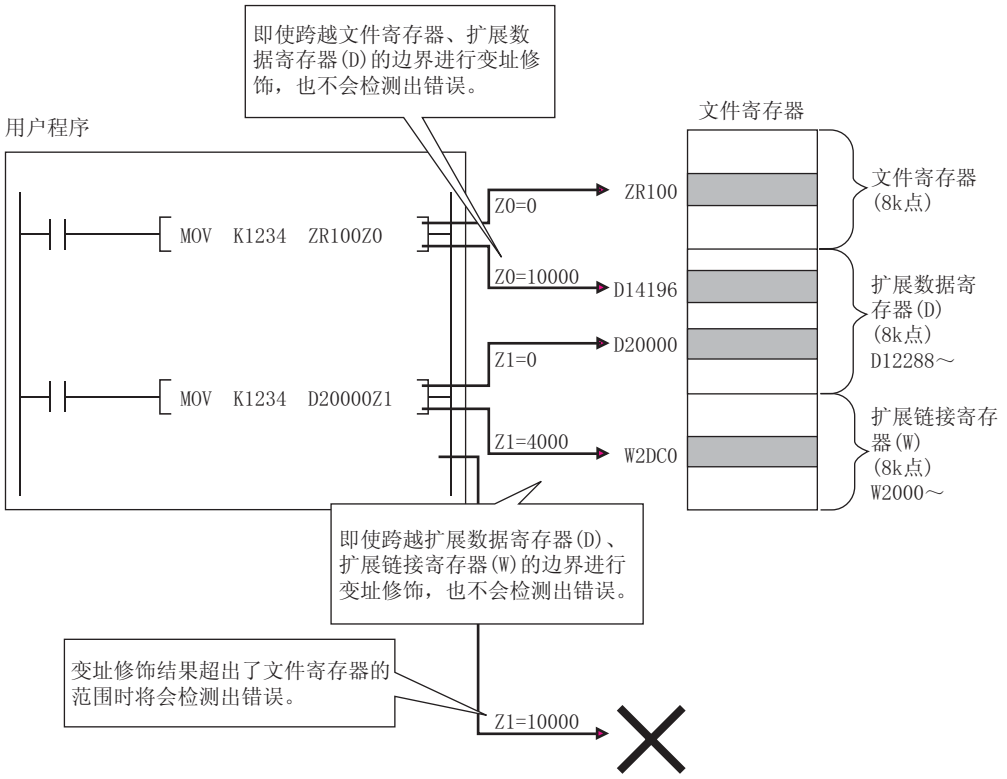
不能跨越内部用户软元件及扩展数据寄存器 (D)、扩展链接寄存器 (W) 进行变址修饰指定。变址修饰时的软元件范围检查有效的情况下，将变为出错状态。(出错代码： 4101)



2) 跨越文件寄存器 (ZR)、扩展数据寄存器 (D)、扩展链接寄存器 (W) 的变址修饰

即使跨越文件寄存器 (ZR)、扩展数据寄存器 (D)、扩展链接寄存器 (W) 进行变址修饰，也不会变为出错状态。

但是，如果文件寄存器 (ZR)、扩展数据寄存器 (D)、扩展链接寄存器 (W) 的变址修饰结果改变了文件寄存器的范围，将会变为出错状态。(出错代码： 4101)

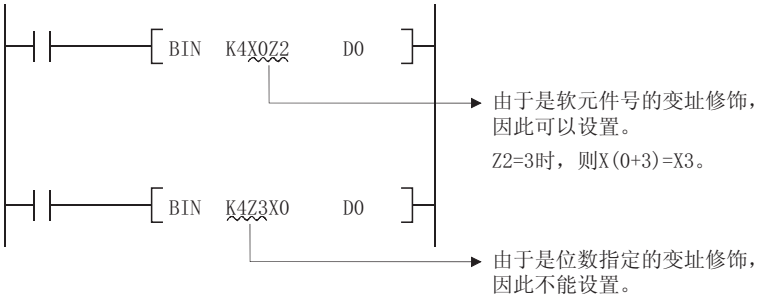


(5) 进行其它的变址修饰时

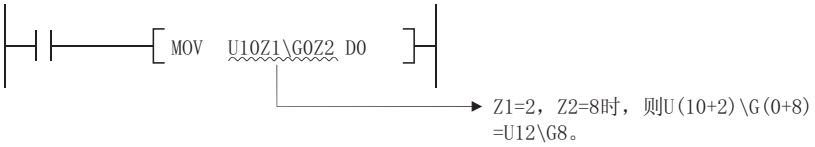
(a) 位数据

在进行了位数指定时，可以进行软元件号的变址修饰。

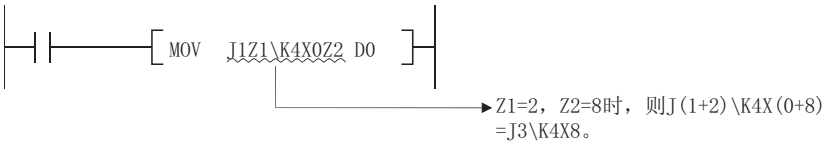
但是，不能进行位数指定的变址修饰。



(b) 在智能功能模块软元件 *1 中，对智能功能模块的起始 I/O 地址号及缓冲存储器地址均可以进行变址修饰。

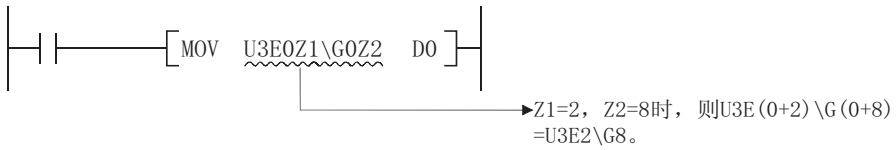


(c) 在链接直接软元件 *1 中，对网络号及软元件号均可以进行变址修饰。



*1: 关于智能功能模块软元件、链接直接软元件，请参阅所使用的 CPU 模块的用户手册（功能解说 / 程序基础篇）。

(d) 在多 CPU 共享软元件 *2 中，对 CPU 模块的起始 I/O 地址号和 CPU 共享存储器地址均可以进行变址修饰。



*2: 关于多 CPU 共享软元件，请参阅所使用的 CPU 模块的用户手册（功能解说 / 程序基础篇）。

(e) 通过扩展数据寄存器 (D)、扩展链接寄存器 (W) 的 32 位进行变址修饰时 (通用型 QCPU)
进行扩展数据寄存器 (D)、扩展链接寄存器 (W) 的变址修饰时，与文件寄存器 (ZR) 的变址修饰相同，可以使用下述 2 种方法进行 32 位的变址修饰：

· 指定用于 32 位变址修饰的变址寄存器。

· 指定通过 “ ZZ ” 表示的 32 位变址修饰。

要 点

通过 “ ZZ ” 表示的 32 位变址修饰只能在版本为 8.68W 以后的 GX Developer 中使用，且只能用于以下 CPU 模块：

· 序列号的前 5 位数为 “ 10042 ” 以后的 QnU(D)(H)CPU

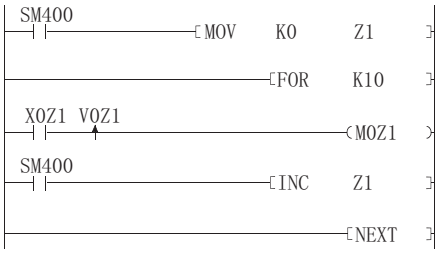
· QnUDE(H)CPU

(6) 注意事项

(a) 在 FOR ～ NEXT 指令之间进行变址修饰时
通过在 FOR ～ NEXT 指令之间使用变址继电器 (V)，可以进行脉冲输出。
但是，不能进行使用 PLS/PLF/ 脉冲化 (□ P) 指令的脉冲输出。

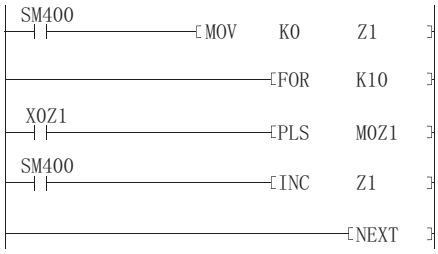
[使用变址继电器时]

(MOZ1 可以进行正常的脉冲输出。)



[未使用变址继电器时]

(MOZ1 不能进行正常的脉冲输出。)



备 注

通过变址继电器 V0Z1 记忆 X0Z1 的 ON/OFF 信息。
例如，X0 的 ON/OFF 信息由 V0 记忆，X1 的 ON/OFF 信息由 V1 记忆。

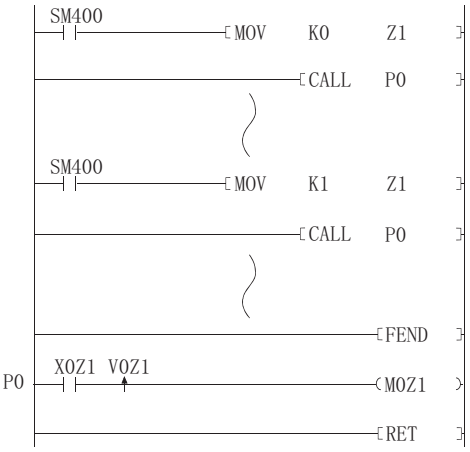
(b) 通过 CALL 指令进行变址修饰时

在 CALL 指令中通过使用变址继电器 (V)，可以进行脉冲输出。

但是，不能通过 PLS/PLF/ 脉冲 (□ P) 指令进行脉冲输出。

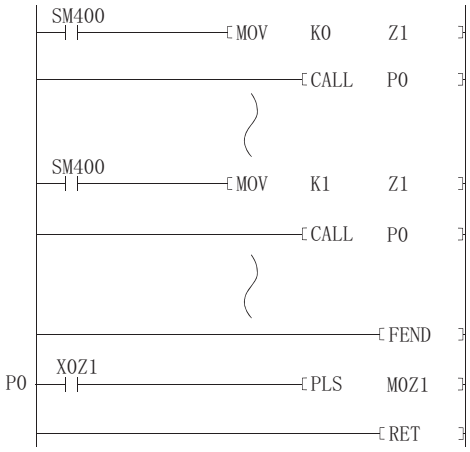
[使用变址继电器时]

(MOZ1 可以进行正常的脉冲输出。)



[未使用变址继电器时]

(MOZ1 不能进行正常的脉冲输出。)



(c) 变址修饰时的软元件范围检查

1) 使用除通用型 QCPU 以外的 CPU 时

在进行变址修饰时不进行软元件范围检查。

因此，进行了变址修饰后的结果超出了用户指定的软元件范围时，不会发生错误，而是将数据写入到其它软元件中。(但是，如果进行了变址修饰后的结果超出了用户软元件范围，被写入到系统用的软元件中时，将会变为出错状态。(出错代码： 1103))

在进行编程的过程中使用变址修饰时，应特别加以注意。

2) 使用通用型 QCPU 时

在进行变址修饰时，进行软元件范围检查。

此外，通过更改 GX Developer 的可编程控制器参数设置，也可以不进行软元件范围检查。

(d) 16 位 ⇔ 32 位变址修饰范围的变更

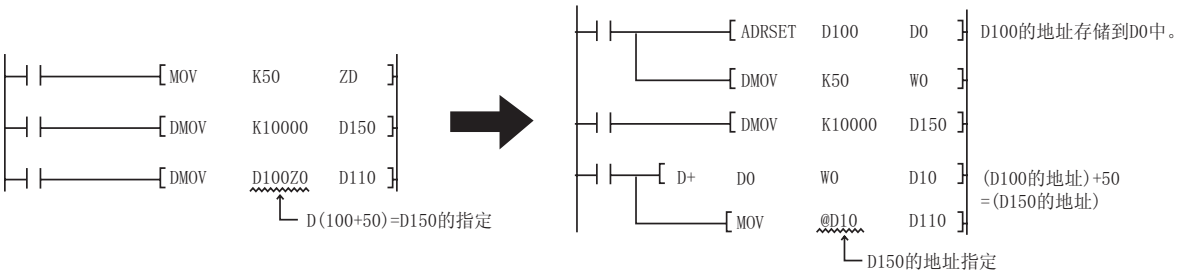
进行 16 位 ⇔ 32 位变址修饰范围的变更时，应重新确认程序内的变址修饰位置。

在 32 位范围的变址修饰中，使用指定的变址寄存器 (Zn) 和紧接着的下一个变址寄存器 (Zn+1)，因此应注意避免所使用的变址寄存器重叠。

3.4 间接指定

(1) 间接指定

(a) 间接指定是指，通过 2 个字的字软元件（字软元件的 2 点）对顺控程序中使用的软元件地址进行指定的方法。
在变址寄存器不足的情况下，应使用这种方法进行变址修饰。



［使用变址寄存器时］　　［使用间接指定时］

(b) 用于进行指定软元件地址指定的软元件是通过“@+(字软元件号)”来指定的。
例如，指定为 @D100 时，D101、D100 的内容将成为软元件地址。

(c) 进行间接指定的软元件地址可以通过 ADRSET 指令进行确认。
关于 ADRSET 指令，请参阅 7.17.6 项。

(2) 可间接指定的软元件

可进行间接指定的 CPU 模块软元件如表 3.3 所示：

表 3.3 可间接指定的软元件一览表

软元件分类		可否间接指定	间接指定示例
内部用户软元件	位软元件 *1	否	----
	字软元件 *1	可	· @D100 · @D100Z2 *2
链接直接软元件	位软元件 *1	否	----
	字软元件 *1	可 *3	· @J1\W10 · @J1Z1\W10Z2 *2
智能功能模块软元件		可 *3	· @U10\G0 · @U10Z1\G0Z2 *2
变址寄存器		否	----
文件寄存器		可	· @R0,@ZR20000 · @R0Z1,@ZR20000Z1 *2
扩展数据寄存器		可	· @D1000 · @W1000
扩展链接寄存器			
嵌套		否	----
指针			----
常数			----
其它	SFC 块软元件		----
	SFC 传送软元件		
	网络号指定软元件		
	I/O 号指定软元件		

*1: 关于软元件名称，请参阅所使用的 CPU 模块的用户手册（功能解说 / 程序基础篇）。
*2: 表示通过变址寄存器进行变址修饰时。
*3: 可以进行间接指定，但是不能通过 ASRSET 指令进行地址写入。

(3) 注意事项

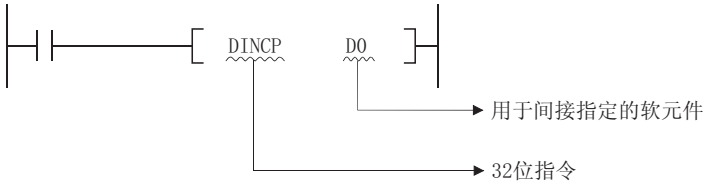
(a) 间接指定的地址

间接指定的地址是通过 2 字进行指定的。

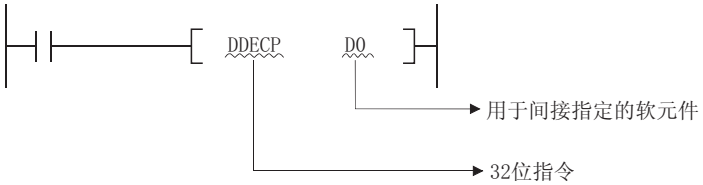
因此，用间接指定替代变址修饰时，应进行 32 位的加减运算。

对存储在 D1 和 D0 中的间接指定用软元件地址进行加减运算的梯形图如下所示。

[间接指定的软元件地址 +1 时]



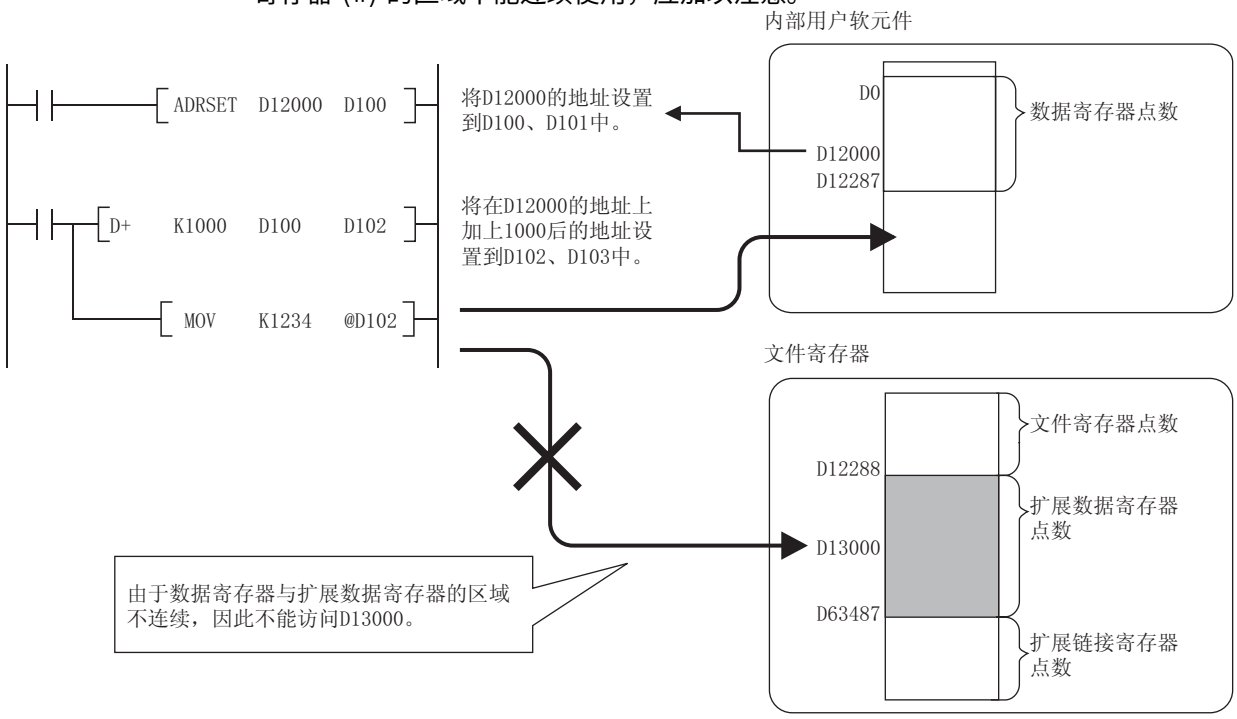
[间接指定的软元件地址 -1 时]



(b) 扩展数据寄存器 (D)、扩展链接寄存器 (W) 的间接指定

扩展数据寄存器 (D)、扩展链接寄存器 (W) 可以通过间接地址进行间接指定。

通过内部用户软元件的数据寄存器 (D)、链接寄存器 (W) 及扩展数据寄存器 (D)、扩展链接寄存器 (W) 进行间接指定时，由于内部用户软元件与扩展数据寄存器 (D)、扩展链接寄存器 (W) 的区域不能连续使用，应加以注意。



3.5 缩短指令处理时间

3.5.1 子集处理

子集处理用来对基本指令及应用指令中使用的软元件设置限制，目的是为了提高处理速度。
但是，指令符号等不发生变化。
希望缩短扫描时间时，应在如下所示的条件下执行指令。

(1) 子集处理中各软元件应满足的条件

(a) 使用字数据时

软元件	条件
位软元件	· 以 16 的倍数指定位软元件号。 · 位数指定只能指定为 K4。 · 不进行变址修饰。
字软元件	· 内部用户软元件 · 文件寄存器 (R、ZR*1) · 多 CPU 共享软元件 *1、*2 · 变址寄存器 (Z)/ 通用运算寄存器 (Z)*1
常数	· 无限制

(b) 使用双字数据时

软元件	条件
位软元件	· 以 16 的倍数指定位软元件号。 · 位数指定只能指定为 K8。 · 不进行变址修饰。
字软元件	· 内部用户软元件 · 文件寄存器 (R、ZR*1) · 多 CPU 共享软元件 *1、*2 · 变址寄存器 (Z)/ 通用运算寄存器 (Z)*1
常数	· 无限制

(c) 使用位数据时

软元件	条件
位软元件	· 内部用户软元件 (可进行变址修饰)
字软元件	· 内部用户软元件的位指定 · 寄存器 (R、ZR*1) 的位指定 · 多 CPU 共享软元件的位指定 *1、*2

*1: 只对应于通用型 QCPU。

*2: 只对多 CPU 高速通信区 (U3En\G10000 ～) 有效。
但是，对 CPU 模块的起始 I/O 地址号进行了变址修饰 (U3EnZn\G10000) 时除外。

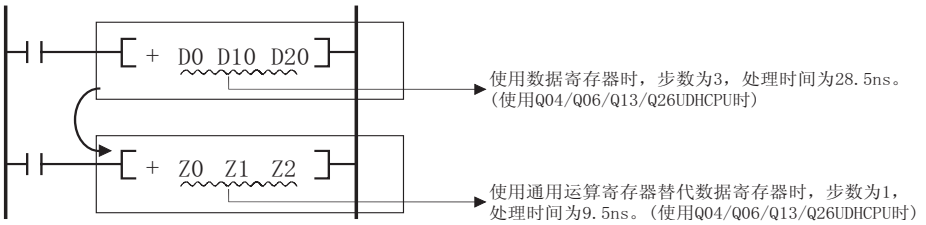
(2) 可进行子集处理的指令

指令分类	指令符号
触点指令	LD、LDI、AND、ANI、OR、ORI、LDP、LDF、ANDP、ANDF、ORP、ORF
输出指令	OUT、SET、RST
比较运算指令	·=、<>、<、<=、>、>=、D=、D<>、D<、D<=、D>、D>=
算术运算	·+、—、*、/、INC、DEC、D+、D—、D*、D/、DINC、DDEC ·B+、B—、B*、B/、E+、E-、E*、E/
数据转换指令	·BCD、BIN、DBCD、DBIN、FLT、DFLT、INT、DINT
数据传送指令	·MOV、DMOV、CML、DCML、XCH、DXCH ·FMOV、BMOV、EMOV(仅 QCPU)
程序分支指令	·CJ、SCJ、JMP
逻辑运算	·WAND、DAND、WOR、DOR、WXOR、DXOR、WXNR、DXNR
旋转指令	·RCL、DRCL、RCR、DRCR、ROL、DROL、ROR、DROR
移位指令	·SFL、DSFL、SFR、DSFR
数据处理指令	·SUM、SEG
结构化指令	·FOR、CALL

3.5.2 使用通用运算寄存器 (Z) 的运算处理 (只对于通用型 QCPU)

使用通用运算寄存器 (Z) 可以缩短运算处理时间。

使用通用运算寄存器 (Z) 的程序示例如下所示：



通过可以进行子集处理的指令，可以缩短运算处理时间。

关于步数的变更，请参阅 3.8 节。

关于各指令的运算时间，请参阅附录 1。

要 点

由于通用运算寄存器与变址寄存器是相同的软元件，因此在进行变址修饰时，不要使通用运算寄存器的软元件号与变址寄存器的软元件号重复。

3.6 编程注意事项

在 CPU 模块中执行基本指令和应用指令时，在下列情况下将会发生运算错误：

- 发生了各指令的说明页面中记载的出错时。
- 使用智能功能模块软元件时，指定的 I/O 地址号的位置上未安装智能功能模块。
- 使用智能功能模块软元件时，指定的缓冲存储器地址不存在。
- 使用链接软元件时，相应的网络不存在。
- 使用链接软元件时，在指定的 I/O 地址号位置上未安装网络模块。
- 使用多 CPU 共享软元件时，在指定 CPU 模块的起始 I/O 地址号的位置上未安装 CPU 模块。
- 使用多 CPU 共享软元件时，指定的共享存储器地址不存在。
- 进行了跨越内部用户软元件及扩展数据寄存器 (D)、扩展链接寄存器 (W) 的设置 (仅通用型 QCPU)

要 点

进行了文件寄存器设置但未安装存储卡时，或在未进行文件寄存器设置的状态下对文件寄存器进行了的读取 / 写入操作时的情况如下所示：

(1) 对于高性能型 QCPU、过程 CPU、冗余 CPU

即使对文件寄存器进行读取 / 写入操作，也不会发生错误。但是，如果从文件寄存器中读取，“ 0H ”将被存储。

(2) 对于通用型 QCPU

如果对文件寄存器执行读取 / 写入操作，将会发生 OPERATION ERROR(出错代码 : 4101)。

(1) 软元件范围检查

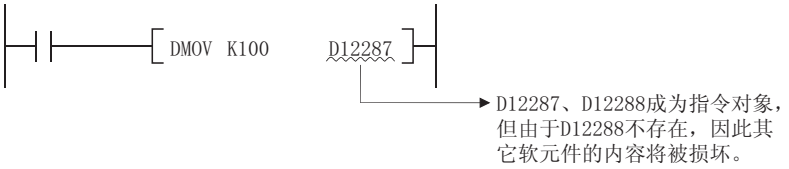
在 CPU 模块中，基本指令和应用指令中使用的软元件范围检查的情况如下所示：

(a) 用于处理固定长度软元件的指令 (MOV、DMOV 等)

1) 对于除通用型 QCPU 以外的 CPU

不进行软元件范围检查。如果超出了相应软元件范围，数据将被写入到其它软元件中。*1

例如，在数据寄存器被分配了 12k 点时，即使超过了 D12287，也不会变为出错状态。



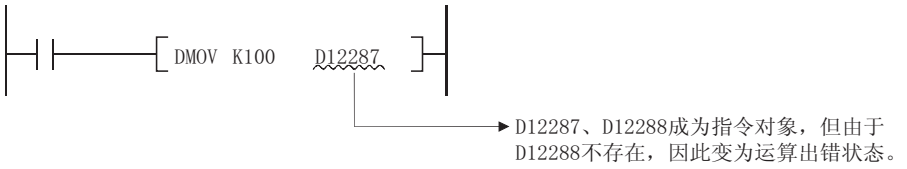
当执行了变址修饰时，也不进行软元件范围检查。

如果变址修饰的执行结果超出了相应的软元件范围，数据将被写入到其它软元件中。*1

*1: 关于内部软元件的分配顺序，请参阅本项 (c) 字符串数据。

2) 对于通用型 QCPU

进行软元件范围检查。超出了相应软元件范围时，将会变为运算出错状态。
例如，如果数据寄存器被分配了 12k 点时，如果超过了 D12287，将会变为出错状态。



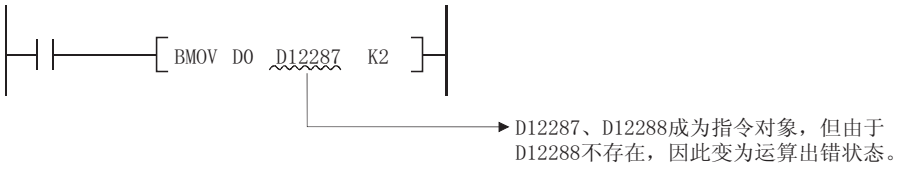
执行了变址修饰时也将进行软元件范围检查。
此外，通过更改 GX Developer 的可编程控制器参数设置，可以设置为不进行软元件范围检查。*2

*2: 关于将设置变更为不进行软元件范围检查的有关内容，请参阅所使用的 CPU 模块的用户手册（功能解说 / 程序基础篇）。

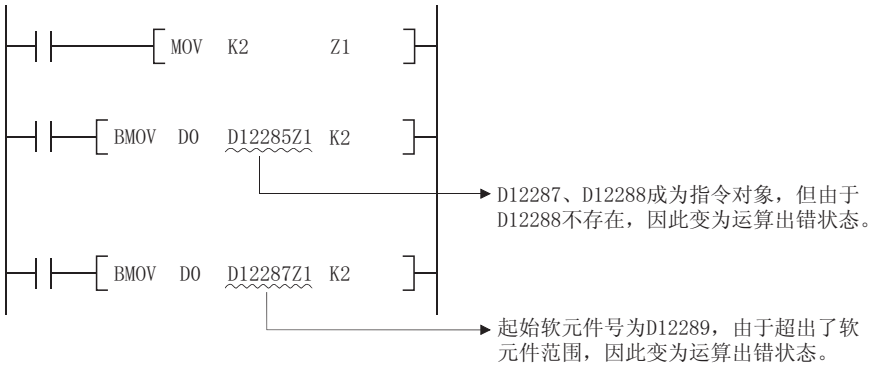
(b) 用于处理可变长度软元件的指令（指定传送数量的 BMOV、FMOV 等）

1) 对于除通用型 QCPU 以外的 CPU

进行软元件范围检查。
如果超出了相应软元件范围，将发生运算错误。
例如，数据寄存器被分配了 12k 点时，如果超过了 D12287，将会变为出错状态。

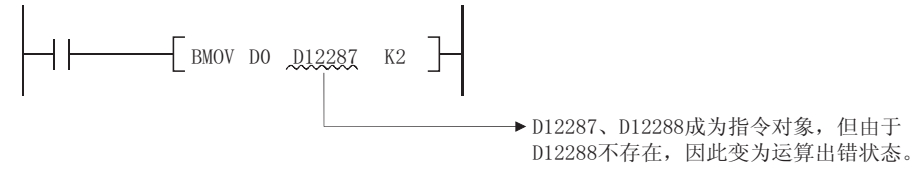


当执行了变址修饰时也将进行软元件范围检查。
但是，即使由于变址修饰导致起始软元件号超出了相应软元件范围，也不会变为出错状态。

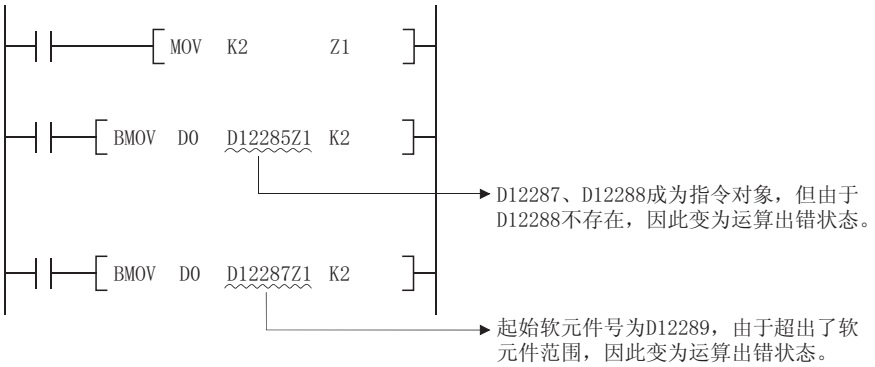


2) 对于通用型 QCPU

进行软元件范围检查。超出了相应软元件范围时，将发生运算错误。
例如，当数据寄存器被分配了 12k 点时，如果软元件号超过了 D12287，将会变为出错状态。



进行了变址修饰时也将进行软元件范围检查。
由于变址修饰的结果导致起始软元件号超出了相应软元件范围时，将变为出错状态。

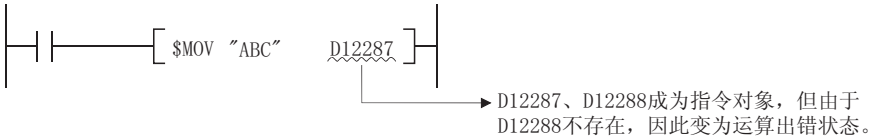


此外，通过更改 GX Developer 的可编程控制器参数设置，可以设置为不进行软元件范围检查。*2

*2: 关于将设置变更为不进行软元件范围检查的有关内容，请参阅所使用的 CPU 模块的用户手册（功能解说 / 程序基础篇）。

(c) 字符串数据

因为所有的字符串数据长度可变，因此将进行软元件的范围检查。
如果超出了相应的软元件范围，将发生运算错误。
例如，数据寄存器被分配了 12k 点时，如果超过了 D12287，将发生错误。



但是，在除通用型 QCPU 以外的 CPU 中，当执行了变址修饰时，在起始软元件号超出了软元件范围的情况下也不会发生运算错误，而是将访问其它的软元件。
在通用型 QCPU 中进行了下述访问时，将会发生错误。(出错代码： 4101)

1) 由于变址修饰，超出了软元件的边界 (区域 A 的范围) 的访问
各软元件的分配顺序如下所示：

2) 由于变址修饰，超出了文件寄存器的边界的访问

3) 未进行文件寄存器文件设置时，对文件寄存器 (R、ZR) 进行的访问

4) 对超出了文件寄存器文件的范围的文件寄存器 (R、ZR) 进行的访问

此外，通过在可编程控制器参数中设置为变址修饰时不进行软元件范围检查，则可实现即使进行了上述 1) ～ 4) 的访问也不会检测出错误。

但是，根据通用型 QCPU 的序列号，其动作情况如下表所示。*2

变址修饰时的软元件范围设置	通用型 QCPU 的序列号的前 5 位数	
	序列号 “ 10021 ” 以前	序列号 “ 10022 ” 以后
实施	进行上述 1) ～ 4) 的访问时检测出错误。	
不实施	进行上述 2) ～ 4) 的访问时检测出错误。	不检测出错误。

*2: 关于如何将变址修饰时的软元件范围检查设置为不进行，请参阅所使用的 CPU 模块的用户手册 (功能解说 / 程序基础篇)。

要 点

对于通用型 QCPU，不能指定跨越内部用户软元件 (SW) 与文件寄存器 (R) 之间的区域的变址修饰。(否则将发生错误，出错代码： 4101)

备注

关于内部用户软元件的分配的变更，请参阅所使用的 CPU 模块的用户手册 (功能解说 / 程序基础篇)。

(d) 如果通过直接的访问输出 (DY) 进行了变址修饰，将进行软元件范围检查。

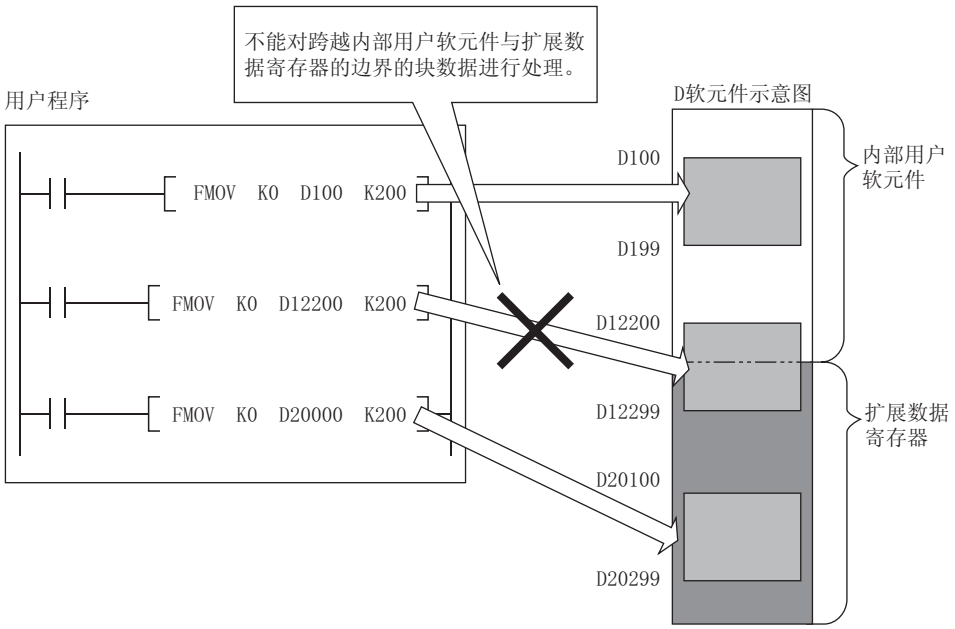
(e) 使用扩展数据寄存器 (D)、扩展链接寄存器 (W) 时的注意事项 (除 Q00UJCPU 以外的通用型 QCPU)

在下述指定方法中，不能跨越内部用户软元件与扩展数据寄存器 (D)、扩展链接寄存器 (W) 的边界进行指定。如果进行了指定，将发生出错 OPERATION ERROR(出错代码： 4101)。

· 变址修饰指定

· 间接指定

· 通过块数据处理指令进行的指定 *1



*1: 块数据是指如下所示的数据：

· 通过以 FMOV、BMOV、BK+ 等多字数据为运算对象的指令所处理的数据

· 通过 SP.FWRITE、SP.FREAD 等指定的 2 字以上数据所构成的控制数据

· 32 位以上数据格式的数据 (BIN32 位、实数、软元件的间接地址)

(2) 软元件的数据检查

在 CPU 模块中，对于基本指令和应用指令中使用的软元件的数据检查情况如下所示：

(a) 使用 BIN 数据时
即使运算结果上溢或下溢，也不会变为出错状态。
此时进位标志也不会为 ON。

(b) 使用 BCD 数据时
1) 将对各个位进行是否为 BCD 值 (0 ～ 9) 的检查。
如果某个位超出了 0 ～ 9 的范围 (A ～ F)，将会变为出错状态。
2) 即使运算结果上溢或下溢，也不会变为出错状态。
此时进位标志同样也不会为 ON。

(c) 使用浮点数据时
1) 使用单精度浮点运算指令时，在运算结果为如下所示的情况下，将会变为运算出错状态。
1.0×2^{-127} 以下时
1.0×2^{128} 以上时
2) 使用双精度浮点运算指令时，在运算结果为如下所示的情况下，将会变为运算出错状态。
1.0×2^{-1023} 以下时
1.0×2^{1024} 以上时

(d) 使用字符串数据时
不进行数据检查。

(3) 至缓冲存储器的访问
对缓冲存储器进行访问时，建议通过使用了智能功能模块软元件 (Un\G0 ～) 的指令进行访问。

(4) 至多 CPU 共享存储器的访问
对多 CPU 共享存储器进行访问时，建议通过使用了多 CPU 共享软元件 (U3En\G10000 ～) 的指令进行访问。

3.7 指令执行条件

对于 CPU 模块的顺控指令、基本指令和应用指令，存在以下 4 种类型的执行条件。

· 常时执行 ..... 执行的指令与软元件 ON/OFF 状态无关。

例 LD X0、OUT Y10

· 在 ON 时执行..... 在输入条件为 ON 的状态下执行的指令。

例 MOV 指令、FROM 指令

· 在上升沿执行..... 只在输入条件的上升沿 (OFF → ON) 时执行的指令。
PLS 指令、MOVP 指令

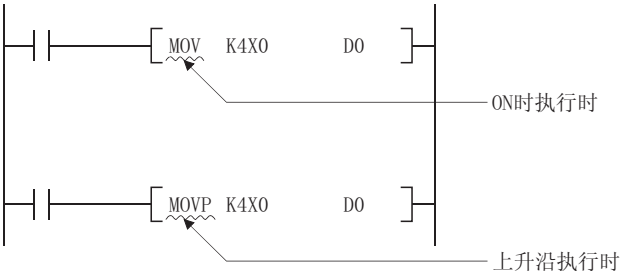
· 在下降沿执行..... 只在输入条件的下降沿 (ON → OFF) 时执行的指令。
PLF 指令

在相当于线圈的基本指令或应用指令中，如果同一指令可以在“ ON 时执行 ”或者在“ 上升沿执行 ”，则在指令名称后附加一个“ P ”用以区别执行条件。

· ON 时执行的指令 指令名称

· 上升沿执行的指令 指令名称 +P

在 MOV 指令中，对 ON 时执行或上升沿执行进行指定的方式如下所示：

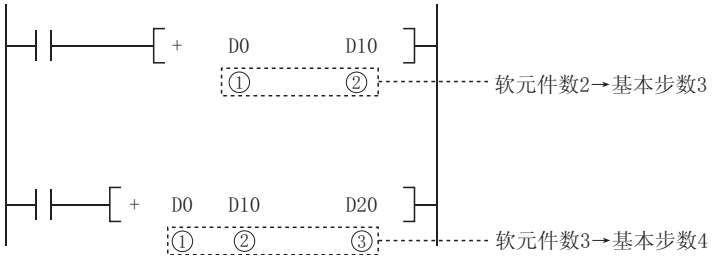


3.8 计算步数

根据使用的软元件、是否进行了间接设置等，CPU 模块的顺控指令、基本指令和应用指令的步数有所不同。

(1) 计算基本步数

基本指令和应用指令的基本步数为 (软元件数 +1) 步。
例如，“ + 指令 ”时的情况如下所示：



(2) 步数增加的条件

当使用软元件的间接指定或增加了步数的软元件时，步数将会超过基本步数。

(a) 软元件的间接指定时
当通过 @ 进行了间接指定时，步数在基本步数上增加 1 步。
例如： 当一个 3 步的 MOV 指令被间接指定时 (例： MOV K4X0@D0)，将会增加 1 步而变为 4 步。

(b) 增加了步数的软元件 (通用型 QCPU 除外)

增加了步数的软元件	增加的步数	示例
智能功能模块软元件	1	MOV U4\G10 D0
多 CPU 共享软元件		MOV U3E1\G0 D0
链接直接软元件		MOV J3\B20 D0
变址寄存器		MOV Z0 D0
连号访问方式文件寄存器		MOV ZR123 D0
32 位常数		DMOV K123 D0
实数常数		EMOV E0.1 D0
字符串常数	偶数时： 字符数÷ 2 奇数时： (字符数 +1) ÷ 2	$MOV “ 123 ” D0

(c) 增加了步数的软元件 (通用型 QCPU)

1) 进行了子集处理的指令

下表所示为进行了子集处理的指令中的软元件的步数。

指令符号	增加了步数的软元件	增加的步数 (指令步数)	基本步数
LD、LDI、AND、ANI、OR、ORI、 LDP、LDF、ANDP、ANDF、ORP、ORF	连号访问方式文件寄存器、 扩展数据寄存器 (D)、 扩展链接寄存器 (W)	1(2)	1
	多 CPU 共享软元件		
SET	连号访问方式文件寄存器、 扩展数据寄存器 (D)、 扩展链接寄存器 (W)	1(2)	1
	多 CPU 共享软元件		
OUT	定时器 · 计数器	3(4)	1
	连号访问方式文件寄存器、 扩展数据寄存器 (D)、 扩展链接寄存器 (W)	1(2)	
	多 CPU 共享软元件		
RST(位软元件)	连号访问方式文件寄存器、 扩展数据寄存器 (D)、 扩展链接寄存器 (W)	1(2)	1
	多 CPU 共享软元件		
RST(字软元件)	定时器 · 计数器 (位 / 字软元件)	2(4)	2
	连号访问方式文件寄存器、 扩展数据寄存器 (D)、 扩展链接寄存器 (W)	1(3)	
	多 CPU 共享软元件	1(3)	
LD=、LD<>、LD<、LD<=、LD>、LD>=、 AND=、AND<>、AND<、AND<=、AND>、AND>=、 OR=、OR<>、OR<、OR<=、OR>、OR>=	通用运算寄存器 *2	-1	3
	连号访问方式文件寄存器、 扩展数据寄存器 (D)、 扩展链接寄存器 (W)	1	
	多 CPU 共享软元件		
LDD=、LDD<>、LDD<、LDD<=、LDD>、LDD>=、 ANDD=、ANDD<>、ANDD<、ANDD<=、ANDD>、 ANDD>=、ORD=、ORD<>、ORD<、ORD<=、 ORD>、ORD>=	通用运算寄存器 *2	-1	3
	连号访问方式文件寄存器、 扩展数据寄存器 (D)、 扩展链接寄存器 (W)	1	
	多 CPU 共享软元件		
	10 进制常数、16 进制常数、实数常数		
+、-、+P、-P、WAND、WOR、WXOR、WXNR、 WANDP、WORP、WXORP、WXNRP (2 个软元件)	通用运算寄存器 *2	(D):-1	3
	连号访问方式文件寄存器、 扩展数据寄存器 (D)、 扩展链接寄存器 (W)	(S1):1、(D):3	
	多 CPU 共享软元件		
D+、D-、D+P、D-P、DAND、DOR、DXOR、DXNR、 DANDP、DORP、DXORP、DXNRP (2 个软元件)	通用运算寄存器 *2	(D):-1	3
	连号访问方式文件寄存器、 扩展数据寄存器 (D)、 扩展链接寄存器 (W)	(S1):1、(D):3	
	多 CPU 共享软元件		
	10 进制常数、16 进制常数、实数常数	(S1):1	

指令符号	增加了步数的软元件	增加的步数 （指令步数）	基本步数
+、-、+P、-P、WAND、WOR、WXOR、WXNR、 WANDP、WORP、WXORP、WXNRP (3 个软元件)*1	连号访问方式文件寄存器、 扩展数据寄存器 (D)、 扩展链接寄存器 (W)	ⓢ1、ⓢ2:1、ⓓ:2	3
	多 CPU 共享软元件		
D+、D-、D+P、D-P、DAND、DOR、DXOR、DXNR、 DANDP、DORP、DXORP、DXNRP (3 个软元件)*1	连号访问方式文件寄存器、 扩展数据寄存器 (D)、 扩展链接寄存器 (W)	ⓢ1、ⓢ2:1、ⓓ:2	3
	多 CPU 共享软元件		
	10 进制常数、16 进制常数、实数常数	ⓢ1、ⓢ2:1	
＊、＊P、/、/P	连号访问方式文件寄存器、 扩展数据寄存器 (D)、 扩展链接寄存器 (W)	ⓢ1、ⓢ2:1、ⓓ:2	3
	多 CPU 共享软元件		
D＊、D＊P、D/、D/P、E＊、E＊P	连号访问方式文件寄存器、 扩展数据寄存器 (D)、 扩展链接寄存器 (W)	ⓢ1、ⓢ2:1、ⓓ:2	3
	多 CPU 共享软元件		
	10 进制常数、16 进制常数、实数常数	ⓢ1、ⓢ2:1	
INC、INCP、DEC、DECP、DINC、DINCP、 DDEC、DDECP	变址寄存器 / 通用运算寄存器 *2	-1	2
	连号访问方式文件寄存器、 扩展数据寄存器 (D)、 扩展链接寄存器 (W)	3	
	多 CPU 共享软元件		

指令符号	增加了步数的软元件	增加的步数（指令步数）	基本步数
MOV、MOVP	连号访问方式文件寄存器、扩展数据寄存器 (D)、扩展链接寄存器 (W)	1	2
	多 CPU 共享软元件		
DMOV、DMOVP、EMOV、EMOVP	连号访问方式文件寄存器、扩展数据寄存器 (D)、扩展链接寄存器 (W)	1	2
	多 CPU 共享软元件		
	10 进制常数、16 进制常数、实数常数		
BCD、BCDP、BIN、BINP、FLT、FLTP、CML、CMLP	连号访问方式文件寄存器、扩展数据寄存器 (D)、扩展链接寄存器 (W)	ⓢ1:1、ⓢ2:2	2
	多 CPU 共享软元件		
DBCD、DBCDP、DBIN、DBINP、INT、INTP、DINT、DINTP、DFLT、DFLTP、DCML、DCMLP	连号访问方式文件寄存器、扩展数据寄存器 (D)、扩展链接寄存器 (W)	ⓢ1:1、ⓢ2:2	2
	多 CPU 共享软元件		
	10 进制常数、16 进制常数、实数常数	ⓢ1:1	

*1: 如果ⓢ1与ⓢ2使用相同的软元件，则基本步数将增加 1 步。

*2: 使用通用运算寄存器时，步数将减少。

在进行了子集处理的指令中，在指令的多处使用了通用运算寄存器时，则步数将减少。下表列出了上述情况下各指令的步数。

指令符号	指定通用运算寄存器的位置	增加的步数（指令步数）	基本步数
LD=、LD<>、LD<、LD<=、LD>、LD>=、AND=、AND<>、AND<、AND<=、AND>、AND>=、OR=、OR<>、OR<、OR<=、OR>、OR>=、LDD=、LDD<>、LDD<、LDD<=、LDD>、LDD>=、ANDD=、ANDD<>、ANDD<、ANDD<=、ANDD>、ANDD>=、ORD=、ORD<>、ORD<、ORD<=、ORD>、ORD>=	ⓢ1与ⓢ2	-2(1)	3
+、-、+P、-P、D+、D-、D+P、D-P、WAND、WOR、WXOR、WXNR、DAND、DOR、DXOR、DXNR、WANDP、WORP、WXORP、WXNRP、DANDP、DORP、DXORP、DXNRP（2 个软元件）	ⓢ1与ⓓ	-2(1)	3
+、-、+P、-P、D+、D-、D+P、D-P、WAND、WOR、WXOR、WXNR、DAND、DOR、DXOR、DXNR、WANDP、WORP、WXORP、WXNRP、DANDP、DORP、DXORP、DXNRP（3 个软元件）*1	ⓢ1与ⓢ2与ⓓ	-2(1)	3
	ⓢ1或者ⓢ2与ⓓ	-1(2)	
	ⓢ1与ⓢ2（只有在ⓓ中指定了不增加步数的软元件时）	± 0(3)	
	ⓢ1与ⓢ2（只有在ⓓ中指定了连号访问方式文件寄存器时）	+2(5)	
＊、＊P、/、/P	ⓢ1与ⓢ2与ⓓ	-2(1)	3
	ⓢ1或者ⓢ2与ⓓ	-1(2)	

*1: 如果ⓢ1与ⓓ使用相同的软元件，则基本步数将增加 1 步。

指令符号	指定通用运算寄存器的位置	增加的步数 （指令步数）	基本步数
D＊、D＊P、D/、D/P、E＊、E＊P	ⓢ1与ⓢ2与Ⓓ	-2(1)	3
	ⓢ1或者ⓢ2与Ⓓ	-1(2)	
	ⓢ1与ⓢ2 （只有在Ⓓ中指定了 不增加步数的软元件时）	± 0(3)	
	ⓢ1与ⓢ2 （只有在Ⓓ中指定了 连号访问方式文件寄存器时）	+2(5)	
MOV、MOVP、DMOV、DMOVP、EMOV、EMOVP	ⓢ1与Ⓓ	-1(1)	2
BCD、BCDP、BIN、BINP、DBCD、DBCDP、 DBIN、DBINP、FLT、FLTP、DFLT、DFLTP、 INT、INTP、DINT、DINTP、CML、CMLP、 DCML、DCMLP	ⓢ1与Ⓓ	-1(1)	2

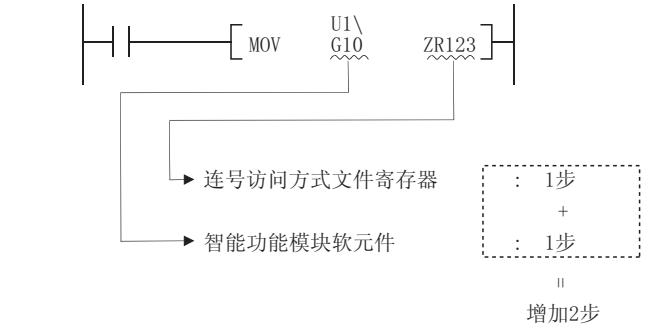
2）未进行子集处理的指令

下表是在未进行子集处理的指令中，增加了步数的软元件的情况。

增加了步数的软元件	增加的步数	示例
智能功能模块软元件	1	MOV U4\G10 D0
多 CPU 共享软元件		MOV U3E1\G10000 D0
链接直接软元件		MOV J3\B20 D0
变址寄存器 / 通用运算寄存器		MOV Z0 D0
连号访问方式文件寄存器		MOV ZR123 D0
32 位常数		DMOV K123 D0
实数常数		EMOV E0.1 D0
字符串常数	偶数时：字符数÷2 奇数时：（字符数+1）÷2	$MOV “123” D0

(d) 如果在上述 (a) ～ (c) 中的某些条件重叠，则步数将被累加。

例如指定了 MOV U1\G10 ZR123，则合计增加 2 步。



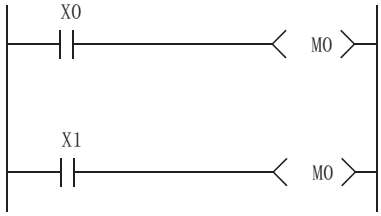
3.9 使用同一软元件的 OUT 指令、SET/RST 指令或 PLS/PLF 指令时的动作

以下介绍在 1 个扫描中多次执行同一软元件的 OUT 指令、SET/RST 指令或 PLS/PLF 指令时的动作。

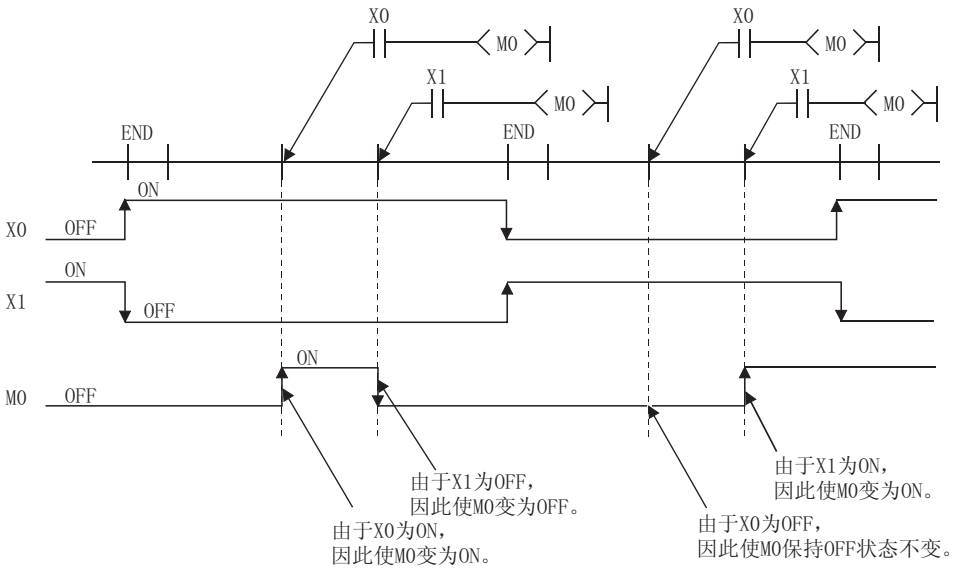
(1) 使用同一软元件的 OUT 指令时

在编程时，应避免在 1 个扫描中多次执行同一软元件的 OUT 指令。
如果在 1 个扫描多次执行了同一软元件的 OUT 指令，则在执行各个 OUT 指令时，指定的软元件都将根据 OUT 指令的运算结果而 ON 或 OFF。
由于各个 OUT 指令的执行都将决定指定软元件的 ON 或 OFF，这样在 1 个扫描内软元件可能会反复变为 ON 或 OFF。
通过输入 X0 和 X1 使同一内部继电器 (M0) 变为 ON/OFF 的梯形图的动作如下所示。

[梯形图]



[时序图]



对于刷新型 CPU 模块，当通过 OUT 指令指定输出 (Y) 后，1 个扫描的最后执行的 OUT 指令的 ON/OFF 状态将被输出。

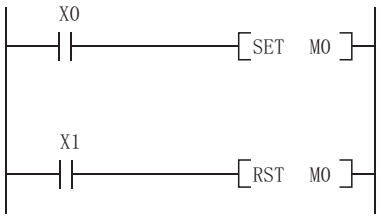
(2) 使用同一软元件的 SET/RST 指令时

(a) 当 SET 指令为 ON 时使指定软元件变为 ON，此后当执行指令为 OFF 时，不执行任何动作。
因此，如果在 1 个扫描中对同一个软元件的 SET 指令执行了多次时，如果某个 SET 指令为 ON，则指定软元件将变为 ON。

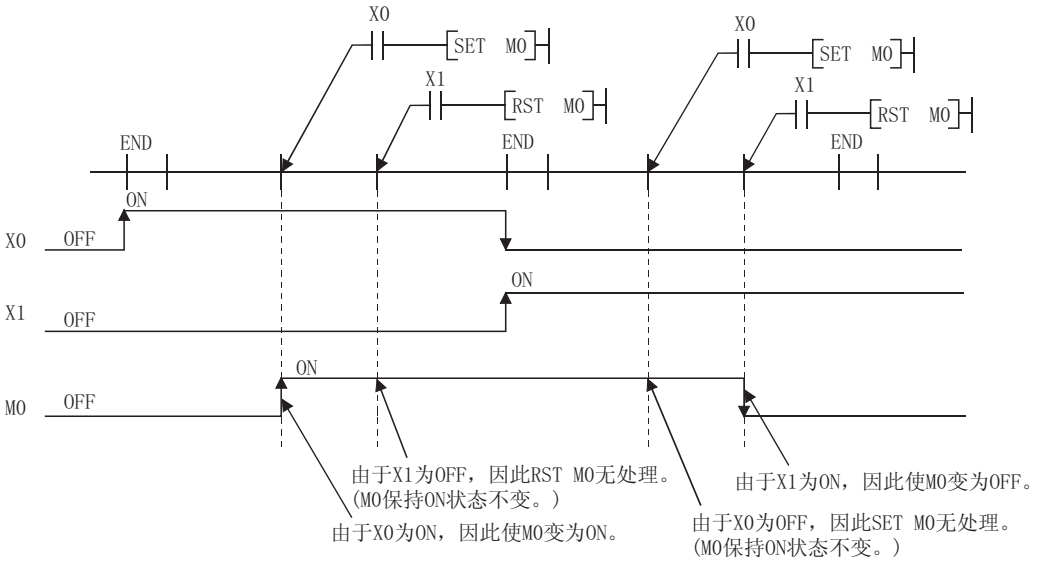
(b) 当 RST 指令为 ON 时使指定软元件变为 OFF，此后当 RST 指令为 OFF 时，不执行任何动作。
因此，如果在 1 个扫描中对同一个软元件的 RST 指令执行了多次时，如果某个 RST 指令为 ON，则指定软元件将变为 OFF。

(c) 如果在 1 个扫描中同时存在有同一个软元件的 SET 指令和 RST 指令，则当 SET 指令为 ON 时，使指定软元件变为 ON，当 RST 指令为 ON 时，使指定软元件变为 OFF。
当 SET 指令和 RST 指令均为 OFF 时，指定软元件的 ON/OFF 状态将不发生变化。

[梯形图]



[时序图]



对于刷新型 CPU 模块，当通过 SET/RST 指令指定输出 (Y) 后，1 个扫描的最后执行的 SET/RST 指令的 ON/OFF 状态将被输出。

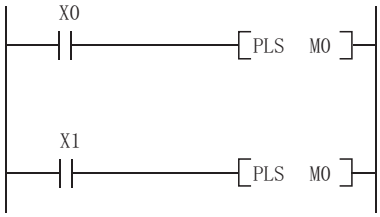
(3) 使用同一软元件的 PLS 指令时

当 PLS 指令由 OFF → ON 时，使指定软元件变为 ON。在除 OFF → ON 以外 (OFF → OFF、ON → ON、ON → OFF) 的情况下，均使指定软元件变为 OFF。

当在 1 个扫描中执行了多次同一软元件的 PLS 指令的情况下，各 PLS 指令由 OFF → ON 时，使指定软元件变为 ON。在除 OFF → ON 以外时，均使指定的软元件变为 OFF。

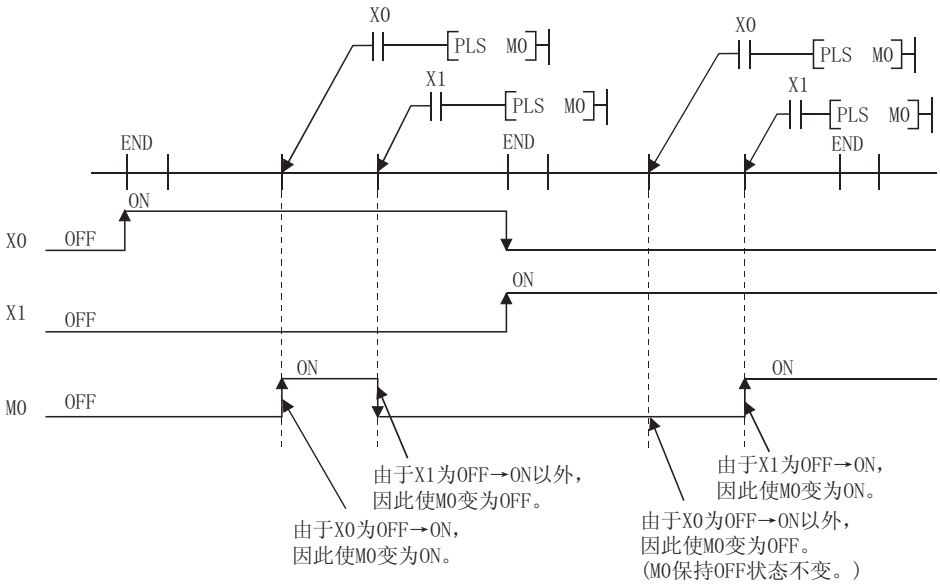
因此，在 1 个扫描中执行了多次同一软元件的 PLS 指令时，通过 PLS 指令变为 ON 的软元件可能在 1 个扫描中不变为 ON。

[梯形图]

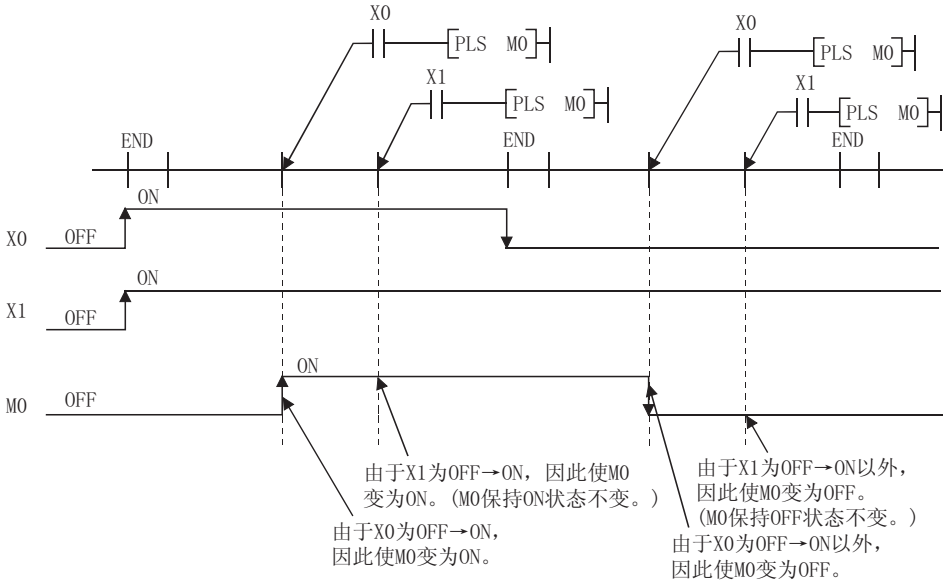


[时序图]

· X0 与 X1 的 ON/OFF 时机不相同时 (指定软元件在 1 个扫描内不变为 ON。)



· X0 与 X1 的 OFF → ON 时机相同时



对于刷新型 CPU 模块，当通过 PLS 指令指定输出 (Y) 后，1 个扫描的最后执行的 PLS 指令的 ON/OFF 状态将被输出。

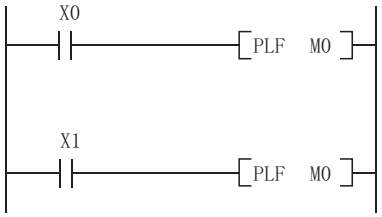
(4) 使用同一软元件的 PLF 指令时

当 PLF 指令由 ON → OFF 时，使指定软元件变为 ON，在除 ON → OFF 以外 (OFF → OFF、OFF → ON、ON → ON) 的情况下，均使指定软元件变为 OFF。

当在 1 个扫描中执行了多次同一软元件的 PLF 指令的情况下，各 PLF 指令由 ON → OFF 时，使指定软元件变为 ON。在除 ON → OFF 以外时，均使指定的软元件变为 OFF。

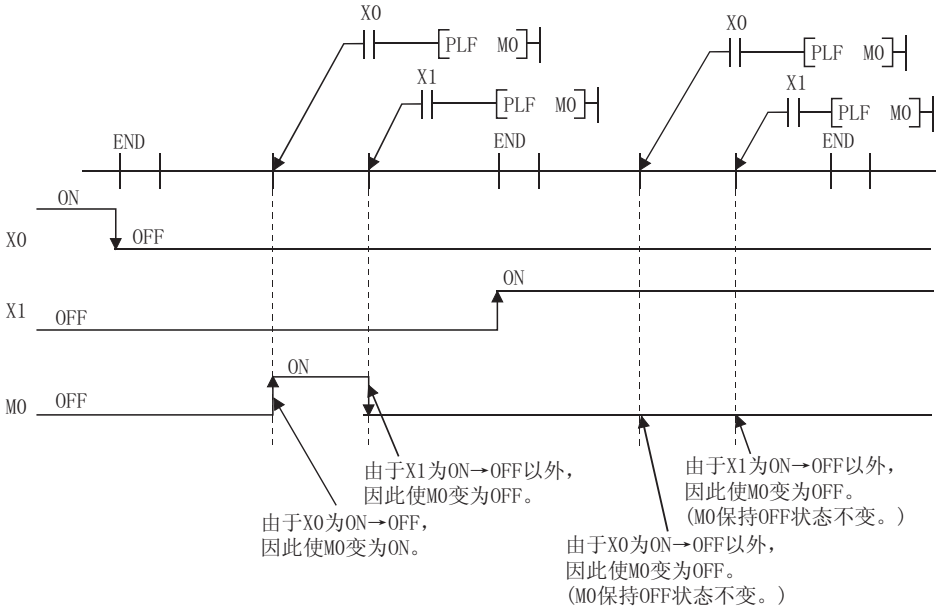
因此，在 1 个扫描中执行了多次同一软元件的 PLF 指令时，通过 PLF 指令变为 ON 的软元件可能在 1 个扫描中不变为 ON。

[梯形图]

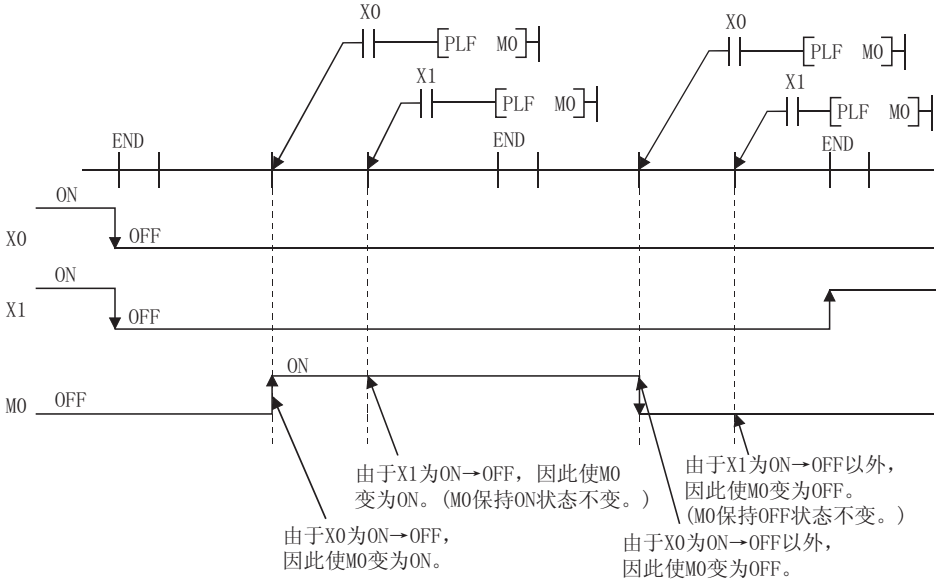


[时序图]

· X0 与 X1 的 ON/OFF 时机不相同时（指定软元件在 1 个扫描内不变为 ON。）



· X0 与 X1 的 OFF → ON 时机相同时



对于刷新型 CPU 模块，当通过 PLF 指令指定输出 (Y) 后，1 个扫描的最后执行的 PLS 指令的 ON/OFF 状态将被输出。

3.10 使用文件寄存器时的注意事项

本节介绍在 QCPU 中使用文件寄存器时的注意事项。

(1) 不能使用文件寄存器的 CPU 模块

Q00JCPU 不能使用文件寄存器。
当使用文件寄存器时，应使用 Q00JCPU 以外的 CPU 模块。

(2) 所用文件寄存器的设置

如果要使用文件寄存器，需要通过可编程控制器参数或 QDRSET 指令对所用文件寄存器进行设置。
(由于 Q00CPU 和 Q01CPU 的可编程控制器参数已被设置为 “ 使用文件寄存器 ”，因此无需再设置。)
如果未对所用的文件寄存器进行设置，则在使用了文件寄存器的指令中不能进行正常运算。

要 点

即使未在可编程控制器参数中对所用的文件寄存器进行设置，也可以创建使用了文件寄存器的程序。此外，对于除通用型 QCPU 以外的 CPU 模块，即使将该程序写入到 CPU 模块中并执行，也不会变为出错状态。
但是，不能对文件寄存器进行正常的数据读取 / 写入，应加以注意。
对于通用型 QCPU，如果执行使用了文件寄存器的程序，将会变为出错状态。

(3) 文件寄存器区域的预留

(a) 高性能型 QCPU、过程 CPU、冗余 CPU、通用型 QCPU
当使用文件寄存器时，将文件寄存器登录到标准 RAM/ 存储卡中，应预留出文件寄存器的区域。

(b) 基本型 QCPU(Q00JCPU 除外)
文件寄存器区域已预先在标准 RAM 中被预留。
用户无需再对文件寄存器区域进行预留。

各 CPU 模块中可使用文件寄存器的存储器如下表所示。

存储器	高性能型 QCPU、过程 CPU、冗余 CPU、通用型 QCPU	基本型 QCPU (Q00JCPU 除外)
标准 RAM	○	○
存储卡 *1 *2	○	×

○ : 可登录 ; × : 不可登录

*1: 使用闪存时，只能从文件寄存器中进行读取。(不能对闪存进行写入。)

*2: 使用 E^2PROM 时，可以通过 PROMWR 指令对 E^2PROM 进行写入。

备注

关于文件寄存器的设置方法及文件寄存器区域的预留方法，请参阅考所使用的 CPU 模块的用户手册 (功能解说 / 程序基础篇)。

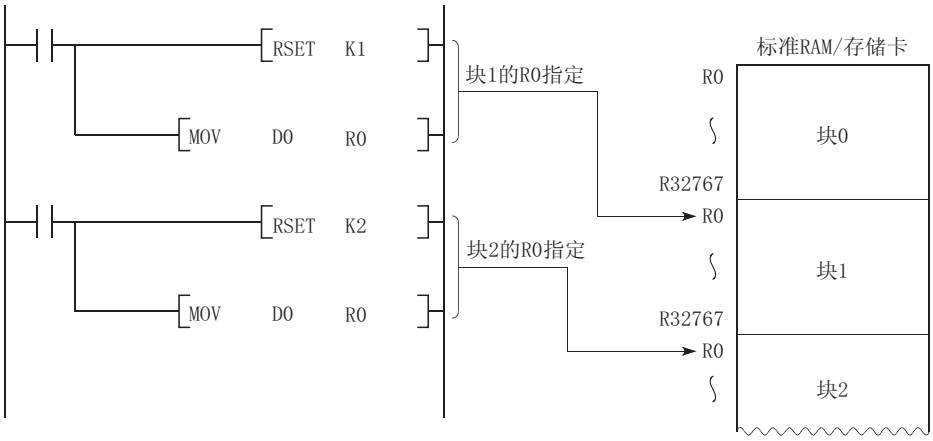
(4) 文件寄存器编号的指定超过了所登录的点数时

(a) 对于除通用型 QCPU 以外的 CPU
即使对超出所登录点数的编号的文件寄存器进行了读取 / 写入，也不会发出出错信息。
但是，不能对文件寄存器进行正常的数据读取 / 写入，应加以注意。

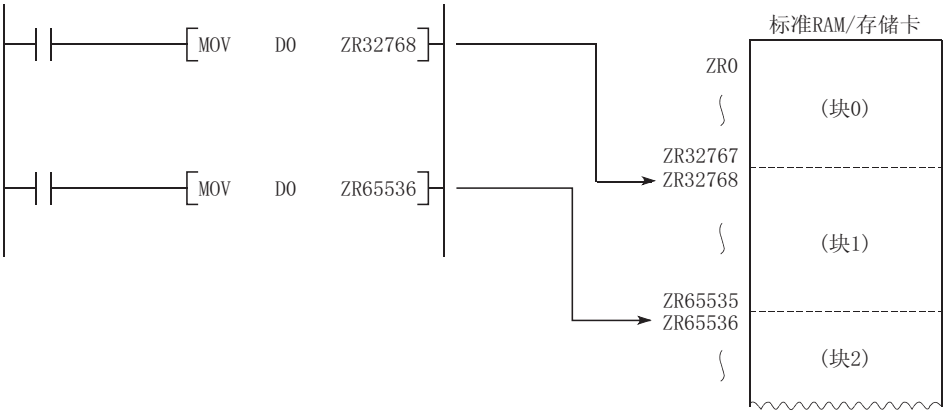
(b) 对于通用型 QCPU
对超出所登录点数的编号的文件寄存器进行了读取 / 写入时，将会变为出错状态。
(出错代码： 4101)

(5) 文件寄存器的指定方法
文件寄存器的指定方法有块切换方式及连号访问方式这两种。

(a) 块切换方式
在块切换方式中，以 32k 点 (1 个块) 为单位对所使用的文件寄存器点数进行分区。
对于 32k 点以上的文件寄存器，通过 RSET 指令对所使用的文件寄存器的块号进行切换后进行指定。
各块的指定范围均为 R0 ～ R32767。



(b) 连号访问方式
使用连号访问方式时，通过连续的软元件号指定超过 32k 点的文件寄存器。
多块的文件寄存器可以作为连续的文件寄存器使用。



(6) 将文件寄存器指定为刷新软元件时的设置及限制

(a) 刷新软元件的设置

刷新软元件的设置可在以下设置中进行：

· CC-Link IE 控制网络的刷新设置

· MELSECNET/H 的刷新设置

· CC-Link 的刷新设置

· 智能功能模块的自动刷新设置

· 多 CPU 系统的自动刷新设置

(b) 限制事项

将文件寄存器指定为刷新软元件时，应注意以下 1) ～ 3) 的限制事项：

1) 在可编程控制器参数中，如果进行了使用与程序同名的文件寄存器的指定，将不能正确地进行刷新。

如果使用与程序同名的文件寄存器，将被刷新到与程序设置中设置为最后编号的程序同名的文件寄存器中。希望读取或写入刷新数据时，应使用 QDRSET 指令切换为相应文件寄存器后进行指定。

2) 如果通过 QDRSET 指令更改了文件寄存器的文件名或驱动号，将不能正确地执行刷新。

如果通过 QDRSET 指令更改了文件寄存器的文件名或驱动号，将被链接刷新到 END 指令执行时的设置文件中。希望读取或写入刷新数据时，应指定为在 END 指令执行时的设置文件。

但是，在除通用型 QCPU 以外的 CPU 模块中，软元件被指定为 “ ZR ” 时，如果通过 QDRSET 指令更改了驱动号，将会发生错误 (LINK PARA ERROR (3101))，应加以注意。(将软元件指定为 “ R ” 时不会变为出错状态。)

3) 当通过 RSET 指令切换了块号时，将被刷新到切换的块号的文件寄存器 (R) 中。

通过 RSET 指令切换了块号时，将被刷新到 END 指令执行时的块号的文件寄存器 (R) 中。希望读取或写入刷新数据时，应指定为 END 指令执行的块号。

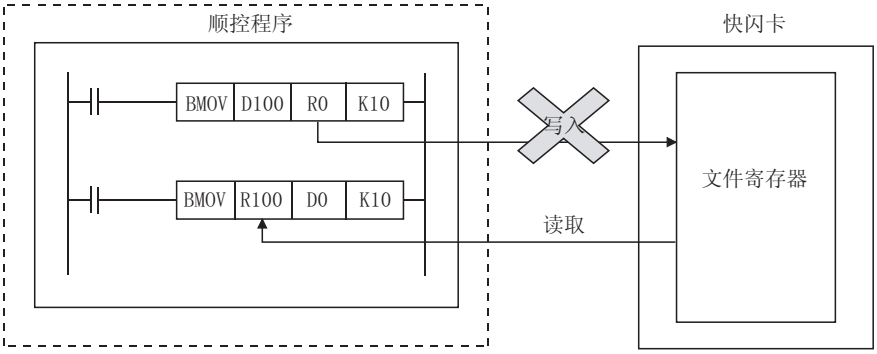
(7) 使用快闪卡中的文件寄存器时的注意事项

以下介绍可使用文件寄存器的快闪卡的注意事项。

(a) 下列快闪卡可以和 QCPU 一起使用。

· QCPU: Flash 卡

(b) 只有通过顺控程序才可以读取快闪卡中的文件寄存器。
(不能通过顺控程序对快闪卡进行写入。)



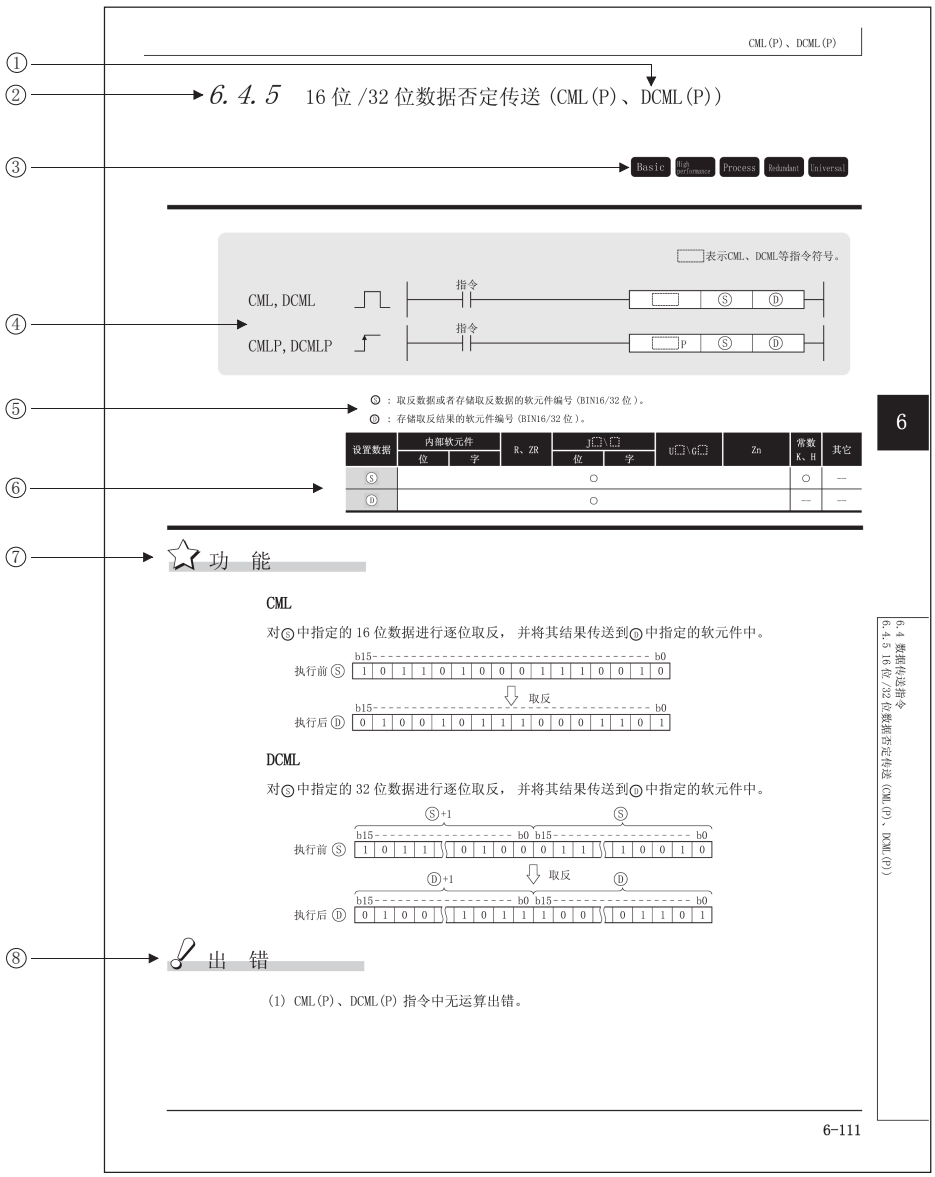
将快闪卡作为文件寄存器使用时，应预先写入数据。

使用 GX Developer，将数据写入快闪卡。

备忘录

4 如何阅读指令

本章以后各章节中指令的说明将按以下形式进行：

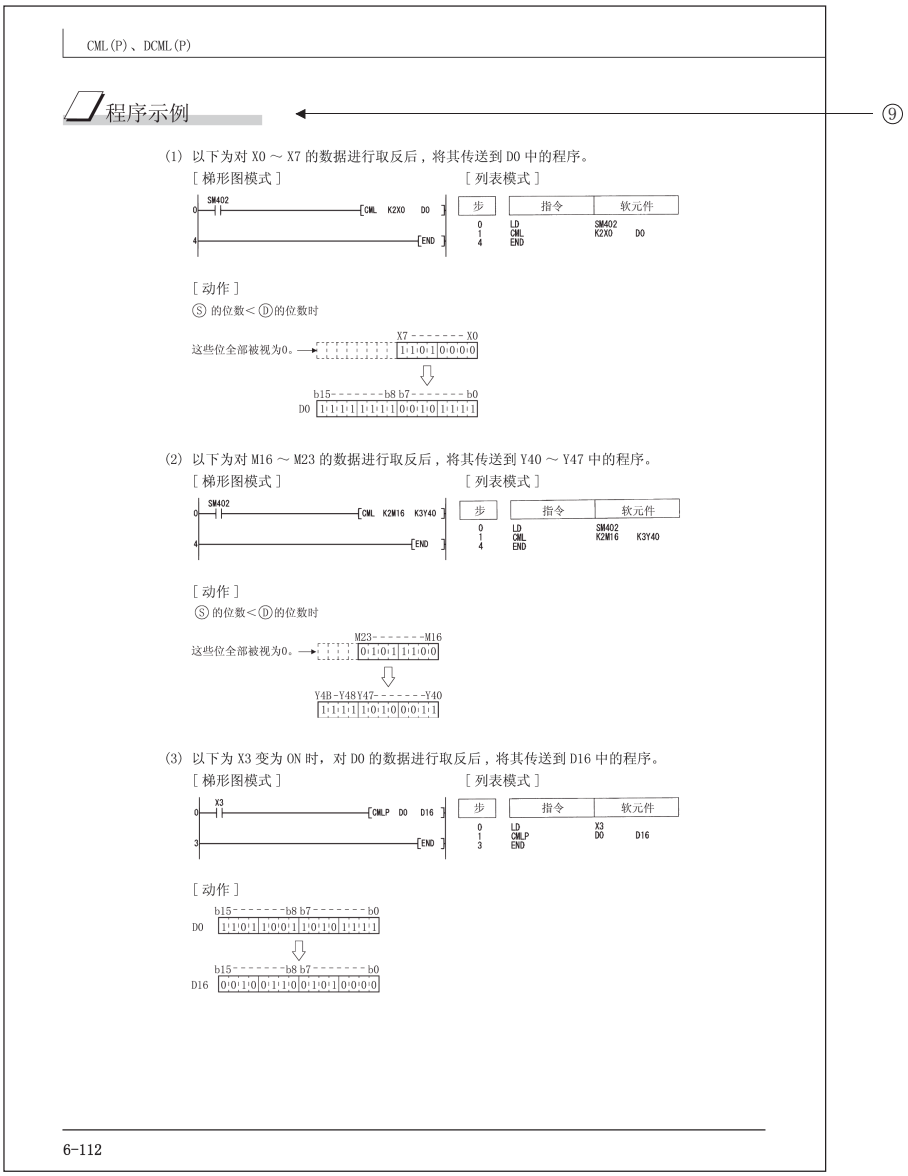


1) 表示指令符号。

2) 表示章节号及指令概要。

3) 表示指令能否在各 CPU 模块类型中使用。

图标					内容
通用型 QCPU	基本型 QCPU	高性能型 QCPU	过程 CPU	冗余 CPU	
Universal	Basic	High performance	Process	Redundant	普通图标，表示可以使用相应指令。
Ver. Universal	Ver. Basic	Ver. High performance	Ver. Process	Ver. Redundant	带版本符号的图标，表示可以使用但有某些限制。（例如：功能版本、软件版本）
Universal	Basic	High performance	Process	Redundant	带×符号的图标，表示不可以使用的相应指令。



4) 表示梯形图模式的表示及指令的执行条件。

执行条件	常时执行	在 ON 时执行	在 ON 时执行 1 次	在 OFF 时执行	在 OFF 时执行 1 次
说明页面的记载符号	未记入				

5) 表示各指令设置数据的说明及数据类型。

数据类型	内容
位	表示位数据或位数据的起始编号。
BIN16 位	表示处理的 BIN16 位数据或字软元件的起始编号。
BIN32 位	表示处理的 BIN32 位数据或双字软元件的起始编号。
BCD4 位	表示处理的 4 位 BCD 数据。
BCD8 位	表示处理的 8 位 BCD 数据。
实数	表示处理的浮点数据。
字符串	表示处理的字符串数据。
软元件名	表示处理的软元件名。

6) 指令中可以使用的软元件用○表示。
可以使用的软元件型号表示如下：

设置数据	内部软元件（系统、用户）		文件寄存器 R、ZR	直接链接软元件 *4 J□\□		智能功能模块 U□\G□	变址寄存器 Zn	常数 *5	其它 *5
	位	字		位	字				
可用软元件 *1	X、Y、M、L、SM、F、B、SB、FX、FY*2	T、ST、C、*3 D、W、SD、SW、FD、@□	R、ZR	J□\X J□\Y J□\B J□\SB	J□\W J□\SW	U□\G□	Z	K、H、E、$	P、I、J、U、DX、DY、N、BL、TR、BL\S、V

*1: 关于各软元件的说明，请参阅所使用的 CPU 模块的用户手册（功能解说 / 程序基础篇）。
*2: FX 和 FY 只能用于位数据，FD 只能用于字数据。
*3: 当 T、ST 和 C 用于除以下指令以外时，只能用于字数据。（不能用于位数据。）
[位数据中可使用的指令]
LD、LDI、AND、ANI、OR、ORI、LDP、LDF、ANDP、ANDF、ORP、ORF、OUT、RST、BKRST
*4: 在 CC-Link IE 控制网络、MELSECNET/H 和 MELSECNET/10 中可以使用。
*5: 可以进行设置的软元件记录在“常数”和“其它”栏中。

7) 表示指令的功能。

8) 表示引起出错的条件及出错号代码。
关于未记载的错误，请参阅 3.6 节。

9) 以梯形图和列表这两种模式表示简单的程序示例。
此外，表示该程序被执行时的各软元件的内容。

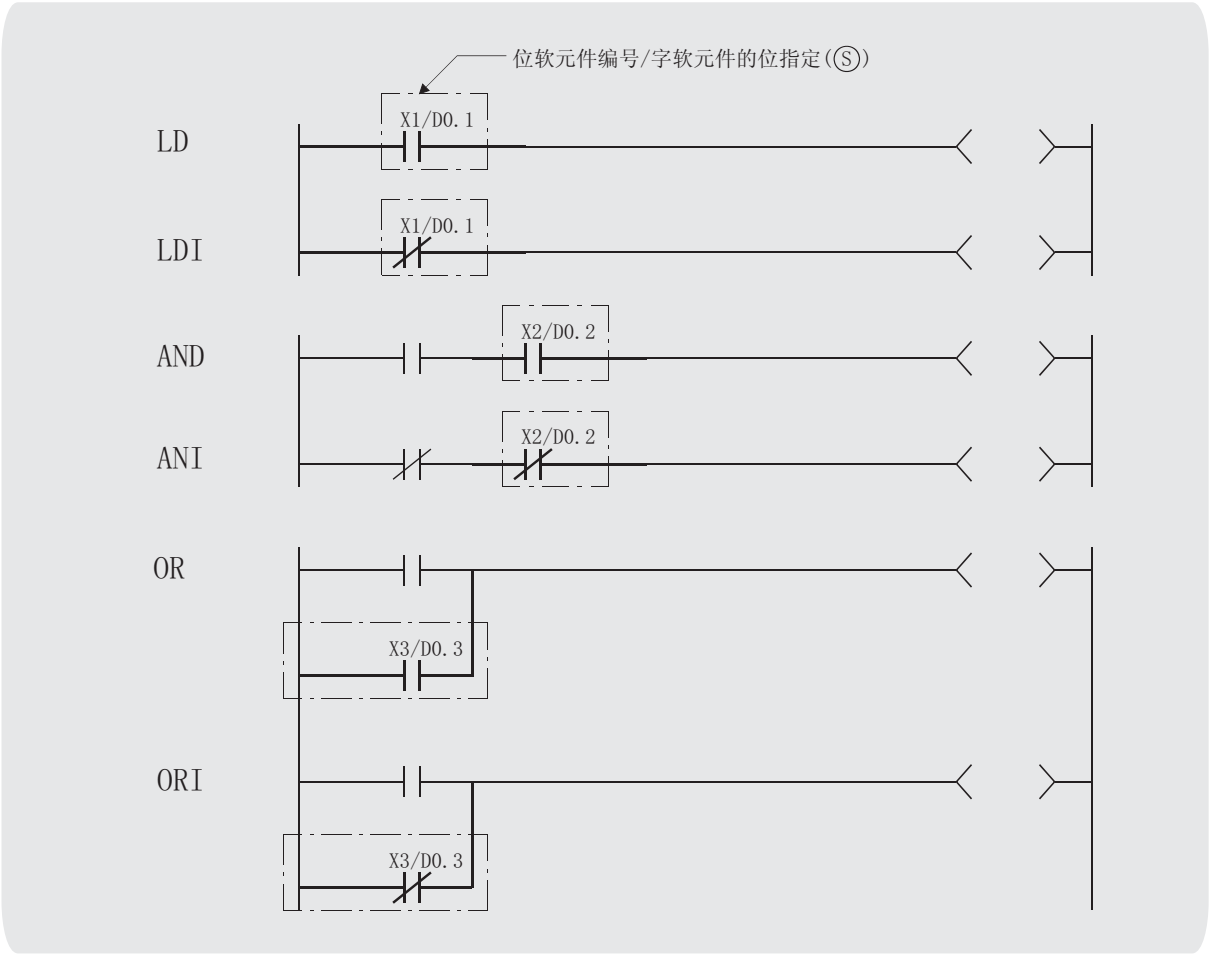
5 顺控程序指令

分类	处理内容	参阅章节
触点指令	运算开始、串行连接、并行连接、否定运算、否定串行连接、否定并行连接	5.1 节
连接指令	梯形图块连接、运算结果的脉冲化、运算结果的记忆 / 读取	5.2 节
输出指令	位软元件的输出、脉冲输出、输出取反	5.3 节
移位指令	位软元件的移位	5.4 节
主控指令	主控	5.5 节
结束指令	结束程序	5.6 节
其它指令	程序停止、无处理等未记入上述分类中指令	5.7 节

5.1 触点指令

5.1.1 运行开始、串行连接、并行连接 (LD、LDI、AND、ANI、OR、ORI)

Basic | High performance | Process | Redundant | Universal



Ⓢ：作为触点使用的软元件（位）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				DX、BL
Ⓢ	○						--		○

功 能

LD、LDI

(1) LD 是 a 触点运算开始指令，LDI 是 b 触点运算开始指令。其功能是从指定的软元件中读取 ON/OFF 信息 *1，并将其作为运算结果。

*1: 进行了字软元件的位指定时，根据指定位的 1/0 而 ON/OFF。

AND、ANI

(1) AND 是 a 触点串行连接指令，ANI 是 b 触点串行连接指令，其功能是读取指定位软元件的 ON/OFF 信息 *2，将其与至目前为止的运算结果执行 AND 运算，并将该值作为运算结果。

*2: 进行了字软元件的位指定时，根据指定位的 1/0 而 ON/OFF。

(2) 对 AND 或 ANI 的使用没有限制，但是在外围设备的梯形图模式中的情况如下所示：

(a) 写入 ..... 当 AND 和 ANI 为串行连接时，最多可以创建 24 级的梯形图。

(a) 读取..... 当 AND 和 ANI 为串行连接时，最多可以显示 24 级的梯形图。
如果超过 24 级，则最多显示 24 级。

OR、ORI

(1) OR 是 1 个 a 触点的并行连接指令，ORI 是 1 个 b 触点的并行连接指令。其功能是从指定的软元件读取 ON/OFF 信息 *3，将其与至目前为止的运算结果执行 OR 运算，并将该值作为运算结果。

*3: 字软元件的位指定时，根据指定位的 1/0 而 ON/OFF。

(2) 对 OR 或 ORI 的使用没有限制，但是在外围设备的梯形图模式中的情况如下所示：

(a) 写入 ..... 最多可以创建通过 23 个 OR 和 ORI 连接的梯形图。

(a) 读取 ..... 最多可以显示通过 23 个 OR 和 ORI 连接的梯形图。
第 24 个及以后的梯形图不能被正确显示。

备 注

字软元件位的指定以 16 进制形式进行设置。
D0 的位 b11 变为 D0.0B。
关于字软元件的位指定，请参阅 3.2.1 项。

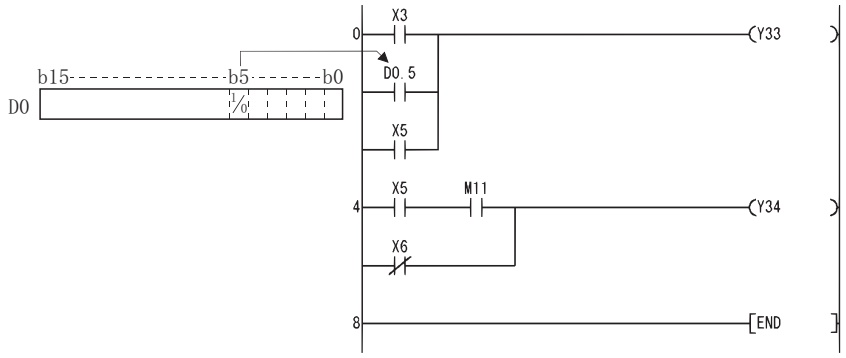
出　错

(1) 在 LD、LDI、AND、ANI、OR、ORI 指令中无运算错误。

程序示例

(1) 使用了 LD、AND、OR、ORI 指令的程序。

[梯形图模式]

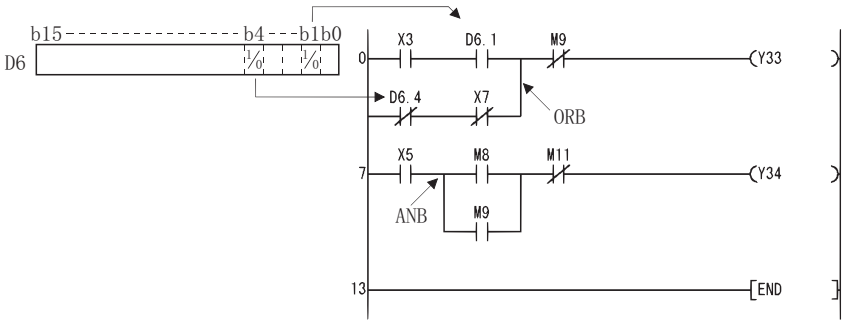


[列表模式]

步	指令	软元件	
0	LD	X3	
1	OR	D0.5	···字软元件
2	OR	X5	的位指定
3	OUT	Y33	
4	LD	X5	
5	AND	M11	
6	ORI	X6	
7	OUT	Y34	
8	END		

(2) 通过使用 ANB 和 ORB 指令进行触点连接的程序。

[梯形图模式]

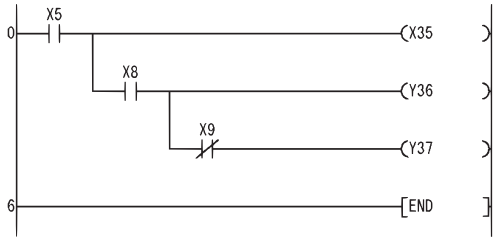


[列表模式]

步	指令	软元件	
0	LD	X3	
1	AND	D6.1	···字软元件
2	LDI	D6.4	
3	ANI	X7	的位设置
4	ORB		
5	ANI	M9	
6	OUT	Y33	
7	LD	X5	
8	LD	M8	
9	OR	M9	
10	ANB		
11	ANI	M11	
12	OUT	Y34	
13	END		

(3) OUT 指令的并行程序。

[梯形图模式]

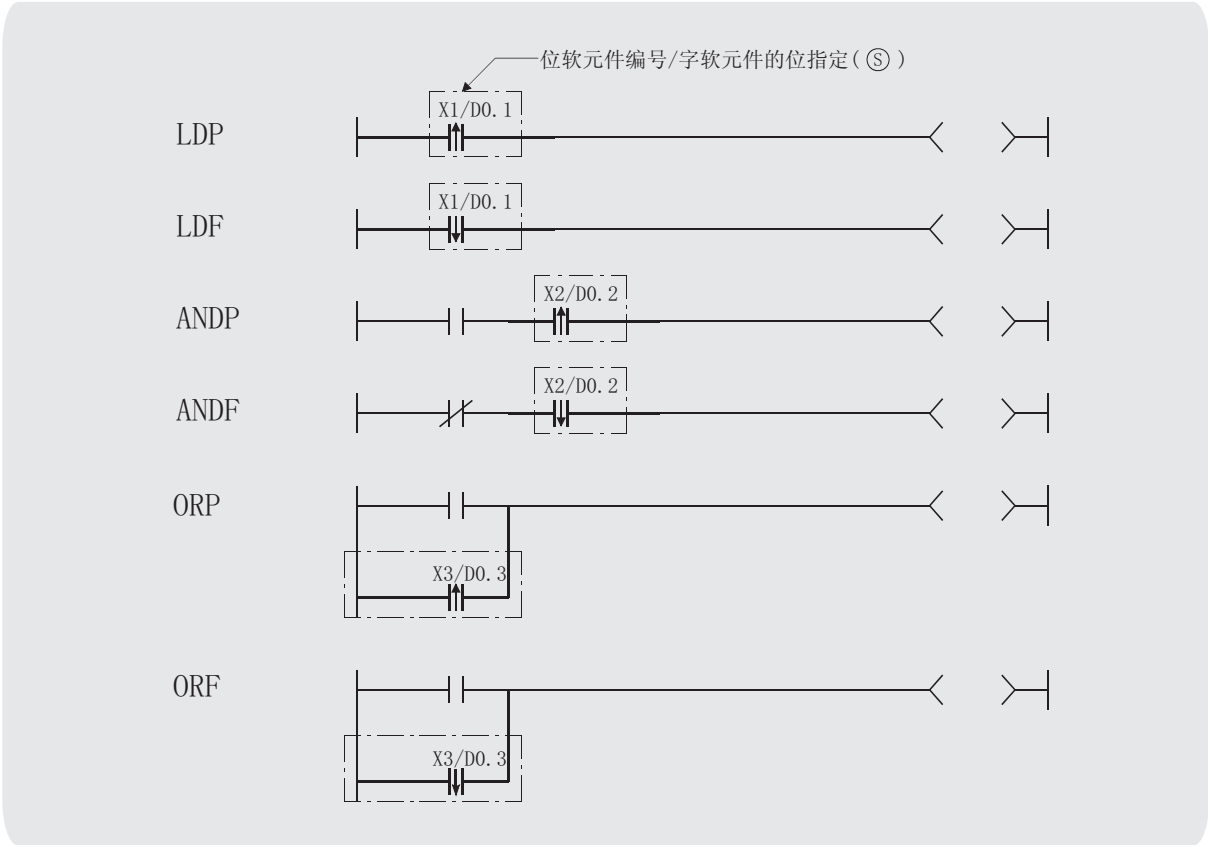


[列表模式]

步	指令	软元件
0	LD	X5
1	OUT	X35
2	AND	X8
3	OUT	Y36
4	ANI	X9
5	OUT	Y37
6	END	

5.1.2 脉冲运算开始、脉冲串行连接、脉冲并行连接 (LDP、LDF、ANDP、ANDF、ORP、ORF)

Basic | High performance | Process | Redundant | Universal



Ⓢ：作为触点使用的软元件(位)。

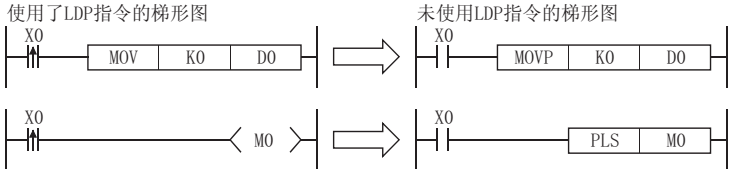
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它 DX
	位	字		位	字				
Ⓢ	○						--		○

功 能

LDP、LDF

(1) LDP 是上升沿脉冲运算开始指令，仅在指定位软元件的上升沿 (OFF → ON) 时导通。字软元件的位指定时，仅在指定位从 0 变为 1 时导通。

如果只有 1 个 LDP 指令，则与 ON 中执行指令的脉冲化指令 (□P) 相同。



(2) LDF 是下降沿脉冲运算开始指令，并且仅在指定位软元件的下降沿 (ON → OFF) 时导通。
字软元件的位指定时，在指定位变为 1 → 0 时导通。

ANDP、ANDF

(1) ANDP 是上升沿脉冲串行连接指令，ANDF 是下降沿脉冲串行连接指令。其功能为对至目前为止的运算结果执行 AND 运算，并将结果值作为运算结果。
ANDP、ANDF 中使用的 ON/OFF 信息如下表所示：

ANDP、ANDF 中指定的软元件		ANDP 的状态	ANDF 的状态
位软元件	字软元件的位指定		
OFF → ON	0 → 1	ON	OFF
OFF	0	OFF	
ON	1		
ON → OFF	1 → 0		ON

ORP、ORF

(2) ORP 是上升沿脉冲并行连接指令，ORF 是下降沿脉冲串行连接指令，其功能为对至目前为止的运算结果进行 OR 运算后将值作为运算结果。
ORP、ORF 中使用的 ON/OFF 信息如下表所示：

ORP、ORF 中指定的软元件		ORP 的状态	ORF 的状态
位软元件	字软元件位指定		
OFF → ON	0 → 1	ON	OFF
OFF	0	OFF	
ON	1		
ON → OFF	1 → 0		ON

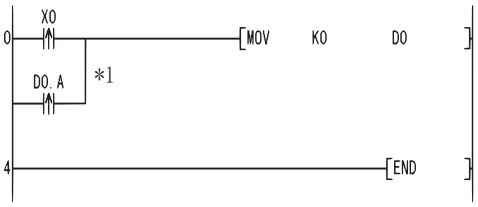
出　错

(1) 在 LDP、LDF、ANDP、ANDF、ORP、ORF 指令中无运算出错。

程序示例

(1) 下述程序在输入 X0 或者在数据寄存器 D0 的 b10(位 11) 的上升沿，执行 MOV 指令：

[梯形图模式]



[列表模式]

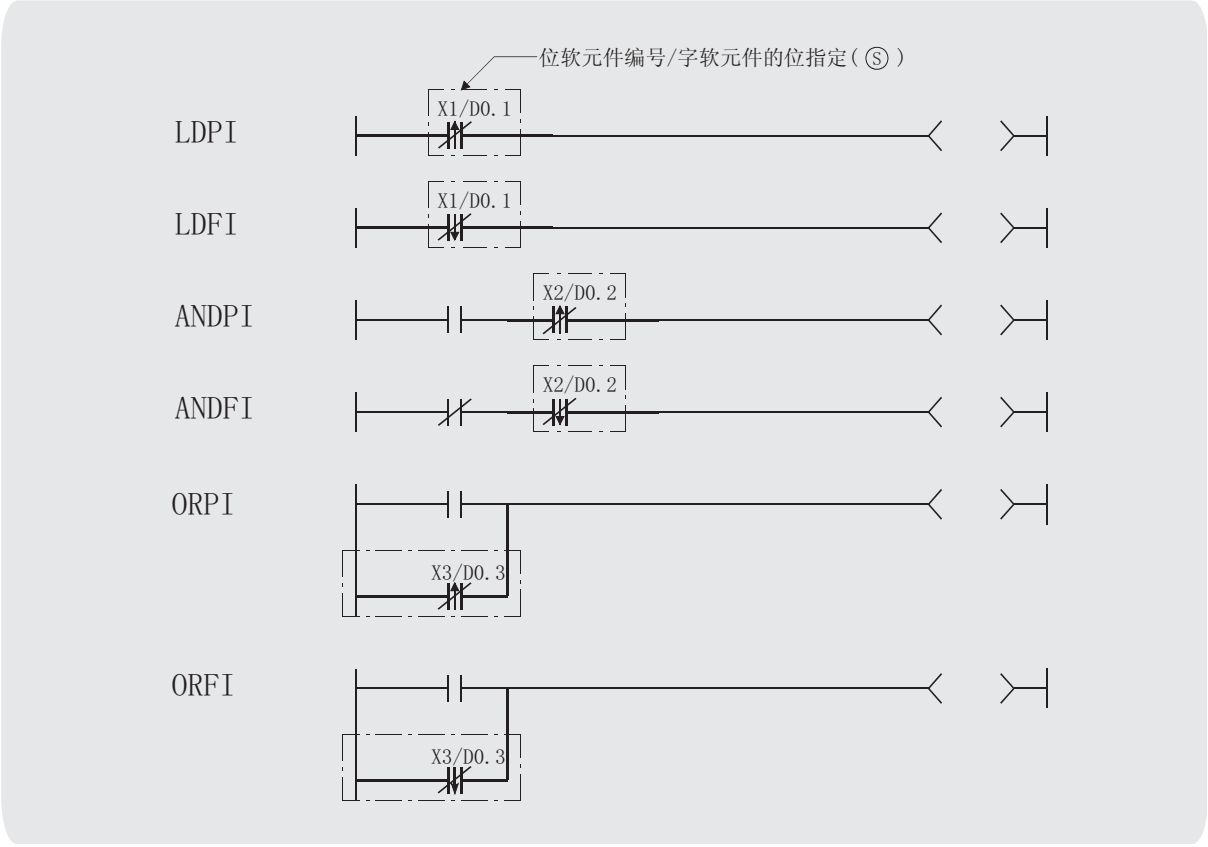
步	指令	软元件	
0	LDP	X0	
1	ORP	D0.A	
2	MOV	K0	D0
4	END		

*1:　字软元件的位指定是以 16 进制数的形式进行的。
D0 的 b10 变为 D0.0A。

5.1.3 脉冲否运算开始、脉冲否串行连接、脉冲否并行连接 (LDPI、LDFI、ANDPI、ANDFI、ORPI、ORFI)

Basic × | High performance × | Process × | Redundant × | Ver. Universal

· 序列号的前 5 位数为 “ 10102 ” 以后的 QnU(D)(H)CPU
· 序列号的前 5 位数为 “ 10102 ” 以后的 QnUDE(H)CPU



设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				DX
Ⓢ	○	--	○	○	--	○	--		○

功 能

LDPI、LDFI

(1) LDPI 是上升沿脉冲否运算开始指令，在指定位软元件为 OFF 时、ON 时、下降沿 (ON → OFF) 时导通。字软元件的位指定时，在指定位为 0 时、1 时、变为 1 → 0 时导通。

(2) LDFI 是下降沿脉冲否运算开始指令，在指定位软元件的上升沿 (OFF → ON) 时、OFF 时、ON 时导通。字软元件的位指定时，在指定位为 0 时、1 时、变为 0 → 1 时导通。

LDPI、LDFI 中使用的 ON/OFF 信息如下表所示：

LDPI、LDFI 中指定的软元件		LDPI 的状态	LDFI 的状态
位软元件	字软元件位指定		
OFF → ON	0 → 1	OFF	ON
OFF	0	ON	ON
ON	1	ON	ON
ON → OFF	1 → 0	ON	OFF

ANDPI、ANDFI

(1) ANDPI 是上升沿脉冲否串行连接指令，ANDFI 是下降沿脉冲否串行连接指令，其功能是对至目前为止的运算结果进行 AND 运算，将值作为运算结果。

ANDPI、ANDFI 中使用的 ON/OFF 信息如下表所示：

ANDPI、ANDFI 中指定的软元件		ANDPI 的状态	ANDFI 的状态
位软元件	字软元件位指定		
OFF → ON	0 → 1	OFF	ON
OFF	0	ON	ON
ON	1	ON	ON
ON → OFF	1 → 0	ON	OFF

ORPI、ORFI

(1) ORPI 是上升沿脉冲否并行连接指令，ORFI 是下降沿脉冲否并行连接指令，其功能是对至目前为止的运算结果进行 OR 运算，将值作为运算结果。

ORPI、ORFI 中使用的 ON/OFF 信息如下表所示：

ORPI、ORFI 中指定的软元件		ORPI 的状态	ORFI 的状态
位软元件	字软元件位指定		
OFF → ON	0 → 1	OFF	ON
OFF	0	ON	ON
ON	1	ON	ON
ON → OFF	1 → 0	ON	OFF

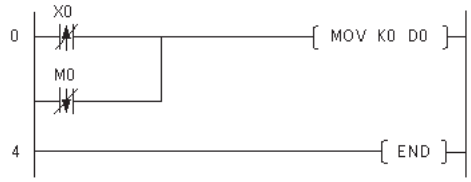
出　错

(1) 在 LDPI、LDFI、ANDPI、ANDFI、ORPI、ORFI 指令中无运算出错。

程序示例

(1) 下述程序在 X0 为 ON/OFF/ON → OFF 时，或者 M0 为 ON/OFF/OFF → ON 时，将 0 存储到 D0 中。

[梯形图模式]

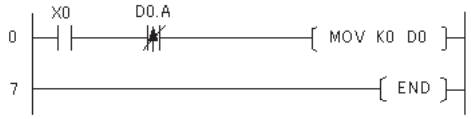


[列表模式]

步	指令	软元件
0	LDPI	X0
3	ORFI	M0
7	MOV	K0 D0
9	END	

(2) 下述程序在 X0 为 ON，且 D0 的 b10(位 11) 为 ON/OFF/ON → OFF 时，将 0 存储到 D0 中。

[梯形图模式]



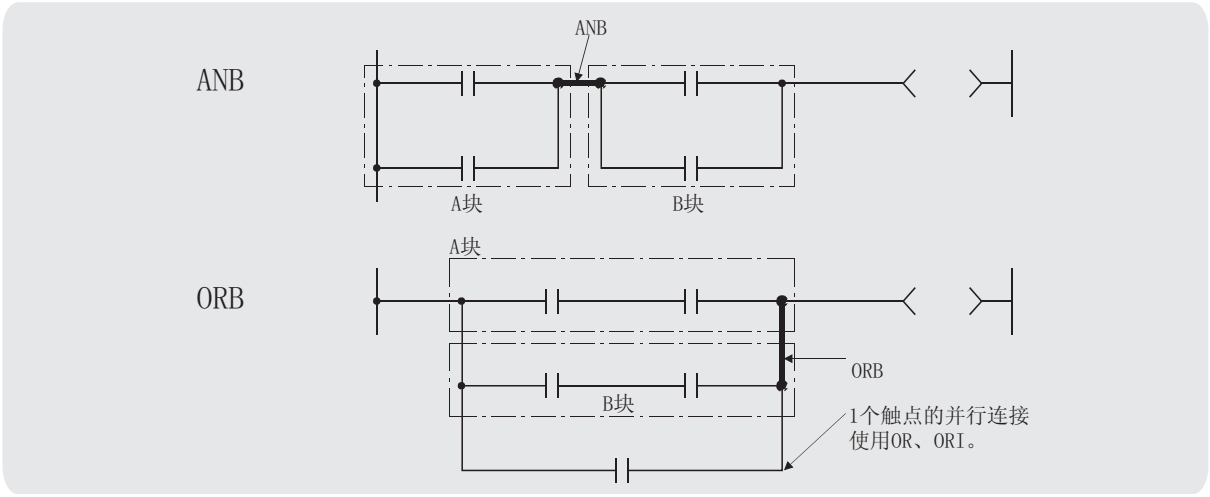
[列表模式]

步	指令	软元件
0	LD	X0
1	ANDPI	D0.A
5	MOV	K0 D0
7	END	

5.2 连接指令

5.2.1 梯形图块串行连接、并行连接 (ANB、ORB)

Basic | High performance | Process | Redundant | Universal



设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
--	--								

功 能

ANB

(1) 进行 A 块与 B 块的 AND 运算，并且将结果值作为运算结果。

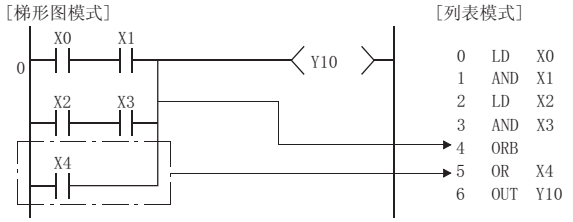
(2) ANB 的符号不是触点符号，而是连接符号。

(3) 当在列表模式下编程时，最多可以连续写入 15 条 ANB 指令 (16 块)。

ORB

(1) 进行 A 块与 B 块的 OR 运算，并且将结果值作为运算结果。

(2) ORB 是用于对有 2 个以上触点的梯形图块执行并行连接的。
对于只有 1 个触点的梯形图块，使用 OR 或 ORI，无需使用 ORB。



(3) ORB 符号不是触点符号，而是连接符号。

(4) 在列表模式中编程时，最多可以连续使用 15 个 ORB 指令 (16 块)。

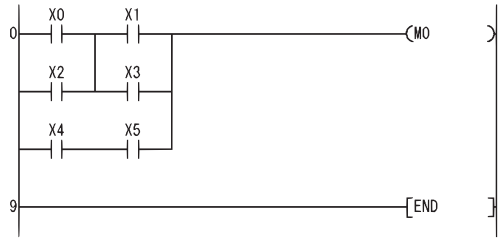
出　错

(1) 在 ANB、ORB 指令中无运算出错。

程序示例

(1) 使用了 ANB、ORB 的程序

[梯形图模式]

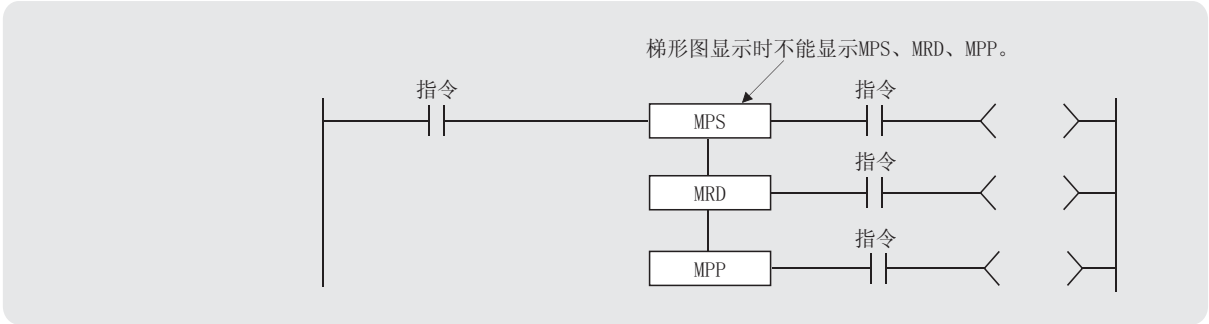


[列表模式]

步	指令	软元件
0	LD	X0
1	OR	X2
2	LD	X1
3	OR	X3
4	ANB	
5	LD	X4
6	AND	X5
7	ORB	
8	OUT	M0
9	END	

5.2.2 运算结果入栈、读取、退栈 (MPS、MRD、MPP)

Basic | High performance | Process | Redundant | Universal



设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
--	--								

功 能

MPS

(1) 记忆 MPS 指令之前的运算结果 (ON/OFF)。

(2) 最多可以连续使用 16 次 MPS 指令。
如果在中途使用了 MPP 指令，那么 MPS 指令的使用数将会减少 1 次。

MRD

(1) 读取通过 MPS 指令记忆的运算结果，并且使用该结果执行下一步运算。

MPP

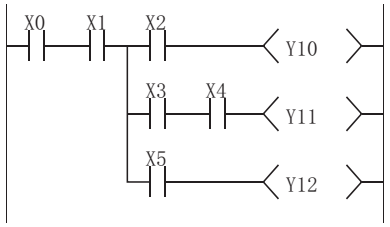
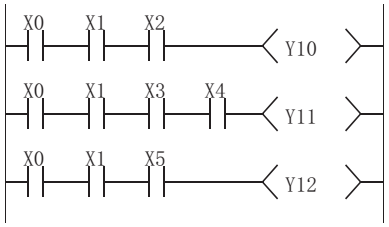
(1) 读取通过 MPS 指令记忆的运算结果，并且使用该结果执行下一步运算。

(2) 清除通过 MPS 指令记忆的运算结果。

(3) 从 MPS 指令使用次数中减去 1。

要 点

1. 使用及未使用 MPS、MRD、MPP 时的梯形图如下所示：

使用了 MPS、MRD、MPP 时的梯形图	未使用 MPS、MRD、MPP 时的梯形图
	

2. MPS 和 MPP 指令必须使用同样的次数。
   如果 MPS 和 MPP 指令的使用数不相同，在外围设备的梯形图模式中将不能正确显示梯形图。

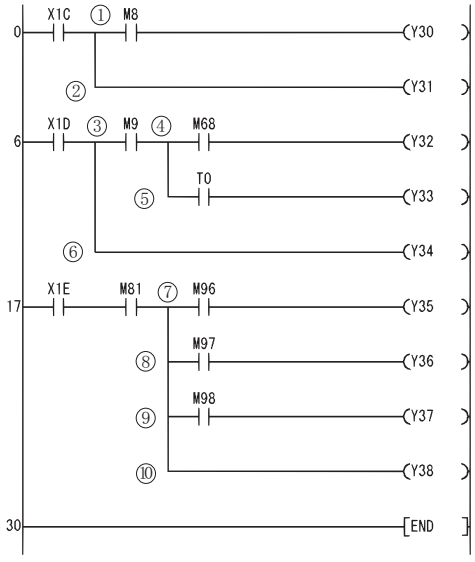
出 错

(1) 在 MPS、MRD、MPP 指令中无运算出错。

程序示例

(1) 使用了 MPS、MRD、MPP 的程序。

[梯形图模式]

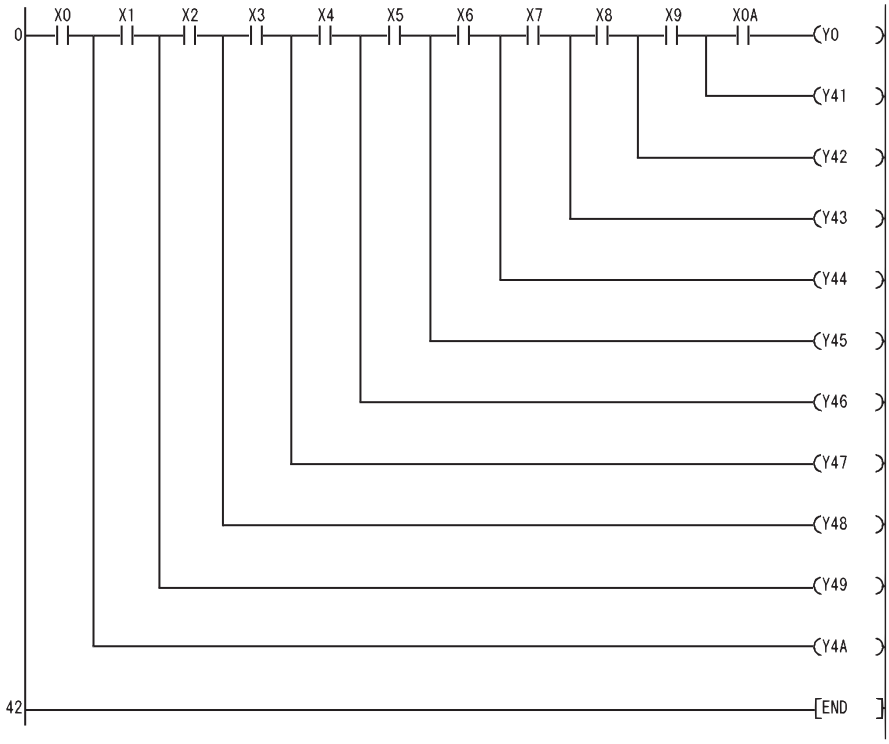


[列表模式]

步	指令	软元件
① 0	LD	X1C
1	MPS	
2	AND	M8
3	OUT	Y30
② 4	MPP	
5	OUT	Y31
6	LD	X1D
③ 7	MPS	
8	AND	M9
④ 9	MPS	
10	AND	M68
11	OUT	Y32
⑤ 12	MPP	
13	AND	T0
14	OUT	Y33
⑥ 15	MPP	
16	OUT	Y34
17	LD	X1E
18	AND	M81
⑦ 19	MPS	
20	AND	M96
21	OUT	Y35
⑧ 22	MRD	
23	AND	M97
24	OUT	Y36
⑨ 25	MRD	
26	AND	M98
27	OUT	Y37
⑩ 28	MPP	
29	OUT	Y38
30	END	

(2) 连续使用 MPS、MPP 的程序。

[梯形图模式]

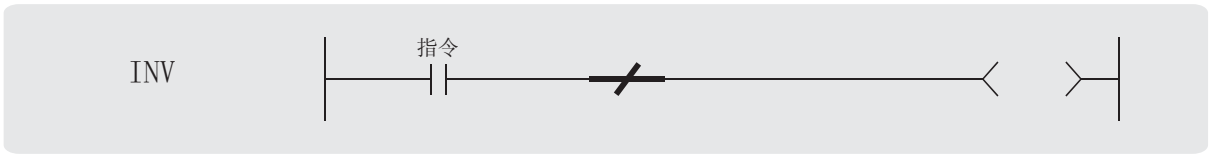


[列表模式]

步	指令	软元件
0	LD	X0
1	MPS	
2	AND	X1
3	MPS	
4	AND	X2
5	MPS	
6	AND	X3
7	MPS	
8	AND	X4
9	MPS	
10	AND	X5
11	MPS	
12	AND	X6
13	MPS	
14	AND	X7
15	MPS	
16	AND	X8
17	MPS	
18	AND	X9
19	MPS	
20	AND	X0A
21	OUT	Y0
22	MPP	
23	OUT	Y41
24	MPP	
25	OUT	Y42
26	MPP	
27	OUT	Y43
28	MPP	
29	OUT	Y44
30	MPP	
31	OUT	Y45
32	MPP	
33	OUT	Y46
34	MPP	
35	OUT	Y47
36	MPP	
37	OUT	Y48
38	MPP	
39	OUT	Y49
40	MPP	
41	OUT	Y4A
42	END	

5.2.3 运算结果取反 (INV)

Basic | High performance | Process | Redundant | Universal



设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
--	--								

功 能

将 INV 指令之前的运算结果取反。

INV 指令之前的运算结果	INV 指令执行之后的运算结果
OFF	ON
ON	OFF

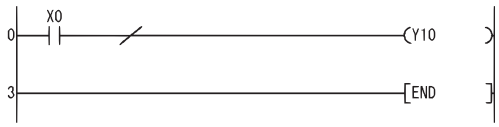
出 错

(1) 在 INV 指令中无运算出错。

程序示例

(1) 以下为将 X0 的 ON/OFF 数据取反后，通过 Y10 输出的程序。

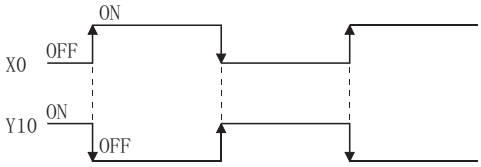
[梯形图模式]



[列表模式]

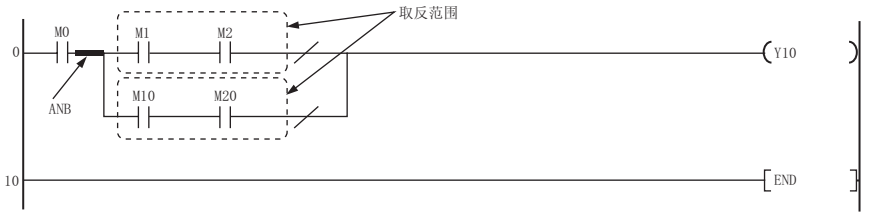
步	指令	软元件
0	LD	X0
1	INV	
2	OUT	Y10
3	END	

[时序图]



要 点

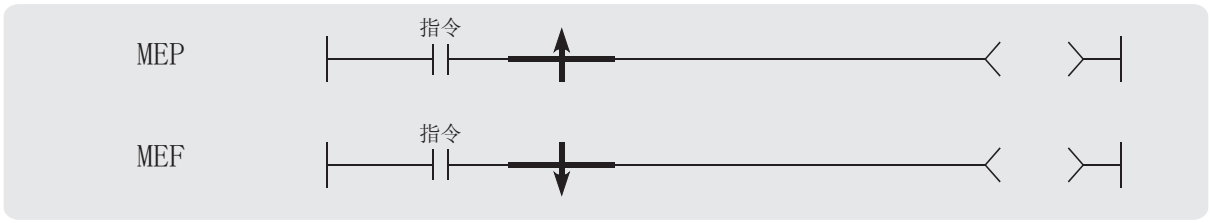
1. 由于 INV 指令是通过 INV 指令之前的运算结果执行动作，因此应在与 AND 指令相同的位置上使用。
   INV 指令不能在 LD、OR 的位置上使用。
2. 使用了梯形图块时，按梯形图块的范围对运算结果进行取反。在使 INV 指令及 ANB 指令并用的梯形图动作时，应注意取反范围。



关于 ANB 指令的详细内容，请参阅 5.2.1 项。

5.2.4 运算结果脉冲化 (MEP、MEF)

Basic | High performance | Process | Redundant | Universal



设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
--	--								

功 能

MEP

(1) MEP 指令前的运算结果为上升沿 (OFF → ON) 时，该指令将变为 ON(导通状态)。
如果 MEP 指令前的运算结果为上升沿以外，则该指令将变为 OFF(非导通状态)。

(2) 当多触点串行连接时，MEP 指令的使用简化了脉冲化处理。

MEF

(1) MEF 指令前的运算结果为下降沿 (ON → OFF) 时，该指令将变为 ON(导通状态)。
如果 MEF 指令前的运算结果为下降沿以外，则该指令将变为 OFF(非导通状态)。

(2) 当多触点串行连接时，MEF 指令的使用简化了脉冲化处理。

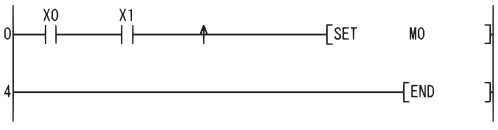
出 错

(1) 在 MEP、MEF 指令中无运算出错。

程序示例

(1) 以下为对 X0 与 X1 的运算结果进行脉冲化的程序：

[梯形图模式]



[列表模式]

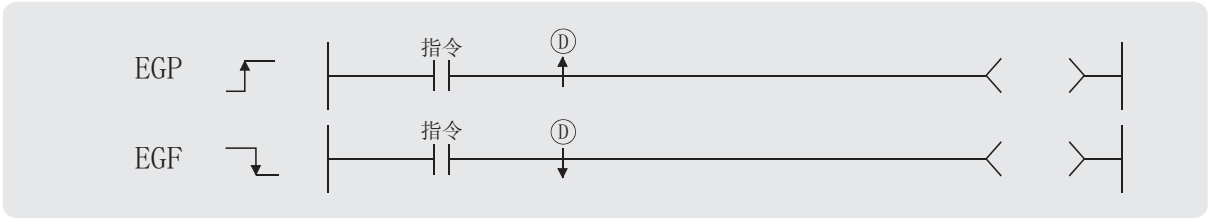
步	指令	软元件
0	LD	X0
1	AND	X1
2	MEP	
3	SET	M0
4	END	

要 点

1. 在子程序或 FOR ～ NEXT 指令等中，如果使用 MEP 和 MEF 指令对进行了变址修饰的触点进行脉冲化，有可能会动作不正常。
在子程序或 FOR ～ NEXT 指令等中对进行了变址修饰的触点执行脉冲化时，应使用 EGP/EGF 指令。
2. 由于 MEP 及 MEF 指令是通过 MEP/MEF 指令之前的运算结果执行动作的，因此应在与 AND 指令相同的位置上使用。
MEP 及 MEF 指令不能在 LD 或 OR 的位置上使用。

5.2.5 变址继电器运算结果的脉冲化 (EGP、EGF)

Basic | High performance | Process | Redundant | Universal



Ⓓ：记忆运算结果的变址继电器编号 (位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				V
Ⓓ	--								○

功 能

EGP

(1) 将 EGP 指令之前的运算结果记忆到变址继电器 (V) 中。

(2) EGP 指令之前的运算结果为上升沿 (OFF → ON) 时，该指令变为 ON(导通状态)。
如果 EGP 指令之前的运算结果为除上升沿以外 (ON → ON、ON → OFF、OFF → OFF)，该指令将变为 OFF(非导通状态)。

(3) EGP 指令用于在子程序或 FOR ～ NEXT 之间对进行了变址修饰的程序执行脉冲运算。

(4) EGP 指令可以同 AND 指令一起使用。

EGF

(1) 将 EGF 指令之前的运算结果通过变址寄存器 (V) 进行记忆。

(2) EGF 指令之前的运算结果为下降沿 (ON → OFF) 时，该指令将变为 ON(导通状态)。
EGF 指令之前的运算结果为下降沿以外 (OFF → ON、ON → ON、OFF → OFF) 时，则变为 OFF (非导通状态)。

(3) EGF 指令用于在子程序及 FOR → NEXT 之间对进行了变址修饰的程序执行脉冲运算。

(4) EGP 指令可以同 AND 指令一起使用。

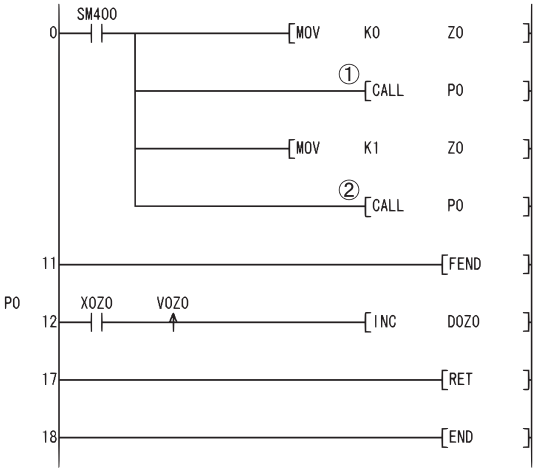
出 错

(1) 在 EGP、EGF 指令中无运算出错。

程序示例

(1) 以下为在子程序中使用 EGP 指令的程序。

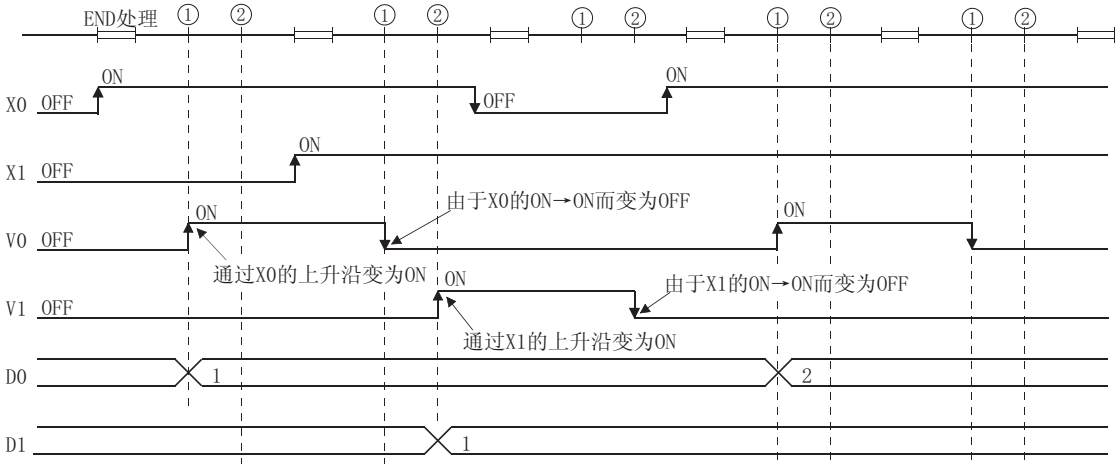
[梯形图模式]



[列表模式]

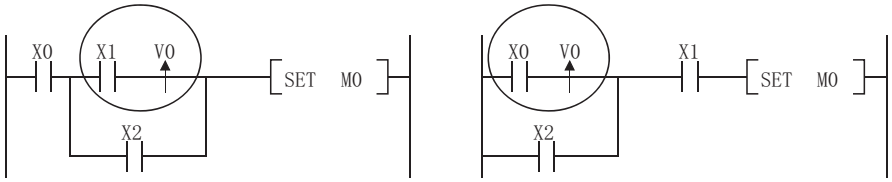
步	指令	软元件	
0	LD	SM400	
1	MOV	K0	Z0
4	CALL	P0	
6	MOV	K1	Z0
9	CALL	P0	
11	FEND		
12	P0		
13	LD	X0Z0	
14	EGP	V0Z0	
15	INC	D0Z0	
17	RET		
18	END		

[动作]



要 点

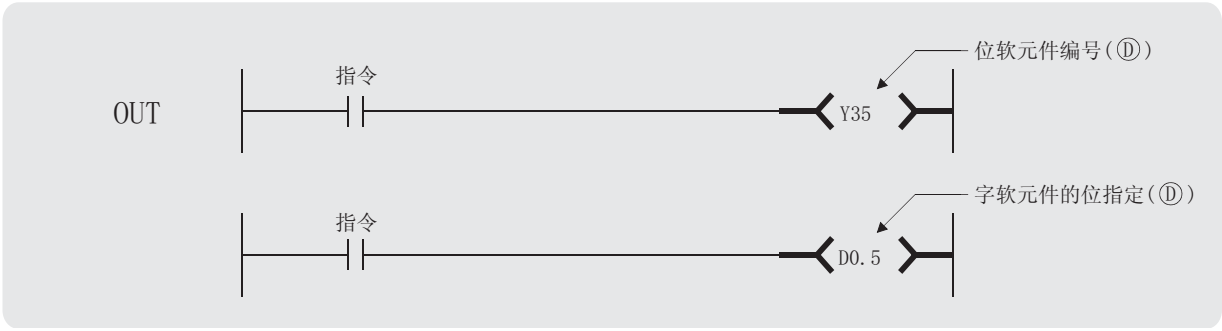
1. 由于 EGP 和 EGF 指令是通过 EGP/EGF 指令之前的运算结果执行动作的，因此应在与 AND 指令（参阅 5.1.1 项）相同的位置上使用。
   EGP 和 EGF 指令不能在 LD 或 OR 指令的位置上使用。
2. EGP 和 EGF 指令不能在下面所示的梯形图块位置使用。



5.3 输出指令

5.3.1 输出指令 (除定时器、计数器、报警器以外) (OUT)

Basic | High performance | Process | Redundant | Universal



Ⓓ :ON/OFF 的软元件编号 (位)

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它 DY
	位	字		位	字				
Ⓓ	○ (除 T、C、F 以外)		○				--		○

功 能

(1) 将 OUT 指令前的运算结果输出到指定的软元件中。

(a) 使用位软元件时

运算结果	线圈
OFF	OFF
ON	ON

(b) 字软元件的位指定时

运算结果	指定位
OFF	0
ON	1

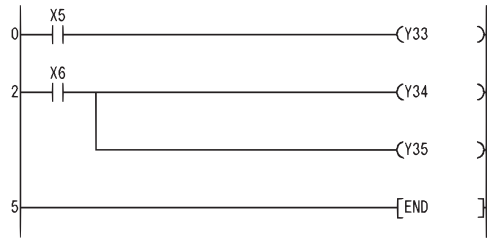
出 错

(1) 在 OUT 指令中无运算出错。

程序示例

(1) 使用位软元件时

[梯形图模式]

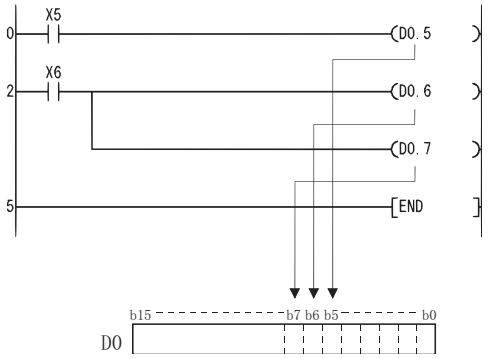


[列表模式]

步	指令	软元件
0	LD	X5
1	OUT	Y33
2	LD	X6
3	OUT	Y34
4	OUT	Y35
5	END	

(2) 字软元件的位指定时

[梯形图模式]



[列表模式]

步	指令	软元件
0	LD	X5
1	OUT	D0.5
2	LD	X6
3	OUT	D0.6
4	OUT	D0.7
5	END	

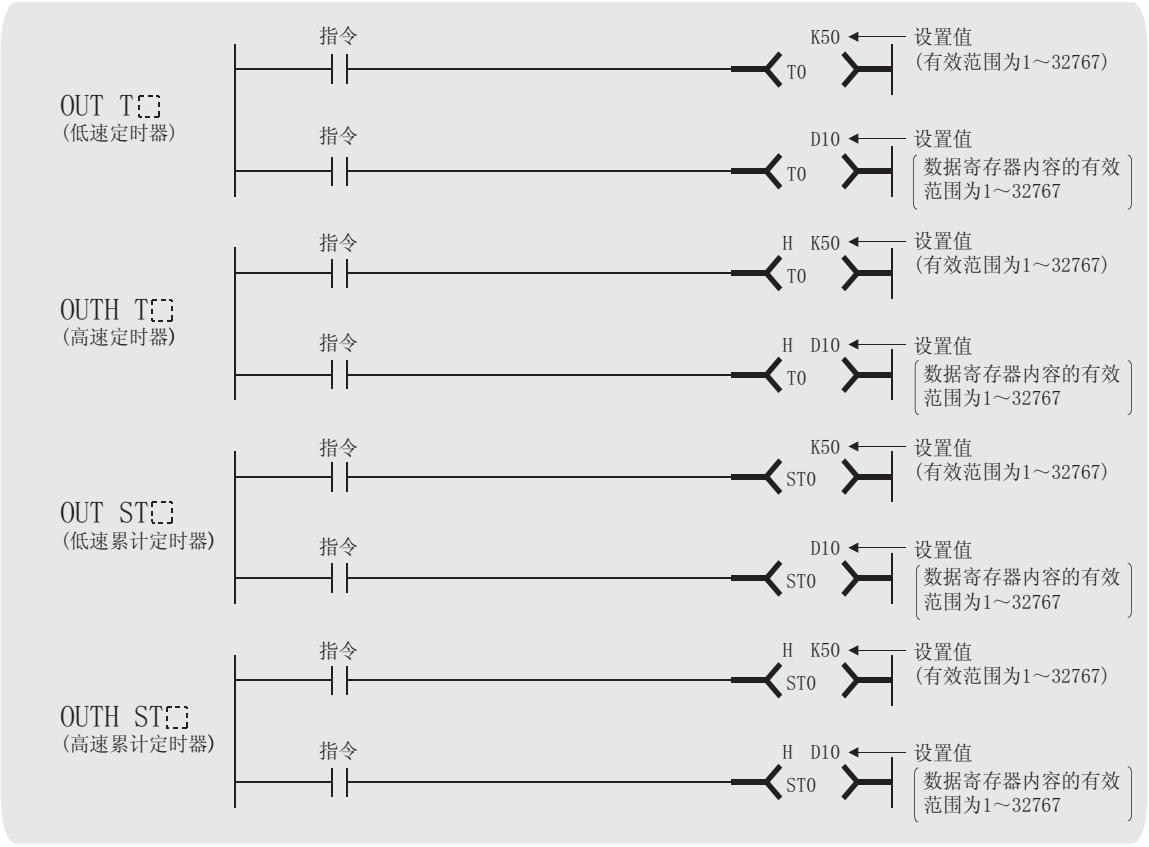
备 注

OUT 指令的基本步数如下：

- 使用内部软元件或文件寄存器 (R) 时 : 1
- 使用直接访问输出 (DY) 时 : 2
- 使用连号访问方式文件寄存器时
  (只对于通用型 QCPU) : 2
  (除通用型 QCPU 以外) : 3
- 使用除上述以外的其它软元件时 : 3

5.3.2 定时器 (OUT T、OUTH T)

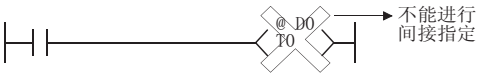
Basic | High performance | Process | Redundant | Universal



Ⓓ　：定时器编号 (位)
设置值 ：定时器的设置值 (BIN16 位 *1)

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K	其它
	位	字		位	字				
Ⓓ	○ (仅 T)	--	--	--	--	--	--	--	--
设置值	--	○T、C 以外	○	--		○	--	○*2	--

*1: 不能对定时器值的设置进行间接指定。



关于间接指定，请参阅 3.4 节。

*2: 定时器的设置值只能使用 10 进制常数 (K)。不能使用 16 进制常数 (H) 和实数。

☆功　能

(1) OUT 指令之前的运算结果为 ON 时，则定时器的线圈将为 ON 且定时器计数到设置值为止：
当到达“ 时间到 ”(计数≥设置值) 状态时，触点变为如下状态：

a 触点	导通
b 触点	非导通

(2) OUT 指令前的运算结果为 ON → OFF 时，触点变为如下状态：

定时器类型	定时器线圈	定时器当前值	在时间到之前		在时间到之后	
			a 触点	b 触点	a 触点	b 触点
低速定时器	OFF	0	非导通	导通	非导通	导通
高速定时器						
低速累计定时器	OFF	保持当前值	非导通	导通	导通	非导通
高速累计定时器						

(3) 在时间到之后，通过 RST 指令进行累计定时器当前值的清除及触点的 OFF。

(4) 不能将设置值设置为负数 (-32768 ～ -1)。 *3
如果设置值为 0，当执行 OUT 指令时，定时器将会变为 “ 时间到 ” 状态。

*3: 通过字软元件 ((D、W、R、ZR、J□\□、U□\G□) 指定定时器设置时，不进行设置值的范围检查。为了防止设置值中被输入了负数，应通过用户程序进行设置值的范围检查。

(5) 执行 OUT 指令时，将进行下列处理：

· OUT T□的线圈的 ON/OFF

· OUT T□的触点的 ON/OFF

· OUT T□的当前值的变更
如果在 OUT T□指令为 ON 的状态下，通过 JMP 指令等跳过了 OUT T□指令时，将不进行当前值的更新以及触点的 ON/OFF 动作。
此外，如果在同一个扫描内同一个 OUT T□指令被执行了 2 次以上，则按执行次数对当前值进行更新。

(6) 定时器线圈 / 触点的变址修饰只能通过 Z0 或 Z1 进行。
对于定时器的设置值，无变址修饰方面的限制。

备 注

1. 关于定时器的时限
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置中进行设置。

定时器类型	QCPU	
	设置范围	设置单位
低速定时器 低速累计定时器	1ms ～ 1000ms (默认值： 100ms)	1ms
高速定时器 高速累计定时器	0.1ms ～ 100ms (默认值： 10.0ms)	0.1ms

2. 关于定时器的计数方法，请参阅所使用的 CPU 模块的用户手册 (功能解说 / 程序基础篇)。
3. OUT T□指令的基本步数为 4 步。

出 错

(1) OUT T□指令中无运算出错。

注意事项

(1) 创建通过定时器的触点开始进行其它定时器的计测的程序时，应按照从后计测的定时器开始的顺序进行编程。

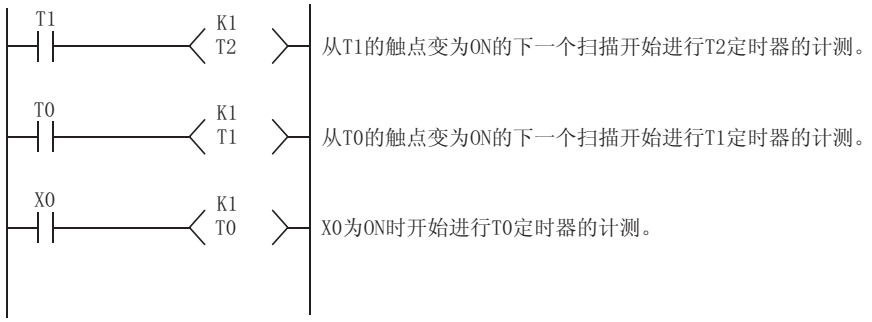
在下列情况下，如果按照计测顺序进行编程，则全部的定时器将在同一个扫描中变为 ON。

· 设置值小于扫描时间时。

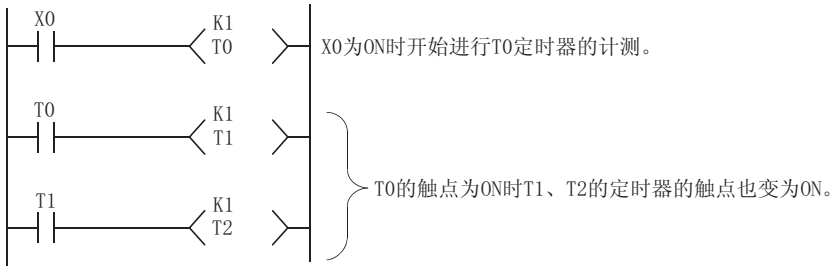
· 设置值为“ 1 ”时。

例

· 对 T0 ～ T2 的定时器按照从后计测的定时器开始的顺序进行了编程时。



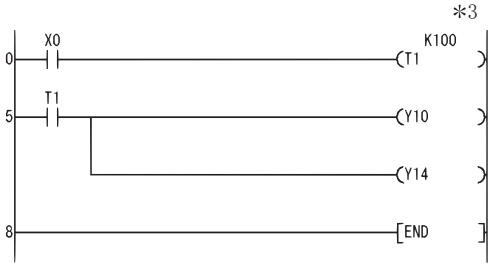
· 对 T0 ～ T2 的定时器按照计测顺序进行了编程时。



程序示例

(1) 以下为在 X0 变为 ON 的 10 秒后使 Y10、Y14 变为 ON 的程序。

[梯形图模式]



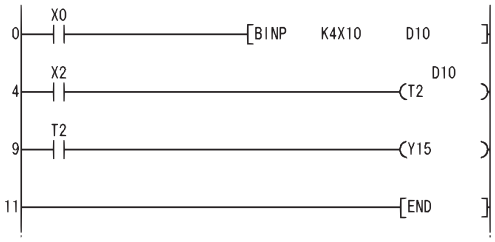
[列表模式]

步	指令	软元件	
0	LD	X0	
1	OUT	T1	K100
5	LD	T1	
6	OUT	Y10	
7	OUT	Y14	
8	END		

*3: 表示低速定时器的设置值为默认值时限 (100ms) 的情况下。

(2) 以下为将 X10 ～ X1F 的 BCD 数据作为定时器的设置值的程序。

[梯形图模式]



将X10～X1F的BCD数据转换为BIN数据后，存储到D10中。

X2变为ON时，将D10中存储的数据作为设置值进行计数。

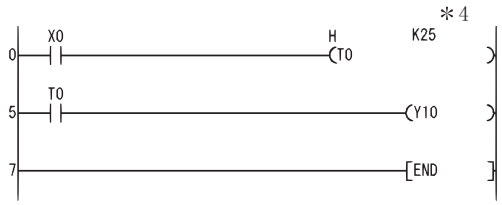
T2变为“计数到”状态时Y15变为ON。

[列表模式]

步	指令	软元件	
0	LD	X0	
1	BINP	K4X10	D10
4	LD	X2	
5	OUT	T2	D10
9	LD	T2	
10	OUT	Y15	
11	END		

(3) 以下为在 X0 变为 ON 的 250ms 后使 Y10 变为 ON 的程序。

[梯形图模式]



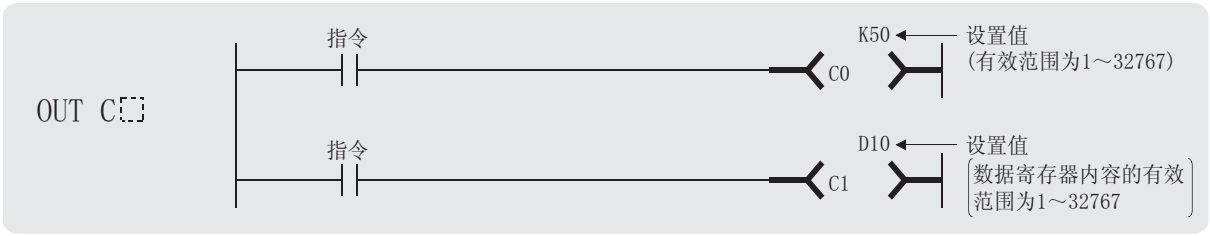
[列表模式]

步	指令	软元件	
0	LD	X0	
1	OUTH	T0	K25
5	LD	T0	
6	OUT	Y10	
7	END		

*4: 表示高速定时器的设置值为默认值时限 (10ms) 的情况下。

5.3.3 计数器 (OUT C)

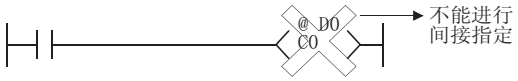
Basic | High performance | Process | Redundant | Universal



Ⓓ　：计数器编号 (位)。
设置值 ：计数器设置值 (BIN16 位 *1)

设置数据	内部软元件		R 、ZR	J□\□		U□\G□	Zn	常数 K	其它
	位	字		位	字				
Ⓓ	○ (仅 C)	--	--	--	--	--	--	--	--
设置值	--	○ (除 T、C 以外)	○	--	○		--	○*2	--

*1:　不能对计数器的设置值进行间接指定。



关于间接指定，请参阅 3.4 节。

*2:　定时器的设置值只能使用 10 进制常数 (K)。不能使用 16 进制常数 (H) 和实数。

功　能

(1) OUT 指令之前的运算结果为 OFF → ON 时，在当前值 (计数值) 上加上 1，变为 “ 计数到 ” (当前值≥设置值) 状态时，触点变为如下状态 :

a 触点	导通
b 触点	非导通

(2) 当运算结果为 ON 不变时，不进行任何计数。(不需要进行计数输入的脉冲化。)

(3) 变为 “ 计数到 ” 状态后，在执行 RST 指令之前，计数值及触点的状态不发生变化。

(4) 设定值不能为负数 (-32768 ～ -1)。
此外，设置值为 0 时，将进行与 1 相同的处理。

(5) 计数器线圈 / 触点的变址修饰只能通过 Z0 或 Z1 进行。
对于计数器的设置值，无变址修饰方面的限制。

备 注

1. 关于计数器的计数方法，请参阅所使用的 CPU 模块的用户手册 (功能解说 / 程序基础篇)。
2. OUT C□指令的基本步数为 4 步。

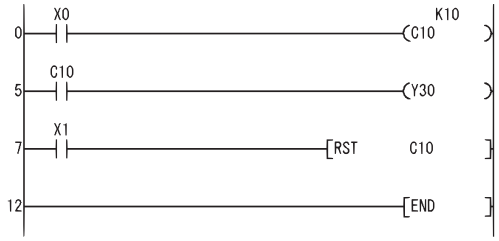
出 错

(1) 在 OUT C□指令中无运算出错。

程序示例

(1) 以下为 X0 10 次 ON 之后 Y30 变为 ON，X1 变为 ON 时对计数器进行复位的程序。

[梯形图模式]

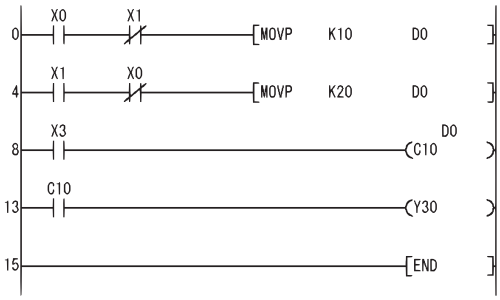


[列表模式]

步	指令	软元件	
0	LD	X0	
1	OUT	C10	K10
5	LD	C10	
6	OUT	Y30	
7	LD	X1	
8	RST	C10	
12	END		

(2) 以下为 X0 为 ON 时将 C10 的设置值设置为 10，X1 为 ON 将 C10 的设置值设置为 20 的程序。

[梯形图模式]

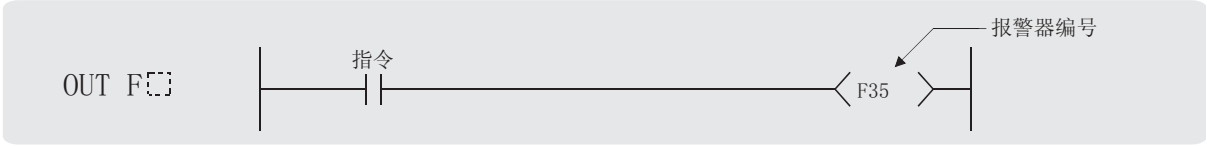


[列表模式]

步	指令	软元件	
0	LD	X0	
1	ANI	X1	
2	MOVP	K10	D0
4	LD	X1	
5	ANI	X0	
6	MOVP	K20	D0
8	LD	X3	
9	OUT	C10	D0
13	LD	C10	
14	OUT	Y30	
15	END		

5.3.4 报警器输出 (OUT F)

Basic | High performance | Process | Redundant | Universal



Ⓓ：变为 ON 的报警器编号 (位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
Ⓓ	○ (仅 F)		--						

功 能

(1) 将 OUT 指令之前的运算结果输出到指定的报警器。

(2) 当报警器 (F) 为 ON 时的情况如下所示：

· “ USER ”/ “ ERR. ” LED 亮灯。

· 将变为 ON 的报警器号 (F 编号) 储存到特殊寄存器 (SD64 ～ SD79) 中。

· 将 SD63 的值增加 1。

(3) SD63 的值变为 16(有 16 个报警器已变为 ON) 时，即使又有报警器变为 ON，变为 ON 的报警器编号也不会储存到 SD64 ～ SD79 中。

(4) 通过 OUT 指令将报警器变为 OFF 时的情况如下所示：

线圈变为 OFF，但是 “ USER ”/ “ ERR. ” LED 的状态及储存在 SD63 ～ SD79 中的内容不发生变化。
希望使 “ USER ”/ “ ERR ” LED 熄灯，从 SD63 ～ SD79 中删除通过 OUT F□指令变为 OFF 的报警器时，可以使用 RST F□指令。

出 错

(1) OUT F□指令中无运算出错。

备 注

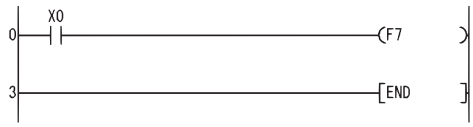
1. 关于报警器的详细内容，请参阅所使用的 CPU 模块的用户手册（功能解说 / 程序基础篇）。
2. OUT F□指令的基本步数为 2 步。
3. CPU 模块前面的 LED 显示器或“ USER ”LED 的 CPU 模块类型如下表所示：

LED 类型	CPU 模块型号
“ USER ”LED	高性能型 QCPU、过程 CPU、冗余 CPU、通用型 QCPU
“ ERR. ”LED	基本型 QCPU

程序示例

(1) 以下为 X0 为 ON 时使 F7 变为 ON，且将 7 存储到 SD64 ～ SD79 中的程序。

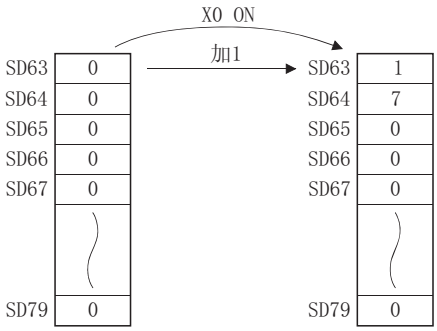
[梯形图模式]



[列表模式]

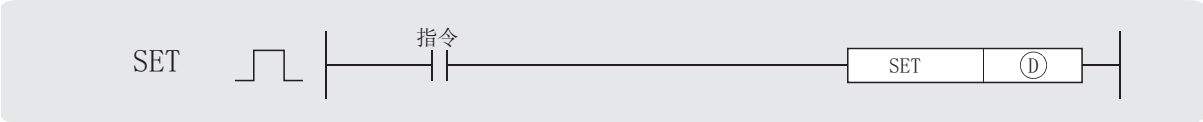
步	指令	软元件
0	LD	X0
1	OUT	F7
3	END	

[动作]



5.3.5 软元件的设置（报警器除外）(SET)

Basic | High performance | Process | Redundant | Universal



D ：设置 (ON) 的位软元件编号 / 字软元件的位指定（位）。

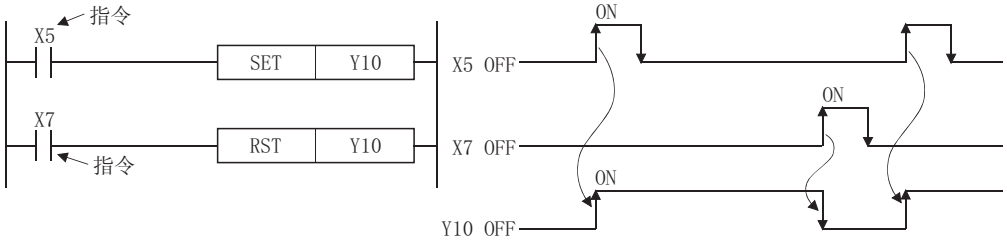
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				BL、DY
D	○	○（除 T、C 以外）		○			--		○

功 能

(1) 当执行指令为 ON 时，指定软元件的状态如下：

软元件	软元件状态
位软元件	使线圈、触点变为 ON。
字软元件的位指定时	将指定位变为 1。

(2) 对于变为 ON 的软元件，即使执行指令变为 OFF，也仍将保持为 ON 状态不变。
通过 SET 指令变为 ON 软元件可以通过 RST 指令使其变为 OFF。



(3) 当执行命令为 OFF 时，软元件的状态不发生变化。

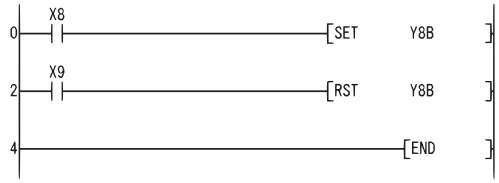
出 错

(1) SET 指令中无运算出错。

程序示例

(1) 以下为在 X8 为 ON 时对 Y8B 进行设置 (ON)，在 X9 为 ON 时对 Y8B 进行复位 (OFF) 的程序。

[梯形图模式]

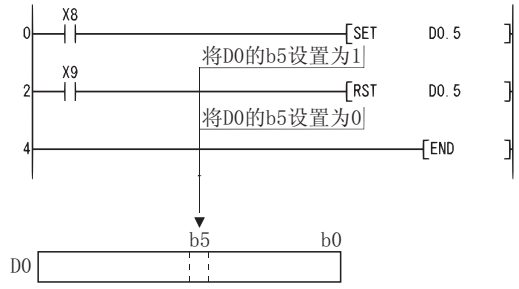


[列表模式]

步	指令	软元件
0	LD	X8
1	SET	Y8B
2	LD	X9
3	RST	Y8B
4	END	

(2) 以下为在 X8 为 ON 时将 D0 的位 5(b5) 设置为 1，在 X9 为 ON 时将 D0 的位 5(b5) 设置为 0 的程序。

[梯形图模式]



[列表模式]

步	指令	软元件
0	LD	X8
1	SET	D0.5
2	LD	X9
3	RST	D0.5
4	END	

备 注

1. SET 指令的基本步数如下所示：
   ·使用内部软元件或文件寄存器 (R0 ～ R32767) 时 : 1
   ·使用直接访问输出 (DY) 或 SFC 程序软元件 (BL) 时 : 2
   ·使用连号访问方式文件寄存器时
   (只对于通用型 QCPU) : 2
   (除通用型 QCPU 以外) : 3
   ·使用除上述以外的其它软元件时 : 3

2. 当 X 被当作软元件使用时，应使用实际输入中未使用的软元件号。如果使用了与实际输入软元件相同的号，则实际输入的数据将被通过 SET 指令指定的输入 X 所覆盖。

5.3.6 软元件的复位（报警器除外）(RST)

Basic High performance Process Redundant Universal

D ：复位的位软元件编号 / 字软元件的位指定（位）。
复位的位软元件编号 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				DY
D	○							--	○

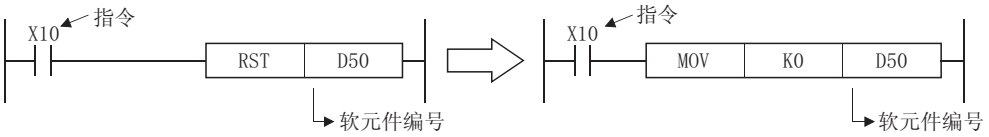
功 能

(1) 当执行命令为 ON 时，指定软元件的状态如下所示：

软元件	软元件状态
位软元件	将线圈及触点变为 OFF。
定时器、计数器	将当前值设置为 0，并将线圈及触点变为 OFF。
字软元件的位指定时	将指定位设置为 0。
除定时器、计数器以外的字软元件	将内容设置为 0。

(2) 当执行指令为 OFF 时，软元件的状态不发生变化。

(3) 通过 RST 指令指定的字软元件的功能等同于下列梯形图：



出 错

(1) RST 指令中无运算出错。

备 注

RST 指令的基本步数情况如下所示：

a) 对于位处理

·内部软元件（位软元件或字软元件的位指定）	: 1
·直接访问输出	: 2
·定时器、计数器	: 4
·使用连号访问方式文件寄存器时	
（只对于通用型 QCPU)	: 2
（除通用型 QCPU 以外）	: 3
·除上述之外	: 3

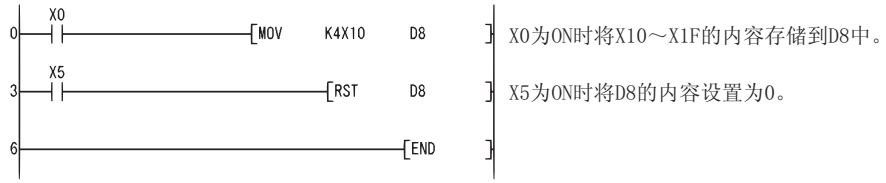
b) 对于字处理

·内部软元件	: 2
·变址寄存器	: 2
·使用连号访问方式文件寄存器时	
（只对于通用型 QCPU)	: 2
（除通用型 QCPU 以外）	: 3
·除上述之外	: 3

程序示例

(1) 以下为将数据寄存器的值设置为 0 的程序。

[梯形图模式]

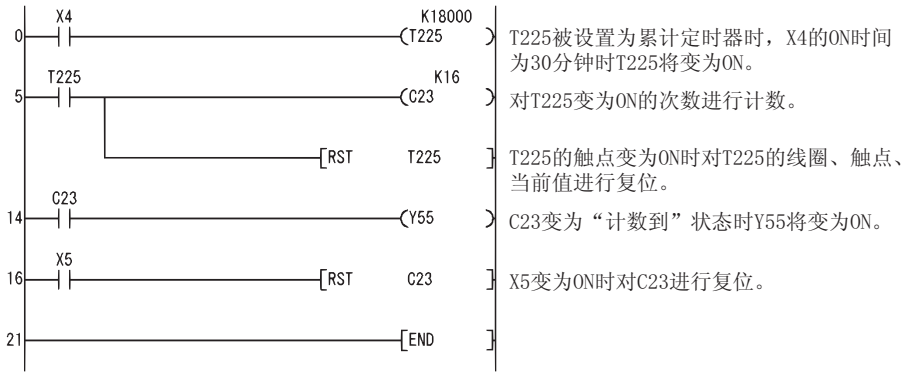


[列表模式]

步	指令	软元件	
0	LD	X0	
1	MOV	K4X10	D8
3	LD	X5	
4	RST	D8	
6	END		

(2) 以下为对 100ms 累计定时器和计数器进行复位的程序。

[梯形图模式]

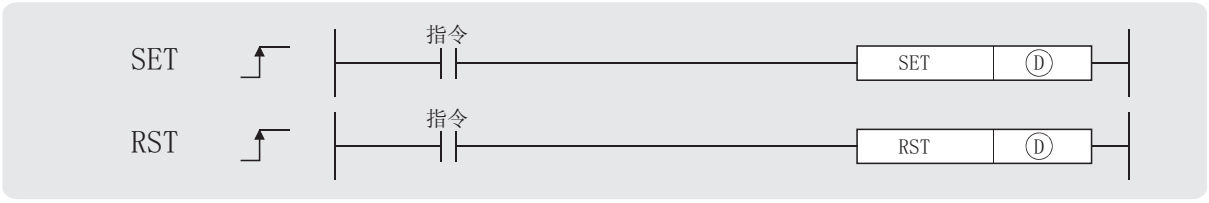


[列表模式]

步	指令	软元件	
0	LD	X4	
1	OUT	T225	K18000
5	LD	T225	
6	OUT	C23	K16
10	RST	T225	
14	LD	C23	
15	OUT	Y55	
16	LD	X5	
17	RST	C23	
21	END		

5.3.7 报警器的设置和复位 (SET F、RST F)

Basic | High performance | Process | Redundant | Universal



SET Ⓓ ：设置的报警器编号 (F 编号) (位)。

RST Ⓓ ：复位的报警器编号 (F 编号) (位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
Ⓓ	○ (仅 F)		--						

功 能

SET

(1) 当执行指令变为 ON 时，将Ⓓ中指定的报警器变为 ON。

(2) 报警器 (F) 变为 ON 的情况如下所示：

· “ USER ” LED 亮灯。*1

· 将变为 ON 的报警器编号 (F 编号) 存储到特殊寄存器 (SD64 ～ SD79) 中。

· SD63 的值加 1。

*1: 使用基本型 QCPU 时，“ ERR. ” LED 亮灯。

(3) SD63 的值为 16(有 16 个报警器变为 ON) 时，即使又有报警器变为 ON，变为 ON 的报警器编号也不会被存储到 SD64 ～ SD79 中。

RST

(1) 执行指令变为 ON 时，将Ⓓ中指定的报警器变为 OFF。

(2) 对于已变为 OFF 的报警器编号 (F 编号)，将其从特殊寄存器 (SD64 ～ SD79) 中删除，且 SD63 的值被减 1。

备 注

1. 关于报警器的详细内容，请参阅所使用的 CPU 模块的用户手册 (功能解说 / 程序基础篇)。

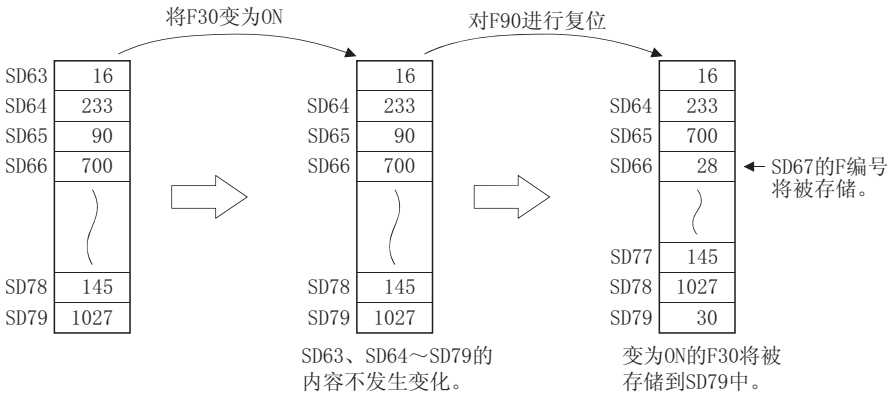
2.SET F□、RST F□指令的基本步数为 2 步。

(3) 当 SD63 的值为 16 时，通过 RST 指令将报警器号从 SD64 ～ SD79 中删除。此外，如果未在 SD64 ～ SD79 中进行编号登录的报警器变为 ON 时，这些报警器号将被新建登录。

如果 SD64 ～ SD79 的报警器号均变为 OFF，在 CPU 模块前面的 LED 显示或“ USER ”LED 将会熄灯。*2

*2：当使用基本型 QCPU 时，“ ERR. ”LED 将熄灯。

[当 SD63 为 16 时的动作]



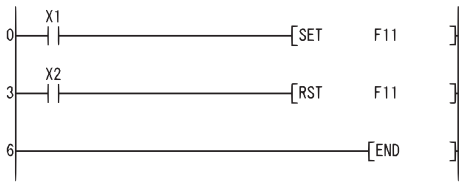
出　错

(1) SET F□、RST F□指令中无运算出错。

程序示例

(1) 以下为 X1 变为 ON 时，将报警器 F11 变为 ON，并将 11 存储到特殊寄存器 (SD64 ～ SD79) 中。此外，当 X2 变为 ON 之后对报警器 F11 进行复位，并从特殊寄存器 (SD64 ～ SD79) 中删除值 11。

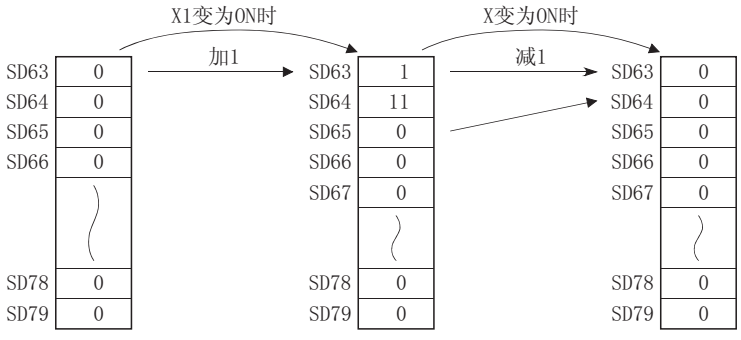
[梯形图模式]



[列表模式]

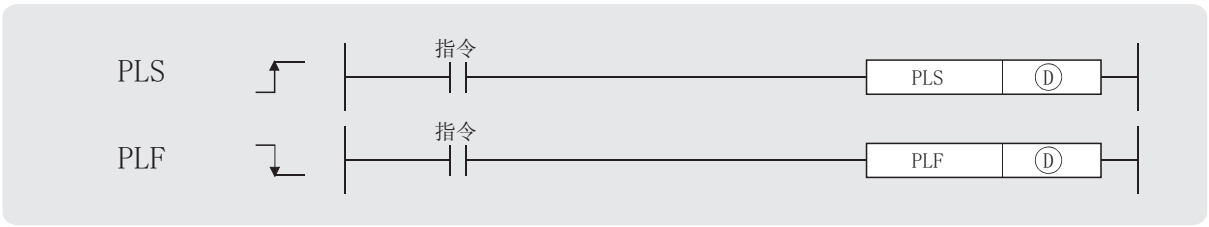
步	指令	软元件
0	LD	X1
1	SET	F11
3	LD	X2
4	RST	F11
6	END	

[动作]



5.3.8 上升沿和下降沿输出 (PLS、PLF)

Basic | High performance | Process | Redundant | Universal



Ⓓ：脉冲化的软元件（位）。

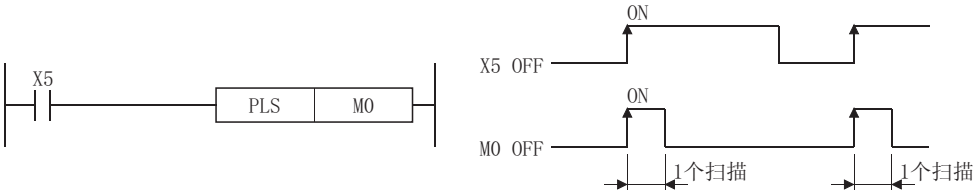
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				DY
Ⓓ	○						--		○

功 能

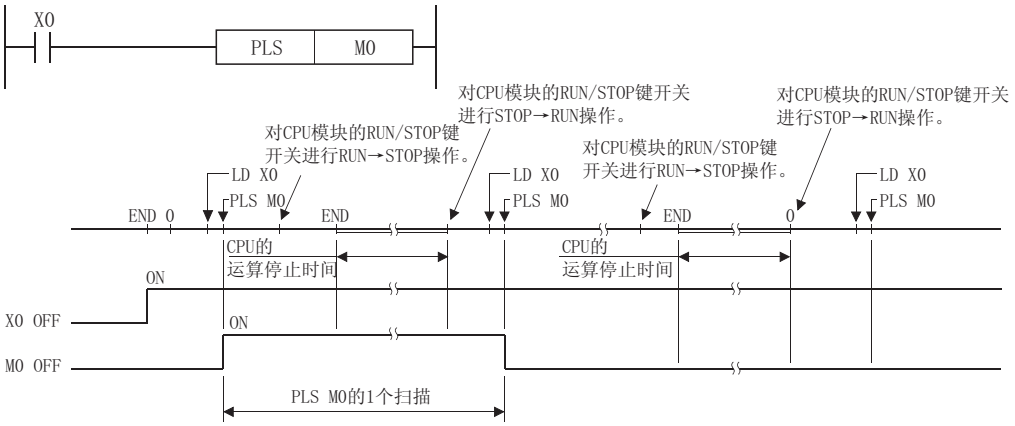
PLS

(1) 执行指令为 OFF → ON 时将指定软元件变为 ON，在执行指令为除 OFF → ON 以外 (ON → ON、ON → OFF、OFF → OFF) 时将指定软元件变为 OFF。

1 个扫描中Ⓓ中指定的软元件的 PLS 指令为 1 个时，指定软元件将 ON 1 个扫描。
关于在 1 个扫描中多次执行了同一软元件的 PLS 指令时的动作，请参阅 3.9 节。



(2) PLS 指令执行后如果对 RUN/STOP 键开关进行了 RUN → STOP 操作，则即使再次置于 RUN，也不能执行 PLS 指令。



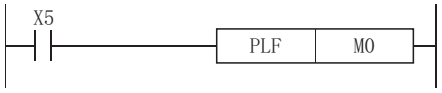
(3) 将锁存继电器 (L) 指定为执行指令后，如果在锁存继电器为 ON 的状态下对电源执行了 OFF → ON 的操作，则执行指令将在第 1 个扫描中变为 OFF → ON，PLS 指令将被执行，指定的软元件将变为 ON。

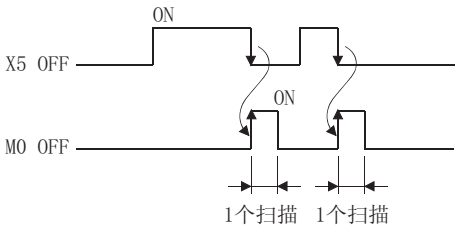
在电源 ON 后的第 1 个扫描中变为 ON 的软元件，通过下一个 PLS 指令变为 OFF。

PLF

(1) 当执行指令为 ON → OFF 时，PLF 指令将指定的软元件变为 ON，在除 ON → OFF 以外 (OFF → OFF、OFF → ON、ON → ON) 时，该指令将指定的软元件变为 OFF。

在 1 个扫描内只有 1 个Ⓓ中指定的软元件的 PLF 指令时，指定的软元件将 ON 1 个扫描周期。
如果同一个软元件在一个扫描周期内 PLF 指令执行多于一次，执行操作请参见 3.9 节。





(2) PLF 指令执行后如果对 RUN/STOP 键开关进行了 RUN → STOP 操作，则即使再次置于 RUN，也不能执行 PLF 指令。

要 点

如果对 PLS、PLF 指令通过 CJ 指令进行了跳转，或未通过 CALL 指令对已执行的子程序进行调用，则Ⓓ中指定的软元件有可能会 ON 1 个扫描以上，应加以注意。

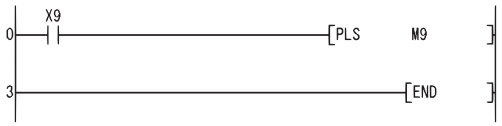
出 错

(1) PLS、PLF 指令中无运算出错。

程序示例

(1) 以下为 X9 为 ON 时执行 PLS 指令的程序。

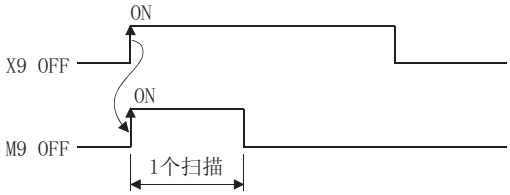
[梯形图模式]



[列表模式]

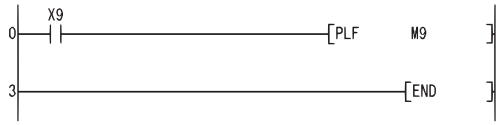
步	指令	软元件
0	LD	X9
1	PLS	M9
3	END	

[时序图]



(2) 以下为 X9 为 ON 时，执行 PLF 指令的程序。

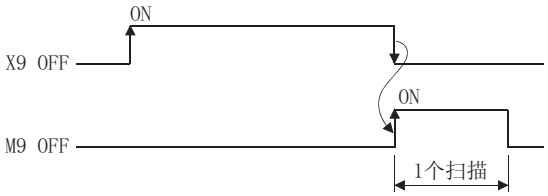
[梯形图模式]



[列表模式]

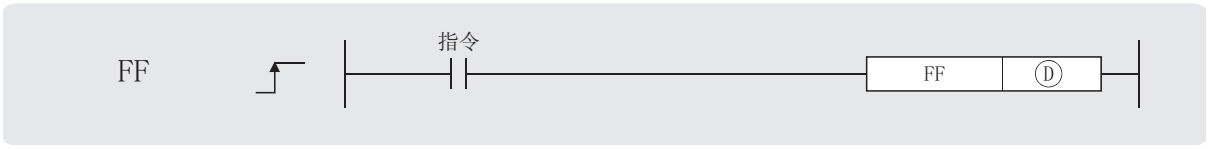
步	指令	软元件
0	LD	X9
1	PLF	M9
3	END	

[时序图]



5.3.9 位软元件输出取反 (FF)

Basic | High performance | Process | Redundant | Universal



Ⓓ：取反的软元件编号（位）

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它 DY
	位	字		位	字				
Ⓓ			○				--		○

功 能

(1) 执行指令为 OFF → ON 时，对Ⓓ中指定的软元件状态进行取反。

软元件	软元件状态	
	在 FF 执行之前	在 FF 执行之后
位软元件	OFF	ON
	ON	OFF
字软元件的位指定	0	1
	1	0

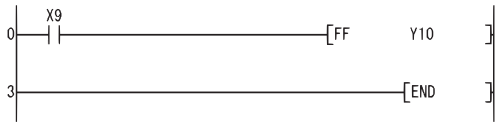
出 错

(1) FF 指令中无运算出错。

程序示例

(1) 以下为 X9 变为 ON 时，对 Y10 的输出进行取反的程序。

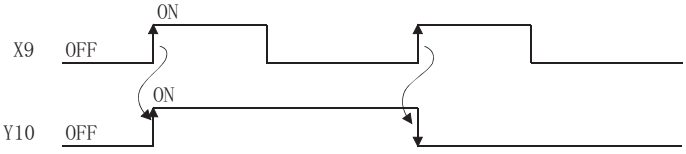
[梯形图模式]



[列表模式]

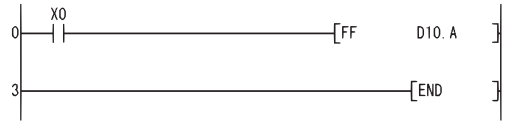
步	指令	软元件
0	LD	X9
1	FF	Y10
3	END	

[时序图]



(2) 以下为 X0 变为 ON 时，对 D10 的 b10(位 10) 进行取反的程序。

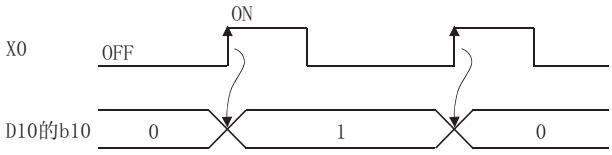
[梯形图模式]



[列表模式]

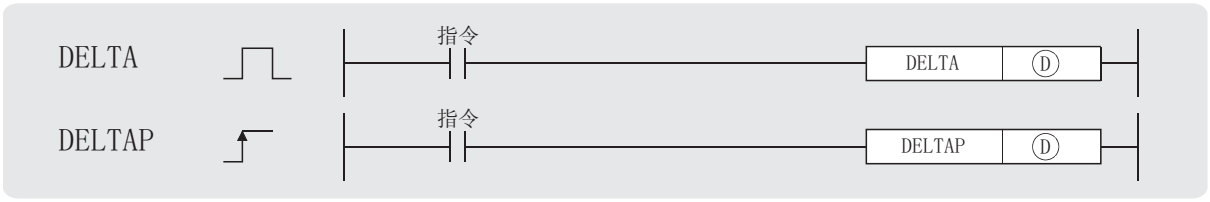
步	指令	软元件
0	LD	X0
1	FF	D10.A
3	END	

[时序图]



5.3.10 直接输出的脉冲化 (DELTA(P))

Basic | High performance | Process | Redundant | Universal



Ⓓ：脉冲化的位（位）

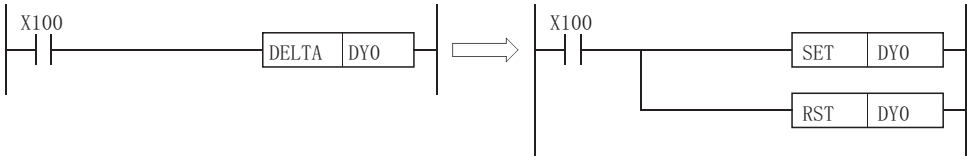
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它 DY
	位	字		位	字				
Ⓓ	--								○

功 能

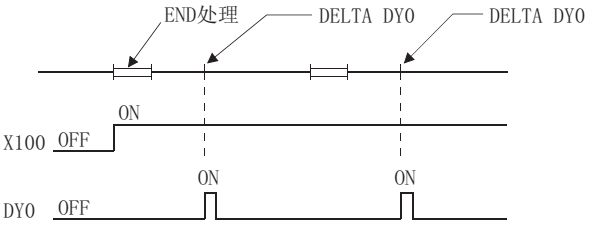
(1) 对Ⓓ中指定的直接访问输出 (DY) 进行脉冲输出。

通过 DELTA　DY0 进行了指定时，其动作与下图所示的使用了 SET/RST 指令的梯形图的动作相同。

[执行了 DELTA 指令的梯形图]　　[使用了 SET/RST 指令的梯形图]



[动作]



(2) DELTA(P) 指令被用作至智能功能模块的启动执行指令。

出 错

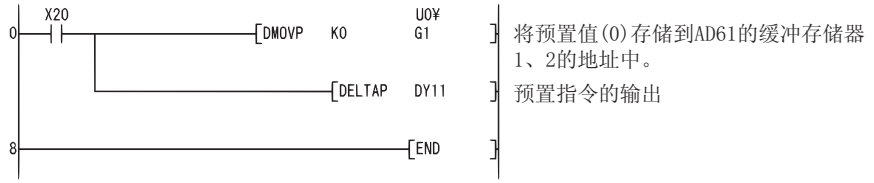
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· Ⓓ中指定的直接访问输出编号超出了 CPU 模块的输出范围时。　　(出错代码： 4101)

程序示例

(1) 以下为 X20 变为 ON 时，对安装在主基板的插槽 0 中的 AD61 的 CH1 进行预置的程序。

[梯形图模式]



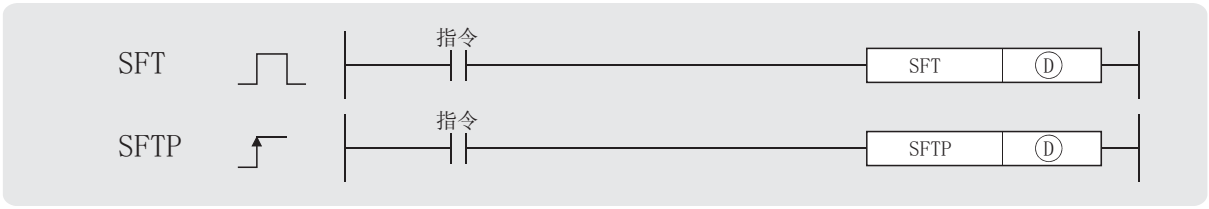
[列表模式]

步	指令	软元件	
0	LD	X20	
1	DMOVP	K0	U0¥G1
6	DELTAP	DY11	
8	END		

5.4 移位指令

5.4.1 位软元件移位 (SFT(P))

Basic | High performance | Process | Redundant | Universal



Ⓓ：移位的软元件的起始编号（位）

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它 DY
	位	字		位	字				
Ⓓ	○（除T、C以外）						--		○

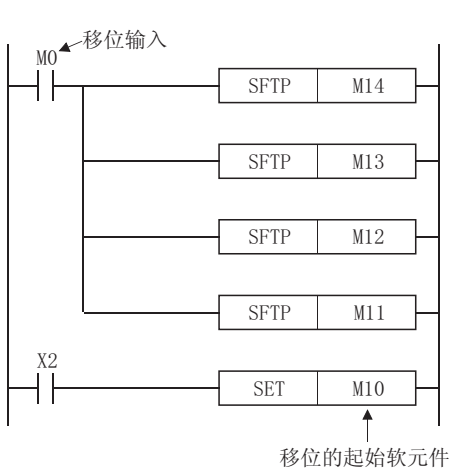
功 能

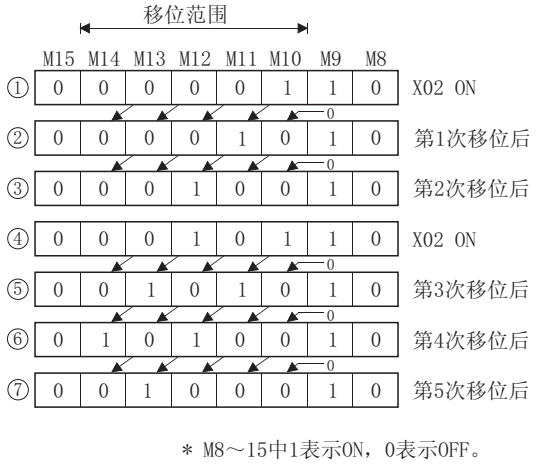
(1) 使用位软元件时

(a) 对于在Ⓓ中指定的软元件，将其前一个号的软元件的 ON/OFF 状态移位到在Ⓓ中指定的软元件中，并且将前一个号的软元件置于 OFF。
例如，通过 SFT 指令指定了 M11 时，在 SFT 指令被执行时将 M10 的 ON/OFF 状态移位至 M11 中，并将 M10 置于 OFF。

(b) 应通过 SET 指令将要移位的起始软元件置于 ON。

(c) 当 SFT 和 SFTP 被连续使用时，应将程序编制为从较大号的软元件处开始执行程序。

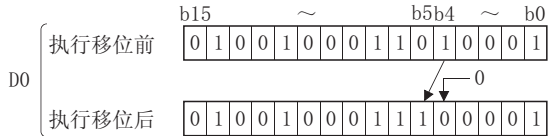




＊ M8～15中1表示ON，0表示OFF。

(2) 使用字软元件的位指定时

(a) 对于在Ⓓ中指定的软元件，将其前一个号的位的 1/0 状态移位到在Ⓓ中指定的位中，并且将前一个号的位置于 0。
例如，通过 SFT 指令指定了 D0.5[D0 的位 5(b5)] 时，SFT 指令执行时将 D0 的 b4 的 1/0 移位到 b5 中，并将 b4 置于 0。



出 错

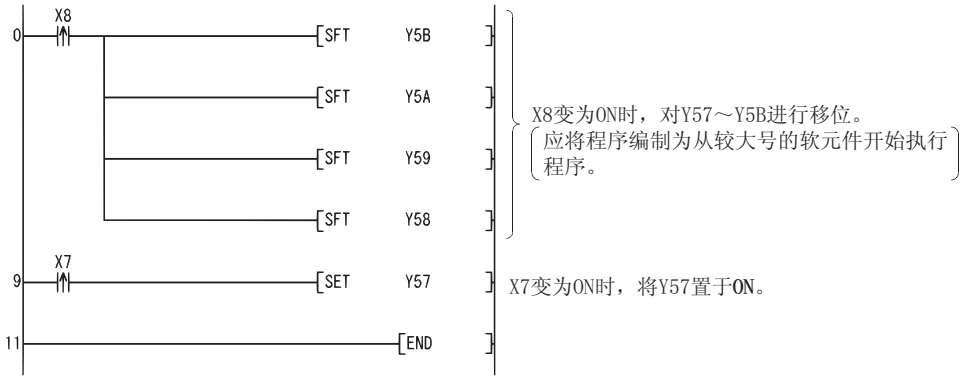
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 在Ⓓ中指定的软元件超出了相应软元件范围时。(仅对于通用型 QCPU)(出错代码： 4101)

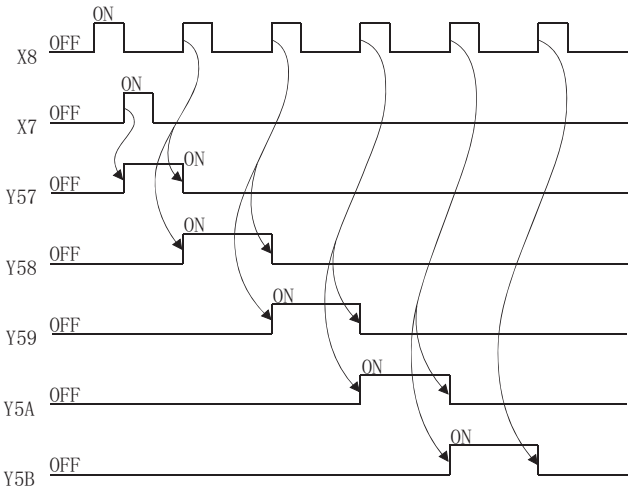
程序示例

(1) 以下为 X8 变为 ON 时，对 Y57 ～ Y5B 进行移位的程序。

[梯形图模式]



[时序图]



[列表模式]

步	指令	软元件
0	LDP	X8
1	SFT	Y5B
3	SFT	Y5A
5	SFT	Y59
7	SFT	Y58
9	LDP	X7
10	SET	Y57
11	END	

5.5 主控指令

5.5.1 主控的设置和复位 (MC、MCR)

Basic | High performance | Process | Redundant | Universal

n ：嵌套 (N0 ～ N14) (嵌套)

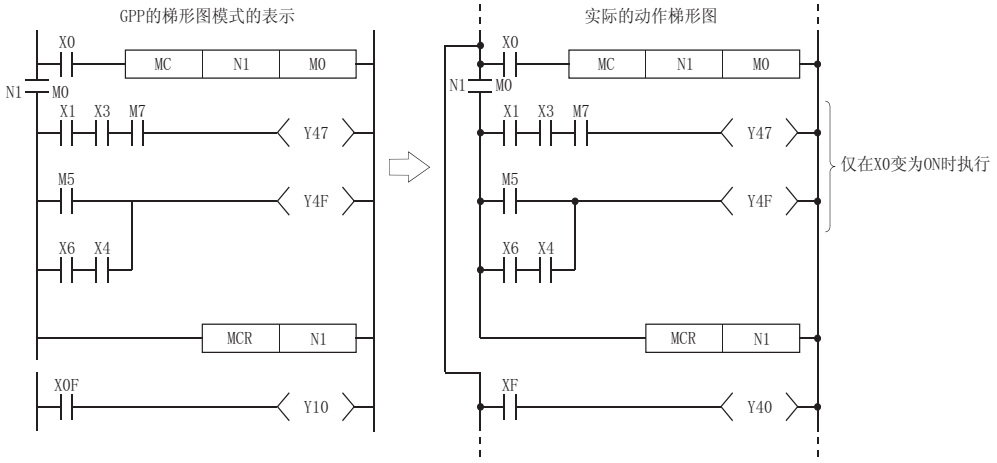
Ⓓ ：使其变为 ON 的软元件编号 (位)

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它	
	位	字		位	字				N	DY
n	--						--		○	--
Ⓓ	○						--		--	○

功 能

主控指令是用于创建通过梯形图公共母线的开闭以进行高效的梯形图切换的顺控程序的指令。

使用了主控的梯形图如下所示：



备 注

在外围设备的写入模式下进行编程时，无需在纵母线上输入触点。

在创建了梯形图之后，执行 “ 转换 ” 操作，置于读取模式时将会自动显示。

MC

(1) 主控开始时，如果 MC 指令的执行指令为 ON，则在 MC 指令与 MCR 指令之间的运算结果如指令 (上述梯形图) 所示。
如果 MC 的执行指令为 OFF，则 MC 指令与 MCR 指令之间的运算结果如下所示：

软元件	软元件状态
高速定时器 低速定时器	计数器值变为 0，将线圈和触点均置于 OFF。
高速累计定时器 低速累计定时器 计数器	线圈变为 OFF 状态，但计数值及触点均保持为当前状态。
在 OUT 指令中的软元件	全部置于 OFF。
SET、RST、 SFT 指令中 } 指令中的软元件	保持为当前状态。

(2) 当 MC 指令为 OFF 时，MC 指令与 MCR 指令之间的指令仍将被执行，因此扫描时间将不会缩短。

要 点

如果在使用了主控的梯形图中存在有不需要触点指令的指令 (FOR ～ NEXT、EI、DI 指令等)，则与 MC 指令的执行指令无关，CPU 模块将执行这些指令。

(3) 通过改变ⓓ中指定的软元件，MC 指令能够任意次使用同一个嵌套 (N) 号。

(4) MC 指令为 ON 时，将ⓓ中指定的软元件线圈置于 ON。
此外，在 OUT 指令等中使用同一软元件时，将会变为双线圈，因此不要在其它指令中使用ⓓ中指定的软元件。

MCR

(1) 该指令为主控的解除指令，表示主控范围的结束。

(2) 在 MCR 指令之前不附加触点指令。

(3) 将具有相同嵌套号的 MC 指令与 MCR 指令配套使用。
但是，在 MCR 指令集中在 1 个位置处的嵌套结构的情况下，通过最小号的 1 个嵌套 (N) 号即可使所有的主控均结束。
(参阅程序示例中的 “ 嵌套结构时的注意事项 ”)

出 错

(1) 在 MC 及 MCR 指令中无运算出错。

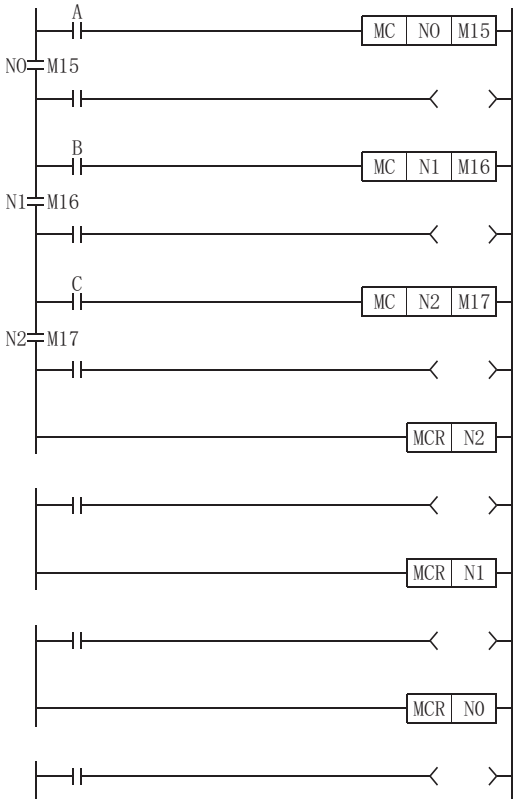
程序示例

主控指令可在嵌套结构中使用。通过嵌套 (N) 把各个主控区分开来。嵌套的最大使用范围为 N0 ～ N14。

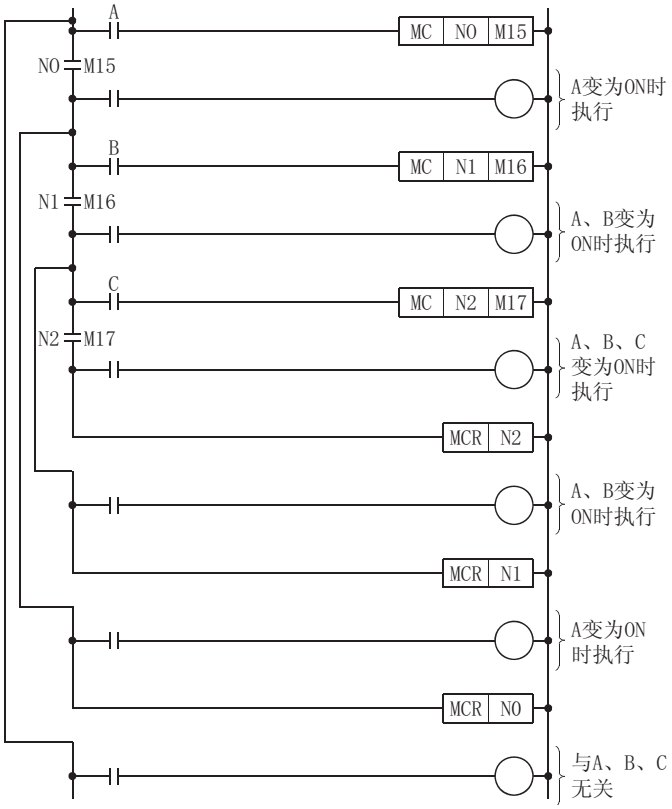
通过使用嵌套结构，可以创建对程序的执行条件进行逐项限制的梯形图。

使用了嵌套结构的梯形图如下所示：

[GPP 梯形图模式下显示的梯形图]



[实际的动作梯形图]



嵌套结构时的注意事项

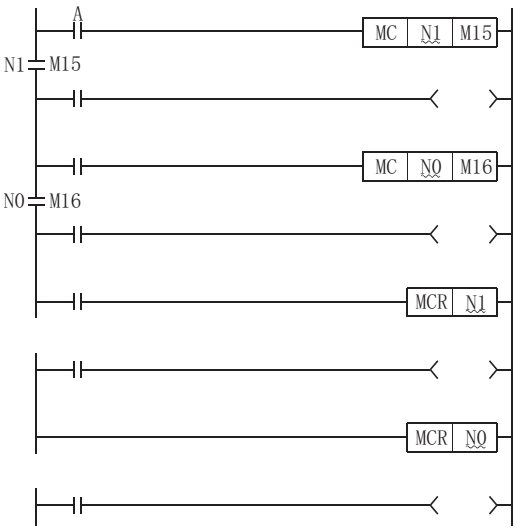
(1) 最多可使用 15 个嵌套 (N0 ～ N14)。

使用嵌套时，在 MC 指令中是按嵌套 (N) 号的从小到大的顺序使用的，而在 MCR 指令中是按从大到小的顺序使用的。

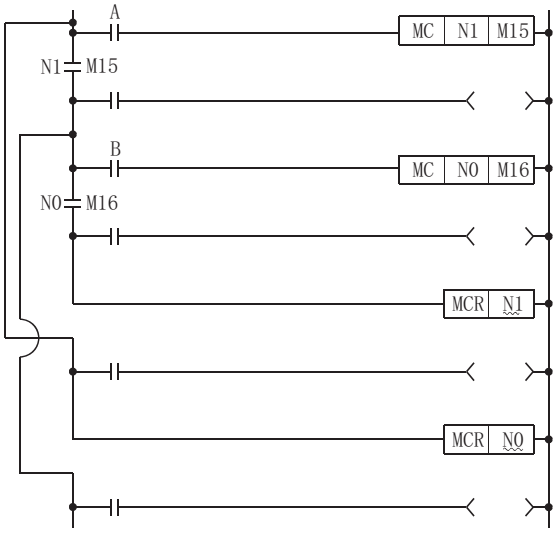
如果该顺序逆向，将不能成为嵌套结构，因此 CPU 模块也不能进行正常运算。

例如，如果在 MC 指令中将嵌套按 N1 → N0 的顺序进行了指定，且在 MCR 指令中也按 N1 → N0 的顺序进行了指定，则纵母线将会交叉。因此不能构成正常的主控梯形图。

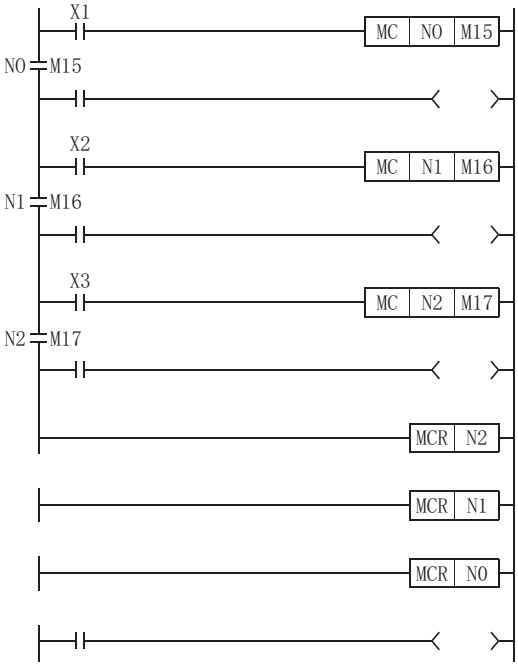
[GPP 梯形图模式下显示的梯形图]

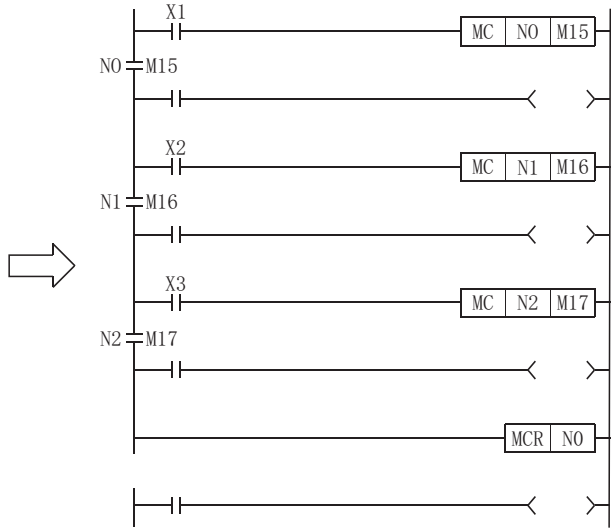


[实际的动作梯形图]



(2) 在 MCR 指令集中在一个位置的嵌套结构的情况下，通过最小号的 1 个嵌套 (N) 号即可使所有的主控均结束。



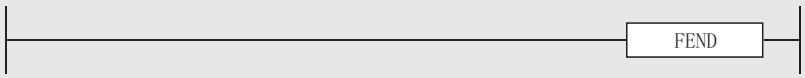


5.6 结束指令

5.6.1 主程序的结束 (FEND)

Basic | High performance | Process | Redundant | Universal

FEND



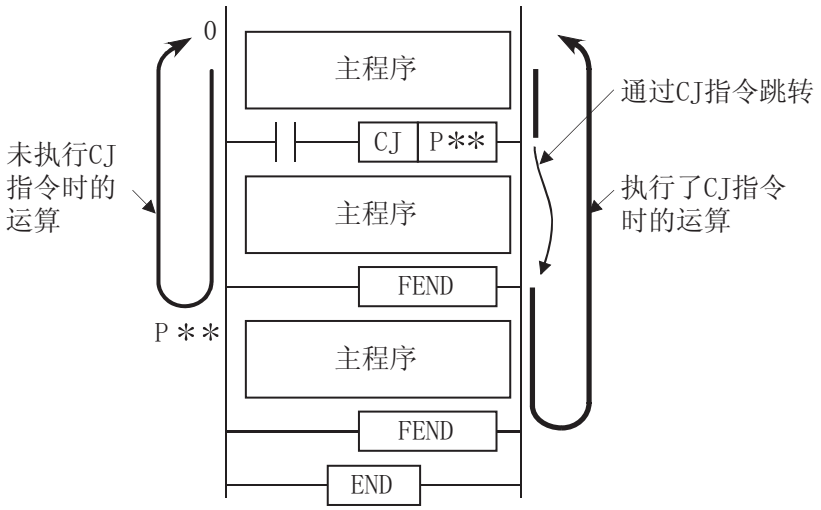
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
--	--								

功 能

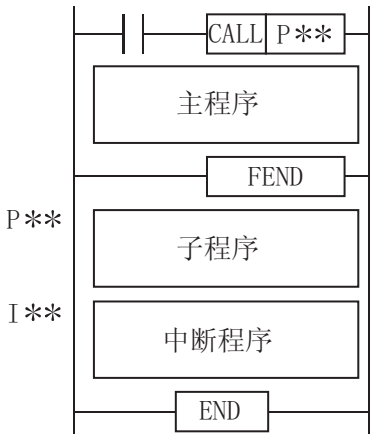
(1) 当通过 CJ 指令等被对顺控程序运算进行分支时，可使用 FEND 指令将主程序从子程序或中断程序中分离出来。

(2) 执行 FEND 指令时，将结束 CPU 模块正在执行的程序。

(3) FEND 指令之后的顺控程序也可通过外围设备进行梯形图显示。
(外围设备持续显示梯形图直至 END 指令为止。)



(a) 使用CJ指令时



(b) 存在有子程序、中断程序时

出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· 在执行 CALL/FCALL/ECALL/EFCALL 指令后，执行 RET 指令之前执行了 FEND 指令时。 (出错代码： 4211)

· 在执行 FOR 指令后，执行 NEXT 指令之前执行了 FEND 指令时。 (出错代码： 4200)

· 在中断程序的执行过程中，在执行 IRET 指令之前执行了 FEND 指令时。 (出错代码： 4221)

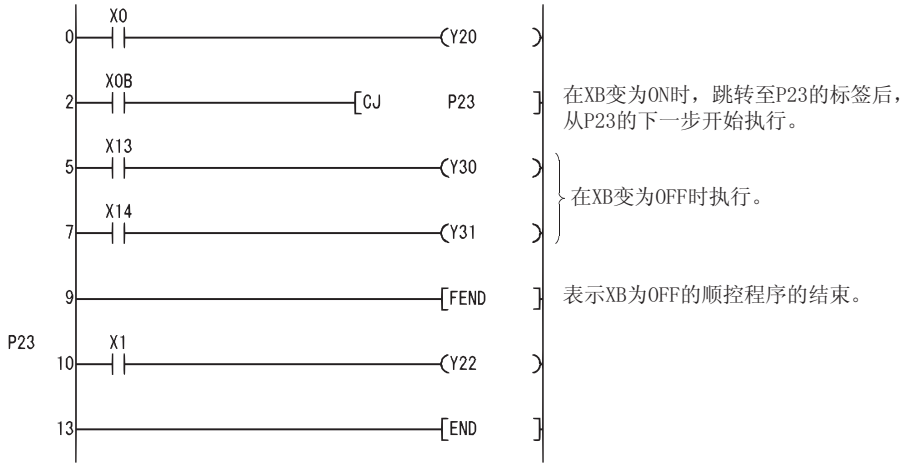
· 在 CHKCIR ～ CHKEND 指令之间执行了 FEND 指令时。 (出错代码： 4230)

· 在 IX ～ IXEND 指令之间执行了 FEND 指令时。 (出错代码： 4231)

程序示例

(1) 下列为使用了 CJ 指令的程序。

[梯形图模式]

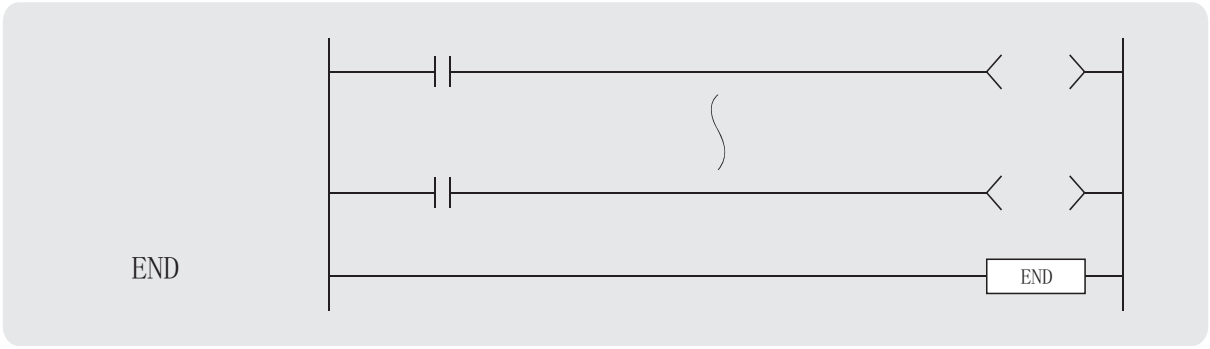


[列表模式]

步	指令	软元件
0	LD	X0
1	OUT	Y20
2	LD	X0B
3	CJ	P23
5	LD	X13
6	OUT	Y30
7	LD	X14
8	OUT	Y31
9	FEND	
10	P23	
11	LD	X1
12	OUT	Y22
13	END	

5.6.2 顺控程序的结束 (END)

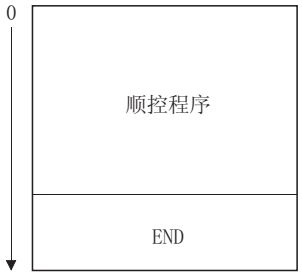
Basic | High performance | Process | Redundant | Universal



设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
--	--								

功 能

(1) 表示包括主程序、子程序和中断程序的程序的结束。
如果执行了 END 指令，CPU 模块正在执行的程序将被结束。



(2) 在主顺控程序的执行过程中不能使用 END 指令。
如果在程序执行过程中需要进行 END 处理，应使用 FEND 指令。

(3) 在外围设备的梯形图模式下进行编程时，无需输入 END 指令。

(4) 在存在有主程序、子程序和中断程序的情况下， END 和 FEND 指令的使用分类如下所示：

出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· 在执行 CALL/FCALL/ECALL/EFCALL 指令后，执行 RET 指令之前执行了 END 指令时。
(出错代码： 4211)

· 在执行 FOR 指令后，执行 NEXT 指令之前执行了 END 指令时。 (出错代码： 4200)

· 在中断程序的执行过程中，在执行 IRET 指令之前执行了 END 指令时。
(出错代码： 4221)

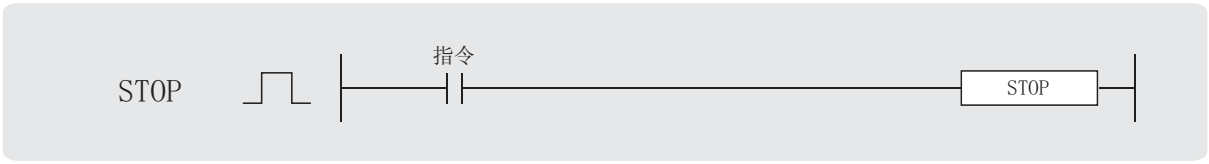
· 在 CHKCIR ～ CHKEND 指令之间执行了 END 指令时。 (出错代码： 4230)

· 在 IX ～ IXEND 指令之间执行了 END 指令时。 (出错代码： 4231)

5.7 其它指令

5.7.1 顺控程序停止 (STOP)

Basic | High performance | Process | Redundant | Universal

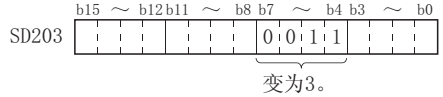


设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
--	--								

功 能

(1) 当执行指令为 ON 时，输出 Y 被复位且 CPU 模块的运算被停止。
(RUN/STOP(键) 开关被置于 STOP 侧时其结果与上相同。)

(2) 执行 STOP 指令时，特殊寄存器 SD203 的 b4 ～ b7 的值将变为 3。



(3) 在执行 STOP 指令之后若要重新开始 CPU 模块的运算，则应对 RUN/STOP(键) 开关进行 RUN → STOP 操作后，再次置于 RUN 位置。

出 错

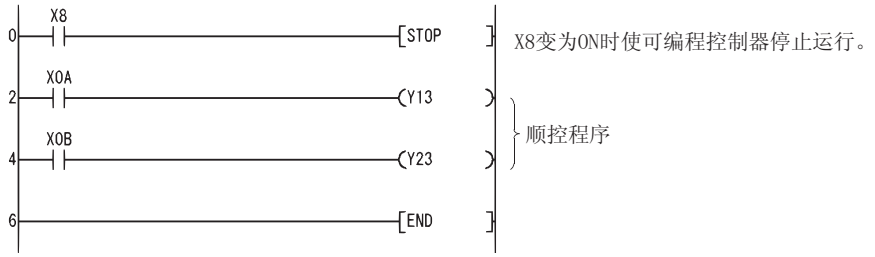
(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 在执行 CALL/FCALL/ECALL/EFCALL 指令后，执行 RET 指令之前执行了 STOP 指令时。 (出错代码： 4211)
- 在执行 FOR 指令后，执行 NEXT 指令之前执行了 STOP 指令时。 (出错代码： 4200)
- 在中断程序的执行过程中，在执行 IRET 指令之前执行了 STOP 指令时。 (出错代码： 4221)
- 在 CHKCIR ～ CHKEND 指令之间执行了 STOP 指令时。 (出错代码： 4230)
- 在 IX ～ IXEND 指令之间执行了 STOP 指令时。 (出错代码： 4231)
- 在恒定周期执行型程序的执行过程中执行了 STOP 指令时。(仅通用型 QCPU) (出错代码： 4223)

程序示例

(1) 以下为 X8 变为 ON 时使 CPU 模块停止运行的程序。

[梯形图模式]

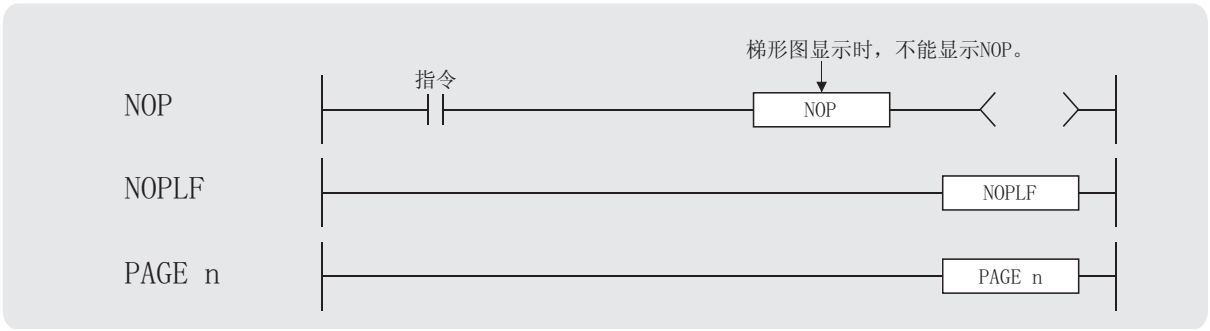


[列表模式]

步	指令	软元件
0	LD	X8
1	STOP	
2	LD	X0A
3	OUT	Y13
4	LD	X0B
5	OUT	Y23
6	END	

5.7.2 无处理 (NOP、NOPLF、PAGE n)

Basic | High performance | Process | Redundant | Universal



设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
--	--								

功 能

NOP

(1) 该指令为无处理指令，对至目前为止的运算不会施加任何影响。

(2) NOP 指令适用于下列情况：

(a) 为顺控程序调试插入空间。

(b) 在不改变步数的状况下删除指令。（用 NOP 替换指令）

(c) 临时性删除指令。

NOPLF

(1) 该指令为无处理指令，对至目前为止的运算不会施加任何影响。

(2) 在通过外围设备进行打印时，可以使用 NOPLF 指令在任意位置进行换页。

(a) 打印梯形图时

· 在梯形图块之间如果有 NOPLF 指令，则将进行换页打印。

· 在梯形图块中如果存在有 NOPLF 指令，则梯形图将无法正常显示。
不要在梯形图快中插入 NOPLF 指令。

(b) 打印指令列表时

· 在打印 NOPLF 指令之后将换页打印。

(3) 关于通过外围设备进行打印输出的详细内容，请参阅所使用的外围设备的操作手册。

PAGE n

(1) 该指令为无处理指令，对至目前为止的运算不会施加任何影响。

(2) 是在外围设备中也执行无处理的指令。

出　错

(1) 在 NOP、NOPLF、PAGE 指令中无运算出错。

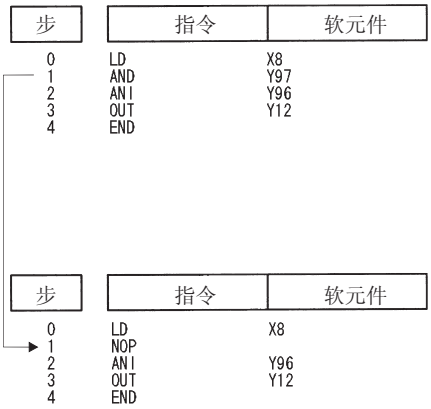
程序示例

NOP

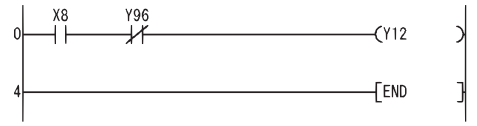
(1) 触点短接 ... 删除 AND 或 AN1 指令。

[梯形图模式]　　[列表模式]

变更前



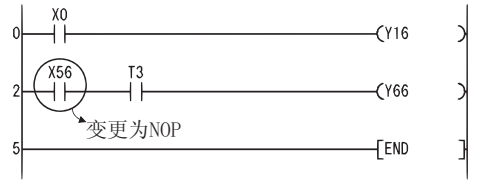
变更后

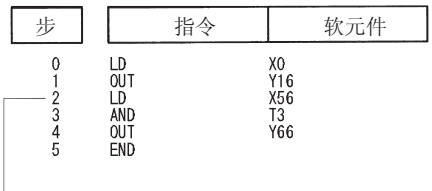


(2) 触点短接 ... 将 LD、LDI 变更为 NOP(请注意将 LD 和 LDI 变更为为 NOP 时，梯形图将被完全改变。)

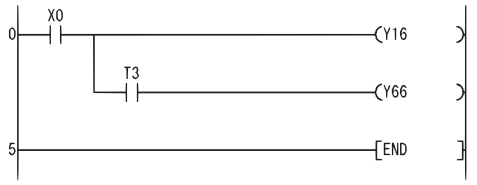
[梯形图模式]　　[列表模式]

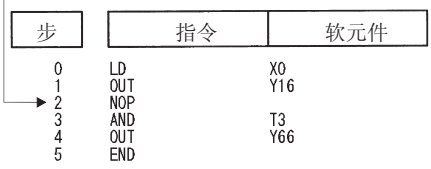
变更前





变更后

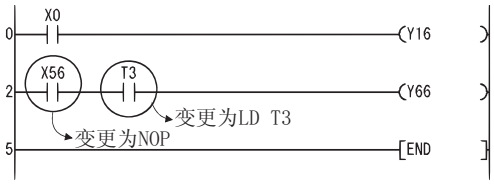


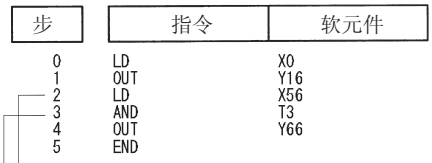


[梯形图模式]

[列表模式]

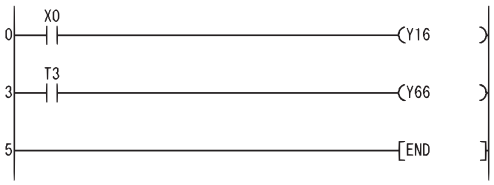
变更前

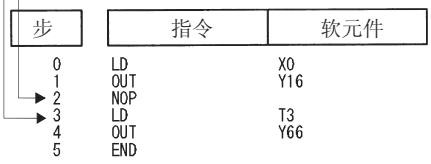




步	指令	软元件
0	LD	X0
1	OUT	Y16
2	LD	X56
3	AND	T3
4	OUT	Y66
5	END	

变更后



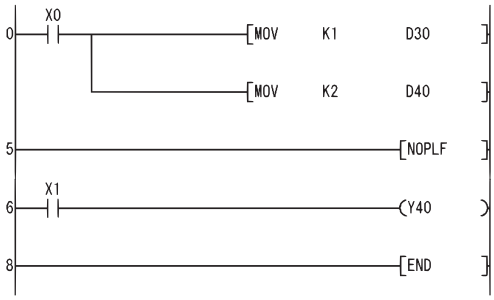


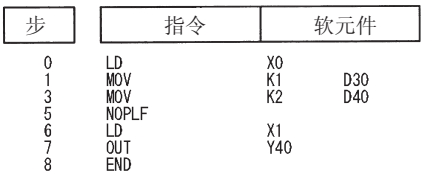
步	指令	软元件
0	LD	X0
1	OUT	Y16
2	NOP	
3	LD	T3
4	OUT	Y66
5	END	

NOPLF

[梯形图模式]

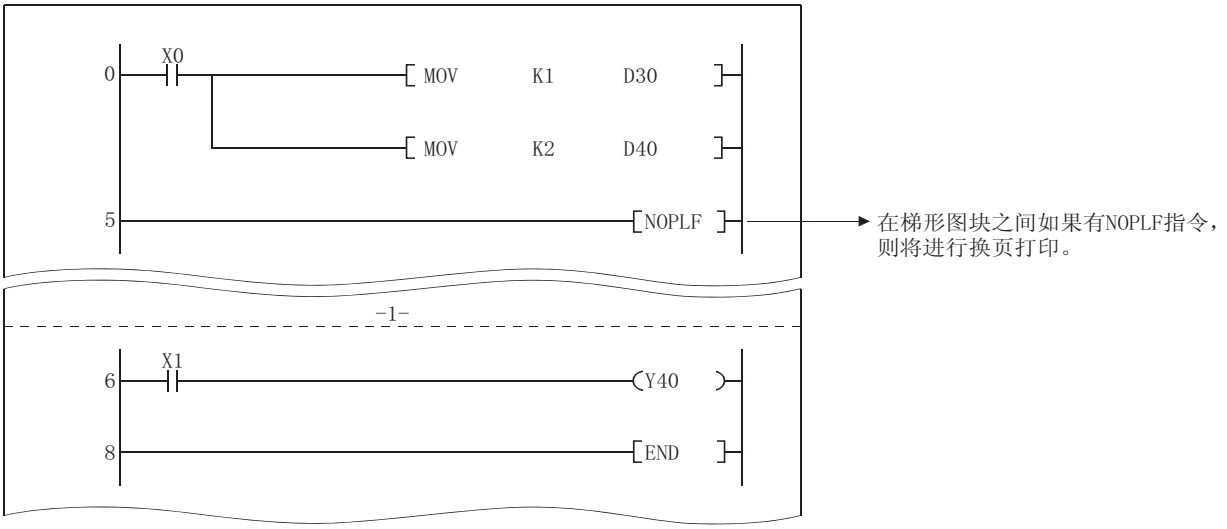
[列表模式]



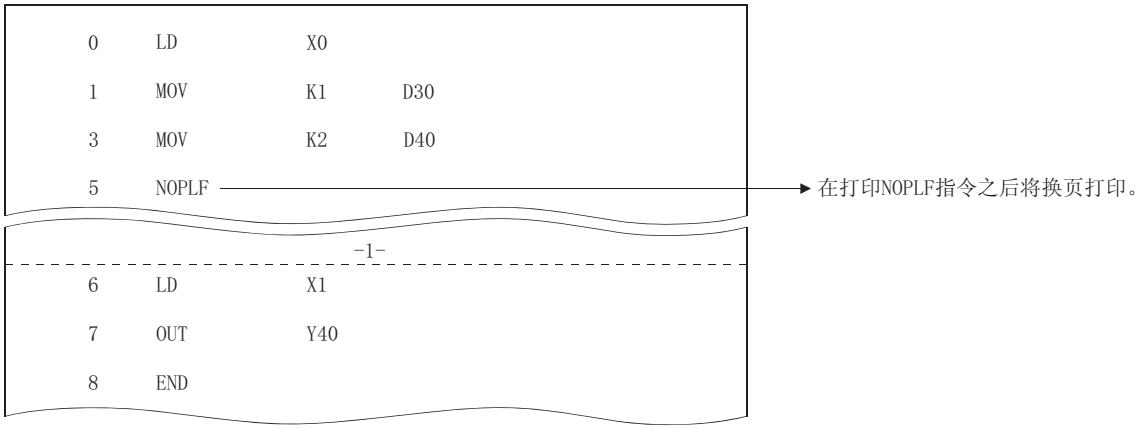


步	指令	软元件
0	LD	X0
1	MOV	K1 D30
3	MOV	K2 D40
5	NOPLF	
6	LD	X1
7	OUT	Y40
8	END	

· 执行梯形图打印时的情况如下所示。

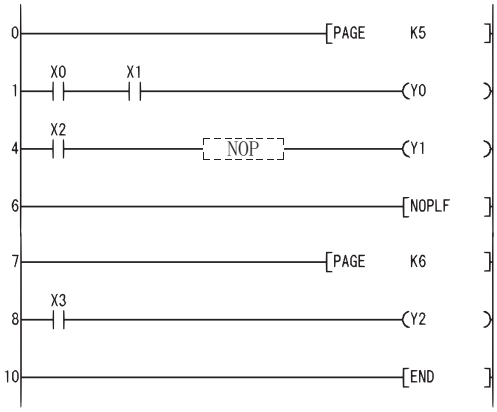


· 执行列表打印时的情况如下所示。



PAGE n

[梯形图模式]



[列表模式]

步	指令	软元件
0	PAGE	K5
1	LD	X0
2	AND	X1
3	OUT	Y0
4	LD	X2
5	NOP	
6	OUT	Y1
7	NOPLF	
8	PAGE	K6
9	LD	X3
10	OUT	Y2
11	END	

备忘录

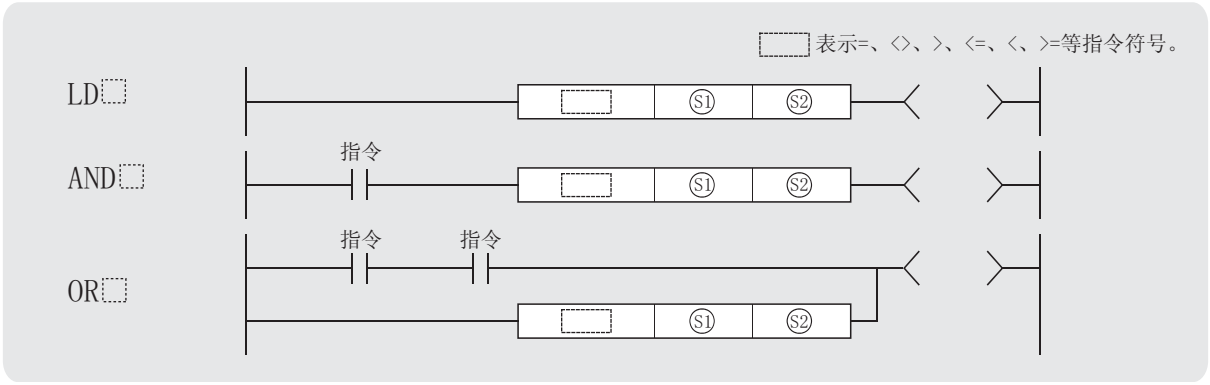
6 基本指令

分类	处理内容	参阅章节
比较运算指令	数据与数据比较。	6.1 节
算术运算指令	数据与的数据加、减、乘、除、递增或递减。	6.2 节
数据转换指令	转换数据类型。	6.3 节
数据传送指令	传送指定的数据。	6.4 节
数据分支指令	程序跳转。	6.5 节
程序执行控制指令	允许 / 禁止程序中断。	6.6 节
I/O 刷新指令	刷新位软元件。	6.7 节
其它方便的指令	上 / 下计数器、教学定时器、特殊功能定时器、旋转台的就近控制等。	6.8 节

6.1 比较运算指令

6.1.1 BIN16 位数据比较 (=、<>、>、<=、<、>=)

Basic | High performance | Process | Redundant | Universal



(S1)、(S2) ：比较数据或者存储比较数据的软元件的起始编号 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
(S1)		○							--
(S2)		○							--

功 能

(1) 将(S1)中指定的软元件的 BIN16 位数据与(S2)中指定的软元件的 BIN16 位数据使用 a 触点进行比较运算。

(2) 各指令的比较运算结果如下所示：

□中的指令符号	条件	比较运算结果	□中的指令符号	条件	比较运算结果
=	(S1)=(S2)	导通状态	=	(S1)≠(S2)	非导通状态
<>	(S1)≠(S2)		<>	(S1)=(S2)	
>	(S1)>(S2)		>	(S1)≤(S2)	
<=	(S1)≤(S2)		<=	(S1)>(S2)	
<	(S1)<(S2)		<	(S1)≥(S2)	
>=	(S1)≥(S2)		>=	(S1)<(S2)	

(3) 在(S1)、(S2)中指定了 16 进制常数的情况下，如果指定了最高位 (b15) 为 1 的数值 (8 ～ F)，则该值将被视为负的 BIN 值进行比较运算。

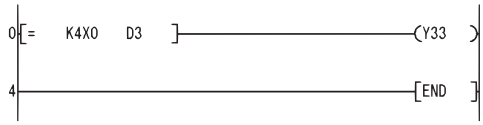
出　错

(1) 在=、<>、>、<=、<、>=指令中无运算出错。

程序示例

(1) 以下为将 X0 ～ XF 的数据与 D3 的数据进行比较，X0 ～ XF 的数据与 D3 的数据一致时，将 Y33 变为 ON 的程序。

[梯形图模式]

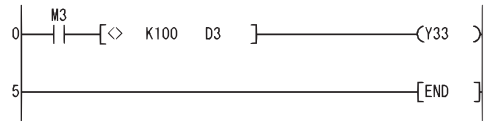


[列表模式]

步	指令	软元件	
0	LD=	K4X0	D3
3	OUT	Y33	
4	END		

(2) 以下为将 BIN 值的 K100 与 D3 的数据进行比较，D3 的数据为除 100 以外时置于导通状态的程序。

[梯形图模式]

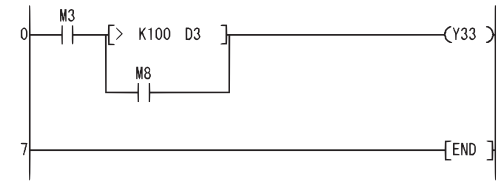


[列表模式]

步	指令	软元件	
0	LD	M3	
1	AND<>	K100	D3
4	OUT	Y33	
5	END		

(3) 以下为将 BIN 值的 100 与 D3 的数据进行比较，D3 的数据小于 100 时置于导通状态的程序。

[梯形图模式]

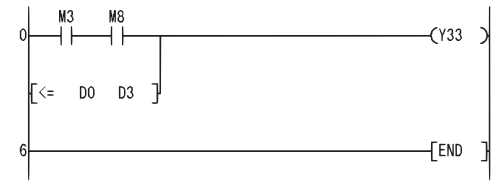


[列表模式]

步	指令	软元件	
0	LD	M3	
1	LD>	K100	D3
4	OR	M8	
5	ANB		
6	OUT	Y33	
7	END		

(4) 以下为将 D0 与 D3 的数据进行比较，(D0 的数据) ≤ (D3 的数据) 时置于导通状态的程序。

[梯形图模式]

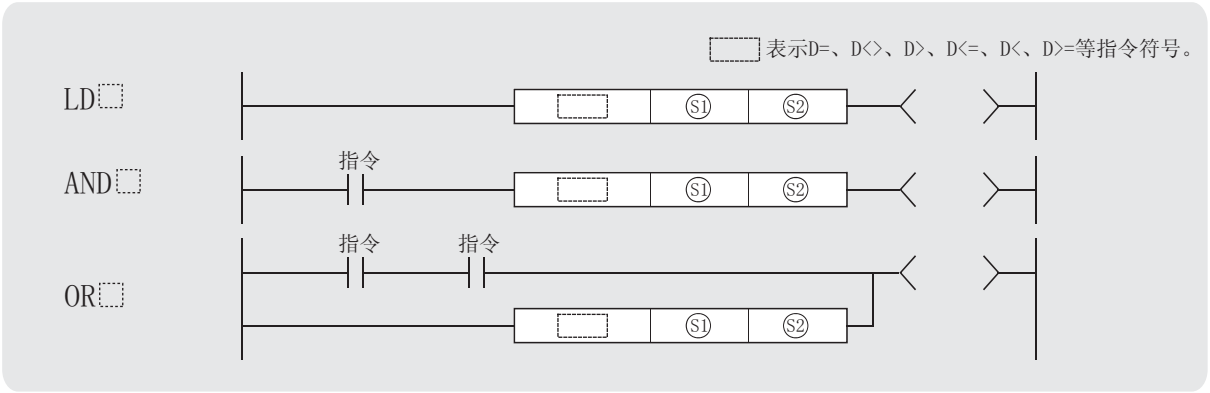


[列表模式]

步	指令	软元件	
0	LD	M3	
1	AND	M8	
2	OR<=	D0	D3
5	OUT	Y33	
6	END		

6.1.2 BIN32 位数据比较 (D=、D<>、D>、D<=、D<、D>=)

Basic | High performance | Process | Redundant | Universal



Ⓢ1、Ⓢ2 ：比较数据或者存储比较数据的软元件的起始编号 (BIN32 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ1	○								--
Ⓢ2	○								--

功 能

(1) 将Ⓢ1 中指定的软元件的 BIN32 位数据与Ⓢ2 中指定的软元件的 BIN32 位数据使用 a 触点进行比较运算。

(2) 各指令的比较运算结果如下所示：

□中的指令符号	条件	比较运算结果	□中的指令符号	条件	比较运算结果
D=	Ⓢ1 = Ⓢ2	导通状态	D=	Ⓢ1 ≠ Ⓢ2	非导通状态
D<>	Ⓢ1 ≠ Ⓢ2		D<>	Ⓢ1 = Ⓢ2	
D>	Ⓢ1 > Ⓢ2		D>	Ⓢ1 ≤ Ⓢ2	
D<=	Ⓢ1 ≤ Ⓢ2		D<=	Ⓢ1 > Ⓢ2	
D<	Ⓢ1 < Ⓢ2		D<	Ⓢ1 ≥ Ⓢ2	
D>=	Ⓢ1 ≥ Ⓢ2		D>=	Ⓢ1 < Ⓢ2	

(3) 在Ⓢ1、Ⓢ2 中指定了 16 进制常数的情况下，如果指定了最高位 (b31) 为 1 的数值 (8 ～ F)，则该值将被视为负的 BIN 值进行比较运算。

(4) 用于比较的数据应通过 32 位指令 (DMOV 指令等) 进行指定。
如果通过 16 位指令 (MOV 指令等) 进行了指定，则大小值的比较将不能正常执行。

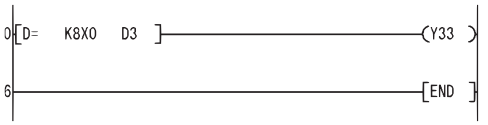
出　错

(1) 在 D=、D<>、D>、D<=、D<、D>=指令中无运算出错。

程序示例

(1) 以下为将 X0 ～ X1F 的数据与 D3、D4 的数据进行比较，X0 ～ X1F 的数据与 D3、D4 的数据一致时，将 Y33 变为 ON 的程序。

[梯形图模式]

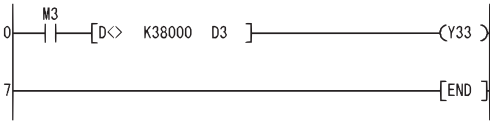


[列表模式]

步	指令	软元件	
0	LDD=	K8X0	D3
5	OUT	Y33	
6	END		

(2) 以下为将 BIN 值的 K38000 与 D3、D4 的数据进行比较，D3、D4 的数据为除 38000 以外时置于导通状态的程序。

[梯形图模式]

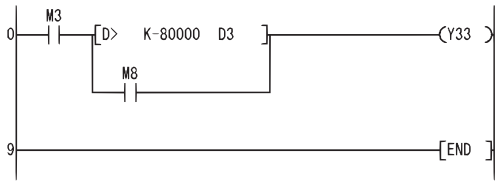


[列表模式]

步	指令	软元件	
0	LD	M3	
1	ANDD<>	K38000	D3
6	OUT	Y33	
7	END		

(3) 以下为将 BIN 值的 K-80000 与 D3、D4 的数据进行比较，D3、D4 的数据小于 -80000 时置于导通状态的程序。

[梯形图模式]

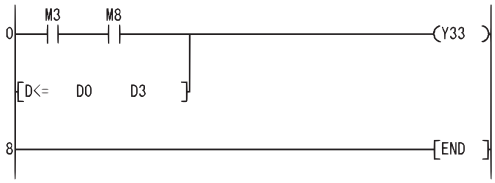


[列表模式]

步	指令	软元件	
0	LD	M3	
1	LDD>	K-80000	D3
6	OR	M8	
7	ANB		
8	OUT	Y33	
9	END		

(4) 以下为将 D0、D1 与 D3、D4 的数据进行比较，(D0、D1 的数据) ≤ (D3、D4 的数据) 时置于导通状态的程序。

[梯形图模式]



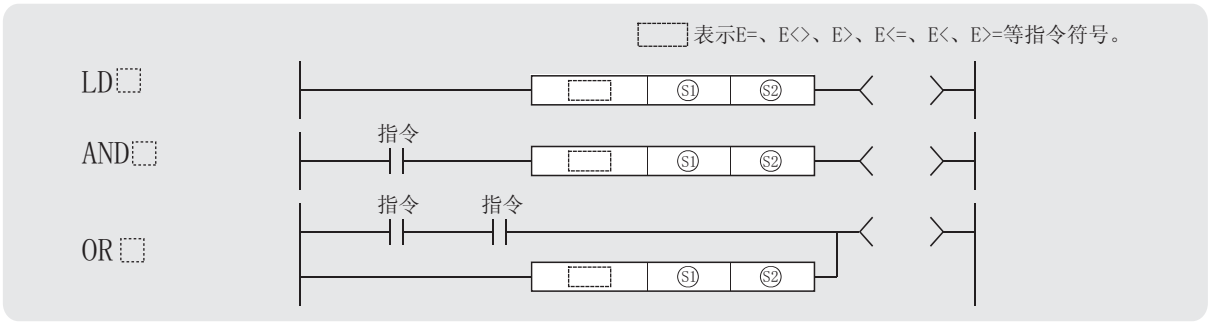
[列表模式]

步	指令	软元件	
0	LD	M3	
1	AND	M8	
2	ORD<=	D0	D3
7	OUT	Y33	
8	END		

6.1.3 浮点数据比较（单精度）(E=、E<>、E>、E<=、E<、E>=)

Ver. Basic | High performance | Process | Redundant | Universal

基本型 QCPU：序列号的前 5 位数为“ 04122 ”以后。



S1、S2 ：比较数据或者存储比较数据的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
S1	--	○		--	○		--	○	--
S2	--	○		--	○		--	○	--

功 能

(1) 将S1中指定的软元件的 32 位浮点数据与S2中指定的软元件的 32 位浮点数据使用 a 触点进行比较运算。

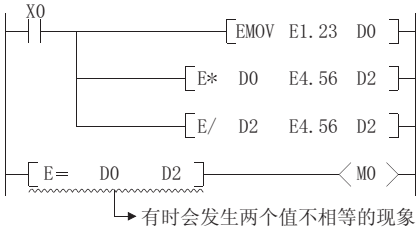
(2) 各指令的比较运算结果如下所示：

□中的指令符号	条件	比较运算结果	□中的指令符号	条件	比较运算结果
E=	S1 = S2	导通状态	E=	S1 ≠ S2	非导通状态
E<>	S1 ≠ S2		E<>	S1 = S2	
E>	S1 > S2		E>	S1 ≤ S2	
E<=	S1 ≤ S2		E<=	S1 > S2	
E<	S1 < S2		E<	S1 ≥ S2	
E>=	S1 ≥ S2		E>=	S1 < S2	

要 点

注意，使用了 E= 指令时，有时会发生由于误差而导致两个值不相等的现象。

例



出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· 指定软元件的内容为 -0 时。*2
(对于基本型 QCPU、高性能型 QCPU、过程 CPU、冗余 CPU) (出错代码 :4100)

*2: 有的 CPU 模块即使指定了 -0 也不会变为运算出错状态。
有关详细内容请参阅 3.2.4 项。

· 指定软元件的内容不在下述范围内时。(仅通用型 QCPU)

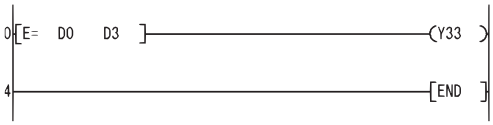
· 0、2^{-126}≦ | 指定软元件的内容 | <2^{128} (出错代码 :4140)

· 指定软元件的内容为 -0、非正规数、非数、±∞时。(仅通用型 QCPU)
(出错代码 :4140)

程序示例

(1) 以下为将 D0、D1 的 32 位浮点实数数据与 D3、D4 的 32 位浮点实数数据进行比较的程序。

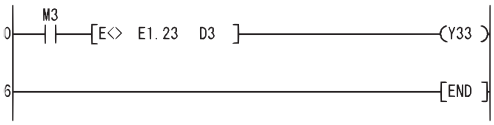
[梯形图模式] [列表模式]



步	指令	软元件	
0	LDE=	D0	D3
3	OUT	Y33	
4	END		

(2) 以下为将浮点实数 1.23 与 D3、D4 的 32 位浮点实数数据进行比较的程序。

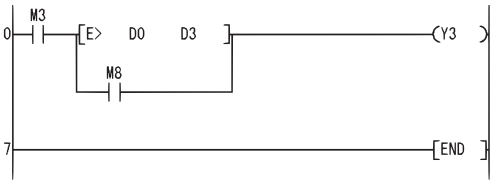
[梯形图模式] [列表模式]



步	指令	软元件	
0	LD	M3	
1	ANDE<>	E1.23	D3
5	OUT	Y33	
6	END		

(3) 以下为将 D0、D1 的 32 位浮点实数数据与 D3、D4 的 32 位浮点实数数据进行比较的程序。

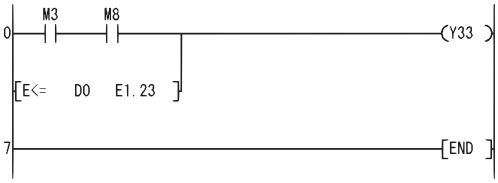
[梯形图模式] [列表模式]



步	指令	软元件	
0	LD	M3	
1	LDE>	D0	D3
4	OR	M8	
5	ANB		
6	OUT	Y3	
7	END		

(4) 以下为将 D0、D1 的 32 位浮点实数数据与浮点实数 1.23 进行比较的程序。

[梯形图模式] [列表模式]

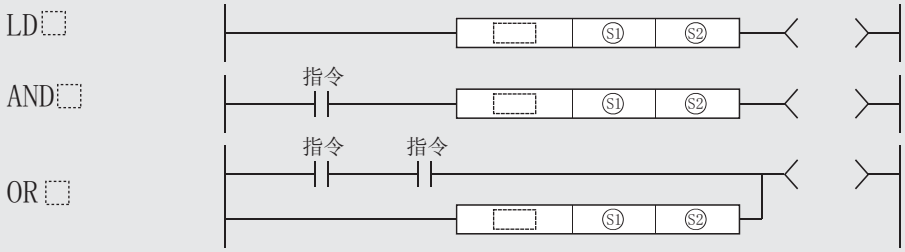


步	指令	软元件	
0	LD	M3	
1	AND	M8	
2	ORE<=	D0	E1.23
6	OUT	Y33	
7	END		

6.1.4 浮点数据比较（双精度）(ED=、ED<>、ED>、ED<=、ED<、ED>=)

Basic ✗ | High performance ✗ | Process ✗ | Redundant ✗ | Universal

□ 表示ED=、ED<>、ED>、ED<=、ED<、ED>=等指令符号。



S1、S2 ：比较数据或者存储比较数据的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数E	其它
	位	字		位	字				
S1	--	○		--			--	○	--
S2	--	○		--			--	○	--

功 能

(1) 将S1中指定的软元件的 64 位浮点数据与S2中指定的软元件的 64 位浮点数据使用 a 触点进行比较运算。

(2) 各指令的比较运算结果如下所示：

□中的指令符号	条件	比较运算结果	□中的指令符号	条件	比较运算结果
ED=	S1 = S2	导通状态	ED=	S1 ≠ S2	非导通状态
ED<>	S1 ≠ S2		ED<>	S1 = S2	
ED>	S1 > S2		ED>	S1 ≤ S2	
ED<=	S1 ≤ S2		ED<=	S1 > S2	
ED<	S1 < S2		ED<	S1 ≥ S2	
ED>=	S1 ≥ S2		ED>=	S1 < S2	

出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· 指定软元件的内容不在下述范围内时。 （出错代码 :4140）

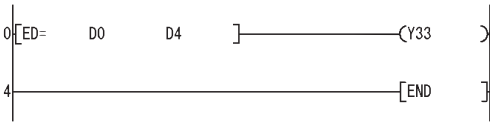
$0、2^{-1022} \leqq |$ 指定软元件的内容 $| < 2^{1024}$

· 指定软元件的内容为 -0 时。 （出错代码 :4140）

程序示例

(1) 以下为将 D0 ～ D3 的 64 位浮点实数数据与 D4 ～ D7 的 64 位浮点实数数据进行比较的程序。

[梯形图模式]

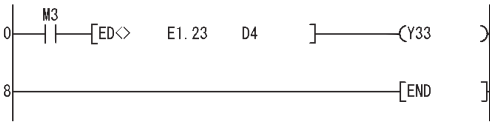


[列表模式]

步	指令	软元件	
0	LDED=	D0	D4
3	OUT	Y33	
4	END		

(2) 以下为将浮点实数 1.23 与 D4 ～ D7 的 64 位浮点实数数据进行比较的程序。

[梯形图模式]

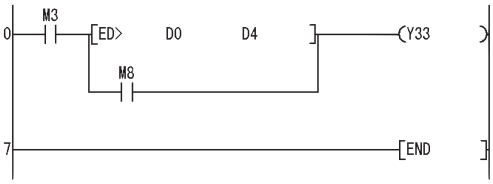


[列表模式]

步	指令	软元件	
0	LD	M3	
1	ANDED<>	E1.23	D4
7	OUT	Y33	
8	END		

(3) 以下为将 D0 ～ D3 的 64 位浮点实数数据与 D4 ～ D7 的 64 位浮点实数数据进行比较的程序。

[梯形图模式]

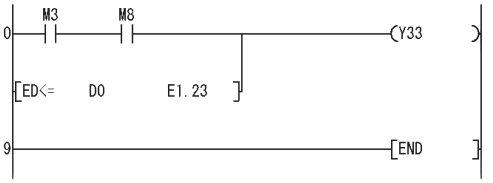


[列表模式]

步	指令	软元件	
0	LD	M3	
1	LDED>	D0	D4
4	OR	M8	
5	ANB		
6	OUT	Y33	
7	END		

(4) 以下为将 D0 ～ D3 的 64 位浮点实数数据与浮点实数 1.23 进行比较的程序。

[梯形图模式]



[列表模式]

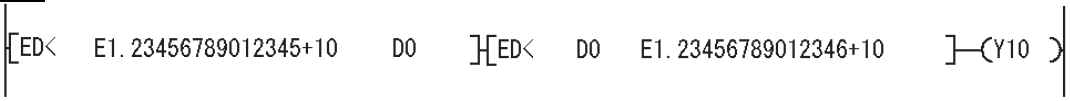
步	指令	软元件	
0	LD	M3	
1	AND	M8	
2	ORED<=	D0	E1.23
8	OUT	Y33	
9	END		

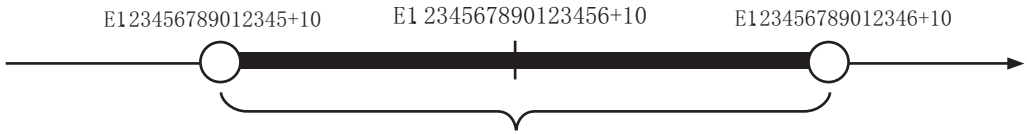
注意事项

(1) 因为可以通过 GX Developer 输入的实数位数最多为 15 位，所以本项中所示的指令不能与有效位数为 16 位以上的实数进行比较。

当用本项中的指令判断与有效位数为 16 位以上的实数的一致 / 不一致时，需要将其与待比较的实数的近似值进行大小比较以判断其是否一致。

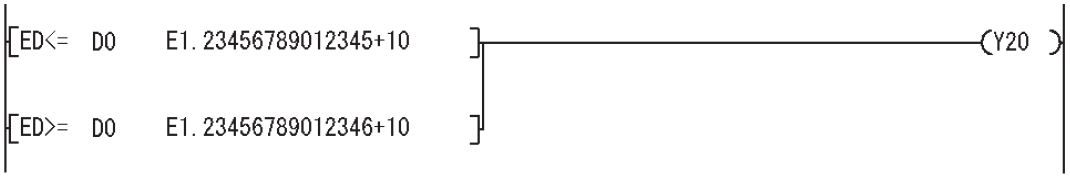
例 当判断 E1.234567890123456+10(有效位数为 16 位) 与双精度浮点数据的一致性时。

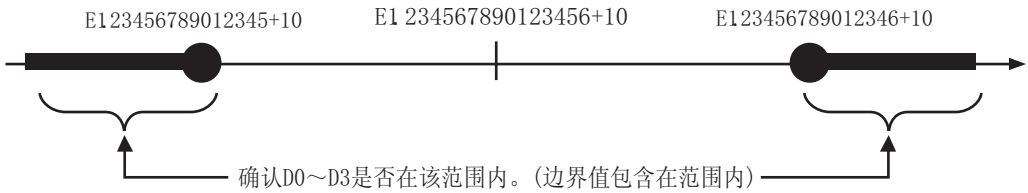




确认D0～D3是否在该范围内。(边界值不包含在范围内)

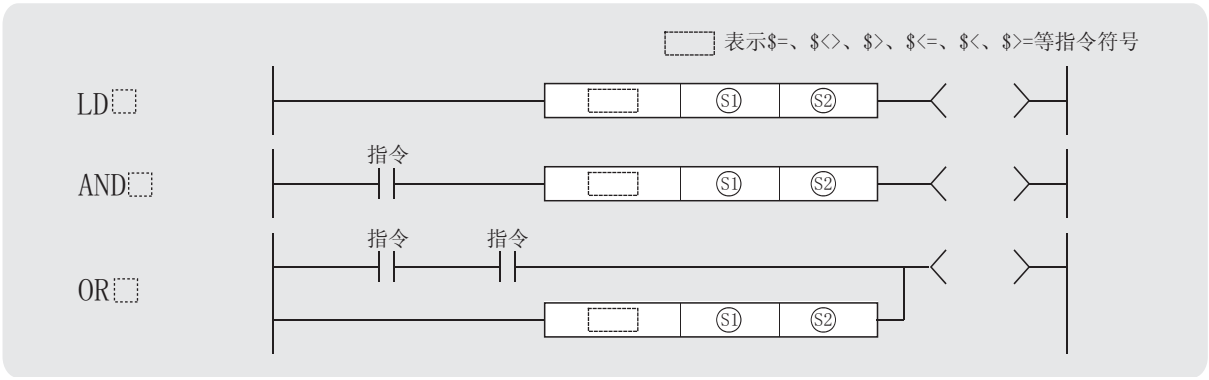
例 当判断 E1.234567890123456+10(有效位数为 16 位) 与双精度浮动小数点数据的不一致性时。





6.1.5 字符串数据比较 ($=、$<>、$>、$<=、$<、$>=)

Basic | High performance | Process | Redundant | Universal



(S1)、(S2) ：比较数据或者存储比较数据的软元件的起始编号（字符串）。

设置数据	内部软元件		R,ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
(S1)	--	○		--				○	--
(S2)	--	○		--				○	--

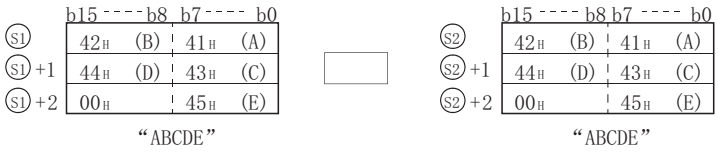
功 能

(1) 将(S1)中指定的字符串数据与(S2)中指定的字符串数据使用 a 触点进行比较运算。

(2) 比较运算将字符串的 ASCII 码从字符串的起始开始逐字进行比较。

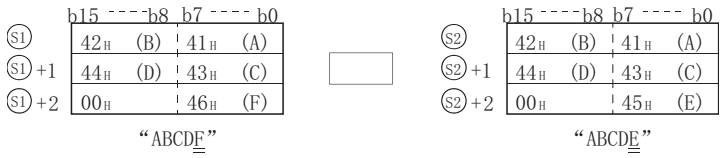
(3) (S1)、(S2)的字符串包含从指定的软元件号开始至存储了“00H”的软元件号为止中的所有字符。

(a) 如果所有字符串都一致，则比较结果一致。



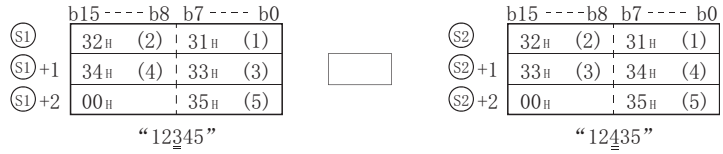
□中的指令符号	比较运算结果	□中的指令符号	比较运算结果
$=	导通状态	$<=	导通状态
$<>	非导通状态	$<	非导通状态
$>	非导通状态	$>=	导通状态

(b) 在字符串不同的情况下，则带较大字符代码的字符串算作大的一方。



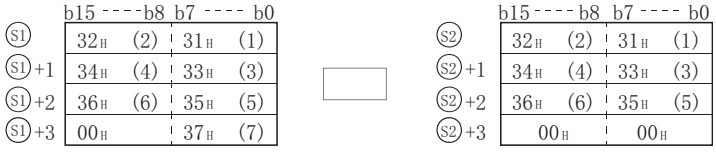
□中的指令符号	比较运算结果	□中的指令符号	比较运算结果
$=	非导通状态	$<=	非导通状态
$<>	导通状态	$<	非导通状态
$>	导通状态	$>=	导通状态

(c) 在字符串不同的情况下，则由第一个不同字符代码的大小决定字符串的大小。



□中的指令符号	比较运算结果	□中的指令符号	比较运算结果
$=	非导通状态	$<=	导通状态
$<>	导通状态	$<	导通状态
$>	非导通状态	$>=	非导通状态

(4) 在(S1)与(S2)中的字符串数据的长度不同的情况下，有较长字符串的数据算作大的一方。



□中的指令符号	比较运算结果	□中的指令符号	比较运算结果
$=	非导通状态	$<=	非导通状态
$<>	导通状态	$<	非导通状态
$>	导通状态	$>=	导通状态

出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

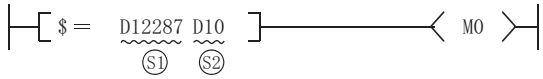
· 在(S1)、(S2)中指定的软元件号的后面，相应软元件的范围中不存在 “ 00H ” 时。
(出错代码 :4101)

· (S1)、(S2)的字符串超过了 16383 个字符时。 (出错代码 :4101)

要 点

在字符串数据比较运算指令中，在进行字符串比较的同时也进行软元件范围检查。因此，即使在相应软元件的范围中不存在 “ 00H ” 时，只要在软元件范围内检测出字符串不一致，则不变为运算出错状态，对比较运算结果进行输出。

例



(S1)的数据

D12287	“B”	“A”
W0	00H	“C”

(S2)的数据

D10	“Z”	“A”
D11	00H	“C”

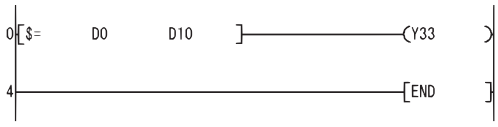
在上述(S1)和(S2)的数据的情况下，由于(S1)的第 2 个字符与(S2)的第 2 个字符不相同，因此(S1)≠(S2)，运算结果为 “ 非导通状态 ”。

此时虽然(S1)的软元件范围中不存在 “ 00H ” 代码，但是由于检测出不一致的软元件为 D12287(在软元件范围内)，因此不变为运算出错状态。

程序示例

(1) 以下为将存储在 D0 后面的字符串与存储在 D10 后面的字符进行比较的程序。

[梯形图模式]

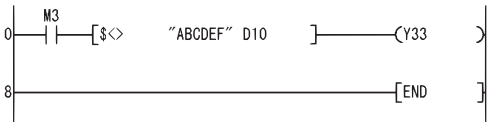


[列表模式]

步	指令	软元件	
0	LD$=	D0	D10
3	OUT	Y33	
4	END		

(2) 以下为将字符串 “ ABCDEF ” 与存储在 D10 后面的字符串进行比较的程序。

[梯形图模式]

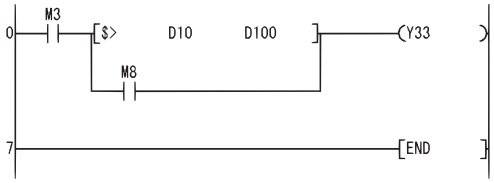


[列表模式]

步	指令	软元件
0	LD	M3
1	AND$<>	"ABCDEF" D10
7	OUT	Y33
8	END	

(3) 以下为将存储在 D10 后面的字符串与存储在 D100 后面的字符串进行比较的程序。

[梯形图模式]

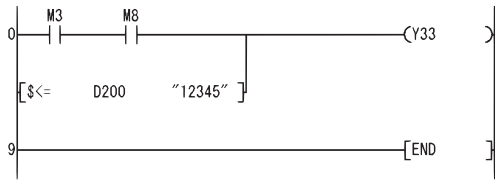


[列表模式]

步	指令	软元件	
0	LD	M3	
1	LD$>	D10	D100
4	OR	M8	
5	ANB		
6	OUT	Y33	
7	END		

(4) 以下为将存储在 D200 后面的字符串与字符串 “ 12345 ” 进行比较的程序。

[梯形图模式]

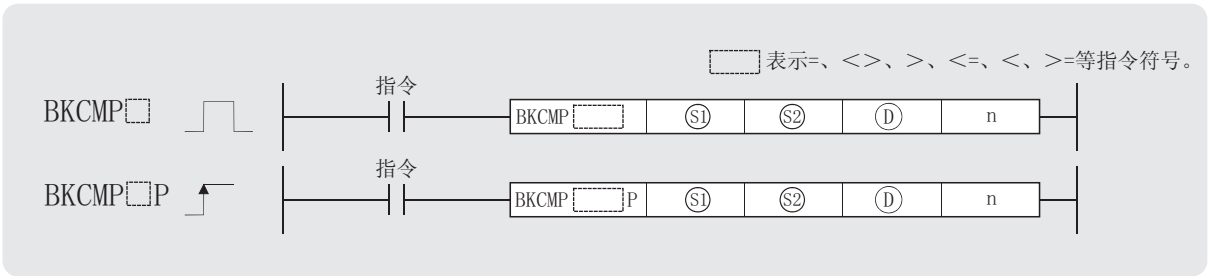


[列表模式]

步	指令	软元件	
0	LD	M3	
1	AND	M8	
2	OR$<=	D200	"12345"
8	OUT	Y33	
9	END		

6.1.6 BIN16 位块数据比较 (BKCMP □、BKCMP □ P)

Basic | High performance | Process | Redundant | Universal



Ⓢ1 ：比较数据或者存储比较数据的软元件的起始编号 (BIN16 位)。

Ⓢ2 ：存储比较数据的软元件的起始编号 (BIN16 位)。

Ⓓ ：存储运算结果的软元件的起始编号 (位)。

n ：比较的数据数 (BIN16 位)。

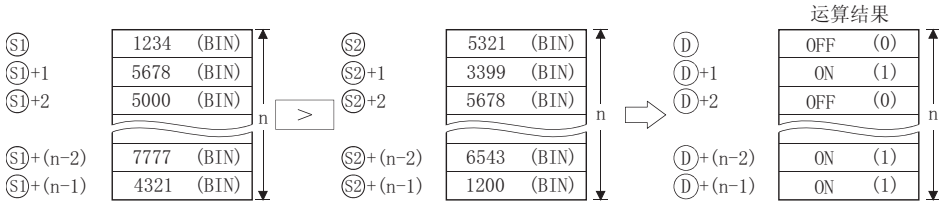
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ1	--	○		--				○	--
Ⓢ2	--	○		--				--	--
Ⓓ	○	○		--				--	--
n	○	○		○				○	--

功 能

(1) 将从Ⓢ1中指定的软元件号开始的 n 点的 BIN16 位数据与Ⓢ2中指定的软元件号开始的 n 点的 BIN16 位数据进行比较，并将运算结果存储到Ⓓ中指定的软元件的后面。

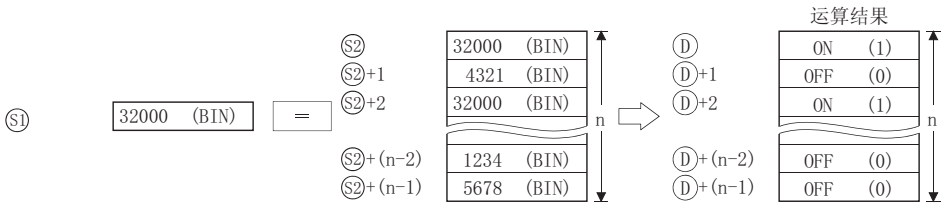
(a) 如果比较条件成立，则Ⓓ的相应软元件将变为 ON。

(b) 如果比较条件不成立，则Ⓓ的相应软元件将变为 OFF。



(2) 比较运算以 16 位为单位进行。

(3) Ⓢ1中可指定的常数范围为 -32768 ～ 32767(BIN16 位)。



(4) 各指令的比较运算结果如下所示：

指令符号	条件	比较运算结果	指令符号	条件	比较运算结果
BKCMP=	ⓢ1 = ⓢ2	ON(1)	BKCMP=	ⓢ1 ≠ ⓢ2	OFF(0)
BKCMP<>	ⓢ1 ≠ ⓢ2		BKCMP<>	ⓢ1 = ⓢ2	
BKCMP>	ⓢ1 > ⓢ2		BKCMP>	ⓢ1 ≤ ⓢ2	
BKCMP<=	ⓢ1 ≤ ⓢ2		BKCMP<=	ⓢ1 > ⓢ2	
BKCMP<	ⓢ1 < ⓢ2		BKCMP<	ⓢ1 ≥ ⓢ2	
BKCMP>=	ⓢ1 ≥ ⓢ2		BKCMP>=	ⓢ1 < ⓢ2	

(5) 如果ⓓ开始的 n 点中存储的比较运算结果全部为 ON(1)，则 SM704(块比较信号) 将变为 ON。

出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· 从ⓢ1、ⓢ2、ⓓ的软元件开始的 n 点的范围超出了相应软元件范围时。
(出错代码 :4101)

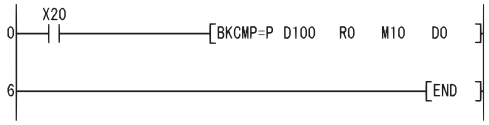
· 从ⓢ1开始至第 n 点为止的软元件范围与从ⓓ开始的至第 n 点为止的软元件范围相重复时。
(出错代码 :4101)

· 从ⓢ2开始至第 n 点为止的软元件范围与从ⓓ开始的至第 n 点为止的软元件范围相重复时。
(出错代码 :4101)

程序示例

(1) 以下为 X20 变为 ON 时，将 D100 ～ D103 中存储的数据值与 R0 ～ R3 中存储的数据值进行比较运算，并将其结果存储到 M10 后面的程序。

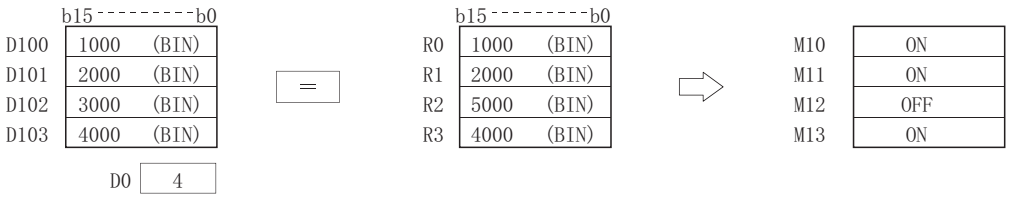
[梯形图模式]



[列表模式]

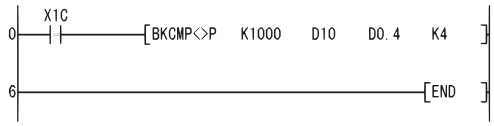
步	指令	软元件			
0	LD	X20			
1	BKCMP=P	D100	R0	M10	D0
6	END				

[动作]



(2) 以下为 X1C 变为 ON 时，将常数 K1000 与 D10 ～ D13 中存储的数据值进行比较运算，并将其结果存储到 D0 的 b4 ～ b7 中的程序。

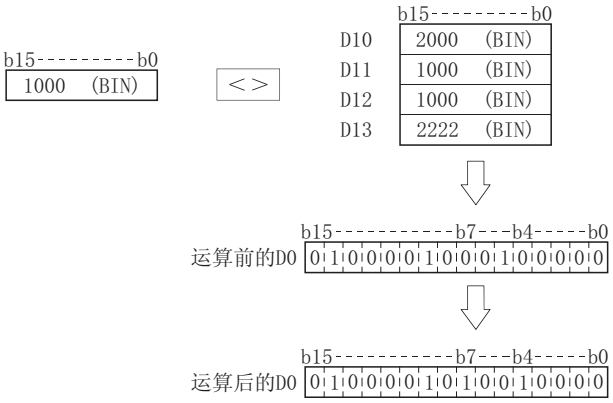
[梯形图模式]



[列表模式]

步	指令	软元件
0	LD	X1C
1	BKCMP<>	K1000 D10 D0.4 K4
6	END	

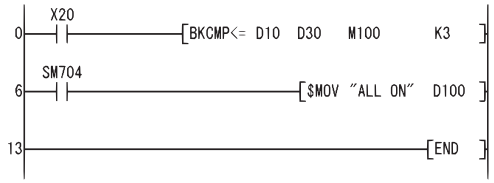
[动作]



(3) 以下为 X20 变为 ON 时，将 D10 ～ D12 中存储的数据与 D30 ～ D32 中存储的数据进行比较，并将其结果存储到 M100 后面的程序。

M100 后面的软元件全部变为 1(ON) 时，将字符串 “ ALL ON ” 传送到 D100 后面。

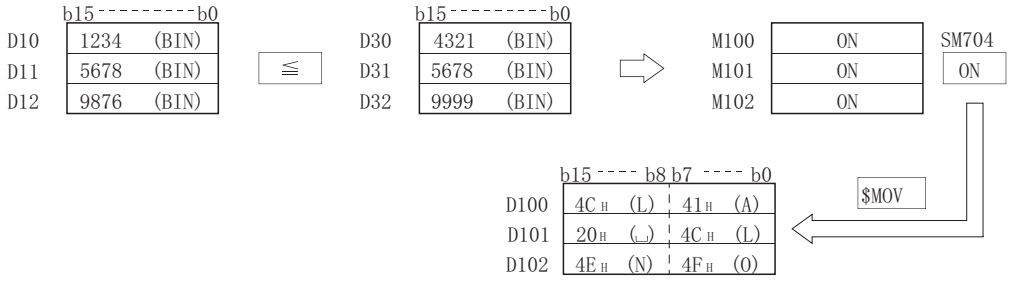
[梯形图模式]



[列表模式]

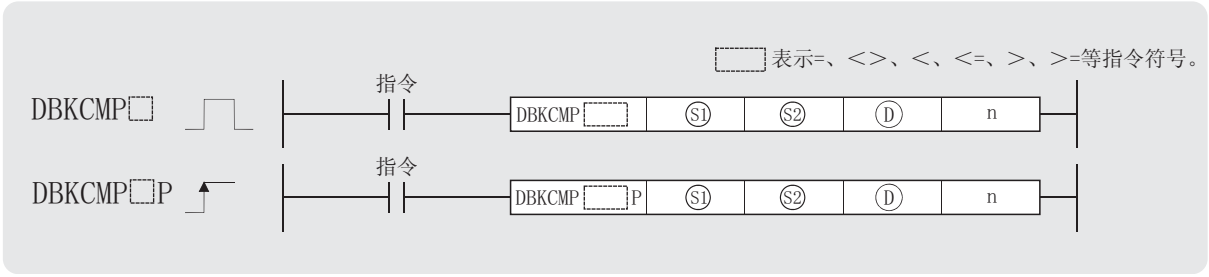
步	指令	软元件
0	LD	X20
1	BKCMP<=	D10 D30 M100 K3
6	LD	SM704
7	$MOV	"ALL ON" D100
13	END	

[动作]



6.1.7 BIN32 位块数据比较 (DBKCMP □、DBKCMP □ P)

· QnU(D)(H)CPU：序列号的前 5 位数为 “ 10102 ” 以后
· QnUDE(H)CPU：序列号的前 5 位数为 “ 10102 ” 以后



Ⓢ1 ：比较数据或者存储比较数据的软元件的起始编号 (BIN32 位)。
Ⓢ2 ：存储比较数据的软元件的起始编号 (BIN32 位)。
Ⓓ ：存储运算结果的软元件的起始编号 (位)。
n ：比较的数据数 (BIN16 位)。

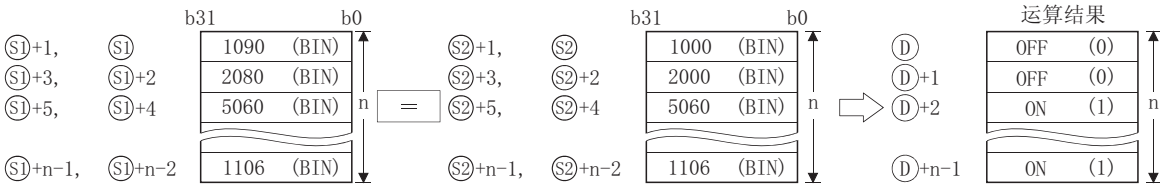
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ1	--	○	○	--				○	--
Ⓢ2	--	○	○	--				--	--
Ⓓ	○	--	○	--				--	--
n	--	○	○	○				○	--

功 能

(1) 将从Ⓢ1中指定的软元件号开始的 n 点的 BIN32 位数据或常数与Ⓢ2中指定的软元件号开始的 n 点的 BIN32 位数据进行比较，并将运算结果存储到Ⓓ中指定的软元件的后面。

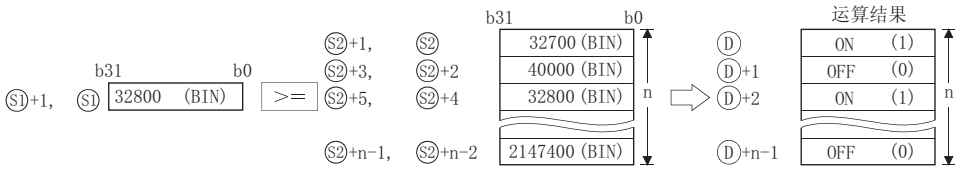
(a) 如果比较条件成立，则Ⓓ的相应软元件将变为 ON。

(b) 如果比较条件不成立，则Ⓓ的相应软元件将变为 OFF。



(2) 比较运算以 32 位为单位进行。

(3) ⓢ1 中可指定的常数范围为 -2147483648 ～ 2147483647（BIN32 位)。



(4) ⓓ 的指定应在从ⓢ1 开始的 n 点的软元件范围及从ⓢ2 开始的 n 点的软元件范围以外。

(5) 各指令的比较运算结果如下所示：

指令符号	条件	比较运算结果	指令符号	条件	比较运算结果
DBKCMP=	ⓢ1=ⓢ2	ON(1)	DBKCMP=	ⓢ1≠ⓢ2	OFF(0)
DBKCMP<>	ⓢ1≠ⓢ2		DBKCMP<>	ⓢ1=ⓢ2	
DBKCMP>	ⓢ1>ⓢ2		DBKCMP>	ⓢ1≤ⓢ2	
DBKCMP<=	ⓢ1≤ⓢ2		DBKCMP<=	ⓢ1>ⓢ2	
DBKCMP<	ⓢ1<ⓢ2		DBKCMP<	ⓢ1≥ⓢ2	
DBKCMP>=	ⓢ1≥ⓢ2		DBKCMP>=	ⓢ1<ⓢ2	

(6) 根据从ⓓ 开始的 n 点中存储的比较运算结果全部为 ON(1) 时，或有一个为 OFF(0) 时的条件，特殊继电器的 ON/OFF 情况如下所示。

序号	编号	比较运算结果全部为 ON(1) 时			比较运算结果中包含有 OFF(0) 时		
		初始化执行 / 扫描	中断 (I45 除外)/ 恒定周期执行	中断 (I45)	初始化执行 / 扫描	中断 (I45 除外)/ 恒定周期执行	中断 (I45)
1	SM704	ON	ON	ON	OFF	OFF	OFF
2	SM716	ON	--	--	OFF	--	--
3	SM717	--	ON	--	--	OFF	--
4	SM718	--	--	ON	--	--	OFF

在待机程序的情况下，为基于调用源程序的特殊继电器的 ON/OFF 情况。

(7) n 中指定的值为 0 时将执行无处理。

出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· n 中指定了负值时。 (出错代码 :4100)

· 从ⓢ1、ⓢ2、ⓓ 中指定的软元件开始至第 n 点为止的范围超出了指定软元件的范围时。 (出错代码 :4101)

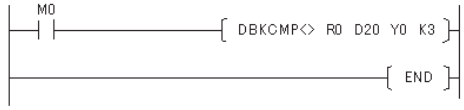
· 从ⓢ1 开始至第 n 点为止的软元件范围与从ⓓ 开始的至第 n 点为止的软元件范围相重复时。 (出错代码 :4101)

· 从ⓢ2 开始至第 n 点为止的软元件范围与从ⓓ 开始的至第 n 点为止的软元件范围相重复时。 (出错代码 :4101)

程序示例

(1) 以下为 M0 变为 ON 时，将 R0 ～ R5 中存储的值与 D20 ～ D25 中存储的值进行比较，并将运算结果存储到 Y0 ～ Y2 中的程序。

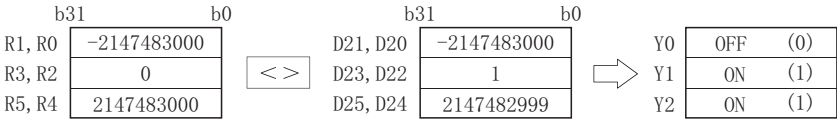
[梯形图模式]



[列表模式]

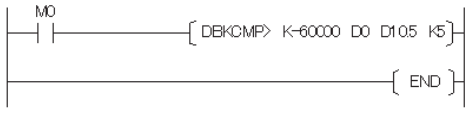
步	指令	软元件
0	LD	M0
1	DBKCMP<>	R0 D20 Y0 K3
6	END	

[动作]



(2) 以下为 M0 变为 ON 时，将常数与 D0 ～ D9 中存储的值进行比较，并将运算结果存储到 D10.5 ～ D10.9 中的程序。

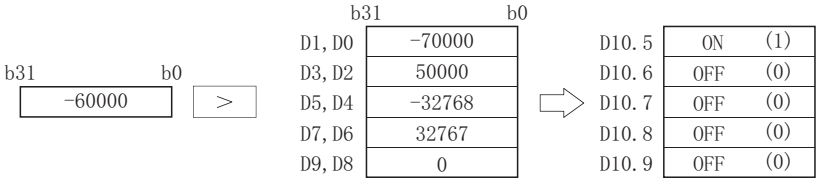
[梯形图模式]



[列表模式]

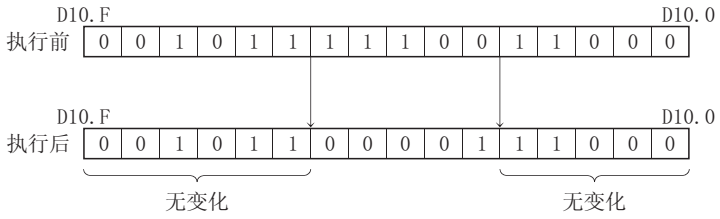
步	指令	软元件
0	LD	M0
1	DBKCMP>	K-60000 D0 D10.5 K5
6	END	

[动作]



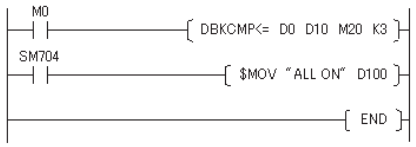
要 点

进行了字软元件的位指定时，除存储运算结果的位指定软元件以外的其它软元件不发生变化。



(3) 以下为 M0 变为 ON 时，将 D0 ～ D5 中存储的值与 D10 ～ D15 中存储的值进行比较，并将运算结果存储到 M20 ～ M22 中的程序，当 M20 ～ M22 的软元件全部变为 ON 时，将字符串 “ ALL ON ” 传送到 D100 的后面。

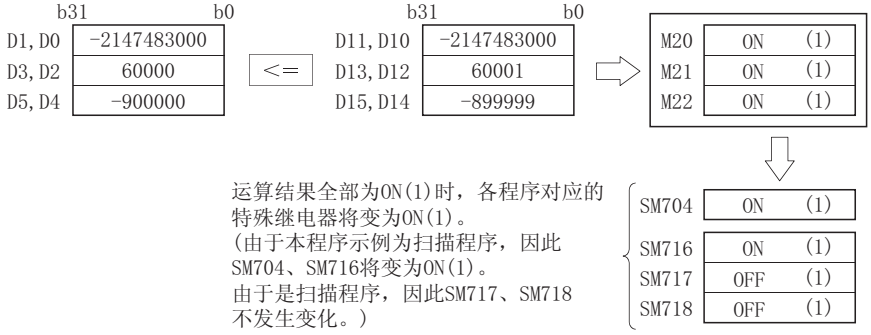
[梯形图模式]



[列表模式]

步	指令	软元件
0	LD	M0
1	DBKCMP<=	D0 D10 M20 K3
6	LD	SM704
7	$MOV	“ALLON” D100
13	END	

[动作]

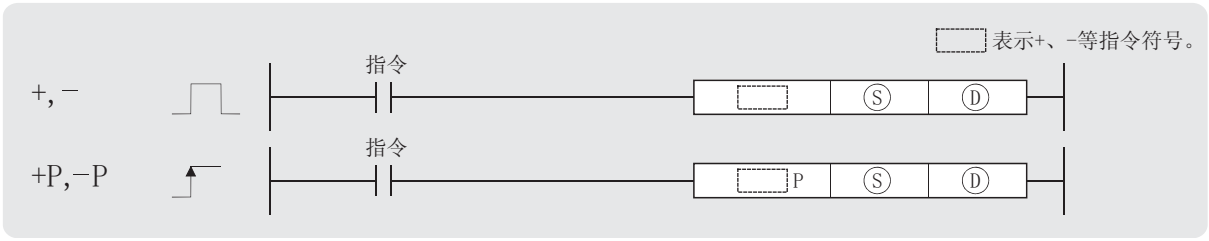


6.2 算术运算指令

6.2.1 BIN16 位加法和减法运算 (+(P)、-(P))

Basic | High performance | Process | Redundant | Universal

1 设置数据为 2 个时 (Ⓓ+Ⓢ→Ⓓ、Ⓓ-Ⓢ→Ⓓ)



Ⓢ : 加法减法数据或者存储加法减法数据的软元件的起始编号 (BIN16 位)。

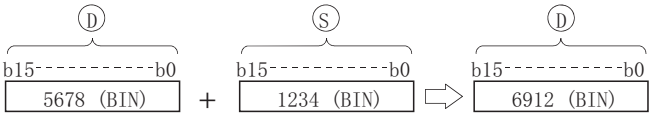
Ⓓ : 存储加法减法数据的软元件的起始编号 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○							○	--
Ⓓ	○							--	--

功 能

+

(1) 将Ⓓ中指定的 BIN16 位数据与Ⓢ中指定的 BIN16 位数据进行加法运算，并将运算结果存储到Ⓓ中指定的软元件中。



(2) Ⓢ、Ⓓ中可指定的范围为 -32768 ～ 32767(BIN16 位)。

(3) 数据的正负判定是在最高位 (b15) 中进行。

· 0... 正

· 1... 负

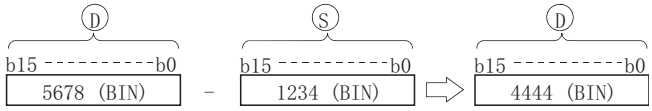
(4) 运算结果中发生了下溢或者上溢时的情况如下所示。
在这种情况下，进位标志不变为 ON。

• K32767 (7FFF$_H$) +K2 (0002$_H$) → K-32767 (8001$_H$) ··· 由于b15为1，因此为负值。

• K-32768 (8000$_H$) +K-2 (FFFE$_H$) → K32766 (7FFE$_H$) ····· 由于b15为0，因此为正值。

-

(1) 将ⓓ中指定的 BIN16 位数据与ⓢ中指定的 BIN16 位数据进行减法运算，并将运算结果存储到ⓓ中指定的软元件中。



(2) ⓢ、ⓓ中可指定的范围为 -32768 ～ 32767(BIN16 位)。

(3) 数据的正负判定是在最高位 (b15) 中进行。

· 0... 正

· 1... 负

(4) 运算结果中发生了下溢或者上溢时的情况如下所示。
在这种情况下，进位标志不变为 ON。

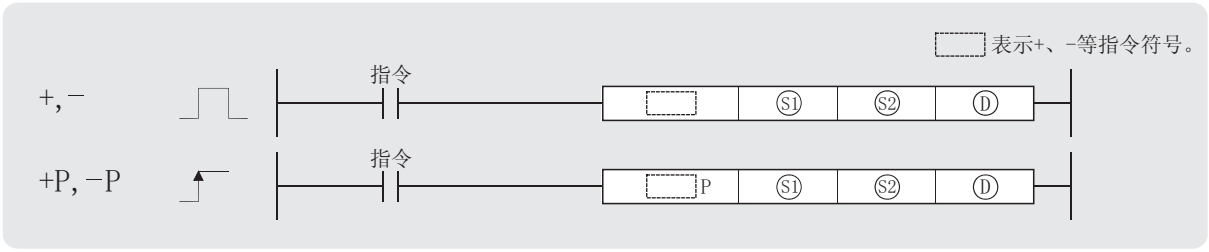
• K-32768- K2 ⟶ K32766·····由于b15为0，因此为正值。
(8000H) (0002H) (7FFEH)

• K32767 -K-2 ⟶ K-32767 ··· 由于b15为1，因此为负值。
(7FFFH) (FFFEH) (8001H)

出 错

(1) 在 +(P)、-(P) 指令中无运算出错。

2 设置数据为 3 个时 (Ⓢ1+Ⓢ2→Ⓓ、Ⓢ1-Ⓢ2→Ⓓ)



Ⓢ1 ： 被加数被减数数据或者存储被加数被减数数据的软元件的起始编号 (BIN16 位)。

Ⓢ2 ： 加数减数数据或者存储加数减数数据的软元件的起始编号 (BIN16 位)。

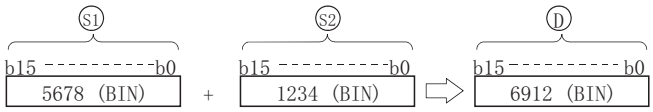
Ⓓ ： 存储运算结果的软元件的起始编号 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ1		○						○	--
Ⓢ2		○						○	--
Ⓓ		○						--	--

功 能

+

(1) 将Ⓢ1中指定的 BIN16 位数据与Ⓢ2中指定的 BIN16 位数据进行加法运算，并将运算结果存储到Ⓓ中指定的软元件中。



(2) Ⓢ1、Ⓢ2、Ⓓ中可指定的范围为 -32768 ～ 32767(BIN16 位)。

(3) 数据的正负判定是在最高位 (b15) 中进行。

· 0... 正

· 1... 负

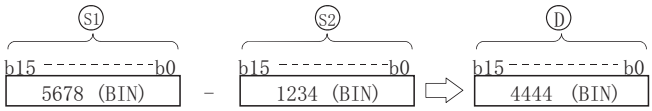
(4) 运算结果中发生了下溢或者上溢时的情况如下所示。
在这种情况下，进位标志不变为 ON。

• K32767 (7FFFH) +K2 (0002H) → K-32767 (8001H) ··· 由于b15为1，因此为负值。

• K-32768 (8000H) +K-2 (FFFEH) → K32766 (7FFEH) ····· 由于b15为0，因此为正值。

-

(1) 将Ⓢ1 中指定的 BIN16 位数据与Ⓢ2 中指定的 BIN16 位数据进行减法运算，并将运算结果存储到Ⓓ中指定的软元件中。



(2) Ⓢ1、Ⓢ2、Ⓓ中可指定的范围为 -32768 ～ 32767(BIN16 位)。

(3) 数据的正负判定是在最高位 (b15) 中进行。

· 0... 正

· 1... 负

(4) 运算结果中发生了下溢或者上溢时的情况如下所示。
在这种情况下，进位标志不变为 ON。

• K－32768－K2 (8000H) (0002H) ⟶ K32766·····由于b15为0，因此为正值。 (7FFEH)

• K32767 －K－2 (7FFFH) (FFFEH) ⟶ K－32767 ···由于b15为1，因此为负值。 (8001H)

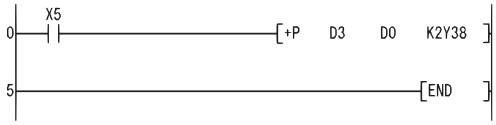
出 错

(1) 在 +(P)、-(P) 指令中无运算出错。

程序示例

(1) 以下为 X5 变为 ON 时，将 D3 与 D0 的内容进行加法运算，并将运算结果存储到 Y38 ～ Y3F 中的程序。

[梯形图模式]

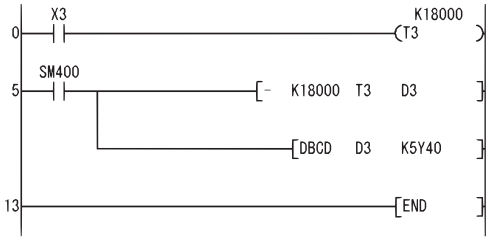


[列表模式]

步	指令	软元件		
0	LD	X5		
1	+P	D3	D0	K2Y38
5	END			

(2) 以下为将定时器 T3 的设置值与当前值的差以 BCD 格式输出到 Y40 ～ Y53 中的程序。

[梯形图模式]



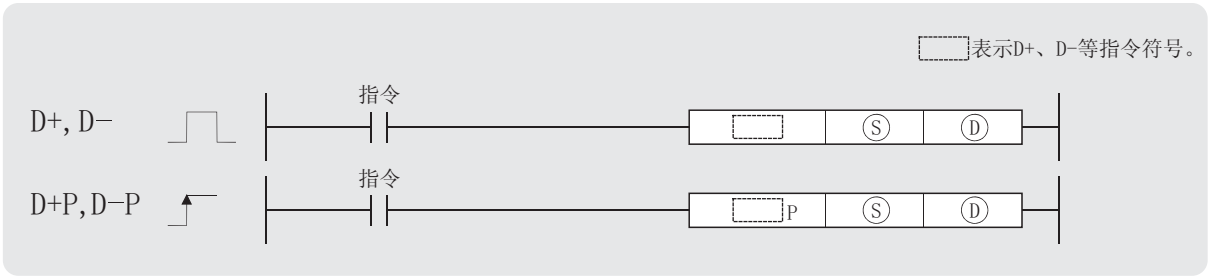
[列表模式]

步	指令	软元件		
0	LD	X3		
1	OUT	T3	K18000	
5	LD	SM400		
6	-	K18000	T3	D3
10	DBCD	D3	K5Y40	
13	END			

6.2.2 BIN 32 位加法和减法运算 (D+(P)、D-(P))

Basic | High performance | Process | Redundant | Universal

1 设置数据为 2 个时 ((Ⓓ+1、Ⓓ)+(Ⓢ+1、Ⓢ) → (Ⓓ+1、Ⓓ)、(Ⓓ+1、Ⓓ)-(Ⓢ+1、Ⓢ) → (Ⓓ+1、Ⓓ))



Ⓢ : 加法减法数据或者存储加法减法数据的软元件的起始编号 (BIN32 位)。

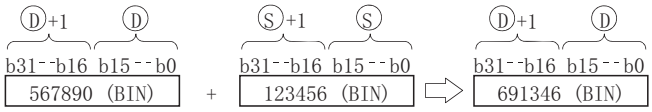
Ⓓ : 存储加法减法数据的软元件的起始编号 (BIN32 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○							○	--
Ⓓ	○							--	--

功 能

D+

(1) 将Ⓓ中指定的 BIN32 位数据与Ⓢ中指定的 BIN32 位数据进行加法运算，并将运算结果存储到Ⓓ中指定的软元件中。



(2) Ⓢ、Ⓓ中可指定的范围为 -2147483648 ～ 2147483647(BIN32 位)。

(3) 数据的正负判定是在最高位 (b31) 中进行。

· 0... 正

· 1... 负

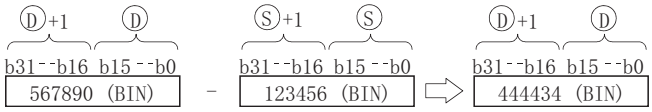
(4) 运算结果中发生了下溢或者上溢时的情况如下所示。
在这种情况下，进位标志不变为 ON。

• K2147483647 (7FFFFFFFH) +K2 (00000002H) → K-2147483647 (80000001H) ··· 由于b31为1，因此为负值。

• K-2147483648 (80000000H) +K-2 (FFFFFFFEH) → K2147483646 (7FFFFFFEH) ····· 由于b31为0，因此为正值。

D-

(1) 将Ⓓ中指定的 BIN32 位数据与Ⓢ中指定的 BIN32 位数据进行减法运算，并将运算结果存储到Ⓓ中指定的软元件中。



(2) Ⓢ、Ⓓ中可指定的范围为 -2147483648 ～ 2147483647(BIN32 位)。

(3) 数据的正负判定是在最高位 (b31) 中进行。

· 0... 正

· 1... 负

(4) 运算结果中发生了下溢或者上溢时的情况如下所示。
在这种情况下，进位标志不变为 ON。

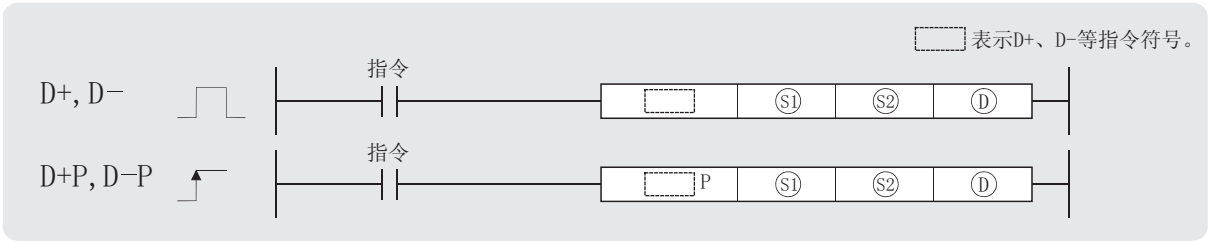
・K-2147483648 (80000000H) - K2 (00000002H) → K2147483646 (7FFFFFFEH) ····· 由于b31为0，因此为正值。

・K2147483647 (7FFFFFFFH) - K-2 (FFFFFFFEH) → K-2147483647 (80000001H) ··· 由于b31为1，因此为负值。

出 错

(1) 在 D+(P)、D-(P) 指令中无运算出错。

2 设置数据为 3 个时 ((Ⓢ1+1、Ⓢ1)+(Ⓢ2+1、Ⓢ2) → (Ⓓ+1、Ⓓ)、
(Ⓢ1+1、Ⓢ1)-(Ⓢ2+1、Ⓢ2) → (Ⓓ+1、Ⓓ))



Ⓢ1 ：被加数被减数数据或者存储被加数被减数数据的软元件的起始编号 (BIN32 位)。

Ⓢ2 ：加数减数数据或者存储加数减数数据的软元件的起始编号 (BIN32 位)。

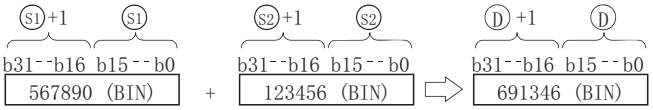
Ⓓ ：存储运算结果的软元件的起始编号 (BIN32 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ1		○						○	--
Ⓢ2		○						○	--
Ⓓ		○						--	--

功 能

D+

(1) 将Ⓢ1中指定的 BIN32 位数据与Ⓢ2中指定的 BIN32 位数据进行加法运算，并将运算结果存储到Ⓓ中指定的软元件中。



(2) Ⓢ1、Ⓢ2、Ⓓ中可指定的范围为 -2147483648 ～ 2147483647(BIN32 位)。

(3) 数据的正负判定是在最高位 (b31) 中进行。

· 0... 正

· 1... 负

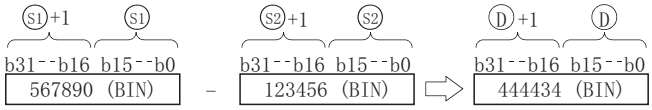
(4) 运算结果中发生了下溢或者上溢时的情况如下所示。
在这种情况下，进位标志不变为 ON。

• K2147483647 (7FFFFFFFH) +K2 (00000002H) → K- 2147483647 ···由于b31为1，因此为负值。 (80000001H)

• K-2147483648 (80000000H) +K-2 (FFFFFFFEH) → K2147483646·····由于b31为0，因此为正值。 (7FFFFFFEH)

D-

(1) 将Ⓢ1中指定的 BIN32 位数据与Ⓢ2中指定的 BIN32 位数据进行减法运算，并将运算结果存储到Ⓓ中指定的软元件中。



(2) Ⓢ1、Ⓢ2、Ⓓ中可指定的范围为 -2147483648 ～ 2147483647(BIN32 位)。

(3) 数据的正负判定是在最高位 (b31) 中进行。

· 0... 正

· 1... 负

(4) 运算结果中发生了下溢或者上溢时的情况如下所示。
在这种情况下，进位标志不变为 ON。

• K-2147483648 (80000000H) -K2 (00000002H) → K2147483646 (7FFFFFFEH) ·····由于b31为0，因此为正值。

• K2147483647 (7FFFFFFFH) -K-2 (FFFFFFFEH) → K-2147483647 (80000001H) ···由于b31为1，因此为负值。

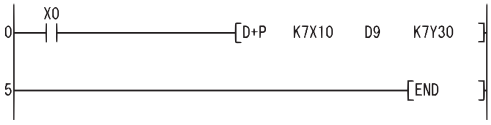
出 错

(1) 在 D+(P)、D-(P) 指令中无运算出错。

程序示例

(1) 以下为 X0 变为 ON 时，将 X10 ～ X2B 的 28 位数据与 D9、D10 的数据进行加法运算，并将运算结果输出到 Y30 ～ Y4B 中的程序。

[梯形图模式]

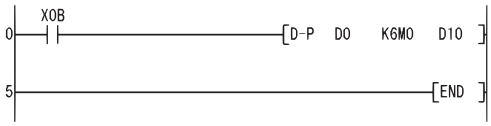


[列表模式]

步	指令	软元件
0	LD	X0
1	D+P	K7X10 D9 K7Y30
5	END	

(2) 以下为 XB 变为 ON 时，将 D0、D1 的数据与 M0 ～ M23 的数据进行减法运算，并将运算结果存储到 D10、D11 中的程序。

[梯形图模式]

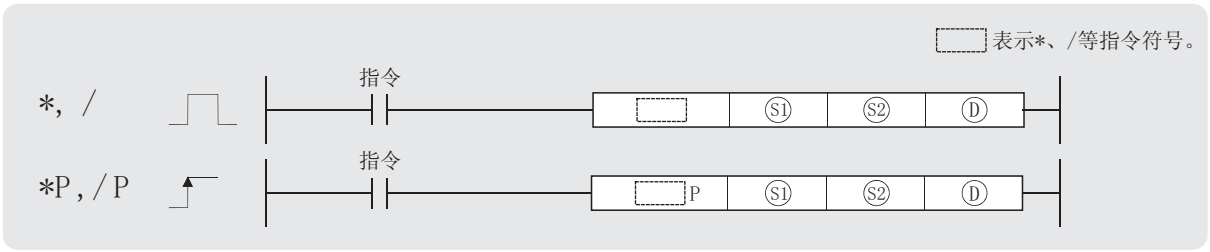


[列表模式]

步	指令	软元件
0	LD	X0B
1	D-P	D0 K6M0 D10
5	END	

6.2.3 BIN 16 位乘法和除法运算 (*(P)、/(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ1 ： 被乘数被除数数据或者存储被乘数被除数数据的软元件的起始编号 (BIN16 位)。

Ⓢ2 ： 乘数除数数据或者存储乘数除数数据的软元件的起始编号 (BIN16 位)。

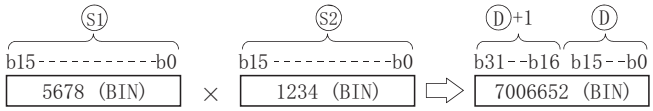
Ⓓ ： 存储运算结果的软元件的起始编号 (BIN32 位)。

设置数据	内部软元件		R ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ1	○							○	--
Ⓢ2	○							○	--
Ⓓ	○							--	--

功 能

*

(1) 将Ⓢ1中指定的 BIN16 位数据与Ⓢ2中指定的 BIN16 位数据进行乘法运算，并将运算结果存储到Ⓓ中指定的软元件中。



(2) Ⓓ为位软元件时则从低位开始指定。

例 K1 .... 低 4 位 (b0 ～ b3)
K4 .... 低 16 位 (b0 ～ b15)
K8 .... 32 位 (b0 ～ b31)

(3) Ⓢ1、Ⓢ2中可指定的范围为 -32768 ～ 32767(BIN16 位)。

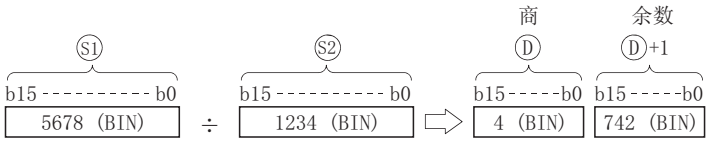
(4) Ⓢ1、Ⓢ2、Ⓓ的数据的正负判定是在最高位 (Ⓢ1、Ⓢ2为 b15、Ⓓ为 b31) 中进行。

· 0... 正

· 1... 负

/

(1) 将Ⓢ1中指定的 BIN16 位数据与Ⓢ2中指定的 BIN16 位数据进行除法运算，并将运算结果存储到Ⓓ中指定的软元件中。



(2) 在字软元件的情况下，则将除法运算结果的商和余数以 32 位进行存储；在位软元件的情况下，则以 16 位格式仅存储商。

商 .... 存储在低 16 位中。
余数 .. 存储在高 16 位中。(只有在字软元件的情况下才可以存储。)

(3) Ⓢ1、Ⓢ2中可指定的范围为 -32768 ～ 32767(BIN16 位)。

(4) Ⓢ1、Ⓢ2、Ⓓ、Ⓓ+1 的数据的正负判定是在最高位 (b15) 中进行。
(商及余数均附加符号。)

· 0... 正

· 1... 负

出 错

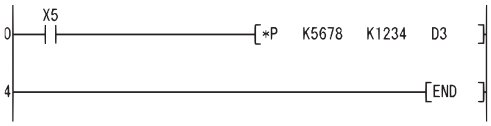
(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· 除数Ⓢ2为 0 时。 (出错代码： 4100)

程序示例

(1) 以下为 X5 变为 ON 时，将 BIN 的 “ 5678 ” 与 “ 1234 ” 的乘法运算结果存储到 D3、D4 中的程序。

[梯形图模式]

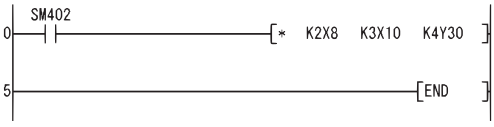


[列表模式]

步	指令	软元件
0	LD	X5
1	*P	K5678 K1234 D3
4	END	

(2) 以下为将 X8 ～ XF 的 BIN 数据与 X10 ～ X1B 的 BIN 数据的乘法结果输出到 Y30 ～ Y3F 中的程序。

[梯形图模式]

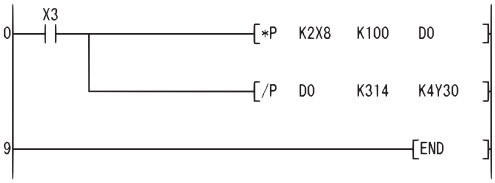


[列表模式]

步	指令	软元件
0	LD	SM402
1	*	K2X8 K3X10 K4Y30
5	END	

(3) 以下为 X3 变为 ON 时，将 X8 ～ XF 的数据用 3.14 相除，并将运算结果输出到 Y30 ～ Y3F 中的程序。

[梯形图模式]

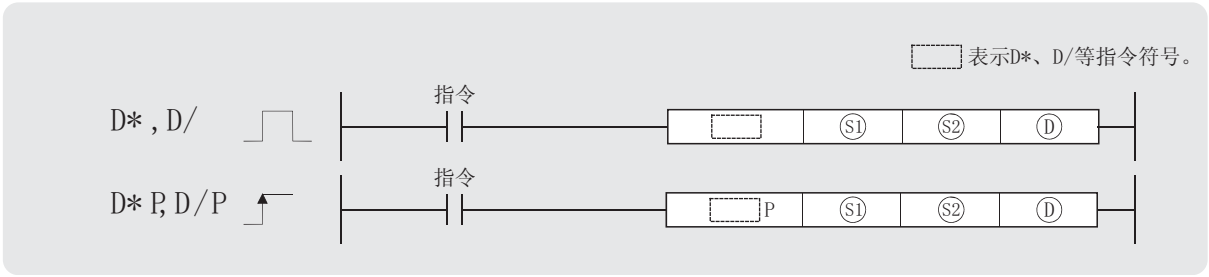


[列表模式]

步	指令	软元件
0	LD	X3
1	*P	K2X8 K100 D0
5	/P	D0 K314 K4Y30
9	END	

6.2.4 BIN32 位乘法和除法运算 (D*(P)、D/(P))

Basic | High performance | Process | Redundant | Universal



(S1) ：被乘数被除数数据或者存储被乘数被除数数据的软元件的起始编号 (BIN32 位)。

(S2) ：乘数除数数据或者存储乘数除数数据的软元件的起始编号 (BIN32 位)。

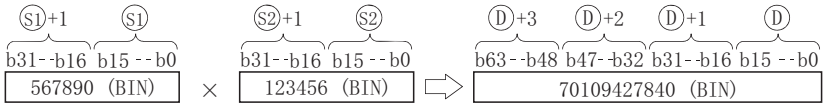
(D) ：存储运算结果的软元件的起始编号 (BIN64 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
(S1)		○			○				--
(S2)		○			○				--
(D)		○			--				--

功 能

D*

(1) 将(S1)中指定的 BIN32 位数据与(S2)中指定的 BIN32 位数据进行乘法运算，并将运算结果存储到(D)中指定的软元件中。



(2) (D)为位软元件时，则乘法结果的低 32 位之前成为对象，不能对高 32 位进行指定。

例 K1 .... 低 4 位 (b0 ～ b3)
K4 .... 低 16 位 (b0 ～ b15)
K8 .... 32 位 (b0 ～ b31)

在位软元件中需要使用乘法运算结果的高 32 位数据时，则应预先将数据暂存在字软元件中，然后将字软元件的 ((D)+2)、((D)+3) 的数据传送到指定的位软元件中。

(3) (S1)、(S2)中可指定的范围为 -2147483648 ～ 2147483647(BIN32 位)。

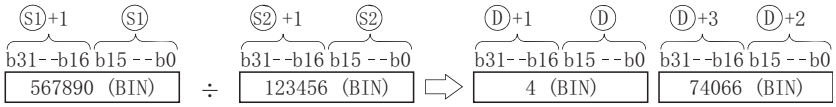
(4) (S1)、(S2)、(D)的数据的正负判定是在最高位 ((S1)、(S2)为 b31、(D)为 b63) 中进行。

· 0... 正

· 1... 负

D/

(1) 将Ⓢ1中指定的 BIN32 位数据与Ⓢ2中指定的 BIN32 位数据进行除法运算，并将运算结果存储到Ⓓ中指定的软元件中。



(2) 在字软元件的情况下，则将除法运算结果的商和余数以 64 位进行存储；在位软元件的情况下，则以 32 位格式仅存储商。

商 .... 存储在低 32 位中。
余数 .. 存储在高 32 位中。(只有在字软元件的情况下才可以存储。)

(3) Ⓢ1、Ⓢ2可指定的范围为 -2147483648 ～ 2147483647(BIN32 位)。

(4) Ⓢ1、Ⓢ2、Ⓓ、Ⓓ+2 的数据的正负判定是在最高位 (b31) 中进行。
(商及余数均附加符号。)
· 0... 正
· 1... 负

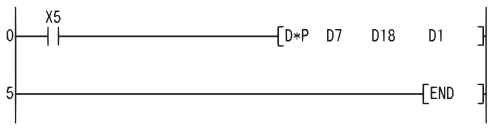
出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。
· 除数Ⓢ2为 0 时。 (出错代码： 4100)

程序示例

(1) 以下为 X5 变为 ON 时，将 D7、D8 的 BIN 数据与 D18、D19 的 BIN 数据的乘法运算结果存储到 D1 ～ D4 中的程序。

[梯形图模式]

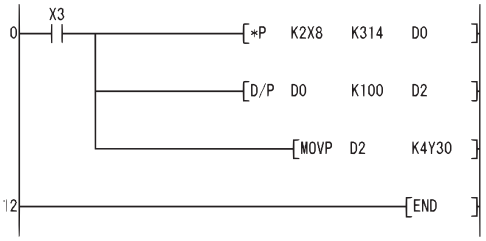


[列表模式]

步	指令	软元件		
0	LD	X5		
1	D*P	D7	D18	D1
5	END			

(2) 以下为 X3 变为 ON 时，将 X8 ～ XF 的数据用 3.14 相除，并将运算结果输出到 Y30 ～ Y3F 中的程序。

[梯形图模式]



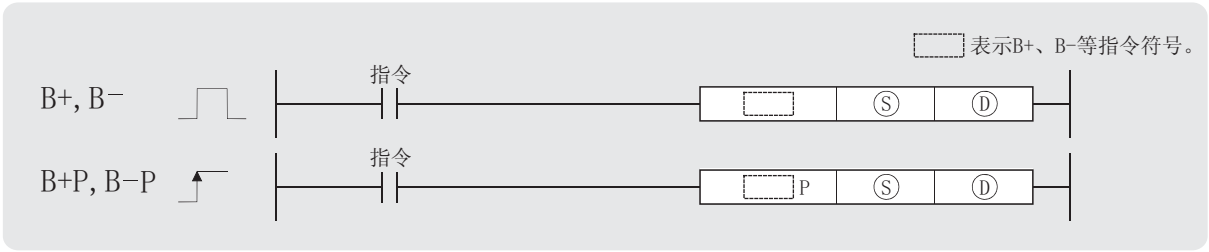
[列表模式]

步	指令	软元件		
0	LD	X3		
1	*P	K2X8	K314	D0
5	D/P	D0	K100	D2
10	MOVP	D2	K4Y30	
12	END			

6.2.5 BCD4 位加法和减法运算 (B+(P)、B-(P))

Basic | High performance | Process | Redundant | Universal

1 设置数据为 2 个时 (Ⓓ+Ⓢ→Ⓓ、Ⓓ-Ⓢ→Ⓓ)



Ⓢ : 加法减法数据或者存储加法减法数据的软元件的起始编号 (BCD4 位)。

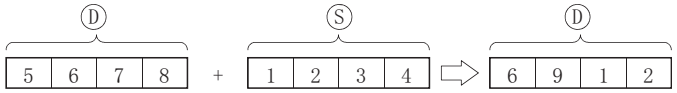
Ⓓ : 存储加法减法数据的软元件的起始编号 (BCD4 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○							○	--
Ⓓ	○							--	--

功 能

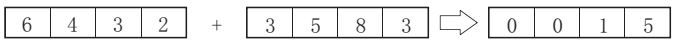
B+

(1) 将Ⓓ中指定的 BCD4 位数据与Ⓢ中指定的 BCD4 位数据进行加法运算，并将加法运算结果存储到Ⓓ中指定的软元件中。



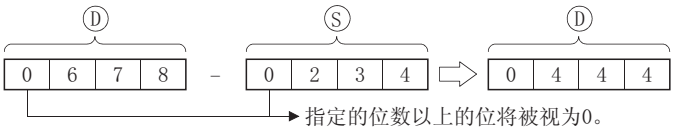
(2) Ⓢ、Ⓓ中可指定的范围为 0 ～ 9999(BCD4 位)。

(3) 加法运算结果超过了 9999 时，进位将被视为无效。
此时，进位标志不变为 ON。



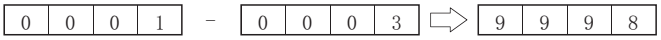
B-

(1) 将Ⓢ中指定的 BCD4 位数据与Ⓓ中指定的 BCD4 位数据进行减法运算，并将减法运算结果存储到Ⓓ中指定的软元件中。



(2) Ⓢ、Ⓓ中可指定的范围为 0 ～ 9999(BCD4 位)。

(3) 减法运算结果中发生了下溢时的情况如下所示。
在这种情况下，进位标志不变为 ON。



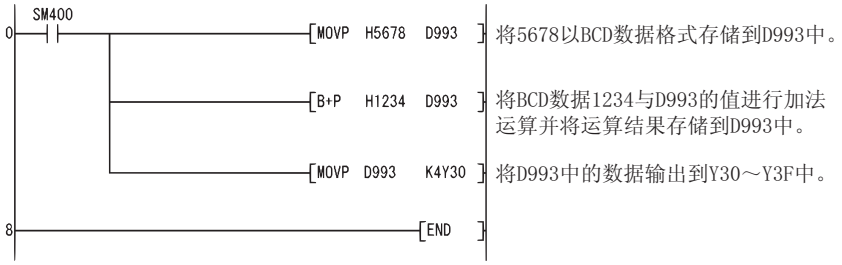
出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。
·Ⓢ、Ⓓ的 BCD 数据超出了 0 ～ 9999 的范围时。 (出错代码 : 4100)

程序示例

(1) 以下为将 5678 及 1234 的 BCD 数据进行加法运算，在将运算结果存储到 D993 中的同时并输出到 Y30 ～ Y3F 中的程序。

[梯形图模式]

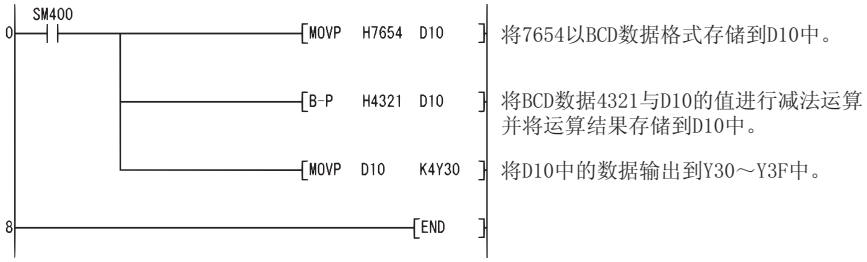


[列表模式]

步	指令	软元件	
0	LD	SM400	
1	MOVP	H5678	D993
3	B+P	H1234	D993
6	MOVP	D993	K4Y30
8	END		

(2) 以下为将 7654 及 4321 的 BCD 数据进行减法运算，在将运算结果存储到 D10 中的同时并输出到 Y30 ～ Y3F 中的程序。

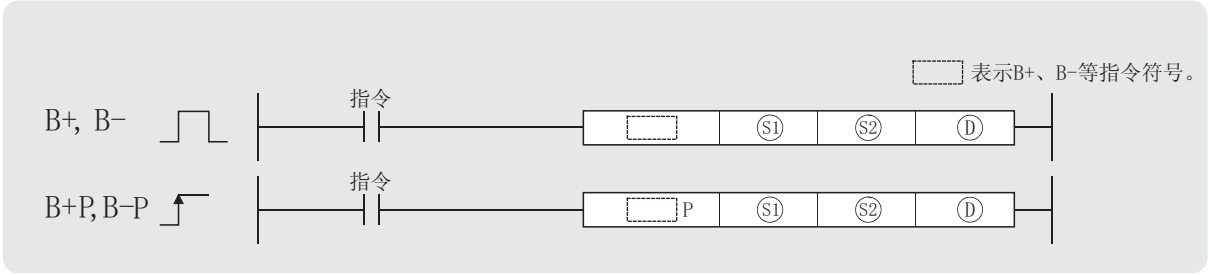
[梯形图模式]



[列表模式]

步	指令	软元件	
0	LD	SM400	
1	MOVP	H7654	D10
3	B-P	H4321	D10
6	MOVP	D10	K4Y30
8	END		

2 设置数据为 3 个时 (Ⓢ1+Ⓢ2→Ⓓ、Ⓢ1-Ⓢ2→Ⓓ)



Ⓢ1 ：被加数被减数数据或者存储被加数被减数数据的软元件的起始编号 (BCD4 位)。

Ⓢ2 ：加数减数数据或者存储加数减数数据的软元件的起始编号 (BCD4 位)。

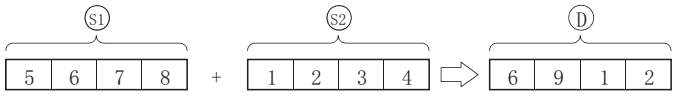
Ⓓ ：存储运算结果的软元件的起始编号 (BCD4 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ1		○						○	--
Ⓢ2		○						○	--
Ⓓ		○						--	--

功 能

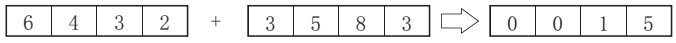
B+

(1) 将Ⓢ1中指定的 BCD4 位数据与Ⓢ2中指定的 BCD4 位数据进行加法运算，并将加法运算结果存储到Ⓓ中指定的软元件中。



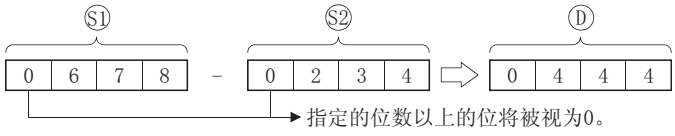
(2) Ⓢ1、Ⓢ2、Ⓓ中可指定的范围为 0 ～ 9999(BCD4 位)。

(3) 加法运算结果超过了 9999 时，进位将被视为无效。
在这种情况下，进位标志不变为 ON。



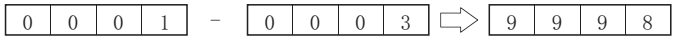
B-

(1) 将Ⓢ1中指定的 BCD4 位数据与Ⓢ2中指定的 BCD4 位数据进行减法运算，并将减法运算结果存储到Ⓓ中指定的软元件中。



(2) Ⓢ1、Ⓢ2、Ⓓ中可指定的范围为 0 ～ 9999(BCD4 位)。

(3) 减法运算结果中发生了下溢时的情况如下所示。
在这种情况下，进位标志不变为 ON。



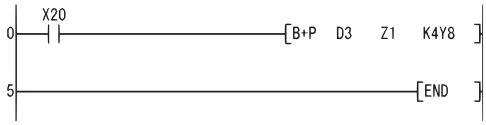
出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。
·Ⓢ1、Ⓢ2、Ⓓ的 BCD 数据超出了 0 ～ 9999 的范围时。 (出错代码： 4100)

程序示例

(1) 以下为 X20 变为 ON 时，将 D3 的 BCD 数据与 Z1 的 BCD 数据进行加法运算，并将运算结果输出到 Y8 ～ Y17 中的程序。

[梯形图模式]

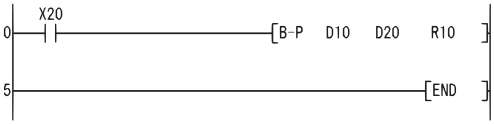


[列表模式]

步	指令	软元件		
0	LD	X20		
1	B+P	D3	Z1	K4Y8
5	END			

(2) 以下为 X20 变为 ON 时，将 D10 的 BCD 数据与 D20 的 BCD 数据进行减法运算，并将运算结果输出到 R10 中的程序。

[梯形图模式]



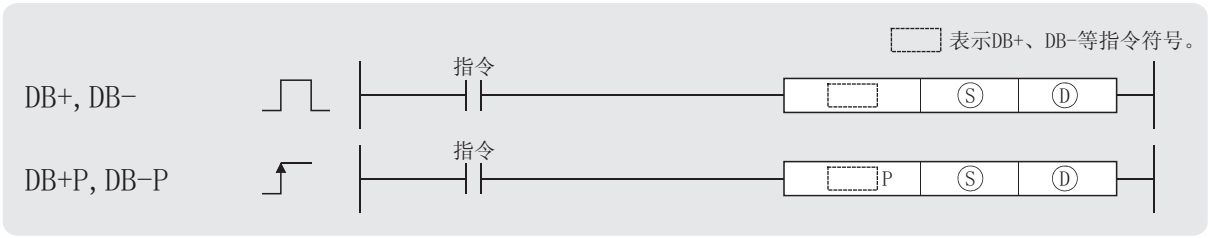
[列表模式]

步	指令	软元件		
0	LD	X20		
1	B-P	D10	D20	R10
5	END			

6.2.6 BCD8 位加法和减法运算 (DB+(P)、DB-(P))

Basic | High performance | Process | Redundant | Universal

1 设置数据为 2 个时 ((Ⓓ+1、Ⓓ)+(Ⓢ+1、Ⓢ) → (Ⓓ+1、Ⓓ)、(Ⓓ+1、Ⓓ)-(Ⓢ+1、Ⓢ) → (Ⓓ+1、Ⓓ))



Ⓢ : 加法减法数据或者存储加法减法数据的软元件的起始编号 (BCD8 位)。

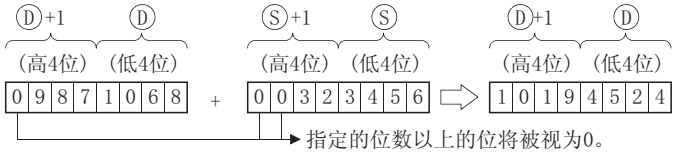
Ⓓ : 存储加法减法数据的软元件的起始编号 (BCD8 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○							○	--
Ⓓ	○							--	--

功 能

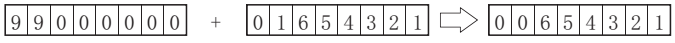
DB+

(1) 将Ⓓ中指定的 BCD8 位数据与Ⓢ中指定的 BCD8 位数据进行加法运算，并将加法运算结果存储到Ⓓ中指定的软元件中。



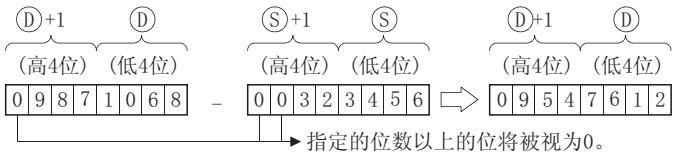
(2) Ⓢ、Ⓓ中可指定的范围为 0 ～ 99999999(BCD8 位)。

(3) 加法运算结果超过了 99999999 时，进位将被视为无效。
在这种情况下，进位标志不变为 ON。



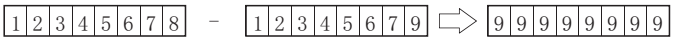
DB-

(1) 将Ⓢ中指定的 BCD8 位数据与Ⓓ中指定的 BCD8 位数据进行减法运算，并将减法运算结果存储到Ⓓ中指定的软元件中。



(2) Ⓢ、Ⓓ中可指定的范围为 0 ～ 99999999(BCD8 位)。

(3) 减法运算结果中发生了下溢时的情况如下所示。
在这种情况下，进位标志不变为 ON。



出 错

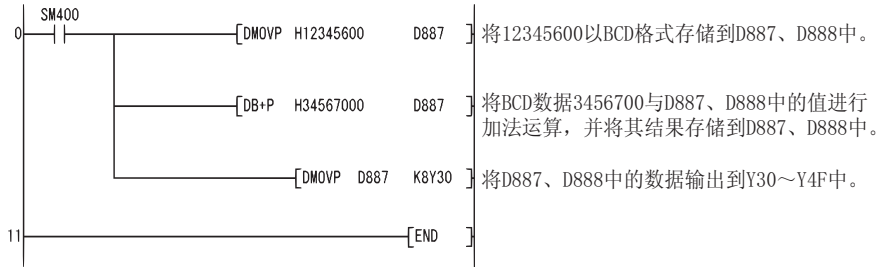
(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

·Ⓢ、Ⓓ的 BCD 数据超出了 0 ～ 99999999 的范围时。 (出错代码 : 4100)

程序示例

(1) 以下为将 12345600 及 34567000 的 BCD 数据进行加法运算，在将运算结果存储到 D887、D888 中的同时并输出到 Y30 ～ Y4F 中的程序。

[梯形图模式]

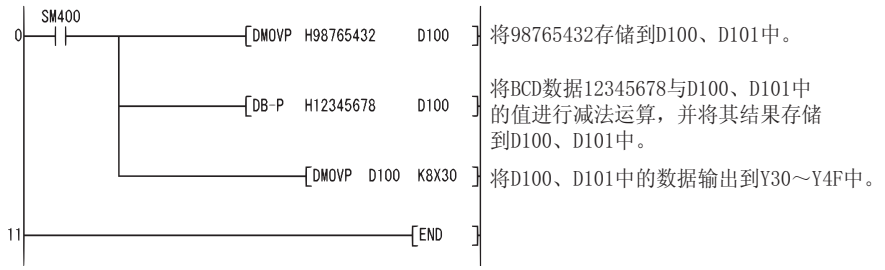


[列表模式]

步	指令	软元件	
0	LD	SM400	
1	DMOVP	H12345600	D887
4	DB+P	H34567000	D887
8	DMOVP	D887	K8Y30
11	END		

(2) 以下为将 98765432 及 12345678 的 BCD 数据进行减法运算，在将运算结果存储到 D100、D101 中的同时并输出到 Y30 ～ Y4F 中的程序。

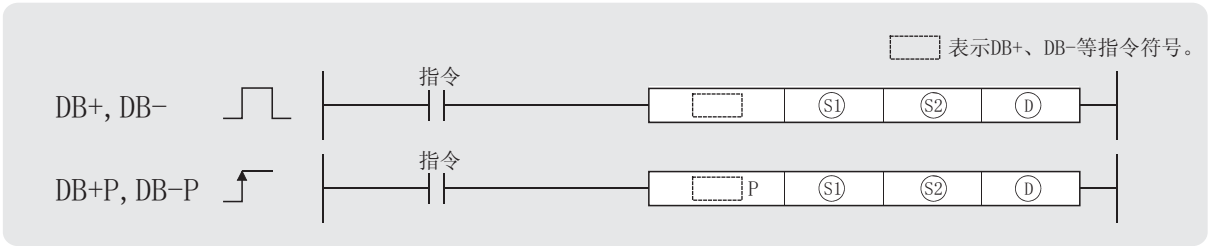
[梯形图模式]



[列表模式]

步	指令	软元件	
0	LD	SM400	
1	DMOVP	H98765432	D100
4	DB-P	H12345678	D100
8	DMOVP	D100	K8X30
11	END		

2 设置数据为 3 个时 ((Ⓢ1+1、Ⓢ1)+(Ⓢ2+1、Ⓢ2) → (Ⓓ+1、Ⓓ)、
(Ⓢ1+1、Ⓢ1)-(Ⓢ2+1、Ⓢ2) → (Ⓓ+1、Ⓓ))



Ⓢ1 ： 被加数被减数数据或者存储被加数被减数数据的软元件的起始编号 (BCD8 位)。

Ⓢ2 ： 加数减数数据或者存储加数减数数据的软元件的起始编号 (BCD8 位)。

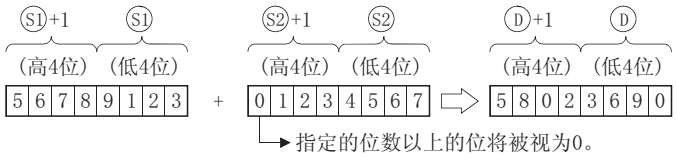
Ⓓ ： 存储运算结果的软元件的起始编号 (BCD8 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ1	○							○	--
Ⓢ2	○							○	--
Ⓓ	○							--	--

功 能

DB+

(1) 将Ⓢ1中指定的 BCD8 位数据与Ⓢ2中指定的 BCD8 位数据进行加法运算，并将加法运算结果存储到Ⓓ中指定的软元件中。

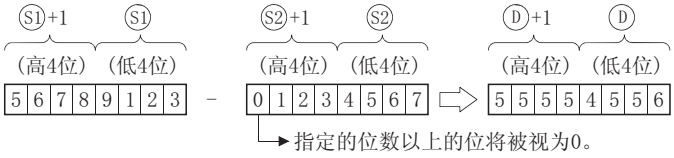


(2) Ⓢ1、Ⓢ2、Ⓓ中可指定的范围为 0 ～ 99999999(BCD8 位)。

(3) 加法运算结果超过了 99999999 时，进位将被视为无效。
在这种情况下，进位标志不变为 ON。

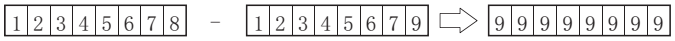
DB-

(1) 将Ⓢ1中指定的 BCD8 位数据与Ⓢ2中指定的 BCD8 位数据进行减法运算，并将减法运算结果存储到Ⓓ中指定的软元件中。



(2) Ⓢ1、Ⓢ2、Ⓓ中可指定的范围为 0 ～ 99999999(BCD8 位)。

(3) 减法运算结果中发生了下溢时的情况如下所示。
在这种情况下，进位标志不变为 ON。



出　错

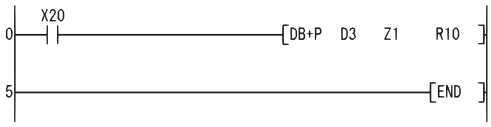
(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

·Ⓢ1、Ⓢ2、Ⓓ的 BCD 数据超出了 0 ～ 99999999 的范围时。　　(出错代码 : 4100)

程序示例

(1) 以下为 X20 变为 ON 时，将 D3、D4 的 BCD 数据与 Z1、Z2 的 BCD 数据进行加法运算，并将运算结果存储到 R10、R11 中的程序。

[梯形图模式]

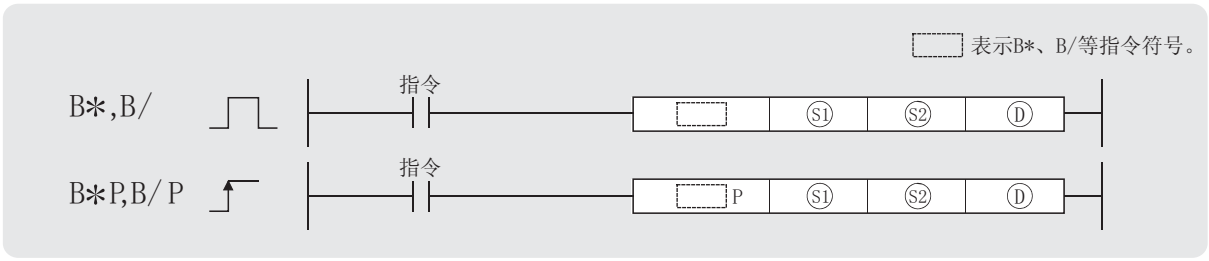


[列表模式]

步	指令	软元件		
0	LD	X20		
1	DB+P	D3	Z1	R10
5	END			

6.2.7 BCD4 位乘法和除法运算 (B*(P)、B/(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ1 ：被乘数被除数数据或者存储被乘数被除数数据的软元件的起始编号 (BCD4 位)。

Ⓢ2 ：乘数除数数据或者存储乘数除数数据的软元件的起始编号 (BCD4 位)。

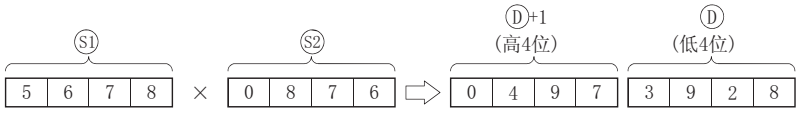
Ⓓ ：存储运算结果的软元件的起始编号 (BCD8 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ1	○							○	--
Ⓢ2	○							○	--
Ⓓ	○							--	--

功 能

B*

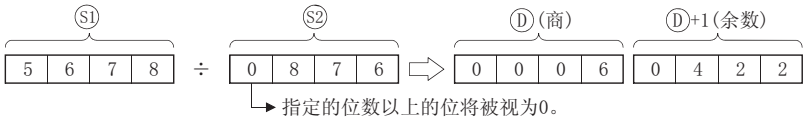
(1) 将Ⓢ1中指定的软元件的 BCD 数据与Ⓢ2中指定的软元件的 BCD 数据进行乘法运算，并将运算结果存储到Ⓓ中指定的软元件中。



(2) Ⓢ1、Ⓢ2中可指定的范围为 0 ～ 9999(BCD4 位)。

B/

(1) 将Ⓢ1中指定的软元件的 BCD 数据与Ⓢ2中指定的软元件的 BCD 数据进行除法运算，并将运算结果存储到Ⓓ中指定的软元件中。



(2) 将除法运算结果的商和余数以 32 位进行存储。
商 (BCD4 位) ..... 存储在低 16 位中。
余数 (BCD4 位) .... 存储在高 16 位中。

(3) 对Ⓓ进行了位软元件指定时，不存储除法运算结果的余数。

出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

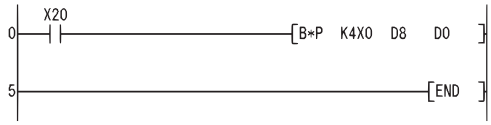
·Ⓢ1、Ⓢ2的 BCD 数据超出了 0 ～ 9999 的范围时。 (出错代码： 4100)

· 除数Ⓢ2为 0 时。 (出错代码： 4100)

程序示例

(1) 以下为 X20 变为 ON 时，将 X0 ～ XF 的 BCD 数据与 D8 的 BCD 数据进行乘法运算，并将运算结果存储到 D0、D1 中的程序。

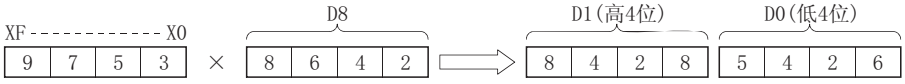
[梯形图模式]



[列表模式]

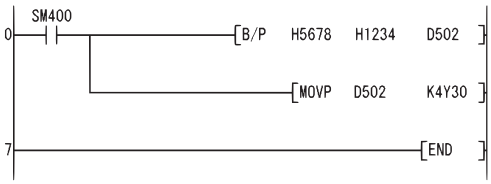
步	指令	软元件		
0	LD	X20		
1	B*P	K4X0	D8	D0
5	END			

[动作]



(2) 以下为将 5678 及 1234 的 BCD 数据进行除法运算，在将运算结果存储到 D502、D503 中的同时并将商输出到 Y30 ～ Y3F 中的程序。

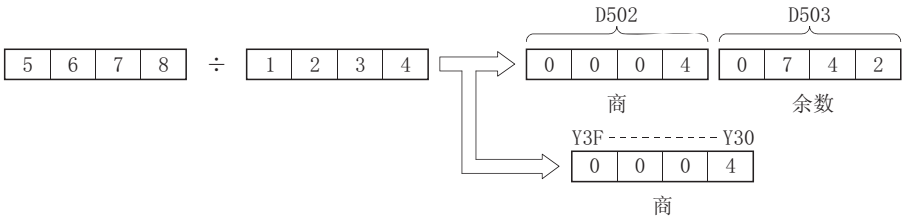
[梯形图模式]



[列表模式]

步	指令	软元件		
0	LD	SM400		
1	B/P	H5678	H1234	D502
5	MOVP	D502	K4Y30	
7	END			

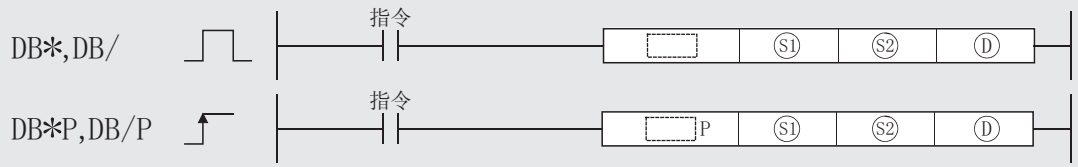
[动作]



6.2.8 BCD8 位乘法和除法运算 (DB*(P)、DB/(P))

Basic | High performance | Process | Redundant | Universal

表示DB*、DB/等指令符号。



(S1) ：被乘数被除数数据或者存储被乘数被除数数据的软元件的起始编号 (BCD8 位)。

(S2) ：乘数除数数据或者存储乘数除数数据的软元件的起始编号 (BCD8 位)。

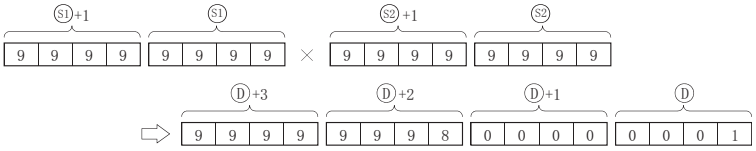
(D) ：存储运算结果的软元件的起始编号 (BCD16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
(S1)		○			○				--
(S2)		○			○				--
(D)		○			--				--

功 能

DB*

(1) 将(S1)中指定的软元件的 BCD8 位数据与(S2)中指定的软元件的 BCD8 位数据进行乘法运算，并将运算结果存储到(D)中指定的软元件中。



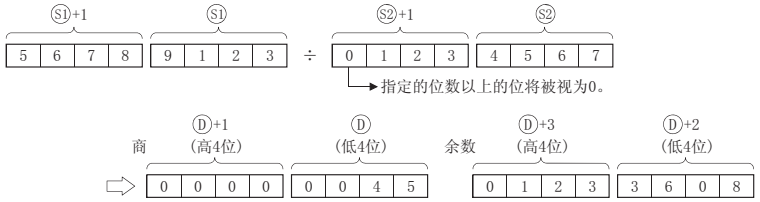
(2) (D)中指定的软元件为位软元件时，乘法运算结果以至低位 8 位数 (低 32 位) 为对象，不能对高位 8 位数 (高 32 位) 进行指定。

K1……低位 1 位数 (b0 ～ 3)，K4……低位 4 位数 (b0 ～ 15)，K8……低位 8 位数 (b0 ～ 31)

(3) (S1)、(S2)中可指定的范围为 0 ～ 99999999(BCD8 位)。

DB/

(1) 将(S1)中指定的软元件的 BCD8 位数据与(S2)中指定的软元件的 BCD8 位数据进行除法运算，并将运算结果存储到(D)中指定的软元件中。



(2) 将除法运算结果的商和余数以 64 位进行存储。

商 (BCD8 位) ..... 存储在低 32 位中。
余数 (BCD8 位) ..... 存储在高 32 位中。

(3) 对Ⓓ进行了位软元件指定时，不存储除法运算结果的余数。

出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

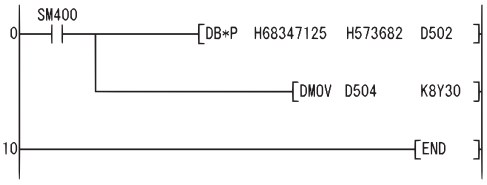
·Ⓢ1、Ⓢ2 的 BCD 数据超出了 0 ～ 99999999 的范围时。 (出错代码： 4100)

· 除数Ⓢ2 为 0 时。 (出错代码： 4100)

程序示例

(1) 以下为将 67347125 及 573682 的 BCD 数据进行乘法运算，在将运算结果存储到 D502 ～ D505 中的同时并将高 8 位输出到 Y30 ～ Y4F 中的程序。

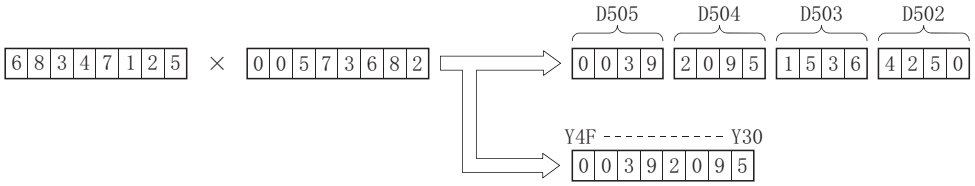
[梯形图模式]



[列表模式]

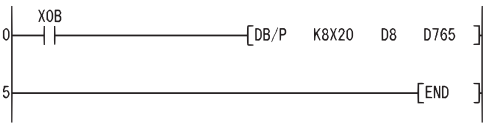
步	指令	软元件
0	LD	SM400
1	DB*P	H68347125 H573682 D502
7	DMOV	D504 K8Y30
10	END	

[动作]



(2) 以下为 X0B 变为 ON 时，将 X20 ～ X3F 的 BCD 数据与 D8、D9 的 BCD 数据进行除法运算，并将运算结果存储到 D765 ～ D768 中的程序。

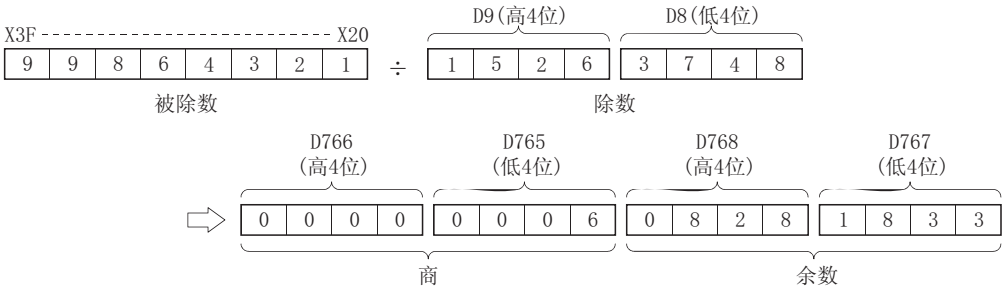
[梯形图模式]



[列表模式]

步	指令	软元件
0	LD	X0B
1	DB/P	K8X20 D8 D765
5	END	

[动作]



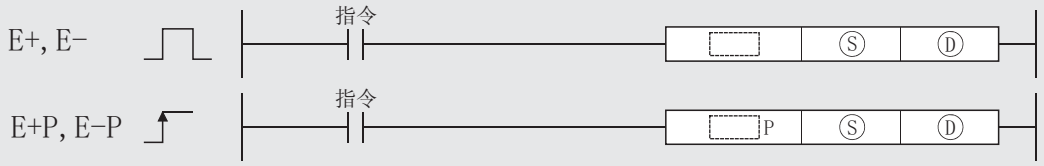
6.2.9 浮点数据的加法和减法运算（单精度）(E+(P)、E-(P))

Ver. Basic | High performance | Process | Redundant | Universal

· 基本型 QCPU：序列号的前 4 位数为“ 04122 ”以后

1 设置数据为 2 个时 ((Ⓓ+1、Ⓓ)+(Ⓢ+1、Ⓢ) → (Ⓓ+1、Ⓓ)、(Ⓓ+1、Ⓓ)-(Ⓢ+1、Ⓢ) → (Ⓓ+1、Ⓓ))

□表示E+、E-等指令符号。



Ⓢ ：加法减法数据或者存储加法减法数据的软元件的起始编号（实数）。

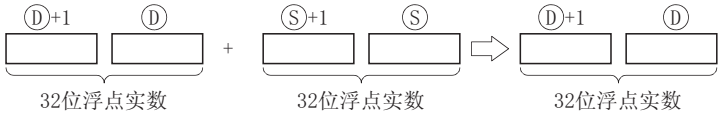
Ⓓ ：存储加法减法数据的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			E	
Ⓢ	--	○		--	○		--	○	--
Ⓓ	--	○		--	○		--	--	--

功 能

E+

(1) 将Ⓓ中指定的 32 位浮点实数与Ⓢ中指定的 32 位浮点实数进行加法运算，并将加法运算结果存储到Ⓓ中指定的软元件中。

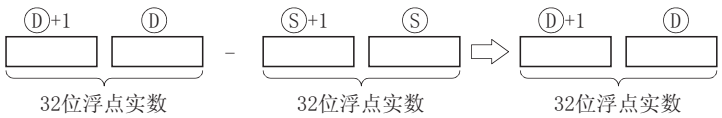


(2) Ⓢ、Ⓓ中可指定的值以及可存储的值如下所示：

0、2^{-126} ≦ | 指定值（存储值） | ＜ 2^{128}

E-

(1) 将Ⓓ中指定的 32 位浮点实数与Ⓢ中指定的 32 位浮点实数进行减法运算，并将减法运算结果存储到Ⓓ中指定的软元件中。



(2) Ⓢ、Ⓓ中可指定的值以及可存储的值如下所示：

0、2^{-126} ≦ | 指定值（存储值） | ＜ 2^{128}

出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· 指定的软元件的内容不在以下范围内时。

0、$2^{-126} \leqq$ | 指定软元件的内容 | $< 2^{128}$

(对于基本型 QCPU、高性能型 QCPU、过程 CPU、冗余 CPU) (出错代码： 4100)

· 指定软元件的内容为 -0 时。*2

(对于基本型 QCPU、高性能型 QCPU、过程 CPU、冗余 CPU) (出错代码： 4100)

*2: 有的 CPU 模块即使指定了 -0 也不会变为运算出错状态。
有关详细内容请参阅 3.2.4 项。

· 运算结果超出以下范围时。(发生了溢出时。)

$2^{128} \leqq$ | 运算结果 | (出错代码： 4141)

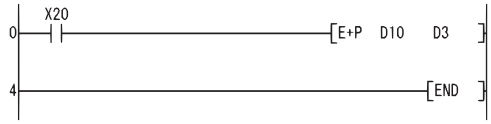
· 指定软元件的内容为 -0、非正规数、非数、±∞时。(仅通用型 QCPU)

(出错代码： 4140)

程序示例

(1) 以下为 X20 变为 ON 时，将 D3、D4 的 32 位浮点实数与 D10、D11 的 32 位浮点实数进行加法运算，并将加法运算结果存储到 D3、D4 中的程序。

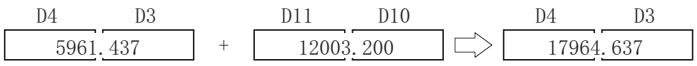
[梯形图模式]



[列表模式]

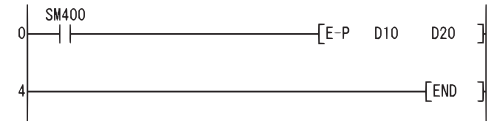
步	指令	软元件	
0	LD	X20	
1	E+P	D10	D3
4	END		

[动作]



(2) 以下为将 D20、D21 的 32 位浮点实数与 D10、D11 的 32 位浮点实数进行减法运算，并将减法运算结果存储到 D20、D21 中的程序。

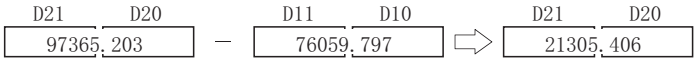
[梯形图模式]



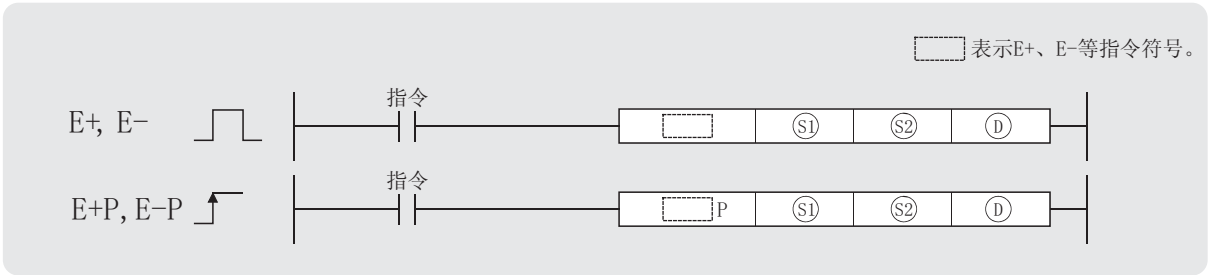
[列表模式]

步	指令	软元件	
0	LD	SM400	
1	E-P	D10	D20
4	END		

[动作]



2 设置数据为 3 个时 ((S1+1、S1)+(S2+1、S2) → (D+1、D)、
(S1+1、S1)-(S2+1、S2) → (D+1、D))



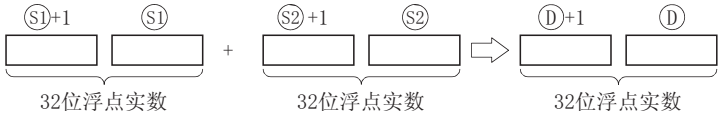
S1 ：被加数被减数数据或者存储被加数被减数数据的软元件的起始编号（实数）。
S2 ：加数减数数据或者存储加数减数数据的软元件的起始编号（实数）。
D ：存储运算结果的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
S1	--	○		--	○		--	○	--
S2	--	○		--	○		--	○	--
D	--	○		--	○		--	--	--

功 能

E+

(1) 将S1中指定的 32 位浮点实数与S2中指定的 32 位浮点实数进行加法运算，并将运算结果存储到D中指定的软元件中。

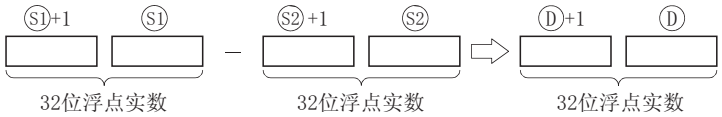


(2) S1、S2、D中可指定的值以及可存储的值如下所示。

0、$2^{-126} \leqq$ | 指定值（存储值） | $< 2^{128}$

E-

(1) 将S1中指定的 32 位浮点实数与S2中指定的 32 位浮点实数进行减法运算，并将运算结果存储到D中指定的软元件中。



(2) S1、S2、D中可指定的值以及可存储的值如下所示。

0、$2^{-126} \leqq$ | 指定值（存储值） | $< 2^{128}$

出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· 指定的软元件的内容不在以下范围内时。

$0、2^{-126} \leqq |$指定软元件的内容$| < 2^{128}$

(对于基本型 QCPU、高性能型 QCPU、过程 CPU、冗余 CPU) (出错代码： 4100)

· 指定软元件的内容为 -0 时。*1

(对于基本型 QCPU、高性能型 QCPU、过程 CPU、冗余 CPU) (出错代码： 4100)

*1: 有的 CPU 模块即使指定了 -0 也不会变为运算出错状态。
有关详细内容请参阅 3.2.4 项。

· 运算结果超出以下范围时。(发生了溢出时。)

(仅通用型 QCPU)

$2^{128} \leqq |$运算结果$|$ (出错代码： 4141)

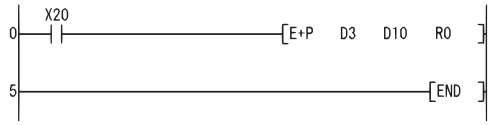
· 指定软元件的内容为 -0、非正规数、非数、±∞时。

(仅通用型 QCPU) (出错代码： 4140)

程序示例

(1) 以下为 X20 变为 ON 时，将 D3、D4 的 32 位浮点实数与 D10、D11 的 32 位浮点实数进行加法运算，并将加法运算结果存储到 R0、R1 中的程序。

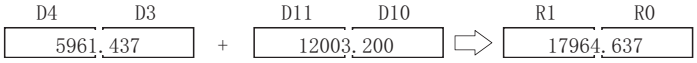
[梯形图模式]



[列表模式]

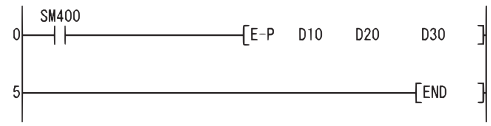
步	指令	软元件		
0	LD	X20		
1	E+P	D3	D10	R0
5	END			

[动作]



(2) 以下为将 D10、D11 的 32 位浮点实数与 D20、D21 的 32 位浮点实数进行减法运算，并将减法运算结果存储到 D30、D31 中的程序。

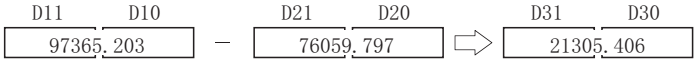
[梯形图模式]



[列表模式]

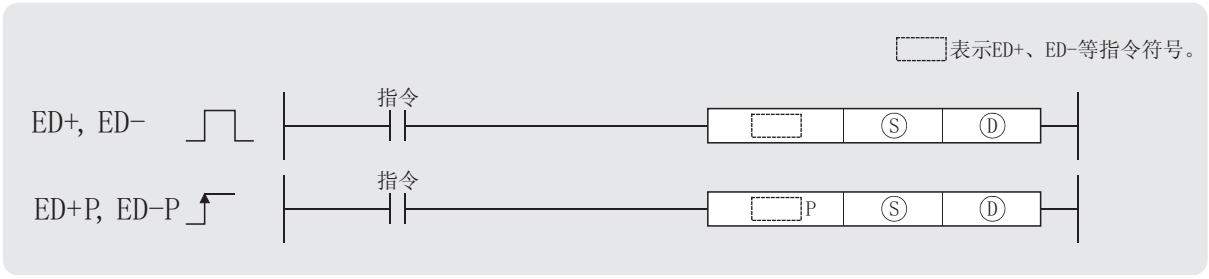
步	指令	软元件		
0	LD	SM400		
1	E-P	D10	D20	D30
5	END			

[动作]



6.2.10 浮点数据的加法和减法运算（双精度）(ED+(P)、ED-(P))

1 设置数据为 2 个时 ((Ⓓ+3、Ⓓ+2、Ⓓ+1、Ⓓ)+(Ⓢ+3、Ⓢ+2、Ⓢ+1、Ⓢ) → (Ⓓ+3、Ⓓ+2、Ⓓ+1、Ⓓ)、(Ⓓ+3、Ⓓ+2、Ⓓ+1、Ⓓ)-(Ⓢ+3、Ⓢ+2、Ⓢ+1、Ⓢ) → (Ⓓ+3、Ⓓ+2、Ⓓ+1、Ⓓ))



Ⓢ ：加法减法数据或者存储加法减法数据的软元件的起始编号（实数）。

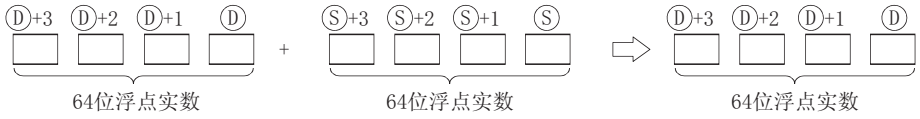
Ⓓ ：存储加法减法数据的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
Ⓢ	--	○		--				○	--
Ⓓ	--	○		--				--	--

功 能

ED+

(1) 将Ⓓ中指定的 64 位浮点实数与Ⓢ中指定的 64 位浮点实数进行加法运算，并将加法运算结果存储到Ⓓ中指定的软元件中。

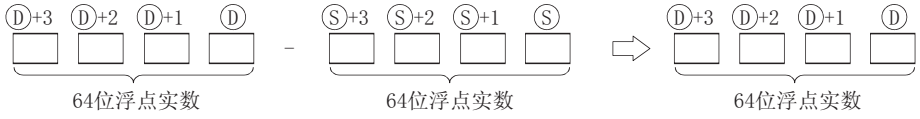


(2) Ⓢ、Ⓓ中可指定的值以及可存储的值如下所示。

0、$2^{-1022} \leqq$ | 指定值（存储值）| $< 2^{1024}$

ED-

(1) 将Ⓓ中指定的 64 位浮点实数与Ⓢ中指定的 64 位浮点实数进行减法运算，并将减法运算结果存储到Ⓓ中指定的软元件中。



(2) Ⓢ、Ⓓ中可指定的值以及可存储的值如下所示。

0、$2^{-1022} \leqq$ | 指定值（存储值）| $< 2^{1024}$

出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· 指定的软元件的内容不在以下范围内时。 (出错代码： 4140)

$0、2^{-1022} \leqq |$ 指定软元件的内容 $| < 2^{1024}$

· 指定软元件的内容为 -0 时。 (出错代码： 4140)

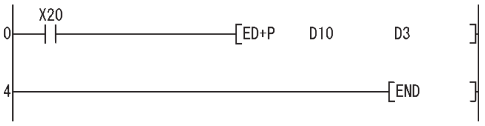
· 运算结果超出以下范围时。(发生了溢出时。)

$2^{1024} \leqq |$ 运算结果 $|$ (出错代码： 4141)

程序示例

(1) 以下为 X20 变为 ON 时，将 D3 ～ D6 的 64 位浮点实数与 D10 ～ D13 的 64 位浮点实数进行加法运算，并将加法运算结果存储到 D3 ～ D6 中的程序。

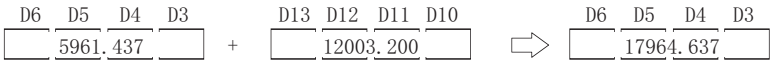
[梯形图模式]



[列表模式]

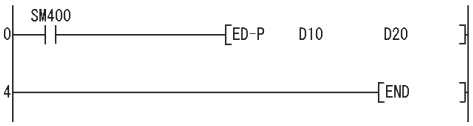
步	指令	软元件	
0	LD	X20	
1	ED+P	D10	D3
4	END		

[动作]



(2) 以下为将 D20 ～ D23 的 64 位浮点实数与 D10 ～ D13 的 64 位浮点实数进行减法运算，并将减法运算结果存储到 D20 ～ D23 中的程序。

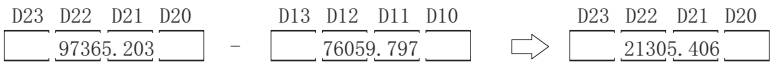
[梯形图模式]



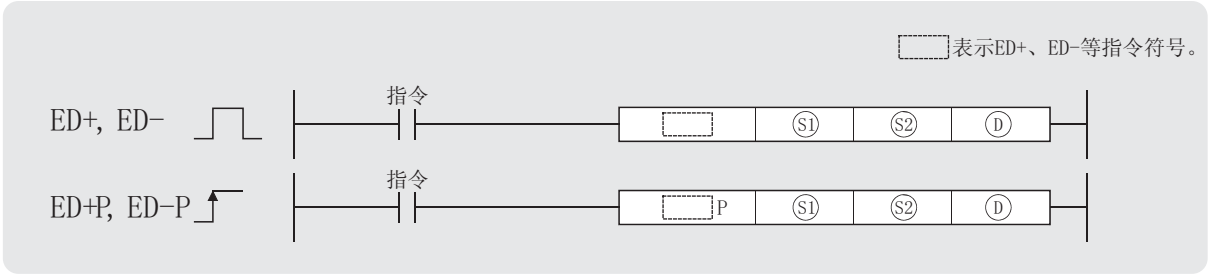
[列表模式]

步	指令	软元件	
0	LD	SM400	
1	ED-P	D10	D20
4	END		

[动作]



2 设置数据为 3 个时 ((S1)+3、(S1)+2、(S1)+1、(S1))+((S2)+3、(S2)+2、(S2)+1、(S2)) → ((D)+3、(D)+2、(D)+1、(D))、((S1)+3、(S1)+2、(S1)+1、(S1))-((S2)+3、(S2)+2、(S2)+1、(S2)) → ((D)+3、(D)+2、(D)+1、(D))



(S1) ：被加数被减数数据或者存储被加数被减数数据的软元件的起始编号（实数）。

(S2) ：加数减数数据或者存储加数减数数据的软元件的起始编号（实数）。

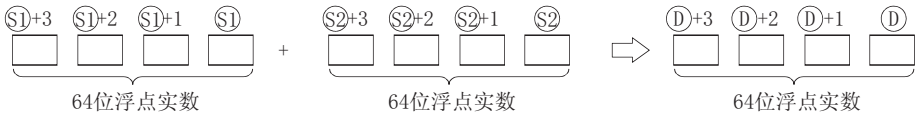
(D) ：存储运算结果的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
(S1)	--	○		--				○	--
(S2)	--	○		--				○	--
(D)	--	○		--				--	--

功 能

ED+

(1) 将(S1)中指定的 64 位浮点实数与(S2)中指定的 64 位浮点实数进行加法运算，并将运算结果存储到(D)中指定的软元件中。

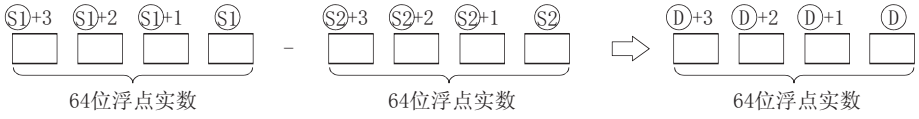


(2) (S1)、(S2)、(D)中可指定的值以及可存储的值如下所示。

0、 $2^{-1022} \leqq$ | 指定值（存储值） | $< 2^{1024}$

ED-

(1) 将(S1)中指定的 64 位浮点实数与(S2)中指定的 64 位浮点实数进行减法运算，并将减法运算结果存储到(D)中指定的软元件中。



(2) (S1)、(S2)、(D)中可指定的值以及可存储的值如下所示。

0、 $2^{-1022} \leqq$ | 指定值（存储值） | $< 2^{1024}$

出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· 指定的软元件的内容不在以下范围内时。 (出错代码 : 4140)

$0、2^{-1022} \leqq |$ 指定软元件的内容 $| < 2^{1024}$

· 指定软元件的内容为 -0 时。 (出错代码 : 4140)

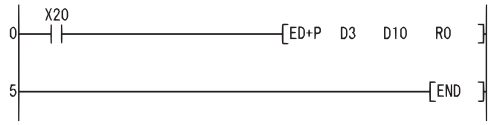
· 运算结果超出以下范围时。(发生了溢出时。)

$2^{1024} \leqq |$ 运算结果 $|$ (出错代码 : 4141)

程序示例

(1) 以下为 X20 变为 ON 时，将 D3 ～ D6 的 64 位浮点实数与 D10 ～ D13 的 64 位浮点实数进行加法运算，并将加法运算结果存储到 R0 ～ R3 中的程序。

[梯形图模式]



[列表模式]

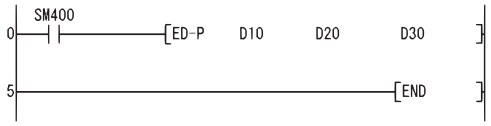
步	指令	软元件		
0	LD	X20		
1	ED+P	D3	D10	R0
5	END			

[动作]

D6	D5	D4	D3		D13	D12	D11	D10		R3	R2	R1	R0
	5961.	437		+		12003.	200		⇨		17964.	637	

(2) 以下为将 D10 ～ D13 的 64 位浮点实数与 D20 ～ D23 的 64 位浮点实数进行减法运算，并将减法运算结果存储到 D30 ～ D33 中的程序。

[梯形图模式]



[列表模式]

步	指令	软元件		
0	LD	SM400		
1	ED-P	D10	D20	D30
5	END			

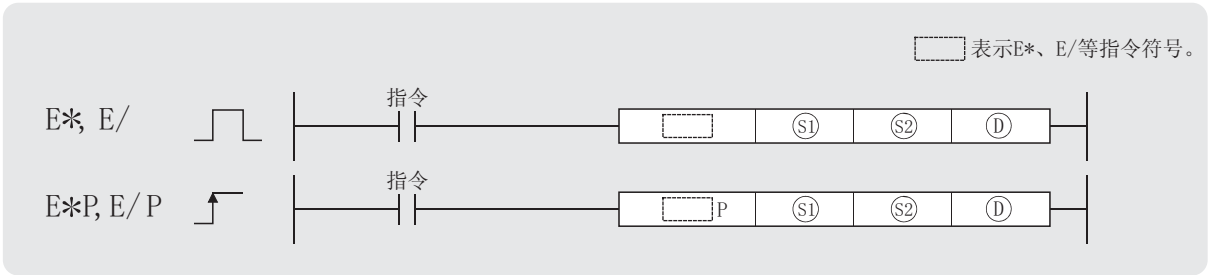
[动作]

D13	D12	D11	D10		D23	D22	D21	D20		D33	D32	D31	D30
	97365.	203		−		76059.	797		⇨		21305.	406	

6.2.11 浮点数据的乘法和除法运算（单精度）(E*(P)、E/(P))

Ver. Basic | High performance | Process | Redundant | Universal

基本型 QCPU：序列号的前 5 位数为“ 04122 ”以后



Ⓢ1 ：被乘数被除数数据或者存储被乘数被除数数据的软元件的起始编号（实数）。

Ⓢ2 ：乘数除数数据或者存储乘数除数数据的软元件的起始编号（实数）。

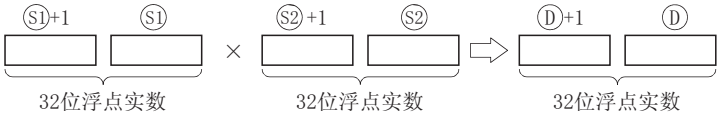
Ⓓ ：存储运算结果的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			E	
Ⓢ1	--	○		--	○		--	○	--
Ⓢ2	--	○		--	○		--	○	--
Ⓓ	--	○		--	○		--	--	--

功 能

E*

(1) 将Ⓢ1中指定的 32 位浮点实数与Ⓢ2中指定的 32 位浮点实数进行乘法运算，并将乘法运算结果存储到Ⓓ中指定的软元件中。

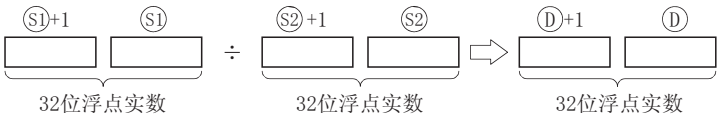


(2) Ⓢ1、Ⓢ2、Ⓓ中可指定的值以及可存储的值如下所示。

0、$2^{-126} \leqq$ | 指定值（存储值）| $< 2^{128}$

E/

(1) 将Ⓢ1中指定的 32 位浮点实数与Ⓢ2中指定的 32 位浮点实数进行除法运算，并将除法运算结果存储到Ⓓ中指定的软元件中。



(2) Ⓢ1、Ⓢ2、Ⓓ中可指定的值以及可存储的值如下所示。

0、$2^{-126} \leqq$ | 指定值（存储值）| $< 2^{128}$

出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· 指定的软元件的内容不在以下范围内时。

$0、2^{-126} \leqq |$指定软元件的内容$| < 2^{128}$

(对于基本型 QCPU、高性能型 QCPU、过程 CPU、冗余 CPU) (出错代码： 4100)

· 指定软元件的内容为 -0 时。*2

(对于基本型 QCPU、高性能型 QCPU、过程 CPU、冗余 CPU) (出错代码： 4100)

*2: 有的 CPU 模块即使指定了 -0 也不会变为运算出错状态。
有关详细内容请参阅 3.2.4 项。

· 运算结果超出以下范围时。(发生了溢出时。)

(仅通用型 QCPU)

$2^{128} \leqq |$运算结果$|$ (出错代码： 4141)

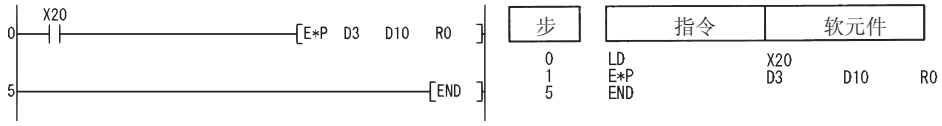
· 指定软元件的内容为 -0、非正规数、非数、±∞时。

(仅通用型 QCPU) (出错代码： 4140)

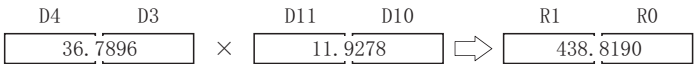
程序示例

(1) 以下为 X20 变为 ON 时，将 D3、D4 的 32 位浮点实数与 D10、D11 的 32 位浮点实数进行乘法运算，并将乘法运算结果存储到 R0、R1 中的程序。

[梯形图模式] [列表模式]

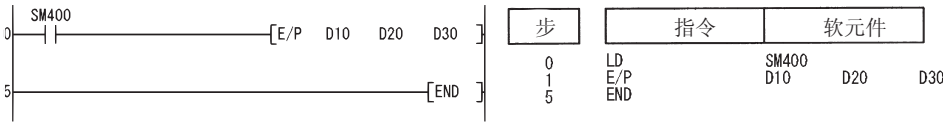


[动作]

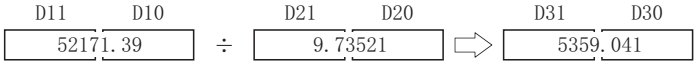


(2) 以下为将 D10、D11 的 32 位浮点实数与 D20、D21 的 32 位浮点实数进行除法运算，并将除法运算结果存储到 D30、D31 中的程序。

[梯形图模式] [列表模式]

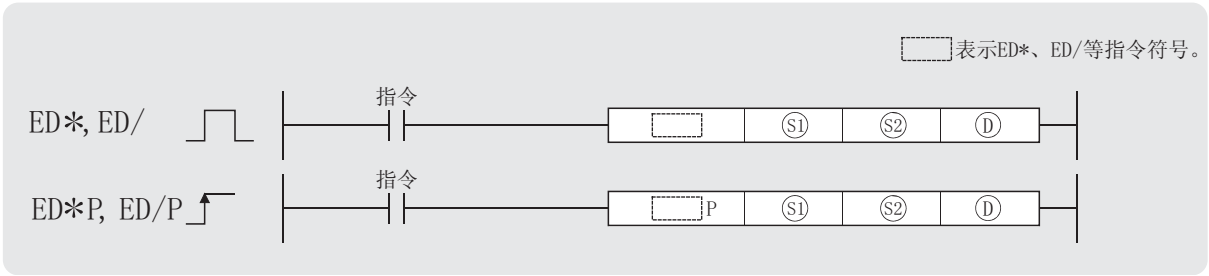


[动作]



6.2.12 浮点数据的乘法和除法运算（双精度）(ED*(P)、ED/(P))

Basic　High performance　Process　Redundant　Universal



Ⓢ1 ：被乘数被除数数据或者存储被乘数被除数数据的软元件的起始编号（实数）。

Ⓢ2 ：乘数除数数据或者存储乘数除数数据的软元件的起始编号（实数）。

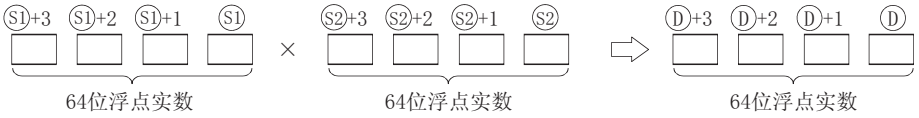
Ⓓ ：存储运算结果的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			E	
Ⓢ1	--	○		--				○	--
Ⓢ2	--	○		--				○	--
Ⓓ	--	○		--				--	--

功　能

ED*

(1) 将Ⓢ1中指定的 64 位浮点实数与Ⓢ2中指定的 64 位浮点实数进行乘法运算，并将乘法运算结果存储到Ⓓ中指定的软元件中。



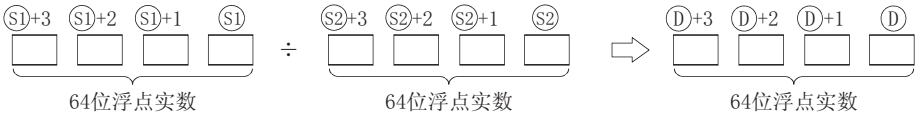
(2) Ⓢ1、Ⓢ2、Ⓓ中可指定的值以及可存储的值如下所示。

0、$2^{-1022} \leqq$ | 指定值（存储值）| $< 2^{1024}$

(3) 运算结果为 -0 或者发生了下溢时，将运算结果作为 0 处理。

ED/

(1) 将Ⓢ1中指定的 64 位浮点实数与Ⓢ2中指定的 64 位浮点实数进行除法运算，并将除法运算结果存储到Ⓓ中指定的软元件中。



(2) Ⓢ1、Ⓢ2、Ⓓ中可指定的值以及可存储的值如下所示。

0、$2^{-1022} \leqq$ | 指定值（存储值）| $< 2^{1024}$

(3) 运算结果为 -0 或者发生了下溢时，将运算结果作为 0 处理。

出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· 指定的软元件的内容不在以下范围内时。 (出错代码 : 4140)

$0、2^{-1022} \leqq |$ 指定软元件的内容 $| < 2^{1024}$

· 指定软元件的内容为 -0 时。 (出错代码 : 4140)

· 除法指令时㉗的值为 0 时。 (出错代码 : 4100)

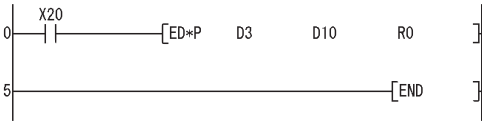
· 运算结果超出以下范围时。(发生了溢出时。)

$2^{1024} \leqq |$ 运算结果 $|$ (出错代码 : 4141)

程序示例

(1) 以下为 X20 变为 ON 时，将 D3 ～ D6 的 64 位浮点实数与 D10 ～ D13 的 64 位浮点实数进行乘法运算，并将乘法运算结果存储到 R0 ～ R3 中的程序。

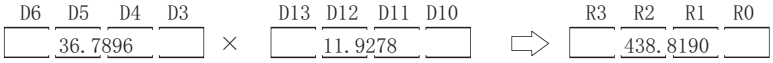
[梯形图模式]



[列表模式]

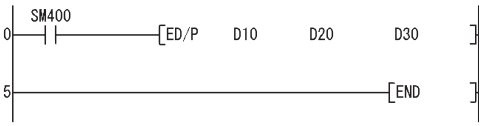
步	指令	软元件		
0	LD	X20		
1	ED*P	D3	D10	R0
5	END			

[动作]



(2) 以下为将 D10 ～ D13 的 64 位浮点实数与 D20 ～ D23 的 64 位浮点实数进行除法运算，并将除法运算结果存储到 D30 ～ D33 中的程序。

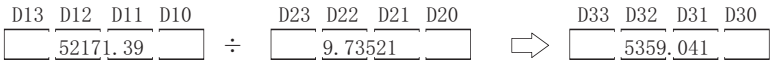
[梯形图模式]



[列表模式]

步	指令	软元件		
0	LD	SM400		
1	ED/P	D10	D20	D30
5	END			

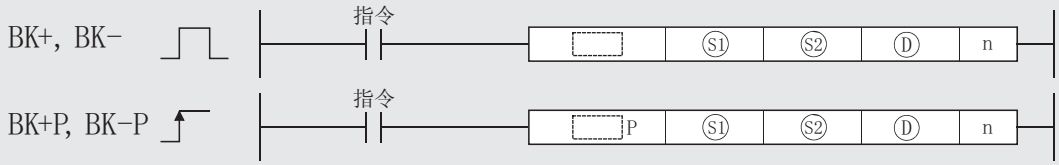
[动作]



6.2.13 BIN 16 位数据块加法和减法运算 (BK+(P)、BK-(P))

Basic | High performance | Process | Redundant | Universal

□表示BK+、BK-等指令符号。



Ⓢ1 ： 被加数被减数数据或者存储被加数被减数数据的软元件的起始编号 (BIN16 位)。

Ⓢ2 ： 加数减数数据或者存储加数减数数据的软元件的起始编号 (BIN16 位)。

Ⓓ ： 存储运算结果的软元件的起始编号 (BIN16 位)。

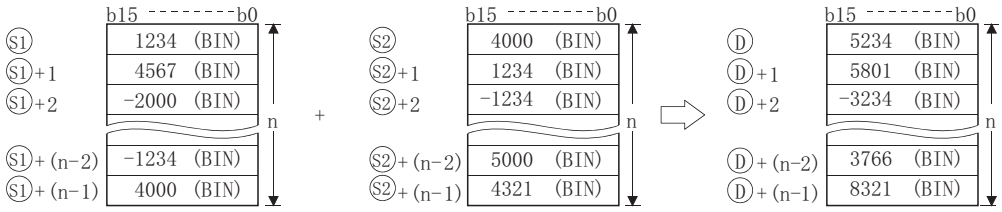
n ： 加法减法运算数据个数 (BIN16 位)

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ1	--	○		--				--	--
Ⓢ2	--	○		--				○	--
Ⓓ	--	○		--				--	--
n	○	○		○				○	--

功 能

BK+

(1) 将从Ⓢ1中指定的软元件开始的 n 点的 BIN 数据与Ⓢ2中指定的软元件开始的 n 点的 BIN 数据进行加法运算，并将运算结果存储到Ⓓ中指定的软元件后面。



(2) 块加法运算是以 16 位为单位进行。

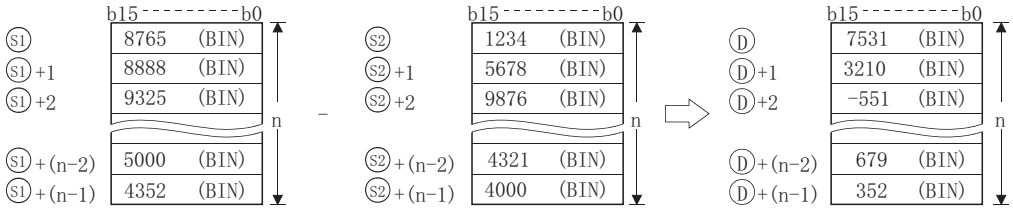
(3) Ⓢ2中可指定的常数范围为 -32768 ～ 32767(BIN16 位)。

(4) 运算结果中发生了下溢或者上溢时的情况如下所示。
在这种情况下，进位标志不变为 ON。

- K32767 (7FFFH) +K2 (0002H) → K-32767 (8001H)
- K-32767 (8001H) +K-2 (FFFEH) → K32767 (7FFFH)

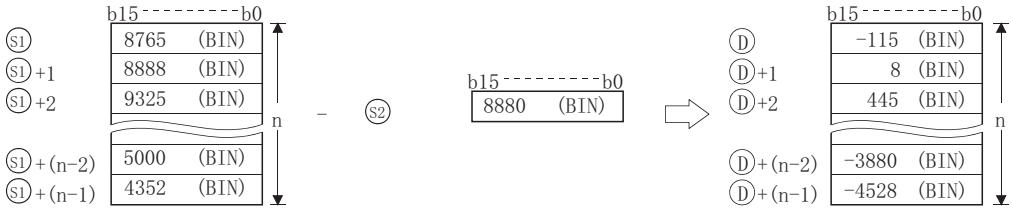
BK-

(1) 将从(S1)中指定的软元件开始的 n 点的 BIN 数据与(S2)中指定的软元件开始的 n 点的 BIN 数据进行减法运算，并将运算结果存储到(D)中指定的软元件后面。



(2) 块减法运算是以 16 位为单位进行。

(3) (S2)中可指定的常数范围为 -32768 ～ 32767(BIN16 位)。



(4) 运算结果中发生了下溢或者上溢时的情况如下所示。
在这种情况下，进位标志不变为 ON。

- K-32768 (8000H) -K2 (0002H) → K32766 (7FFEH)
- K32767 (7FFFH) -K-2 (FFFEH) → -32767 (8001H)

出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· 从(S1)、(S2)、(D)的软元件开始的 n 点的范围超出了相应软元件范围时。
(出错代码： 4101)

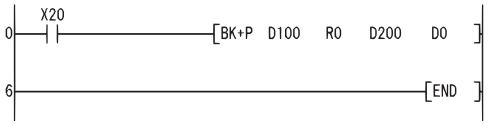
· (S1)与(D)的软元件范围相重复时。((S1)与(D)中指定了同一个软元件时除外。)
(出错代码： 4101)

· (S2)与(D)的软元件范围相重复时。((S2)与(D)中指定了同一个软元件时除外。)
(出错代码： 4101)

程序示例

(1) 以下为 X20 变为 ON 时，将 D100 ～ D103 中存储的数据值与 R0 ～ R3 中存储的数据值进行加法运算，并将其结果存储到 D200 后面的程序。

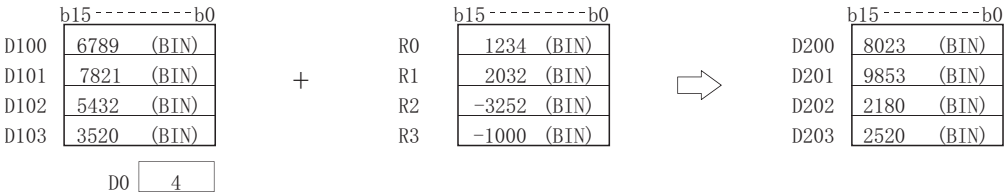
[梯形图模式]



[列表模式]

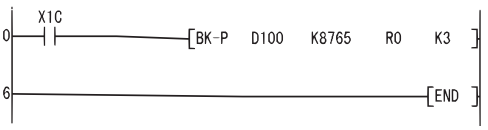
步	指令	软元件			
0	LD	X20			
1	BK+P	D100	R0	D200	D0
6	END				

[动作]



(2) 以下为 X1C 变为 ON 时，将常数 D100 ～ D102 的数据与常数 8765 进行减法运算，并将其结果存储到 R0 后面的程序。

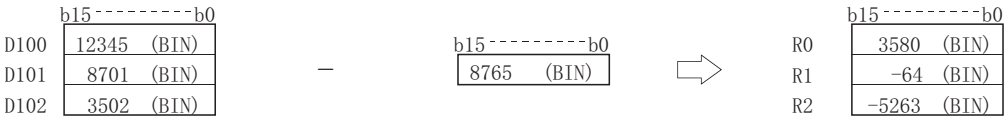
[梯形图模式]



[列表模式]

步	指令	软元件			
0	LD	X1C			
1	BK-P	D100	K8765	R0	K3
6	END				

[动作]

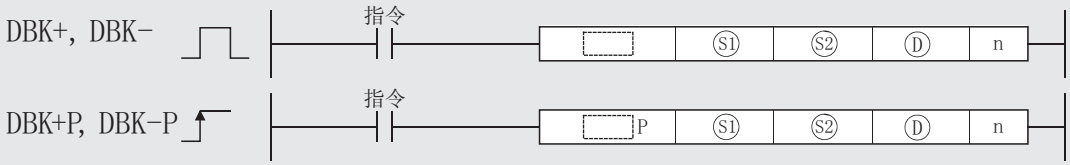


6.2.14 BIN 32 位数据块加法和减法运算 (DBK+(P)、DBK-(P))

Basic / High performance / Process / Redundant / Ver. Universal

· QnU(D)(H)CPU：序列号的前 5 位数为“ 10102 ”以后
· QnUDE(H)CPU：序列号的前 5 位数为“ 10102 ”以后

□表示DBK+、DBK-等指令符号。



S1 ： 被加数被减数数据或者存储被加数被减数数据的软元件的起始编号 (BIN32 位)。
S2 ： 加数减数数据或者存储加数减数数据的软元件的起始编号 (BIN32 位)。
D ： 存储运算结果的软元件的起始编号 (BIN32 位)。
n ： 加法减法运算数据个数 (BIN16 位)

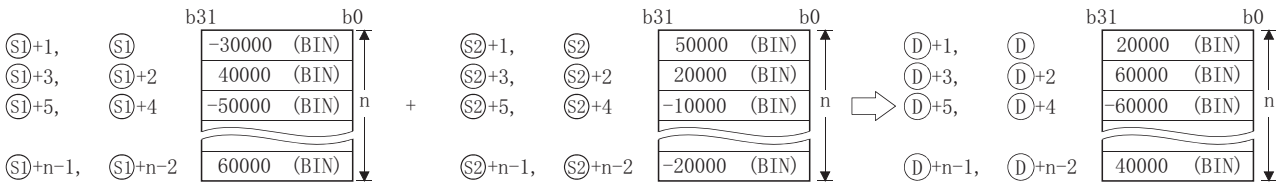
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
S1	--	○		--				--	--
S2	--	○		--				○	--
D	--	○		--				--	--
n	--	○		○				○	--

功 能

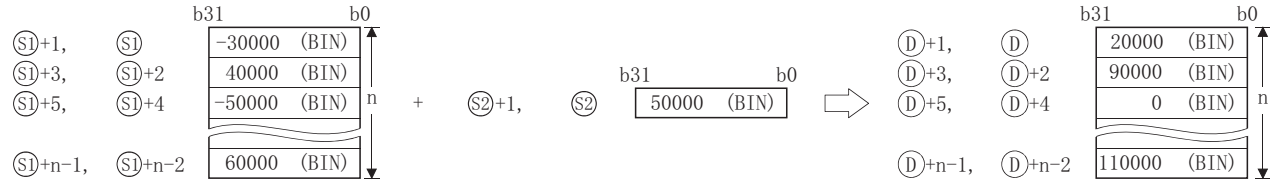
DBK+

(1) 将从S1中指定的软元件开始的 n 点的 BIN32 位数据与S2中指定的软元件开始的 n 点的 BIN32 位数据或者常数进行加法运算，并将运算结果存储到D中指定的软元件后面。

S2中指定了软元件时



S2中指定了常数时



(2) 块加法运算是以 32 位为单位进行。

(3) ⓢ2 中可指定的常数范围为 -2147483648 ～ 2147483647(BIN32 位)。

(4) n 中指定的值为 0 时将变为无处理。

(5) 运算结果中发生了溢出时的情况如下所示。
此时，进位标志不变为 ON。

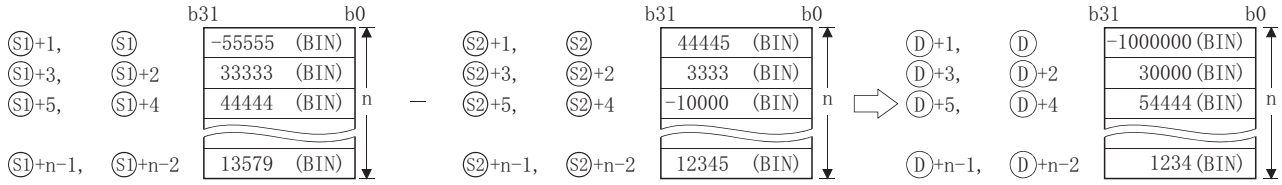
- K2147483647 (7FFFFFFFH) +K2 (00000002H) → K-2147483647 (80000001H)

- K-2147483647 (80000001H) +K-2 (FFFFFFFEH) → K2147483647 (7FFFFFFFH)

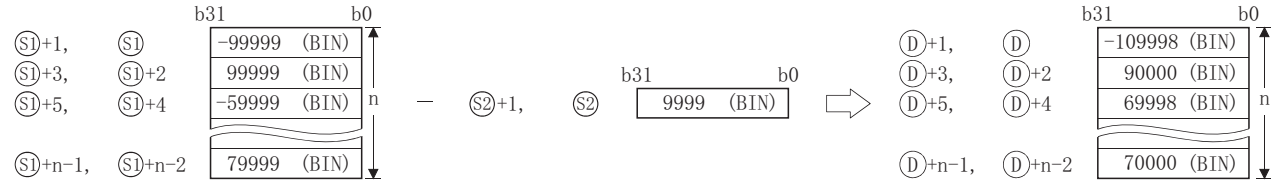
DBK-

(1) 将从ⓢ1 中指定的软元件开始的 n 点的 BIN32 位数据与ⓢ2 中指定的软元件开始的 n 点的 BIN32 位数据或者常数进行减法运算，并将运算结果存储到ⓓ中指定的软元件后面。

ⓢ2 中指定了软元件时



ⓢ2 中指定了常数时



(2) 块减法运算是以 32 位为单位进行。

(3) ⓢ2 中可指定的常数范围为 -2147483648 ～ 2147483647(BIN32 位)。

(4) n 中指定的值为 0 时将变为无处理。

(5) ⓓ的指定应在从ⓢ1 开始的 n 点的软元件范围及从ⓢ2 开始的 n 点的软元件范围以外。
但是，与ⓢ1 或者ⓢ2 指定了同一个软元件时除外。

(6) 运算结果中发生了溢出时的情况如下所示。
此时，进位标志不变为 ON。

- K2147483647 (7FFFFFFFH) -K - 2 (00000002H) → K-2147483647 (8000001H)

- K-2147483647 (80000001H) -K2 (FFFFFFFEH) → K2147483647 (7FFFFFFFH)

出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· n 中指定了负值时。 (出错代码： 4100)

· 从Ⓢ1、Ⓢ2、Ⓓ中指定的软元件开始的 n 点的范围超出了指定软元件的范围时。
(出错代码： 4101)

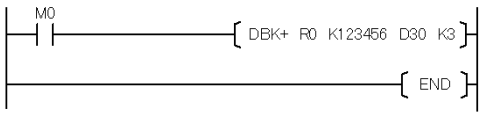
· 从Ⓢ1开始的 n 点的软元件范围与从Ⓓ开始的 n 点的软元件范围重复时。
(Ⓢ1与Ⓓ中指定了同一个软元件时除外。) (出错代码： 4101)

· 从Ⓢ2开始的 n 点的软元件范围与从Ⓓ开始的 n 点的软元件范围重复时。
(出错代码： 4101)

程序示例

(1) 以下为 M0 变为 ON 时，将 R0 ～ R5 中存储的值与常数进行加法运算，并将运算结果存储到 D30 ～ D35 中的程序。

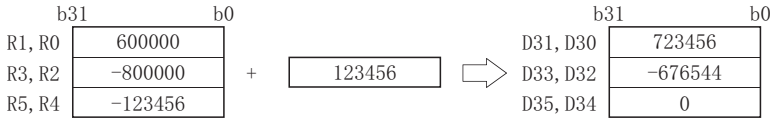
[梯形图模式]



[列表模式]

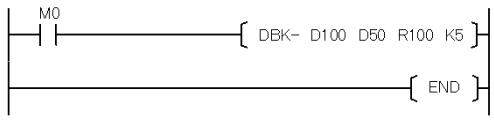
步	指令	软元件
0	LD	M0
1	DBK+	R0 K123456 D30 K3
6	END	

[动作]



(2) 以下为 M0 变为 ON 时，将 D100 ～ D109 中存储的值与 D50 ～ D59 中存储的值进行减法运算，并将运算结果存储到 R100 ～ R109 中的程序。

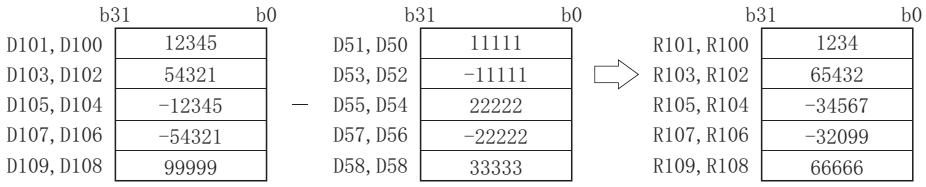
[梯形图模式]



[列表模式]

步	指令	软元件
0	LD	M0
1	DBK-	D100 D50 R100 K5
6	END	

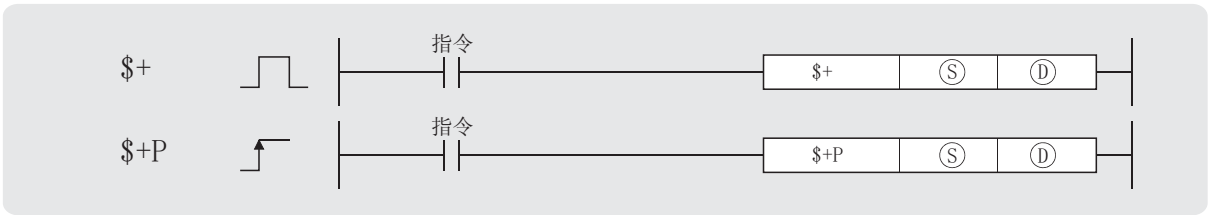
[动作]



6.2.15 字符串的合并 ($+(P))

Basic | High performance | Process | Redundant | Universal

1 设置数据为 2 个时 (Ⓓ+Ⓢ→Ⓓ)



Ⓢ : 合并的数据或者存储合并数据的软元件的起始编号 (字符串)。

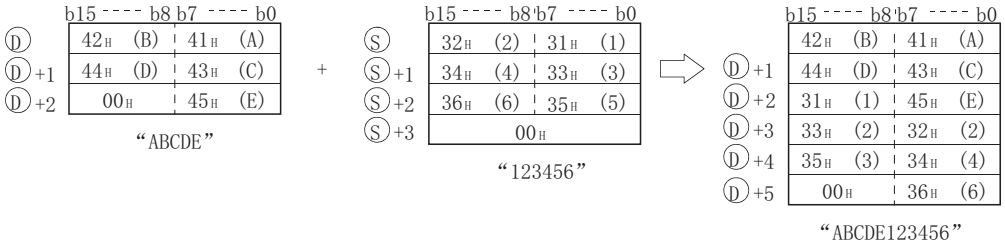
Ⓓ : 存储合并数据的软元件的起始编号 (字符串)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
Ⓢ	--	○		--				○	--
Ⓓ	--	○		--				--	--

功 能

(1) 将Ⓢ中指定的字符串数据连接到Ⓓ中指定的字符串数据的后面，并将结果存储到Ⓓ中指定编号的软元件的后面。

字符串数据是以从Ⓓ、Ⓢ中指定的软元件编号开始，至存储了“ 00H ”的软元件编号为止中存储的字符串数据为对象。



(2) 字符串合并时，表示Ⓓ中指定的字符串的结束的“ 00H ”将被视为无效，Ⓢ中指定的字符串将被连接到Ⓓ中字符串的最后字符处。

出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· 从ⓓ中指定的软元件编号后面起，至相应软元件的最终软元件号为止的点数不能容纳合并后的字符串时。 (出错代码： 4101)

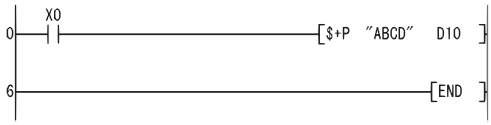
· Ⓢ与ⓓ中指定的存储字符串的软元件编号重复时。 (出错代码： 4101)

· Ⓢ与ⓓ中的字符串超过了 16383 个字符时。 (出错代码： 4101)

程序示例

(1) 以下为 X0 变为 ON 时，将 D10 ～ D12 中存储的字符串与字符串 “ ABCD ”合并的程序。

[梯形图模式]

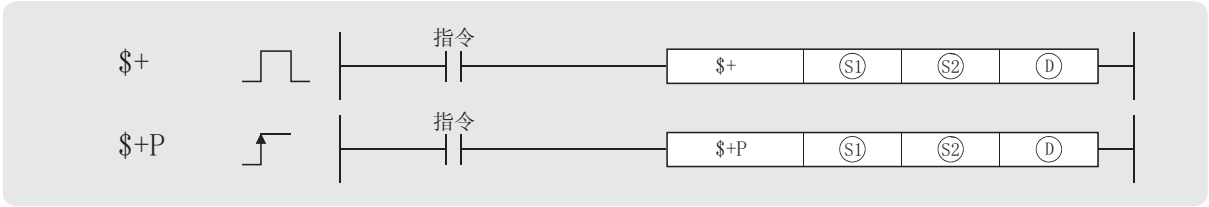


[列表模式]

步	指令	软元件
0	LD	X0
1	$+P	"ABCD" D10
6	END	

[动作]

2 设置数据为 3 个时 (Ⓢ1+Ⓢ2→Ⓓ)



Ⓢ1 ：合并的数据或者存储合并数据的软元件的起始编号（字符串）。

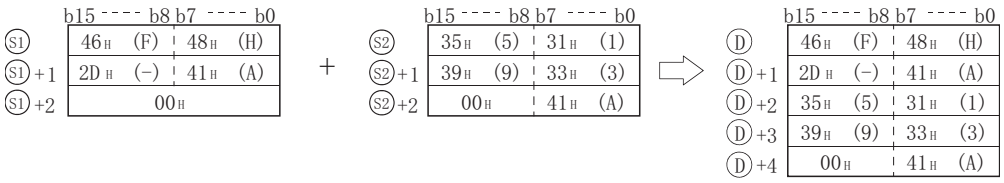
Ⓢ2 ：被合并的数据或者存储被合并数据的软元件的起始编号（字符串）。

Ⓓ ：存储合并结果的软元件的起始编号（字符串）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
Ⓢ1	--	○		--				○	--
Ⓢ2	--	○		--				○	--
Ⓓ	--	○		--				--	--

功 能

(1) 将Ⓢ2中指定的字符串数据连接到Ⓢ1中指定的字符串数据的后面，并将结果存储到Ⓓ中指定编号的软元件的后面。



(2) 字符串合并时，表示Ⓢ1中指定的字符串的结束的“00H”将被视为无效，Ⓢ2中指定的字符串将被连接到Ⓢ1中字符串的最后字符处。

出 错

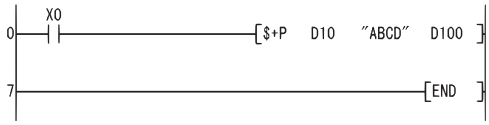
(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 从ⓓ中指定的软元件编号后面起，至相应软元件的最终软元件号为止的点数不能容纳合并后的字符串时。 (出错代码： 4101)
- ⓢ1与ⓢ2中指定的软元件编号重复时。 (出错代码： 4101)
- ⓢ2与ⓓ中指定的软元件编号重复时。 (出错代码： 4101)
- ⓢ1、ⓢ2、ⓓ的字符串超过了 16383 个字符时。 (出错代码： 4101)

程序示例

(1) 以下为 X0 变为 ON 时，将 D10 ～ D12 中存储的字符串与字符串 “ ABCD ”合并，并将结果存储到 D100 后面的程序。

[梯形图模式]



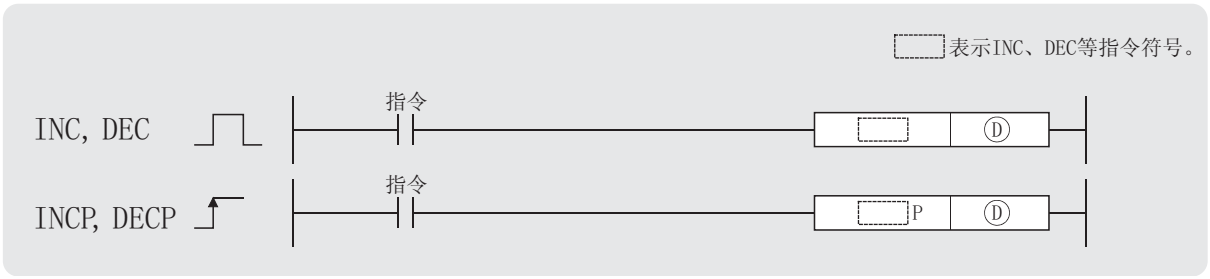
[列表模式]

步	指令	软元件		
0	LD	X0		
1	$+P	D10	"ABCD"	D100
7	END			

[动作]

6.2.16 16 位 BIN 数据的递增和递减运算 (INC(P)、DEC(P))

Basic | High performance | Process | Redundant | Universal



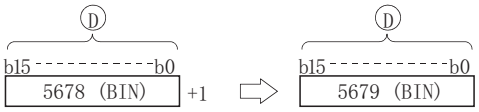
Ⓓ : 执行 INC(+1)、DEC(-1) 的软元件的起始编号 (BIN16 位)

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
Ⓓ	○							--	

功 能

INC

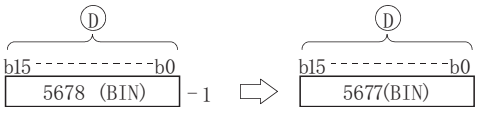
(1) 对Ⓓ中指定的软元件 (16 位数据) 进行 +1。



(2) 在Ⓓ中指定的软元件的值为 32767 时如果执行了 INC、INCP，将在Ⓓ中指定的软元件中存储 -32768。

DEC

(1) 对Ⓓ中指定的软元件 (16 位数据) 进行 -1。



(2) 在Ⓓ中指定的软元件的值为 -32768 时如果执行了 DEC、DECP，将在Ⓓ中指定的软元件中存储 32767。

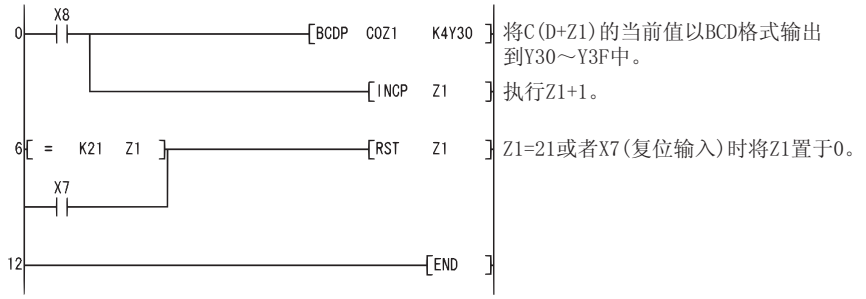
出 错

(1) 在 INC(P)/DEC(P) 指令中无运算出错。

程序示例

(1) 以下为每当 X8 变为 ON 时，将计数器 C0 ～ C20 的当前值以 BCD 格式输出到 Y30 ～ Y3F 中的程序。(当前值＜ 9999 时)

[梯形图模式]

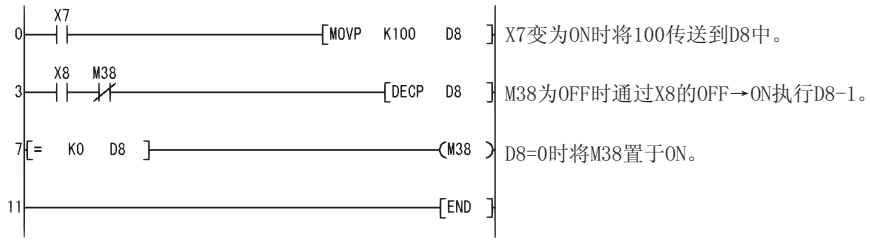


[列表模式]

步	指令	软元件	
0	LD	X8	
1	BCDP	C0Z1	K4Y30
4	INCP	Z1	
6	LD=	K21	Z1
9	OR	X7	
10	RST	Z1	
12	END		

(2) 减法计数器的程序

[梯形图模式]

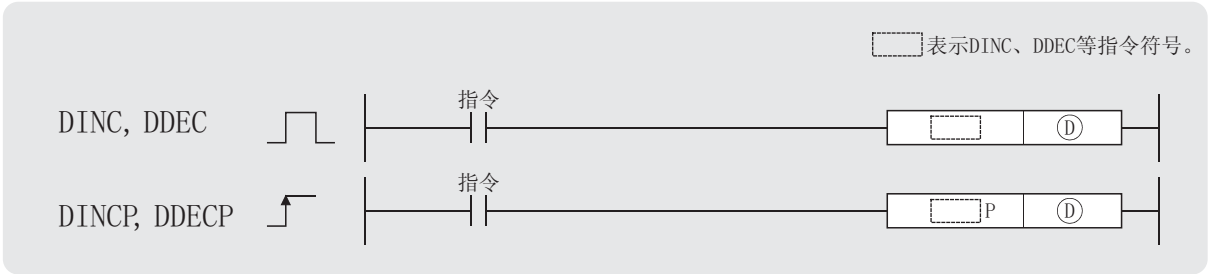


[列表模式]

步	指令	软元件	
0	LD	X7	
1	MOVP	K100	D8
3	LD	X8	
4	ANI	M38	
5	DECP	D8	
7	LD=	K0	D8
10	OUT	M38	
11	END		

6.2.17 32 位 BIN 数据的递增和递减运算 (DINC(P)、DDEC(P))

Basic | High performance | Process | Redundant | Universal



Ⓓ : 执行 DINC(+1)、DDEC(-1) 的软元件的起始编号 (BIN32 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
Ⓓ	○							--	

功 能

DINC

(1) 对Ⓓ中指定的软元件 (32 位数据) 进行 +1。

(2) 在Ⓓ中指定的软元件的值为 2147483647 时如果执行了 DINC、DINCP，将在Ⓓ中指定的软元件中存储 -2147483648。

DDEC

(1) 对Ⓓ中指定的软元件 (32 位数据) 进行 -1。

(2) 在Ⓓ中指定的软元件的值为 0 时如果执行了 DDEC、DDECP，将在Ⓓ中指定的软元件中存储 -1。

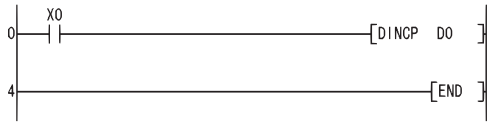
出 错

(1) 在 DINC(P)/DDEC(P) 指令中无运算出错。

程序示例

(1) 以下为 X0 变为 ON 时，对 D0、D1 的数据进行 +1 的程序。

[梯形图模式]

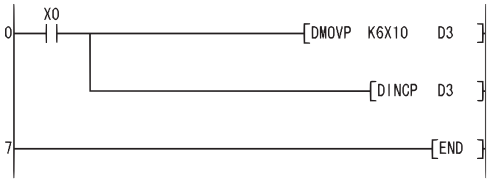


[列表模式]

步	指令	软元件
0	LD	X0
1	DINCP	D0
4	END	

(2) 以下为 X0 变为 ON 时，对设置到 X10 ～ X27 中的数据进行 +1，并将结果存储到 D3、D4 中的程序。

[梯形图模式]

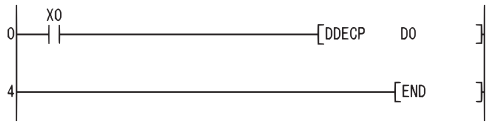


[列表模式]

步	指令	软元件
0	LD	X0
1	DMOVP	K6X10 D3
4	DINCP	D3
7	END	

(3) 以下为 X0 变为 ON 时，对 D0、D1 的数据进行 -1 的程序。

[梯形图模式]

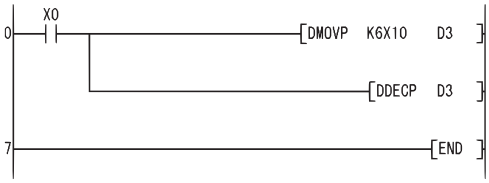


[列表模式]

步	指令	软元件
0	LD	X0
1	DDECP	D0
4	END	

(4) 以下为 X0 变为 ON 时，对设置到 X10 ～ X27 中的数据进行 -1，并将结果存储到 D3、D4 中的程序。

[梯形图模式]



[列表模式]

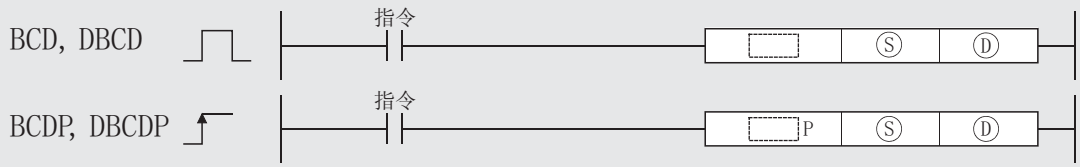
步	指令	软元件
0	LD	X0
1	DMOVP	K6X10 D3
4	DDECP	D3
7	END	

6.3 数据转换指令

6.3.1 BIN 数据→4 位、8 位 BCD 数据的转换 (BCD(P)、DBCD(P))

Basic | High performance | Process | Redundant | Universal

□表示BCD、DBCD等指令符号。



Ⓢ : BIN 数据或者存储 BIN 数据的软元件的起始编号 (BIN16/32 位)。

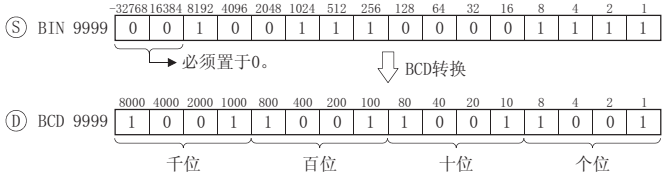
Ⓓ : 存储 BCD 数据的软元件的起始编号 (BCD4/8 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ		○						○	--
Ⓓ		○						--	--

功 能

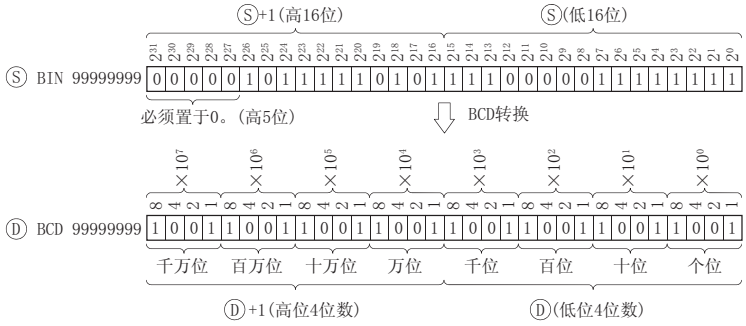
BCD

对Ⓢ中指定的软元件的 BIN 数据 (0 ～ 9999) 进行 BCD 转换后，存储到Ⓓ中指定的软元件中。



DBCD

对Ⓢ中指定的软元件的 BIN 数据 (0 ～ 99999999) 进行 BCD 转换后，存储到Ⓓ中指定的软元件中。



出 错

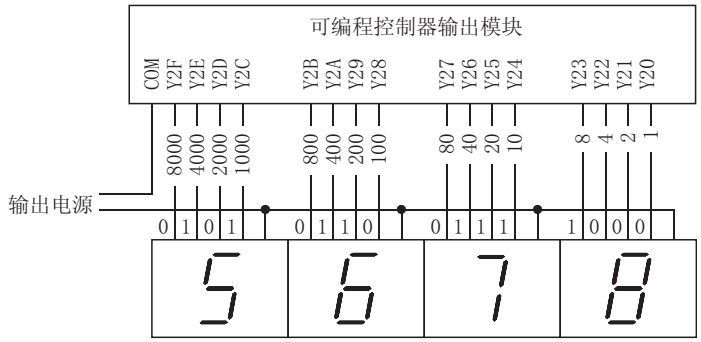
(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· BCD 指令时ⓢ的数据超出了 0 ～ 9999 的范围时。 (出错代码： 4100)

· DBCD 指令时ⓢ+1、ⓢ的数据超出了 0 ～ 99999999 的范围时。 (出错代码： 4100)

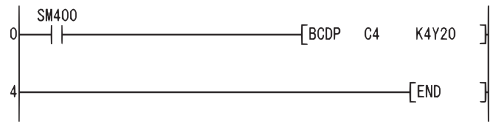
程序示例

(1) 以下为将 C4 的当前值从 Y20 ～ Y2F 输出到 BCD 显示器中的程序。



7因子显示器

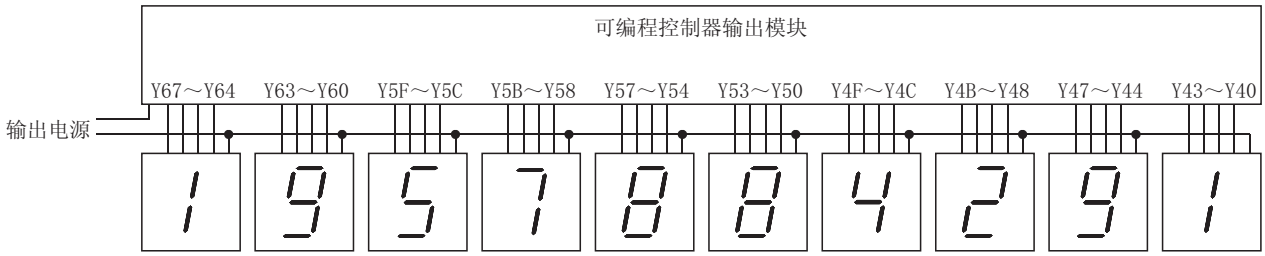
[梯形图模式]



[列表模式]

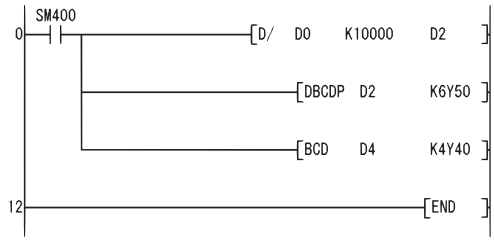
步	指令	软元件	
0	LD	SM400	
1	BCDP	C4	K4Y20
4	END		

(2) 以下为将 D0 ～ D1 的 32 位数据输出到 Y40 ～ Y67 中的程序。



7因子显示器

[梯形图模式]

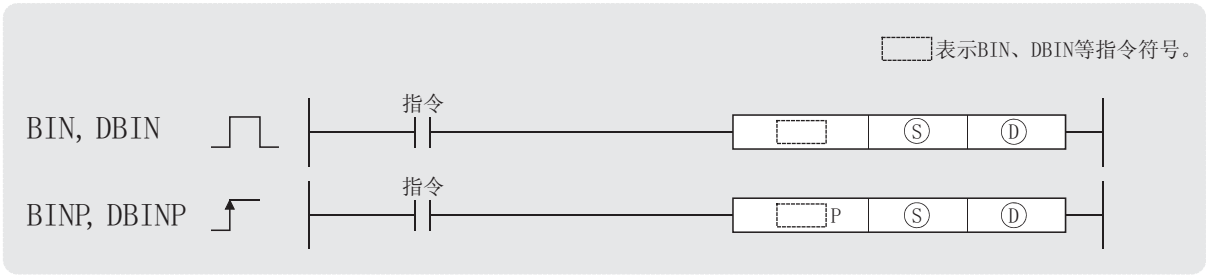


[列表模式]

步	指令	软元件		
0	LD	SM400		
1	D/	D0	K10000	D2
6	DBCDP	D2	K6Y50	
9	BCD	D4	K4Y40	
12	END			

6.3.2 BCD4 位 /8 位→ BIN 数据的转换 (BIN(P)、DBIN(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ : BCD 数据或者存储 BCD 数据的软元件的起始编号 (BCD4/8 位)。

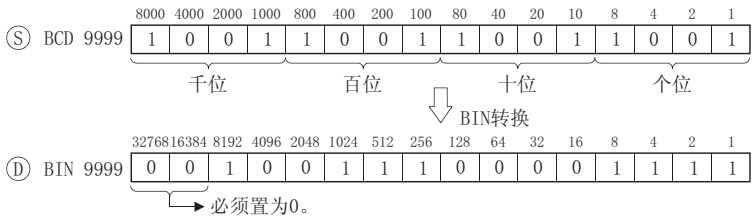
Ⓓ : 存储 BIN 数据的软元件的起始编号 (BIN16/32 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ		○						○	--
Ⓓ		○						--	--

功 能

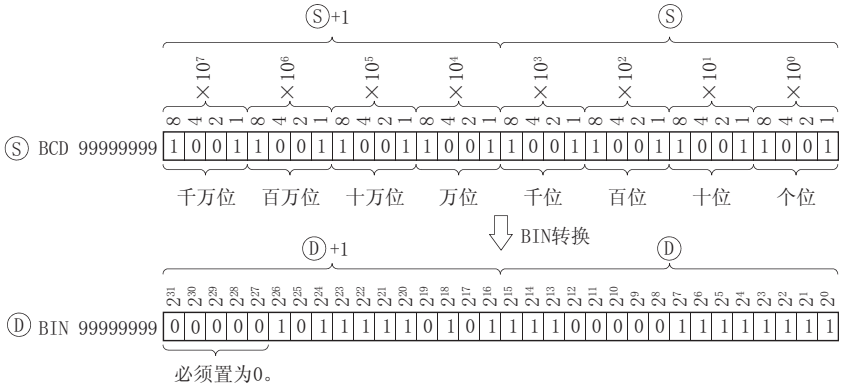
BIN

将Ⓢ中指定的软元件的 BCD 数据 (0 ～ 9999) 转换为 BIN 后，存储到Ⓓ中指定的软元件中。



DBIN

对Ⓢ中指定的软元件的 BCD 数据 (0 ～ 99999999) 进行 BIN 转换后，存储到Ⓓ中指定的软元件中。



出 错

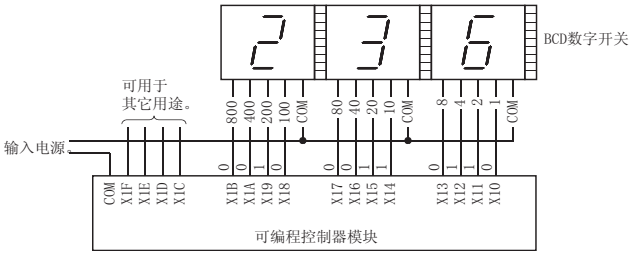
(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· Ⓢ的各位数中存在有超出 0 ～ 9 范围的值时。 (出错代码： 4100)

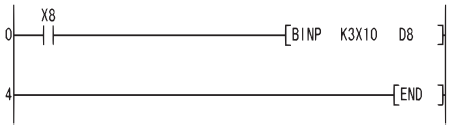
此时，设置了超出范围的数值时，与 SM722 的 ON/OFF 状态无关，不执行指令。
此外，BINP/DBINP 指令的情况下，与有无出错无关，在对指令 (执行条件) 进行 OFF → ON 操作之前，不执行运算。

程序示例

(1) 以下为 X8 变为 ON 时，将 X10 ～ X1B 的 BCD 数据进行 BIN 转换后，存储到 D8 中的程序。



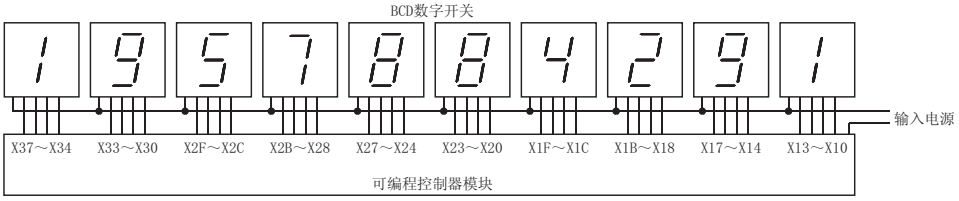
[梯形图模式]



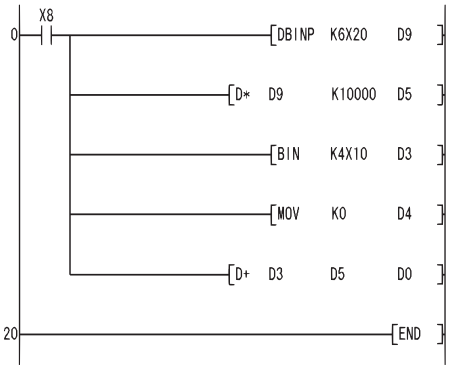
[列表模式]

步	指令	软元件	
0	LD	X8	
1	BINP	K3X10	D8
4	END		

(2) 以下为 X8 变为 ON 时，将 X10 ～ X37 的 BCD 数据进行 BIN 转换后，存储到 D0、D1 中的程序。
(将 X20 ～ X37 的 BCD 数据进行了 BIN 转换后的值与将 X10 ～ X1F 的 BCD 数据进行 BIN 转换后的值进行加法运算。)



[梯形图模式]



[列表模式]

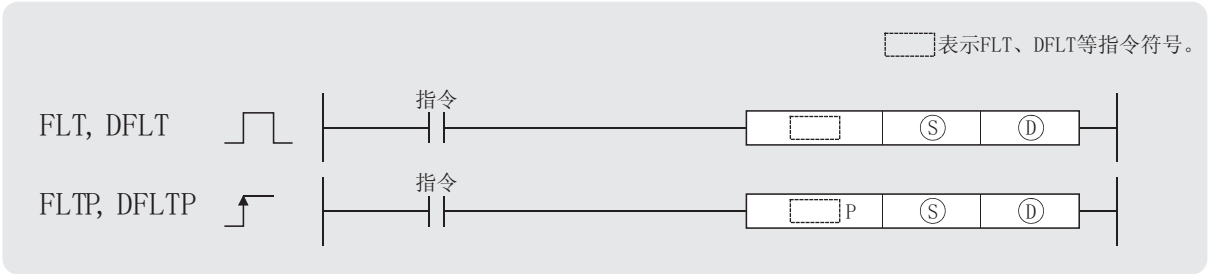
步	指令	软元件		
0	LD	X8		
1	DBINP	K6X20	D9	
4	D*	D9	K10000	D5
9	BIN	K4X10	D3	
12	MOV	K0	D4	
14	D+	D3	D5	D0
20	END			

在 X10 ～ X37 中设置了超过 2147483647 的 BCD 值的情况下，由于超出了 32 位软元件的数值处理范围，因此 D0、D1 中的值将变为负数。

6.3.3 BIN16 位 /32 位数据→浮点数据的转换（单精度）(FLT(P)、DFLT(P))

Ver. Basic | High performance | Process | Redundant | Universal

基本型 QCPU：序列号的前 5 位数为“ 04122 ”以后



Ⓢ ：转换为 32 位浮点数据的整数或者存储整数数据的起始软元件号 (BIN16/32 位)。

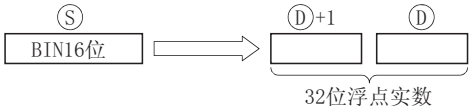
Ⓓ ：存储转换为 32 位浮点数据的起始软元件号 (实数)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○	○		○	○		○		--
Ⓓ	--	○		--	○		--		--

功 能

FLT

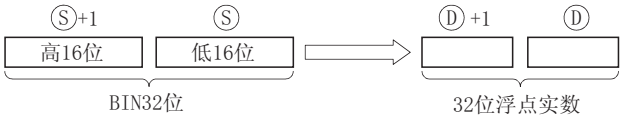
(1) 将Ⓢ中指定的 BIN16 位数据转换为 32 位浮点实数后，存储到Ⓓ中指定编号的软元件中。



(2) Ⓢ中指定的值为 BIN 值且在 -32768 ～ 32767 的范围内。

DFLT

(1) 将Ⓢ中指定的 BIN32 位数据转换为 32 位浮点实数后，存储到Ⓓ中指定编号的软元件中。

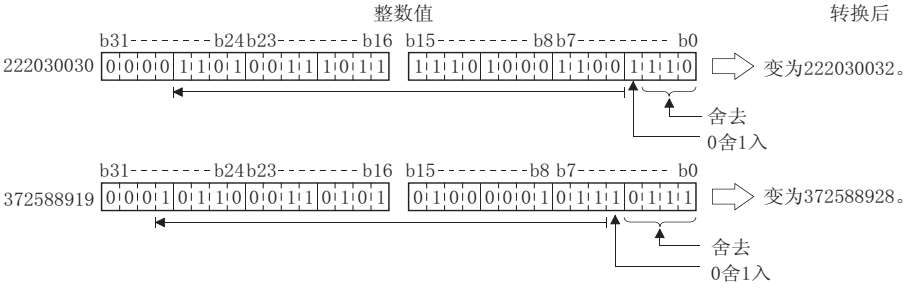


(2) Ⓢ+1、Ⓢ中指定的值为 BIN 值且在 -2147483648 ～ 2147483647 的范围内。

(3) 由于 32 位浮点实数是以 32 位单精度进行处理，2 进制数表示时有效位数为 24 位，10 进制数表示时有效位数约为 7 位数。

因此，整数值超出了 -16777216 ～ 16777215(24 位 BIN 值) 的范围时，转换的值中将产生误差。

将转换结果从整数值的高位起第 25 位进行 0 舍 1 入，第 26 位以后将被舍去。



出　错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

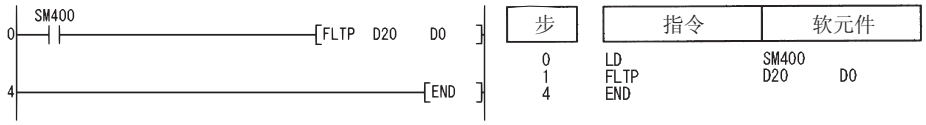
· 运算结果超出了下述范围时。(发生了溢出时。)
(仅通用型 QCPU)

$2^{128} \leqq$ | 运算结果 |　　(出错代码： 4141)

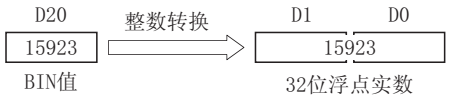
程序示例

(1) 以下为将 D20 的 BIN16 位数据转换为 32 位浮点实数后，存储到 D0、D1 中的程序。

[梯形图模式]　　[列表模式]

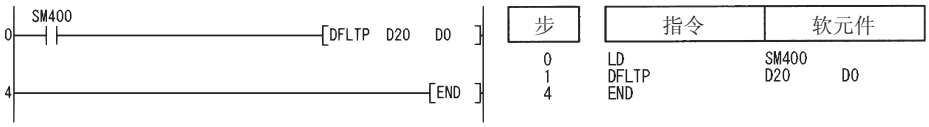


[动作]

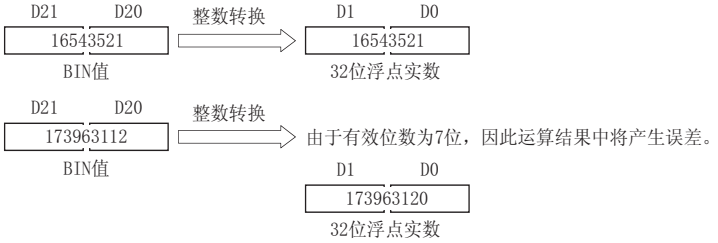


(2) 以下为将 D20、D21 的 BIN32 位数据转换为 32 位浮点实数后，存储到 D0、D1 中的程序。

[梯形图模式]　　[列表模式]



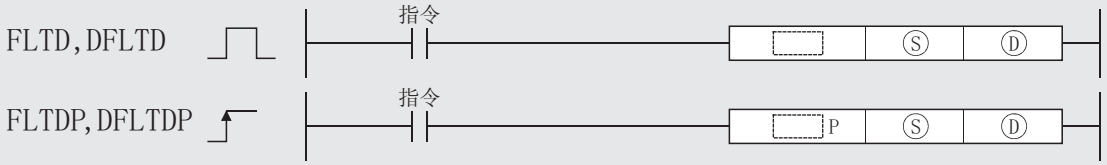
[动作]



6.3.4 BIN16 位 /32 位数据→浮点数据的转换（双精度）(FLTD(P)、DFLTD(P))

Basic Process Redundant Universal

□表示FLTD、DFLTD等指令符号。



Ⓢ ：转换为 64 位浮点数据的整数或者存储整数数据的起始软元件号 (BIN16/32 位)。

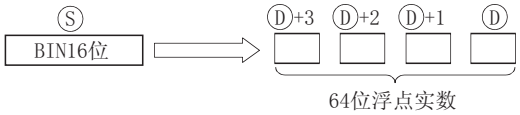
Ⓓ ：存储转换为 64 位浮点数据的起始软元件号 (实数)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	--	○		--			○		--
Ⓓ	--	○		--			--		--

功 能

FLTD

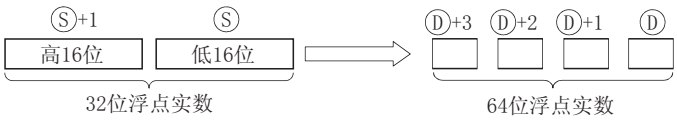
(1) 将Ⓢ中指定的 BIN16 位数据转换为 64 位浮点实数后，存储到Ⓓ中指定编号的软元件中。



(2) Ⓢ中指定的值为 BIN 值且在 -32768 ～ 32767 的范围内。

DFLTD

(1) 将Ⓢ中指定的 BIN32 位数据转换为 64 位浮点实数后，存储到Ⓓ中指定编号的软元件中。



(2) Ⓢ+1、Ⓢ中指定的值为 BIN 值且在 -2147483648 ～ 2147483647 的范围内。

出　错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

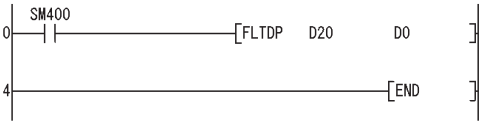
· 运算结果超出了下述范围时。(发生了溢出时。) (仅通用型 QCPU)

$2^{1024} \leqq$ | 运算结果 |　　(出错代码： 4141)

程序示例

(1) 以下为将 D20 的 BIN16 位数据转换为 64 位浮点实数后，存储到 D0 ～ D3 中的程序。

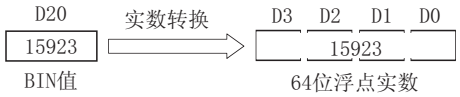
[梯形图模式]



[列表模式]

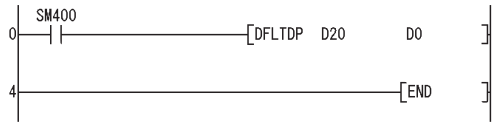
步	指令	软元件	
0	LD	SM400	
1	FLTDP	D20	D0
4	END		

[动作]



(2) 以下为将 D20、D21 的 BIN32 位数据转换为 64 位浮点实数后，存储到 D0 ～ D3 中的程序。

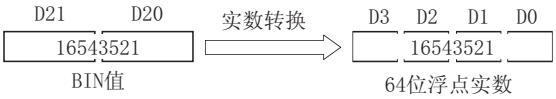
[梯形图模式]



[列表模式]

步	指令	软元件	
0	LD	SM400	
1	DFLTDP	D20	D0
4	END		

[动作]

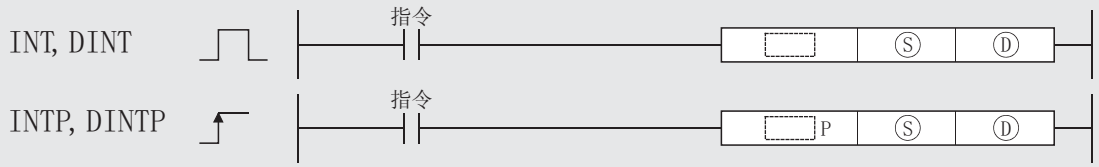


6.3.5 浮点数据→ BIN16 位 /32 位数据的转换（单精度）(INT(P)、DINT(P))

Ver. Basic | High performance | Process | Redundant | Universal

基本型 QCPU：序列号的前 5 位数为“ 04122 ”以后

□表示INT、DINT等指令符号。



Ⓢ ：要转换为 BIN 值的 32 位浮点数据或者存储浮点数据的起始软元件号（实数）。

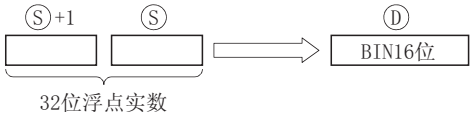
Ⓓ ：存储转换后的 BIN 值的起始软元件号 (BIN16/32 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
Ⓢ	--	○		--	○		--	○	--
Ⓓ	○	○		○	○		○	--	--

功 能

INT

(1) 将Ⓢ中指定的 32 位浮点实数转换为 BIN16 位数据后，存储到Ⓓ中指定编号的软元件中。



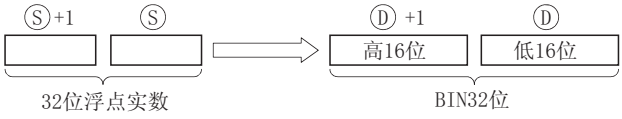
(2) Ⓢ+1、Ⓢ中指定的 32 位浮点实数的可指定范围为 -32768 ～ 32767。

(3) Ⓓ中存储的整数值是以 BIN16 位格式存储的。

(4) 转换后的实数数据的小数点以下第 1 位被四舍五入。

DINT

(1) 将Ⓢ中指定的 32 位浮点实数转换为 BIN32 位数据后，存储到Ⓓ中指定编号的软元件中。



(2) Ⓢ+1、Ⓢ中指定的 32 位浮点实数的可指定范围为 -2147483648 ～ 2147483647。

(3) ⓓ+1、ⓓ中存储的整数值是以 BIN32 位格式存储的。

(4) 转换后的实数数据的小数点以下第 1 位被四舍五入。

出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· 指定软元件的内容超出了下述范围时。(仅通用型 QCPU)

$0, 2^{-126} \leqq |$指定软元件的内容$| < 2^{128}$ (出错代码： 4140)

· 指定软元件的内容为 -0、非正规数、非数、±∞时。
(仅通用型 QCPU) (出错代码： 4140)

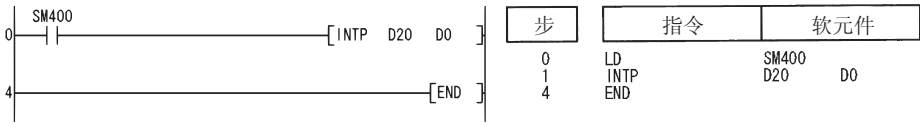
· 使用 INT 指令时，ⓢ中设置的 32 位浮点数据超出了 -32768 ～ 32767 的范围时。
(出错代码： 4100)

· 使用 DINT 指令时，ⓢ中设置的 32 位浮点数据超出了 -2147483648 ～ 2147483647 的范围时。 (出错代码： 4100)

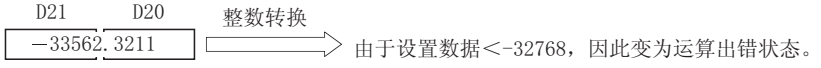
程序示例

(1) 以下为将 D20、D21 的 32 位浮点实数转换为 BIN16 位数据后，存储到 D0 中的程序。

[梯形图模式] [列表模式]

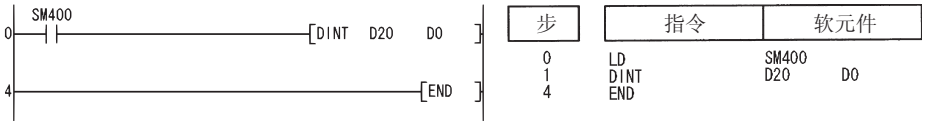


[动作]

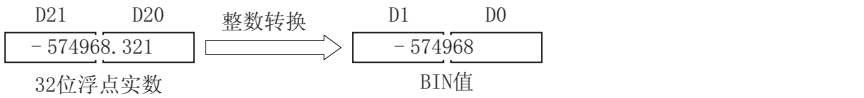


(2) 以下为将 D20、D21 的 32 位浮点实数转换为 BIN32 位数据后，存储到 D0、D1 中的程序。

[梯形图模式] [列表模式]



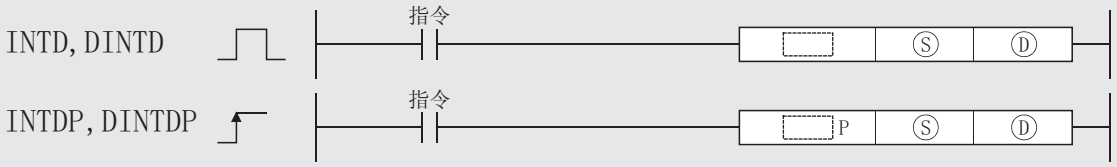
[动作]



6.3.6 浮点数据→ BIN16 位 /32 位数据的转换（双精度）(INTD(P)、DINTD(P))

Basic　High performance　Process　Redundant　Universal

□表示INTD、DINTD等指令符号。



Ⓢ ：要转换为 BIN 值的 64 位浮点实数数据或者存储浮点实数数据的起始软元件号（实数）。

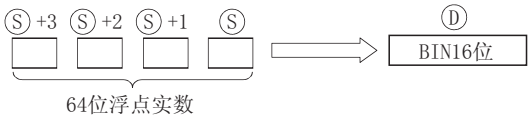
Ⓓ ：存储转换后的 BIN 值的起始软元件号 (BIN16/32 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
Ⓢ	--	○		--			--	○	--
Ⓓ	--	○		--			○	--	--

功　能

INTD

(1) 将Ⓢ中指定的 64 位浮点实数转换为 BIN16 位数据后，存储到Ⓓ中指定编号的软元件中。



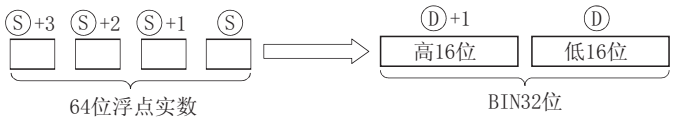
(2) Ⓢ+3、Ⓢ+2、Ⓢ+1、Ⓢ中指定的 64 位浮点实数的可指定范围为 -32768 ～ 32767。

(3) Ⓓ中存储的整数值是以 BIN16 位格式存储的。

(4) 转换后的 64 位浮点实数数据的小数点以下第 1 位被四舍五入。

DINTD

(1) 将Ⓢ中指定的 64 位浮点实数转换为 BIN32 位数据后，存储到Ⓓ中指定编号的软元件中。



(2) Ⓢ+3、Ⓢ+2、Ⓢ+1、Ⓢ中指定的 64 位浮点实数的可指定范围为 -2147483648 ～ 2147483647。

(3) ⓓ+1、ⓓ中存储的整数值是以 BIN32 位格式存储的。

(4) 转换后的 64 位浮点实数数据的小数点以下第 1 位被四舍五入。

出　错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· 指定软元件的内容超出了下述范围时。 (出错代码： 4140)

$0、2^{-1022} \leqq |$ 指定软元件的内容 $| < 2^{1024}$

· 指定软元件的内容为 -0 时。 (出错代码： 4140)

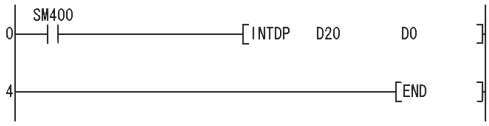
· 使用 INTD 指令时，ⓢ中设置的 64 位浮点数据超出了 -32768 ～ 32767 的范围时。 (出错代码： 4100)

· 使用 DINTD 指令时，ⓢ中设置的 64 位浮点数据超出了 -2147483648 ～ 2147483647 的范围时。 (出错代码： 4100)

程序示例

(1) 以下为将 D20 ～ D23 的 64 位浮点实数转换为 BIN16 位数据后，存储到 D0 中的程序。

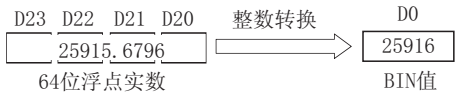
[梯形图模式]



[列表模式]

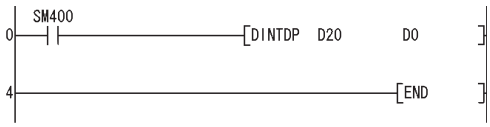
步	指令	软元件	
0	LD	SM400	
1	INTDP	D20	D0
4	END		

[动作]



(2) 以下为将 D20 ～ D23 的 64 位浮点实数转换为 BIN32 位数据后，存储到 D0、D1 中的程序。

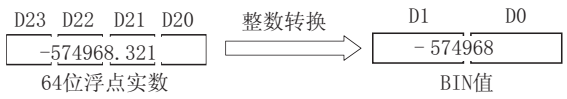
[梯形图模式]



[列表模式]

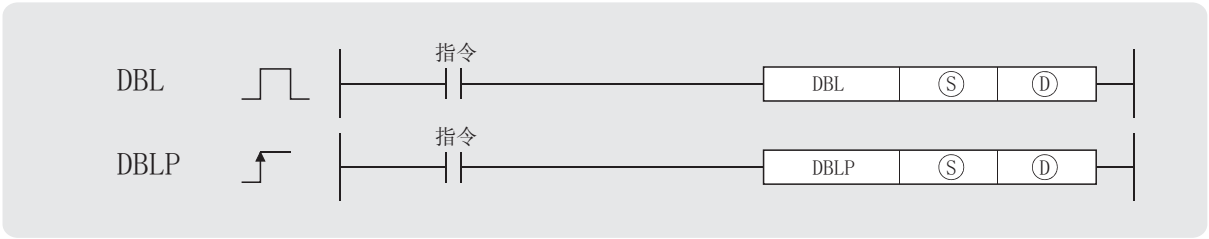
步	指令	软元件	
0	LD	SM400	
1	DINTDP	D20	D0
4	END		

[动作]



6.3.7 BIN16 位数据→ BIN32 位数据的转换 (DBL(P))

Basic | High performance | Process | Redundant | Universal



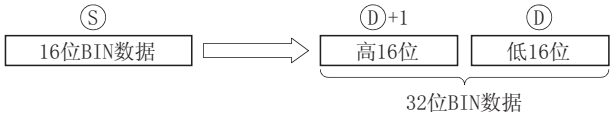
Ⓢ : BIN16 位数据或者存储 BIN16 位数据的起始软元件号 (BIN16 位)。

Ⓓ : 存储转换后的 BIN32 位数据的起始软元件号 (BIN32 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			K、H	
Ⓢ	○							○	--
Ⓓ	○							--	--

功 能

将Ⓢ中指定的 BIN16 位数据转换为带符号 BIN32 位数据后，存储到Ⓓ中指定编号的软元件中。



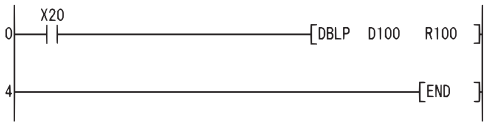
出 错

(1) 在 DBL(P) 指令中无出错。

程序示例

(1) 以下为 X20 变为 ON 时，将 D100 的 BIN16 位数据转换为 BIN32 位数据后，存储到 R100、R101 中的程序。

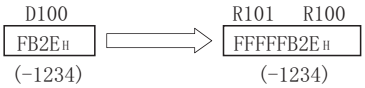
[梯形图模式]



[列表模式]

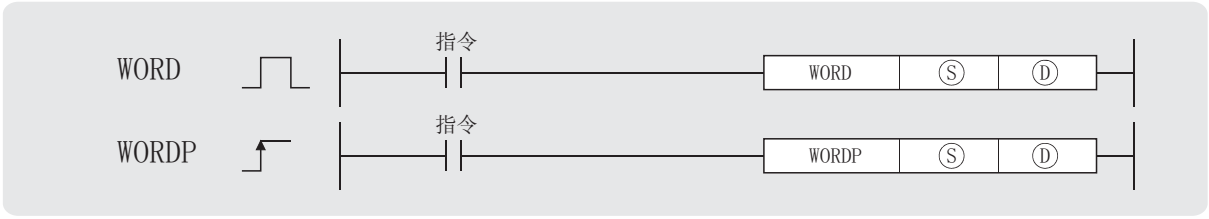
步	指令	软元件
0	LD	X20
1	DBLP	D100 R100
4	END	

[动作]



6.3.8 BIN32 位→ BIN16 位数据的转换 (WORD(P))

Basic | High performance | Process | Redundant | Universal



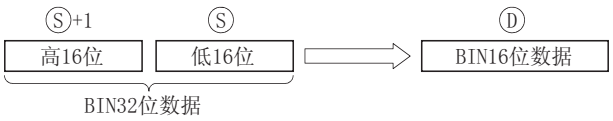
Ⓢ ： BIN32 位数据或者存储 BIN32 位数据的起始软元件号 (BIN32 位)。

Ⓓ ： 存储转换后的 BIN16 位数据的起始软元件号 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ		○						○	--
Ⓓ		○						--	--

功 能

将Ⓢ中指定的 BIN32 位数据转换为带符号 BIN16 位数据后，存储到Ⓓ中指定编号的软元件中。
可指定的范围为 -32768 ～ 32767。



出 错

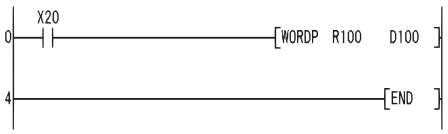
(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。
·Ⓢ+1、Ⓢ中指定的软元件的内容超出了 -32768 ～ 32767 的范围时。

(出错代码： 4100)

程序示例

(1) 以下为 X20 变为 ON 时，将 R100、R101 的 BIN32 位数据转换为 BIN16 位数据后，存储到 D100 中的程序。

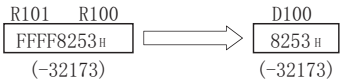
[梯形图模式]



[列表模式]

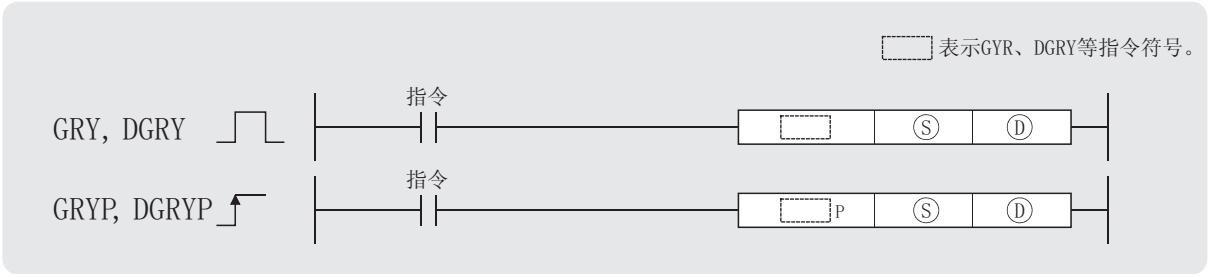
步	指令	软元件	
0	LD	X20	
1	WORDP	R100	D100
4	END		

[动作]



6.3.9 BIN16 位 /32 位数据→格雷码的转换 (GRY(P)、DGRY(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ ：BIN 数据或者存储 BIN 数据的起始软元件号 (BIN16/32 位)。

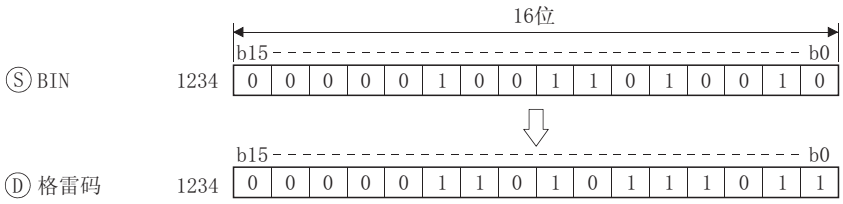
Ⓓ ：存储转换后的格雷码的起始软元件号 (BIN16/32 位)。

设置数据	内部软元件 位	内部软元件 字	R、ZR	J□\□ 位	J□\□ 字	U□\G□	Zn	常数 K、H	其它
Ⓢ		○						○	--
Ⓓ		○						--	--

功 能

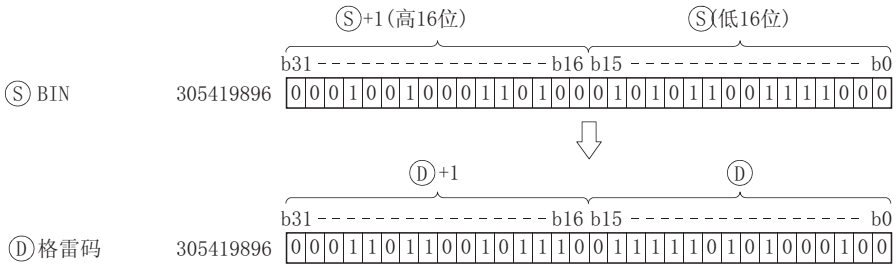
GRY

将Ⓢ中指定的软元件的 BIN16 位数据转换为格雷码后，存储到Ⓓ中指定编号的软元件中。



DGRY

将Ⓢ中指定的软元件的 BIN32 位数据转换为格雷码后，存储到Ⓓ中指定编号的软元件中。



出　错

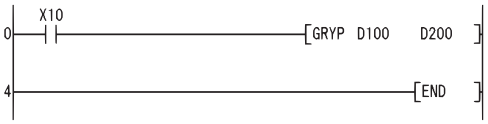
(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· ⓢ的数据为负数时。　　(出错代码： 4100)

程序示例

(1) 以下为 X10 变为 ON 时，将 D100 的 BIN 数据转换为格雷码后，存储到 D200 中的程序。

[梯形图模式]

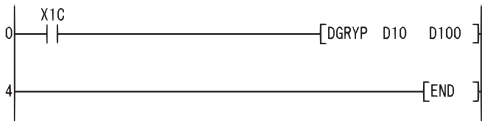


[列表模式]

步	指令	软元件	
0	LD	X10	
1	GRYP	D100	D200
4	END		

(2) 以下为 X1C 变为 ON 时，将 D10、D11 的 BIN 数据转换为格雷码后，存储到 D100、D101 中的程序。

[梯形图模式]

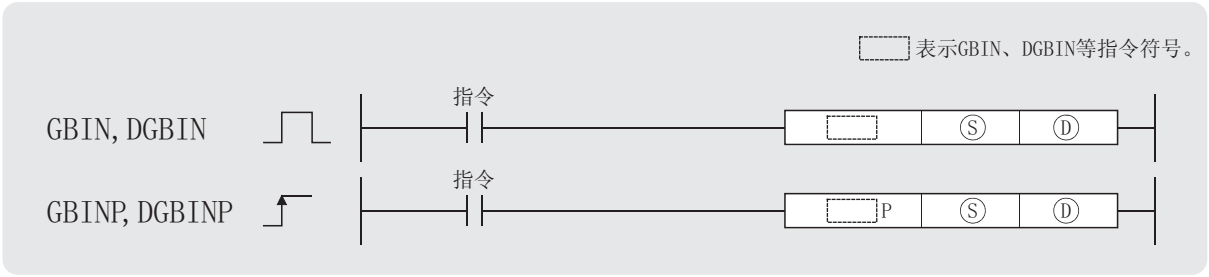


[列表模式]

步	指令	软元件	
0	LD	X1C	
1	DGRYP	D10	D100
4	END		

6.3.10 格雷码→ BIN16 位 /32 位数据的转换 (GBIN(P)、DGBIN(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ ： 格雷码数据或者存储格雷码数据的起始软元件号 (BIN16/32 位)。

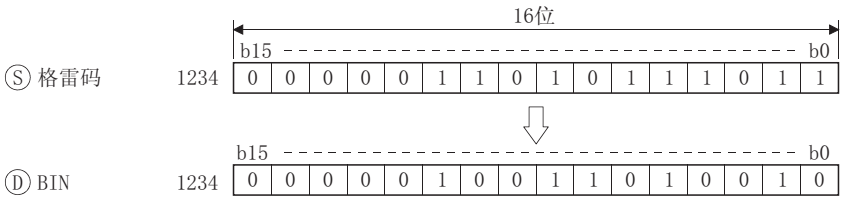
Ⓓ ： 存储转换后的 BIN 值的起始软元件号 (BIN16/32 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ		○						○	--
Ⓓ		○						--	--

功 能

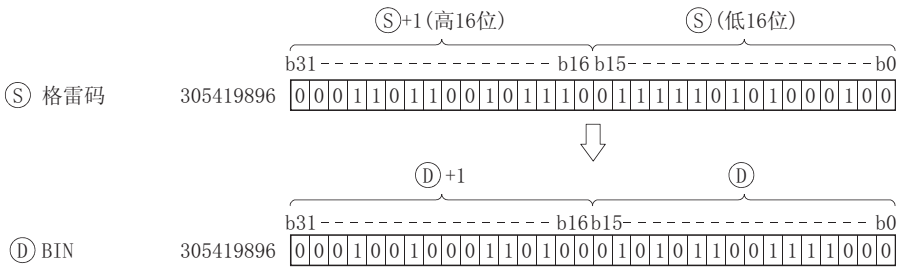
GBIN

将Ⓢ中指定的软元件中存储的格雷码数据转换为 BIN16 位数据后，存储到Ⓓ中指定的软元件中。



DGBIN

将Ⓢ中指定的软元件中存储的格雷码数据转换为 BIN32 位数据后，存储到Ⓓ中指定的软元件中。



出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

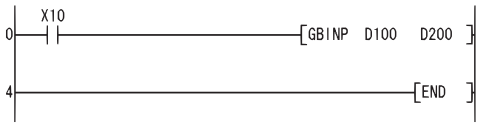
· GBIN 指令时，Ⓢ的数据超出了 0 ～ 32767 的范围时。 (出错代码： 4100)

· DGBIN 指令时，Ⓢ的数据超出了 0 ～ 2147483647 的范围时。 (出错代码： 4100)

程序示例

(1) 以下为 X10 变为 ON 时，将 D100 的格雷码数据转换为 BIN 数据后，存储到 D200 中的程序。

[梯形图模式]

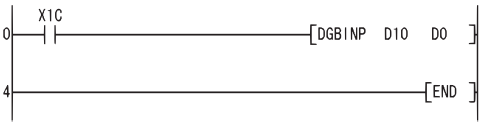


[列表模式]

步	指令	软元件	
0	LD	X10	
1	GBINP	D100	D200
4	END		

(2) 以下为 X1C 变为 ON 时，将 D10、D11 的格雷码数据转换为 BIN 数据后，存储到 D0、D1 中的程序。

[梯形图模式]



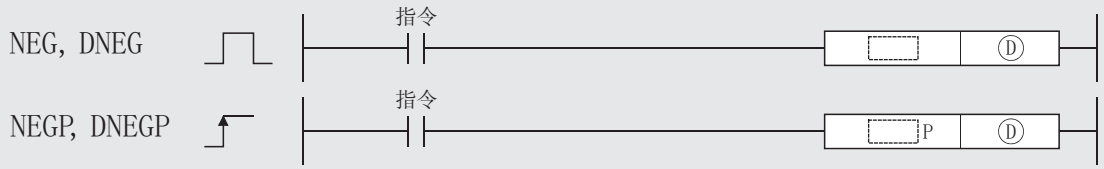
[列表模式]

步	指令	软元件	
0	LD	X1C	
1	DGBINP	D10	D0
4	END		

6.3.11 BIN16 位 /32 位数据的 2 进制补码（符号取反）(NEG(P)、DNEG(P))

Basic | High performance | Process | Redundant | Universal

□表示NEG、DNEG等指令符号。



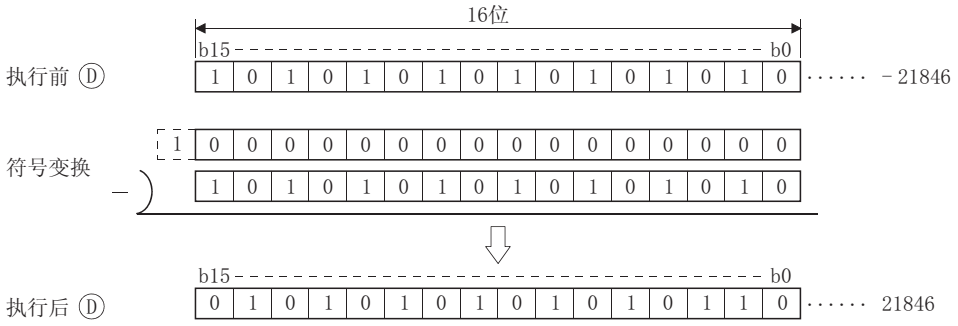
Ⓓ ：存储进行 2 进制补码的数据的起始软元件号 (BIN16/32 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
Ⓓ	○							--	

功 能

NEG

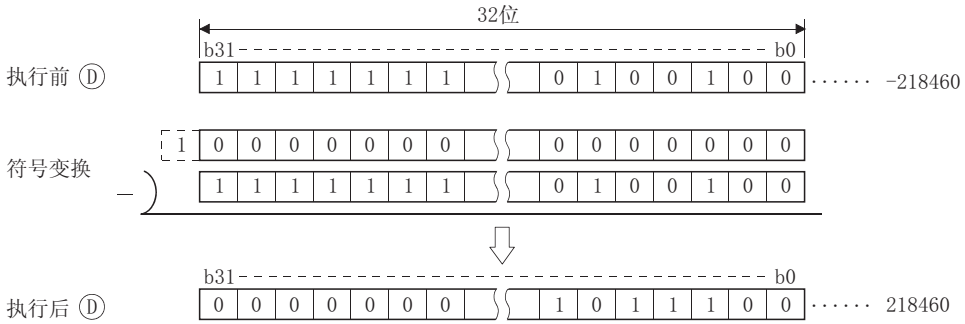
(1) 将Ⓓ中指定的 16 位软元件的符号取反后，存储到Ⓓ中指定的软元件中。



(2) 用于对正负符号进行取反。

DNEG

(1) 将Ⓓ中指定的 32 位软元件的符号取反后，存储到Ⓓ中指定的软元件中。



(2) 用于对正负符号进行取反。

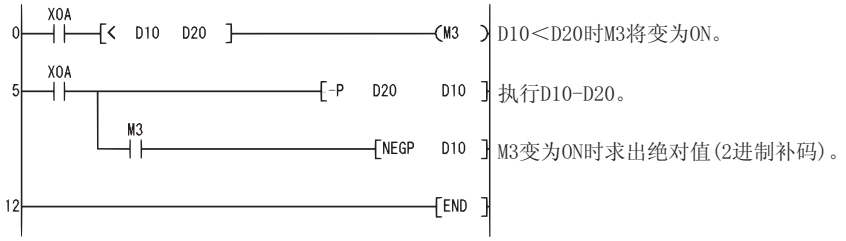
出　错

(1) 在 NEG(P)、DNEG(P) 指令中无出错。

程序示例

(1) 以下为 XA 变为 ON 时，进行 D10-D20 的计算，其结果为负时求出其绝对值的程序。

[梯形图模式]



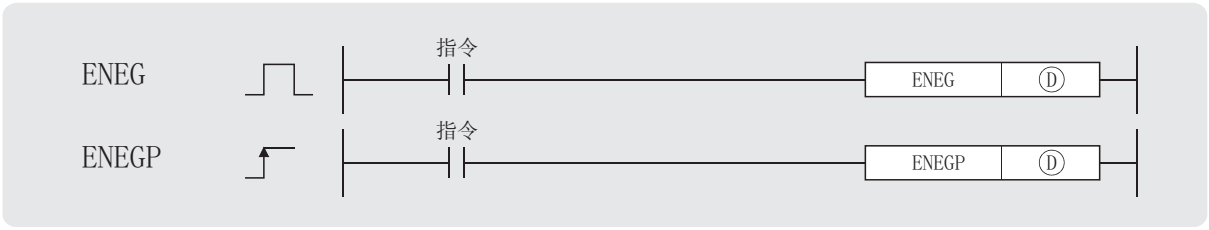
[列表模式]

步	指令	软元件	
0	LD	XOA	
1	AND<	D10	D20
4	OUT	M3	
5	LD	XOA	
6	-P	D20	D10
9	AND	M3	
10	NEGP	D10	
12	END		

6.3.12 浮点数据的符号取反（单精度）(ENEG(P))

Ver. Basic | High performance | Process | Redundant | Universal

基本型 QCPU：序列号的前 5 位数为“ 04122 ”以后。



Ⓓ ：存储进行符号取反的 32 位浮点数据的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
Ⓓ	--	○		--	○		--		

功 能

(1) 将Ⓓ中指定的软元件的 32 位浮点实数数据的符号取反后，存储到Ⓓ中指定的软元件中。

(2) 用于对正负符号进行取反。

出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· 指定软元件的内容不在下述范围时。（仅通用型 QCPU）

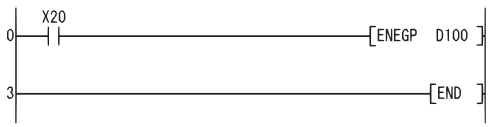
$0、2^{-126} \leqq |$ 指定软元件的内容 $| < 2^{128}$ （出错代码：4140）

· 指定软元件的内容为 -0、非正规数、非数、±∞时。
（仅通用型 QCPU） （出错代码：4140）

程序示例

(1) 以下为 X20 变为 ON 时，将 D100、D101 的 32 位浮点实数数据的符号取反后，存储到 D100、D101 中的程序。

[梯形图模式]



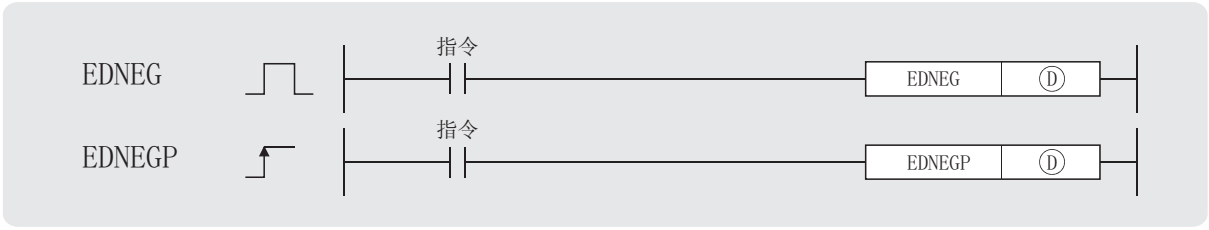
[列表模式]

步	指令	软元件
0	LD	X20
1	ENEGP	D100
3	END	

[动作]

6.3.13 浮点数据的符号取反（双精度）(EDNEG(P))

Basic　High performance　Process　Redundant　Universal



Ⓓ：存储进行符号取反的 64 位浮点数据的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
Ⓓ	--	○		--					

功　能

(1) 将Ⓓ中指定的软元件的 64 位浮点实数数据的符号取反后，存储到Ⓓ中指定的软元件中。

(2) 用于对正负符号进行取反。

出　错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· 指定软元件的内容不在下述范围时。　（出错代码：4140）

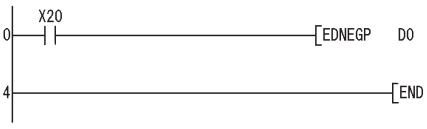
$0、2^{-1022} \leqq |$指定软元件的内容$| < 2^{1024}$

· 指定软元件的内容为 -0 时。　（出错代码：4140）

程序示例

(1) 以下为 X20 变为 ON 时，将 D0 ～ D3 的 64 位浮点实数数据的符号取反后，存储到 D0 ～ D3 中的程序。

[梯形图模式]



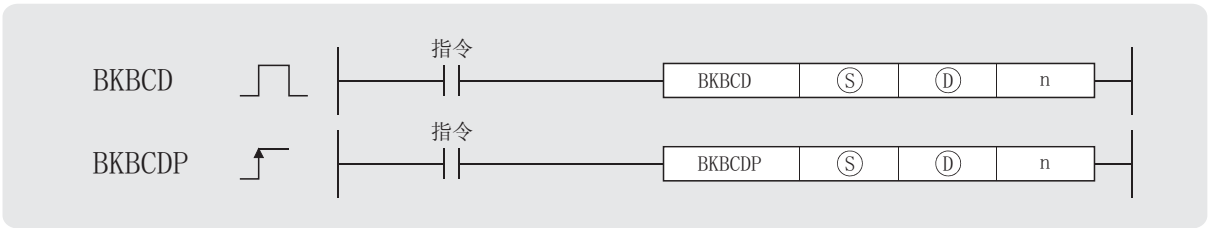
[列表模式]

步	指令	软元件
0	LD	X20
1	EDNEGP	D0
3	END	

[动作]

6.3.14 块 BIN16 位数据→块 BCD4 位数据的转换 (BKBCD(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ ：存储 BIN 数据的软元件的起始编号 (BIN16 位)

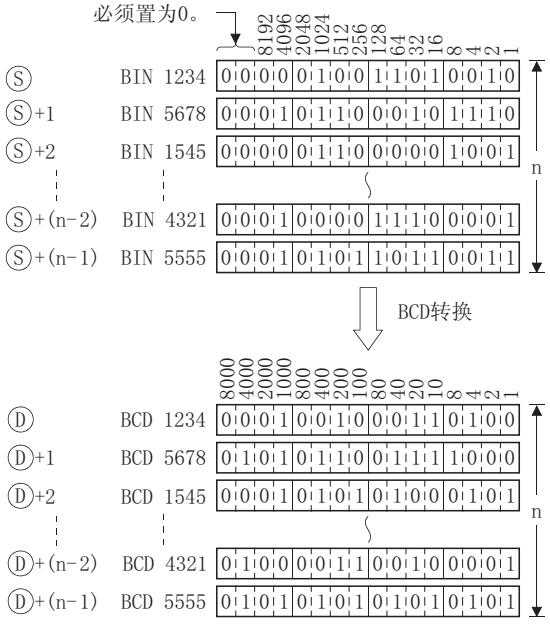
Ⓓ ：存储转换后的 BCD 数据的软元件的起始编号 (BCD4 位)

n ：变量数据数 (BIN16 位)

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	--	○		--					--
Ⓓ	--	○		--					--
n	○	○		--		○			--

功 能

(1) 将从Ⓢ中指定的软元件开始的 n 点的 BIN 数据 (0 ～ 9999) 进行 BCD 转换后，存储到Ⓓ中指定的软元件的后面。



出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· 从Ⓢ、Ⓓ的软元件开始的 n 点的范围超出了相应软元件时。 (出错代码： 4101)

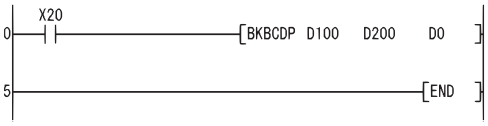
· 从Ⓢ的软元件开始的 n 点的数据超出 0 ～ 9999 的范围时。 (出错代码： 4100)

· Ⓢ、Ⓓ的软元件重复时。 (出错代码： 4101)

程序示例

(1) 以下为 X20 变为 ON 时，将 D100 ～ D102 中存储的 BIN 数据值进行 BCD 转换后，将其结果存储到 D200 后面的程序。

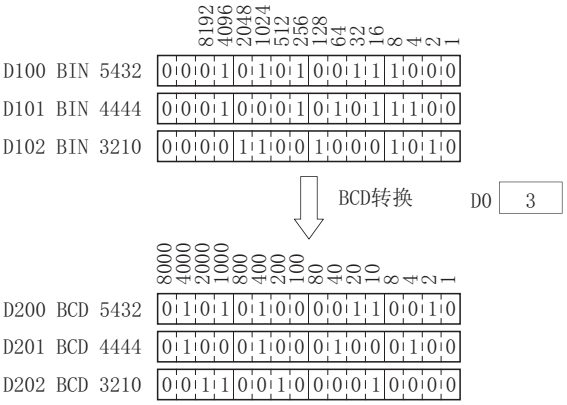
[梯形图模式]



[列表模式]

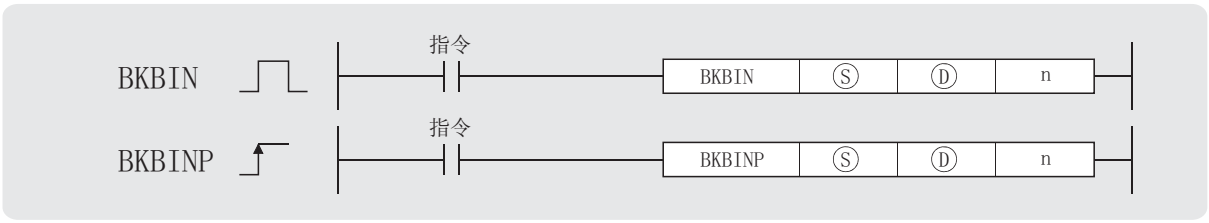
步	指令	软元件		
0	LD	X20		
1	BKBCDP	D100	D200	D0
5	END			

[动作]



6.3.15 块 BCD4 位数据→块 BIN16 位数据的转换 (BKBIN(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ ：存储 BCD 数据的软元件的起始编号 (BCD4 位)。

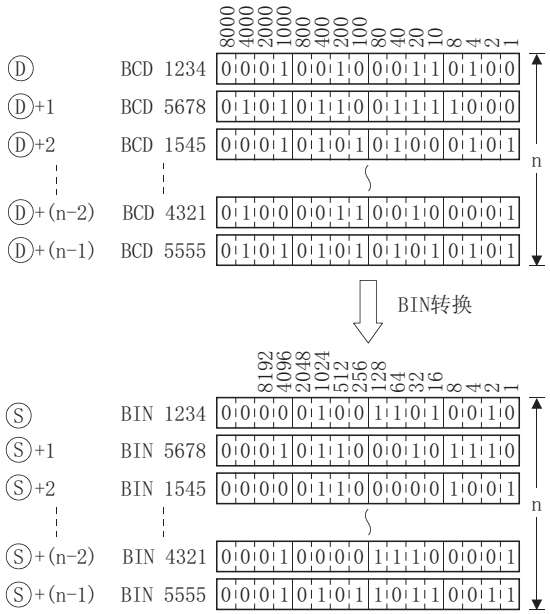
Ⓓ ：存储转换后的 BIN 数据的软元件的起始编号 (BIN16 位)。

n ：变量数据数 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	--	○		--					--
Ⓓ	--	○		--					--
n	○	○		○					--

功 能

(1) 将从Ⓢ中指定的软元件开始的 n 点的 BCD 数据 (0 ～ 9999) 进行 BIN 转换后，存储到Ⓓ中指定的软元件的后面。



出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· 从Ⓢ、Ⓓ的软元件开始的 n 点的范围超出了相应软元件时。 (出错代码： 4101)

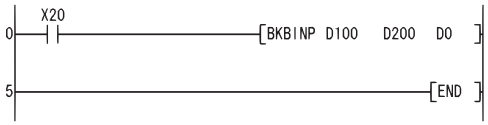
· 从Ⓢ的软元件开始的 n 点的数据超出 0 ～ 9999 的范围时。 (出错代码： 4100)

· Ⓢ、Ⓓ的软元件重复时。 (出错代码： 4101)

程序示例

(1) 以下为 X20 变为 ON 时，将 D100 ～ D102 中存储的 BCD 数据值进行 BIN 转换后，将其结果存储到 D200 后面的程序。

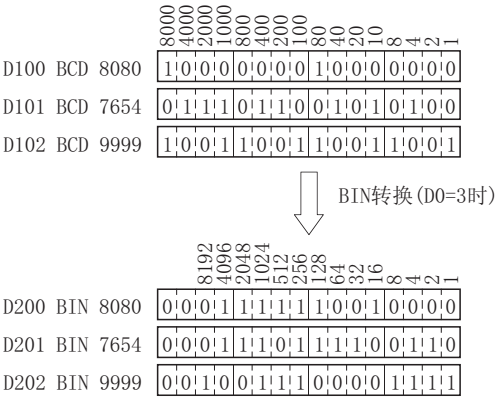
[梯形图模式]



[列表模式]

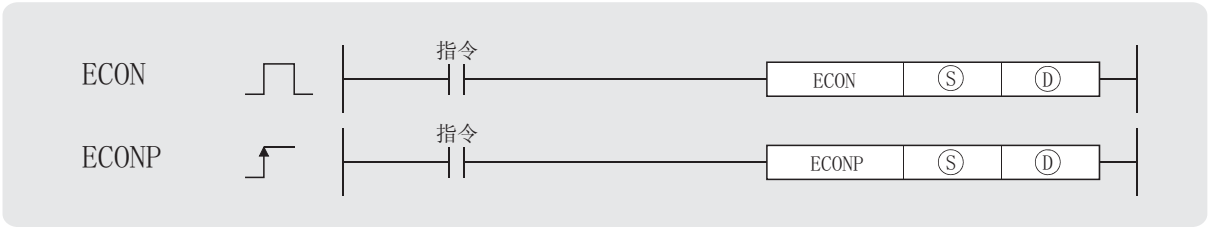
步	指令	软元件		
0	LD	X20		
1	BKBINP	D100	D200	D0
5	END			

[动作]



6.3.16 单精度→双精度转换 (ECON(P))

Basic　High performance　Process　Redundant　Universal



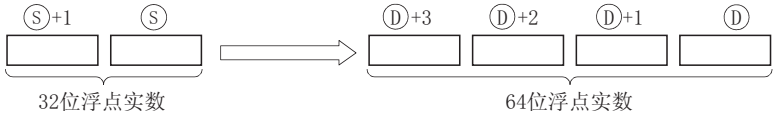
Ⓢ：转换源数据或者存储转换源数据的软元件的起始编号（实数（单精度））。

Ⓓ：存储转换后的数据的软元件的起始编号（实数（双精度））。

设置数据	内部软元件 位	内部软元件 字	R、ZR	J□\□ 位	J□\□ 字	U□\G□	Zn	常数 E	其它
Ⓢ	--	○		--			○		--
Ⓓ	--	○		--			--		--

功　能

将Ⓢ中指定的 32 位浮点实数转换为 64 位浮点实数后，将转换结果存储到Ⓓ中指定的软元件中。



出　错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· 指定软元件的内容不在下述范围时。　　（出错代码：4140）

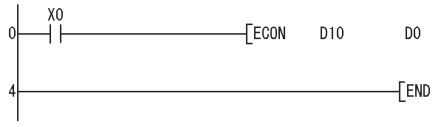
$0、2^{-126} \leqq |$指定软元件的内容$| < 2^{128}$

· 指定软元件的内容为 -0、非正规数、非数、±∞时。　　（出错代码：4140）

程序示例

(1) 以下为 X0 变为 ON 时，将 D10 ～ D11 中的 32 位浮点实数转换为 64 位浮点实数后，输出到 D0 ～ D3 中的程序。

[梯形图模式]

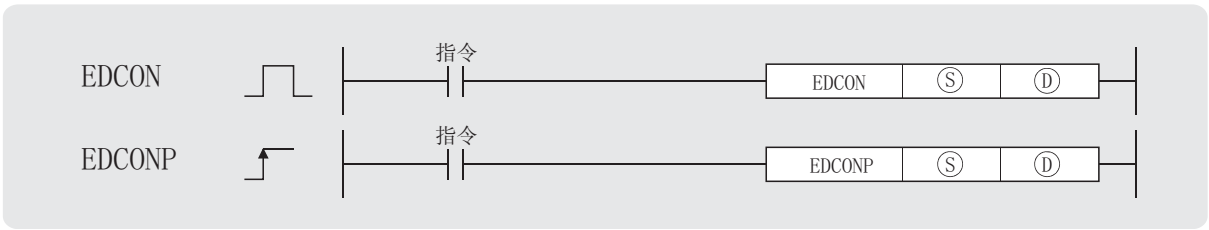


[列表模式]

步	指令	软元件	
0	LD	X0	
1	ECON	D10	D0
4	END		

6.3.17 双精度→单精度转换 (EDCON(P))

Basic High performance Process Redundant Universal



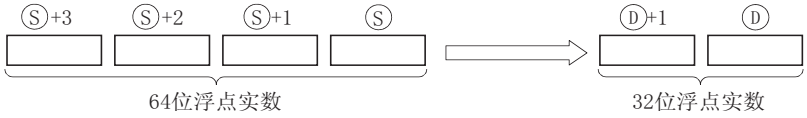
Ⓢ ：转换源数据或者存储转换源数据的软元件的起始编号（实数（双精度））。

Ⓓ ：存储转换后的数据的软元件的起始编号（实数（单精度））。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
Ⓢ	--	○		--			--	○	--
Ⓓ	--	○		--			○	--	--

功 能

将Ⓢ中指定的 64 位浮点实数转换为 32 位浮点实数后，将转换结果存储到Ⓓ中指定的软元件中。



出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· 指定软元件的内容不在下述范围时。 （出错代码： 4140）

$0、2^{-1022} \leqq |$指定软元件的内容$| < 2^{1024}$

· 指定软元件的内容为 -0 时。 （出错代码： 4140）

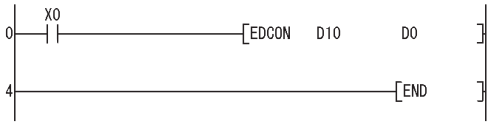
· 转换结果超出了以下范围时。（发生了溢出时。）

$2^{128} \leqq |$转换结果$|$ （出错代码： 4141）

程序示例

(1) 以下为 X0 变为 ON 时，将 D10 ～ D13 中的 64 位浮点实数转换为 32 位浮点实数后，输出到 D0 ～ D1 中的程序。

[梯形图模式]



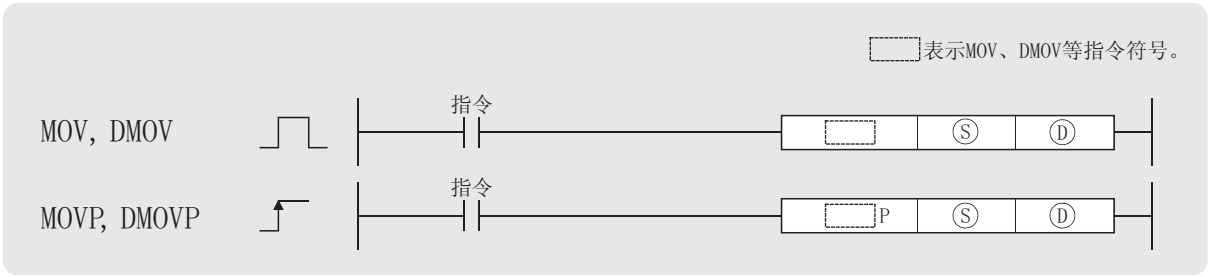
[列表模式]

步	指令	软元件	
0	LD	X0	
1	EDCON	D10	D0
4	END		

6.4 数据传送指令

6.4.1 16 位 /32 位数据传送 (MOV(P)、DMOV(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ : 传送源数据或者存储数据的软元件编号 (BIN16/32 位)。

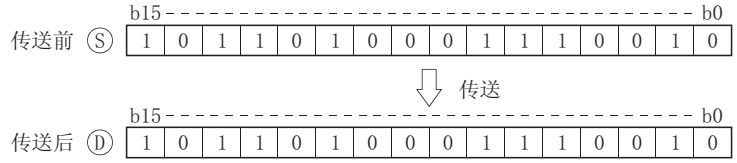
Ⓓ : 传送目标的软元件编号 (BIN16/32 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○							○	--
Ⓓ	○							--	--

功 能

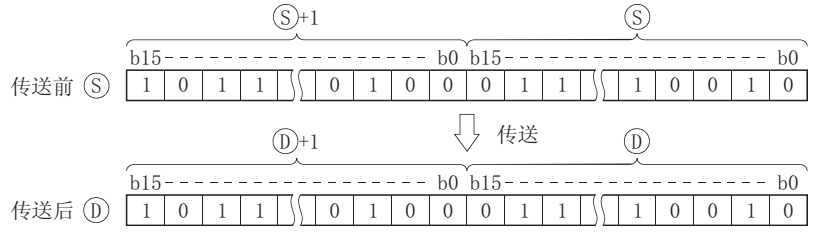
MOV

将Ⓢ中指定的软元件的 16 位数据传送到Ⓓ中指定的软元件中。



DMOV

将Ⓢ中指定的软元件的 32 位数据传送到Ⓓ中指定的软元件中。



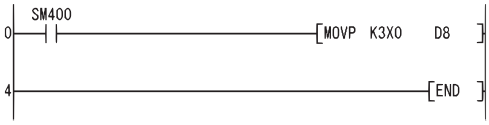
出 错

(1) 在 MOV(P)、DMOV(P) 指令中无运算出错。

程序示例

(1) 以下为将输入 X0 ～ XB 的数据存储到 D8 中的程序。

[梯形图模式]

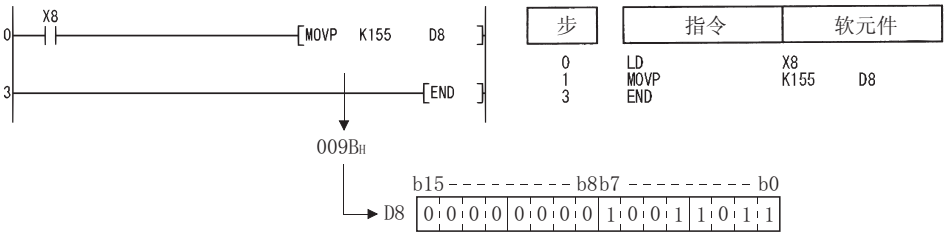


[列表模式]

步	指令	软元件	
0	LD	SM400	
1	MOVP	K3X0	D8
4	END		

(2) 以下为 X8 变为 ON 时，将常数 K155 存储到 D8 中的程序。

[梯形图模式]

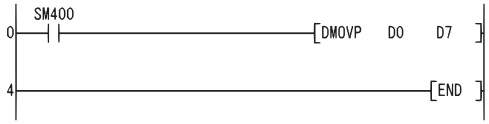


[列表模式]

步	指令	软元件	
0	LD	X8	
1	MOVP	K155	D8
3	END		

(3) 以下为将 D0、D1 的数据存储到 D7、D8 中的程序。

[梯形图模式]

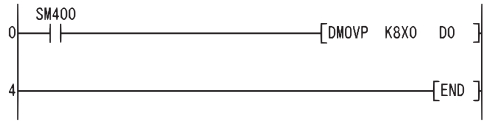


[列表模式]

步	指令	软元件	
0	LD	SM400	
1	DMOVP	D0	D7
4	END		

(4) 以下为将 X0 ～ X1F 的数据存储到 D0、D1 中的程序。

[梯形图模式]



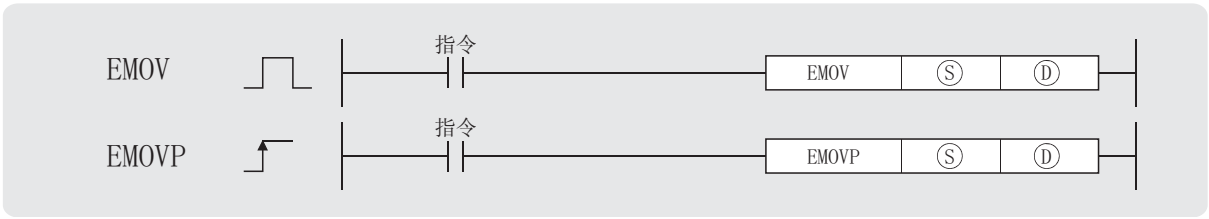
[列表模式]

步	指令	软元件	
0	LD	SM400	
1	DMOVP	K8X0	D0
4	END		

6.4.2 浮点数据传送（单精度）(EMOV(P))

Ver. Basic | High performance | Process | Redundant | Universal

基本型 QCPU：序列号的前 5 位数为“ 04122 ”以后



Ⓢ：传送数据或者存储传送数据的软元件编号（实数）

Ⓓ：存储传送目标数据的软元件编号（实数）

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
Ⓢ	--	○		--	○		--	○	--
Ⓓ	--	○		--	○		--	--	--

功 能

将Ⓢ中指定的软元件中存储的 32 位浮点实数数据传送到Ⓓ中指定的软元件中。

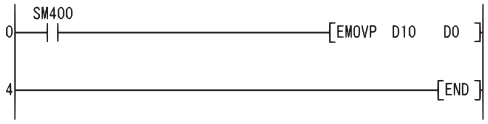
出 错

(1) EMOV(P) 指令中无运算出错。

程序示例

(1) 以下为将 D10、D11 的实数存储到 D0、D1 中的程序。

[梯形图模式]



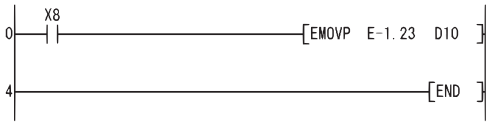
[列表模式]

步	指令	软元件	
0	LD	SM400	
1	EMOVP	D10	D0
4	END		

[动作]

(2) 以下为 X8 变为 ON 时，将实数 -1.23 存储到 D10、D11 中的程序。

[梯形图模式]



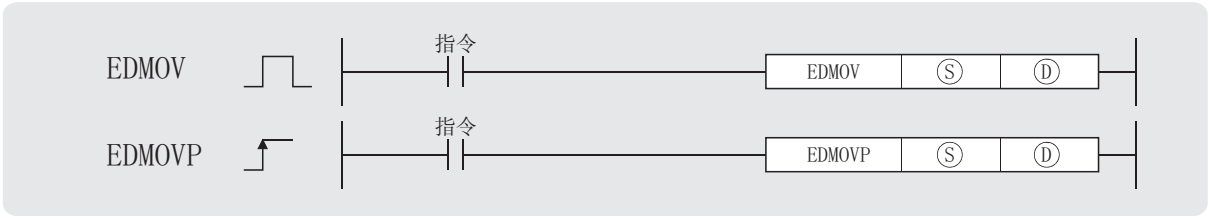
[列表模式]

步	指令	软元件	
0	LD	X8	
1	EMOVP	E-1.23	D10
4	END		

[动作]

6.4.3 浮点数据传送（双精度）(EDMOV(P))

Basic　High performance　Process　Redundant　Universal

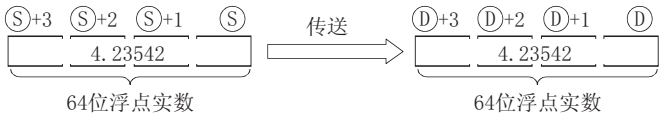


Ⓢ：传送数据或者存储传送数据的软元件编号（实数）。
Ⓓ：存储传送目标数据的软元件编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
Ⓢ	--	○		--				○	--
Ⓓ	--	○		--				--	--

功　能

将Ⓢ中指定的软元件中存储的 64 位浮点实数数据传送到Ⓓ中指定的软元件中。



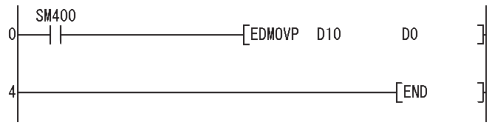
出　错

(1) EDMOV(P) 指令中无运算出错。

程序示例

(1) 以下为将 D10 ～ D13 的 64 位浮点实数存储到 D0 ～ D3 中的程序。

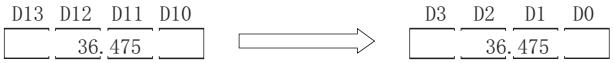
[梯形图模式]



[列表模式]

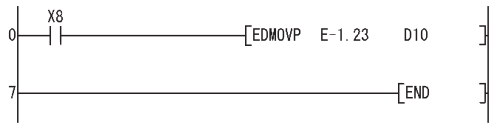
步	指令	软元件	
0	LD	SM400	
1	EDMOVP	D10	D0
4	END		

[动作]



(2) 以下为 X8 变为 ON 时，将实数 -1.23 存储到 D10 ～ D13 中的程序。

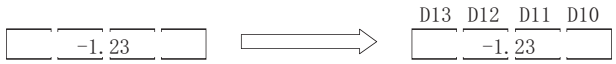
[梯形图模式]



[列表模式]

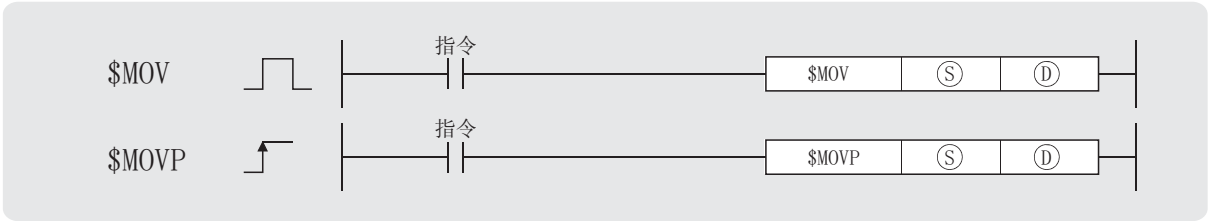
步	指令	软元件	
0	LD	X8	
1	EDMOVP	E-1.23	D10
7	END		

[动作]



6.4.4 字符串传送 ($MOV(P))

Basic | High performance | Process | Redundant | Universal



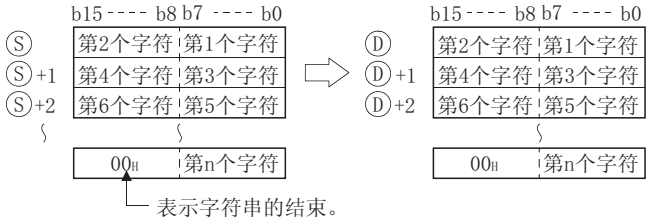
Ⓢ ：传送字符串（最大字符串： 32 个字符）或者存储传送字符串的软元件的起始编号（字符串）。

Ⓓ ：存储传送字符串的软元件的起始编号（字符串）。

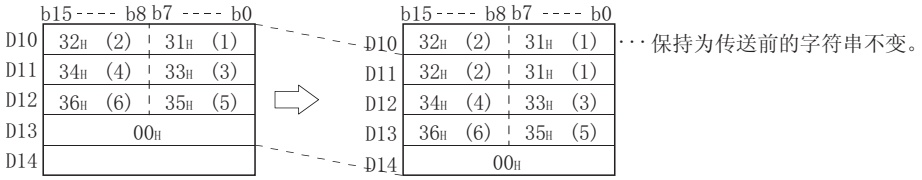
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
Ⓢ	--	○		--				○	--
Ⓓ	--	○		--				--	--

功 能

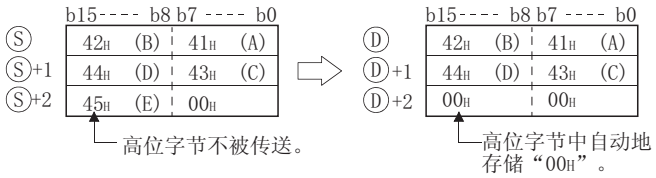
(1) 将Ⓢ中指定的字符串数据传送到Ⓓ中指定的软元件后面。
在字符串传送中，对Ⓢ中指定的用 “ ”（双引号）围住的字符串或者从Ⓢ中指定的软元件号开始至存储了 “ 00H ”的软元件号为止的字符串进行一次传送。



(2) 即使在存储要传送的字符串数据的软元件范围 (Ⓢ～Ⓢ+n) 与存储已传送的字符串数据的软元件范围 (Ⓓ～Ⓓ+n) 相互重叠的情况下，也将正常进行处理。
将存储在 D10 ～ D13 中的字符串数据传送到 D11 ～ D14 中时的情况如下所示：



(3) Ⓢ+n 的低位字节中存储了 “ 00H ” 时，Ⓓ+n 的高位字节、低位字节中均将存储 “ 00H ”。



(4) Ⓢ中指定了汉字等 2 字节数据时，将被转换为移位 JIS 码。
如果执行了 $MOV 指令，Ⓓ中存储的高位字节与低位字节将被相互换位。
(参阅程序示例 (3))

出 错

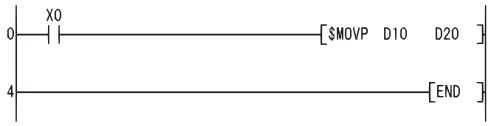
(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- Ⓢ中指定的软元件号以后至相应软元件号为止之间不存在 “ 00H ” 时。
(出错代码 : 4101)
- Ⓓ中指定的软元件号以后至相应软元件的最终软元件号为止的点数无法容纳全部的字符串时。 (出错代码 : 4101)
- Ⓢ的字符串超过了 16383 个字符时。 (出错代码 : 4101)

程序示例

(1) 以下为 X0 变为 ON 时，将 D10 ～ D12 中存储的字符串数据传送到 D20 ～ D22 中的程序。

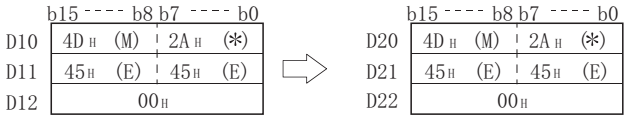
[梯形图模式]



[列表模式]

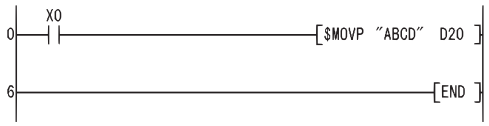
步	指令	软元件	
0	LD	X0	
1	$MOVP	D10	D20
4	END		

[动作]



(2) 以下为 X0 变为 ON 时，将字符串 “ ABCD ” 传送到 D20 ～ D21 中的程序。

[梯形图模式]

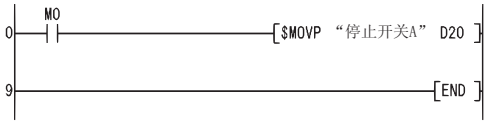


[列表模式]

步	指令	软元件	
0	LD	X0	
1	$MOVP	"ABCD"	D20
6	END		

(3) 以下为 X0 变为 ON 时，将字符串 “ 停止开关 A ” 传送到 D20 ～ D24 中的程序。

[梯形图模式]



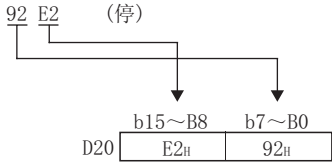
[列表模式]

步	指令	软元件	
0	LD	M0	
1	$MOVP	“停止开关A”	D20
9	END		

[动作]

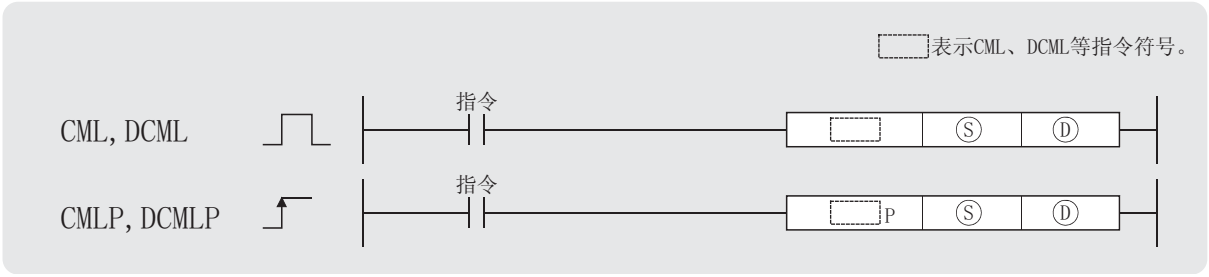
	b15～B8	b7～B0	
D20	7E$_H$ *1	92$_H$ *1	(停)
D21	7E$_H$	8E$_H$	(止)
D22	B2$_H$	BD$_H$	(开)
D23	C1$_H$	AF$_H$	(关)
D24	00$_H$	41$_H$	(A)

*1: 2 个字节的情况下，存储的高位字节与低位字节将被相互换位。
停止开关 A 的 “ 停 ” 的移位码为 92E2$_H$，通过执行了 $MOV 指令，D20 中将存储 E292$_H$。



6.4.5 16 位 /32 位数据否定传送 (CML(P)、DCML(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ : 取反数据或者存储取反数据的软元件编号 (BIN16/32 位)。

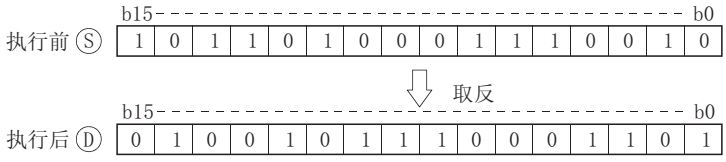
Ⓓ : 存储取反结果的软元件编号 (BIN16/32 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ				○				○	--
Ⓓ				○				--	--

功 能

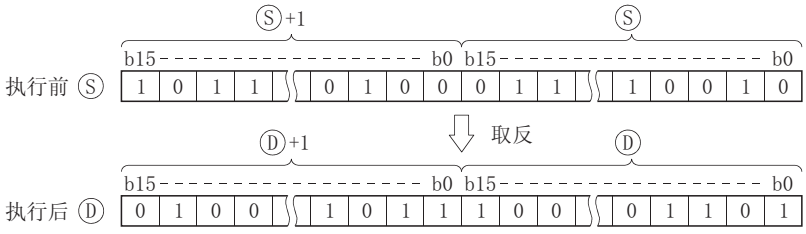
CML

对Ⓢ中指定的 16 位数据进行逐位取反， 并将其结果传送到Ⓓ中指定的软元件中。



DCML

对Ⓢ中指定的 32 位数据进行逐位取反， 并将其结果传送到Ⓓ中指定的软元件中。



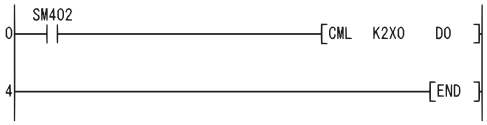
出 错

(1) CML(P)、DCML(P) 指令中无运算出错。

程序示例

(1) 以下为对 X0 ～ X7 的数据进行取反后，将其传送到 D0 中的程序。

[梯形图模式]

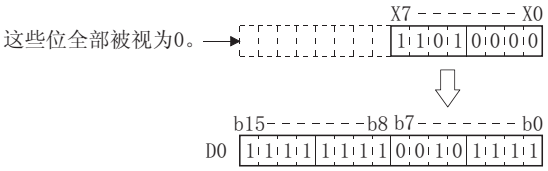


[列表模式]

步	指令	软元件	
0	LD	SM402	
1	CML	K2X0	D0
4	END		

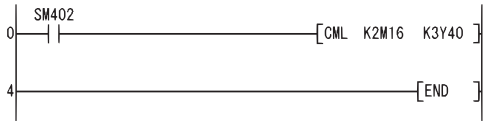
[动作]

Ⓢ 的位数＜Ⓓ的位数时



(2) 以下为对 M16 ～ M23 的数据进行取反后，将其传送到 Y40 ～ Y47 中的程序。

[梯形图模式]

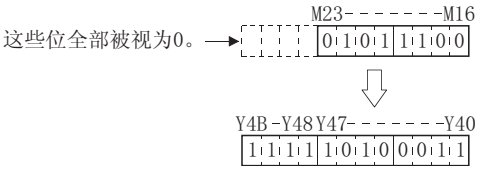


[列表模式]

步	指令	软元件	
0	LD	SM402	
1	CML	K2M16	K3Y40
4	END		

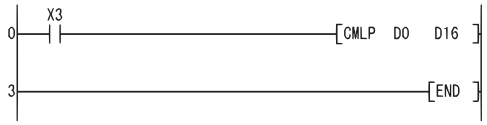
[动作]

Ⓢ 的位数＜Ⓓ的位数时



(3) 以下为 X3 变为 ON 时，对 D0 的数据进行取反后，将其传送到 D16 中的程序。

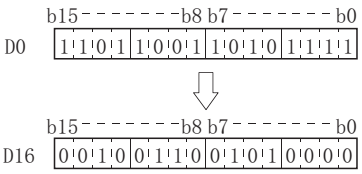
[梯形图模式]



[列表模式]

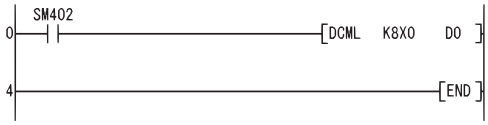
步	指令	软元件	
0	LD	X3	
1	CMLP	D0	D16
3	END		

[动作]



(4) 以下为对 X0 ～ X1F 的数据进行取反后，将其传送到 D0、D1 中的程序。

[梯形图模式]

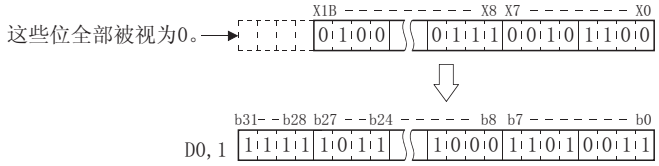


[列表模式]

步	指令	软元件	
0	LD	SM402	
1	DCML	K8X0	D0
4	END		

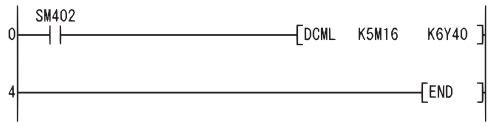
[动作]

Ⓢ 的位数＜Ⓓ的位数时



(5) 以下为对 M16 ～ M35 的数据进行取反后，将其传送到 Y40 ～ Y63 中的程序。

[梯形图模式]

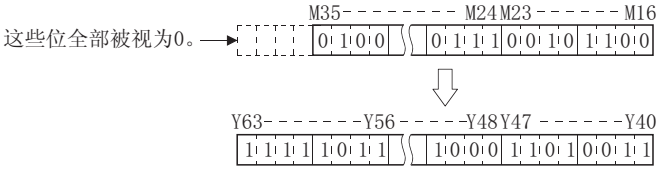


[列表模式]

步	指令	软元件	
0	LD	SM402	
1	DCML	K5M16	K6Y40
4	END		

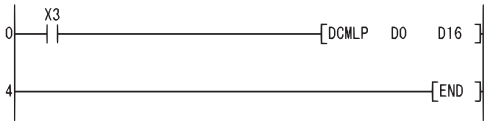
[动作]

Ⓢ 的位数＜Ⓓ的位数时



(6) 以下为 X3 变为 ON 时，对 D0、D1 的数据进行取反后，将其传送到 D16、D17 中的程序。

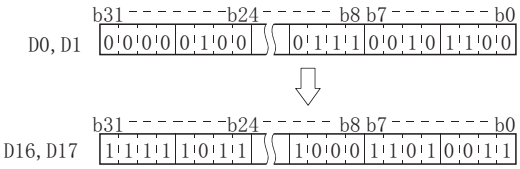
[梯形图模式]



[列表模式]

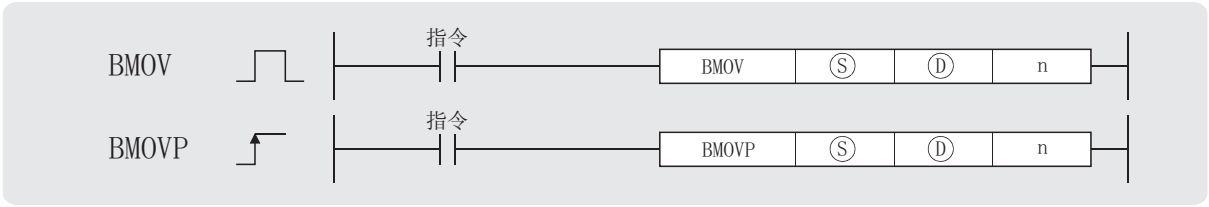
步	指令	软元件	
0	LD	X3	
1	DCMLP	D0	D16
4	END		

[动作]



6.4.6 块 16 位数据传送 (BMOV(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ ： 传送数据或者存储传送数据的软元件的起始编号 (BIN16 位)。

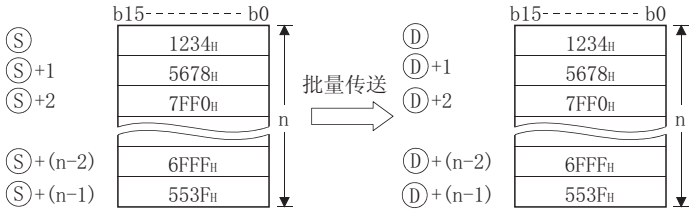
Ⓓ ： 存储传送目标软元件的起始编号 (BIN16 位)。

n ： 传送数 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ			○				--		--
Ⓓ			○				--		--
n			○				○		--

功 能

(1) 将Ⓢ中指定的软元件开始的 n 点的 16 位数据批量传送到Ⓓ中指定的软元件开始的 n 点中。



(2) 传送源与传送目标的软元件重复时，也可进行传送。
传送到软元件号小的一方的情况下，从Ⓢ开始传送；传送到软元件号大的一方的情况下，从Ⓢ+(n－1) 开始传送。
但是，从 R 传送到 ZR 或者从 ZR 传送到 R 时，应注意不要使下述 ZR 与 R 的各自的传送范围重叠。
从 R 传送到 R 以及从 ZR 传送到 ZR 时不会有问题。

· ZR 的传送范围 ((指定的 ZR 起始号) ～ (指定的 ZR 起始号 + 传送数 -1))

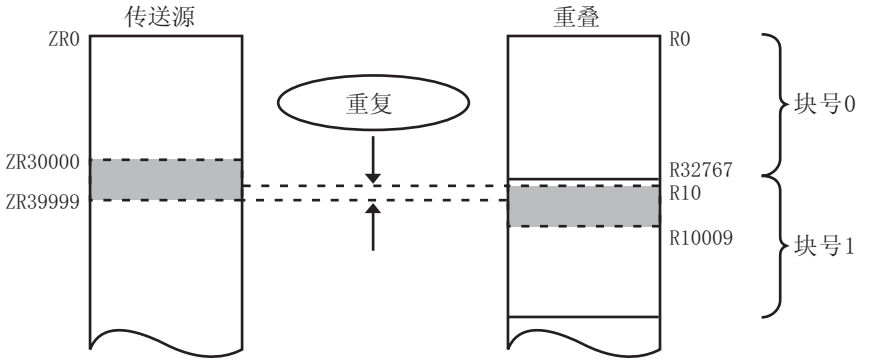
· R 的传送范围 ((指定的 R 起始号 + 文件寄存器块号× 32768) ～
(指定的 R 起始号 + 文件寄存器块号× 32768+ 传送数 -1))

例 将 10000 点的数据从传送源 ZR30000 传送到传送目标块号 1 的 R10 中时，传送范围重叠。

· ZR 的传送范围→ (30000) ～ (30000 + 10000 − 1) → (30000) ～ (39999)

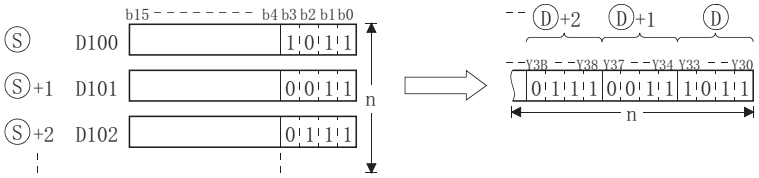
· R 的传送范围→ (10 + (1 × 32768)) ～ (10 + (1 × 32768) + 10000 − 1) → (32778) ～ (42777)

由于 32778 起至 39999 为止的范围重叠，因此无法进行正常的值传送。



(3) 在Ⓢ为字软元件而Ⓓ为位软元件的情况下，字软元件的对象为位软元件的位数指定中指定的位数。

Ⓓ中指定了 K1Y30 时，Ⓢ中指定的字软元件的低 4 位将成为对象。



(4) Ⓢ、Ⓓ中指定了位软元件时，必须将Ⓢ、Ⓓ的位数设置为相同。

(5) Ⓢ、Ⓓ中使用链接直接软元件及智能功能模块软元件时，只能指定Ⓢ或Ⓓ中的一个。

(6) 软元件范围检查的执行有无选择

通过软元件范围检查禁止标志 (SM237)，可以选择在执行 BMOV 指令时是否进行软元件范围检查。(仅在子集条件成立时)

SM237 变为 ON 时，不对Ⓢ～Ⓢ+(n-1)、Ⓓ～Ⓓ+(n-1) 进行是否在软元件范围内的检查。

注意事项

SM237 变为 ON 时，不要进行以下访问：

· 变址修饰目标超出了软元件范围的访问

· Ⓓ～Ⓓ+(n-1) 跨越了软元件范围边界的访问 *1

· 在未进行文件寄存器设置状态下至文件寄存器的访问

· 对不存在多 CPU 高速通信区软元件的区域进行的访问

*1: 请参阅 DFMOV 指令有关内容。

要 点

SM237 只能使用下述 CPU 模块：

· 序列号的前 5 位数为 “ 10012 ” 以后的通用型 QCPU

出 错

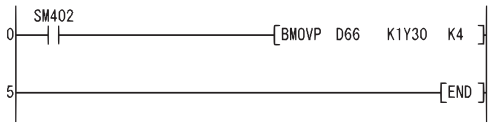
(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 从Ⓢ、Ⓓ开始的 n 点的软元件范围超出了相应软元件时。 (出错代码： 4101)

程序示例

(1) 以下为将 D66 ～ D69 的低 4 位数据以 4 点为单位输出到 Y30 ～ Y3F 中的程序。

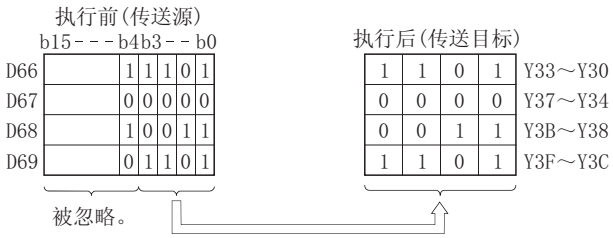
[梯形图模式]



[列表模式]

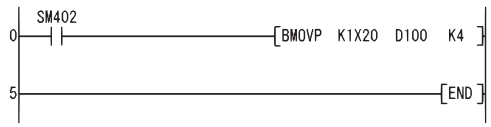
步	指令	软元件		
0	LD	SM402		
1	BMOVP	D66	K1Y30	K4
5	END			

[动作]



(2) 以下为将 X20 ～ X2F 的数据以 4 点为单位输出到 D100 ～ D103 中的程序。

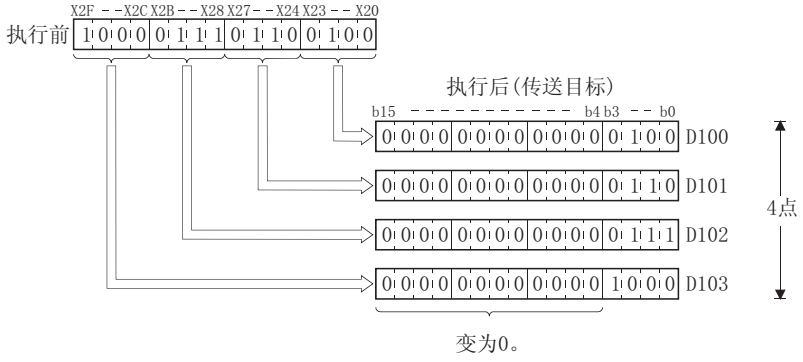
[梯形图模式]



[列表模式]

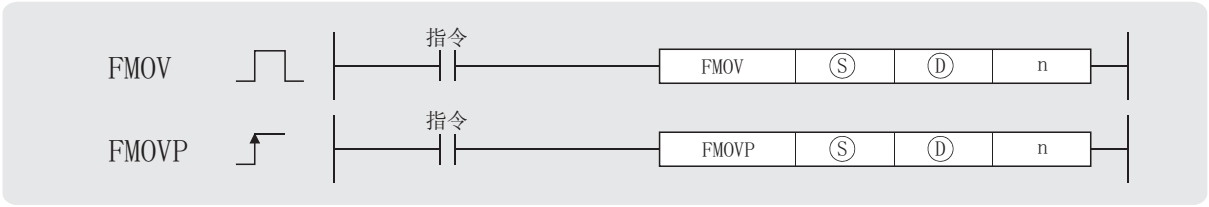
步	指令	软元件		
0	LD	SM402		
1	BMOVP	K1X20	D100	K4
5	END			

[动作]



6.4.7 相同 16 位数据块传送 (FMOV(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ ：传送数据或者存储传送数据的软元件的起始编号 (BIN16 位)。

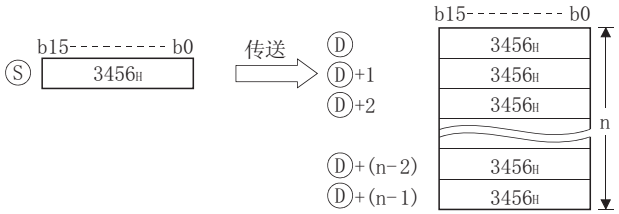
Ⓓ ：传送目标的的软元件的起始编号 (BIN16 位)。

n ：传送数 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○						○		--
Ⓓ	○						--		--
n	○						○		--

功 能

(1) 将Ⓢ中指定的软元件的 16 位数据传送到从Ⓓ中指定的软元件开始的 n 点中。



(2) 在Ⓢ为字软元件而Ⓓ为位软元件的情况下，Ⓢ的字软元件的对象为位软元件的位数指定中指定的位数。

Ⓓ中指定了 K1Y30 时，Ⓢ中指定的字软元件的低 4 位将成为对象。

(3) Ⓢ、Ⓓ中指定了位软元件时，必须将Ⓢ、Ⓓ的位数设置为相同。

(4) 软元件范围检查的执行有无选择

通过软元件范围检查禁止标志 (SM237)，可以选择在执行 FMOV 指令时是否进行软元件范围检查。(仅在子集条件成立时)

SM237 变为 ON 时，不对Ⓓ～Ⓓ+(n-1) 进行是否在软元件范围内的检查。

关于 SM237 的详细内容，请参阅附录 3 的特殊继电器一览表。

注意事项

SM237 变为 ON 时，不要进行以下访问：
· 变址修饰目标超出了软元件范围的访问
· ⓓ～ⓓ+(n-1) 跨越了软元件范围边界的访问 *1
· 在未进行文件寄存器设置状态下至文件寄存器的访问
· 对不存在多 CPU 高速通信区软元件的区域进行的访问
*1：请参阅 DFMOV 指令有关内容。

要 点

SM237 只能使用下述 CPU 模块：
· 序列号的前 5 位数为 “ 10012 ” 以后的通用型 QCPU

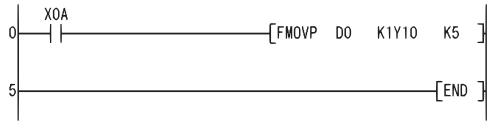
出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。
· 从ⓓ开始的 n 点的软元件范围超出了相应软元件时。 (出错代码： 4101)

程序示例

(1) 以下为 XA 变为 ON 时，将 D0 的低 4 位数据以 4 点为单位输出到 Y10 ～ Y23 中的程序。

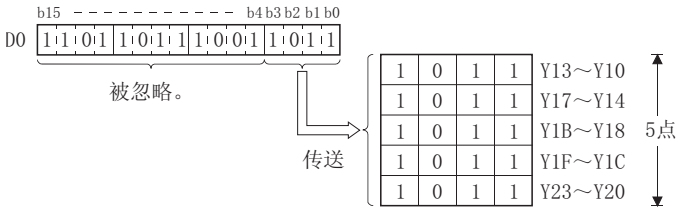
[梯形图模式]



[列表模式]

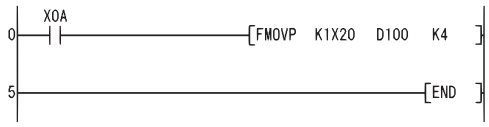
步	指令	软元件		
0	LD	XOA		
1	FMOVP	D0	K1Y10	K5
5	END			

[动作]



(2) 以下为 XA 变为 ON 时，将 X20 ～ X23 的数据输出到 D100 ～ D103 中的程序。

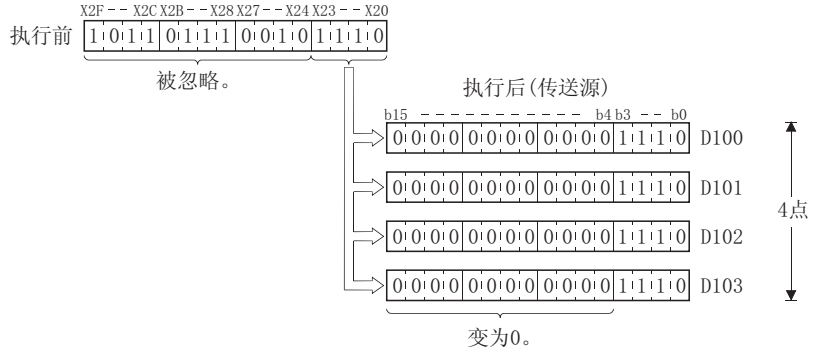
[梯形图模式]



[列表模式]

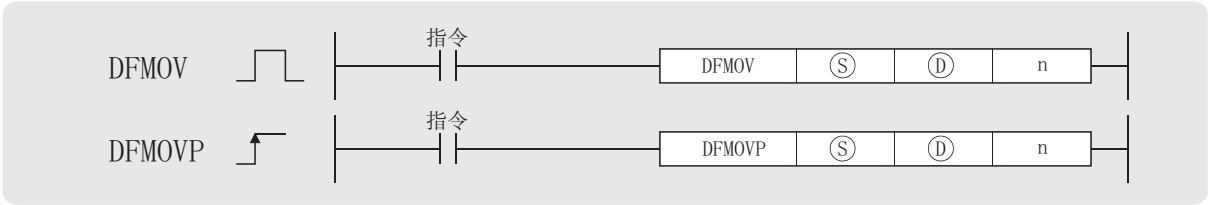
步	指令	软元件		
0	LD	X0A		
1	FMOVP	K1X20	D100	K4
5	END			

[动作]



6.4.8 相同 32 位数据块传送 (DFMOV(P))

· QnU(D)(H)CPU：序列号的前 5 位数为“ 10102 ”以后
· QnUDE(H)CPU：序列号的前 5 位数为“ 10102 ”以后

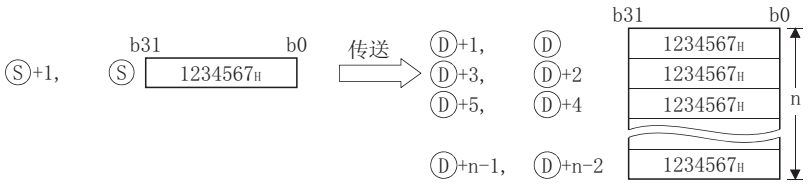


Ⓢ ：传送数据或者存储传送数据的软元件的起始编号 (BIN32 位)。
Ⓓ ：传送目标的的软元件的起始编号 (BIN32 位)。
n ：传送数 (BIN16 位)。

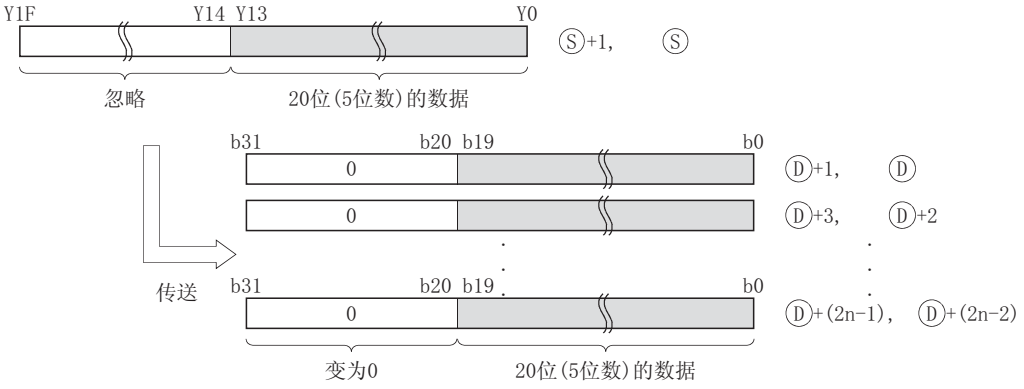
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ			○				○		--
Ⓓ			○				--		--
n			○				○		--

功 能

(1) 将Ⓢ中指定的软元件的 32 位数据传送到从Ⓓ中指定的软元件开始的 n 点中。



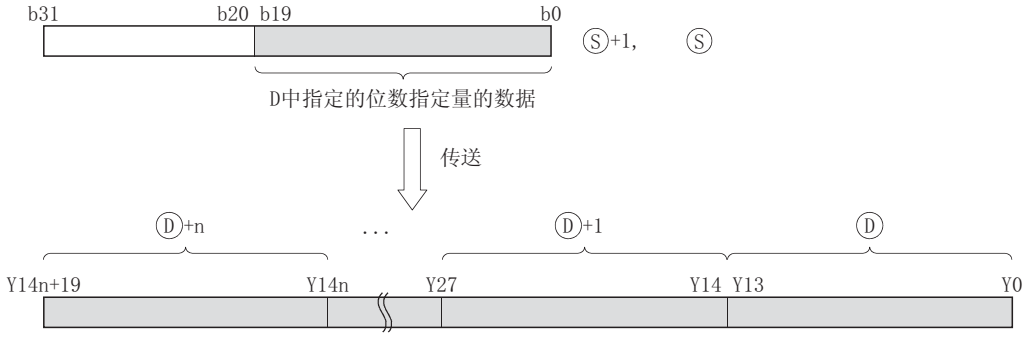
(2) 在Ⓢ中进行了位数指定时，将只按位数指定量进行数据传送。
在Ⓢ中指定了 K5Y0 时，Ⓢ的字软元件的低 20 位 (5 位数) 将成为对象。



(3) 在ⓓ中进行了位数指定时，将只按ⓓ中进行的位数指定量进行数据传送。

在ⓓ中指定了 K5Y0 时，ⓢ的字软元件的低 20 位将成为对象。

ⓢ、ⓓ二者中均进行了位数指定时，与位数无关，将只按ⓓ中指定的位数指定量进行数据传送



(4) n 中指定的值为 0 时将执行无处理。

(5) 通过软元件范围检查禁止标志 (SM237)，可以选择在执行 DFMOV 指令时是否进行软元件范围检查。(仅在子集条件成立时)

出　错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

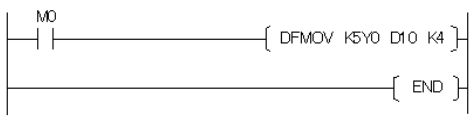
· n 中指定的数据为负数时。 (出错代码 : 4100)

· 传送的数据点数 n 超过了ⓓ的软元件范围时。 (出错代码 : 4101)

程序示例

(1) 以下为 M0 变为 ON 时，将 Y0 ～ Y13(20 位) 的数据存储到 D10 ～ D17 中的程序。

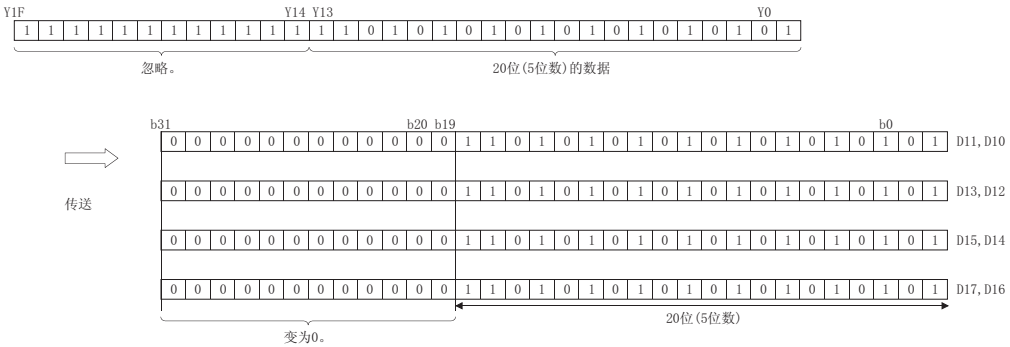
[梯形图模式]



[列表模式]

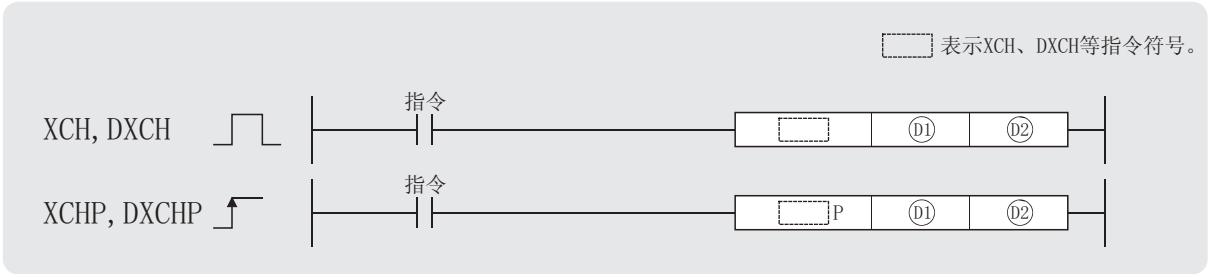
步	指令	软元件
0	LD	M0
1	DFMOV	K5Y0 D10 K4
5	END	

[动作]



6.4.9 16 位 /32 位数据交换 (XCH(P)、DXCH(P))

Basic | High performance | Process | Redundant | Universal



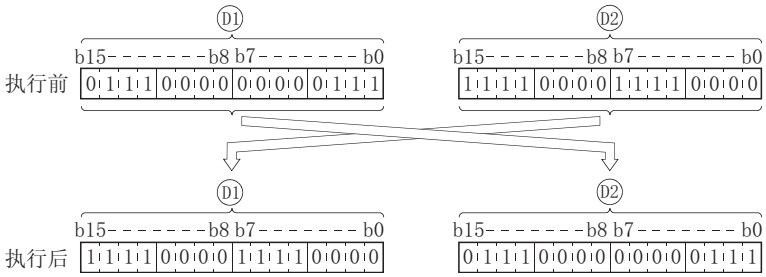
(D1)、(D2) ：存储交换数据的软元件的起始编号 (BIN16/32 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
(D1)				○				--	
(D2)				○				--	

功 能

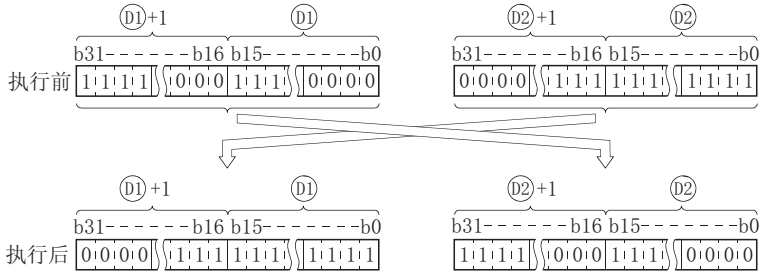
XCH

对(D1)、(D2)的 16 位数据进行交换。



DXCH

对(D1)+1、(D1)与(D2)+1、(D2)的 32 位数据进行交换。



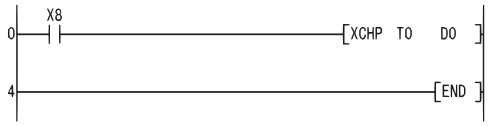
出　错

(1) 在 XCH(P)、DXCH(P) 指令中无运算出错。

程序示例

(1) 以下为 X8 变为 ON 时，将 T0 的当前值与 D0 的内容进行交换的程序。

[梯形图模式]

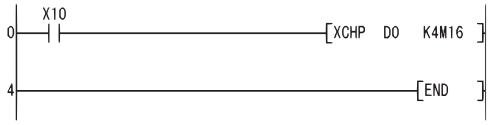


[列表模式]

步	指令	软元件	
0	LD	X8	
1	XCHP	T0	D0
4	END		

(2) 以下为 X10 变为 ON 时，将 D0 的内容与 M16 ～ M31 的数据进行交换的程序。

[梯形图模式]

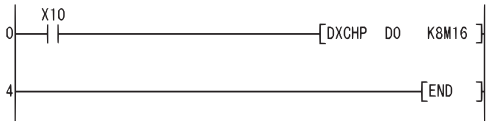


[列表模式]

步	指令	软元件	
0	LD	X10	
1	XCHP	D0	K4M16
4	END		

(3) 以下为 X10 变为 ON 时，将 D0、D1 的内容与 M16 ～ M47 的数据进行交换的程序。

[梯形图模式]

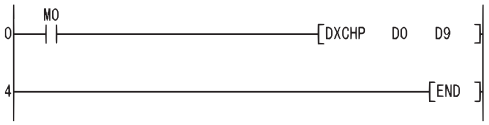


[列表模式]

步	指令	软元件	
0	LD	X10	
1	DXCHP	D0	K8M16
4	END		

(4) 以下为 M0 变为 ON 时，将 D0、D1 的内容与 D9、D10 的数据进行交换的程序。

[梯形图模式]

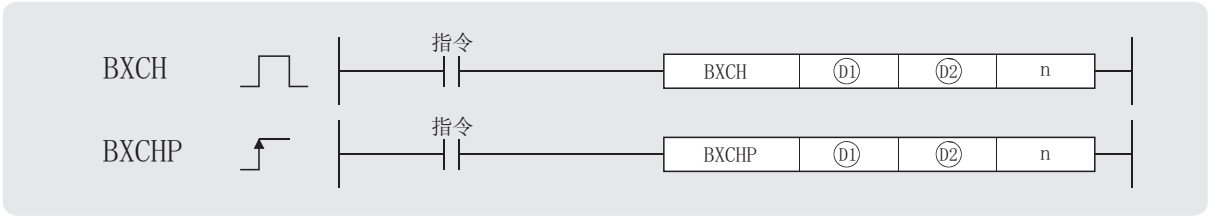


[列表模式]

步	指令	软元件	
0	LD	M0	
1	DXCHP	D0	D9
4	END		

6.4.10 块 16 位数据交换 (BXCH(P))

Basic | High performance | Process | Redundant | Universal

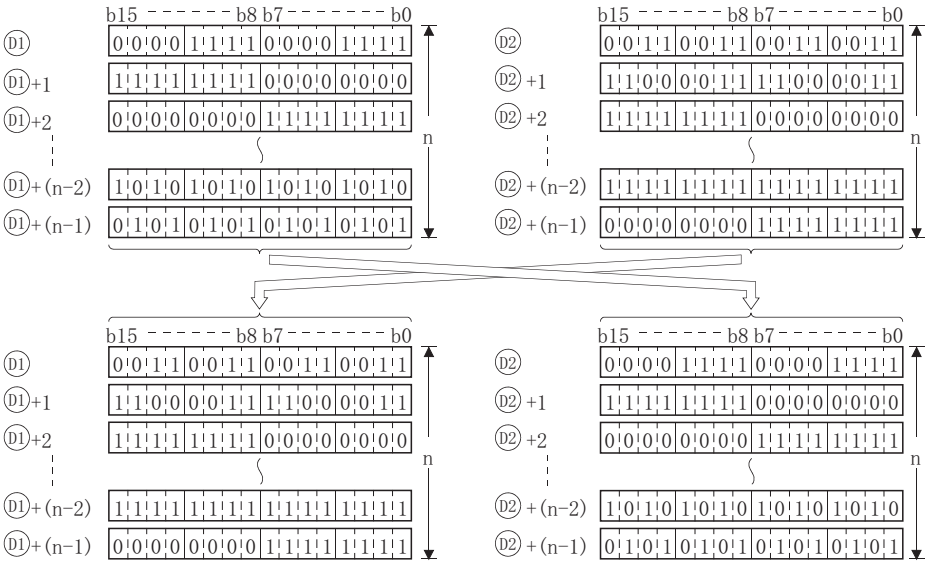


D1、D2 ：存储交换数据的软元件的起始编号 (BIN16 位)。
n ： 交换数 (BIN16 位)

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
D1	--	○		--					--
D2	--	○		--					--
n	○	○		○					--

功 能

将从D1中指定的软元件开始的 n 点的 16 位数据，与从D2中指定的软元件开始的 n 点的 16 位数据进行交换。



出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

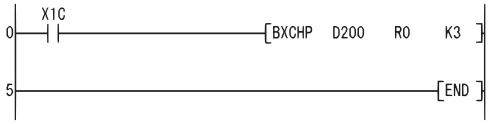
· 从(D1)、(D2)的软元件开始的 n 点的范围超出了相应软元件时。 (出错代码： 4101)

· (D1)、(D2)的软元件重复时。 (出错代码： 4101)

程序示例

(1) 以下为 X1C 变为 ON 时，将从 D200 开始的 3 点的数据与从 R0 开始的 3 点的数据进行交换的程序。

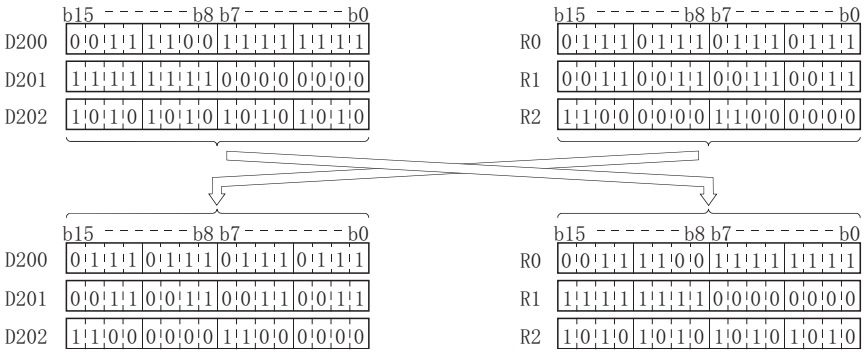
[梯形图模式]



[列表模式]

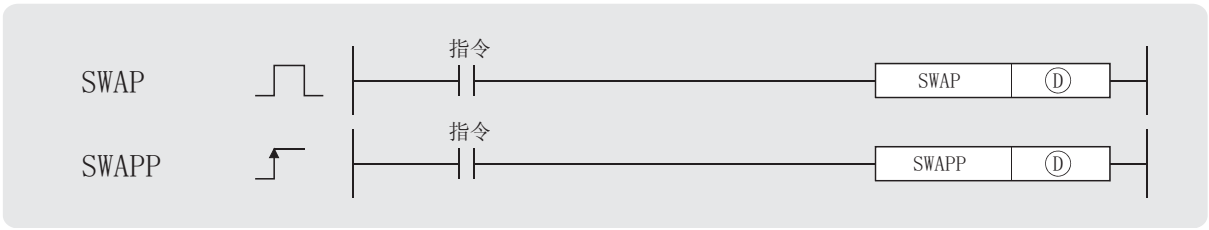
步	指令	软元件		
0	LD	X1C		
1	BXCHP	D200	R0	K3
5	END			

[动作]



6.4.11 高字节和低字节交换 (SWAP(P))

Basic | High performance | Process | Redundant | Universal

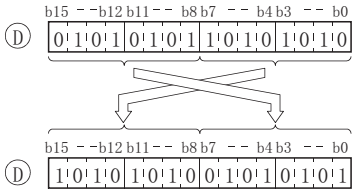


Ⓓ : 存储交换数据的软元件的起始编号 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
Ⓓ	○							--	

功 能

对Ⓓ中指定的软元件的高低各 8 位的值进行交换。



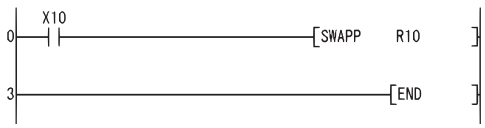
出 错

(1) 在 SWAP(P) 指令中无运算出错。

程序示例

(1) 以下为 X10 变为 ON 时，将 R10 的高 8 位与低 8 位进行交换的程序。

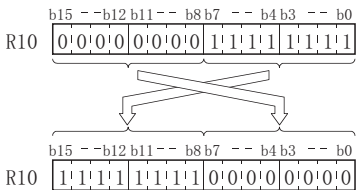
[梯形图模式]



[列表模式]

步	指令	软元件
0	LD	X10
1	SWAPP	R10
3	END	

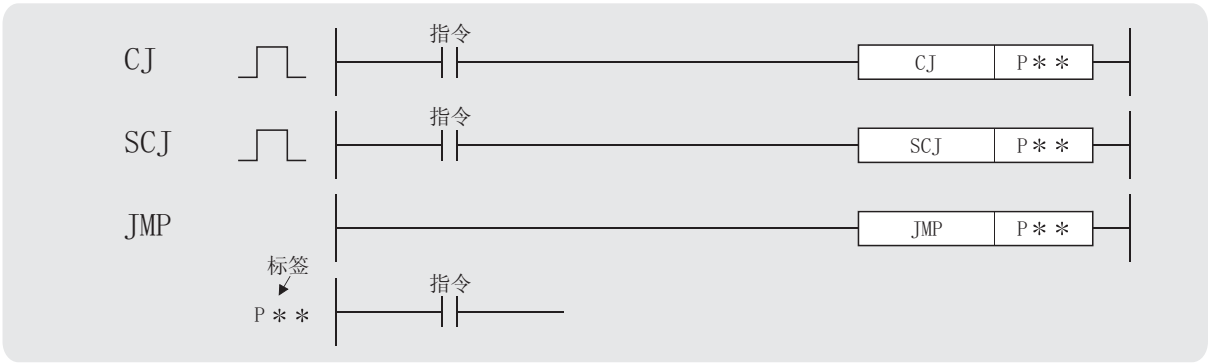
[动作]



6.5 程序分支指令

6.5.1 指针分支指令 (CJ、SCJ、JMP)

Basic | High performance | Process | Redundant | Universal



P＊＊ 跳转目标的指针编号（软元件名）

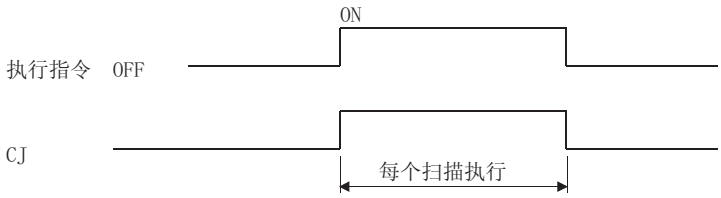
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				P
P	--								○

功 能

CJ

(1) 执行指令变为 ON 时，执行同一程序文件内的指定的指针号的程序。

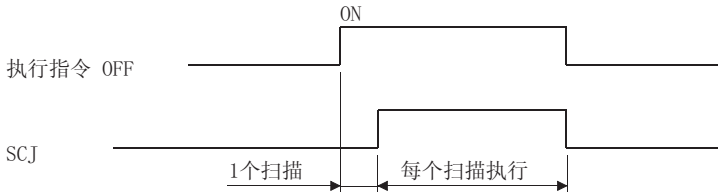
(2) 执行指令为 OFF 时，执行下一步的程序。



SCJ

(1) 执行指令 OFF → ON 时从下一个扫描开始，执行同一程序文件内的指定的指针号的程序。

(2) 执行指令为 OFF 或者 OFF → ON 时，执行下一步的程序。



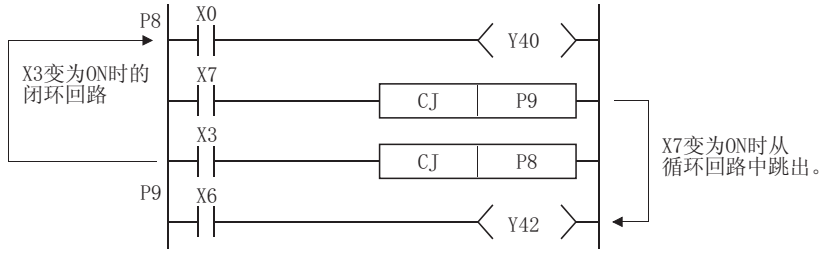
JMP

(1) 无条件地执行同一程序文件内的指定的指针号的程序。

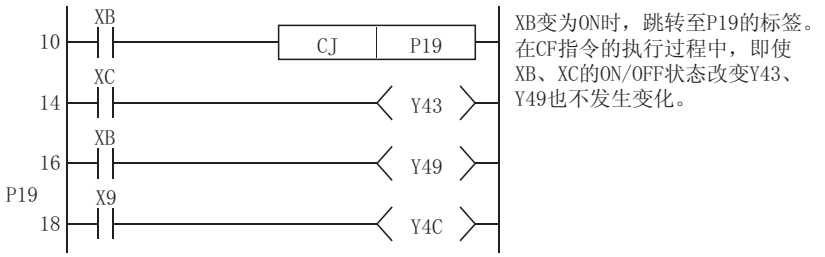
要 点

使用跳转指令时的注意事项如下所示：

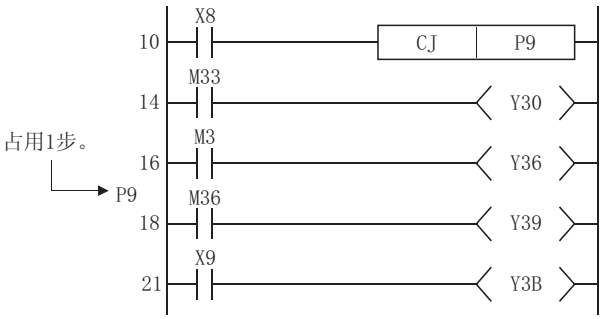
1. 在定时器线圈变为 ON 后，如果使用 CJ、SCJ、JMP 指令对处于 ON 状态的定时器进行跳转，则无法进行正常计测。
2. 如果使用 CJ、SCJ、JMP 指令对 OUT 指令进行跳转，则扫描时间将变短。
3. 如果使用 CJ、SCJ、JMP 指令强制跳转到程序尾部，则扫描时间将变短。
4. CJ、SCJ、JMP 指令可用于跳转到当前正执行的步之前的某一步。
   但是，需要考虑跳出该段循环回路的方法，以防止看门狗定时器超时。



5. 已通过 CJ、SCJ 、JMP 实现了的跳转的软元件不发生变化。



6. 标签 (P*) 占用 1 步。



7. 跳转指令只能指定同一程序文件内的指针号。
8. 在跳转运行过程中，如果跳转到跳转范围内的某个指针号，则将执行跳转目标指针号后面的程序。

出 错

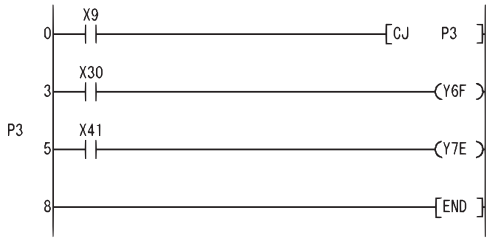
(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- 在 END 指令之前指定的指针号不存在时。 (出错代码： 4210)
- 在同一程序中指定了未被作为标签使用的指针号时。 (出错代码： 4210)
- 指定了一个也存在于其它程序中的公共指针时。 (出错代码： 4210)

程序示例

(1) 以下为 X9 变为 ON 时，跳转至 P3 的程序。

[梯形图模式]

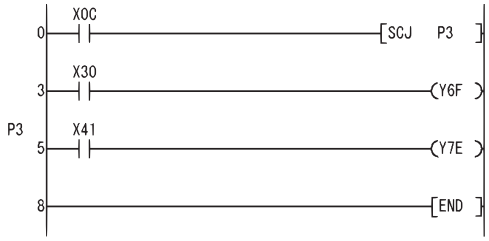


[列表模式]

步	指令	软元件
0	LD	X9
1	CJ	P3
3	LD	X30
4	OUT	Y6F
5	P3	
6	LD	X41
7	OUT	Y7E
8	END	

(2) 以下为 XC 变为 ON 时，从下一个扫描跳转至 P3 的程序。

[梯形图模式]



[列表模式]

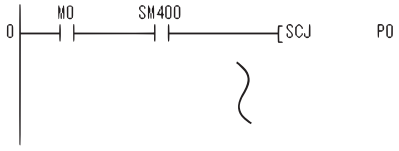
步	指令	软元件
0	LD	XOC
1	SCJ	P3
3	LD	X30
4	OUT	Y6F
5	P3	
6	LD	X41
7	OUT	Y7E
8	END	

注意事项

(1) 在通用型 QCPU 中，如果使用 SCJ 指令，需要在 SCJ 指令之前插入 AND SM400 (或者 NOP 指令)。

[程序示例 1]

[梯形图模式]

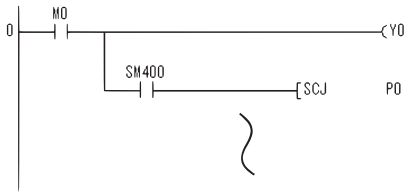


[列表模式]

步	指令	软元件
0	LD	M0
1	AND	SM400
2	SCJ	P0

[程序示例 2]

[梯形图模式]



[列表模式]

步	指令	软元件
0	LD	M0
1	OUT	Y0
2	AND	SM400
3	SCJ	P0

6.5.2 跳转至 END(GOEND)

Basic | High performance | Process | Redundant | Universal

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
--	--								

功 能

跳转至同一程序中的 FEND 或者 END 指令。

出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 在 CALL/ECALL 指令执行后 RET 指令执行前执行了 GOEND 指令时。 (出错代码： 4211)

· 在 FOR 指令执行后 NEXT 指令执行前执行了 GOEND 指令时。 (出错代码： 4200)

· 在中断程序执行过程中，在 IRET 指令执行前执行了 GOEND 指令时。 (出错代码： 4221)

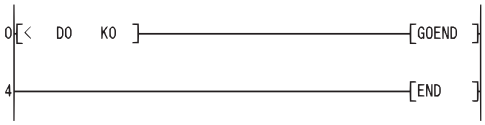
· 在 CHKCIR ～ CHKEND 指令之间执行了 GOEND 指令时。 (出错代码： 4230)

· 在 IX ～ IXEND 指令之间执行了 GOEND 指令时。 (出错代码： 4231)

程序示例

(1) 以下为 D0 的值为负数时，跳转至 END 的程序。

[梯形图模式]



[列表模式]

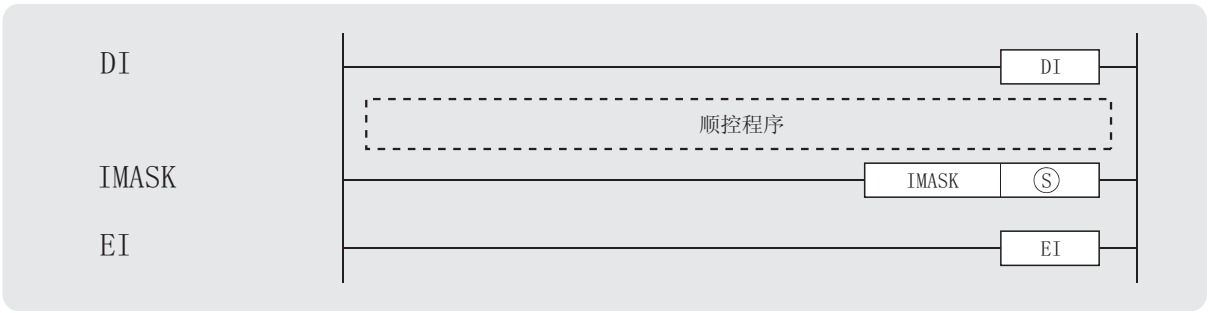
步	指令	软元件	
0	LD<	D0	K0
3	GOEND		
4	END		

6.6 程序执行控制指令

6.6.1 中断禁止 / 允许指令、中断程序屏蔽 (DI、EI、IMASK)

Basic | High performance | Process | Redundant | Universal

1 使用基本型 QCPU 时



Ⓢ : 中断屏蔽数据或者存储中断屏蔽数据的软元件的起始编号 (BIN16 位)

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
Ⓢ	--	○		--					

功 能

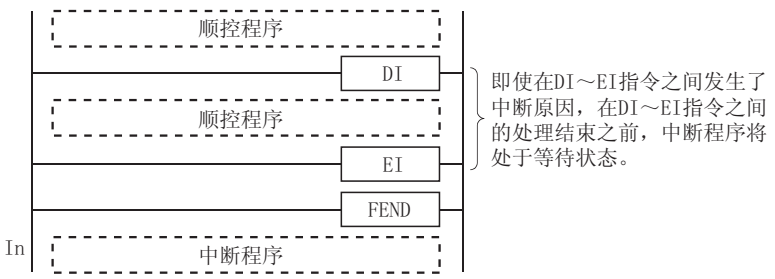
DI

(1) 即使发生了中断程序的启动原因。也只能在执行了 EI 指令之后才允许执行中断程序。

(2) 当电源接通或者 CPU 模块复位时，将变为 DI 状态。

EI

EI 指令用于解除 DI 指令执行时的中断禁止状态，并将通过 IMASK 指令设为允许的中断指针号的中断程序置于可执行状态。
未执行 IMASK 指令时，I32 ～ I47 处于中断禁止状态。



IMASK

(1) 根据Ⓢ中指定的软元件开始的 8 点的位模式，将指定中断指针号的中断程序置于执行允许状态 / 执行禁止状态。

- 1(ON)..... 中断程序的执行允许状态
- 0(OFF)..... 中断程序的执行禁止状态

(2) 对应于各个位的中断指针号如下所示：

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Ⓢ	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0
Ⓢ+1	I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16
Ⓢ+2	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32
Ⓢ+3	I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48
Ⓢ+4	I79	I78	I77	I76	I75	I74	I73	I72	I71	I70	I69	I68	I67	I66	I65	I64
Ⓢ+5	I95	I94	I93	I92	I91	I90	I89	I88	I87	I86	I85	I84	I83	I82	I81	I80
Ⓢ+6	I111	I110	I109	I108	I107	I106	I105	I104	I103	I102	I101	I100	I99	I98	I97	I96
Ⓢ+7	I127	I126	I125	I124	I123	I122	I121	I120	I119	I118	I117	I116	I115	I114	I113	I112

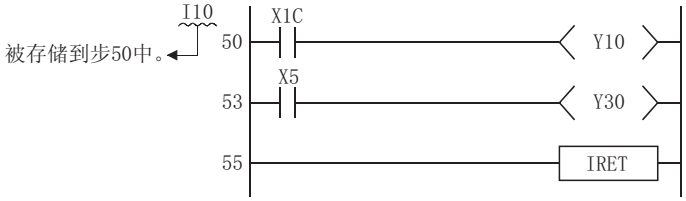
(3) 当电源接通或者 CPU 模块复位时，I0 ～ I31、I48 ～ I127 的中断程序将变为执行允许状态，此外，I32 ～ I47 的中断程序将变为执行禁止状态。

(4) Ⓢ、Ⓢ+1、Ⓢ+2、Ⓢ+3 ～Ⓢ+7 的软元件状态将被存储到 SD715 ～ SD717、SD781 ～ SD785(IMASK 指令屏蔽模式存储区域) 中。

(5) 虽然特殊寄存器被分隔为 SD715 ～ SD717、SD781 ～ SD785，但Ⓢ～Ⓢ+7 的软元件号应设置为连续的编号。

要 点

1. 中断指针占用 1 步。



2. 关于中断条件，请参阅所使用的 CPU 模块的用户手册 (功能解说 / 程序基础篇)。
3. 在中断程序执行过程中是处于 DI(中断禁止) 状态的。不要在中断程序中插入 EI 指令，不要在中断程序执行过程中执行其它中断程序，应避免进行多重中断。
4. 如果在主控制中存在有 EI、DI 指令，则不管 MC 指令处于执行还是非执行状态，都将执行 EI、DI 指令。

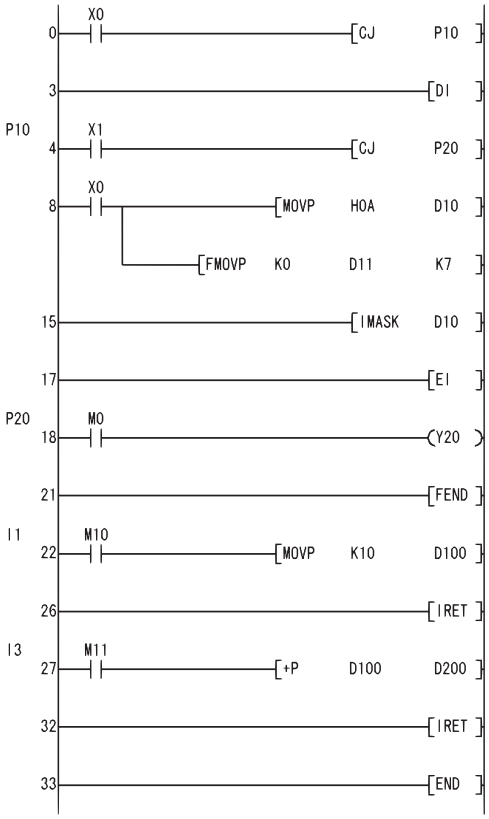
出　错

(1) 在 DI、EI、IMASK 指令中无运算出错。

程序示例

(1) 以下为 X0 为 ON 状态时，仅将中断指针号为 I1、I3 的中断程序置于执行允许状态的程序。

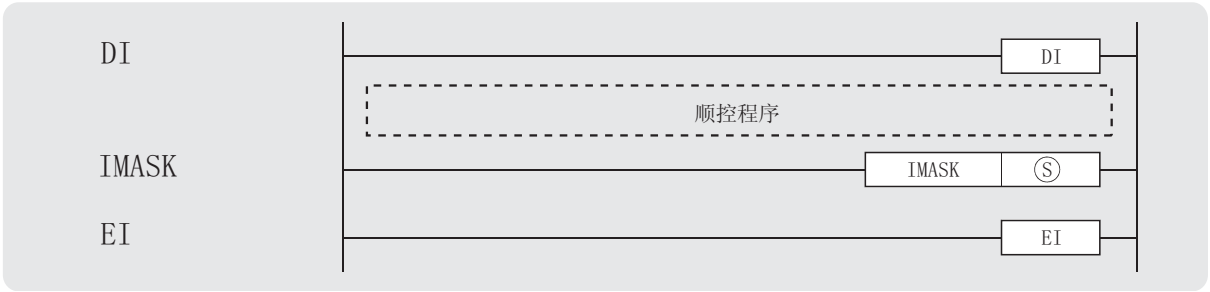
[梯形图模式]



[列表模式]

步	指令	软元件		
0	LD	X0		
1	CJ	P10		
3	DI			
4	P10			
5	LD	X1		
6	CJ	P20		
8	LD	X0		
9	MOVP	HOA	D10	
11	FMOVP	K0	D11	K7
15	IMASK	D10		
17	EI			
18	P20			
19	LD	M0		
20	OUT	Y20		
21	FEND			
22	I1			
23	LD	M10		
24	MOVP	K10	D100	
26	IRET			
27	I3			
28	LD	M11		
29	+P	D100	D200	
32	IRET			
33	END			

2 使用高性能型 QCPU/ 过程 CPU/ 冗余 CPU/ 通用型 QCPU 时



Ⓢ : 存储中断屏蔽数据的软元件的起始编号 (BIN16 位)

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
Ⓢ	--	○		--					

功 能

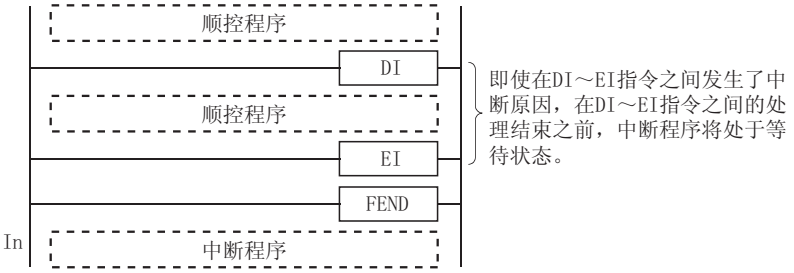
DI

(1) 即使发生了中断程序的启动原因。也只能在执行了 EI 指令之后才允许执行中断程序。

(2) 当电源接通或者 CPU 模块复位时，将变为 DI 状态。

EI

EI 指令用于解除 DI 指令执行时的中断禁止状态，并将通过 IMASK 指令设为允许的中断指针号的中断程序及恒定周期执行型程序置于可执行状态。
未执行 IMASK 指令时，I32 ～ I47 处于中断禁止状态。



IMASK

(1) 根据Ⓢ中指定的软元件开始的 16 点的位模式，将指定中断指针号的中断程序置于执行允许状态 / 执行禁止状态。

- 1(ON) ..... 中断程序的执行允许状态
- 0(OFF) ..... 中断程序的执行禁止状态

(2) 对应于各个位的中断指针号如下所示：

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Ⓢ	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0
Ⓢ+ 1	I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16
Ⓢ+ 2	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32
Ⓢ+ 3	I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48
Ⓢ+ 4	I79	I78	I77	I76	I75	I74	I73	I72	I71	I70	I69	I68	I67	I66	I65	I64
Ⓢ+ 5	I95	I94	I93	I92	I91	I90	I89	I88	I87	I86	I85	I84	I83	I82	I81	I80
Ⓢ+ 6	I111	I110	I109	I108	I107	I106	I105	I104	I103	I102	I101	I100	I99	I98	I97	I96
Ⓢ+ 7	I127	I126	I125	I124	I123	I122	I121	I120	I119	I118	I117	I116	I115	I114	I113	I112
Ⓢ+ 8	I143	I142	I141	I140	I139	I138	I137	I136	I135	I134	I133	I132	I131	I130	I129	I128
Ⓢ+ 9	I159	I158	I157	I156	I155	I154	I153	I152	I151	I150	I149	I148	I147	I146	I145	I144
Ⓢ+10	I175	I174	I173	I172	I171	I170	I169	I168	I167	I166	I165	I164	I163	I162	I161	I160
Ⓢ+11	I191	I190	I189	I188	I187	I186	I185	I184	I183	I182	I181	I180	I179	I178	I177	I176
Ⓢ+12	I207	I206	I205	I204	I203	I202	I201	I200	I199	I198	I197	I196	I195	I194	I193	I192
Ⓢ+13	I223	I222	I221	I220	I219	I218	I217	I216	I215	I214	I213	I212	I211	I210	I209	I208
Ⓢ+14	I239	I238	I237	I236	I235	I234	I233	I232	I231	I230	I229	I228	I227	I226	I225	I224
Ⓢ+15	I255	I254	I253	I252	I251	I250	I249	I248	I247	I246	I245	I244	I243	I242	I241	I240

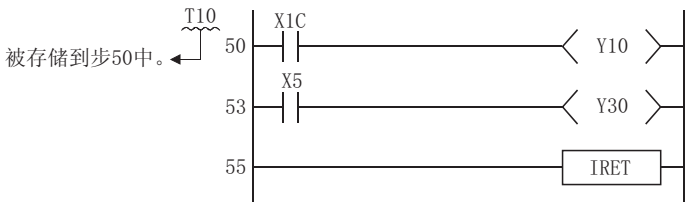
(3) 当电源接通或者 CPU 模块复位时，I0 ～ I31、I48 ～ I127 的中断程序将变为执行允许状态，此外，I32 ～ I47 的中断程序将变为执行禁止状态。

(4) Ⓢ、Ⓢ+1、Ⓢ+2、Ⓢ+3 ～Ⓢ+15 的软元件状态将被存储到 SD715 ～ SD717、SD781 ～ SD793(IMASK 指令屏蔽模式存储区域) 中。

(5) 虽然特殊寄存器被分隔为 SD715 ～ SD717、SD781 ～ SD793，但Ⓢ～Ⓢ+15 的软元件号应设置为连续的编号。

要 点

1. 中断指针占用 1 步。



2. 关于中断条件，请参阅下述手册。
   · 所使用的 CPU 模块的用户手册 (功能解说 / 程序基础篇)。
3. 在中断程序执行过程中是处于 DI(中断禁止) 状态的。不要在中断程序中插入 EI 指令，不要在中断程序执行过程中执行其它中断程序，应避免进行多重中断。
4. 如果在主控制中存在有 EI、DI 指令，则不管 MC 指令处于执行还是非执行状态，都将执行 EI、DI 指令。

出　错

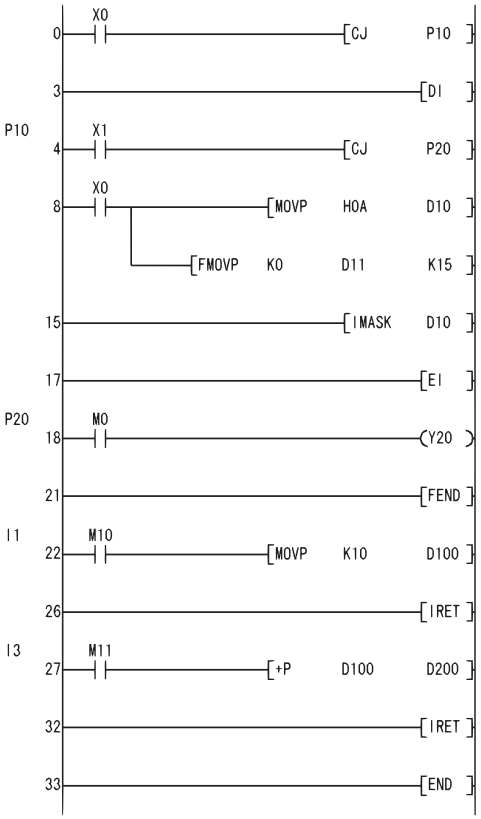
(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· ⓢ中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)　　(出错代码 : 4101)

程序示例

(1) 以下为 X0 为 ON 状态时，将中断指针号的中断程序置于执行允许状态的程序。

[梯形图模式]

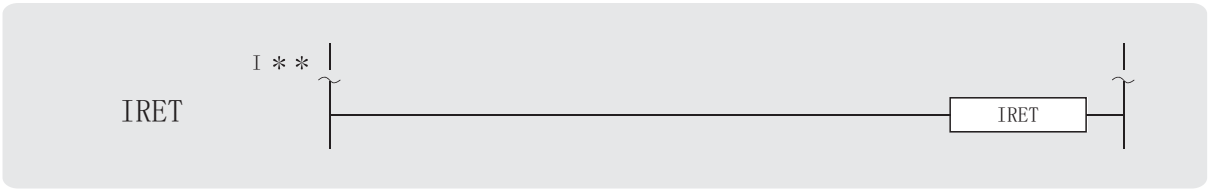


[列表模式]

步	指令	软元件		
0	LD	X0		
1	CJ	P10		
3	DI			
4	P10			
5	LD	X1		
6	CJ	P20		
8	LD	X0		
9	MOVP	H0A	D10	
11	FMOVP	K0	D11	K15
15	IMASK	D10		
17	EI			
18	P20			
19	LD	M0		
20	OUT	Y20		
21	FEND			
22	I1			
23	LD	M10		
24	MOVP	K10	D100	
26	IRET			
27	I3			
28	LD	M11		
29	+P	D100	D200	
32	IRET			
33	END			

6.6.2 中断程序的恢复 (IRET)

Basic | High performance | Process | Redundant | Universal



设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
--	--								

功 能

(1) 表示中断程序的处理结束。

(2) 执行 IRET 指令后，返回至顺序程序处理。

出 错

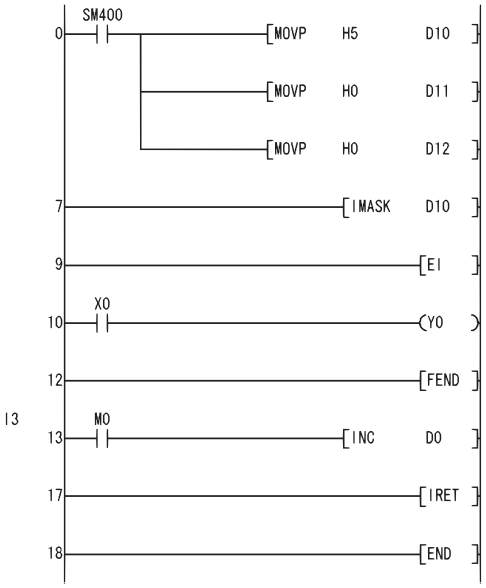
(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- 中断号对应的指针不存在时。 (出错代码： 4220)
- 在执行中断程序之前执行了 IRET 指令时。 (出错代码： 4223)
- 在发生中断后 IRET 指令执行前执行了 END、FEND、GOEND、STOP 指令时。 (出错代码： 4221)
- 在恒定周期执行型程序中执行了 IRET 指令时。 (仅通用型 QCPU) (出错代码： 4223)

程序示例

(1) 以下为发生了 3 号中断时，如果 M0 变为 ON 则进行 D0+1 的程序。

[梯形图模式]



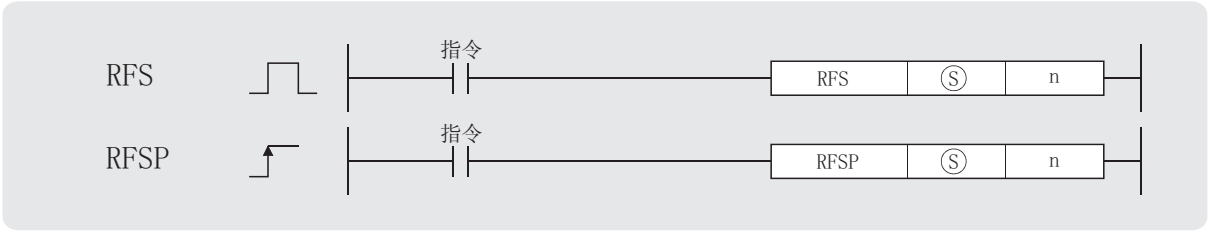
[列表模式]

步	指令	软元件	
0	LD	SM400	
1	MOVP	H5	D10
3	MOVP	H0	D11
5	MOVP	H0	D12
7	IMASK	D10	
9	EI		
10	LD	X0	
11	OUT	Y0	
12	FEND		
13	I3		
14	LD	M0	
15	INC	D0	
17	IRET		
18	END		

6.7 I/O 刷新指令

6.7.1 I/O 刷新 (RFS(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ : 刷新的软元件的起始编号 (位)。
n : 刷新点数 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K，H	其它
	位	字		位	字				
Ⓢ	○ (仅 X,Y)		--						--
n	○		○						--

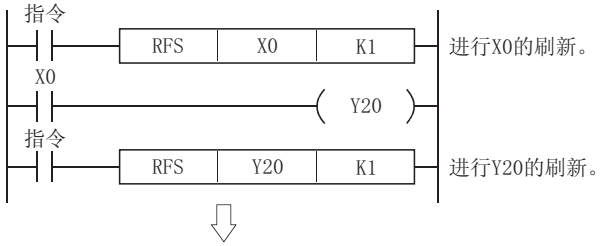
功 能

(1) 该功能是只对 1 个扫描中相应的软元件进行刷新，并进行外部输入的获取或者至输出模块的输出的功能。

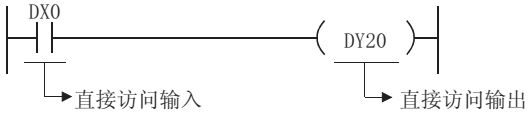
(2) 由于输入的获取及至外部的输出只是在执行了程序的 END 指令之后批量地执行，因此不能在 1 个扫描中将脉冲信号输出到外部。
执行 I/O 刷新指令时，由于在程序执行过程中对相应的输入 (X) 或输出 (Y) 进行强制刷新，因此可以在 1 个扫描中将脉冲信号输出到外部。

(3) 以 1 点为单元进行输入 (X) 或输出 (Y) 刷新时，应使用直接访问输入 (DX)、直接访问输出 (DY)。

[使用RFS指令的程序]



[使用直接访问输入、直接访问输出的程序]



出　错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

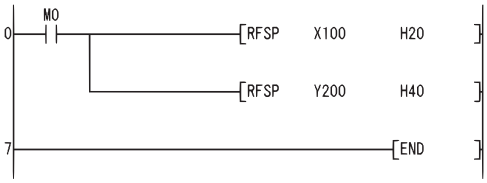
· 从Ⓢ中指定的软元件开始的 n 点的范围超出了相邻 I/O 的范围时。

(出错代码： 4101)

程序示例

(1) 以下为 M0 变为 ON 时，对 X100 ～ X11F、Y200 ～ Y23F 进行刷新的程序。

[梯形图模式]



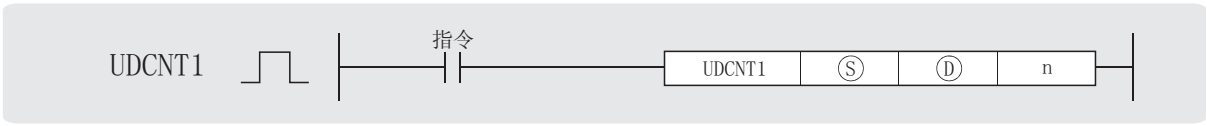
[列表模式]

步	指令	软元件	
0	LD	M0	
1	RFSP	X100	H20
4	RFSP	Y200	H40
7	END		

6.8 其它方便的指令

6.8.1 单相输入加法 / 减法计数器 (UDCNT1)

Basic High performance Process Redundant Universal



Ⓢ：Ⓢ+0：用于计数输入的输入编号（位）。

Ⓢ+1：用于设置加法 / 减法计数（位）。

· OFF：加法计数（计数时数字增加）。

· ON ：减法计数（计数时数字减少）。

Ⓓ：通过 UDCNT1 指令进行计数的计数器编号（软元件名）。

n：设置值 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○(仅X) *1	--	--	--		--			--
Ⓓ	--	△*2(仅C)	--	--		--			--
n	△*2	△*2	△*2	○					--

*1: Ⓢ只能使用 X 软元件。
但是，X 软元件只能在 I/O 点数（可进行实际 I/O 模块访问的点数）的范围内使用。

*2: 不能使用局部软元件及各程序中设置的文件寄存器。

功 能

(1) Ⓢ中指定的输入从 OFF 变为 ON 时，对Ⓓ中指定的计数器的当前值进行更新。

(2) 计数的方向取决于Ⓢ+1 中指定输入的 ON/OFF 状态。

· OFF：加法计数（以增加当前值的方式计数）

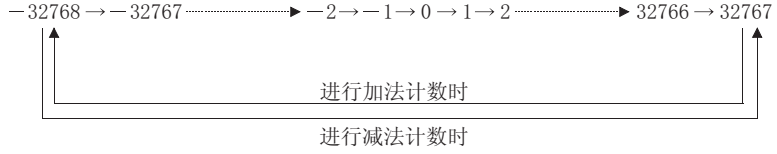
· ON ：减法计数（以减少当前值的方式计数）

(3) 计数处理的执行方式如下：

· 在加法计数过程中，当前值变为与 n 中指定的设置值相等时，则Ⓓ中指定的计数器的触点将变为 ON。
但是，即使Ⓓ中指定的计数器的触点变为 ON 时，当前值的计数也将继续。（参阅程序示例 (1)）

· 在减法计数过程中，当前值变为设置值 -1 时，则Ⓓ中指定的计数器的触点将变为 OFF。（参阅程序示例 (1)）

· Ⓓ中指定的计数器是环型计数器。
在当前值为 32767 时，如果正进行加法计数，则当前值将变为 -32768。
此外，在当前值为 -32768 时，如果正进行减法计数，则当前值将变为 32767。
当前值的的计数处理内容如下所示：



(4) 当执行指令从 OFF 变为 ON 时，使用了 UDCNT1 指令的计数处理将开始计数；当执行指令从 ON 变为 OFF 时，中止计数。
当执行指令再次从 OFF 变为 ON 时，从计数中止时的值开始重新进行计数。

(5) Ⓓ中指定的计数器的当前值清除及触点的 OFF 是通过 RST 指令进行的。

要 点

1.UDCNT1 指令将变量的软元件数据登录到 CPU 模块的工作区，而实际的计数动作是通过系统中断进行处理。
（当执行指令变为 OFF 或者 CPU 模块 STOP → RUN 时，CPU 模块工作区中登录的软元件数据将被清除）。
因此，可计数的脉冲的 ON 和 OFF 时间必须比 CPU 模块的中断间隔长。
CPU 模块的中断间隔如下所示。

CPU 模块型号	中断间隔
高性能型 QCPU、过程 CPU、通用型 QCPU	1ms

2.在使用了 UDCNT1 指令的计数过程中（执行指令处于 ON 状态时），不能对设置值进行更改。
如果要更改设置值，应先将执行指令置为 OFF。

3.其它指令不能使用 UDCNT1 指令中指定的计数器。
如果这些计数器为其它指令所使用，将不能进行正常计数。

4.在所有正执行的程序中，UDCNT1 指令最多可以使用 6 次。
第 7 个及以后的 UDCNT1 指令将执行无处理。

出 错

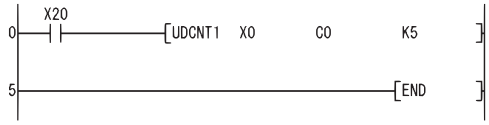
(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· Ⓢ中指定的软元件超出了相应软元件的范围时。 （出错代码： 4101）

程序示例

(1) 以下为 X20 变为 ON 之后，将 X0 的 OFF → ON 次数通过 C0(加法计数器 / 减法计数器) 进行计数的程序。

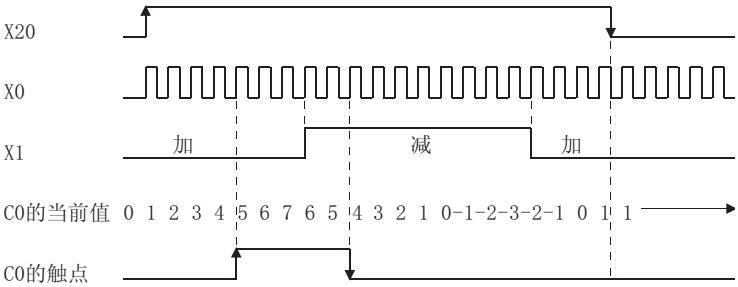
[梯形图模式]



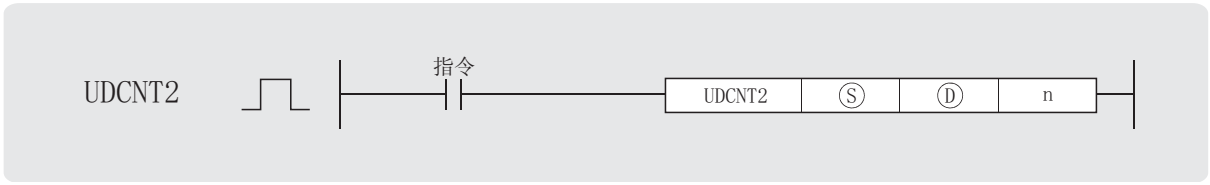
[列表模式]

步	指令	软元件		
0	LD	X20		
1	UDCNT1	X0	C0	K5
5	END			

[动作]



6.8.2 两相输入加法 / 减法计数器 (UDCNT2)



Ⓢ : Ⓢ+0: 用于计数输入的输入编号 (A 相脉冲) (位)。
Ⓢ+1: 用于计数输入的输入编号 (B 相脉冲) (位)。
Ⓓ : 通过 UDCNT2 指令进行计数的计数器编号 (软元件名)
n : 设置值 (BIN16 位)

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○ (仅 X) *1	--	--	--		--			--
Ⓓ	--	△ *2 (仅 C)	--	--		--			--
n	△ *2	△ *2	△ *2	○					--

*1: Ⓢ只能使用 X 软元件。
但是，X 软元件只能在 I/O 点数 (可进行实际 I/O 模块访问的点数) 的范围内使用。
*2: 不能使用局部软元件及各程序中设置的文件寄存器。

功 能

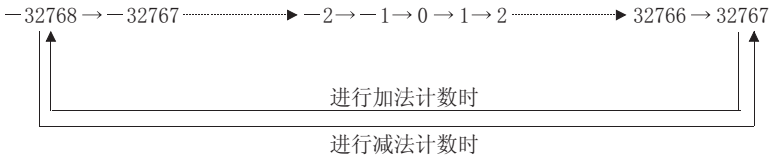
(1) 根据Ⓢ中指定的输入 (A 相脉冲) 及Ⓢ+1 中指定的输入 (B 相脉冲) 状态，对Ⓓ中指定的计数器的当前值进行更新。

(2) 计数方向的确定如下所示。

- 当Ⓢ为 ON 时，如果Ⓢ+1 从 OFF 变为 ON，则执行加法计数 (计数器的当前值增加)。
- 当Ⓢ为 ON 时，如果Ⓢ+1 从 ON 变为 OFF，则执行减法计数运算 (计数器的当前值减少)。
- 如果Ⓢ为 OFF，则不执行计数。

(3) 计数处理的执行方式如下：

- 在加法计数过程中，当前值变为与 n 中指定的设置值相等时，则Ⓓ中指定的计数器的触点将变为 ON。
  但是，即使Ⓓ中指定的计数器的触点变为 ON 时，当前值的计数也将继续。(参阅程序示例 (1))
- 在减法计数过程中，当前值变为设置值 -1 时，则Ⓓ中指定的计数器的触点将变为 OFF。
  (参阅程序示例 (1))
- Ⓓ中指定的计数器是环型计数器。
  在当前值为 32767 时，如果正进行加法计数，则当前值将变为 -32768。
  此外，在当前值为 -32768 时，如果正进行减法计数，则当前值将变为 32767。
  当前值的的计数处理内容如下所示：



(4) 当执行指令从 OFF 变为 ON 时，使用了 UDCNT2 指令的计数处理将开始计数；当执行指令从 ON 变为 OFF 时，中止计数。
当执行指令再次从 OFF 变为 ON 时，从计数中止时的值开始重新进行计数。

(5) ⓓ中指定的计数器的当前值清除及触点的 OFF 是通过 RST 指令进行的。

要 点

1.UDCNT2 指令将变量的软元件数据登录到 CPU 模块的工作区，而实际的计数动作是通过系统中断进行处理。
(当执行指令变为 OFF 或者 CPU 模块 STOP → RUN 时，CPU 模块工作区中登录的软元件数据将被清除)。
因此，可计数的脉冲的 ON 和 OFF 时间必须比 CPU 模块的中断间隔长。
CPU 模块的中断间隔如下所示。

CPU 模块型号	中断间隔
高性能型 QCPU、过程 CPU、通用型 QCPU	1ms

2.在使用了 UDCNT2 指令的计数过程中 (执行指令处于 ON 状态时)，不能对设置值进行更改。
如果要更改设置值，应先将执行指令置为 OFF。
3.其它指令不能使用 UDCNT2 指令中指定的计数器。
如果这些计数器为其它指令所使用，将不能进行正常计数。
4.在所有正执行的程序中，UDCNT2 指令最多可以使用 5 次。
第 6 个及以后的 UDCNT2 指令将执行无处理。

出 错

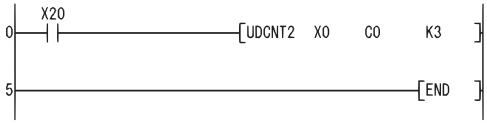
(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· ⓢ中指定的软元件超出了相应软元件的范围时。 (出错代码： 4101)

程序示例

(1) 以下为 X20 变为 ON 之后，将 X0、X1 的状态通过 C0(加法计数器 / 减法计数器) 进行计数的程序。

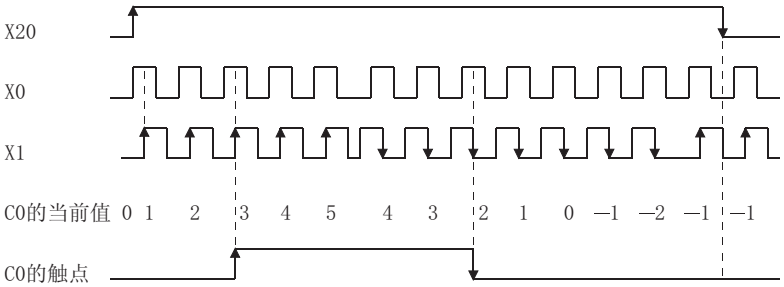
[梯形图模式]



[列表模式]

步	指令	软元件		
0	LD	X20		
1	UDCNT2	X0	C0	K3
5	END			

[动作]



6.8.3 教学定时器 (TTMR)

Ⓓ ：Ⓓ+0：存储测量值的软元件 (BIN16 位)。
Ⓓ+1：为 CPU 模块的系统所用 (BIN16 位)。
n ：测量值的乘数 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓓ	--	○		--					--
n	--	○		○					--

功 能

(1) 对执行指令处于 ON 状态的时间以秒为单位进行测量，然后乘以 n 中指定的乘数，并将结果存储到Ⓓ中指定的软元件中。

(2) 当执行指令从 OFF 变为 ON 时，对Ⓓ+0、Ⓓ+1 中指定的软元件进行清除。

(3) n 中可指定的乘数如下所示。

n	乘数
0	1
1	10
2	100

要 点

1. 在执行 TTMR 指令时进行时间计测。
   如果使用 JMP 指令等对 TTMR 指令进行跳转，将无法进行正确测量。
2. 在 TTMR 指令的执行过程中不要更改 n 中指定的乘数。
   更改 n 中指定的乘数会导致值不正确。
3. TTMR 指令也可用于低速执行型程序中。
4. 由于Ⓓ+1 中指定的软元件为 CPU 模块的系统所使用，因此用户不要对该值进行变更。
   如果用户更改了该值，则Ⓓ中指定的软元件中存储的值将为不正确的值。

(4) 如果 n 中指定的值超出了 0 ～ 2 的范围，则执行无处理。

出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

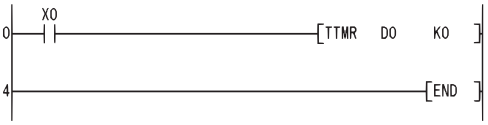
· ⓓ中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)

(出错代码： 4101)

程序示例

(1) 以下为将 X0 处于 ON 状态的时间存储到 D0 中的程序。

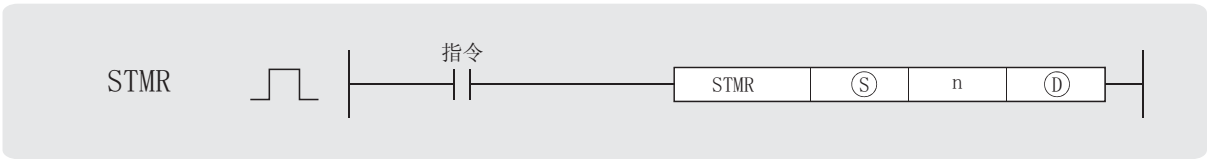
[梯形图模式]



[列表模式]

步	指令	软元件	
0	LD	X0	
1	TTMR	D0	K0
4	END		

6.8.4 特殊功能定时器 (STMR)



Ⓢ ： 定时器编号 (字)。
n ： 设置值 (BIN16 位)。
Ⓓ ： Ⓓ+0 ： OFF 延迟定时器输出 (位)。
Ⓓ+1 ： OFF 后一次定时器输出 (位)。
Ⓓ+2 ： ON 后一次定时器输出 (位)。
Ⓓ+3 ： ON 延迟 +OFF 延迟定时器 (位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	--	△*1	--	--					--
n	○	○	○	○					--
Ⓓ	○	--	--	--					--

*1：仅定时器 (T) 可用。

功 能

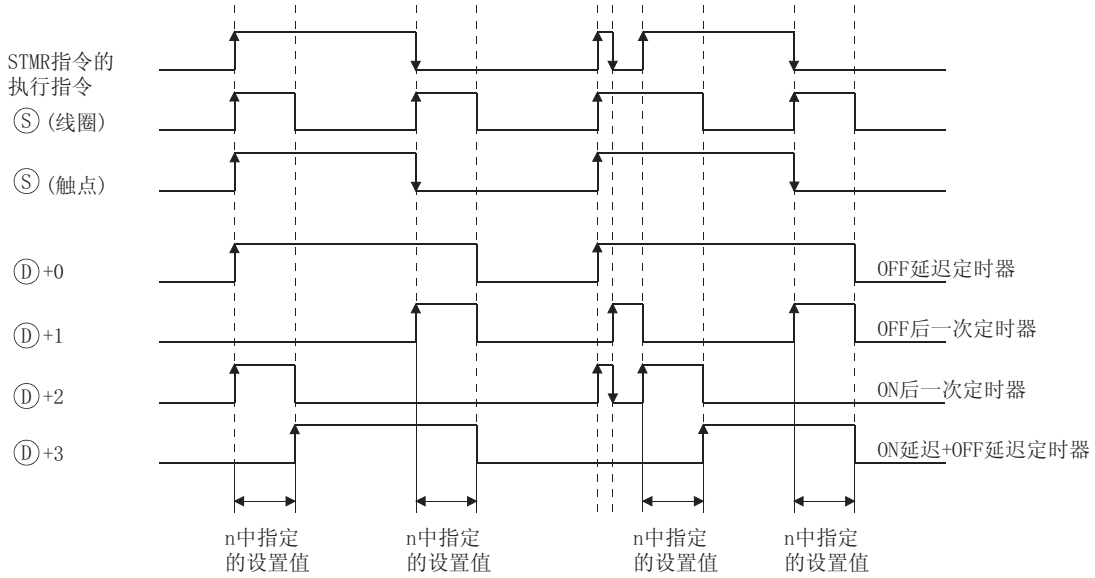
(1) 在 STMR 指令中，使用从Ⓓ中指定的软元件开始的 4 点，进行 4 种类型的定时器输出。

- OFF 延迟定时器输出 (Ⓓ+0)
  在 STMR 指令的执行指令的上升沿变为 ON，在执行指令的下降沿后经过了 n 中指定的时间之后变为 OFF。
- OFF 后一次定时器输出 (Ⓓ+1)
  在 STMR 指令的执行指令的下降沿变为 ON，经过了 n 中指定的时间后变为 OFF。
- ON 后一次定时器输出 (Ⓓ+2)
  在 STMR 指令的执行指令的上升沿变为 ON，经过了 n 中指定的时间后或者 STMR 指令的执行指令 OFF 时变为 OFF。
- ON 延迟 +OFF 延迟定时器 (Ⓓ+3)
  在定时器线圈的下降沿变为 ON，在执行指令的下降沿后经过了 n 中指定的时间之后变为 OFF。

(2) Ⓢ中指定的定时器线圈通过 STMR 指令的执行指令的上升沿或下降沿变为 ON 后，开始当前值的计测。

- 定时器线圈在到达 n 中指定的设置值之前进行计测，变为 “ 时间到 ” 时变为 OFF。
- 定时器线圈在到达 “ 时间到 ” 之前如果 STMR 指令的执行指令变为 OFF，则保持为 ON 状态不变。
  此时定时器的计测将继续进行。
  如果 STMR 指令再次变为 ON，则在将当前值置为 0 后，重新开始计测。

(3) 定时器触点在 STMR 指令的执行指令的上升沿变为 ON，在定时器线圈的下降沿之后，在 STMR 指令的执行指令的下降沿变为 OFF。
定时器触点为 CPU 模块的系统所用，因此用户不能使用。



(4) STMR 指令中指定的定时器的当前值的计测与 STMR 指令的执行指令的 ON/OFF 状态无关。
通过 JMP 指令等跳过了 STMR 指令时，将无法进行正常计测。

(5) Ⓓ中指定的定时器的计测单位与低速定时器相同。

(6) n 的设置值的指定范围为 0 ～ 32767。
0 ～ 32767 以外时将执行无处理。

(7) Ⓢ中指定的定时器不能被用于 OUT 指令。
如果 STMR 指令与 OUT 指令使用了相同的定时器编号，将无法执行正常动作。

出　错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

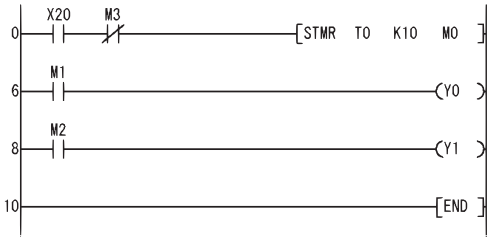
· Ⓓ中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)

(出错代码： 4101)

程序示例

(1) 以下为 X20 变为 ON 时，Y0、Y1 每隔 1 秒 ON/OFF (闪烁) 的程序。
(定时器使用 100ms 定时器)

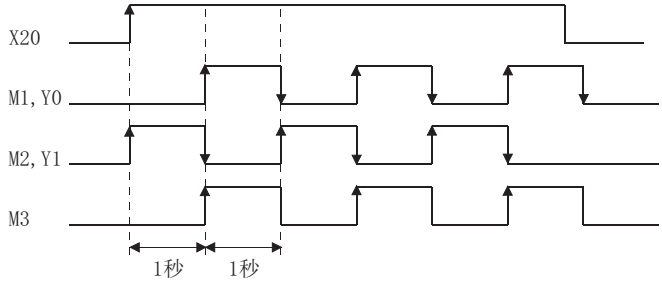
[梯形图模式]



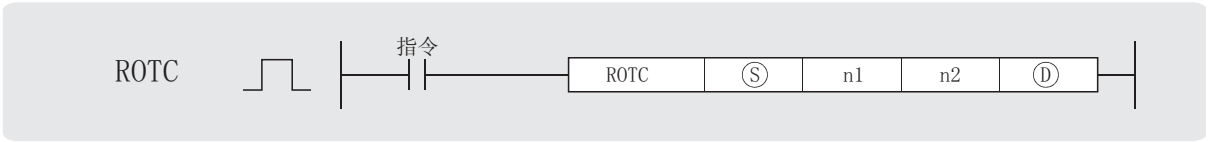
[列表模式]

步	指令	软元件		
0	LD	X20		
1	ANI	M3		
2	STMR	T0	K10	M0
6	LD	M1		
7	OUT	Y0		
8	LD	M2		
9	OUT	Y1		
10	END			

[时序图]



6.8.5 旋转台就近控制 (ROTC)



Ⓢ ： Ⓢ+0：用于测量旋转台旋转次数。(系统用)(BIN16 位)。
Ⓢ+1：调用窗口编号 (BIN16 位)。
Ⓢ+2：调用部件编号 (BIN16 位)。
n1 ： 旋转台的分区数 (2 ～ 32767)(BIN16 位)。
n2 ： 低速区间数 (0 ～ n1 以内的值)(BIN16 位)。
Ⓓ ： Ⓓ+0：A 相输入信号 (位)。
Ⓓ+1：B 相输入信号 (位)。
Ⓓ+2：0 点检测输入信号 (位)。
Ⓓ+3：高速正转输出信号 (系统用)(位)。
Ⓓ+4：低速正转输出信号 (系统用)(位)。
Ⓓ+5：停止输出信号 (系统用)(位)。
Ⓓ+6：低速逆转输出信号 (系统用)(位)。
Ⓓ+7：高速逆转输出信号 (系统用)(位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	--	○		--					--
n1	○	○		○					--
n2	○	○		○					--
Ⓓ	○	--		--					--

功 能

(1) 该功能是为了执行以下控制： 在以 n1 中指定的值进行了均等分区的旋转台上，为了对Ⓢ+2 中指定编号的部件进行取放，使旋转台就近旋转到Ⓢ+1 中指定编号的窗口位置处。

(2) 部件编号以及窗口编号是基于对以逆时针旋转方向分配的部件进行控制。

(3) Ⓢ+0 是系统用计数器，用于对 0 号窗口中的部件进行计数。
不要通过顺控程序等对其数据进行改写。
如果用户进行了改写，将无法进行正常控制。

(4) n2 的值应小于 n1 中指定的旋转台的分区数。

(5) Ⓓ+0 及Ⓓ+1 是用于检测旋转台的正转 / 逆转的 A 相及 B 相输入信号。
旋转方向的判别是通过 A 相为 ON 状态时的 B 相的上升沿 / 下降沿来进行的。
· B 相上升沿时： 正转 (顺时针旋转)
· B 相下降沿时： 逆转 (逆时针旋转)

(6) ⓓ+2 是 0 号部件到达 0 号窗口时将变为 ON 的 0 点检测信号。
在 ROTC 指令的执行过程中，ⓓ+2 中指定的软元件变为 ON 时，ⓢ+0 将被清除。
应预先进行此清除操作之后，再通过 ROTC 指令开始就近控制。

(7) ⓓ+3 ～ⓓ+7 是用于进行旋转台动作控制的输出信号。
根据 ROTC 指令的执行结果，ⓓ+3 ～ⓓ+7 中的某个输出信号将变为 ON。

(8) ROTC 指令的执行指令变为 OFF 时，不执行就近控制，将ⓓ+3 ～ⓓ+7 全部置为 OFF。

(9) 在所有正在执行的程序中只能使用 1 次 ROTC 指令。
如果使用了 2 次或以上，将无法正常执行。

(10) ⓢ+0 ～ⓢ+2 或者 n2 的值大于 n1 的值时，将执行无处理。

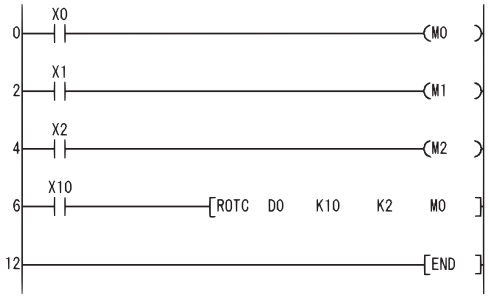
出　错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。
· ⓢ或者ⓓ中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)
(出错代码： 4101)

程序示例

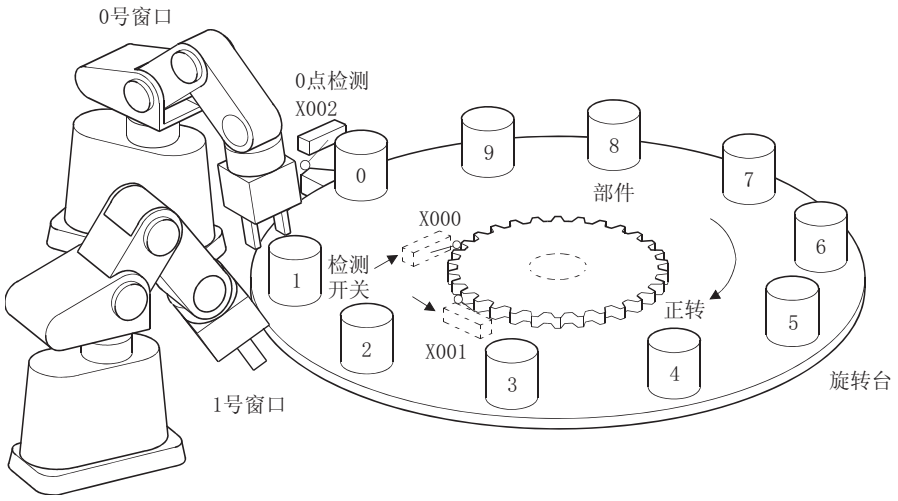
(1) 以下为将放置在 10 等份的旋转台上的 D2 编号的部件在 D1 编号窗口处进行取放，使旋转台在前后 2 个分区低速旋转时，求出马达的旋转方向及控制速度的程序。

[梯形图模式]

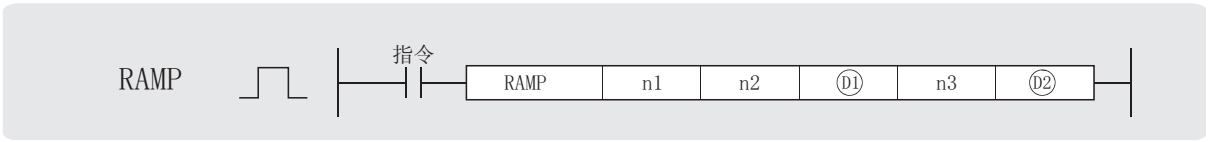


[列表模式]

步	指令	软元件
0	LD	X0
1	OUT	M0
2	LD	X1
3	OUT	M1
4	LD	X2
5	OUT	M2
6	LD	X10
7	ROTC	D0 K10 K2 M0
12	END	



6.8.6 斜坡信号 (RAMP)



n1 ： 初始值 (BIN16 位)。
n2 ： 最终值 (BIN16 位)。
D1 ： D1+0：当前值 (BIN16 位)。
D1+1：执行次数 (BIN16 位)。
n3 ： 移位次数 (BIN16 位)。
D2 ： D2+0：结束软元件 (位)。
D2+1：结束时数据保持选择位 (位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
n1	○			○				○	--
n2	○			○				○	--
D1	○			○				--	--
n3	○			○				○	--
D2	○			--				--	--

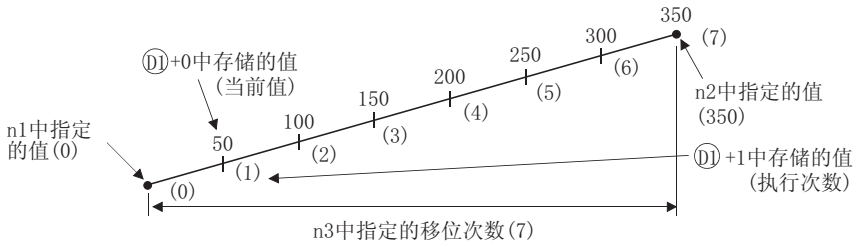
功 能

(1) 执行指令变为 ON 时，进行以下处理：

- 从 n1 中指定的值开始至 n2 中指定的值为止，按 n3 中指定的次数进行移位。
- 在 n3 中，对从 n1 处移位至 n2 处的扫描次数 (移位次数) 进行指定。
  如果不符合 0<n3<32768 的条件，则进行无处理。
- D1+1 为系统所用，用于存储本指令的执行次数。
- 1 次 (1 个扫描) 的变化值通过下列公式进行计算：

$$1次的变化值=\frac{(n2中指定的值)-(n1中指定的值)}{(n3中指定的值)}$$

例 通过 7 个扫描从 0 变为 350 时的示意图如下所示。



如果使用算出的 1 次的变化值不能整除，则需要进行补偿，以便按 n3 中指定的移位次数达到 n2 中指定的值。
因此有可能导致不能生成直线斜坡。

(2) 执行了 n3 中指定的移位次数的扫描后，(D2)+0 中指定的结束软元件将变为 ON。

结束软元件的 ON/OFF 状态以及(D1)+0 的值取决于(D2)+1 中指定的软元件的 ON/OFF 状态。

· (D2)+1 变为 OFF 时，在下 1 个扫描中将(D2)+0 变为 OFF 后，RAMP 指令将从初始值开始再次进行移位。

· (D2)+1 变为 ON 时，(D2)+0 保持为 ON 状态不变，(D1)+0 的值不发生变化。

(3) 如果在本指令的执行过程中其执行指令变为 OFF，则此后的(D1)+0 的值将不发生变化。
其执行指令再次变为 ON 时，RAMP 指令将从初始值开始再次进行移位。

(4) 在(D2)+0 中指定的结束软元件变为 ON 之前，不要对 n1 及 n2 的值进行变更。

由于每个扫描是以相同的计算公式算出(D1)+1 中存储的值，因此如果对 n1 及 n2 的值进行了变更，有可能导致急剧变化。

出　错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· (D1)或者(D2)中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)

(出错代码 : 4101)

注意事项

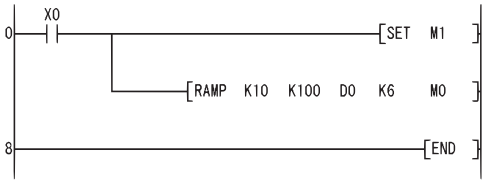
(1) 对(D1)进行了位软元件的位数指定时，只有在满足以下条件的情况下其指定才有效。

· 位数的指定 : K8

程序示例

(1) 以下为 X0 变为 ON 时，将 D0 的值以 6 个扫描从 10 变为 100，在移位结束时对 D0 的内容进行保持的程序。

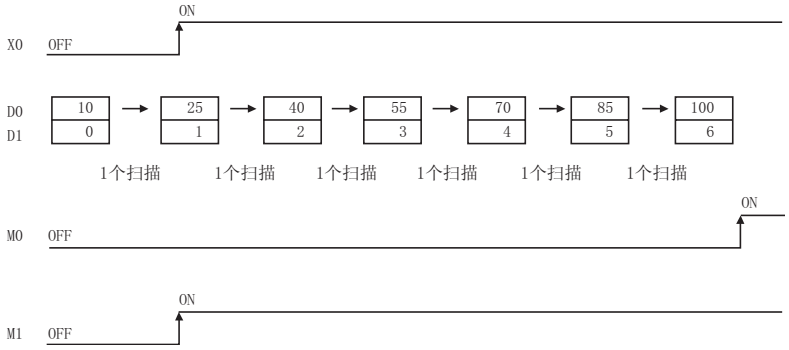
[梯形图模式]



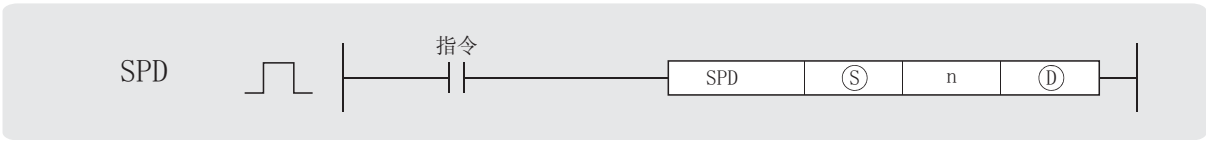
[列表模式]

步	指令	软元件
0	LD	X0
1	SET	M1
2	RAMP	K10 K100 D0 K6 M0
8	END	

[时序图]



6.8.7 脉冲密度测定 (SPD)



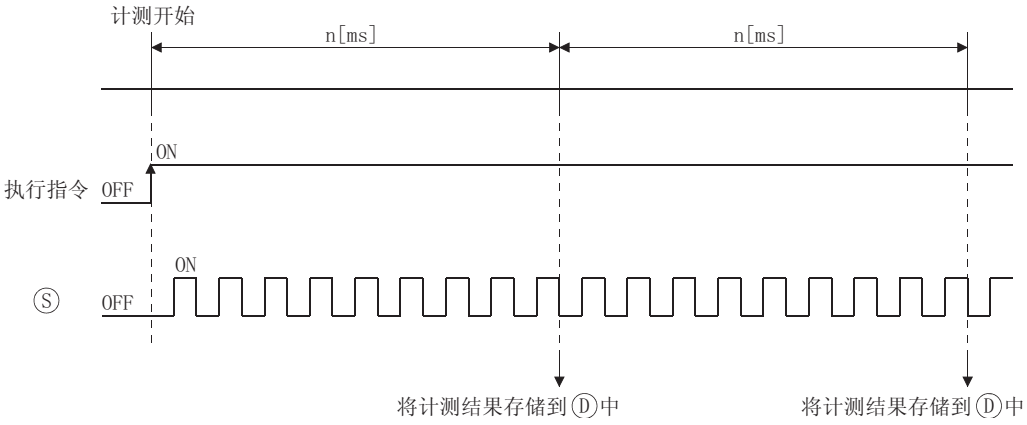
Ⓢ : 脉冲输入 (位)。
n : 测定时间 (单位 : ms)(BIN16 位)。
Ⓓ : 存储测定结果的软元件的起始编号 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○ (仅 X)	--	--	--					--
n	△ *1	△ *1	○						--
Ⓓ	--	△ *1	--	--					--

*1: 不能使用局部软元件及各程序中设置的文件寄存器。

功 能

(1) 将Ⓢ中指定的软元件输入的 OFF → ON 的次数在 n 中指定的时间内进行计数，并将计数结果存储到Ⓓ中指定的软元件中。



(2) 如果通过 SPD 指令进行的计测结束，将再次从 0 开始进行计测。
中止通过 SPD 指令进行的计测时，应将执行指令置于 OFF。

要 点

1.SPD 指令将变量软元件中的数据登录到 CPU 模块的工作区后，实际的计数动作将通过系统中断进行。
（将执行指令置于 OFF 或者 CPU 模块 STOP → RUN 时，CPU 模块的工作区中登录的软元件数据将被清除）。
因此，可计数的脉冲的 ON 和 OFF 时间必须比 CPU 模块的中断间隔长。
CPU 模块的中断间隔如下所示。

CPU 模块型号	中断间隔
高性能型 QCPU、过程 CPU、通用型 QCPU	1ms

2.使用高性能型 QCPU 或过程 CPU 时：
当 n=0 时，执行无处理。
3.在所有正执行的程序中，SPD 指令最多可以使用 6 次。
第 7 个以及以后的 SPD 指令将执行无处理。
4.在通过 SPD 指令进行测量的过程中（指令输入处于 ON 状态），不能进行设置值的变更。
在进行设置值变更时，应先将指令输入置于 OFF 之后再进行变更。

出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

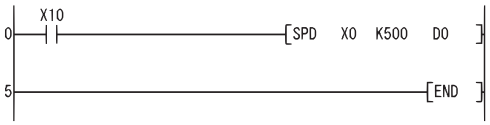
· Ⓢ中指定的软元件超出了相应软元件的范围时。（仅通用型 QCPU）

（出错代码： 4101）

程序示例

(1) 以下为 X10 变为 ON 时，在 500ms 的时间内对输入到 X0 中的脉冲进行计测，并将结果存储到 D0 中的程序。

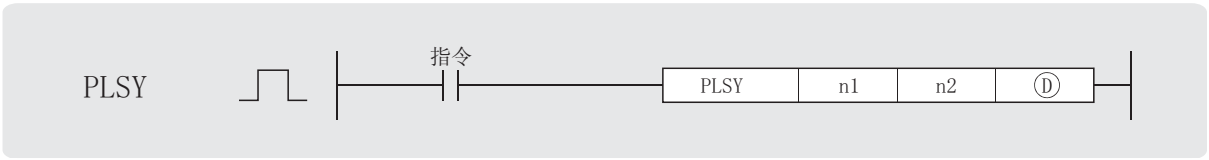
[梯形图模式]



[列表模式]

步	指令	软元件		
0	LD	X10		
1	SPD	X0	K500	D0
5	END			

6.8.8 恒定周期脉冲输出 (PLSY)



n1 ： 频率或者存储频率的软元件编号 (BIN16 位)。
n2 ： 输出次数或者存储输出次数的软元件编号 (BIN16 位)。
Ⓓ ： 脉冲输出的软元件编号 (位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
n1	○				○				--
n2	○				○				--
Ⓓ	△*1				--				--

*1: 仅输出 (Y) 可用。

功 能

(1) 将 n1 中指定频率的脉冲按 n2 中指定的次数存储到Ⓓ中指定的输出编号 (Y) 的输出模块中。

(2) n1 的可设置频率范围为 1Hz ～ 100Hz。
n1 的值超出了 1Hz ～ 100Hz 的范围时，将执行无处理。

(3) n2 的输出次数的可设置范围为 0 ～ 65535(0000H ～ FFFFH)。
n2 的值为 0 时，将连续输出脉冲。

(4) Ⓓ中指定的脉冲输出只能指定与输出模块相对应的输出号。

(5) 通过 PLSY 指令的执行指令的上升沿开始脉冲输出。
PLSY 指令的执行指令变为 OFF 时，脉冲输出将停止。

要 点

1.PLSY 指令将变量软元件数据登录到 CPU 模块的工作区后，实际的输出动作将通过系统中断进行处理。
（将执行指令置于 OFF 或者 CPU 模块 STOP → RUN 时，CPU 模块的工作区中登录的软元件数据将被清除）。
因此，可输出的脉冲的 ON 和 OFF 时间必须比 CPU 模块的中断间隔长。
CPU 模块的中断间隔如下所示。

CPU 模块型号	中断间隔
高性能型 QCPU、过程 CPU、通用型 QCPU	1 ms

2.在通过 PLSY 指令进行的脉冲输出过程中（指令输入处于 ON 状态），不要对 PLSY 指令的变量进行变更。
进行变量变更时，应先将指令输入置于 OFF 之后再进行变更。
3.在 CPU 模块的正在执行的全部程序中，只能使用 1 次 PLSY 指令。
第 2 次及以后的 PLSY 指令将被执行无处理。

出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

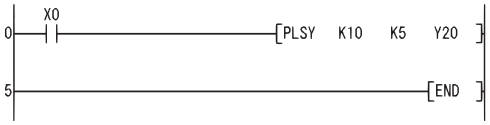
· ⓓ中指定的软元件超出了相应软元件的范围时。（仅通用型 QCPU）

（出错代码： 4101）

程序示例

(1) 以下为 X0 变为 ON 时，将 10Hz 的脉冲输出 5 次到 Y20 中的程序。

[梯形图模式]

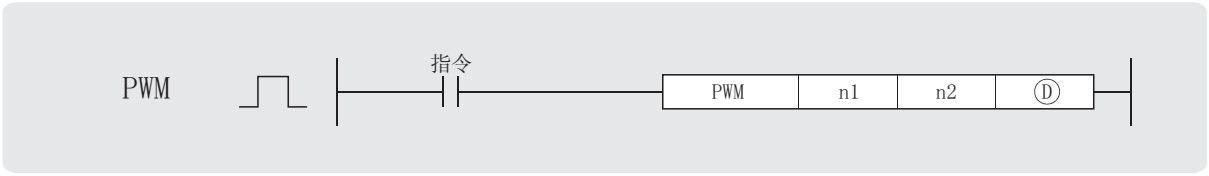


[列表模式]

步	指令	软元件		
0	LD	X0		
1	PLSY	K10	K5	Y20
5	END			

6.8.9 脉冲宽度调制 (PWM)

Basic High performance Process Redundant Universal



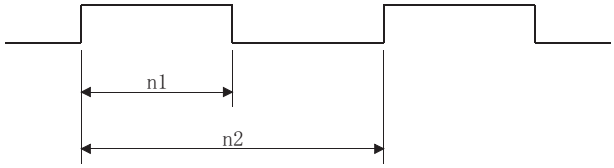
n1 : ON 状态时间或存储 ON 状态时间的软元件编号 (BIN16 位)。
n2 : 周期或存储周期的软元件编号 (BIN16 位)。
Ⓓ : 脉冲输出的软元件编号 (位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
n1	○			○					--
n2	○			○					--
Ⓓ	△*1			--					--

*1: 仅输出 (Y) 可用。

功 能

(1) 将 n1 中指定的 ON 状态时间及 n2 中指定的周期脉冲输入到Ⓓ中指定的输出模块中。



(2) n1、n2 的设置时间如下所示：

CPU 模块型号	n1、n2 的设置范围 [ms]*2
高性能型 QCPU、过程 CPU、通用型 QCPU	1 ～ 65535(0001H ～ FFFFH)

*2: n1 中指定的值应小于 n2 中指定的值。

要 点

1.PWM 指令将变量软元件数据登录到 CPU 模块的工作区后，实际的输出动作将通过系统中断进行处理。
（将执行指令置于 OFF 或者 CPU 模块 STOP → RUN 时，CPU 模块的工作区中登录的软元件数据将被清除）。
CPU 模块的中断间隔如下所示。

CPU 模块型号	中断间隔
高性能型 QCPU、过程 CPU、通用型 QCPU	1ms

因此，在 CPU 模块正在执行的全部程序中，只能使用 1 次 PWM 指令。

2.在以下情况下将执行无处理：
- n1、n2 为 0 时。
- n1、n2 不为 5 的倍数时（仅使用 QnACPU 时）。
- n1 ≥ n2 时。
- 执行了 2 次或以上的 PWM 指令时。

3.在通过 PWM 指令进行的脉冲输出过程中（指令输入处于 ON 状态），不要对 PWM 指令的变量进行变更。
进行变量变更时，应先将执行指令置于 OFF 之后再进行变更。

出 错

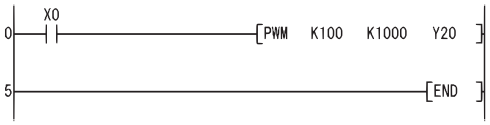
(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。
- ⓓ中指定的软元件超出了相应软元件的范围时。（仅通用型 QCPU）

（出错代码： 4101）

程序示例

(1) 以下为 X0 变为 ON 时，将 100ms 的脉冲每隔 1 秒输出到 Y20 中的程序。

[梯形图模式]

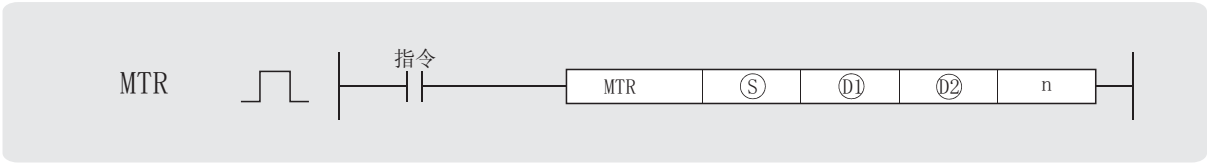


[列表模式]

步	指令	软元件		
0	LD	X0		
1	PWM	K100	K1000	Y20
5	END			

6.8.10 矩阵输入 (MTR)

Basic High performance Process Redundant Universal



Ⓢ ： 输入的起始编号 (位)。
Ⓓ1 ： 输出的起始编号 (位)。
Ⓓ2 ： 存储矩阵输入数据的软元件的起始编号 (位)。
n ： 输入列数 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○ (仅 X)		--						--
Ⓓ1	○ (仅 Y)		--						--
Ⓓ2	○		--						--
n	○		○						--

功 能

(1) 对Ⓢ中指定的输入编号后面连接的 16 点× n 列的输入依次进行读取，将读取的输入数据存储到Ⓓ2中指定的软元件后面。

(2) 在 1 个扫描中可以读取 1 列 (16 点)。

(3) 按从第 1 列至第 n 列的顺序反复进行读取。

(4) Ⓓ2中指定的软元件后面，从起始开始的 16 点存储第 1 列数据，下一个 16 点存储第 2 列的数据。
因此，从Ⓓ2中指定的软元件开始的 16 × n 点被 MTR 指令所占用。

(5) Ⓓ1是用于选择执行读取的列的输出，由系统自动地进行 ON/OFF。
该软元件占用从Ⓓ1中指定的软元件开始的 n 点。

(6) Ⓢ、Ⓓ1、Ⓓ2中只能指定 16 的倍数的软元件号。

(7) n 值的指定范围为 2 ～ 8。

(8) 在以下情况下将执行无处理：

·Ⓢ、Ⓓ1、Ⓓ2中指定的软元件号不为 16 的倍数时。

·Ⓢ中指定的软元件超出了实际输入范围时。

·Ⓓ1中指定的软元件超出了实际输出范围时。

·Ⓓ2中指定的软元件后面的 16 × n 点超出了相应软元件范围时。

· n 值超出了 2 ～ 8 的范围时。

出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

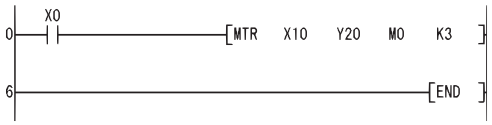
· ⓢ中指定了除输入 (X) 以外的软元件时。 (出错代码： 4101)

· ⓓ1中指定了除输出 (Y) 以外的软元件时。 (出错代码： 4101)

程序示例

(1) 以下为 X0 变为 ON 时， 对 X10 后面连接的 16 点× 3 列的矩阵进行读取后，将结果存储到 M0 后面的程序。

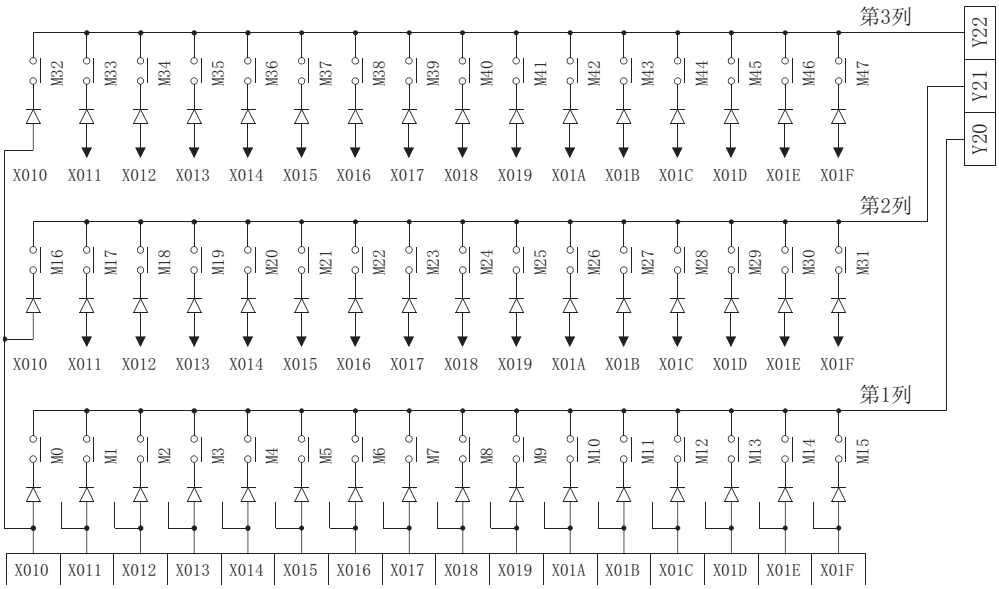
[梯形图模式]



[列表模式]

步	指令	软元件			
0	LD	X0			
1	MTR	X10	Y20	M0	K3
6	END				

[动作]



注意事项

(1) 由于 MTR 指令直接对输入输出进行操作，因此应加以注意。

即使 MTR 指令变为 OFF，通过 MTR 指令变为 ON 的输出ⓓ1也不会变为 OFF。

应通过顺控程序将ⓓ1中指定的输出置于 OFF。

(2) MTR 指令的执行间隔必须长于输入模块与输出模块的响应时间的合计值。

如果 MTR 指令的执行间隔设置短于上述时间，则无法正常地读取输入。

如果顺序程序的扫描时间过短，则应通过恒定扫描将扫描时间改为长于响应时间的合计值。

7 应用指令

分类	处理内容	参阅章节
逻辑运算指令	逻辑和、逻辑积等逻辑运算	7.1 节
旋转指令	指定数据的旋转移动	7.2 节
移位指令	指定数据的移位	7.3 节
位处理指令	位数据的设置 / 复位、位提取	7.4 节
数据处理指令	数据查找、排序、解码、编码等数据处理	7.5 节
结构化指令	重复运算、子程序调用、梯形图单位的变址修饰	7.6 节
数据表操作指令	数据表的读 / 写	7.7 节
缓冲存储器访问指令	智能功能模块的缓冲存储器的读 / 写	7.8 节
显示指令	将字符代码输出到外部，并显示在显示器上	7.9 节
调试、故障诊断指令	检查、状态锁存、采样跟踪、程序跟踪	7.10 节
字符串处理指令	字符串 (ASCII 码数据) 的处理	7.11 节
特殊函数指令	BCD 型实数处理、浮点实数处理	7.12 节
数据控制指令	基于输入数据范围检查的输出值控制	7.13 节
文件寄存器切换指令	文件寄存器的设置、块号切换	7.14 节
时钟指令	时钟数据的读 / 写、时钟比较、日期比较	7.15 节
外围设备用指令	至外围设备的信息显示、按键输入	7.16 节
程序控制指令	程序执行条件的切换指令	7.16 节
其它指令	看门狗定时器复位指令、计时时钟指令等未列入上述分类的指令。	7.17 节

7.1 逻辑运算指令

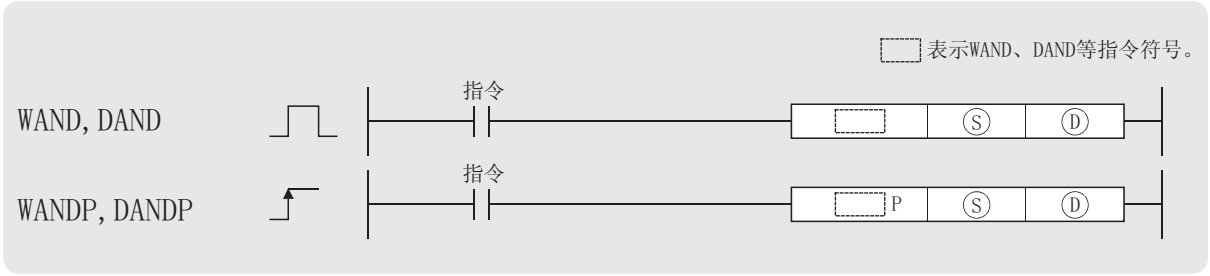
(1) 逻辑运算指令以 1 位为单位进行逻辑和及逻辑积等逻辑运算。

分类	处理内容	运算公式	示例		
			A	B	Y
逻辑积 (AND)	只有当输入 A 和输入 B 都为 1 时才为 1，除此以外均为 0。	$Y=A\cdot B$	0	0	0
			0	1	0
			1	0	0
			1	1	1
逻辑和 (OR)	只有当输入 A 和输入 B 都为 0 时才为 0，除此以外均为 1。	$Y=A+B$	0	0	0
			0	1	1
			1	0	1
			1	1	1
排他逻辑和 (XOR)	如果输入 A 和输入 B 相等，则变为 0，不相等时变为 1。	$Y=\bar{A}\cdot B+A\cdot\bar{B}$	0	0	0
			0	1	1
			1	0	1
			1	1	0
否定排他逻辑和 (XNR)	如果输入 A 和输入 B 相等，则变为 1，不相等时变为 0。	$Y=(\bar{A}+B)(A+\bar{B})$	0	0	1
			0	1	0
			1	0	0
			1	1	1

7.1.1 16 位 /32 位数据的逻辑积 (WAND(P)、DAND(P))

Basic | High performance | Process | Redundant | Universal

1 设置数据为 2 个时 (Ⓓ∧Ⓢ→Ⓓ，(Ⓓ+1，Ⓓ) ∧ (Ⓢ+1，Ⓢ) → (Ⓓ+1，Ⓓ))



Ⓢ：执行逻辑积的数据或者存储数据的软元件的起始编号 (BIN16/32 位)。

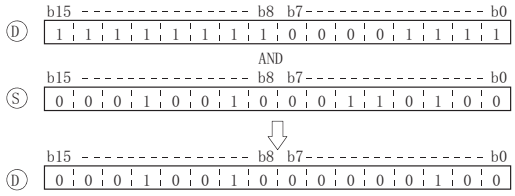
Ⓓ：存储逻辑积结果的软元件的起始编号 (BIN16/32 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○							○	--
Ⓓ	○							--	--

功 能

WAND

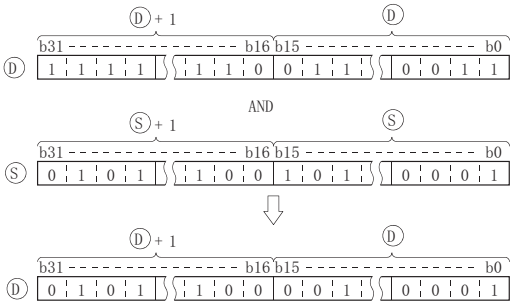
(1) 将Ⓓ中指定的软元件的 16 位数据与Ⓢ中指定的软元件的 16 位数据逐位进行逻辑积运算，并将结果存储到Ⓓ中指定的软元件中。



(2) 位软元件的情况下，位数指定点数以后的位软元件将被作为 0 进行运算。(参阅程序示例 (2))

DAND

(1) 将Ⓓ中指定的软元件的 32 位数据与Ⓢ中指定的软元件的 32 位数据逐位进行逻辑积运算，并将结果存储到Ⓓ中指定的软元件中。



(2) 位软元件的情况下，位数指定点数以后的位软元件将被作为 0 进行运算。(参阅程序示例 (2))

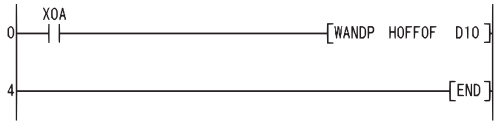
出 错

(1) 在 WAND(P)、DAND(P) 指令中无运算出错。

程序示例

(1) 以下为 XA 变为 ON 时，将 D10 的 BCD4 位数值中的十位 (从低位起第 2 位) 上的值屏蔽为 0 的程序。

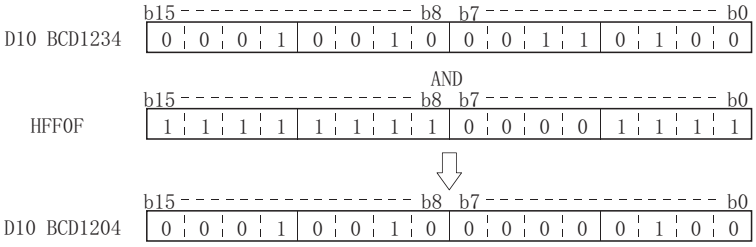
[梯形图模式]



[列表模式]

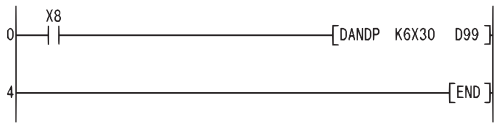
步	指令	软元件	
0	LD	XOA	
1	WANDP	HOFFOF	D10
4	END		

[动作]



(2) 以下为 X8 变为 ON 时，将 D99、D100 的数据与 X30 ～ X47 的 24 位数据进行逻辑积运算，并将结果存储到 D99、D100 中的程序。

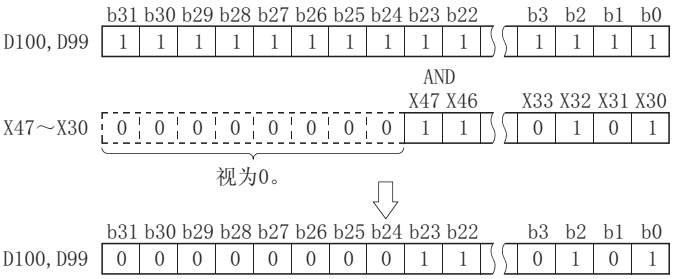
[梯形图模式]



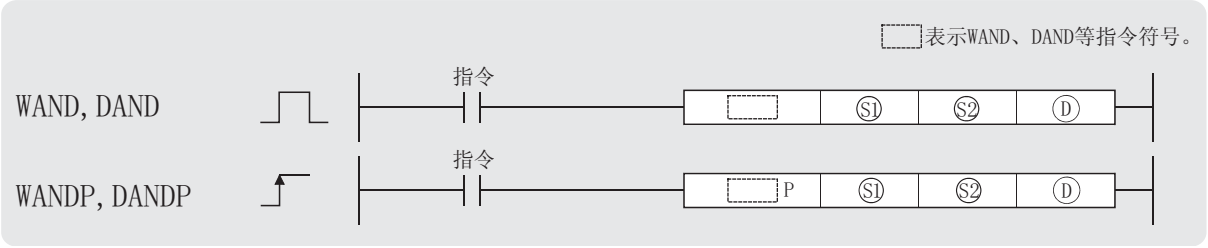
[列表模式]

步	指令	软元件	
0	LD	X8	
1	DANDP	K6X30	D99
4	END		

[动作]



2 设置数据为 3 个时 (Ⓢ1 ∧Ⓢ2→Ⓓ，(Ⓢ1+1，Ⓢ1) ∧ (Ⓢ2+1，Ⓢ2) → (Ⓓ+1，Ⓓ))



Ⓢ1，Ⓢ2 ：执行逻辑积的数据或者存储数据的软元件的起始编号 (BIN16/32 位)。

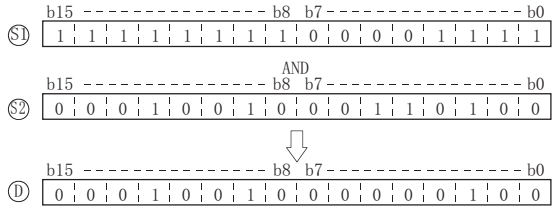
Ⓓ ：存储逻辑积结果的软元件的起始编号 (BIN16/32 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ1	○							○	--
Ⓢ2	○							○	--
Ⓓ	○							--	--

功 能

WAND

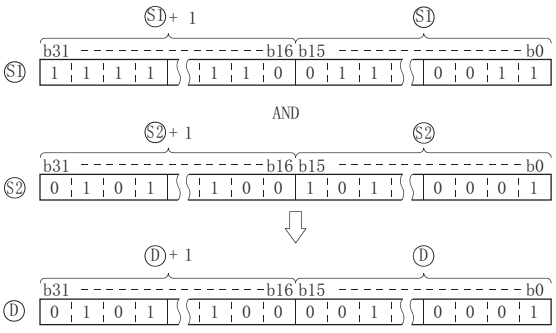
(1) 将Ⓢ1中指定的软元件的 16 位数据与Ⓢ2中指定的软元件的 16 位数据逐位进行逻辑积运算，并将结果存储到Ⓓ中指定的软元件中。



(2) 位软元件的情况下，位数指定点数以后的位软元件将被作为 0 进行运算。(参阅程序示例 (1),(2))

DAND

(1) 将Ⓢ1中指定的软元件的 32 位数据与Ⓢ2中指定的软元件的 32 位数据逐位进行逻辑积运算，并将结果存储到Ⓓ中指定的软元件中。



(2) 位软元件的情况下，位数指定点数以后的位软元件将被作为 0 进行运算。(参阅程序示例 (3))

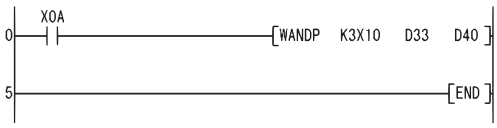
出　错

(1) 在 WAND(P)、DAND(P) 指令中无运算出错。

程序示例

(1) 以下为 XA 变为 ON 时，将 X10 ～ X1B 的数据与 D33 进行逻辑积运算，并将其结果存储到 D40 中的程序。

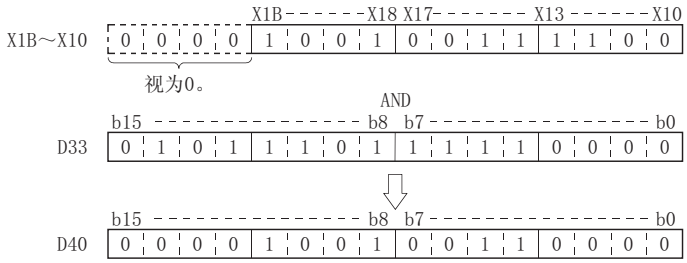
[梯形图模式]



[列表模式]

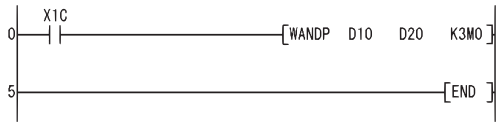
步	指令	软元件		
0	LD	X0A		
1	WANDP	K3X10	D33	D40
5	END			

[动作]



(2) 以下为 X1C 变为 ON 时，将 D10 与 D20 的数据进行逻辑积运算，并将结果存储到 M0 ～ M11 中的程序。

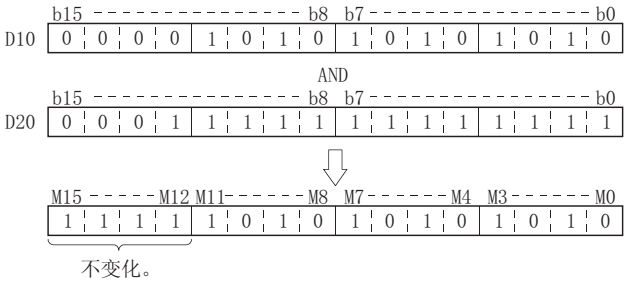
[梯形图模式]



[列表模式]

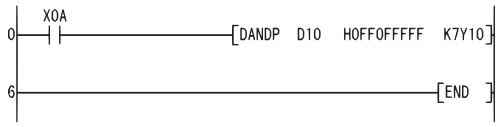
步	指令	软元件		
0	LD	X1C		
1	WANDP	D10	D20	K3M0
5	END			

[动作]



(3) 以下为 XA 变为 ON 时，将 D10、D11 的 BCD8 位数值中的十万位（从低位起第 6 位）上的值屏蔽为 0，并将运算结果存储到 Y10 ～ Y2B 中的程序。

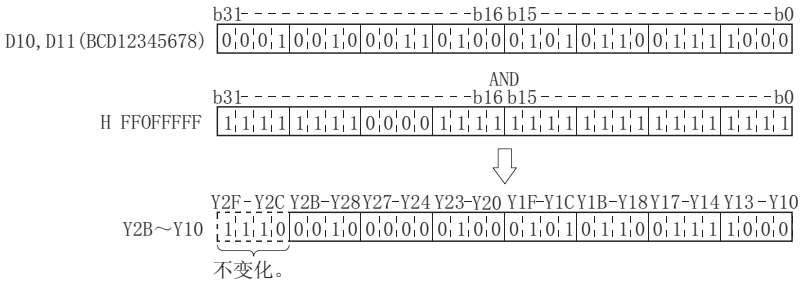
[梯形图模式]



[列表模式]

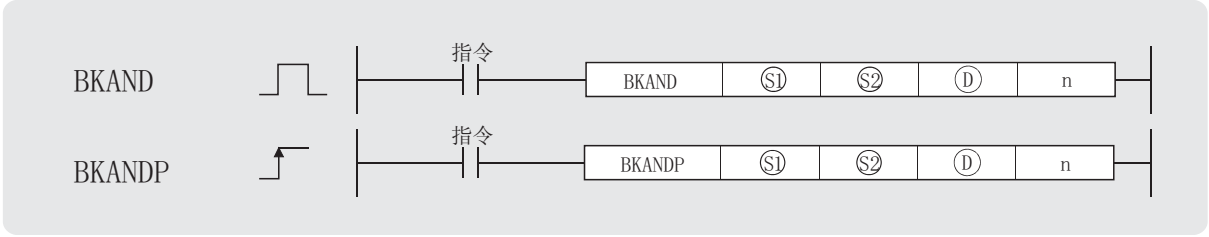
步	指令	软元件
0	LD	XOA
1	DANDP	D10 HOFFOFFFFF K7Y10
6	END	

[动作]



7.1.2 块逻辑积 (BKAND(P))

Basic | High performance | Process | Redundant | Universal



(S1)*1 ：存储执行逻辑积的数据的软元件的起始编号 (BIN16 位)。

(S2)*1 ：执行逻辑积的数据或者存储数据的软元件的起始编号 (BIN16 位)。

(D)*1 ：存储逻辑积结果的软元件的起始编号 (BIN16 位)。

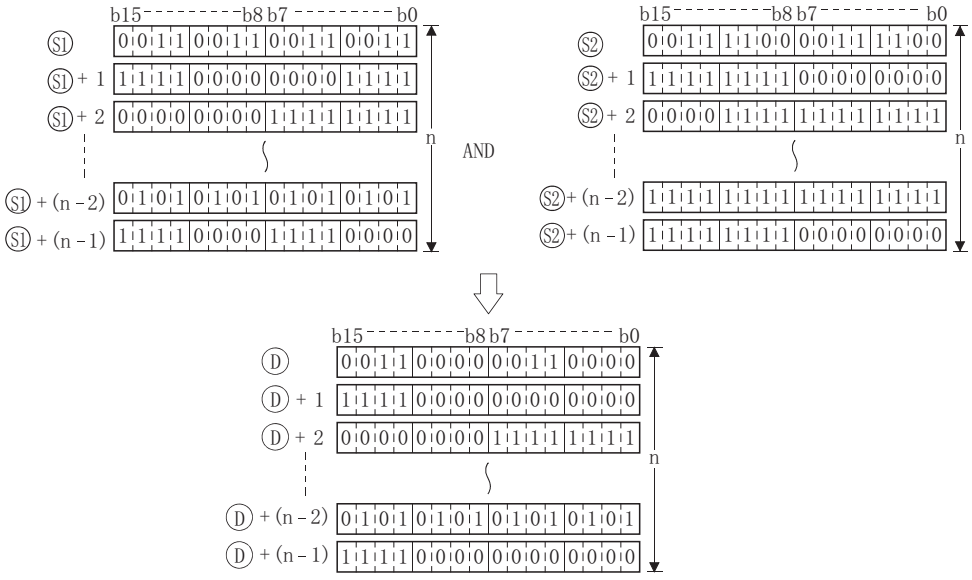
n ：运算数据个数 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
(S1)*1	--	○		--				--	--
(S2)*1	--	○		--				○	--
(D)*1	--	○		--				--	--
n	○	○		○				○	--

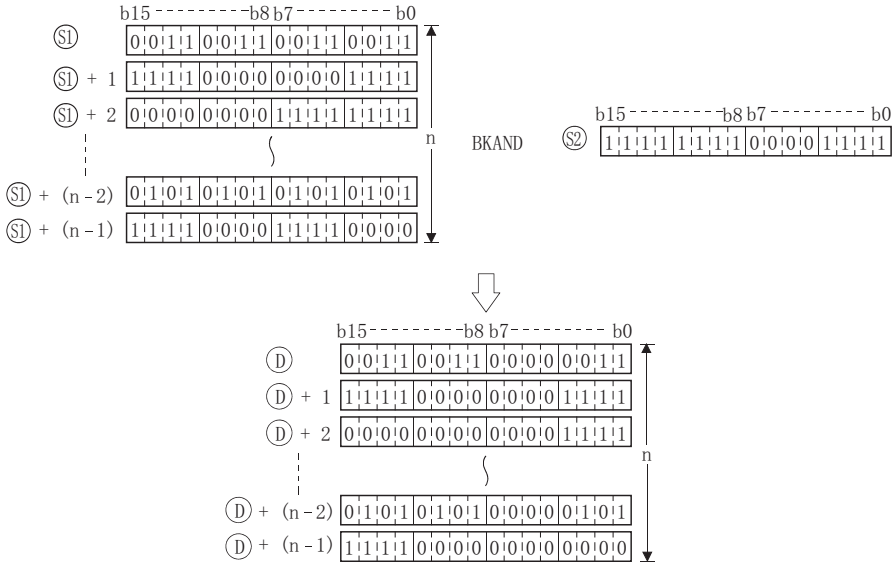
＊1:(S1)与(D)或者(S2)与(D)可以指定为相同的软元件编号。

功 能

(1) 将从(S1)中指定的软元件开始的 n 点的内容与从(S2)中指定的软元件开始的 n 点的内容执行逻辑积运算，并将结果存储到(D)中指定的软元件后面。



(2) ⓢ2中可以指定为 -32768 ～ 32767(BIN16 位) 的常数。



出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 从ⓢ1、ⓢ2、ⓓ的软元件开始的 n 点的范围超出了相应软元件范围时。
(出错代码 :4101)

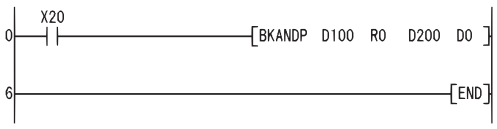
· 从ⓢ1开始至第 n 点为止的软元件范围与从ⓓ开始的至第 n 点为止的软元件范围有部分重复时。
(ⓢ1与ⓓ中指定了同一个软元件时除外。) (出错代码 :4101)

· 从ⓢ2开始至第 n 点为止的软元件范围与从ⓓ开始的至第 n 点为止的软元件范围相重复时。
(ⓢ2与ⓓ中指定了同一个软元件时除外。) (出错代码 :4101)

程序示例

(1) 以下为 X20 变为 ON 时，将 D100 ～ D102 中存储的数据值与 R0 ～ R2 中存储的数据值进行逻辑积运算，并将其结果存储到 D200 后面的程序。

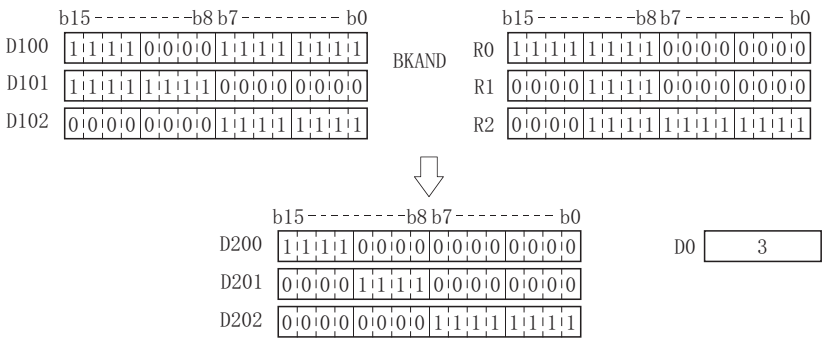
[梯形图模式]



[列表模式]

步	指令	软元件			
0	LD	X20			
1	BKANDP	D100	R0	D200	D0
6	END				

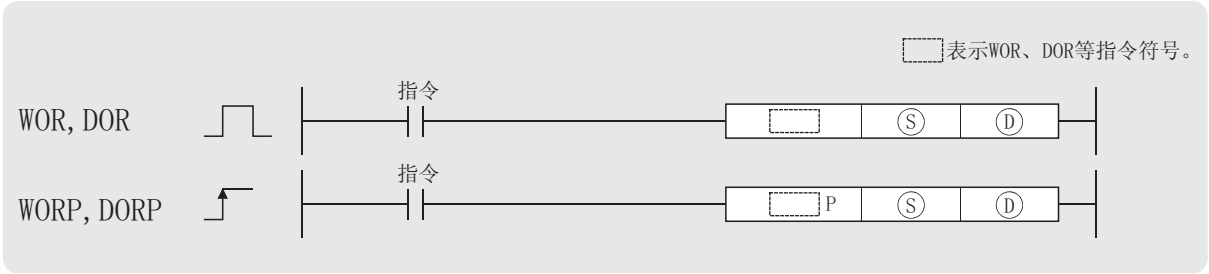
[动作]



7.1.3 16 位 /32 位数据的逻辑和 (WOR(P)、DOR(P))

Basic | High performance | Process | Redundant | Universal

1 设置数据为 2 个时 (ⓓ ∨ Ⓢ → ⓓ，(ⓓ+1，ⓓ) ∨ (Ⓢ+1，Ⓢ) → (ⓓ+1，ⓓ))



Ⓢ ：执行逻辑和的数据或者存储数据的软元件的起始编号 (BIN16/32 位)。

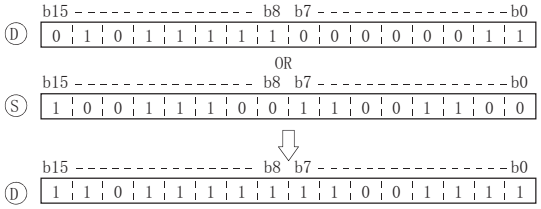
ⓓ ：存储逻辑和结果的软元件的起始编号 (BIN16/32 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○							○	--
ⓓ	○							--	--

功 能

WOR

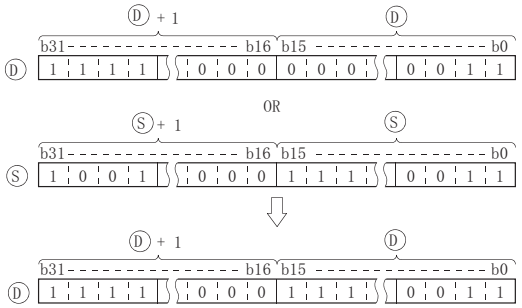
(1) 将ⓓ中指定的软元件的 16 位数据与Ⓢ中指定的软元件的 16 位数据逐位进行逻辑和运算，并将结果存储到ⓓ中指定的软元件中。



(2) 位软元件的情况下，位数指定点数以后的位软元件将被作为 0 进行运算。

DOR

(1) 将ⓓ中指定的软元件的 32 位数据与Ⓢ中指定的软元件的 32 位数据逐位进行逻辑和运算，并将结果存储到ⓓ中指定的软元件中。



(2) 位软元件的情况下，位数指定点数以后的位软元件将被作为 0 进行运算。

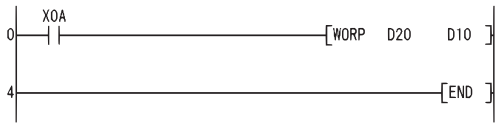
出 错

(1) 在 WOR(P)、DOR(P) 指令中无运算出错。

程序示例

(1) 以下为 XA 变为 ON 时，将 D10 的数据与 D20 的数据执行逻辑和运算，并将其结果存储到 D10 中的程序。

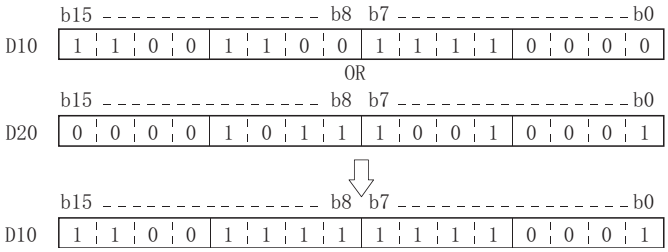
[梯形图模式]



[列表模式]

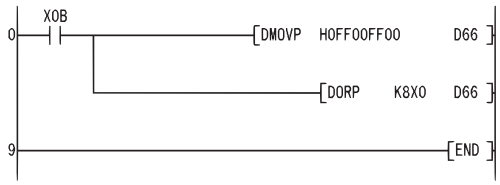
步	指令	软元件	
0	LD	X0A	
1	WORP	D20	D10
4	END		

[动作]



(2) 以下为 XB 变为 ON 时，将 X0 ～ X1F 的 32 位数据与 16 进制数 FF00FF00H 进行逻辑和运算，并将其结果存储到 D66、D67 中的程序。

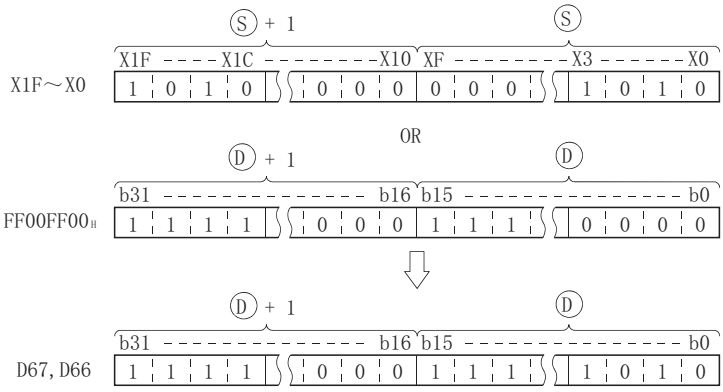
[梯形图模式]



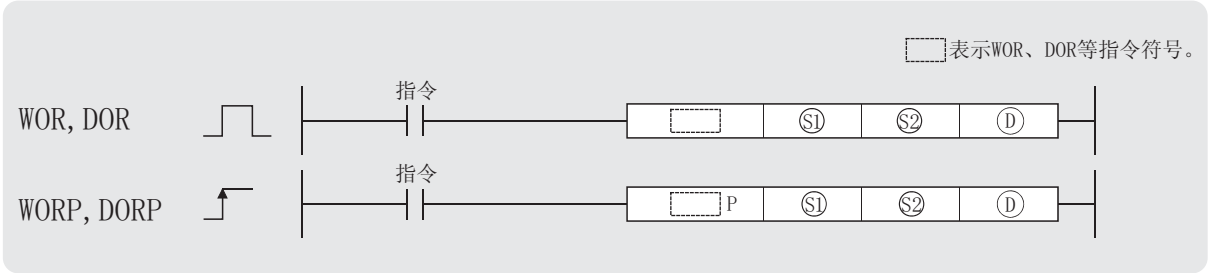
[列表模式]

步	指令	软元件	
0	LD	X0B	
1	DMOVP	HOFF00FF00	D66
4	DORP	K8X0	D66
9	END		

[动作]



2 设置数据为 3 个时。(Ⓢ1 ∨Ⓢ2→Ⓓ，(Ⓢ1+1，Ⓢ1) ∨ (Ⓢ2+1，Ⓢ2) → (Ⓓ+1，Ⓓ))



Ⓢ1，Ⓢ2 ：执行逻辑和的数据或者存储数据的软元件的起始编号 (BIN16/32 位)。

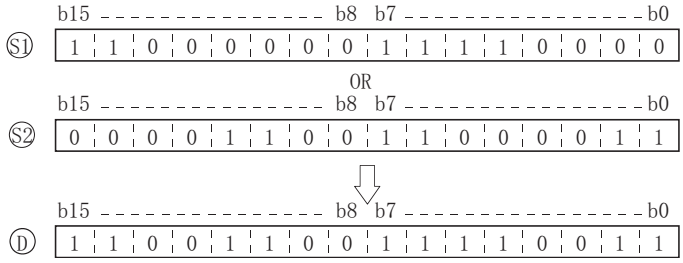
Ⓓ ：存储逻辑和结果的软元件的起始编号 (BIN16/32 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ1	○							○	--
Ⓢ2	○							○	--
Ⓓ	○							--	--

功 能

WOR

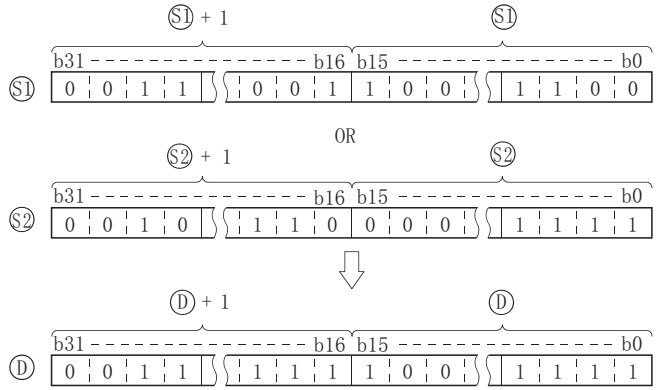
(1) 将Ⓢ1 中指定的软元件的 16 位数据与Ⓢ2 中指定的软元件的 16 位数据逐位进行逻辑和运算，并将结果存储到Ⓓ 中指定的软元件中。



(2) 位软元件的情况下，位数指定点数以后的位软元件将被作为 0 进行运算。(参阅程序示例 (1))

DOR

(1) 将Ⓢ1 中指定的软元件的 32 位数据与Ⓢ2 中指定的软元件的 32 位数据逐位进行逻辑和运算，并将结果存储到Ⓓ 中指定的软元件中。



(2) 位软元件的情况下，位数指定点数以后的位软元件将被作为 0 进行运算。(参阅程序示例 (2))

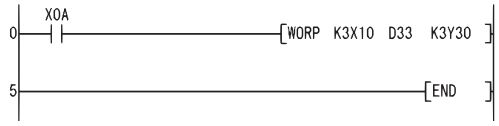
出 错

(1) 在 WOR(P)、DOR(P) 指令中无运算出错。

程序示例

(1) 以下为 XA 变为 ON 时，将 X10 ～ X1B 的数据与 D33 进行逻辑和运算，并将其结果存储到 Y30 ～ Y3B 中的程序。

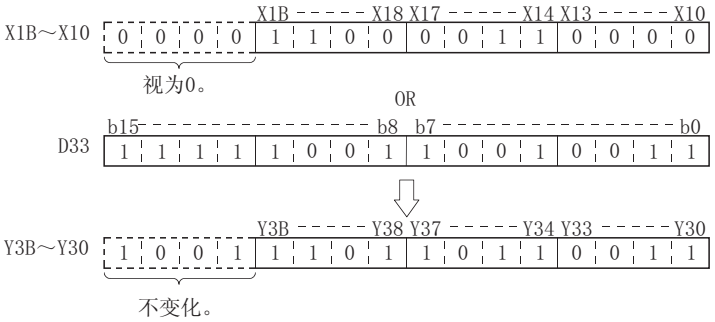
[梯形图模式]



[列表模式]

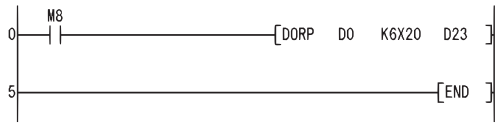
步	指令	软元件		
0	LD	XOA		
1	WORP	K3X10	D33	K3Y30
5	END			

[动作]



(2) 以下为 M8 变为 ON 时，将 D0、D1 的 32 位数据与 X20 ～ X37 的 24 位数据进行逻辑和运算，并将结果存储到 D23、D24 中的程序。

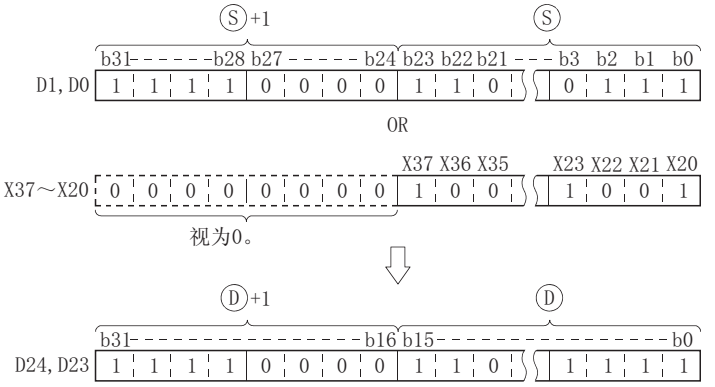
[梯形图模式]



[列表模式]

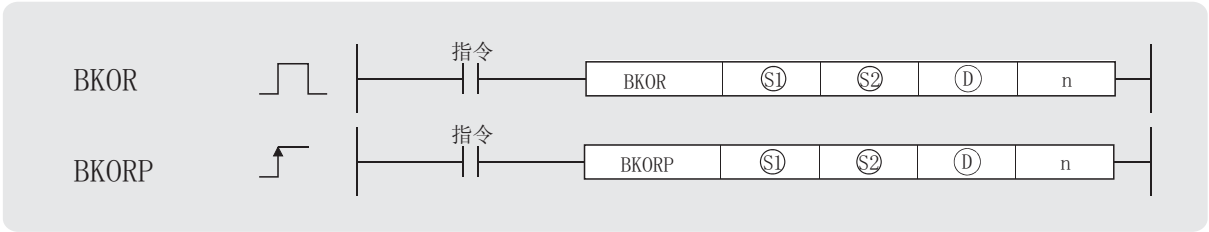
步	指令	软元件		
0	LD	M8		
1	DORP	D0	K6X20	D23
5	END			

[动作]



7.1.4 块逻辑和 (BKOR(P))

Basic | High performance | Process | Redundant | Universal



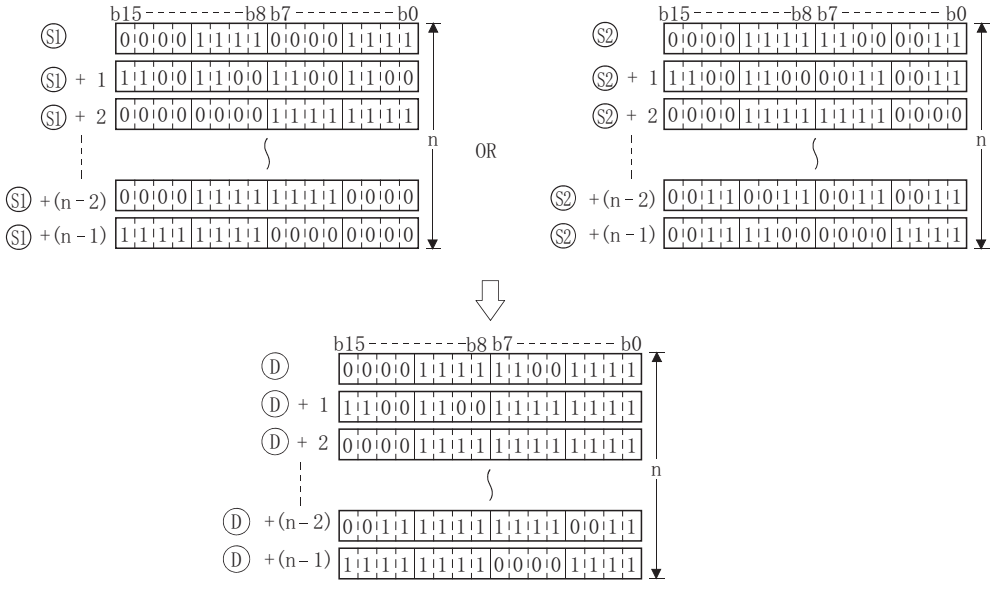
(S1)*1 ：存储执行逻辑和的数据的软元件的起始编号 (BIN16 位)。
(S2)*1 ：执行逻辑和的数据或者存储数据的软元件的起始编号 (BIN16 位)。
(D)*1 ：存储逻辑和结果的软元件的起始编号 (BIN16 位)。
n ：运算数据个数 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
(S1)*1	--	○		--				--	--
(S2)*1	--	○		--				○	--
(D)*1	--	○		--				--	--
n	○	○		○				○	--

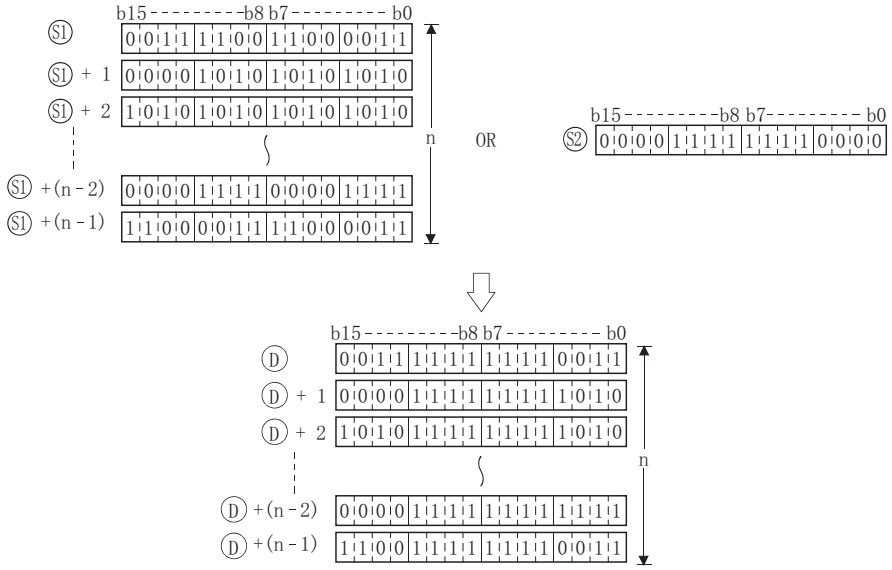
＊1:(S1)与(D)或者(S2)与(D)可以指定为相同的软元件编号。

功 能

(1) 将从(S1)中指定的软元件开始的 n 点的内容与从(S2)中指定的软元件开始的 n 点的内容执行逻辑和运算，并将结果存储到(D)中指定的软元件后面。



(2) ⓢ2中可以指定为 -32768 ～ 32767(BIN16 位) 的常数。



出　错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 从ⓢ1、ⓢ2、ⓓ的软元件开始的 n 点的范围超出了相应软元件范围时。
(出错代码 :4101)

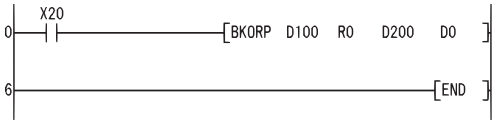
· 从ⓢ1开始至第 n 点为止的软元件范围与从ⓓ开始的至第 n 点为止的软元件范围有部分重复时。
(ⓢ1与ⓓ中指定了同一个软元件时除外。) (出错代码 :4101)

· 从ⓢ2开始至第 n 点为止的软元件范围与从ⓓ开始的至第 n 点为止的软元件范围相重复时。
(ⓢ2与ⓓ中指定了同一个软元件时除外。) (出错代码 :4101)

程序示例

(1) 以下为 X20 变为 ON 时，将 D100 ～ D102 中存储的数据值与 R0 ～ R2 中存储的数据值进行逻辑和运算，并将其结果存储到 D200 后面的程序。

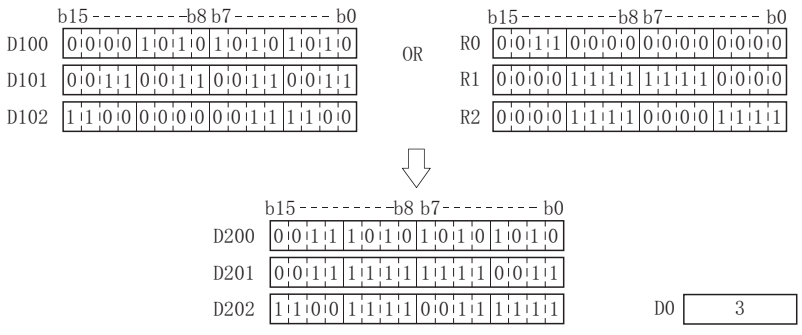
[梯形图模式]



[列表模式]

步	指令	软元件
0	LD	X20
1	BKORP	D100 R0 D200 D0
6	END	

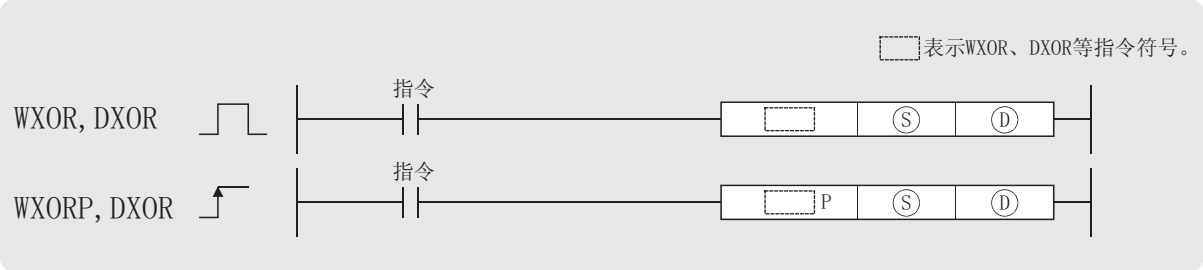
[动作]



7.1.5 16 位 /32 位数据排他逻辑和 (WXOR(P)、DXOR(P))

Basic | High performance | Process | Redundant | Universal

1 设置数据为 2 个时 ((Ⓓ) ∨ (Ⓢ)→(Ⓓ)，((Ⓓ)+1，(Ⓓ)) ∨ ((Ⓢ)+1，(Ⓢ)) → ((Ⓓ)+1，(Ⓓ))



Ⓢ：执行排他逻辑和的数据或者存储数据的软元件的起始编号 (BIN16/32 位)。

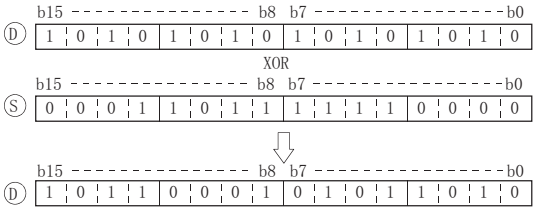
Ⓓ：存储排他逻辑和结果的软元件的起始编号 (BIN16/32 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○							○	--
Ⓓ	○							--	--

功 能

WXOR

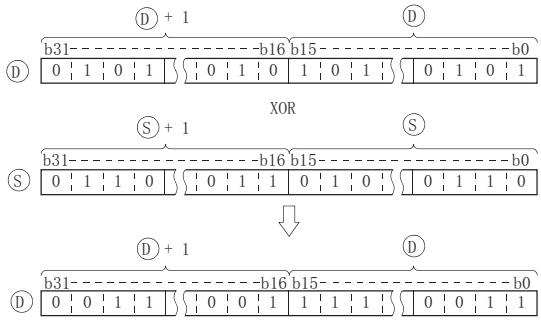
(1) 将Ⓓ中指定的软元件的 16 位数据与Ⓢ中指定的软元件的 16 位数据逐位进行排他逻辑和运算，并将结果存储到Ⓓ中指定的软元件中。



(2) 位软元件的情况下，位数指定点数以后的位软元件将被作为 0 进行运算。

DXOR

(1) 将Ⓓ中指定的软元件的 32 位数据与Ⓢ中指定的软元件的 32 位数据逐位进行排他逻辑和运算，并将结果存储到Ⓓ中指定的软元件中。



(2) 位软元件的情况下，位数指定点数以后的位软元件将被作为 0 进行运算。

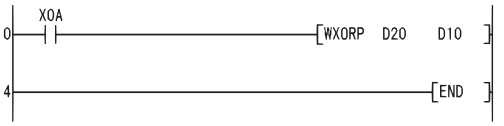
出 错

(1) 在 WXOR(P)、DXOR(P) 指令中无运算出错。

程序示例

(1) 以下为 XA 变为 ON 时，将 D10 的数据与 D20 的数据执行排他逻辑和运算，并将其结果存储到 D10 中的程序。

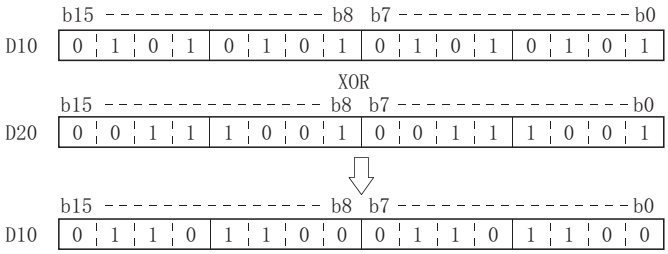
[梯形图模式]



[列表模式]

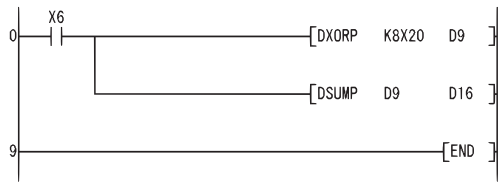
步	指令	软元件	
0	LD	X0A	
1	WXORP	D20	D10
4	END		

[动作]



(2) 以下为 X6 变为 ON 时，将 X20 ～ X3F 的 32 位数据与 D9、D10 的数据的位模式进行比较，并将不同的位数存储到 D16 中的程序。

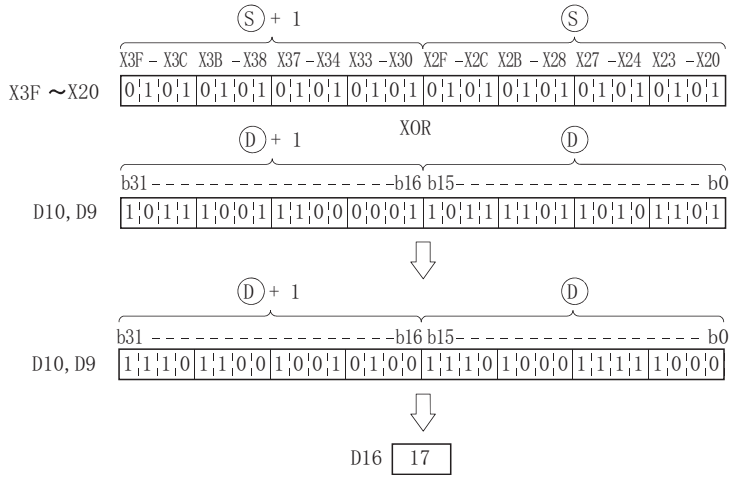
[梯形图模式]



[列表模式]

步	指令	软元件	
0	LD	X6	
1	DXORP	K8X20	D9
6	DSUMP	D9	D16
9	END		

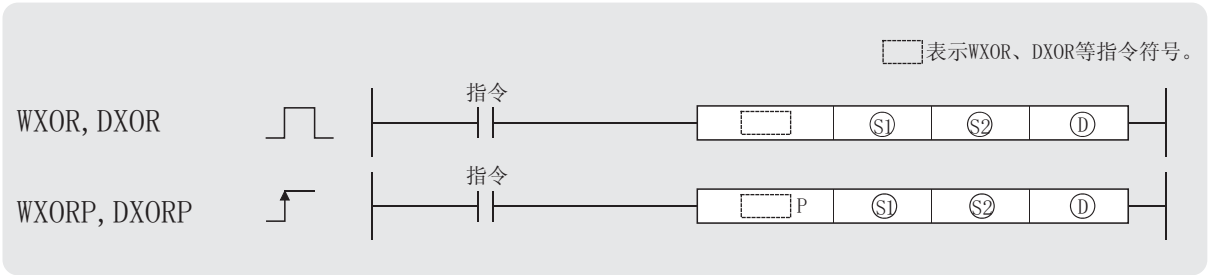
[动作]



备 注

关于 DSUMP 指令的有关内容，请参阅 7.5.2 项。

2 设置数据为 3 个时。(Ⓢ1 ∨ Ⓢ2→Ⓓ，(Ⓢ1+1，Ⓢ1) ∨ (Ⓢ2+1，Ⓢ2) → (Ⓓ+1，Ⓓ))



Ⓢ1，Ⓢ2 ：执行排他逻辑和的数据或者存储数据的软元件的起始编号 (BIN16/32 位)。

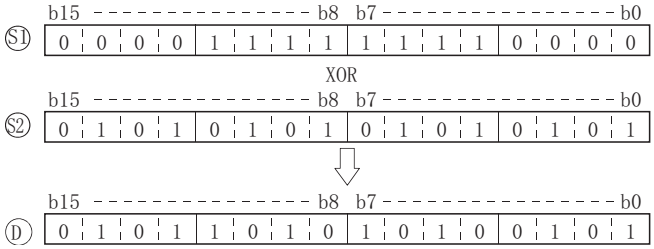
Ⓓ ：存储排他逻辑和结果的软元件的起始编号 (BIN16/32 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ1	○							○	--
Ⓢ2	○							○	--
Ⓓ	○							--	--

功 能

WXOR

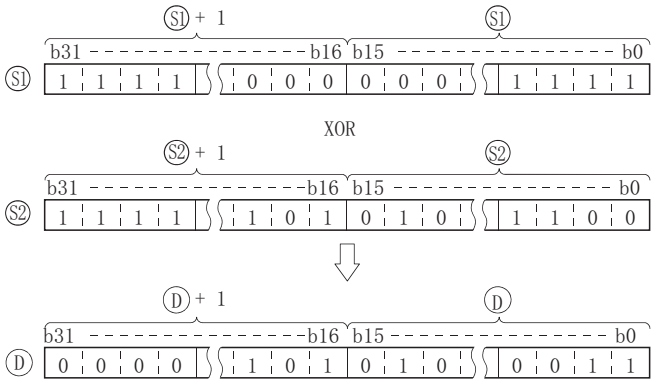
(1) 将Ⓢ1中指定的软元件的 16 位数据与Ⓢ2中指定的软元件的 16 位数据逐位进行排他逻辑和运算，并将结果存储到Ⓓ中指定的软元件中。



(2) 位软元件的情况下，位数指定点数以后的位软元件将被作为 0 进行运算。(参阅程序示例 (1))

DXOR

(1) 将Ⓢ1中指定的软元件的 32 位数据与Ⓢ2中指定的软元件的 32 位数据逐位进行排他逻辑和运算，并将结果存储到Ⓓ中指定的软元件中。



(2) 位软元件的情况下，位数指定点数以后的位软元件将被作为 0 进行运算。

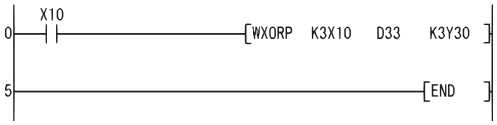
出 错

(1) 在 WXOR(P)、DXOR(P) 指令中无运算出错。

程序示例

(1) 以下为 X10 变为 ON 时，将 X10 ～ X1B 的数据与 D33 的数据进行排他逻辑和运算，并将结果存储到 Y30 ～ Y3B 中的程序。

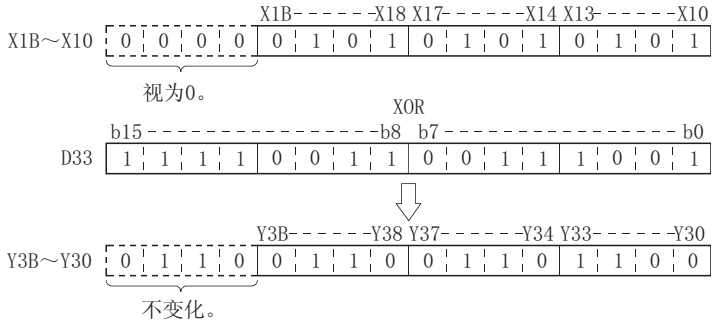
[梯形图模式]



[列表模式]

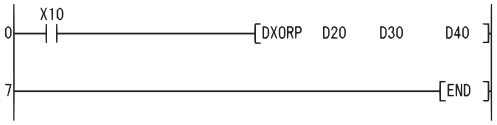
步	指令	软元件		
0	LD	X10		
1	WXORP	K3X10	D33	K3Y30
5	END			

[动作]



(2) 以下为 X10 变为 ON 时，将 D20、D21 的数据与 D30、D31 的数据进行排他逻辑和运算，并将结果存储到 D40、D41 中的程序。

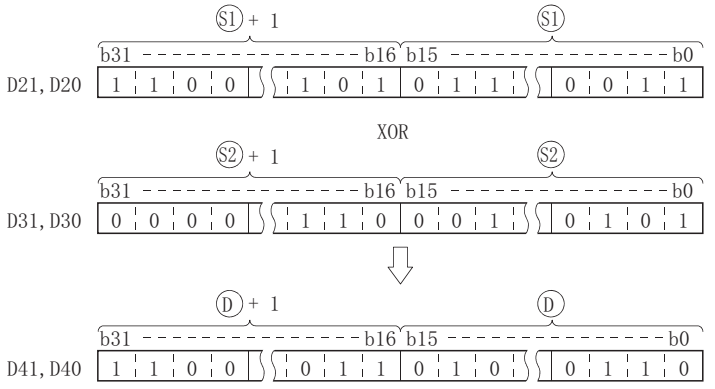
[梯形图模式]



[列表模式]

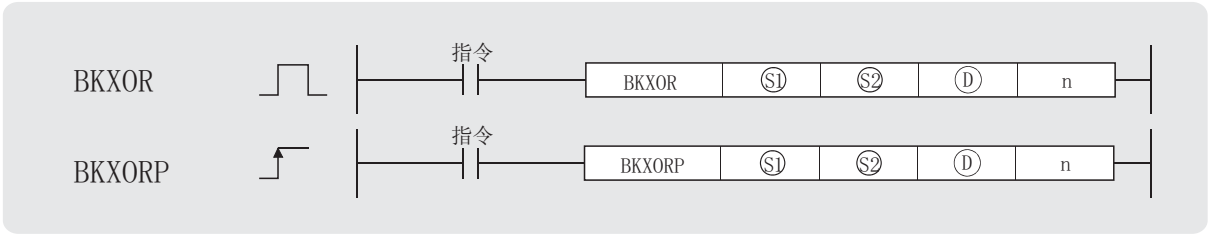
步	指令	软元件		
0	LD	X10		
1	DXORP	D20	D30	D40
7	END			

[动作]



7.1.6 块排他逻辑和 (BKXOR(P))

Basic | High performance | Process | Redundant | Universal



(S1)*1 ：存储执行逻辑运算数据的软元件的起始编号 (BIN16 位)。

(S2)*1 ：执行逻辑运算的数据或者存储执行逻辑运算数据的软元件的起始编号 (BIN16 位)。

(D)*1 ：存储运算结果的软元件的起始编号 (BIN16 位)。

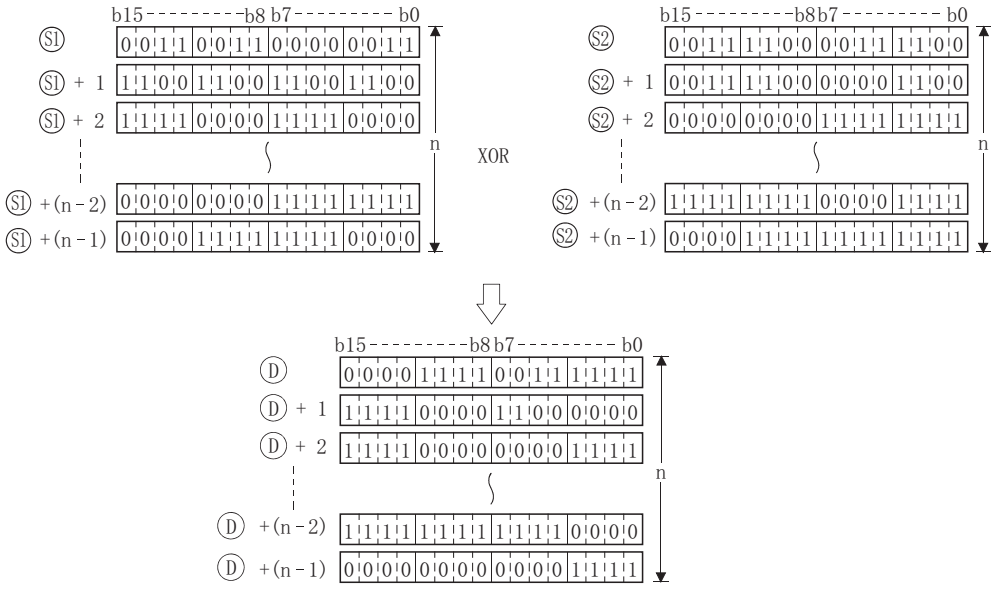
n ：运算数据个数 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
(S1)*1	--	○		--				--	--
(S2)*1	--	○		--				○	--
(D)*1	--	○		--				--	--
n	○	○		○				○	--

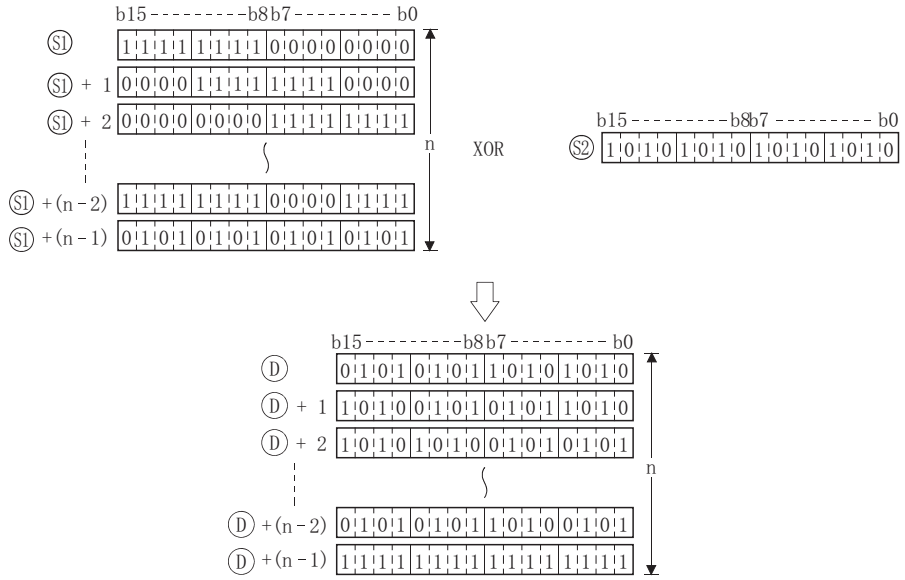
＊1 :(S1) 与(D) 或者(S2) 与(D) 可以指定为相同的软元件编号。

功 能

(1) 将从(S1) 中指定的软元件开始的 n 点的内容与从(S2) 中指定的软元件开始的 n 点的内容执行排他逻辑和运算，并将结果存储到(D) 中指定的软元件后面。



(2) ⓢ2 中可以指定为 -32768 ～ 32767(BIN16 位) 的常数。



出　错

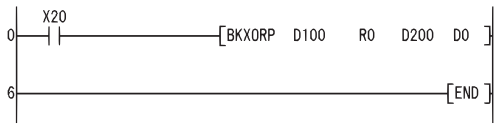
(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- 从ⓢ1、ⓢ2、ⓓ的软元件开始的 n 点的范围超出了相应软元件范围时。
  (出错代码 :4101)
- 从ⓢ1开始至第 n 点为止的软元件范围与从ⓓ开始的至第 n 点为止的软元件范围有部分重复时。
  (ⓢ1与ⓓ中指定了同一个软元件时除外。) (出错代码 :4101)
- 从ⓢ2开始至第 n 点为止的软元件范围与从ⓓ开始的至第 n 点为止的软元件范围相重复时。
  (ⓢ2与ⓓ中指定了同一个软元件时除外。) (出错代码 :4101)

程序示例

(1) 以下为 X20 变为 ON 时，将 D100 ～ D102 中存储的数据值与 R0 ～ R2 中存储的数据值进行排他逻辑和运算，并将其结果存储到 D200 后面的程序。

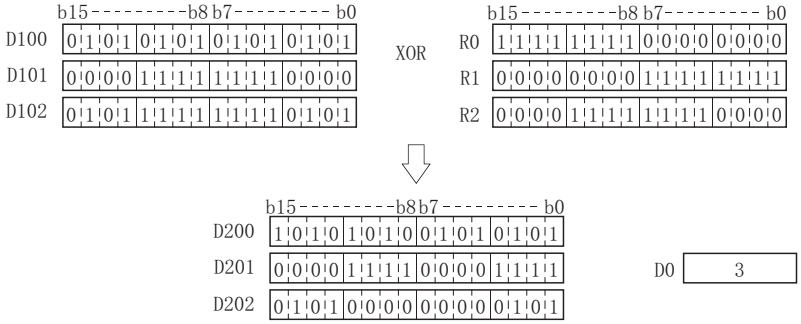
[梯形图模式]



[列表模式]

步	指令	软元件			
0	LD	X20			
1	BKXORP	D100	R0	D200	D0
6	END				

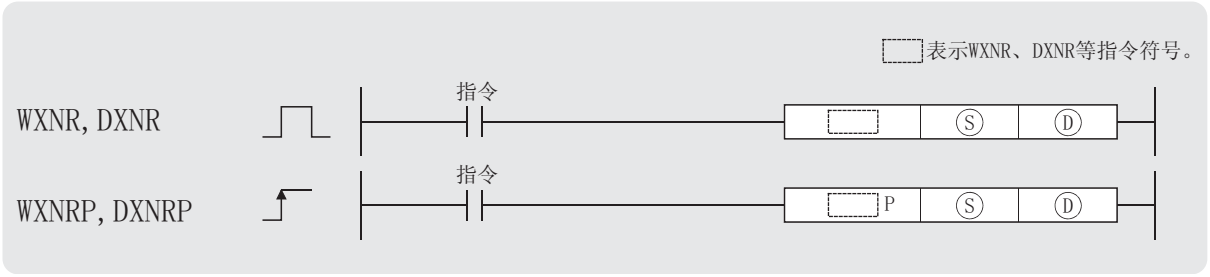
[动作]



7.1.7 16 位 /32 位数据否定排他逻辑和 (WXNR(P)、DXNR(P))

Basic | High performance | Process | Redundant | Universal

1 设置数据为 2 个时 ($\overline{(D) \veebar (S) \rightarrow (D)}$，$\overline{(D)+1, (D)) \veebar (S)+1, (S))} \rightarrow ((D)+1, (D))$)



(S)：执行否定排他逻辑和的数据或者存储数据的软元件的起始编号 (BIN16/32 位)。

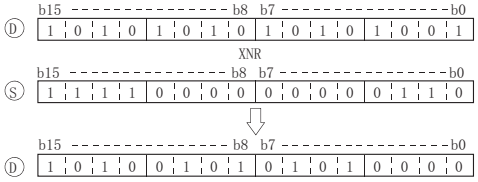
(D)：存储否定排他逻辑和结果的软元件的起始编号 (BIN16/32 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
(S)	○							○	--
(D)	○							--	--

功 能

WXNR

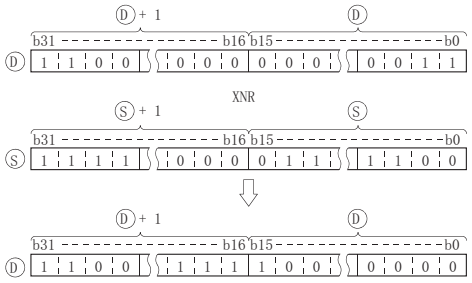
(1) 将(D)中指定的软元件的 16 位数据与(S)中指定的软元件的 16 位数据逐位进行否定排他逻辑和运算，并将结果存储到(D)中指定的软元件中。



(2) 位软元件的情况下，位数指定点数以后的位软元件将被作为 0 进行运算。

DXNR

(1) 将(D)中指定的软元件的 32 位数据与(S)中指定的软元件的 32 位数据逐位进行否定排他逻辑和运算，并将结果存储到(D)中指定的软元件中。



(2) 位软元件的情况下，位数指定点数以后的位软元件将被作为 0 进行运算。

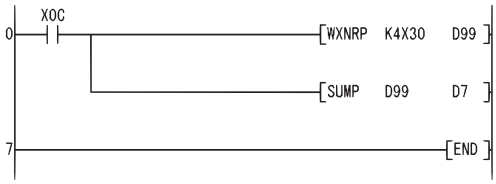
出　错

(1) 在 WXNR(P)、DXNR(P) 指令中无运算出错。

程序示例

(1) 以下为 XC 变为 ON 时，将 X30 ～ X3F 的 16 位数据与 D99 的 16 位数据执行比较，并将相同的位模式数存储到 D7 中的程序。

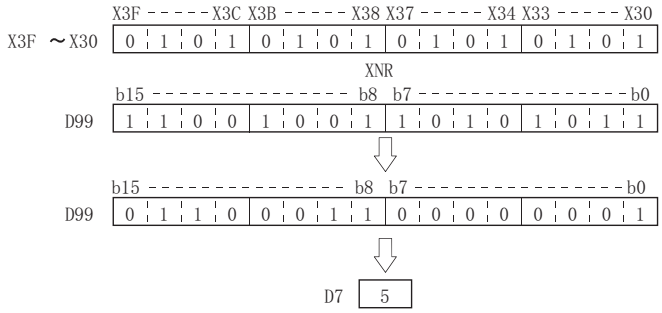
[梯形图模式]



[列表模式]

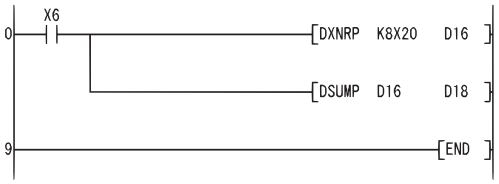
步	指令	软元件	
0	LD	X0C	
1	WXNRP	K4X30	D99
4	SUMP	D99	D7
7	END		

[动作]



(2) 以下为 X6 变为 ON 时，将 X20 ～ X3F 的 32 位数据与 D16、D17 的数据的位模式进行比较，并将相同的位模式数存储到 D18 中的程序。

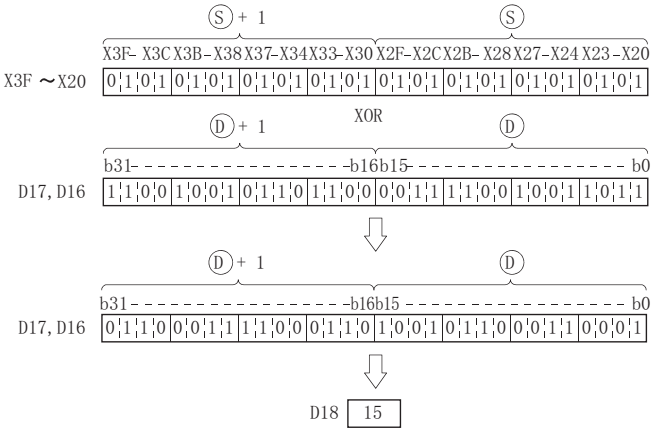
[梯形图模式]



[列表模式]

步	指令	软元件	
0	LD	X6	
1	DXNRP	K8X20	D16
6	DSUMP	D16	D18
9	END		

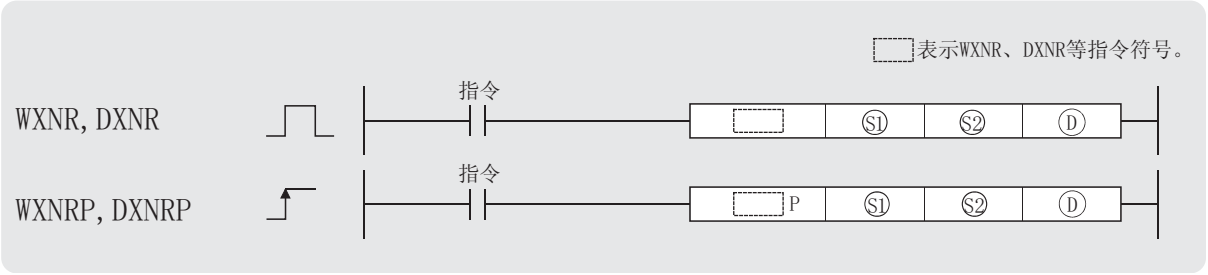
[动作]



备 注

关于 SUMP/DSUMP 指令的有关内容，请参阅 7.5.2 项。

2 设置数据为 3 个时 ($\overline{(S1) \veebar (S2)} \rightarrow (D)$，$\overline{(S1)+1, (S1)) \veebar ((S2)+1, (S2))} \rightarrow ((D)+1, (D))$)



(S1)，(S2)：执行否定排他逻辑和的数据或者存储数据的软元件的起始编号 (BIN16/32 位)。

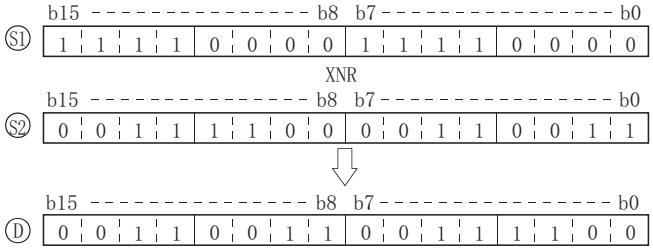
(D)：存储否定排他逻辑和结果的软元件的起始编号 (BIN16/32 位)。

设置数据	内部软元件 位	内部软元件 字	R、ZR	J□\□ 位	J□\□ 字	U□\G□	Zn	常数 K、H	其它
(S1)	○							○	--
(S2)	○							○	--
(D)	○							--	--

功 能

WXNR

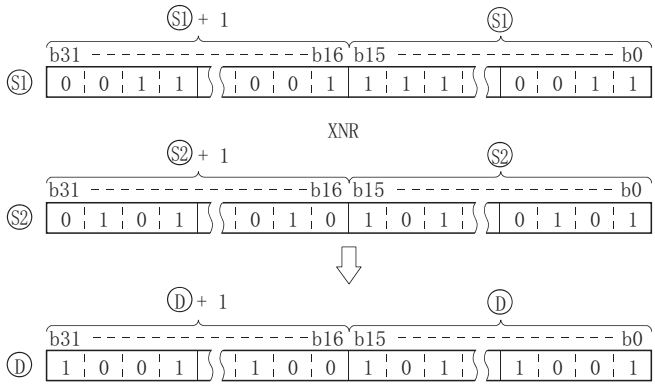
(1) 将(S1)中指定的软元件的 16 位数据与(S2)中指定的软元件的 16 位数据进行否定排他逻辑和运算，并将结果存储到(D)中指定的软元件中。



(2) 位软元件的情况下，位数指定点数以后的位软元件将被作为 0 进行运算。

DXNR

(1) 将(S1)中指定的软元件的 32 位数据与(S2)中指定的软元件的 32 位数据进行否定排他逻辑和运算，并将结果存储到(D)中指定的软元件中。



(2) 位软元件的情况下，位数指定点数以后的位软元件将被作为 0 进行运算。

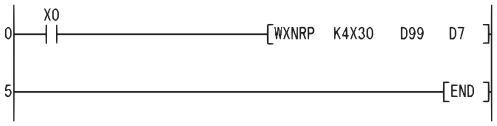
出　错

(1) 在 WXNR(P)、DXNR(P) 指令中无运算出错。

程序示例

(1) 以下为 X0 变为 ON 时，将 X30 ～ X3F 的 16 位数据与 D99 的数据进行否定排他逻辑和运算，并将结果存储到 D7 中的程序。

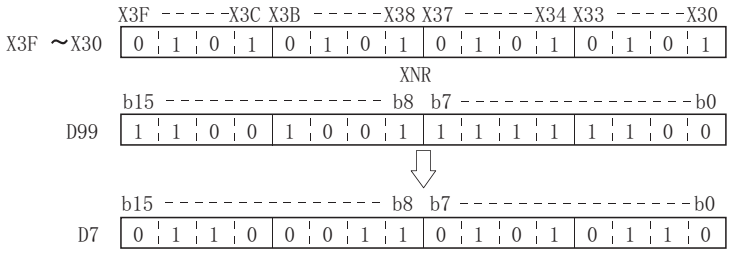
[梯形图模式]



[列表模式]

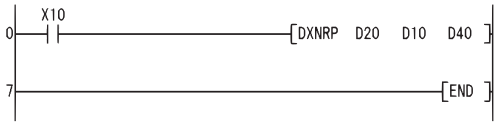
步	指令	软元件		
0	LD	X0		
1	WXNRP	K4X30	D99	D7
5	END			

[动作]



(2) 以下为 X10 变为 ON 时，将 D20、D21 的 32 位数据与 D10、D11 的数据进行否定排他逻辑和运算，并将结果存储到 D40、D41 中的程序。

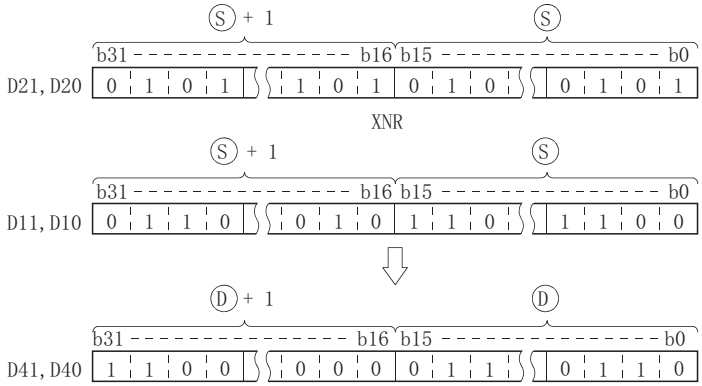
[梯形图模式]



[列表模式]

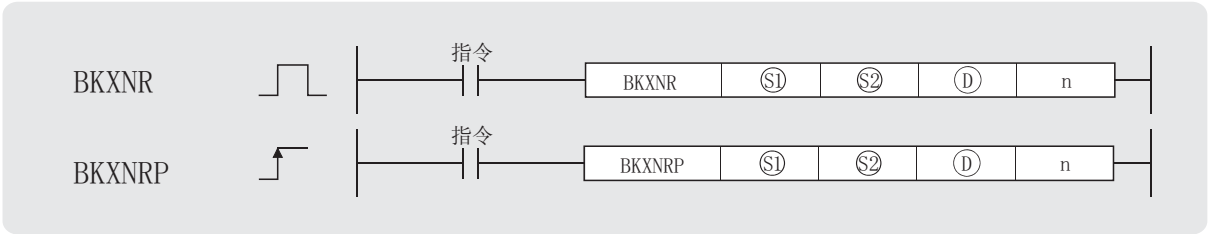
步	指令	软元件		
0	LD	X10		
1	DXNRP	D20	D10	D40
7	END			

[动作]



7.1.8 块否定排他逻辑和 (BKXNR(P))

Basic | High performance | Process | Redundant | Universal



(S1)*1 ：存储执行逻辑运算数据的软元件的起始编号 (BIN16 位)。

(S2)*1 ：执行逻辑运算的数据或者存储执行逻辑运算数据的软元件的起始编号 (BIN16 位)。

(D)*1 ：存储运算结果的软元件的起始编号 (BIN16 位)。

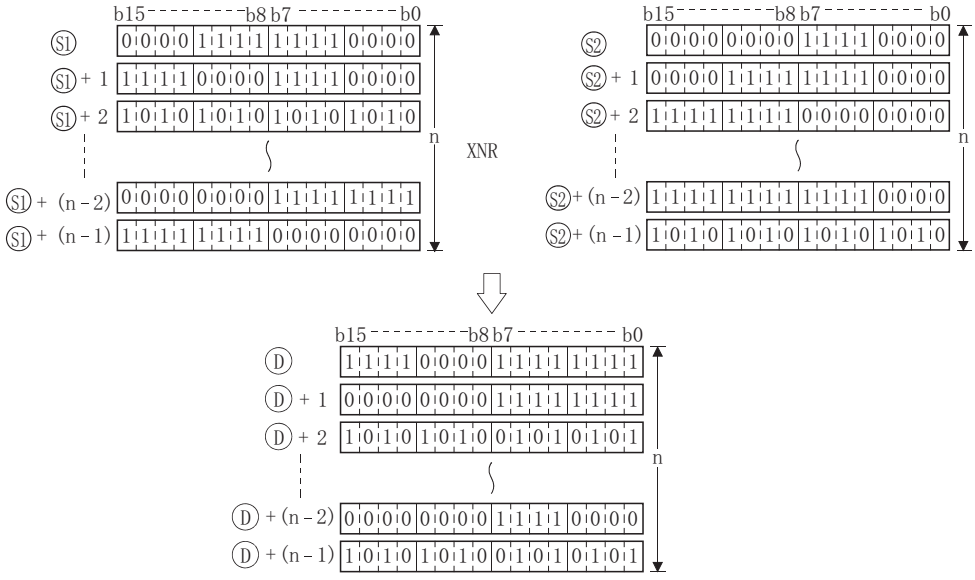
n ：运算数据个数 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
(S1)*1	--	○		__				--	--
(S2)*1	--	○		__				○	--
(D)*1	--	○		__				--	--
n	○	○		○				○	--

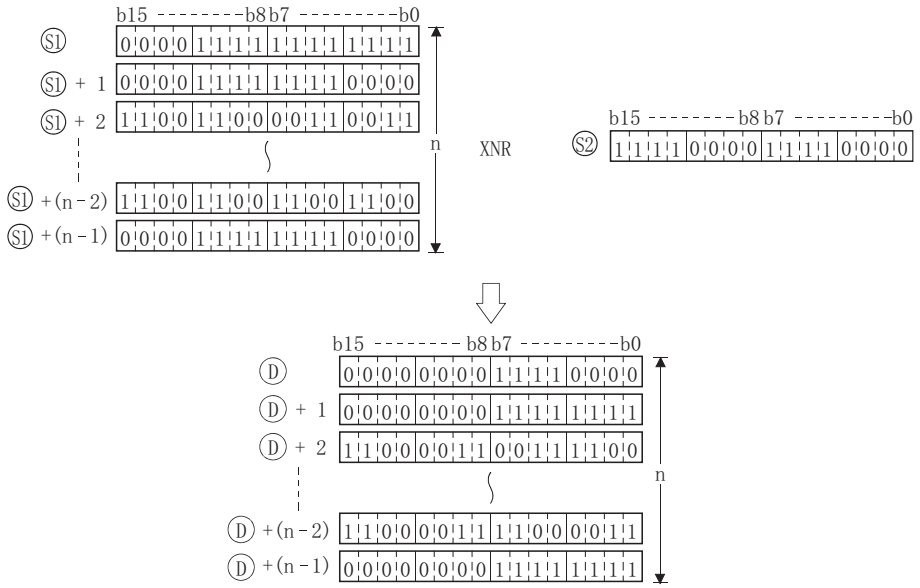
*1 :(S1)与(D)或者(S2)与(D)可以指定为相同的软元件编号。

功 能

(1) 将从(S1)中指定的软元件开始的 n 点的内容与从(S2)中指定的软元件开始的 n 点的内容执行否定排他逻辑和运算，并将结果存储到(D)中指定的软元件后面。



(2) ⓢ2中可以指定为 -32768 ～ 32767(BIN16 位) 的常数。



出 错

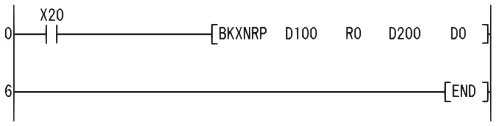
(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- 从ⓢ1、ⓢ2、ⓓ的软元件开始的 n 点的范围超出了相应软元件范围时。
  (出错代码 :4101)
- 从ⓢ1开始至第 n 点为止的软元件范围与从ⓓ开始的至第 n 点为止的软元件范围有部分重复时。
  (ⓢ1与ⓓ中指定了同一个软元件时除外。) (出错代码 :4101)
- 从ⓢ2开始至第 n 点为止的软元件范围与从ⓓ开始的至第 n 点为止的软元件范围相重复时。
  (ⓢ2与ⓓ中指定了同一个软元件时除外。) (出错代码 :4101)

程序示例

(1) 以下为 X20 变为 ON 时，将 D100 ～ D102 中存储的数据值与 R0 ～ R2 中存储的数据值进行否定排他逻辑和运算，并将其结果存储到 D200 后面的程序。

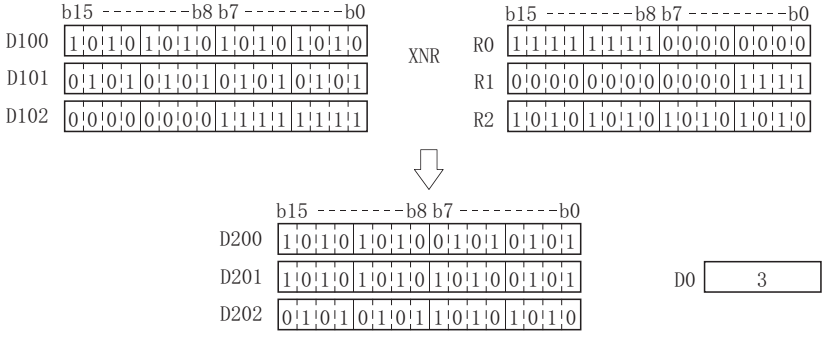
[梯形图模式]



[列表模式]

步	指令	软元件			
0	LD	X20			
1	BKXNRP	D100	R0	D200	D0
6	END				

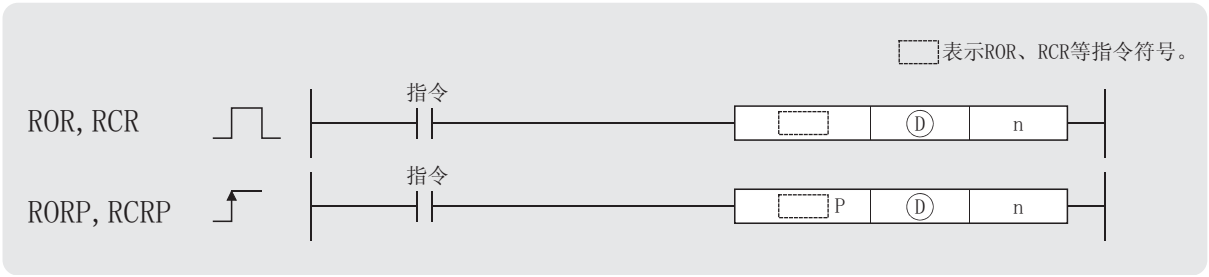
[动作]



7.2 旋转指令

7.2.1 16 位数据的右旋转 (ROR(P)、RCR(P))

Basic | High performance | Process | Redundant | Universal



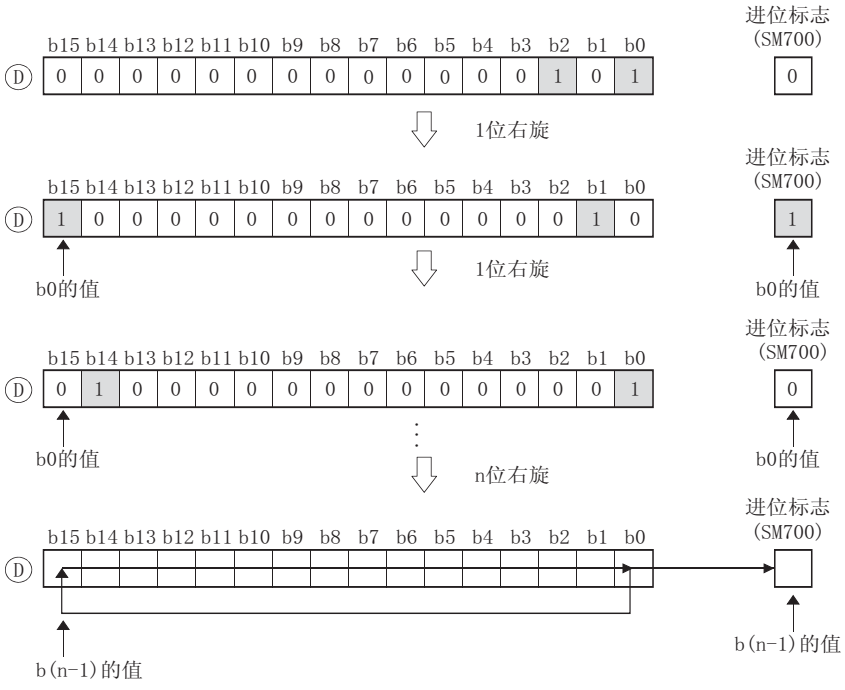
Ⓓ ：进行旋转的软元件的起始编号 (BIN16 位)。
n ：旋转次数 (0～15)(BIN16 位)。

设置数据	内部软元件 位	内部软元件 字	R、ZR	J□\□ 位	J□\□ 字	U□\G□	Zn	常数 K、H	其它
Ⓓ	○							--	--
n	○							○	--

功 能

ROR

(1) 将Ⓓ中指定的软元件的 16 位数据在不包含进位标志的状态下进行 n 位右旋转。
进位标志根据 ROR 执行前的状态而 ON/OFF。

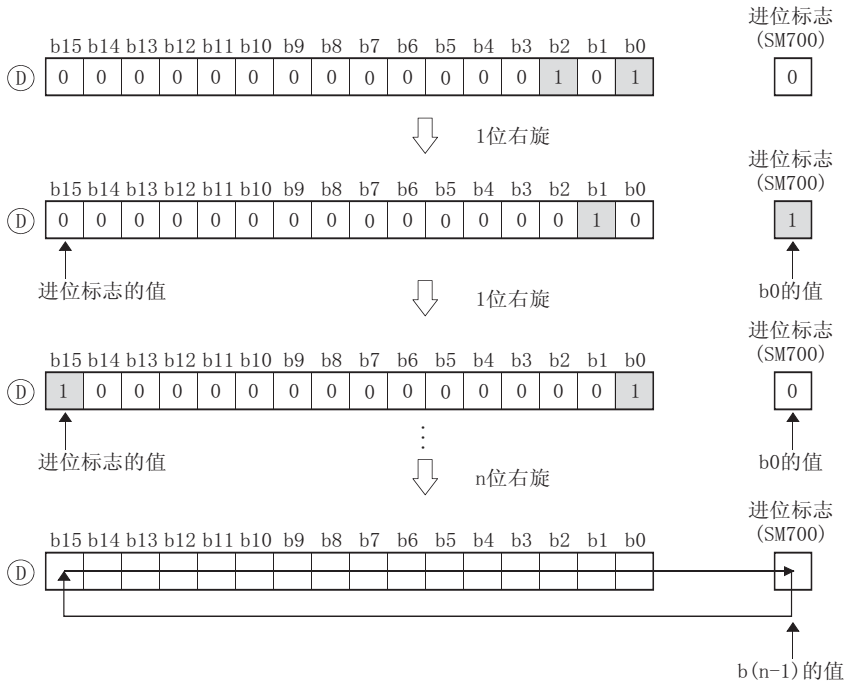


(2) Ⓓ中指定了软元件的情况下，按位数指定中指定的软元件范围进行旋转。
此时，实际旋转的位数为 n ÷（位数指定中指定的点数）的余数。
例如，n=15，（位数指定中指定的点数）=12 位时，15 ÷ 12=1 余数为 3，因此旋转 3 位。

(3) n 的指定范围为 0 ～ 15。
n 中指定了 16 以上的值时，将按 n ÷ 16 的余数值进行旋转。
例如，n=18 时，18 ÷ 16=1 余数为 2，因此进行 2 位右旋。

RCR

(1) 将Ⓓ中指定的软元件的 16 位数据在包含进位标志的状态下进行 n 位右旋转。
进位标志根据 RCR 执行前的状态而 ON/OFF。



(2) 在Ⓓ中指定了软元件的情况下，按位数指定中指定的软元件范围进行旋转。
此时，实际旋转的位数为 n ÷（位数指定中指定的点数）的余数。
例如，n=15，（位数指定中指定的点数）=12 位时，15 ÷ 12=1 余数为 3，因此旋转 3 位。

(3) n 的指定范围为 0 ～ 15。
n 中指定了 16 以上的值时，将按 n ÷ 16 的余数值进行旋转。
例如，n=18 时，18 ÷ 16=1 余数为 2，因此进行 2 位右旋。

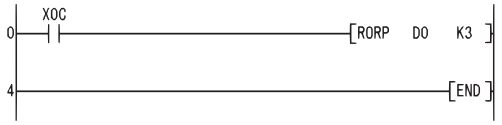
出　错

(1) 在 ROR(P)、RCR(P) 指令中无运算出错。

程序示例

(1) 以下为 XC 变为 ON 时，将 D0 的内容在不包含进位标志的状态下进行 3 位右旋转的程序。

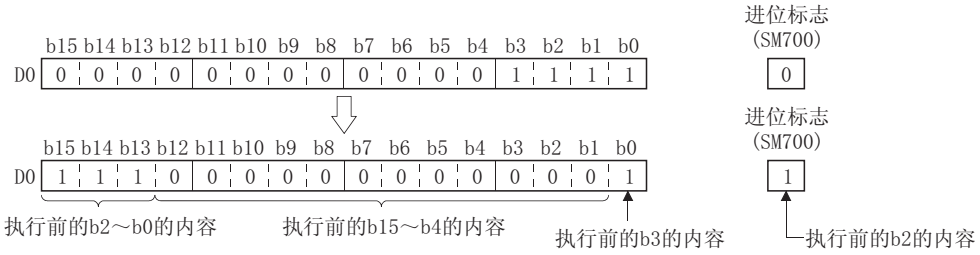
[梯形图模式]



[列表模式]

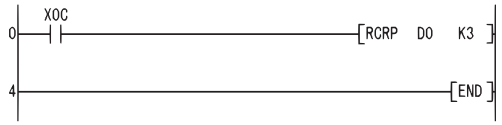
步	指令	软元件	
0	LD	XOC	
1	RORP	D0	K3
4	END		

[动作]



(2) 以下为 XC 变为 ON 时，将 D0 的内容在包含进位标志的状态下进行 3 位右旋转的程序。

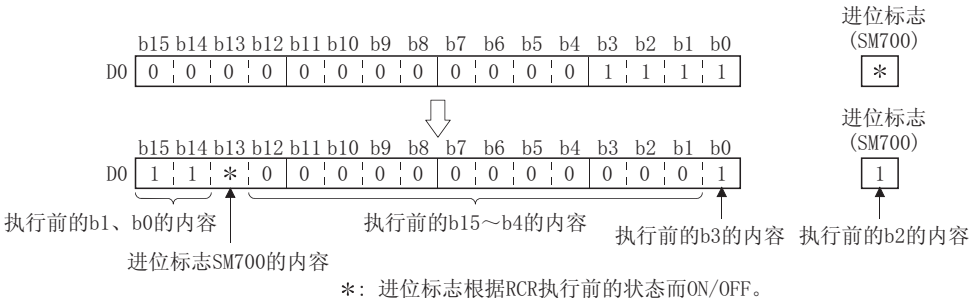
[梯形图模式]



[列表模式]

步	指令	软元件	
0	LD	XOC	
1	RCRP	D0	K3
4	END		

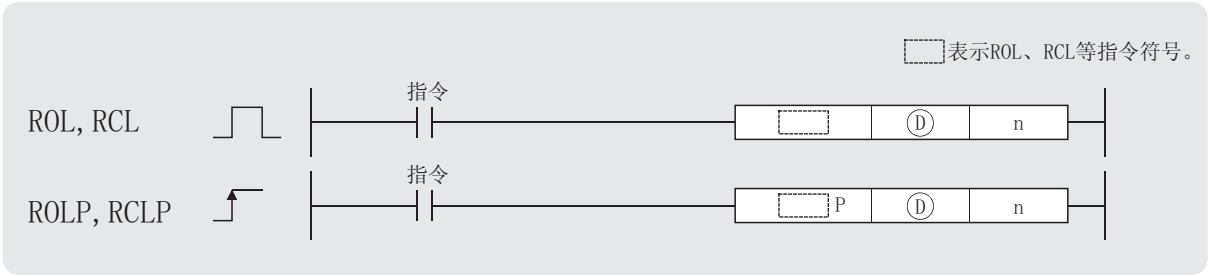
[动作]



*：进位标志根据RCR执行前的状态而ON/OFF。

7.2.2 16 位数据左旋转 (ROL(P)、RCL(P))

Basic | High performance | Process | Redundant | Universal



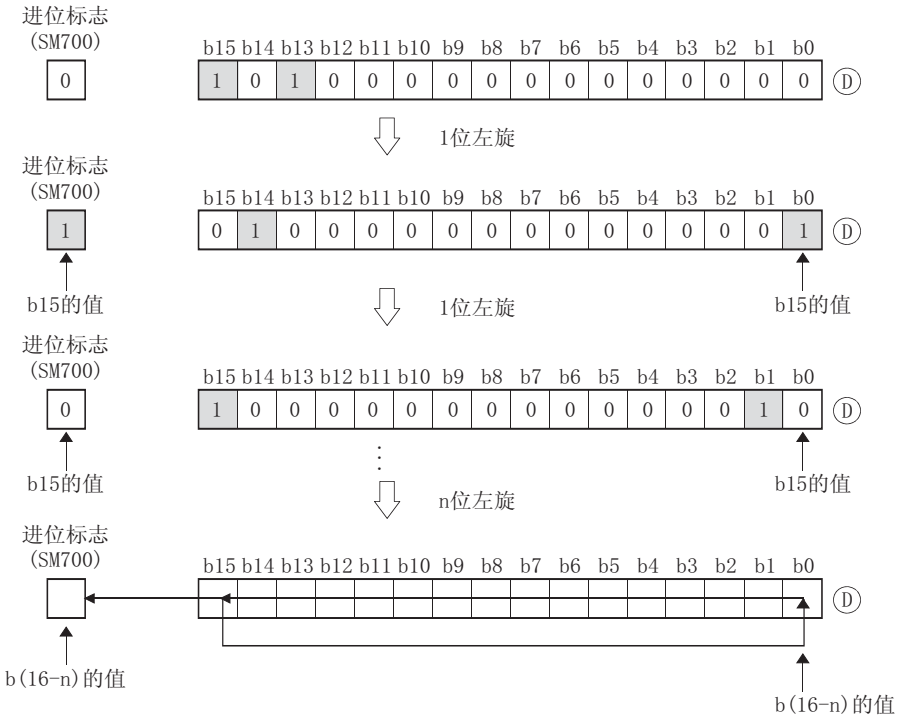
Ⓓ : 进行旋转的软元件的起始编号 (BIN16 位)。
n : 旋转次数 (0 ～ 15)(BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓓ				○				--	--
n				○				○	--

功 能

ROL

(1) 将Ⓓ中指定的软元件的 16 位数据在不包含进位标志的状态下进行 n 位左旋转。
进位标志根据 ROL 执行前的状态而 ON/OFF。

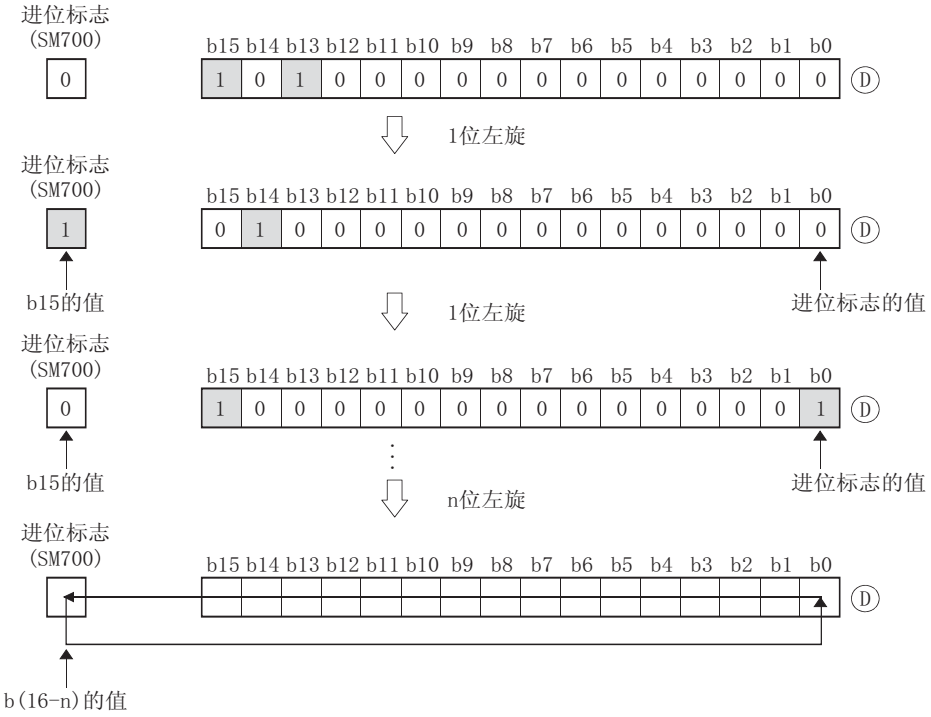


(2) 在ⓓ中指定了软元件的情况下，按位数指定中指定的软元件范围进行旋转。
此时，实际旋转的位数为 n ÷（位数指定中指定的点数）的余数。
例如，n=15，（位数指定中指定的点数）=12 位时，15 ÷ 12=1 余数为 3，因此旋转 3 位。

(3) n 的指定范围为 0 ～ 15。
n 中指定了 16 以上的值时，将按 n ÷ 16 的余数值进行旋转。
例如，n=18 时，18 ÷ 16=1 余数为 2，因此进行 2 位左旋。

RCL

(1) 将ⓓ中指定的软元件的 16 位数据在包含进位标志的状态下进行 n 位左旋转。
进位标志根据 RCL 执行前的状态而 ON/OFF。



(2) 在ⓓ中指定了软元件的情况下，按位数指定中指定的软元件范围进行旋转。
此时，实际旋转的位数为 n ÷（位数指定中指定的点数）的余数。
例如，n=15，（位数指定中指定的点数）=12 位时，15 ÷ 12=1 余数为 3，因此旋转 3 位。

(3) n 的指定范围为 0 ～ 15。
n 中指定了 16 以上的值时，将按 n ÷ 16 的余数值进行旋转。
例如，n=18 时，18 ÷ 16=1 余数为 2，因此进行 2 位左旋。

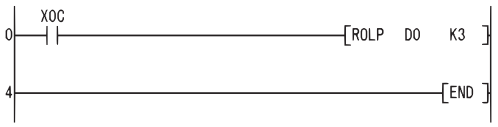
出 错

(1) 在 ROL(P)、RCL(P) 指令中无运算出错。

程序示例

(1) 以下为 XC 变为 ON 时，将 D0 的内容在不包含进位标志的状态下进行 3 位左旋转的程序。

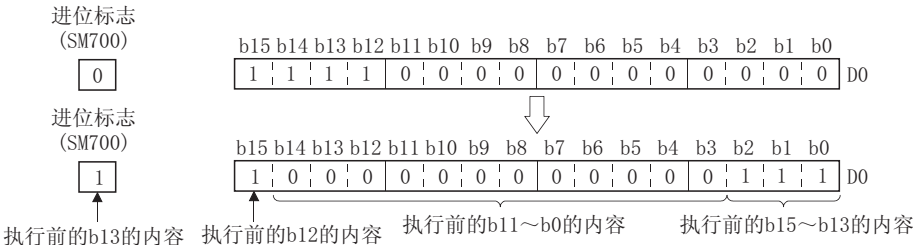
[梯形图模式]



[列表模式]

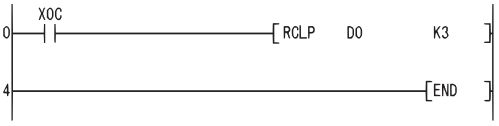
步	指令	软元件	
0	LD	XOC	
1	ROLP	D0	K3
4	END		

[动作]



(2) 以下为 XC 变为 ON 时，将 D0 的内容在包含进位标志的状态下进行 3 位左旋转的程序。

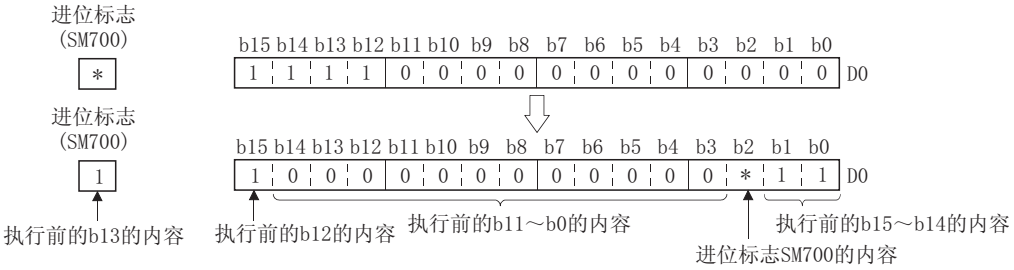
[梯形图模式]



[列表模式]

步	指令	软元件	
0	LD	XOC	
1	RCLP	D0	K3
4	END		

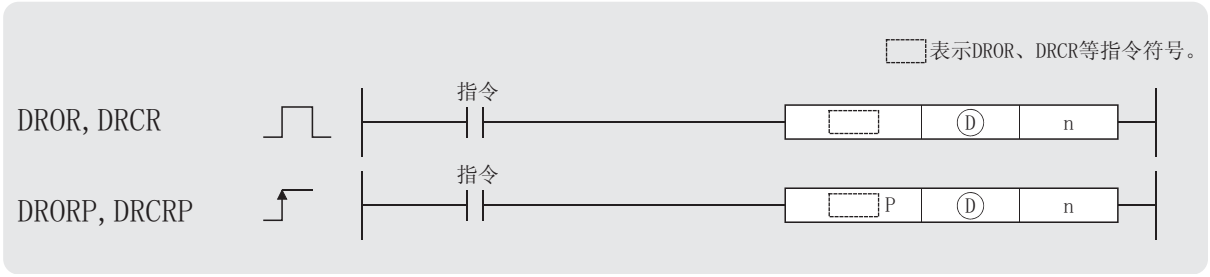
[动作]



*: 进位标志根据RCL执行前的状态而ON/OFF。

7.2.3 32 位数据的右旋转 (DROR(P)、DRCR(P))

Basic | High performance | Process | Redundant | Universal



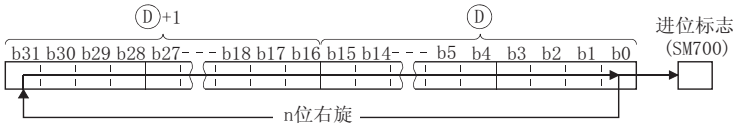
Ⓓ ：进行旋转的软元件的起始编号 (BIN32 位)。
n ：旋转次数 (0 ～ 31)(BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓓ		○						--	--
n		○						○	--

功 能

DROR

(1) 将Ⓓ中指定的软元件的 32 位数据在不包含进位标志的状态下进行 n 位右旋转。
进位标志根据 DROR 执行前的状态而 ON/OFF。

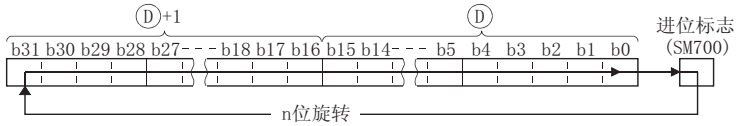


(2) 在Ⓓ中指定了软元件的情况下，按位数指定中指定的软元件范围进行旋转。
此时，实际旋转的位数为 n ÷ (位数指定中指定的点数) 的余数。
例如，n=31，(位数指定中指定的点数)=24 位时，31 ÷ 24=1 余数为 7，因此旋转 7 位。

(3) n 的指定范围为 0 ～ 31。
n 中指定了 32 以上时，按 n ÷ 32 的余数值进行旋转。
例如，n=34 时，34 ÷ 32=1 余数为 2，因此进行 2 位右旋。

DRCR

(1) 将Ⓓ中指定的软元件的 32 位数据在包含进位标志的状态下进行 n 位右旋转。
进位标志根据 DRCR 执行前的状态而 ON/OFF。



(2) 在Ⓓ中指定了软元件的情况下，按位数指定中指定的软元件范围进行旋转。
此时，实际旋转的位数为 n ÷（位数指定中指定的点数）的余数。
例如，n=31，（位数指定中指定的点数）=24 位时，31 ÷ 24=1 余数为 7，因此旋转 7 位。

(3) n 的指定范围为 0 ～ 31。
n 中指定了 32 以上的值时，将按 n ÷ 32 的余数值进行旋转。
例如，n=34 时，34 ÷ 32=1 余数为 2，因此进行 2 位右旋。

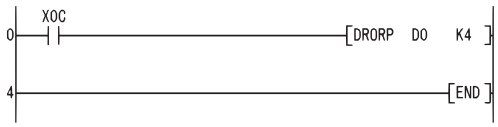
出　错

(1) 在 DROR(P)、DRCR(P) 指令中无运算出错。

程序示例

(1) 以下为 XC 变为 ON 时，将 D0、D1 的内容在不包含进位标志的状态下进行 4 位右旋转的程序。

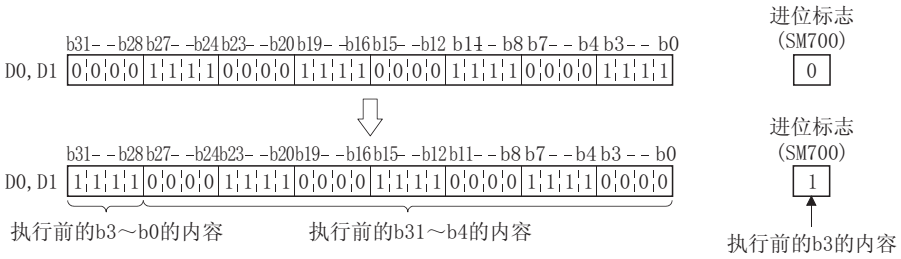
[梯形图模式]



[列表模式]

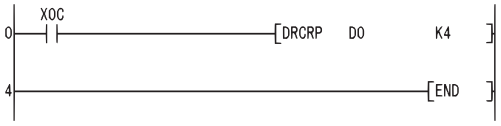
步	指令	软元件	
0	LD	XOC	
1	DRORP	D0	K4
4	END		

[动作]



(2) 以下为 XC 变为 ON 时，将 D0、D1 的内容在包含进位标志的状态下进行 4 位右旋转的程序。

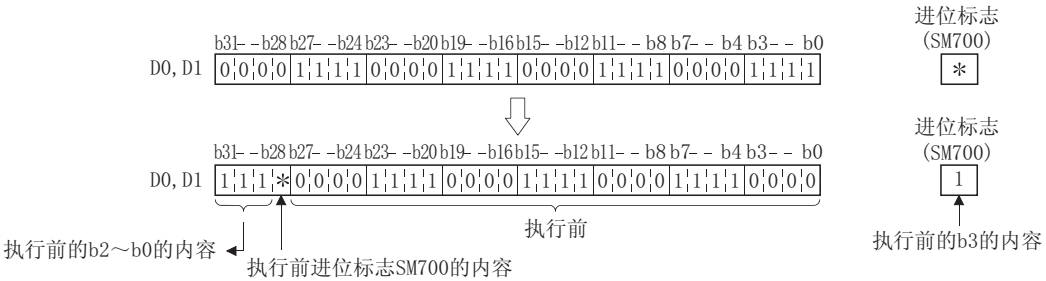
[梯形图模式]



[列表模式]

步	指令	软元件	
0	LD	XOC	
1	DRCRP	D0	K4
4	END		

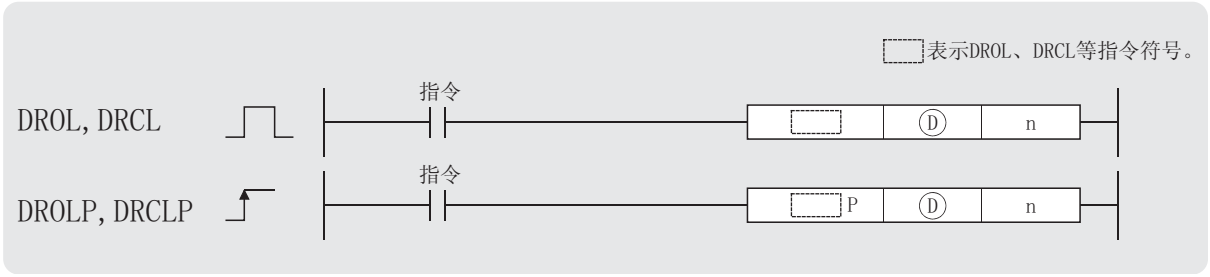
[动作]



*：进位标志根据DRCR执行前的状态而ON/OFF。

7.2.4 32 位数据左旋转 (DROL(P)、DRCL(P))

Basic | High performance | Process | Redundant | Universal



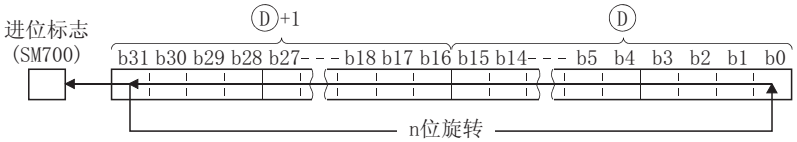
Ⓓ ：进行旋转的软元件的起始编号 (BIN32 位)。
n ：旋转次数 (0 ～ 31)(BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓓ		○						--	--
n		○						○	--

功 能

DROL

(1) 将Ⓓ中指定的软元件的 32 位数据在不包含进位标志的状态下进行 n 位左旋转。
进位标志根据 DROL 执行前的状态而 ON/OFF。

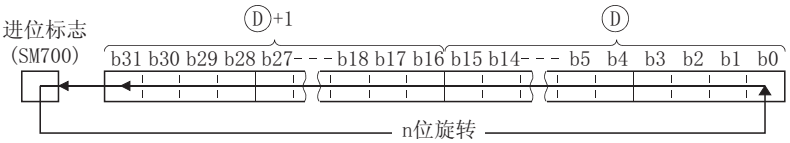


(2) 在Ⓓ中指定了软元件的情况下，按位数指定中指定的软元件范围进行旋转。
此时，实际旋转的位数为 n ÷ (位数指定中指定的点数) 的余数。
例如，n=31，(位数指定中指定的点数)=24 位时，31 ÷ 24=1 余数为 7，因此旋转 7 位。

(3) n 的指定范围为 0 ～ 31。
n 中指定了 32 以上时，按 n ÷ 32 的余数值进行旋转。
例如，n=34 时，34 ÷ 32=1 余数为 2，因此进行 2 位左旋。

DRCL

(1) 将Ⓓ中指定的软元件的 32 位数据在包含进位标志的状态下进行 n 位左旋转。
进位标志根据 DRCL 执行前的状态而 ON/OFF。



(2) 在ⓓ中指定了软元件的情况下，按位数指定中指定的软元件范围进行旋转。
此时，实际旋转的位数为 n ÷（位数指定中指定的点数）的余数。
例如，n=31，（位数指定中指定的点数）=24 位时，31 ÷ 24=1 余数为 7，因此旋转 7 位。

(3) n 的指定范围为 0 ～ 31。
n 中指定了 32 以上的值时，将按 n ÷ 32 的余数值进行旋转。
例如，n=34 时，34 ÷ 32=1 余数为 2，因此进行 2 位左旋。

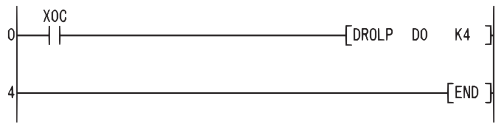
出　错

(1) 在 DROL(P)、DRCL(P) 指令中无运算出错。

程序示例

(1) 以下为 XC 变为 ON 时，将 D0、D1 的内容在不包含进位标志的状态下进行 4 位左旋转的程序。

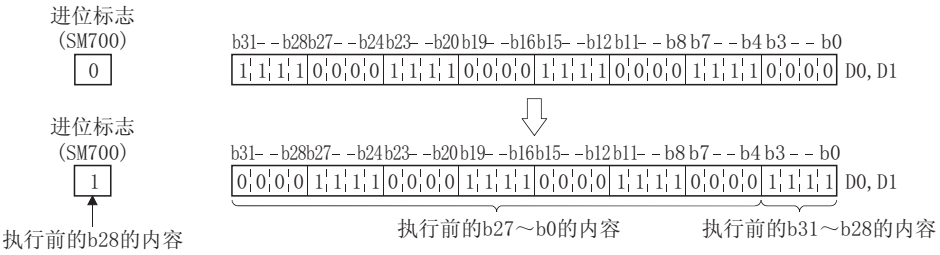
[梯形图模式]



[列表模式]

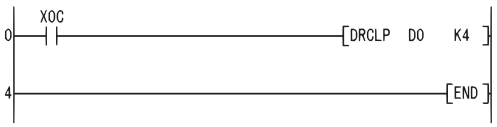
步	指令	软元件	
0	LD	XOC	
1	DROLP	D0	K4
4	END		

[动作]



(2) 以下为 XC 变为 ON 时，将 D0、D1 的内容在包含进位标志的状态下进行 4 位左旋转的程序。

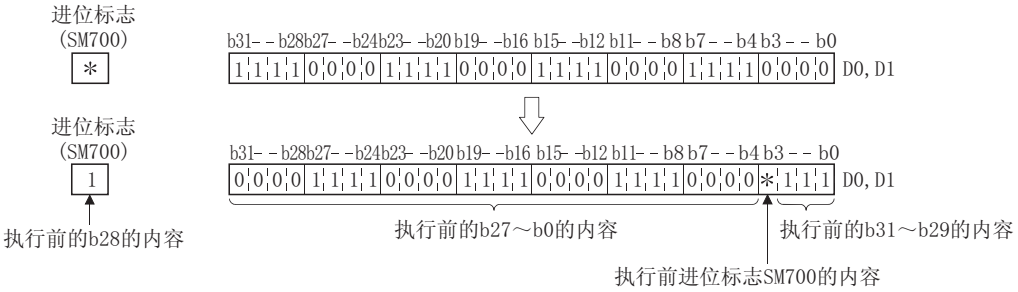
[梯形图模式]



[列表模式]

步	指令	软元件	
0	LD	XOC	
1	DRCLP	D0	K4
4	END		

[动作]

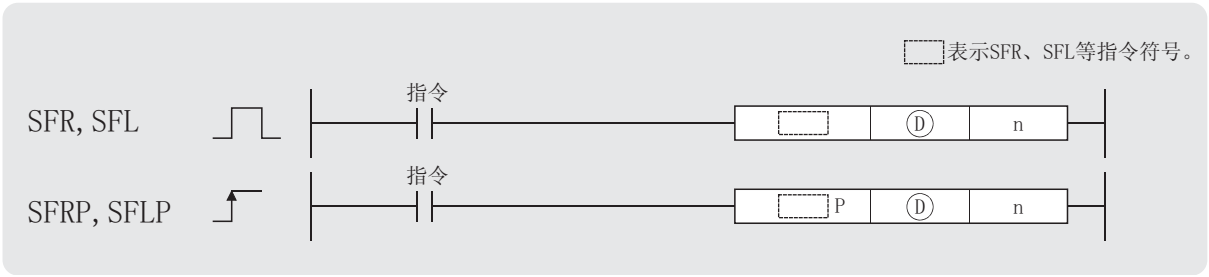


*：进位标志根据DRCL执行前的状态而ON/OFF。

7.3 移位指令

7.3.1 16 位数据的 n 位右移或左移 (SFR(P)、SFL(P))

Basic | High performance | Process | Redundant | Universal



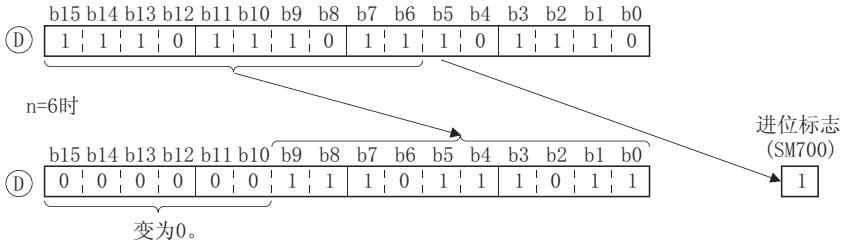
Ⓓ ：存储移位数据的软元件的起始编号 (BIN16 位)。
n ：移位次数 (0 ～ 15)(BIN16 位)。

设置数据	内部软元件 位	内部软元件 字	R、ZR	J□\□ 位	J□\□ 字	U□\G□	Zn	常数 K、H	其它
Ⓓ	○							--	--
n	○							○	--

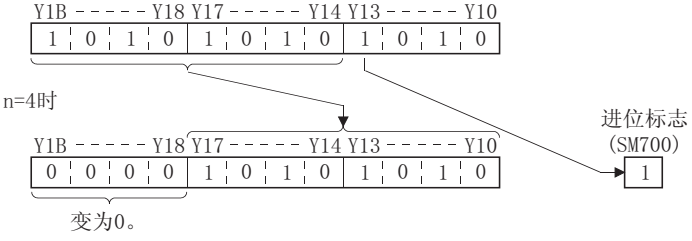
功 能

SFR

(1) 将Ⓓ中指定的软元件的 16 位数据右移 n 位。
从最高位开始至 n 位为止将变为 0。



(2) Ⓓ中指定了位软元件时，按位数指定中指定的软元件范围向右移位。

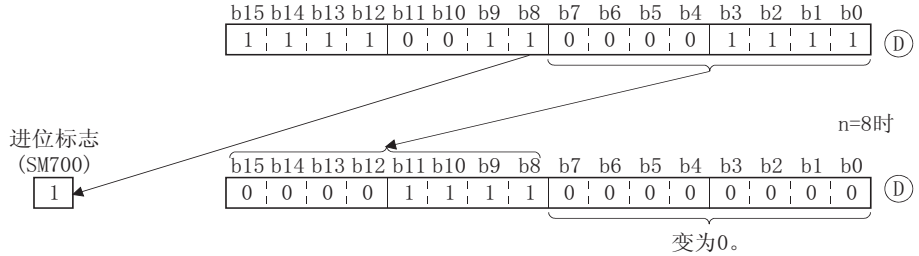


此时实际的移位位数为 n ÷（位数指定中指定的点数）的余数。
例如，n=15，（位数指定中指定的点数）=8 位时，15 ÷ 8=1 余数为 7，因此进行 7 位的移位。

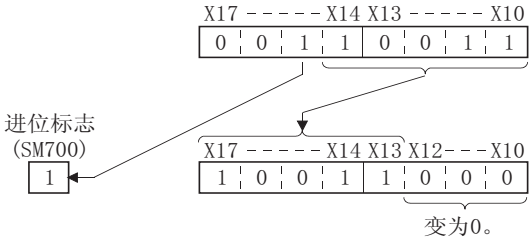
(3) n 的指定范围为 0 ～ 15。
在 n 中指定了 16 以上的值时，按 n ÷ 16 的余数值进行右移。
例如，n=18 时，18 ÷ 16=1 余数为 2，因此右移 2 位。

SFL

(1) 将Ⓓ中指定的软元件的 16 位数据左移 n 位。
从最低位开始至 n 位为止将变为 0。



(2) Ⓓ中指定了位软元件时，按位数指定中指定的软元件范围向左移位。



此时实际的移位位数为 n ÷（位数指定中指定的点数）的余数。
例如，n=15，（位数指定中指定的点数）=8 位时，15 ÷ 8=1 余数为 7，因此进行 7 位的移位。

(3) n 的指定范围为 0 ～ 15。
在 n 中指定了 16 以上的值时，按 n ÷ 16 的余数值进行左移。
例如，n=18 时，18 ÷ 16=1 余数为 2，因此左移 2 位。

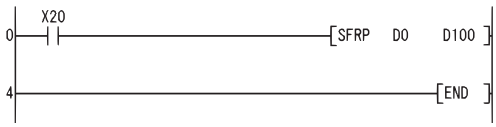
出　错

(1) 在 SFR(P)、SFL(P) 指令中无运算出错。

程序示例

(1) 以下为 X20 变为 ON 时，将 D0 的内容按 D100 中指定的位数右移的程序。

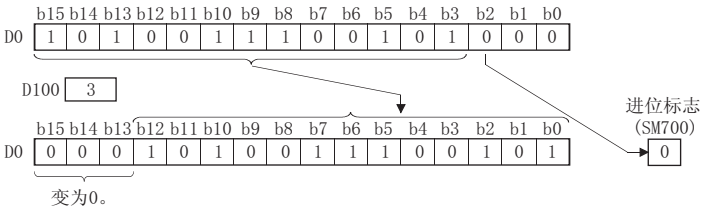
[梯形图模式]



[列表模式]

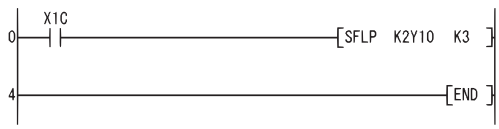
步	指令	软元件	
0	LD	X20	
1	SFRP	D0	D100
4	END		

[动作]



(2) 以下为 X1C 变为 ON 时，将 X10 ～ X17 的内容左移 3 位的程序。

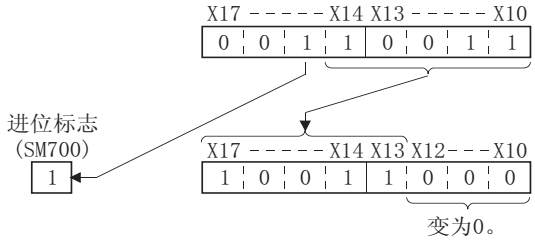
[梯形图模式]



[列表模式]

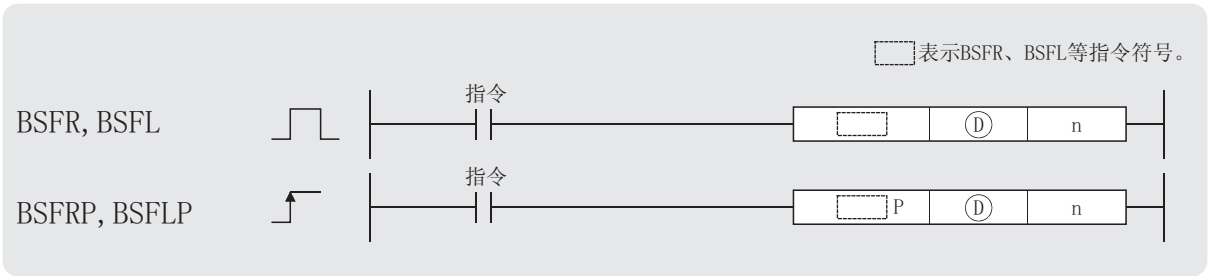
步	指令	软元件	
0	LD	X1C	
1	SFLP	K2Y10	K3
4	END		

[动作]



7.3.2 n 位数据的 1 位右移或左移 (BSFR(P)、BSFL(P))

Basic | High performance | Process | Redundant | Universal



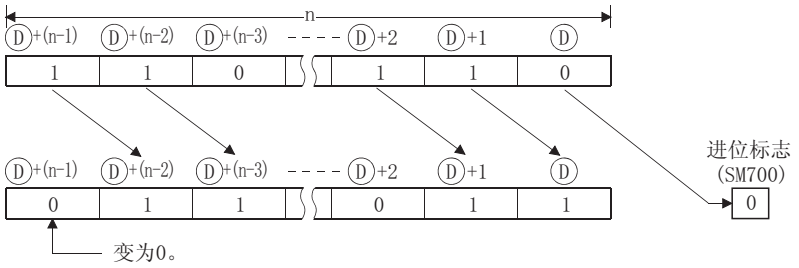
Ⓓ ：存储移位数据的软元件的起始编号 (BIN16 位)。
n ：移位次数 (0 ～ 15)(BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓓ	○	--							--
n	○	○							--

功 能

BSFR

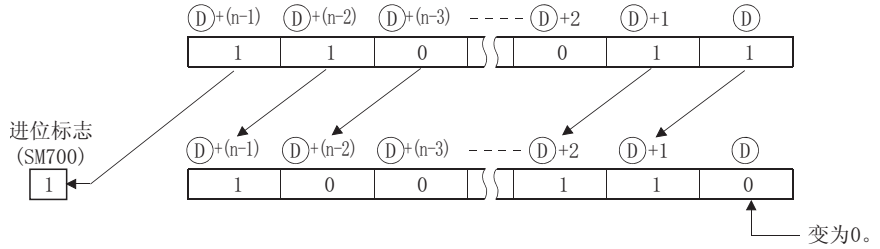
(1) 将Ⓓ中指定的软元件开始的 n 点数据右移 1 位。



(2) Ⓓ+(n-1) 中指定的软元件将变为 0。

BSFL

(1) 将Ⓓ中指定的软元件开始的 n 点数据左移 1 位。



(2) Ⓓ中指定的软元件将变为 0。

出 错

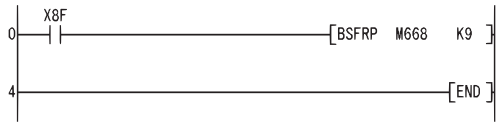
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 从ⓓ的软元件开始的 n 点范围超出了相应软元件的范围时。 (出错代码： 4101)

程序示例

(1) 以下为 X8F 变为 ON 时，将 M668 ～ M676 的数据右移的程序。

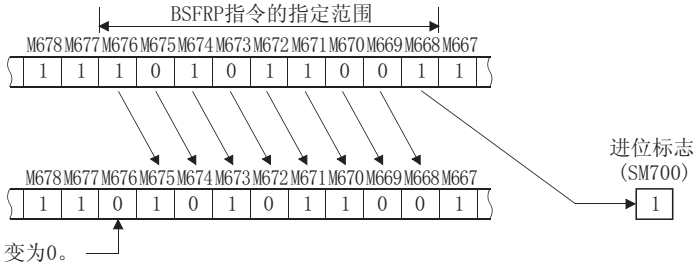
[梯形图模式]



[列表模式]

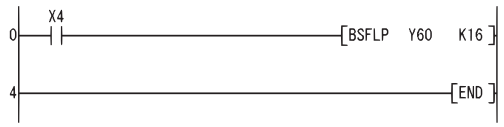
步	指令	软元件	
0	LD	X8F	
1	BSFRP	M668	K9
4	END		

[动作]



(2) 以下为 X4 变为 ON 时，将 Y60 ～ Y6F 的数据左移的程序。

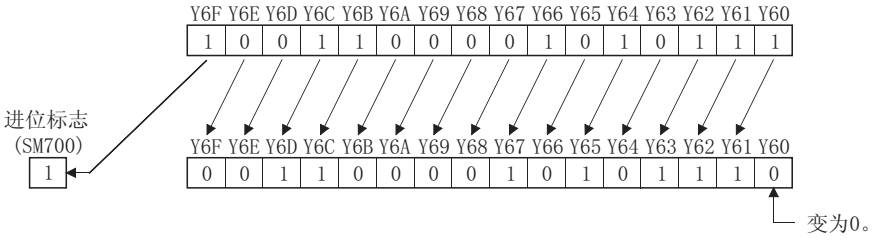
[梯形图模式]



[列表模式]

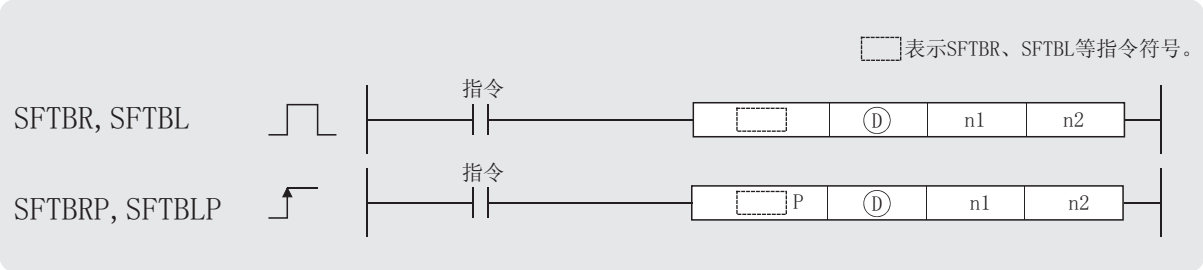
步	指令	软元件	
0	LD	X4	
1	BSFLP	Y60	K16
4	END		

[动作]



7.3.3 n 位数据的 n 位右移或左移 (SFTBR(P)、SFTBL(P))

· QnU(D)(H)CPU：序列号的前 5 位数为“ 10102 ”以后
· QnUDE(H)CPU：序列号的前 5 位数为“ 10102 ”以后



Ⓓ ：移位的软元件的起始编号 (位)。
n1 ：进行移位的位数 (BIN16 位)。
n2 ：移位数 (BIN16 位)。

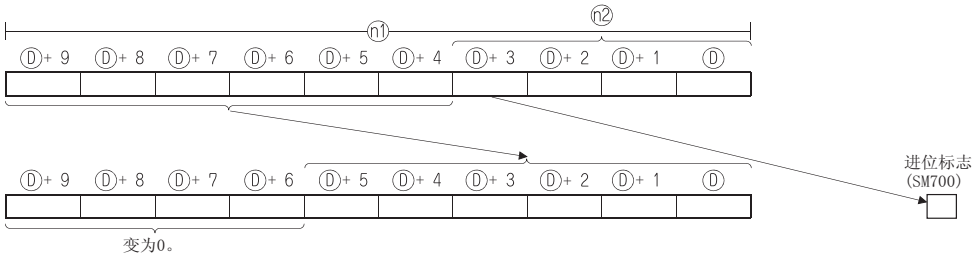
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓓ	○ *2	--	○	--					--
n1	--	○	○	○					--
n2	--	○	○	○					--

*2：T、C、ST、S 软元件不能使用。

功 能

SFTBR(P)

(1) 将从Ⓓ中指定的软元件开始至 n1 位为止的数据右移 n2 位。
n1=10，n2=4 时



(2) n1、n2 的指定应满足 n1 ＞ n2 的条件。n1 ≤ n2 的情况下，将按 n2 ÷ n1 的余数值进行移位。
但是，n2 ÷ n1 的余数值为 0 时，将变为无处理。

(3) n1 的设置范围为 1 ～ 64。

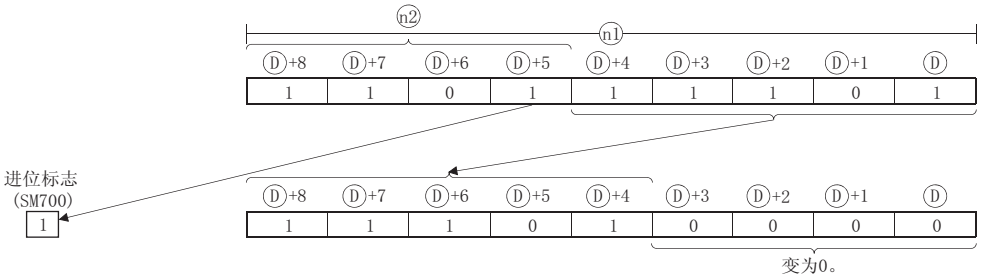
(4) 从最低位开始至 n2 位为止的数据将变为 0。n1 ＜ n2 的情况下，n2 ÷ n1 的余数值部分将变为 0。

(5) n1 或者 n2 中指定的值为 0 时，将变为无处理。

SFTBL(P)

(1) 将从ⓓ中指定的软元件开始至 n1 位为止的数据左移 n2 位。

n1=10，n2=4 时



(2) n1、n2 的指定应满足 n1 ＞ n2 的条件。n1 ≤ n2 的情况下，将按 n2 ÷ n1 的余数值进行移位。

但是，n2 ÷ n1 的余数值为 0 时，将变为无处理。

(3) n1 的设置范围为 1 ～ 64。

(4) 从最低位开始至 n2 位为止的数据将变为 0。n1 ＜ n2 的情况下，n2 ÷ n1 的余数值部分将变为 0。

(5) n1 或者 n2 中指定的值为 0 时，将变为无处理。

出　错

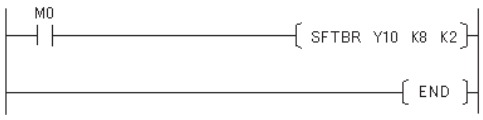
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- n1 超出了 0 ～ 64 的范围时。　　(出错代码： 4100)
- n2 为负数时。　　(出错代码： 4100)
- n1 中指定的软元件点数超出了ⓓ中指定的软元件范围时。　　(出错代码： 4101)

程序示例

(1) 以下为 M0 变为 ON 时，将ⓓ中指定的 Y10 ～ Y17(8 位) 的内容右移 2(n2) 位的程序。

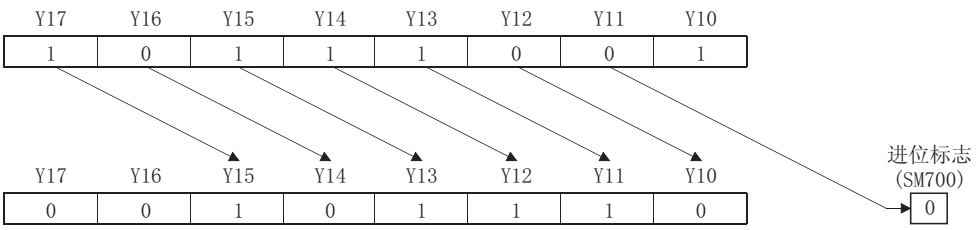
[梯形图模式]



[列表模式]

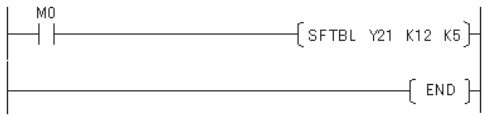
步	指令	软元件
0	LD	M0
1	SFTBR	Y10 K8 K2
5	END	

[动作]



(2) 以下为 M0 变为 ON 时，将ⓓ中指定的 Y21 ～ Y2C(12 位) 的内容左移 5(n2) 位的程序。

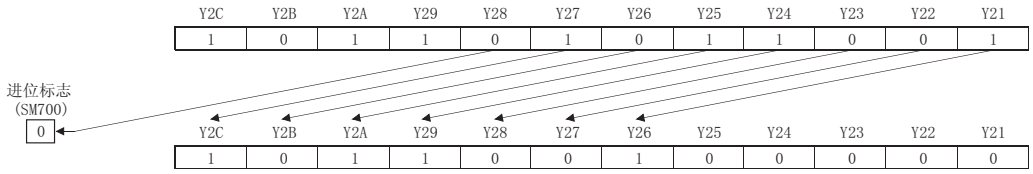
[梯形图模式]



[列表模式]

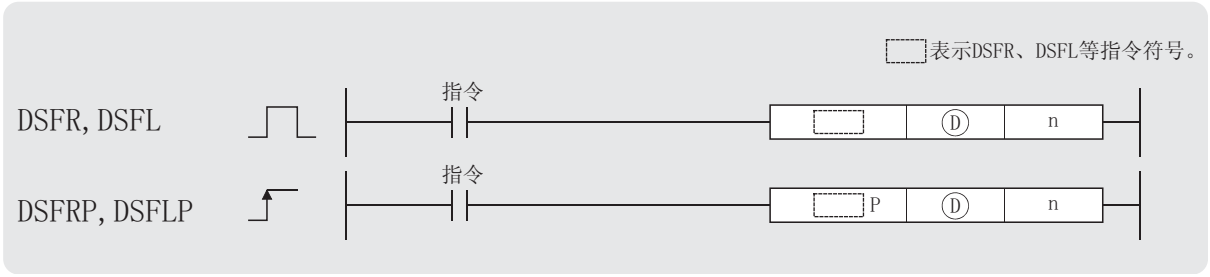
步	指令	软元件
0	LD	M0
1	SFTBL	Y21 K12 K5
5	END	

[动作]



7.3.4 n字数据的1字右移或左移 (DSFR(P)、DSFL(P))

Basic | High performance | Process | Redundant | Universal



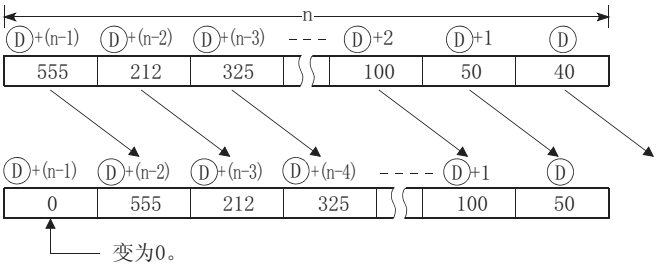
Ⓓ ：移位的软元件的起始编号 (BIN16 位)。
n ：移位的软元件的数 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓓ	--	○		--					--
n	○	○		○					--

功 能

DSFR

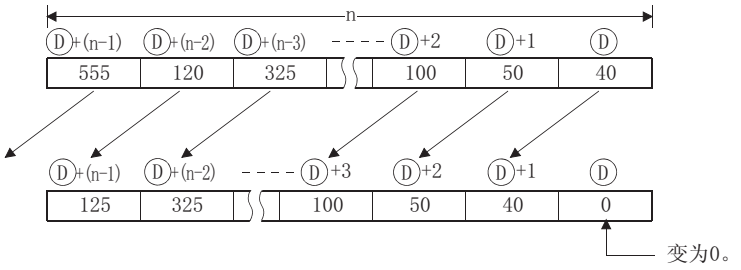
(1) 将从Ⓓ中指定的软元件开始的 n 点数据右移 1 字。



(2) Ⓓ+(n-1) 中指定的软元件将变为 0。

DSFL

(1) 将从Ⓓ中指定的软元件开始的 n 点数据左移 1 字。



(2) Ⓓ+(n-1) 中指定的软元件将变为 0。

出 错

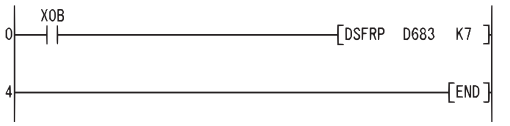
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 从ⓓ的软元件开始的 n 点的范围超出了相应软元件时。 (出错代码： 4101)

程序示例

(1) 以下为 XB 变为 ON 时，将 D683 ～ D689 的内容右移的程序。

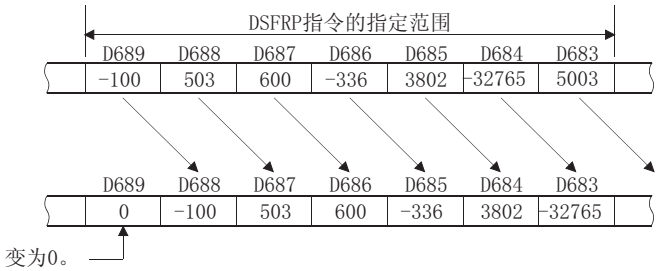
[梯形图模式]



[列表模式]

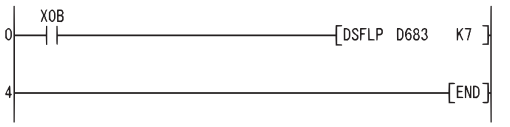
步	指令	软元件	
0	LD	X0B	
1	DSFRP	D683	K7
4	END		

[动作]



(2) 以下为 XB 变为 ON 时，将 D683 ～ D689 的内容左移的程序。

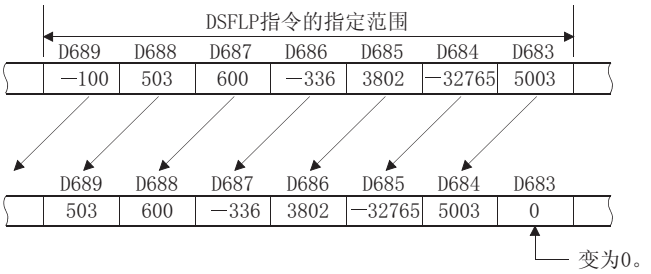
[梯形图模式]



[列表模式]

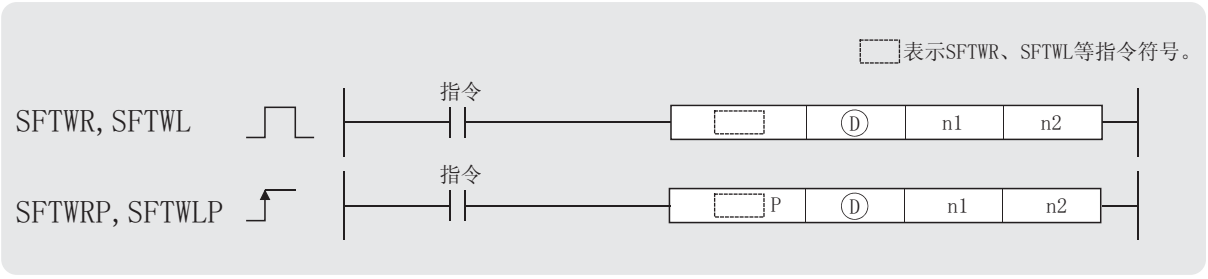
步	指令	软元件	
0	LD	X0B	
1	DSFLP	D683	K7
4	END		

[动作]



7.3.5 n 字数据的 n 字右移或左移 (SFTWR(P)、SFTWL(P))

· QnU(D)(H)CPU：序列号的前 5 位数为“ 10102 ”以后
· QnUDE(H)CPU：序列号的前 5 位数为“ 10102 ”以后



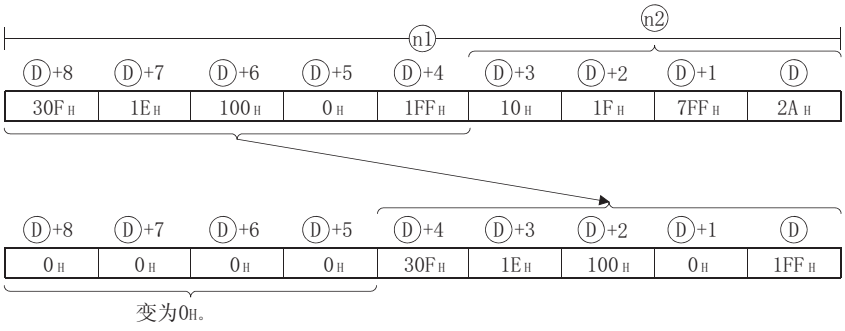
Ⓓ ：移位的软元件的起始编号 (BIN16 位)。
n1 ：进行移位的字数 (BIN16 位)。
n2 ：移位数 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓓ	--	○	○	--					--
n1	--	○	○	○					--
n2	--	○	○	○					--

功 能

SFTWR(P)

(1) 将从Ⓓ中指定的软元件开始至 n1 字为止的数据右移 n2 字。
n1=9，n2=4 时



(2) 从最高位开始至 n2 字为止将变为 0。

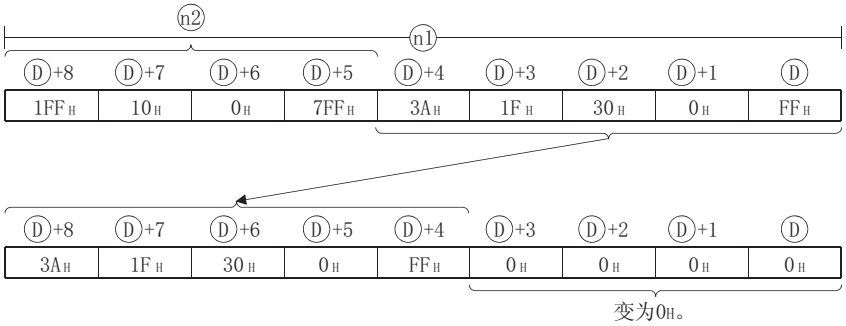
(3) n1 或者 n2 中指定的值为 0 时，将变为无处理。

(4) n1≦n2 的情况下，从Ⓓ中指定的软元件开始至 n1 字为止的数据将全部变为 0。

SFTWL(P)

(1) 将从ⓓ中指定的软元件开始至 n1 字为止的数据左移 n2 字。

n1=9，n2=4 时



(2) 从最低位开始至 n2 字为止将变为 0。

(3) n1 或者 n2 中指定的值为 0 时，将变为无处理。

(4) n1≦n2 的情况下，从ⓓ中指定的软元件开始至 n1 字为止的数据将全部变为 0。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

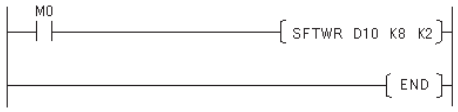
· n1 或者 n2 为负数时。 (出错代码： 4100)

· n1 中指定的软元件点数超出了ⓓ中指定的软元件的范围时。 (出错代码： 4101)

程序示例

(1) 以下为 M0 变为 ON 时，将ⓓ中指定的 D10 开始的 8(n1) 字的内容右移 2(n2) 字的程序。

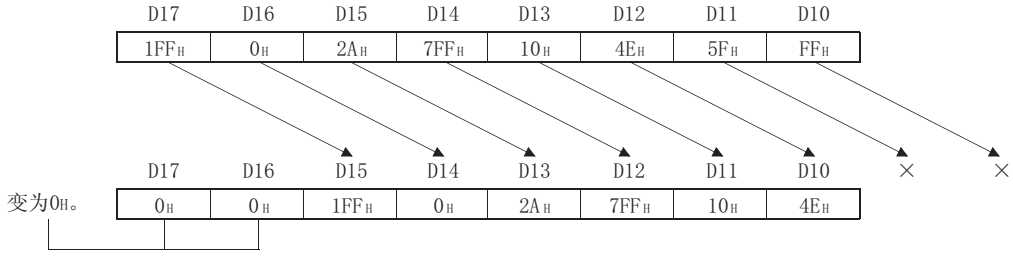
[梯形图模式]



[列表模式]

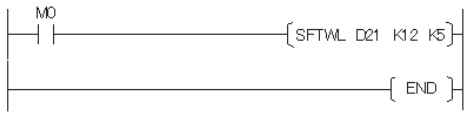
步	指令	软元件
0	LD	M0
1	SFTWR	D10 K8 K2
5	END	

[动作]



(2) 以下为 M0 变为 ON 时，将ⓓ中指定的 D21 开始的 12(n1) 字的内容左移 5(n2) 字的程序。

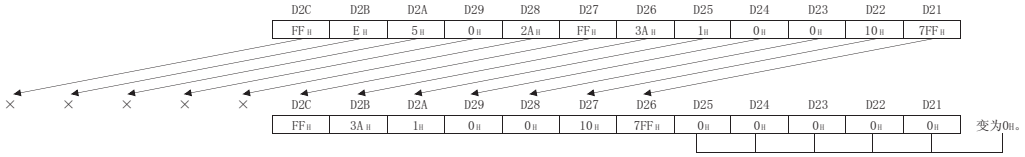
[梯形图模式]



[列表模式]

步	指令	软元件
0	LD	M0
1	SFTWL	D21 K12 K5
5	END	

[动作]



7.4 位处理指令

7.4.1 字软元件的位设置 / 复位 (BSET(P)、BRST(P))

Basic | High performance | Process | Redundant | Universal

Ⓓ ：进行位设置、复位的软元件编号 (BIN16 位)。
n ：进行位设置、复位的位数 (0 ～ 15)(BIN16 位)。

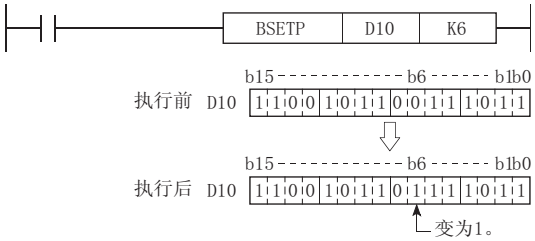
设置数据	内部软元件 位	内部软元件 字	R、ZR	J□\□ 位	J□\□ 字	U□\G□	Zn	常数 K、H	其它
Ⓓ	○							--	--
n	○							○	--

功 能

BSET

(1) 对Ⓓ中指定的字软元件的第 n 位进行设置 (1)。

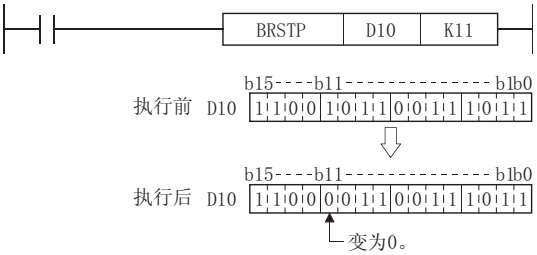
(2) n 中超过了 15 时，以低 4 位的数据执行。



BRST

(1) 对Ⓓ中指定的字软元件的第 n 位进行复位 (0)。

(2) n 中超过了 15 时，以低 4 位的数据执行。



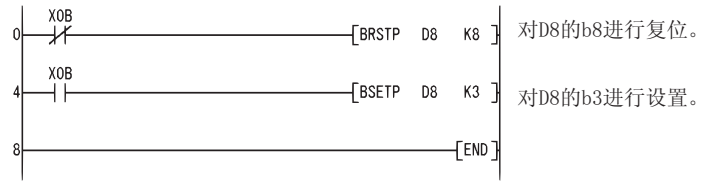
出　错

(1) 在 BSET(P)、BRST(P) 指令中无运算出错。

程序示例

(1) 以下为 XB 变为 OFF 时，对 D8 的第 8 位 (8) 进行复位 (0)，XB 变为 ON 时，对 D8 的第 3 位 (b3) 进行设置 (1) 的程序。

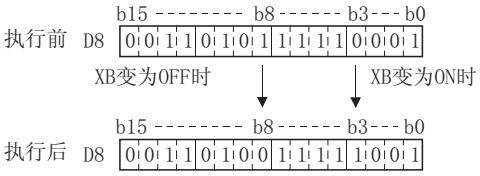
[梯形图模式]



[列表模式]

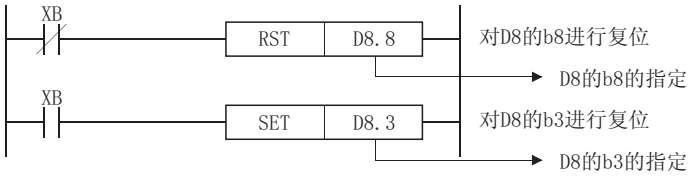
步	指令	软元件	
0	LDI	X0B	
1	BRSTP	D8	K8
4	LD	X0B	
5	BSETP	D8	K3
8	END		

[动作]



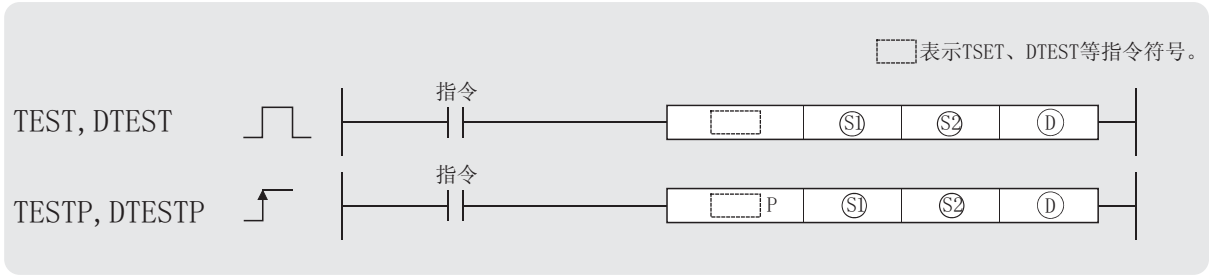
备 注

对于字软元件的位设置 / 复位，也可以通过字软元件的位指定来进行。
关于字软元件的位指定，请参阅所使用的 CPU 模块的用户手册 (功能解说 / 程序基础篇)。
如果使用了字软元件的位指定，程序示例 (1) 的处理过程如下所示。



7.4.2 位测试 (TEST(P)、DTEST(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ1 ：存储被提取的位数据的软元件编号 (BIN16 位)。

Ⓢ2 ：被提取的位数据的位置 (0 ～ 15(TEST)/0 ～ 31(DTEST))(BIN16/32 位)。

Ⓓ ：存储被提取的位数据的位软元件编号 (位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ1			○				○	--	--
Ⓢ2			○				○	○	--
Ⓓ			○				--	--	--

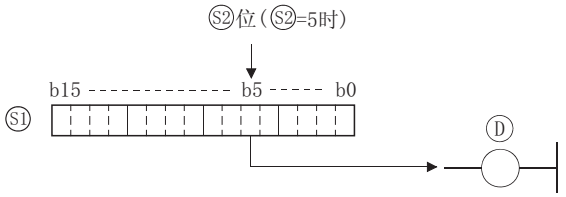
功 能

TEST

(1) 在Ⓢ1中指定的软元件内，对Ⓢ2中指定位置的位数据进行提取后，写入到Ⓓ中指定的位软元件中。

(2) 对于Ⓓ中指定的位软元件，其相应位为“ 0 ”时变为 OFF，为“ 1 ”时变为 ON。

(3) Ⓢ2中指定的位置表示 1 字数据的各个位位置 (0 ～ 15)。
在Ⓢ2中指定了 16 以上时，n ÷ 16 的余数值位置的位数据将成为对象。
例如，n=18 时，18 ÷ 16=1 余数为 2，因此变为 b2 的数据。

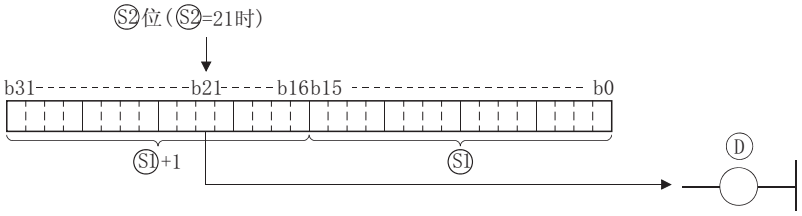


DTEST

(1) 在Ⓢ1、Ⓢ1+1 中指定的 2 字软元件内，对Ⓢ2中指定位置的位数据进行提取后，写入到Ⓓ中指定的位软元件中。

(2) 对于Ⓓ中指定的位软元件，其相应位为“ 0 ”时变为 OFF，为“ 1 ”时变为 ON。

(3) ⓢ2 中指定的位置表示 2 字数据的各个位位置 (0 ～ 31)。
在ⓢ2 中指定了 32 以上时，n ÷ 32 的余数值位置的位数据将成为对象。
例如，n=34 时，34 ÷ 32=1 余数为 2，因此变为 b2 的数据。



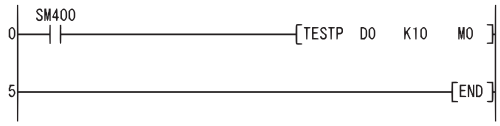
出　错

(1) 在 TEST(P)、DTEST(P) 指令中无运算出错。

程序示例

(1) 以下为根据 1 字数据 (D0) 的第 10 位的状态，对 M0 进行 ON/OFF 的程序。

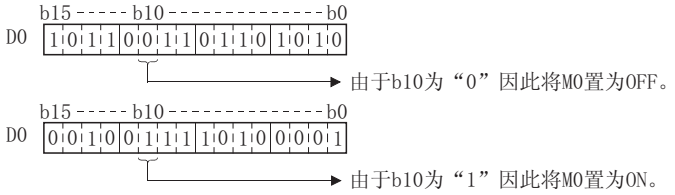
[梯形图模式]



[列表模式]

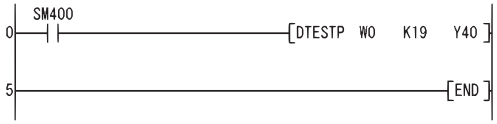
步	指令	软元件		
0	LD	SM400		
1	TESTP	D0	K10	M0
5	END			

[动作]



(2) 以下为根据 2 字数据 (W0、W1) 的第 19 位的状态，对 Y40 进行 ON/OFF 的程序。

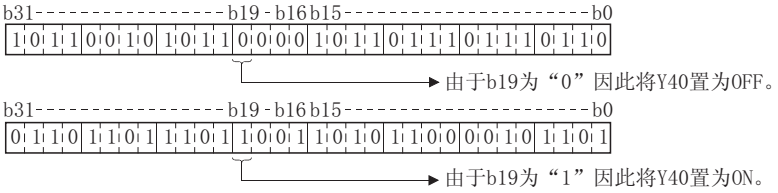
[梯形图模式]



[列表模式]

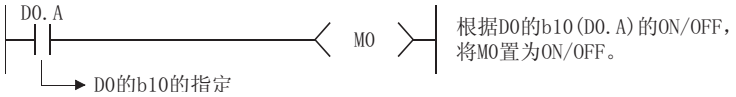
步	指令	软元件		
0	LD	SM400		
1	DTESTP	W0	K19	Y40
5	END			

[动作]



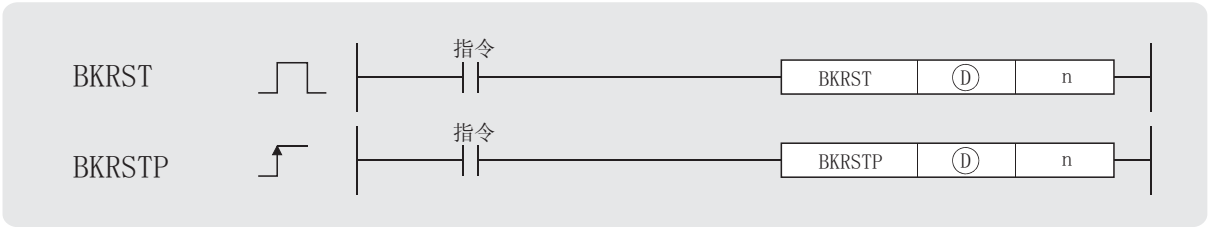
备 注

对于使用位测试指令的程序，可以用使用字软元件的位指定的程序进行替换。
对程序示例 (1) 的程序使用字软元件的位指定时的情况如下所示：



7.4.3 位软元件的批量复位 (BKRST(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ ：进行复位的软元件的起始编号 (位)。
n ：进行复位的软元件数 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓓ	○			--					--
n	○			○					--

功 能

(1) 从Ⓓ中指定的位软元件开始，对 n 点的位软元件进行复位。

软元件	状态
报警器 (F)	· 将Ⓓ中指定的报警器 (F) 号开始的 n 点软元件置为 OFF。 · 将 OFF 状态的报警器号从 SD64 ～ SD79 中删除后将剩余的向前对齐。 · 将存储在 SD64 ～ SD79 中的报警器数存储到 SD63 中。
定时器 (T) 记数器 (C)	· 将从Ⓓ中指定的定时器 (T) 或计数器 (C) 开始的 n 点的当前值置为 0 后，将线圈触点置为 OFF。
除上述以外的其它位软元件	· 将从Ⓓ中指定的软元件开始的 n 点的线圈、触点置为 OFF。

(2) 指定的软元件为 OFF 时，软元件的状态不发生变化。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 从Ⓓ的软元件开始的 n 点的范围超出了相应软元件时。 (出错代码： 4101)

程序示例

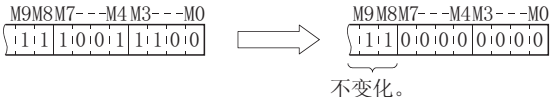
(1) 以下为 X0 变为 ON 时，将 M0 ～ M7 置为 OFF 的程序。

[梯形图模式]

[列表模式]

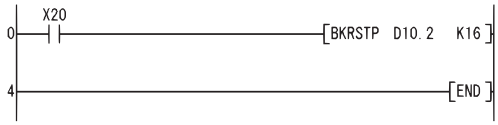
步	指令	软元件	
0	LD	X0	
1	BKRSTP	M0	K8
4	END		

[动作]



(2) 以下为 X20 变为 ON 时，将 D10 的第 2 位 (b2) ～ D11 的第 1 位 (b1) 置为 0 的程序。

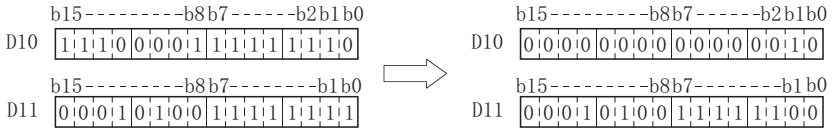
[梯形图模式]



[列表模式]

步	指令	软元件	
0	LD	X20	
1	BKRSTP	D10.2	K16
4	END		

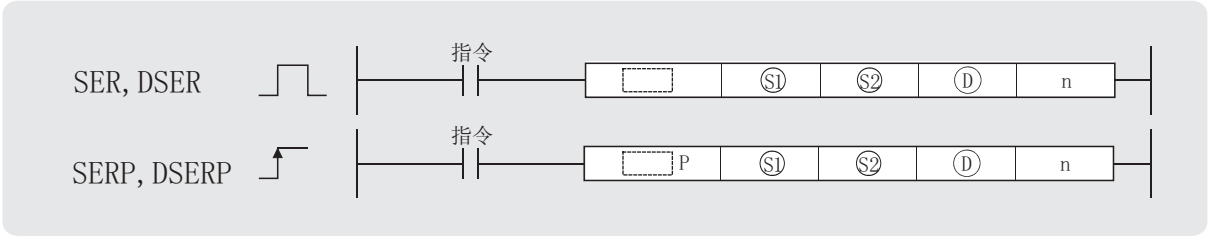
[动作]



7.5 数据处理指令

7.5.1 16 位 /32 位数据搜索 (SER(P)、DSER(P))

Basic | High performance | Process | Redundant | Universal



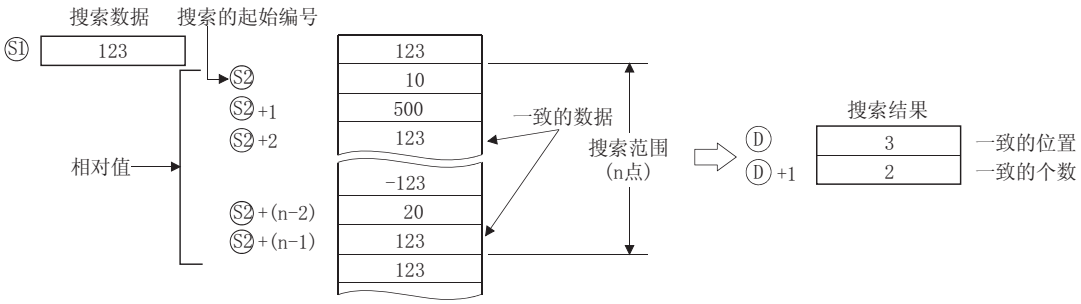
S1 ：搜索的数据或者存储搜索数据的软元件的起始编号 (BIN16/32 位)。
S2 ：被搜索的数据或者存储被搜索数据的软元件的起始编号 (BIN16 位)。
D ：存储搜索结果的软元件的起始编号 (BIN16 位)。
n ：搜索数 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
S1	○	○		○	○			○	--
S2	--	○		--	--			--	--
D	--	○		--	○			--	--
n	○	○		○	○			○	--

功 能

SER

(1) 将S1中指定的软元件的 16 位数据作为关键字，从S2中指定的软元件的 16 位数据开始至 n 点为止进行搜索。
将与关键字一致的个数存储到D+1 中指定的软元件中，将最先一致的软元件号的从S2算起的相对值存储到D中指定的软元件中。

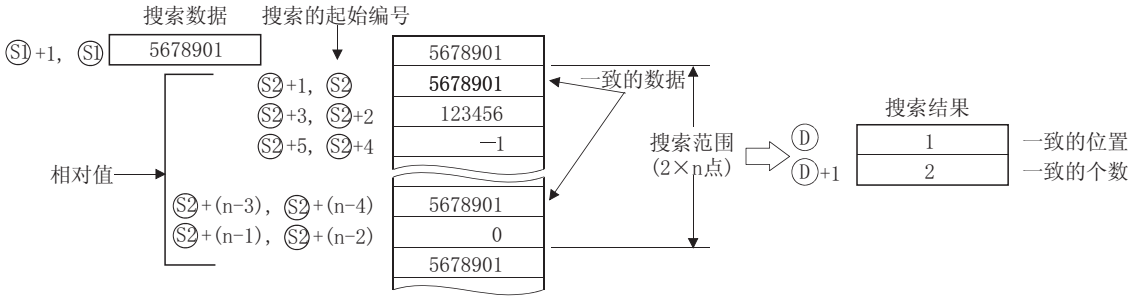


(2) n 为 0 或者负数时，执行无处理。

(3) 未搜索到一致的数据时，D、D+1 中指定的软元件将变为 “ 0 ”。

DSER

(1) 将Ⓢ1+1,Ⓢ1中指定的软元件的 32 位数据作为关键字，从Ⓢ2中指定的软元件开始以 32 位为单位对 n 点（以 16 位为单位时为 2 × n 点）的范围进行搜索。
将与关键字一致的个数存储到Ⓓ+1 中指定的软元件中，将最先一致的软元件号的从Ⓢ2算起的相对值存储到Ⓓ中指定的软元件中。



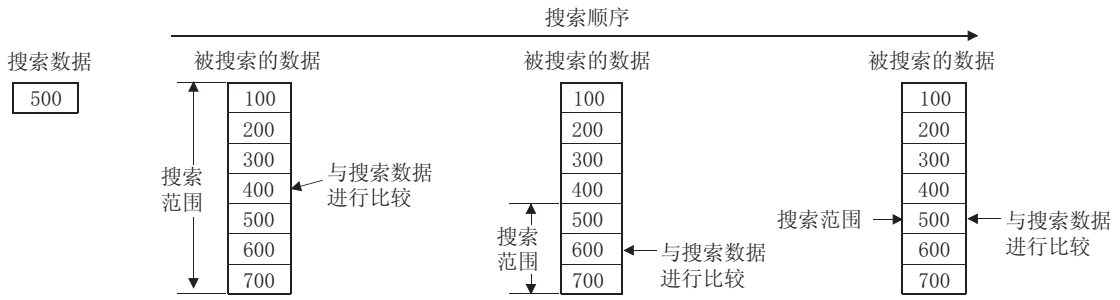
(2) n 为 0 或者负数时，执行无处理。

(3) 未搜索到一致的数据时，Ⓓ、Ⓓ+1 中指定的软元件将变为“ 0 ”。

要 点

在 SER、DSER 指令中，当被搜索的数据为升序排序时，如果将 SM702[*1] 置为 ON，通过二分搜索法进行搜索，可以加快搜索处理速度。
被搜索的数据未以升序排序时，如果将 SM702 置为 ON，将无法获得正常的搜索结果。

* 1: SM702 是用于设置搜索方法的特殊继电器。
・SM702 为 OFF 时：逐次搜索法（线性搜索法）
（从被搜索的数据的起始开始与搜索数据进行比较的方法。）
・SM702 为 ON 时：二分搜索法
（该方法对于以升序排序的数据，找出位于搜索范围中间的值，将该值与要搜索的值进行大小比较，以确定搜索方向，锁定及缩小搜索范围。如此反复执行此操作，找出要搜索的数据。）



出 错

(1) 在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

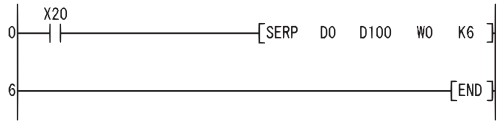
・ 从Ⓢ2的软元件开始的 n 点的范围超出了指定软元件范围时。 （出错代码： 4101）

・ Ⓓ中指定的软元件超出了相应软元件的范围时。（仅通用型 QCPU）
（出错代码： 4101）

程序示例

(1) 以下为 X20 变为 ON 时，在 D100 ～ D103 中对 D0 的内容进行搜索，并将搜索结果存储到 W0、W1 中的程序。

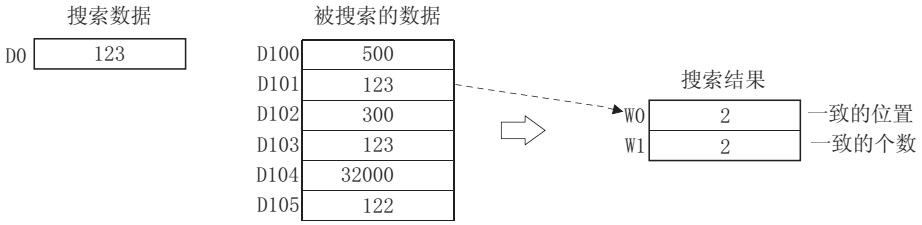
[梯形图模式]



[列表模式]

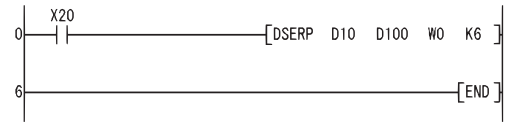
步	指令	软元件
0	LD	X20
1	SERP	D0 D100 W0 K6
6	END	

[动作]



(2) 以下为 X20 变为 ON 时，在 D100 ～ D111 中对 D11、D10 的内容进行搜索，并将搜索结果存储到 W0、W1 中的程序。

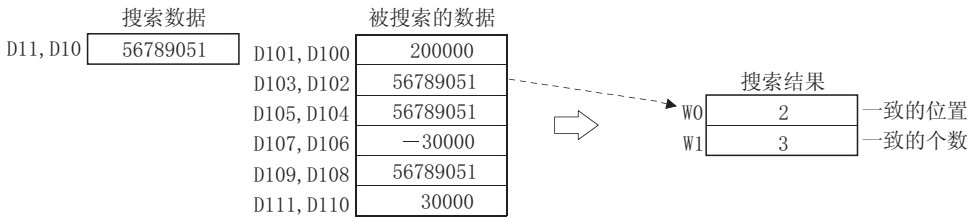
[梯形图模式]



[列表模式]

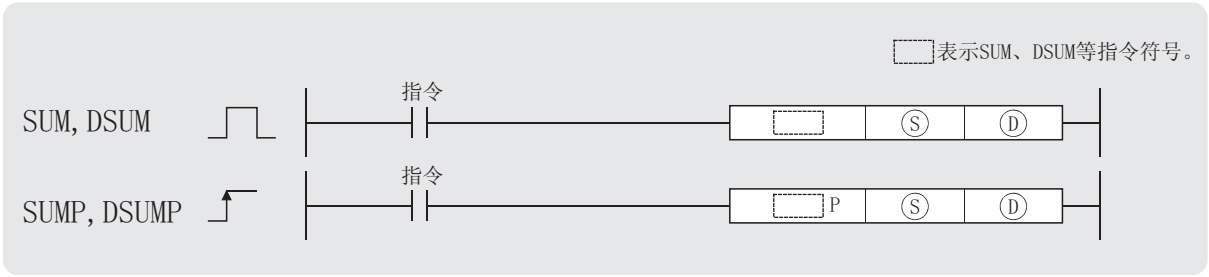
步	指令	软元件
0	LD	X20
1	DSERP	D10 D100 W0 K6
6	END	

[动作]



7.5.2 16 位 /32 位数据的位检查 (SUM(P)、DSUM(P))

Basic High performance Process Redundant Universal



Ⓢ ：对处于 1 状态的位的总数进行计数的软元件的起始编号 (BIN16/32 位)。

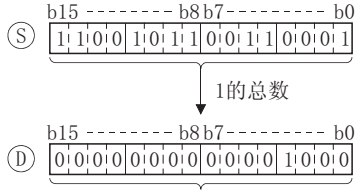
Ⓓ ：存储位的总数的软元件的起始编号 (BIN16/32 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ		○						○	--
Ⓓ		○						--	--

功 能

SUM

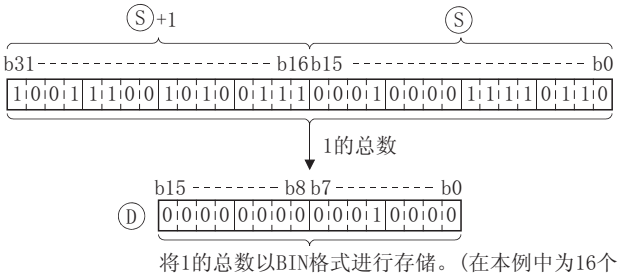
在Ⓢ中指定的软元件的 16 位数据中，将处于 1 状态的位的总数存储到Ⓓ中指定的软元件中。



将1的总数以BIN格式进行存储。(在本例中为8个。)

DSUM

在Ⓢ中指定的软元件的 32 位数据中，将处于 1 状态的位的总数存储到Ⓓ中指定的软元件中。



将1的总数以BIN格式进行存储。(在本例中为16个。)

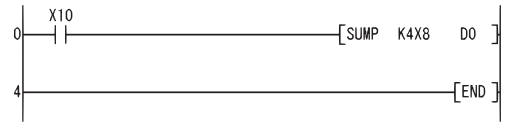
出　错

(1) 在 SUM(P)、DSUM(P) 指令中无运算出错。

程序示例

(1) 以下为 X10 变为 ON 时，将 X8 ～ X17 中处于 ON 状态的位数存储到 D0 中的程序。

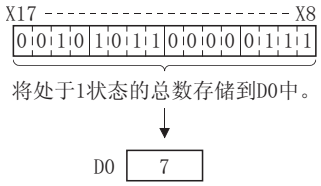
[梯形图模式]



[列表模式]

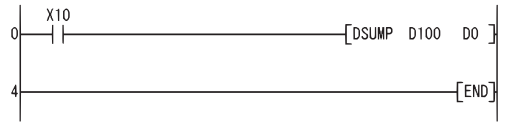
步	指令	软元件	
0	LD	X10	
1	SUMP	K4X8	D0
4	END		

[动作]



(2) 以下为 X10 变为 ON 时，将 D100、D101 中处于 ON 状态的位数存储到 D0 中的程序。

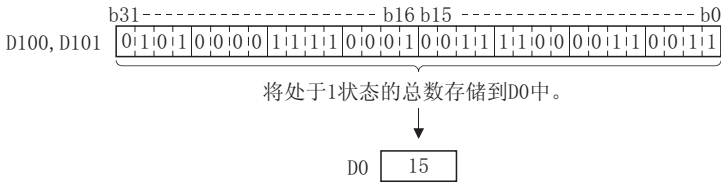
[梯形图模式]



[列表模式]

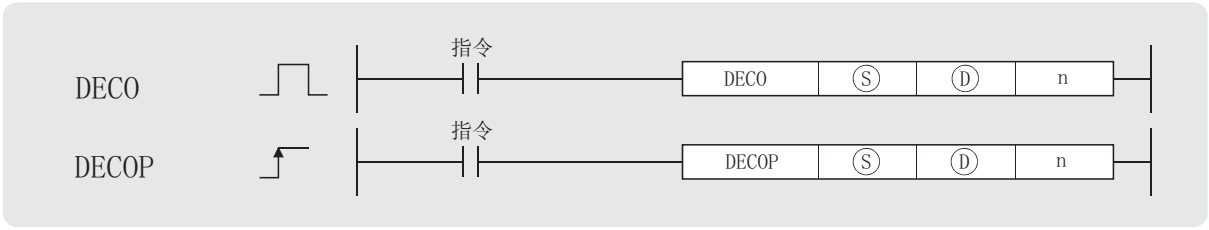
步	指令	软元件	
0	LD	X10	
1	DSUMP	D100	D0
4	END		

[动作]



7.5.3 8 位→ 256 位的解码 (DECO(P))

Basic | High performance | Process | Redundant | Universal

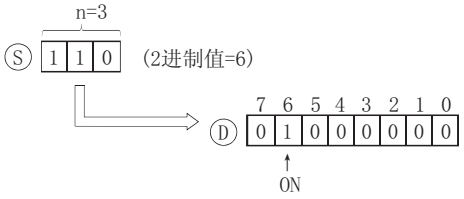


Ⓢ ：解码数据或者存储解码数据的软元件编号 (BIN16 位)。
Ⓓ ：存储解码结果的软元件的起始编号 (软元件名)。
n ：有效位长 (1 ～ 8)，0 时无处理 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○			○				○	--
Ⓓ	○			--				--	--
n	○			○				○	--

功 能

(1) 将Ⓢ的低 n 位中指定的 2 进制值对应的Ⓓ的位位置置为 ON。



(2) n 的指定范围为 1 ～ 8。

(3) n=0 时执行无处理，从Ⓓ中指定的软元件开始的 2^n 位的内容不发生变化。

(4) 位软元件作为 1 位处理，字软元件作为 16 位处理。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

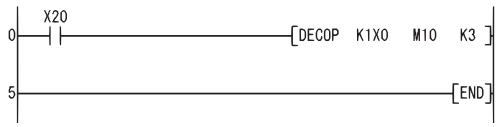
· n 超出了 1 ～ 8 的范围时。 (出错代码： 4100)

· 从⓪开始的 2^n 位的范围超出了相应软元件的范围时。 (出错代码： 4101)

程序示例

(1) 以下为 X20 变为 ON 时，从 X0 开始进行 3 位解码，并将结果存储到 M10 后面的程序。

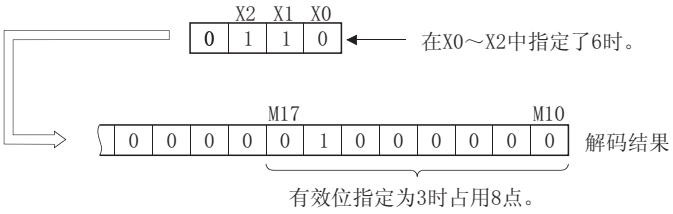
[梯形图模式]



[列表模式]

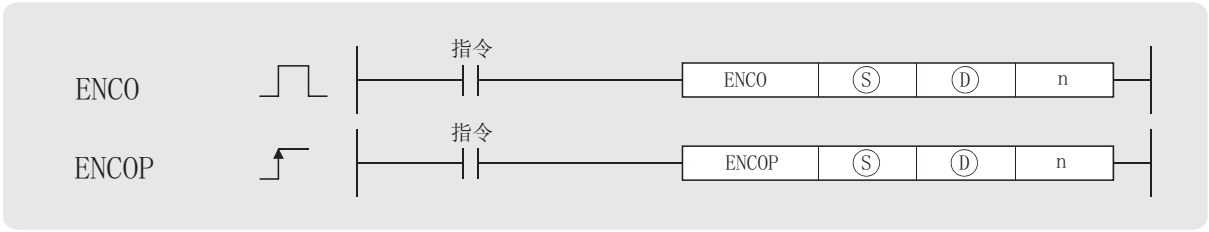
步	指令	软元件		
0	LD	X20		
1	DECOP	K1X0	M10	K3
5	END			

[动作]



7.5.4 256 → 8 位编码 (ENCO(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ ：存储编码数据的软元件的起始编号（软元件名）。

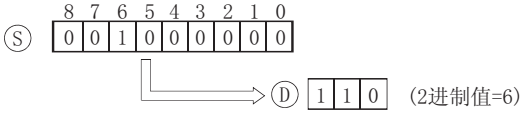
Ⓓ ：存储编码结果的软元件编号 (BIN16 位)。

n ：有效位长 (1 ～ 8)，0 时无处理 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ		○		--				--	--
Ⓓ		○		○				--	--
n		○		○				○	--

功 能

(1) 从Ⓢ的 2^n 位的数据开始，将处于 1 状态的位所对应的 2 进制值存储到Ⓓ中。



(2) n 的指定范围为 1 ～ 8。

(3) n=0 时执行无处理，Ⓓ的内容不发生变化。

(4) 位软元件作为 1 位处理，字软元件作为 16 位处理。

(5) 多个位为 1 时，按高位的位位置进行处理。

出 错

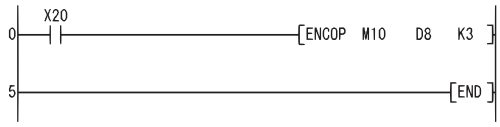
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- n 超出了 0 ～ 8 的范围时。 (出错代码： 4100)
- 从Ⓢ开始的 2^n 位的范围超出了相应软元件的范围时。 (出错代码： 4101)
- 从Ⓢ开始的 2^n 位的数据全部为 0 时。 (出错代码： 4100)

程序示例

(1) 以下为 X20 变为 ON 时，从 M10 开始进行 3 位编码，并将结果存储到 D8 中的程序。

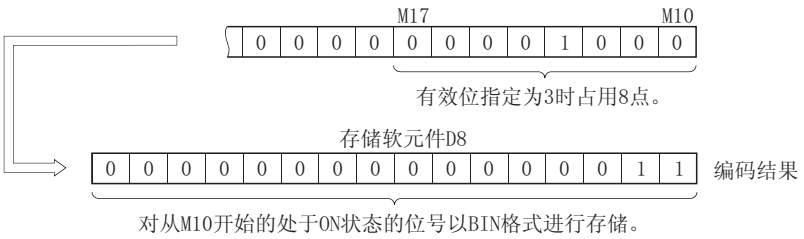
[梯形图模式]



[列表模式]

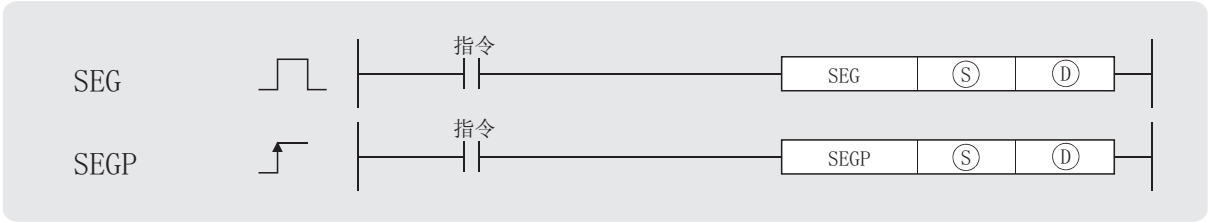
步	指令	软元件		
0	LD	X20		
1	ENCOP	M10	D8	K3
5	END			

[动作]



7.5.5 7 段解码 (SEG(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ ：解码数据或者存储编码数据的软元件的起始编号 (BIN16 位)。

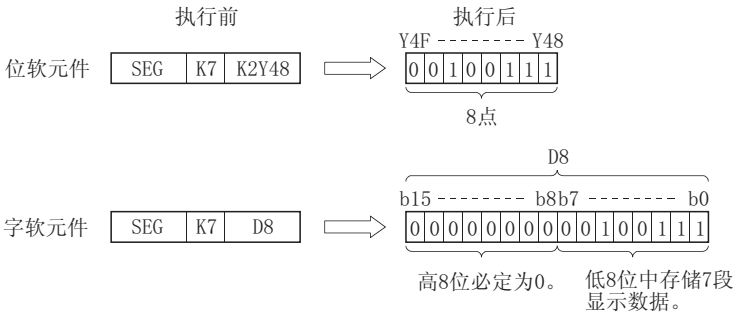
Ⓓ ：存储解码结果的软元件的起始编号 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○							○	--
Ⓓ	○							--	--

功 能

(1) Ⓢ的低 4 位中指定的 0 ～ F 的数据解码为 7 段显示数据后，存储到Ⓓ中。

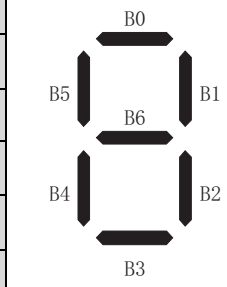
(2) 位软元件时，Ⓓ表示存储 7 段显示数据的软元件的起始编号。在字软元件中，表示存储的软元件编号。



出 错

(1) 在 SEG(P) 指令中无运算出错。

7 段解码表

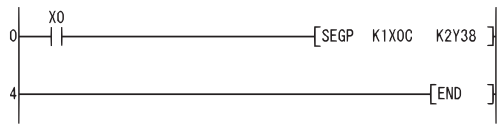
ⓢ		7 段的构成	ⓓ								显示数据
16 进制	位模式		B7	B6	B5	B4	B3	B2	B1	B0	
0	0000		0	0	1	1	1	1	1	1	0
1	0001		0	0	0	0	0	1	1	0	1
2	0010		0	1	0	1	1	0	1	1	2
3	0011		0	1	0	0	1	1	1	1	3
4	0100		0	1	1	0	0	1	1	0	4
5	0101		0	1	1	0	1	1	0	1	5
6	0110		0	1	1	1	1	1	0	1	6
7	0111		0	0	1	0	0	1	1	1	7
8	1000		0	1	1	1	1	1	1	1	8
9	1001		0	1	1	0	1	1	1	1	9
A	1010		0	1	1	1	0	1	1	1	A
B	1011		0	1	1	1	1	1	0	0	b
C	1100		0	0	1	1	1	0	0	1	C
D	1101		0	1	0	1	1	1	1	0	d
E	1110		0	1	1	1	1	0	0	1	E
F	1111		0	1	1	1	0	0	0	1	F

位软元件的起始
字软元件的最低位

程序示例

(1) 以下为 X0 变为 ON 时，将 XC ～ XF 的数据转换为 7 段显示数据后，输出到 Y38 ～ Y3F 中的程序。

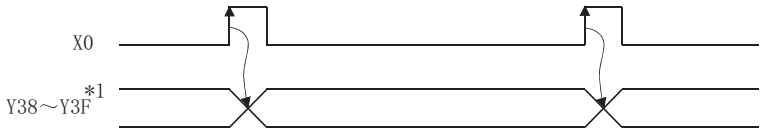
[梯形图模式]



[列表模式]

步	指令	软元件	
0	LD	X0	
1	SEGP	K1X0C	K2Y38
4	END		

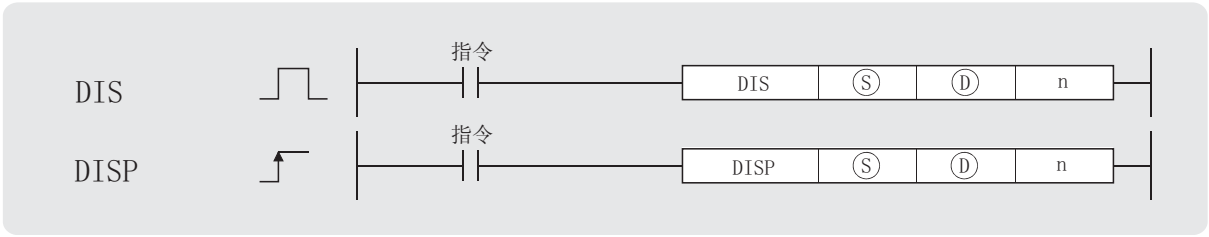
[时序图]



*1：Y38～Y3F在下一个数据被输出之前不发生变化。

7.5.6 16 位数据的 4 位分离 (DIS(P))

Basic | High performance | Process | Redundant | Universal

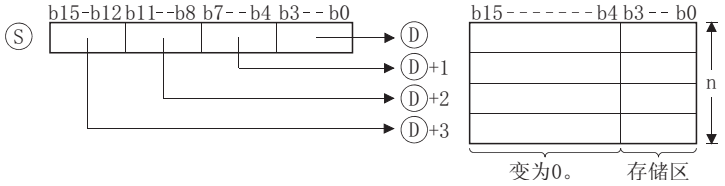


Ⓢ ：存储分离数据的软元件的起始编号 (BIN16 位)。
Ⓓ ：存储已被分离的数据的软元件的起始编号 (BIN16 位)。
n ：分离数 (1 ～ 4)，0 时无处理 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○	○		○					--
Ⓓ	--	○		--					--
n	○	○		○					--

功 能

(1) 将Ⓢ中指定的 16 位数据的低 n 位数 (1 位数 4 位) 的数据，存储到Ⓓ中指定的软元件开始的 n 点的低 4 位中。



(2) 从Ⓢ中指定的软元件开始的 n 点的高 12 位将变为 0。

(3) n 的可指定范围为 1 ～ 4。

(4) n=0 时执行无处理，从Ⓓ的软元件开始的 n 点的内容不发生变化。

出　错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

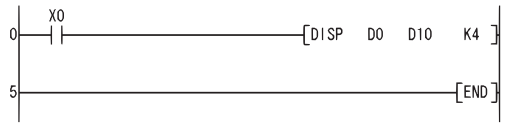
· 从ⓓ开始的 n 点的范围超出了相应软元件的范围时。　　(出错代码 : 4101)

· n 超出了 0 ～ 4 的范围时。　　(出错代码 : 4100)

程序示例

(1) 以下为 X0 变为 ON 时，对从 D0 开始的 16 位数据以 4 位为单位进行分离，并将结果存储到 D10 ～ D13 中的程序。

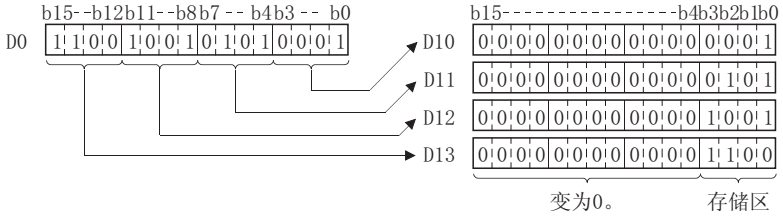
[梯形图模式]



[列表模式]

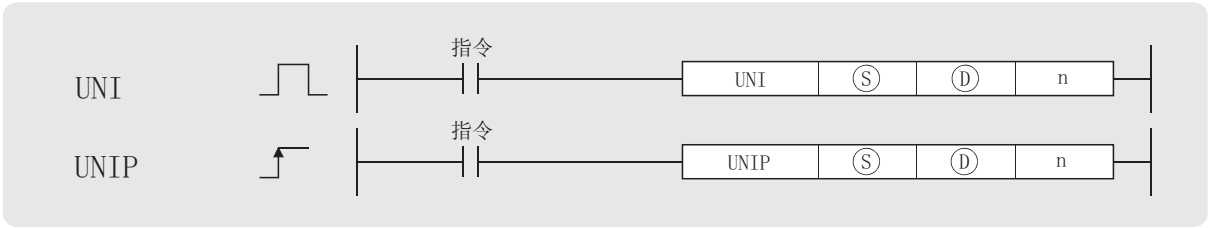
步	指令	软元件		
0	LD	X0		
1	DISP	D0	D10	K4
5	END			

[动作]



7.5.7 16 位数据的 4 位合并 (UNI(P))

Basic | High performance | Process | Redundant | Universal

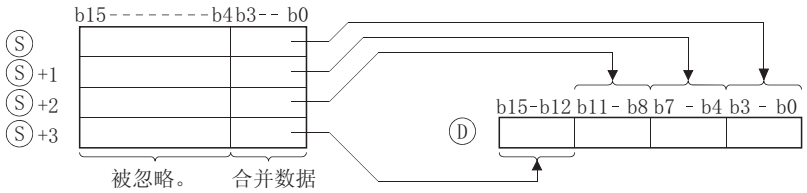


Ⓢ ：存储合并数据的软元件的起始编号 (BIN16 位)。
Ⓓ ：存储被合并的数据的软元件的起始编号 (BIN16 位)。
n ：合并数 (1 ～ 4)，0 时无处理 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	--	○		--				--	--
Ⓓ	○	○		○				--	--
n	○	○		○				○	--

功 能

(1) 将Ⓢ中指定的软元件开始的 n 点的 16 位数据的低 4 位，与Ⓓ指定的 16 位软元件进行合并。



(2) Ⓓ中指定的软元件的高位 (4-n) 的位数的位将变为 0。

(3) n 的可指定范围为 1 ～ 4。

(4) n=0 时执行无处理，Ⓓ的软元件的内容不发生变化。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

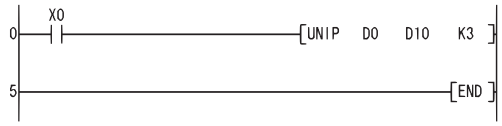
· 从Ⓢ开始的 n 点的范围超出了相应软元件的范围时。 (出错代码 : 4101)

· n 超出了 0 ～ 4 的范围时。 (出错代码 : 4100)

程序示例

(1) 以下为 X0 变为 ON 时，将 D0 ～ D2 的低 4 位数据进行合并，并将结果存储到 D10 中的程序。

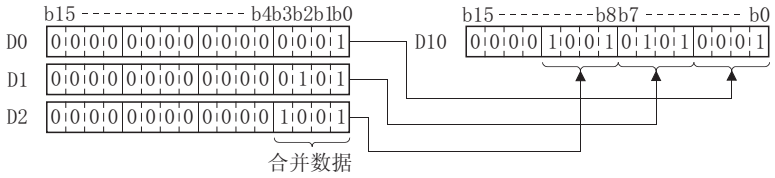
[梯形图模式]



[列表模式]

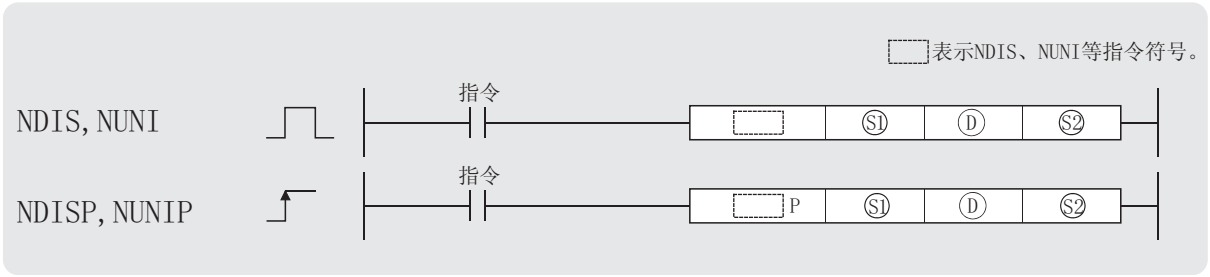
步	指令	软元件		
0	LD	X0		
1	UNIP	D0	D10	K3
5	END			

[动作]



7.5.8 任意数据的位分离、合并 (NDIS(P)、NUNI(P))

Basic | High performance | Process | Redundant | Universal



(S1) ：存储分离、合并数据的软元件的起始编号 (BIN16 位)。

(D) ：存储已被分离、合并的数据的软元件的起始编号 (BIN16 位)。

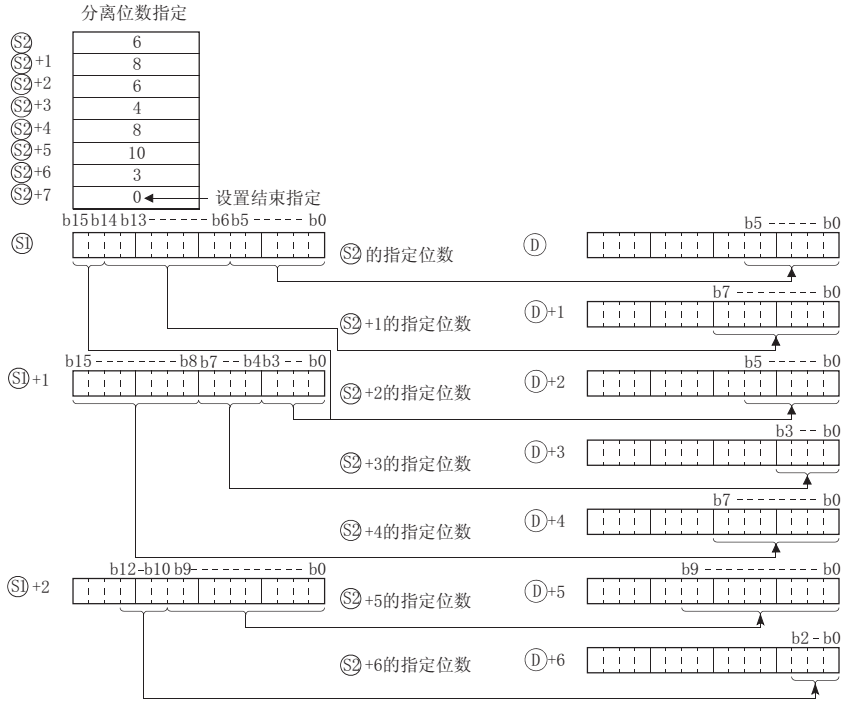
(S2) ：存储分离、合并单位的软元件的起始编号 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
(S1)	--	○		--					
(D)	--	○		--					
(S2)	--	○		--					

功 能

NDIS

(1) 将(S1)中指定的软元件号后面存储的数据的各个位，分别分离成(S2)中指定的位数，并存储到(D)中指定的软元件号后面。



(2) ⓢ2中指定的分离位数可在 1 ～ 16 位的范围内进行指定。

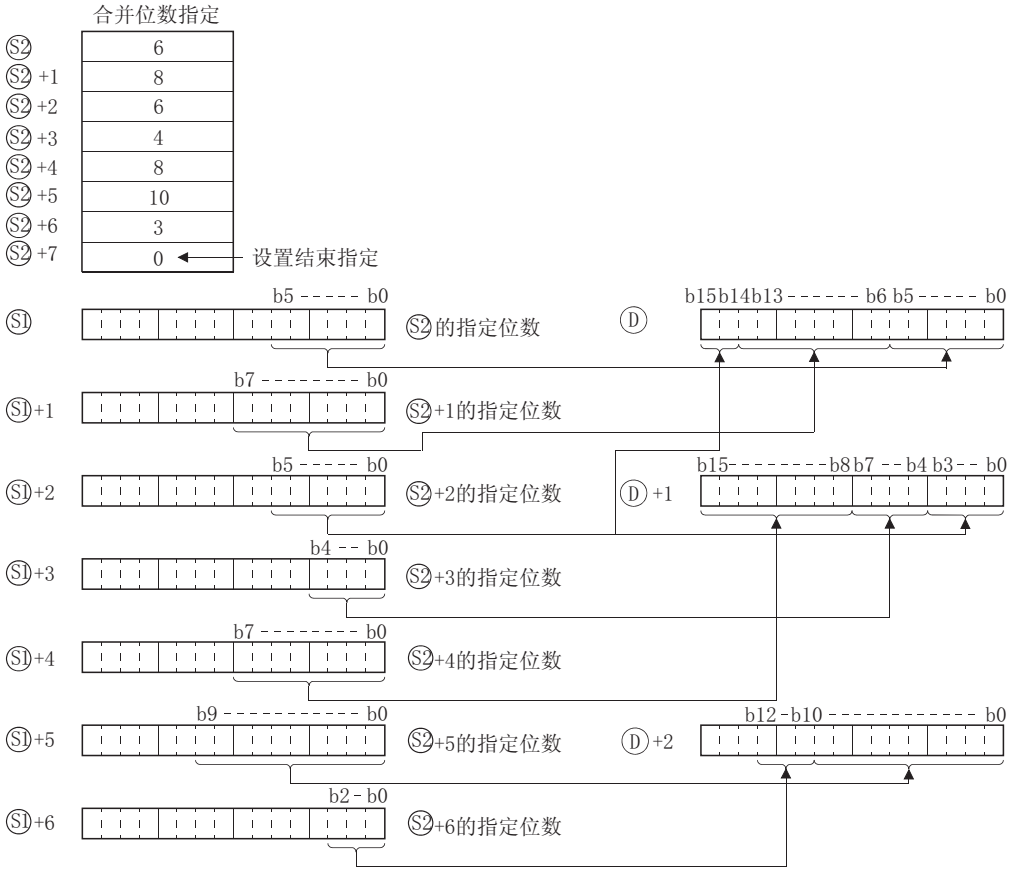
(3) 将从ⓢ2中指定的软元件号开始，至存储了“ 0 ”的软元件号为止作为分离位数进行处理。

(4) 应避免使分离数据的软元件范围（ⓢ1～ⓢ1的结束范围）与存储分离后的数据的软元件范围（ⓓ～ⓓ的结束范围）重复。如果重复，将可能无法获得正确的运算结果。

(5) 应避免使ⓢ1、ⓢ2、ⓓ中指定的软元件号重复。如果重复，将无法进行正常运算。

NUNI

(1) 将ⓢ1中指定的软元件号后面存储的数据的各个位，分别合并为ⓢ2中指定的位数，并存储到ⓓ中指定的软元件号后面。



(2) ⓢ2中指定的合并位数可在 1 ～ 16 位的范围内进行指定。

(3) 将从ⓢ2中指定的软元件号开始，至存储了“ 0 ”的软元件号为止作为合并位数进行处理。

(4) 应避免使合并数据的软元件范围（ⓢ1～ⓢ1的结束范围）与存储合并后的数据的软元件范围（ⓓ～ⓓ的结束范围）重复。如果重复，将可能无法获得正确的运算结果。

(5) 应避免使ⓢ1、ⓢ2、ⓓ中指定的软元件号重复。如果重复，将无法进行正常运算。

出　错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

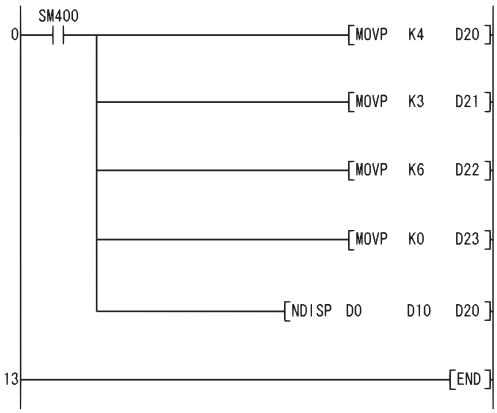
· 根据Ⓢ2 中指定的分离、合并位数指定，Ⓢ1 或者Ⓓ中指定的软元件的使用范围超出了各自的软元件的最终软元件号时。 (出错代码： 4101)

· Ⓢ2 中指定的分离、合并位数指定超出了 1 ～ 16 位的范围时。 (出错代码： 4100)

程序示例

(1) 以下为从 D0 的数据的低位开始分别进行 4、3、6 位分离后，存储到 D10 ～ D12 中的程序。

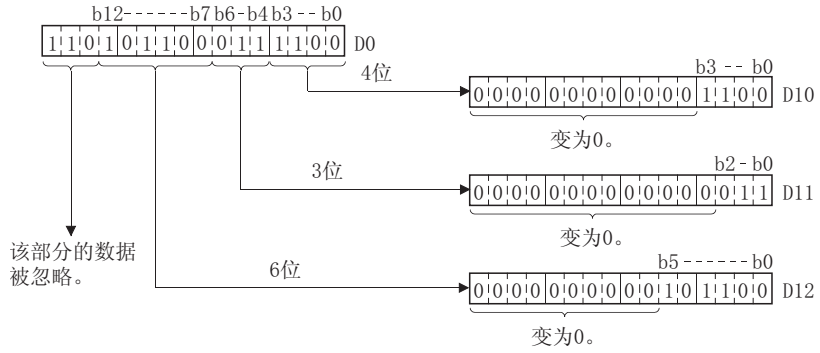
[梯形图模式]



[列表模式]

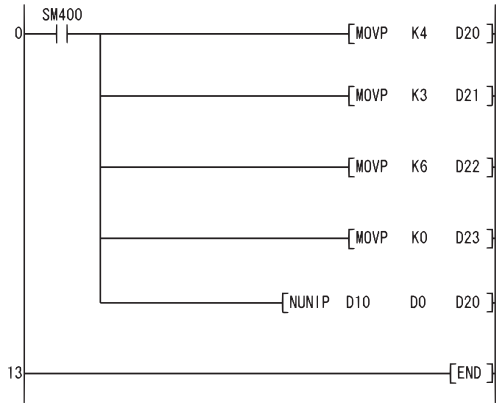
步	指令	软元件		
0	LD	SM400		
1	MOVP	K4	D20	
3	MOVP	K3	D21	
5	MOVP	K6	D22	
7	MOVP	K0	D23	
9	NDISP	D0	D10	D20
13	END			

[动作]



(2) 以下为将 D10 的数据的低 4 位、D11 的数据的低 3 位及 D12 的数据的低 6 位进行合并后，存储到 D0 中的程序。

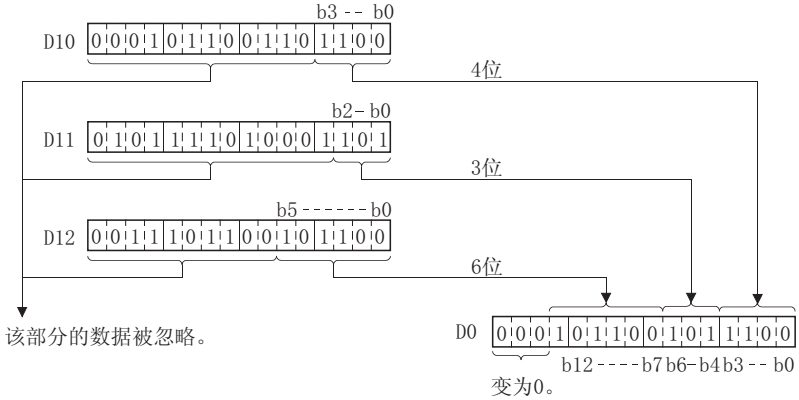
[梯形图模式]



[列表模式]

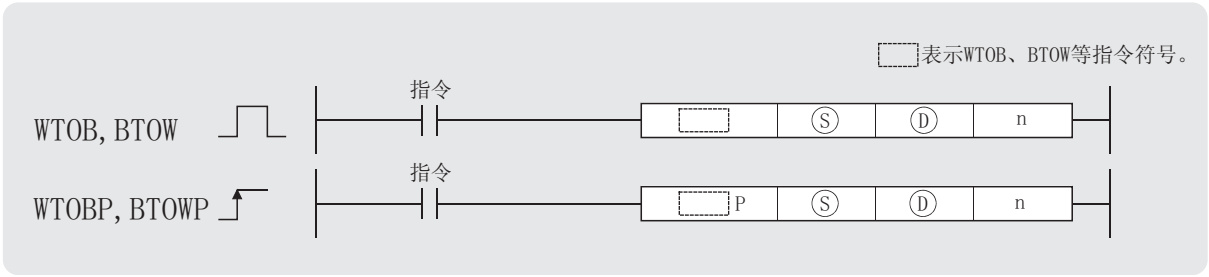
步	指令	软元件		
0	LD	SM400		
1	MOVP	K4	D20	
3	MOVP	K3	D21	
5	MOVP	K6	D22	
7	MOVP	K0	D23	
9	NUNIP	D10	D0	D20
13	END			

[动作]



7.5.9 以字节为单位的数据分离、合并 (WTOB(P)、BTOW(P))

Basic | High performance | Process | Redundant | Universal



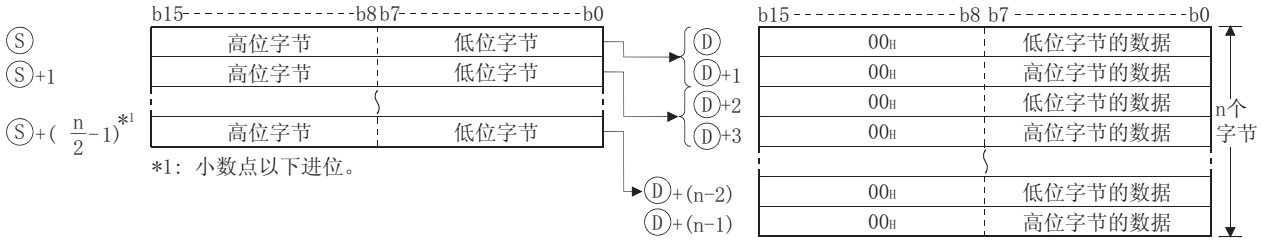
Ⓢ ：存储以字节为单位的分离、合并数据的软元件的起始编号 (BIN16 位)。
Ⓓ ：存储以字节为单位已被分离、合并的结果的软元件的起始编号 (BIN16 位)。
n ：分离、合并的字节的个数 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	--	○		--					--
Ⓓ	--	○		--					--
n	○	○		○					--

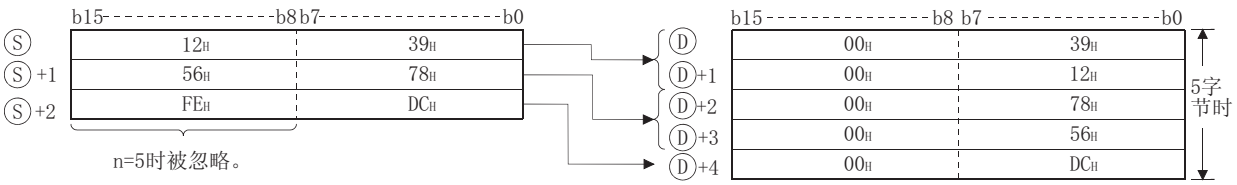
功 能

WTOB

(1) 将Ⓢ中指定的软元件号后面存储的 16 位数据，分离为 n 字节后，存储到Ⓓ中指定的软元件号后面。



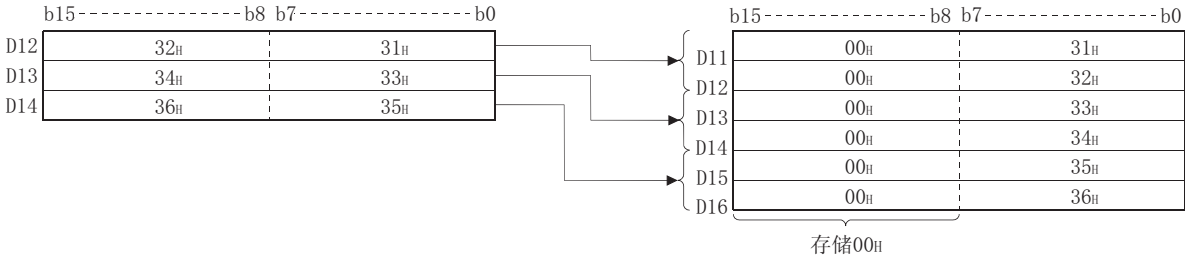
例如，n=5 时，将Ⓢ～ (Ⓢ+2) 的低 8 位为止的数据存储到Ⓓ～ (Ⓓ+4) 中。



(2) 通过在 n 中设置字节数，Ⓢ中指定的 16 位数据的范围以及存储Ⓓ中指定的字节数据的软元件的范围将被自动确定。

(3) n 中指定的字节数为“ 0 ”时，不执行处理。

(4) 存储Ⓓ中指定的字节数据的软元件的高 8 位中，将自动地存储“ 00H ”。



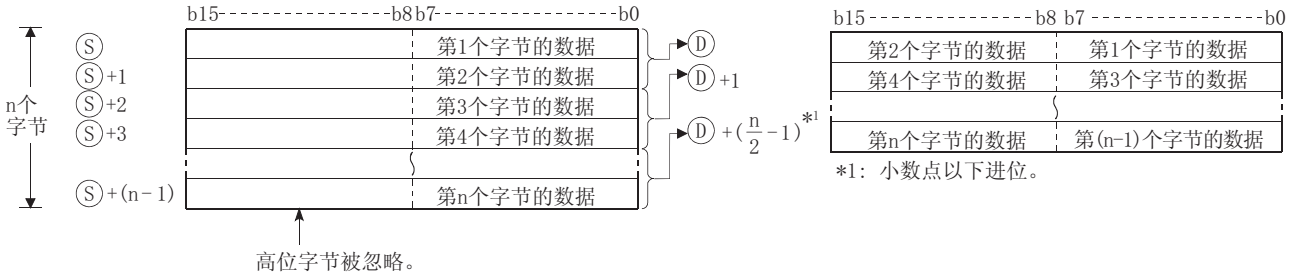
(5) 即使在存储分离数据的软元件范围 (Ⓢ～Ⓢ+($\frac{n}{2}$-1)) 与存储已分离数据的软元件范围 (Ⓓ～Ⓓ+(n-1)) 重复的情况下，也可正常执行处理。

BTOW

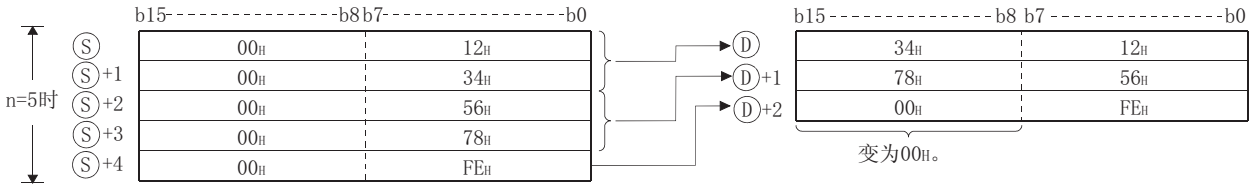
(1) 将Ⓢ中指定的软元件号后面存储的 n 字的 16 位数据的低 8 位，以字为单位进行合并后，存储到Ⓓ中指定的软元件号后面。

Ⓢ中指定的软元件号后面存储的 n 字数据的高 8 位将被忽略。

此外，n 为奇数时，存储了第 n 个字节数据的软元件的高 8 位中将存储 0。



例如，n=5 时，将Ⓢ～ (Ⓢ+4) 的低 8 位的数据合并后，存储到Ⓓ～ (Ⓓ+2) 中。



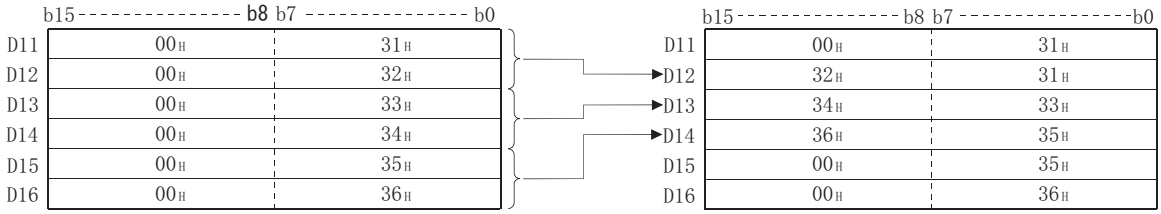
(2) 通过在 n 中设置字节数，Ⓢ中指定的字节数据的范围以及存储Ⓓ中指定的合并数据的软元件的范围将被自动确定。

(3) n 中指定的字节数为“ 0 ”时，不执行处理。

(4) 存储Ⓢ中指定的字节数据的软元件的高 8 位将被忽略，低 8 位将成为对象。

(5) 即使在存储合并数据的软元件范围 (Ⓢ～Ⓢ+(n-1)) 与存储已合并数据的软元件范围 (Ⓓ～Ⓓ+($\frac{n}{2}$-1)) 重复的情况下，也可正常执行处理。

例如，将 D11 ～ D16 的低 8 位存储到 D12 ～ D14 中时的情况如下图所示。



出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· Ⓢ中指定的软元件号后面，n 中指定的字节数的范围超出了相应软元件的范围时。

(出错代码： 4101)

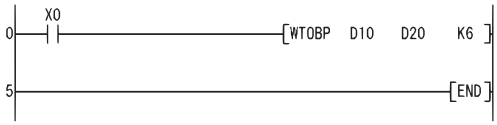
· Ⓓ中指定的软元件号后面，n 中指定的字节数的范围超出了相应软元件的范围时。

(出错代码： 4101)

程序示例

(1) 以下为从 X0 变为 ON 时，将 D10 ～ D12 的数据以字节为单位进行分离后，存储到 D20 ～ D25 中的程序。

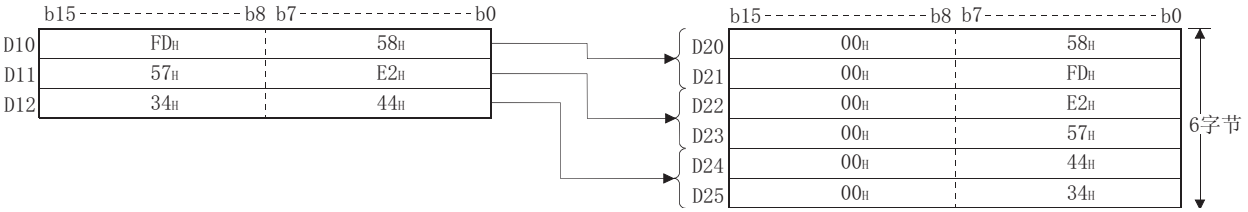
[梯形图模式]



[列表模式]

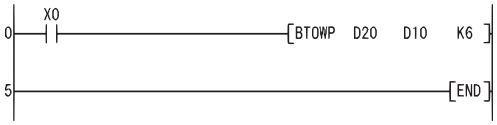
步	指令	软元件		
0	LD	X0		
1	WTOBP	D10	D20	K6
5	END			

[动作]



(2) 以下为从 X0 变为 ON 时，将 D20 ～ D25 的低 8 位数据合并后，存储到 D10 ～ D12 中的程序。

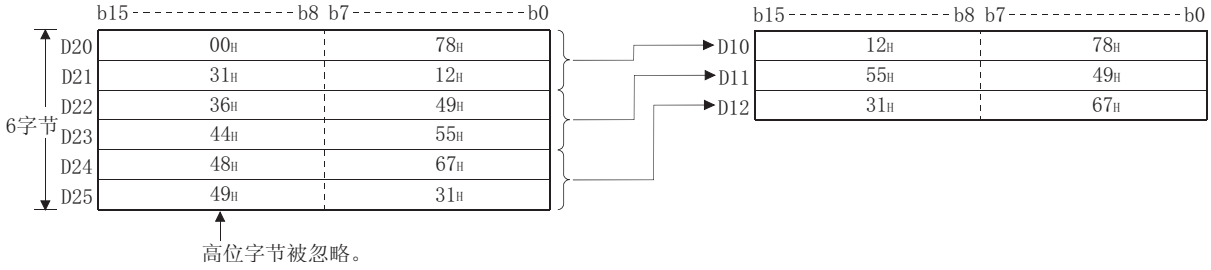
[梯形图模式]



[列表模式]

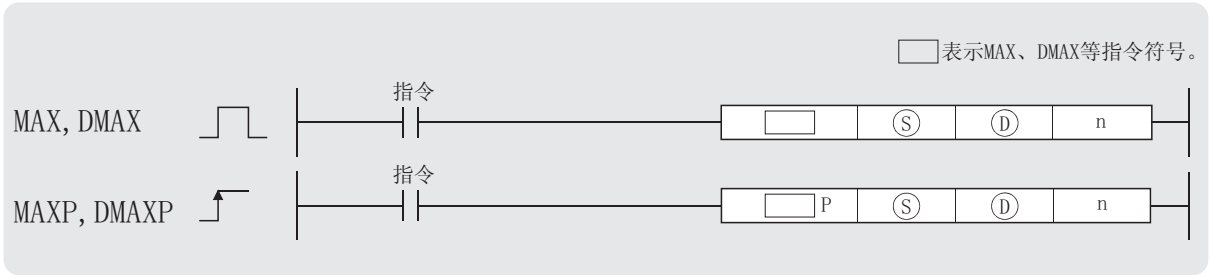
步	指令	软元件		
0	LD	X0		
1	BTOWP	D20	D10	K6
5	END			

[动作]



7.5.10 16 位 /32 位数据的最大值搜索 (MAX(P)、DMAX(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ ：搜索最大值的软元件的起始编号 (BIN16/32 位)。

Ⓓ ：存储最大值搜索结果的软元件的起始编号 (BIN16/32 位)。

n ：搜索的数据数 (BIN16 位)。

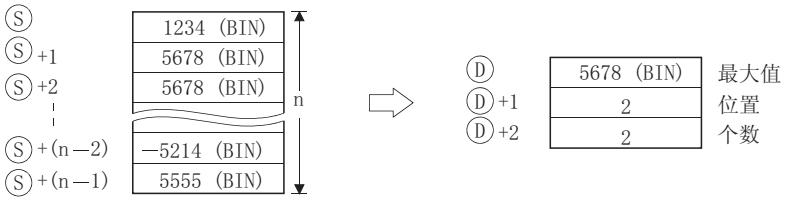
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	--	○		--					--
Ⓓ	--	○		--					--
n	○	○		○					--

功 能

MAX

从Ⓢ中指定的软元件开始，在 n 点的 16 位 BIN 数据中搜索最大值，并将最大值存储到Ⓓ中指定的软元件中。

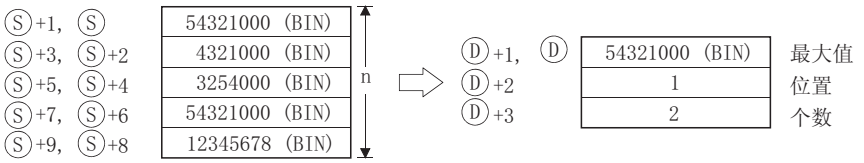
从Ⓢ中指定的软元件开始搜索，将最先检测出的存储最大值的软元件号从Ⓢ算起的点数存储到Ⓓ+1 中，将最大值的个数存储到Ⓓ+2 中。



DMAX

从Ⓢ中指定的软元件开始，在 n 点的 32 位 BIN 数据中搜索最大值，并将最大值存储到Ⓓ、Ⓓ+1 中指定的软元件中。

从Ⓢ中指定的软元件开始搜索，将最先检测出的存储最大值的软元件号从Ⓢ算起的点数存储到Ⓓ+2 中，将最大值的个数存储到Ⓓ+3 中。



出　错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· 从Ⓢ的软元件开始的 n 点的范围超出了指定软元件范围时。　　(出错代码： 4101)

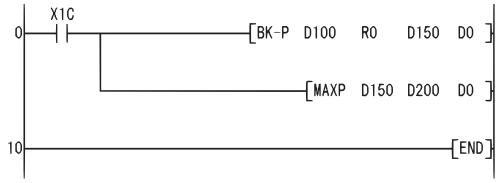
· Ⓓ中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)

(出错代码： 4101)

程序示例

(1) 以下为 X1C 变为 ON 时，将 D100 ～ D103 中存储的数据值与 R0 ～ R3 中存储的数据值进行减法运算，在其结果中搜索出最大值后，存储到 D200 ～ D202 中的程序。

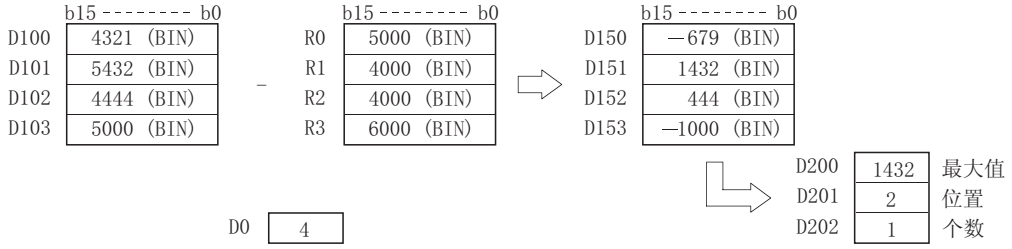
[梯形图模式]



[列表模式]

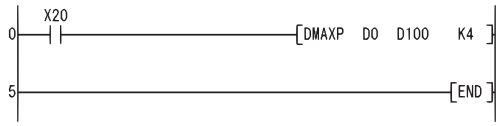
步	指令	软元件			
0	LD	X1C			
1	BK-P	D100	R0	D150	D0
6	MAXP	D150	D200	D0	
10	END				

[动作]



(2) 以下为 X20 变为 ON 时，在 D0 ～ D7 的 32 位数据中搜索最大值，并将结果存储到 D100 ～ D103 中的程序。

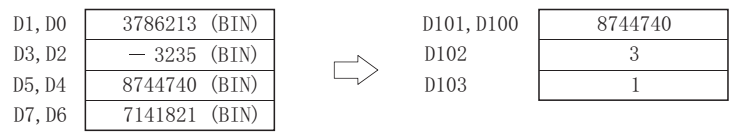
[梯形图模式]



[列表模式]

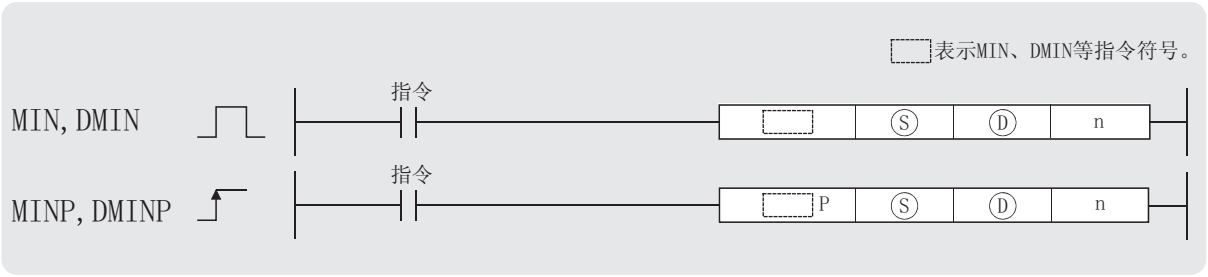
步	指令	软元件		
0	LD	X20		
1	DMAXP	D0	D100	K4
5	END			

[动作]



7.5.11 16 位 /32 位数据的最小值查找 (MIN(P)、DMIN(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ ： 搜索最小值的软元件的起始编号 (BIN16/32 位)。
Ⓓ ： 存储最小值搜索结果的软元件的起始编号 (BIN16/32 位)。
n ： 搜索的数据数 (BIN16 位)。

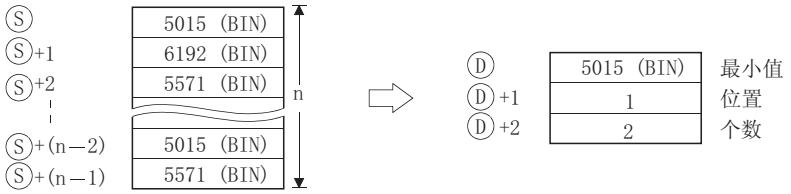
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	--	○		--					--
Ⓓ	--	○		--					--
n	○	○		○					--

功 能

MIN

从Ⓢ中指定的软元件开始，在 n 点的 16 位 BIN 数据中搜索最小值，并将最小值存储到Ⓓ中指定的软元件中。

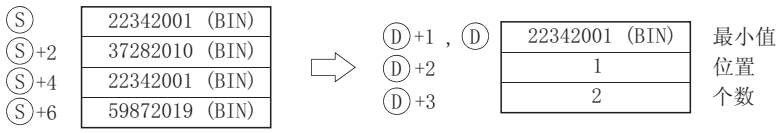
从Ⓢ中指定的软元件开始搜索，将最先检测出的存储最小值的软元件号从Ⓢ算起的点数存储到Ⓓ+1 中，将最小值的个数存储到Ⓓ+2 中



DMIN

从Ⓢ中指定的软元件开始，在 n 点的 32 位 BIN 数据中搜索最小值，并将最小值存储到Ⓓ、Ⓓ+1 中指定的软元件中。

从Ⓢ中指定的软元件开始搜索，将最先检测出的存储最小值的软元件号从Ⓢ算起的点数存储到Ⓓ+2 中，将最小值的个数存储到Ⓓ+3 中。



出　错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· 从Ⓢ的软元件开始的 n 点的范围超出了指定软元件范围时。　　(出错代码： 4101)

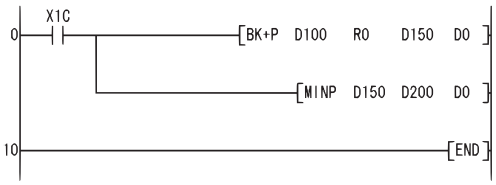
· Ⓓ中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)

(出错代码： 4101)

程序示例

(1) 以下为 X1C 变为 ON 时，将 D100 ～ D103 中存储的数据值与 R0 ～ R3 中存储的数据值进行加法运算，在其结果中搜索出最小值后，存储到 D200 ～ D202 中的程序。

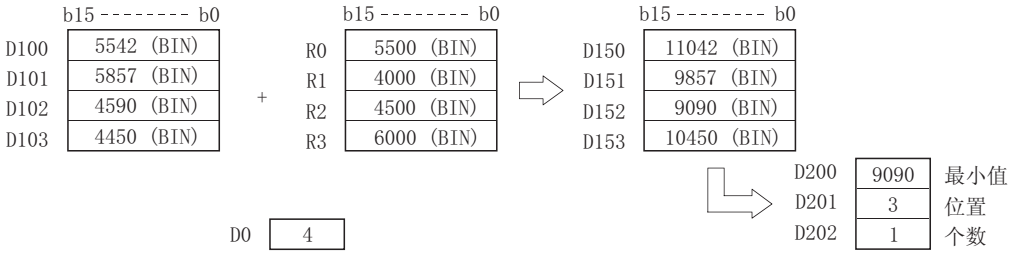
[梯形图模式]



[列表模式]

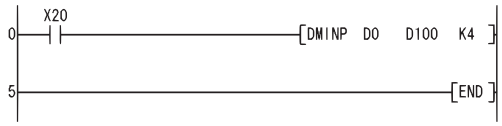
步	指令	软元件			
0	LD	X1C			
1	BK+P	D100	R0	D150	D0
6	MINP	D150	D200	D0	
10	END				

[动作]



(2) 以下为 X20 变为 ON 时，在 D0 ～ D7 的 32 位数据中搜索最小值，并将结果存储到 D100 ～ D103 中的程序。

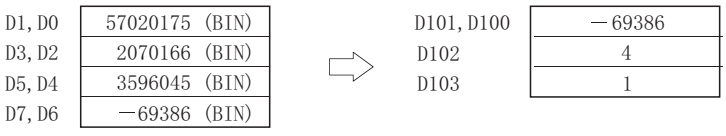
[梯形图模式]



[列表模式]

步	指令	软元件		
0	LD	X20		
1	DMINP	D0	D100	K4
5	END			

[动作]

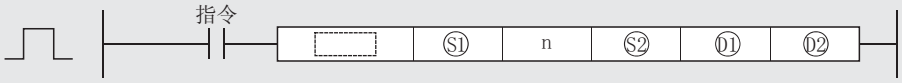


7.5.12 16 位 /32 位数据的排序 (SORT、DSORT)

Basic | High performance | Process | Redundant | Universal

□表示SORT、DSORT等指令符号。

SORT, DSORT



(S1) ：排序的表格的起始软元件编号 (BIN16/32 位)。
n ：排序的数据数 (BIN16 位)。
(S2) ：1 次执行中进行比较的数据数 (BIN16 位)。
(D1) ：排序结束时置为 ON 的软元件编号 (位)。
(D2) ：系统用软元件 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
(S1)	--	○		--					--
n	○	○		○					--
(S2)	○	○		○					--
(D1)	○	--		--					--
(D2)	--	○		--					--

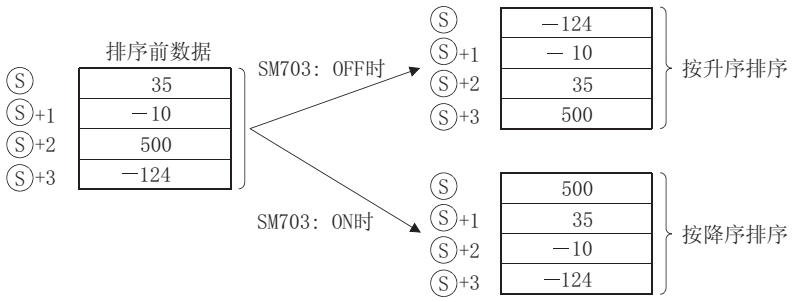
功 能

SORT

(1) 将从(S1)开始的 n 点的 BIN16 位数据进行升序 / 降序排序 (排列替换)。
排序指定是通过 SM703 的 ON/OFF 进行的。

· SM703 为 OFF 时： 升序排序

· SM703 为 ON 时： 降序排序



(2) 通过 SORT 指令进行排序时，需要数次扫描。
至执行结束为止的扫描次数为，将至排序执行结束为止的最多执行次数，与(S2)中指定的 1 次执行中进行比较的数据数相除后的值。(小数点以下进位。)
如果(S2)的值较大，则至排序结束为止的扫描次数将变少，但扫描时间将延长。

(3) 至排序执行结束为止的最多执行次数应通过下述公式算出：
至排序执行结束为止的最多执行次数 =(n) × (n-1) ÷ 2(次)

例 n=10 时，需要 10 × (10-1) ÷ 2=45(次)。
此时，如果设置为(S2)=2，则至排序结束为止的扫描次数为 45 ÷ 2=22.5 → 23(扫描)。

(4) (D1)中指定的软元件 (结束软元件) 在 SORT 指令执行开始时变为 OFF，在排序结束时变为 ON。
排序结束后，(D1)中指定的软元件将保持为 ON 状态不变，因此应由用户根据需要将其置为 OFF。

(5) 从(D2)中指定的软元件开始的 2 点为 SORT 指令执行时的系统所用。
用户不要对从(D2)中指定的软元件开始的 2 点进行变更。
如果进行了变更，有可能导致出错。 (出错代码： 4100)

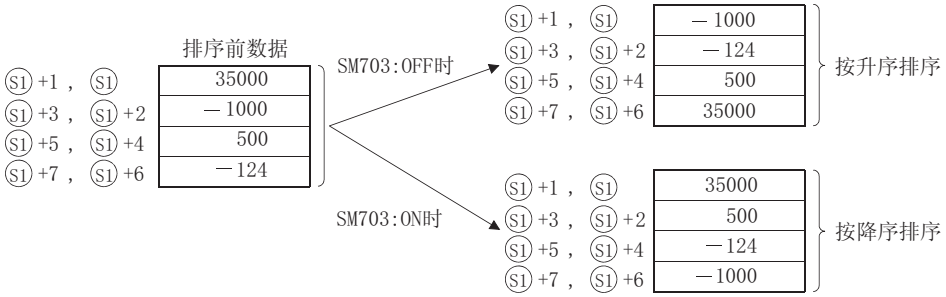
(6) 如果在排序执行过程中对 n 进行了变更，将按变更后的排序数据数进行排序。

(7) 如果在排序执行过程中将其执行指令置为 OFF，则排序将被中断。
再次将其执行指令置为 ON 时，将从头开始重新进行排序。

(8) 排序执行结束后，连续进行下次排序时，需要将执行指令置为一次 OFF 后，再次将执行指令置为 ON。

DSORT

(1) 将从(S1)开始的 n 点的 BIN32 位数据进行升序 / 降序排序 (排列替换)。
排序指定是通过 SM703 的 ON/OFF 进行的。
· SM703 为 OFF 时： 升序排序
· SM703 为 ON 时： 降序排序



(2) 通过 DSORT 指令进行排序时，需要数次扫描。
至执行结束为止的扫描次数为，将至排序执行结束为止的最多执行次数，与Ⓢ2中指定的 1 次执行中进行比较的数据数相除后的值。(小数点以下进位。)
如果Ⓢ2的值较大，则至排序结束为止的扫描次数将变少，但扫描时间将延长。

(3) 至排序执行结束为止的最多执行次数应通过下述公式算出：
至排序执行结束为止的最多执行次数 =(n) × (n-1) ÷ 2(次)

例 n=10 时，需要 10 × (10-1) ÷ 2=45(次)。
此时，如果设置为Ⓢ2=2，则至排序结束为止的扫描次数为 45 ÷ 2=22.5 → 23(扫描)。

(4) Ⓓ1中指定的软元件 (结束软元件) 在 DSORT 指令执行开始时变为 OFF，在排序结束时变为 ON。
排序结束后，Ⓓ1中指定的软元件将保持为 ON 状态不变，因此应由用户根据需要将其置为 OFF。

(5) 从Ⓓ2中指定的软元件开始的 2 点为 DSORT 指令执行时的系统所用。
用户不要对从Ⓓ2中指定的软元件开始的 2 点进行变更。
如果进行了变更，有可能导致出错。 (出错代码： 4100)

(6) 如果在排序执行过程中对 n 进行了变更，将按变更后的排序数据数进行排序。

(7) 如果在排序执行过程中将其执行指令置为 OFF，则排序将被中断。
再次将其执行指令置为 ON 时，将从头开始重新进行排序。

(8) 排序执行结束后，连续进行下次排序时，需要将执行指令置为一次 OFF 后，再次将执行指令置为 ON。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· SORT(P) 指令中从Ⓢ1开始的 n 点的范围超出了相应软元件的范围时。
(出错代码： 4101)

· DSORT(P) 指令中从Ⓢ1开始的 2 × n 点的范围超出了相应软元件的范围时。
(出错代码： 4101)

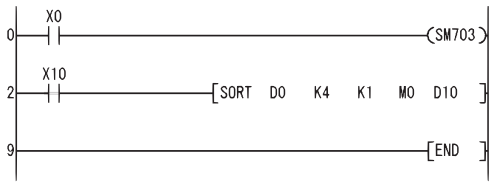
· 从Ⓢ1开始的 n 点 ÷ 2 × n 点的软元件范围与从Ⓓ2开始的 2 点的软元件范围重复时。
(出错代码： 4101)

· Ⓢ2为 0 或负数时。 (出错代码： 4100)

程序示例

(1) 以下为 X10 变为 ON 时，对从 D0 开始的 10 点的 BIN16 位数据进行升序 / 降序排序的程序。

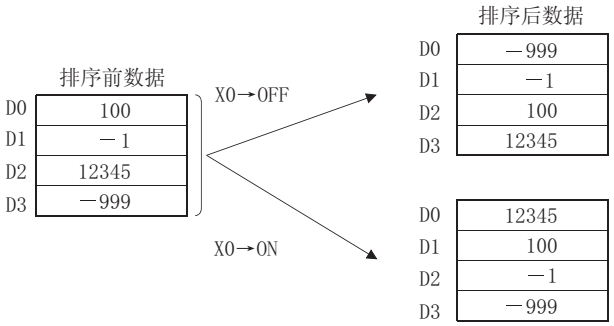
[梯形图模式]



[列表模式]

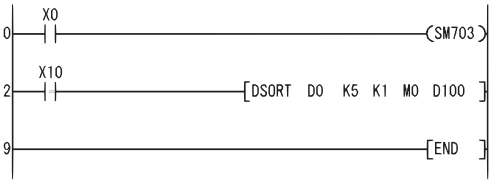
步	指令	软元件
0	LD	X0
1	OUT	SM703
2	LD	X10
3	SORT	D0 K4 K1 M0 D10
9	END	

[动作]



(2) 以下为 X10 变为 ON 时，对从 D0 开始的 20 点的 BIN32 位数据进行升序 / 降序排序的程序。

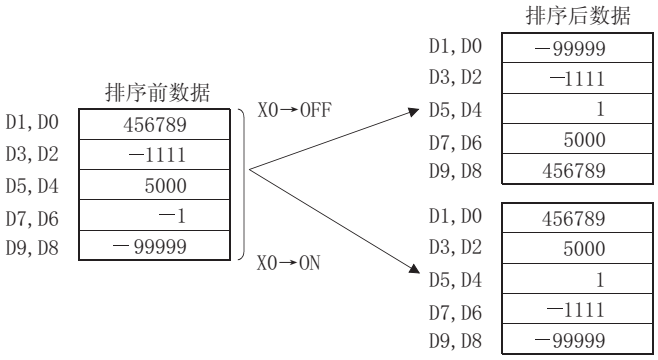
[梯形图模式]



[列表模式]

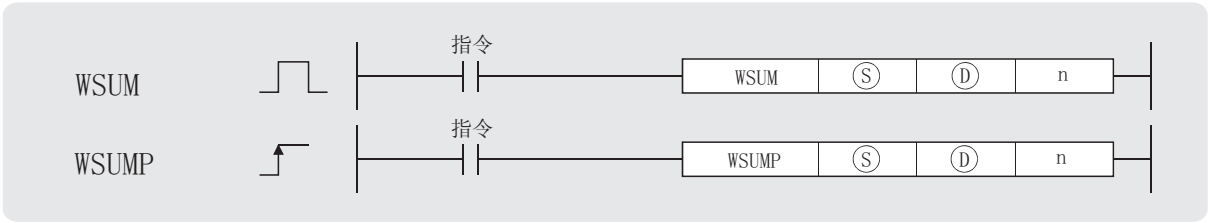
步	指令	软元件
0	LD	X0
1	OUT	SM703
2	LD	X10
3	DSORT	D0 K5 K1 M0 D100
9	END	

[动作]



7.5.13 16 位数据的合计值计算 (WSUM(P))

Basic | High performance | Process | Redundant | Universal

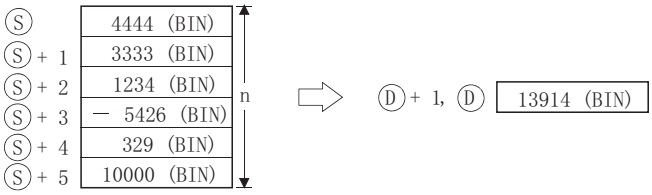


Ⓢ : 存储进行合计值计算数据的软元件的起始编号 (BIN16 位)。
Ⓓ : 存储合计值的软元件的起始编号 (BIN32 位)。
n : 数据个数 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	--	○		--				--	--
Ⓓ	○	○		○				--	--
n	○	○		○				○	--

功 能

将Ⓢ中指定的软元件开始的 n 点的 16 位 BIN 数据全部进行加法运算后，将结果存储到Ⓓ中指定的软元件中。



出 错

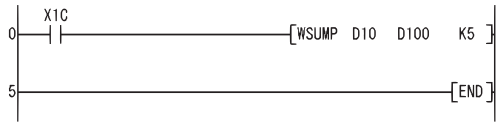
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· 从Ⓢ的软元件开始的 n 点的范围超出了相应软元件时。 (出错代码： 4101)

程序示例

(1) 以下为 X1C 变为 ON 时，对 D0 ～ D14 的 16 位 BIN 数据进行加法运算后，将结果存储到 D100、D101 中的程序。

[梯形图模式]



[列表模式]

步	指令	软元件		
0	LD	X1C		
1	WSUMP	D10	D100	K5
5	END			

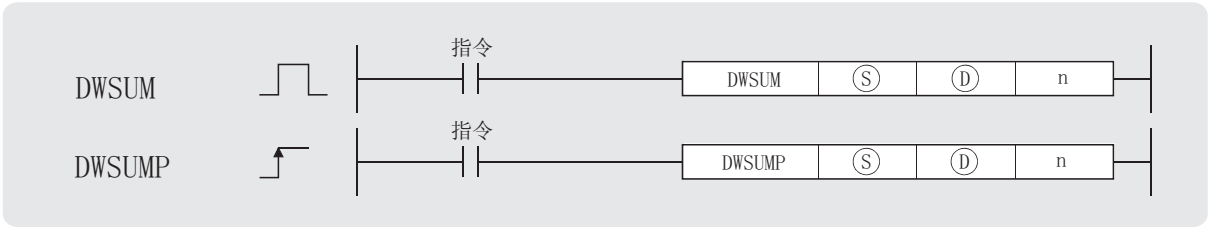
[动作]

D10	4500 (BIN)
D11	2500 (BIN)
D12	— 3276 (BIN)
D13	6780 (BIN)
D14	4444 (BIN)

⇨ D101, D100 | 14948 (BIN) |

7.5.14 32 位数据的合计值计算 (DWSUM(P))

Basic | High performance | Process | Redundant | Universal

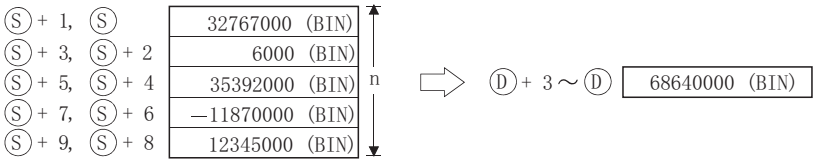


Ⓢ : 存储进行合计值计算数据的软元件的起始编号 (BIN32 位)。
Ⓓ : 存储合计值的软元件的起始编号 (BIN64 位)。
n : 数据个数 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	--	○		--				--	--
Ⓓ	○	○		--				--	--
n	○	○		○				○	--

功 能

将Ⓢ中指定的软元件开始的 n 点的 32 位 BIN 数据全部进行加法运算后，将结果存储到Ⓓ中指定的软元件开始的 4 点 (4 字) 中。



出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

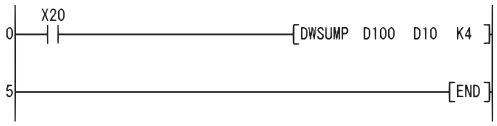
· 从Ⓢ的软元件开始的 n 点的范围超出了相应软元件时。 (出错代码： 4101)

· Ⓓ中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)
(出错代码： 4101)

程序示例

(1) 以下为 X20 变为 ON 时，对 D100 ～ D107 的 32 位 BIN 数据进行加法运算后，将结果存储到 D10、D13 中的程序。

[梯形图模式]



[列表模式]

步	指令	软元件		
0	LD	X20		
1	DWSUMP	D100	D10	K4
5	END			

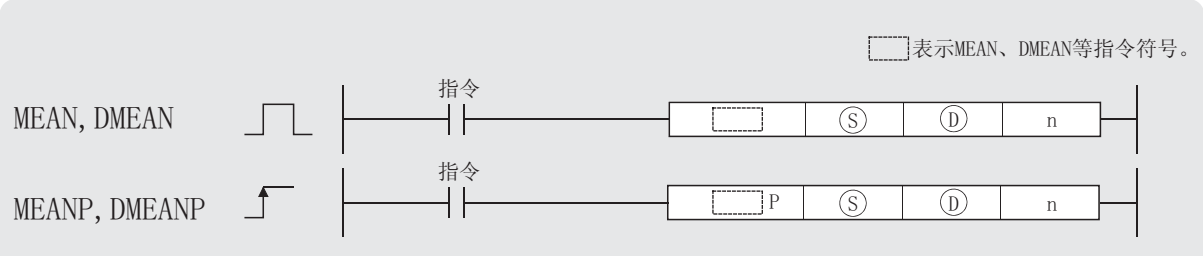
[动作]

D101, D100	11245600 (BIN)
D103, D102	27543200 (BIN)
D105, D104	558800 (BIN)
D107, D106	— 15675000 (BIN)

⇨ D13 ～D10 | 23672600 (BIN)

7.5.15 16 位 /32 位数据的平均值计算 (MEAN(P)、DMEAN(P))

· QnU(D)(H)CPU：序列号的前 5 位数为“ 10102 ”以后
· QnUDE(H)CPU：序列号的前 5 位数为“ 10102 ”以后



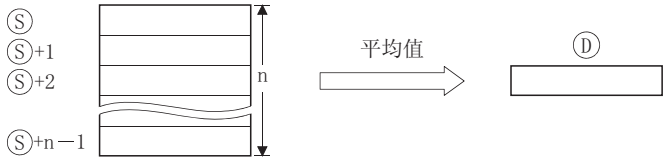
Ⓢ ：存储进行平均值计算数据的软元件的起始编号 (BIN16/32 位)。
Ⓓ ：存储平均值的软元件的起始编号 (BIN16/32 位)。
n ：数据个数或者存储数据个数的软元件编号 (设置范围为 1 ～ 32767)(BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	--	○	○	--				--	--
Ⓓ	--	○	○	--				--	--
n	--	○	○	○				○	--

功 能

MEAN(P)

(1) 将Ⓢ中指定的软元件开始的 n 点 (16 位 BIN 数据) 的平均值进行计算后，将结果存储到Ⓓ中指定的软元件中。

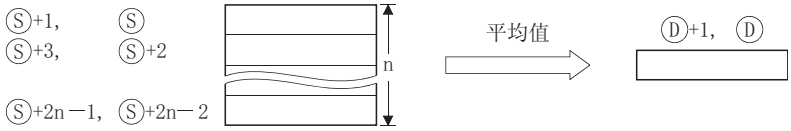


(2) 在计算结果不是整数值的情况下，小数点以下将进位。

(3) n 中指定的值为 0 时执行无处理。

DMEAN(P)

(1) 将Ⓢ中指定的软元件开始的 n 点 (32 位 BIN 数据) 的平均值进行计算后，将结果存储到Ⓓ中指定的软元件中。



(2) 在计算结果不是整数值的情况下，小数点以下将进位。

(3) n 中指定的值为 0 时执行无处理。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· n 中指定的值超出了 0 ～ 32767 的范围时。 (出错代码 : 4100)

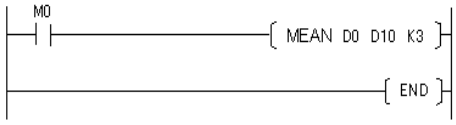
· 从Ⓢ中指定的软元件开始的 n 点的范围超出了指定软元件的范围时。

(出错代码 : 4101)

程序示例

(1) 以下为 M0 变为 ON 时，将 D0 ～ D2 的 16 位数据的平均值存储到 D10 中的程序。

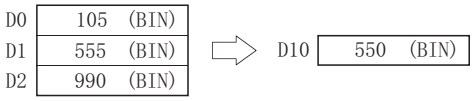
[梯形图模式]



[列表模式]

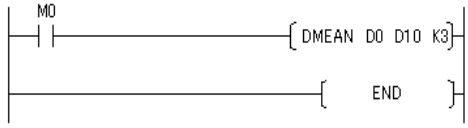
步	指令	软元件
0	LD	M0
1	MEAN	D0 D10 K3
5	END	

[动作]



(2) 以下为 M0 变为 ON 时，将 D0 ～ D5 的 32 位数据的平均值存储到 D10、D11 中的程序。

[梯形图模式]



[列表模式]

步	指令	软元件
0	LD	M0
1	DMEAN	D0 D10 K3
5	END	

[动作]

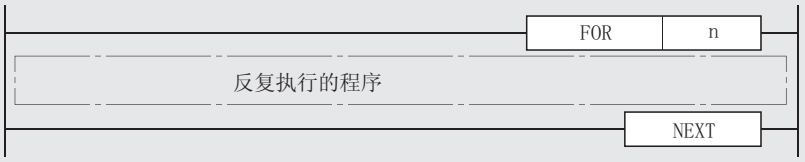
7.6 结构化指令

7.6.1 FOR ～ NEXT 指令循环 (FOR、NEXT)

Basic | High performance | Process | Redundant | Universal

FOR

NEXT



n ： FOR ～ NEXT 之间的反复次数。(1 ～ 32767)(BIN16 位)

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
n	○								--

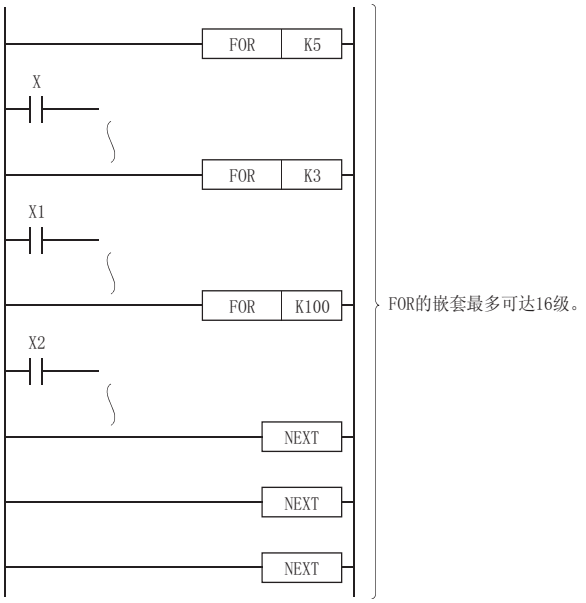
功 能

(1) 将 FOR ～ NEXT 指令之间的处理无条件地执行了 n 次时，执行 NNEXT 指令的下一步的处理。

(2) n 的可指定范围为 1 ～ 32767。如果指定了 -32768 ～ 0 的值，将执行等同于 n=1 时的处理。

(3) 不希望执行 FOR ～ NEXT 指令之间的处理时，应通过 CJ、SCJ 指令进行跳转。

(4) FOR 的嵌套最多可达 16 级。



出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

· 在 FOR 指令执行后，NEXT 指令执行前执行了 END、FEND、GOEND 指令时。 (出错代码： 4200)

· 在 FOR 指令执行前执行了 NEXT 指令时。 (出错代码： 4201)

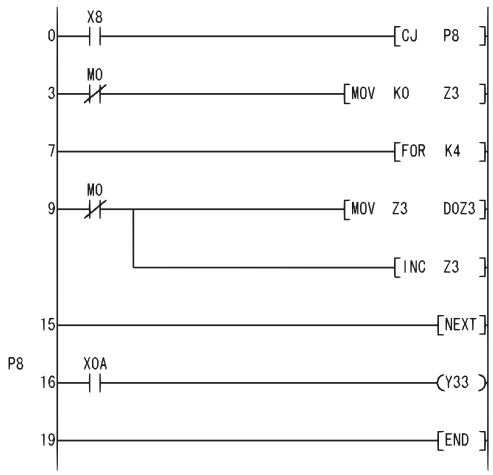
· 在 FOR ～ NEXT 指令之间存在有 STOP 指令时。 (出错代码： 4200)

· 在执行 FOR 指令的嵌套时，执行了 17 级以上的情况下。 (出错代码： 4202)

程序示例

(1) 以下为 X8 变为 OFF 时，执行 FOR ～ NEXT 指令，X8 变为 ON 时，不执行 FOR ～ NEXT 指令的程序。

[梯形图模式]

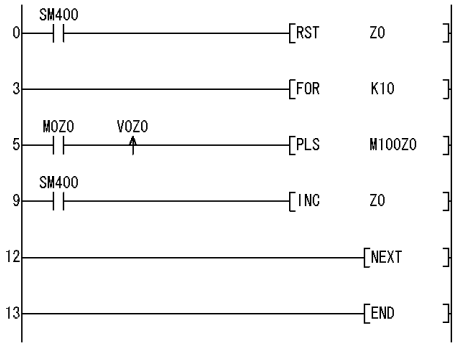


[列表模式]

步	指令	软元件	
0	LD	X8	
1	CJ	P8	
3	LDI	M0	
4	MOV	K0	Z3
7	FOR	K4	
9	LDI	M0	
10	MOV	Z3	DOZ3
13	INC	Z3	
15	NEXT		
16	P8		
17	LD	X0A	
18	OUT	Y33	
19	END		

备 注

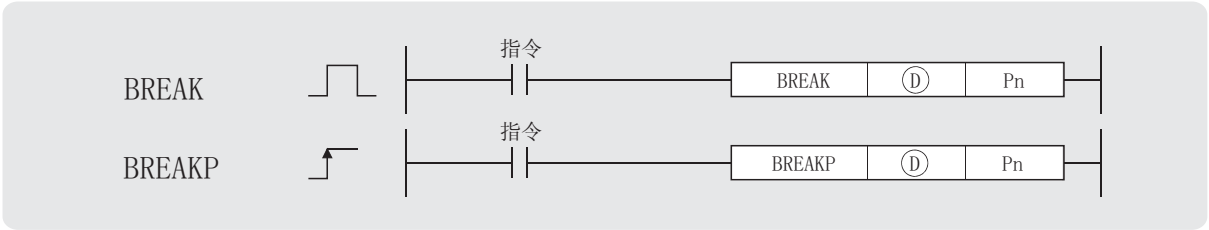
1. 在 FOR ～ NEXT 的反复执行过程中希望结束执行的情况下，应使用 BREAK 指令。
   关于 BREAK 指令的详细内容，请参阅 7.6.2 项。
2. 在 FOR ～ NEXT 之间执行变址修饰程序的脉冲运算时，应使用 EGP/EGF 指令。
   关于 EGP/EGF 指令的详细内容，请参阅 5.2.5 项。其样本程序如下所示：



3. 不能从 FOR ～ NEXT 的外面对 FOR ～ NEXT 指令内通过 JMP 指令等进行分支。

7.6.2 FOR ～ NEXT 指令循环的强制结束 (BREAK(P))

Basic | High performance | Process | Redundant | Universal

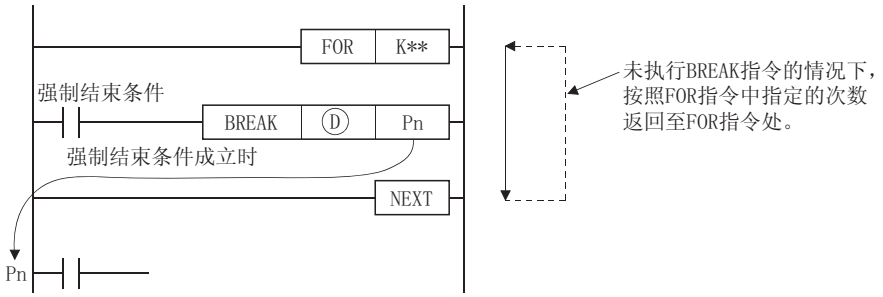


Ⓓ ：存储剩余反复次数的软元件编号 (BIN16 位)。
Pn ：反复处理强制结束时的分支目标指针号 (软元件名 (指针))。

设置数据	内部软元件 位	内部软元件 字	R、ZR	J□\□ 位	J□\□ 字	U□\G□	Zn	常数	其它 P
Ⓓ	○							--	--
Pn	--							--	○

功 能

(1) 对通过 FOR ～ NEXT 指令进行的反复处理执行强制结束，将执行切换到 Pn 中指定的指针处。
在 Pn 中只能指定同一个程序文件内的指针。
如果在 Pn 中指定了其它程序文件内的指针，将发生运算出错。



(2) 强制结束时，将 FOR ～ NEXT 指令中剩余的反复处理执行次数存储到Ⓓ中。
但是，执行 BREAK 指令时的次数也包括在剩余的反复处理次数中。

(3) BREAK 指令只能在 FOR ～ NEXT 指令之间才可使用。

(4) BREAK 指令只能用于 1 个嵌套。
对多重嵌套执行强制结束时，则应执行与嵌套级数相同数量的 BREAK 指令。

出 错

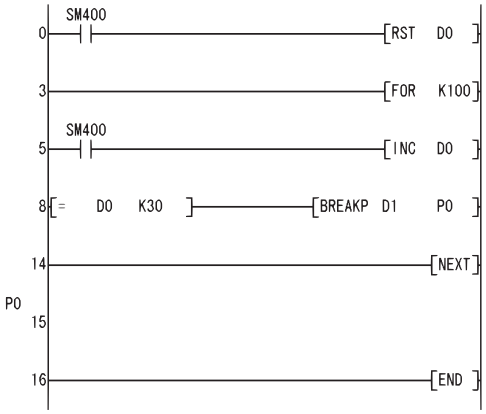
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- 用于除 FOR ～ NEXT 指令以外时。 (出错代码： 4203)
- Pn 中指定的指针的跳转目标不存在时。 (出错代码： 4210)
- Pn 中指定了其它程序文件的指针时。 (出错代码： 4210)

程序示例

(1) 以下为 D0 变为 30 时 (执行了 30 次 FOR ～ NEXT 时)，强制结束 FOR ～ NEXT 的程序。

[梯形图模式]



[列表模式]

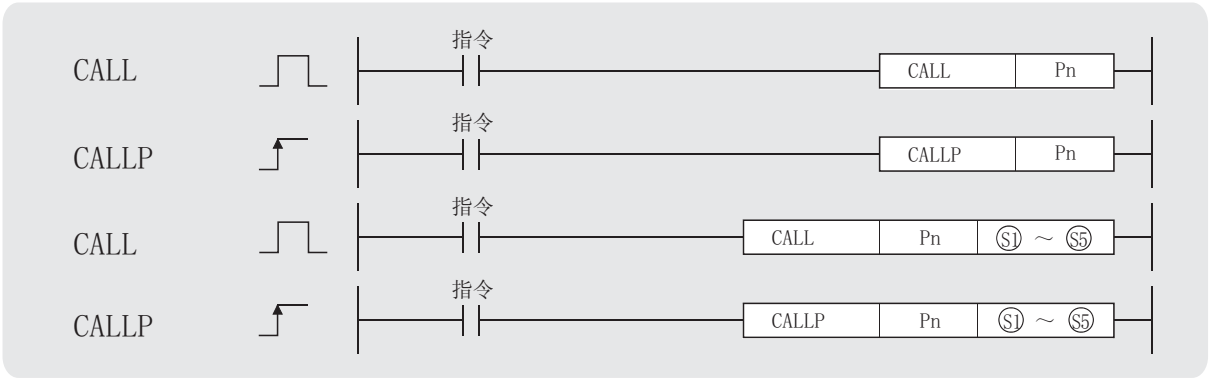
步	指令	软元件	
0	LD	SM400	
1	RST	D0	
3	FOR	K100	
5	LD	SM400	
6	INC	D0	
8	LD=	D0	K30
11	BREAKP	D1	P0
14	NEXT		
15	P0		
16	END		

备 注

执行 BREAK 指令时，D1 中将存储 71。

7.6.3 子程序调用 (CALL(P))

Basic | High performance | Process | Redundant | Universal



Pn ： 子程序的起始指针编号 (软元件名)。

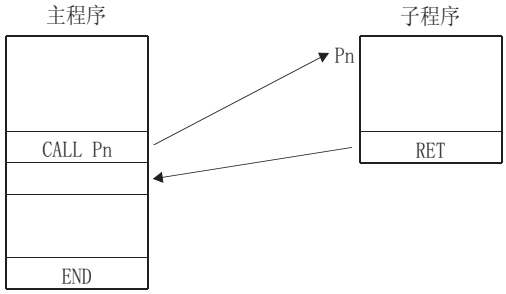
(S1) ～ (S5) ： 作为变量传送到子程序中的软元件编号 (位、BIN16 位、BIN32 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它 P
	位	字		位	字				
Pn	--	--		--					○
(S1)～(S5)	○ (除 F 以外)	○		○					--

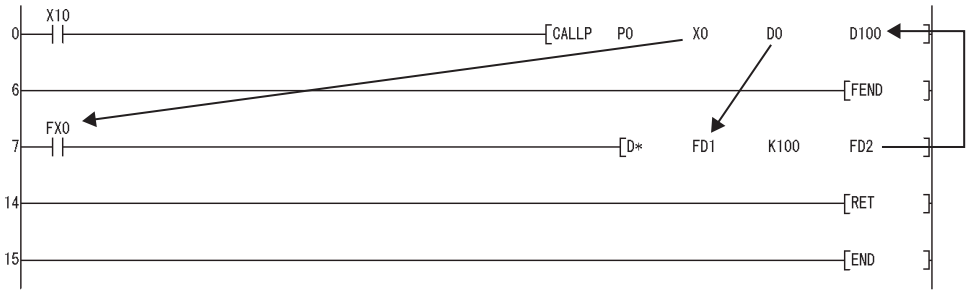
功 能

(1) 如果执行 CALL(P) 指令，将执行 Pn 中指定的指针的子程序。

[CALL(P) 指令可以执行由同一个程序文件中的指针指定的子程序及由公共指针指定的子程序。]



(2) 在子程序中使用功能软元件 (FX、FY、FD) 时，在Ⓢ1～Ⓢ5中指定与功能软元件对应的软元件。
Ⓢ1～Ⓢ5中指定的软元件的内容如下所示。



(a) 在子程序执行之前，将位数据的内容传送到 FX 中，将字数据的内容传送到 FD 中。

(b) 在子程序执行之后，将 FY、FD 的内容传送到对应的软元件中。

(c) 功能软元件的处理单位如下所示。

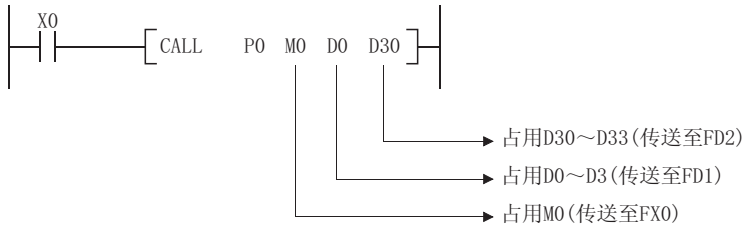
· FX、FY：位单位

· FD：4 字单位
根据变量中指定的软元件的种类，所使用的数据大小有所不同。
功能软元件中指定的软元件应预留出相当于该数据大小的容量。
未能预留出相当于该数据大小的容量时将会变为出错状态。

功能软元件	使用软元件	数据大小	备注
· FX · FY	位软元件	1 点	---
	字软元件的位指定时	1 位	
· FD	位软元件的位数指定时 *1	4 字	数据大小根据所使用的指令而有所不同。
	字软元件	4 字	

*1:位软元件的位数指定时，即使在Ⓢ1～Ⓢ5中指定的软元件号不是 16 的倍数时，也不会变为出错状态。

[主程序]



(3) 在 CALL(P) 指令中，可使用范围为Ⓢ1～Ⓢ5。

(4) 子程序中使用的功能软元件数必须与 CALL(P) 指令中的变量数相同。
此外，功能软元件与 CALL(P) 的变量类型应该完全一致。

(5) CALL(P) 指令的变量中指定的软元件号不应重复。
如果重复，有可能导致无法正常运算。

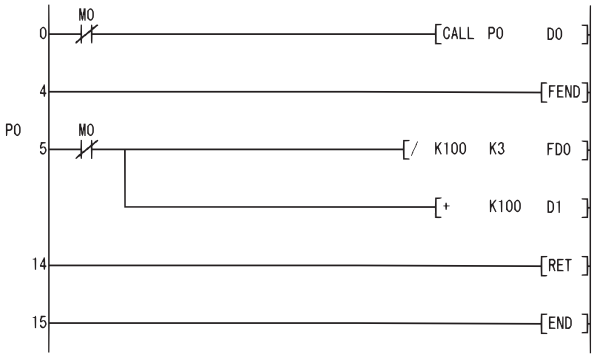
(6) CALL(P) 指令的变量中使用的软元件不应在子程序中使用。
如果在子程序中使用了 CALL(P) 指令的变量中使用的软元件，将无法正常进行运算。(参阅以下程序示例。)

(7) CALL(P) 指令的变量的软元件中使用了定时器 / 计数器时，只进行当前值的接收发送。

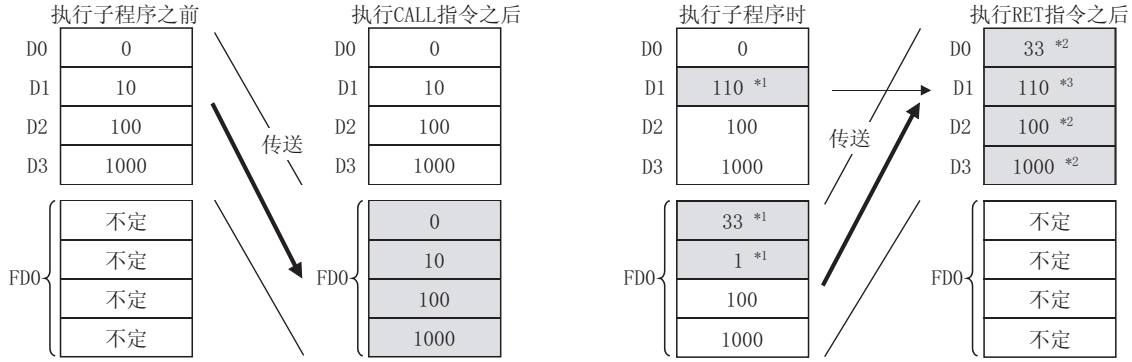
不正常的运算示例

将 D0 指定到子程序的 FD0 中，且在子程序中使用了 D1 时的动作如下所示。

[程序示例]



[执行了子程序后的动作]



*1: 存储子程序的执行结果。

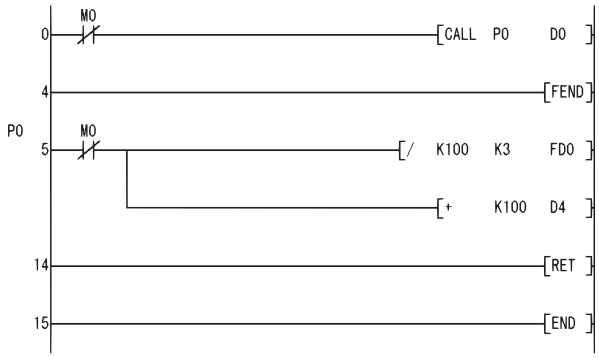
*2: 替换为功能软元件的值。

*3: D1 中不能反映功能软元件的值。

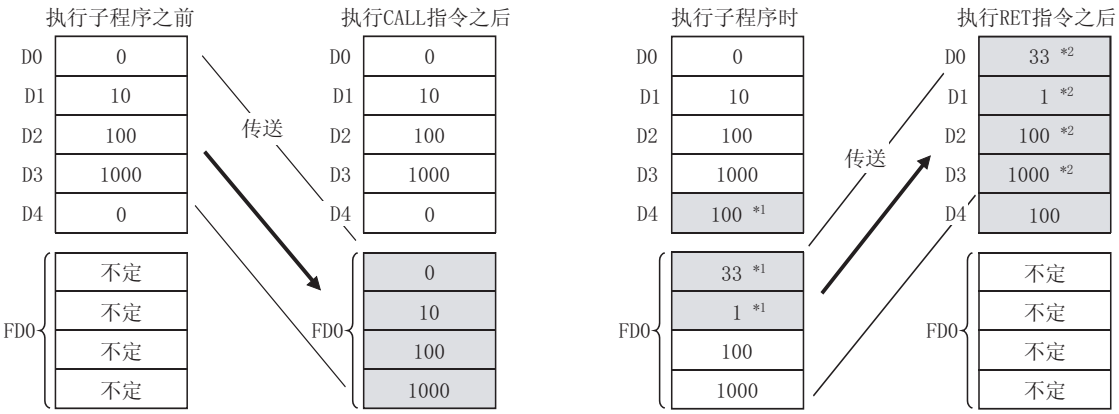
正常的运算示例

将 D0 指定到子程序的 FD0 中，且在子程序中使用了 D4 时的动作如下所示。

[程序示例]



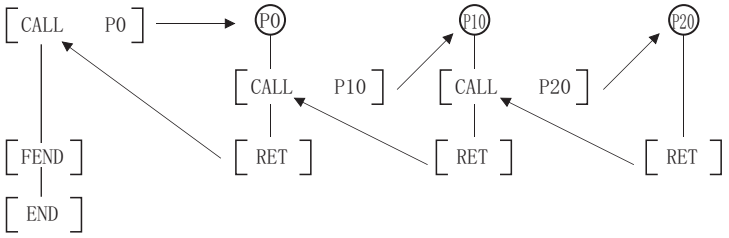
[执行了子程序后的动作]



*1：存储子程序的执行结果。

*2：替换为功能软元件的值。

(8) CALL(P) 指令最多可以有 16 级嵌套。
然而，该 16 级是 CALL(P)、FCALL(P)、ECALL(P)、EFCALL(P)、XCALL 指令中嵌套的总级数。



(9) 对于在子程序内使其变为 ON 的软元件，即使子程序停止后其 ON 状态也仍将被保持。
对于在执行子程序时变为 ON 的软元件，可以通过 FCALL(P) 指令使其变为 OFF。

出 错

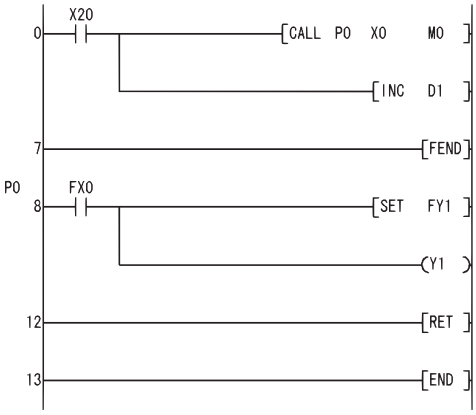
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- 变量中指定的软元件未能预留出相当于数据大小的容量时。 (出错代码： 4101)
- 在 CALL(P) 指令执行后，RET 指令执行前执行了 END、FEND、GOEND、STOP 指令时 (出错代码： 4211)
- 在 CALL(P) 指令执行前执行了 RET 指令时。 (出错代码： 4212)
- 执行了第 17 级嵌套时。 (出错代码： 4213)
- CALL(P) 指令中指定的指针在子程序中不存在时。 (出错代码： 4210)

程序示例

(1) 以下为 X20 变为 ON 时，执行带变量子程序的程序。

[梯形图模式]



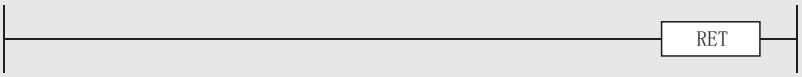
[列表模式]

步	指令	软元件
0	LD	X20
1	CALL	P0 X0 M0
5	INC	D1
7	FEND	
8	P0	
9	LD	FX0
10	SET	FY1
11	OUT	Y1
12	RET	
13	END	

7.6.4 从子程序返回 (RET)

Basic | High performance | Process | Redundant | Universal

RET

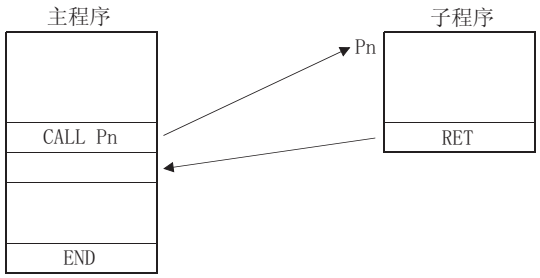


设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
--	--								

功 能

(1) 表示子程序的结束。

(2) 执行了 RET 指令时，调用了子程序的 CALL(P)、FCALL(P)、ECALL(P)、EFCALL(P)、XCALL 指令将返回至下一步。



出 错

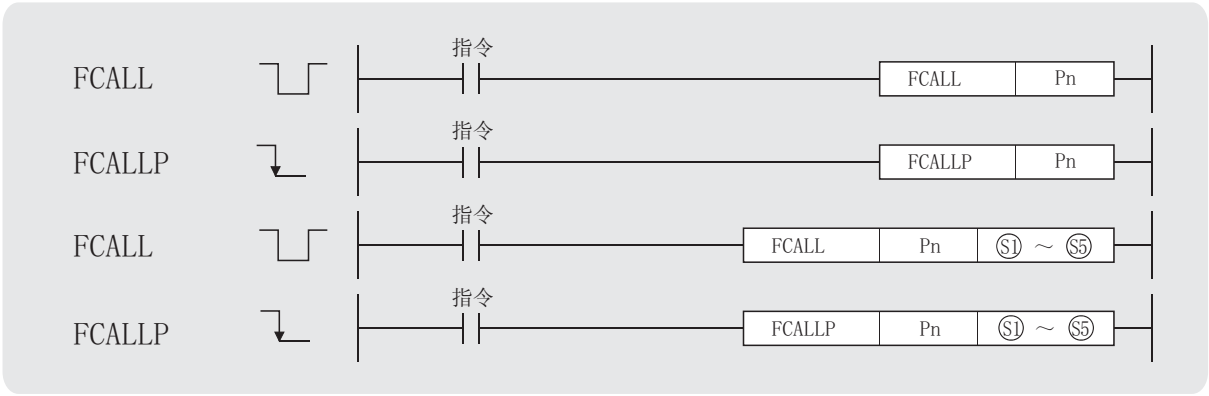
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 执行 CALL(P)、FCALL(P)、ECALL(P)、EFCALL(P)、XCALL 指令后，在执行 RET 指令之前执行了 END、FEND、GOEND、STOP 指令时。 (出错代码： 4211)

· 在执行 CALLL(P)、FCALL(P)、ECALL(P)、EFCALL(P)、XCALL 指令之前执行了 RET 指令时。 (出错代码： 4212)

7.6.5 子程序输出 OFF 调用 (FCALL(P))

Basic | High performance | Process | Redundant | Universal



Pn ：子程序的起始指针编号（软元件名）。

ⓢ1 ～ ⓢ5 ：作为变量传送到子程序中的软元件编号（位、BIN16 位、BIN32 位）。

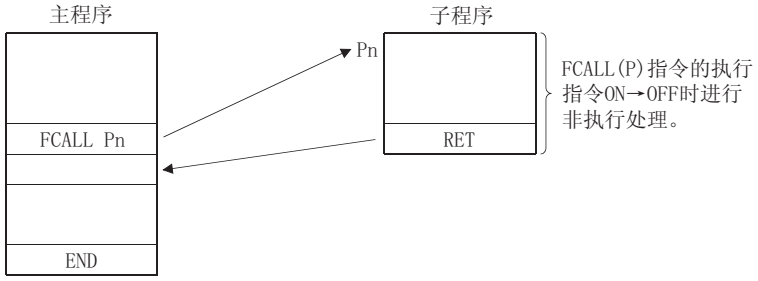
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				P
Pn	--	--		--					○
ⓢ1～ⓢ5	○（除 F 以外）	○		○					--

功 能

(1) 如果执行 FCALL(P) 指令，将执行 Pn 中指定的指针的子程序的非执行处理。

[FCALL(P) 指令可以执行由同一个程序文件中的指针指定的子程序及由公共指针指定的子程序。]

(a) 非执行处理是指，对各线圈指令执行与条件设置处于 OFF 状态时相同的处理。

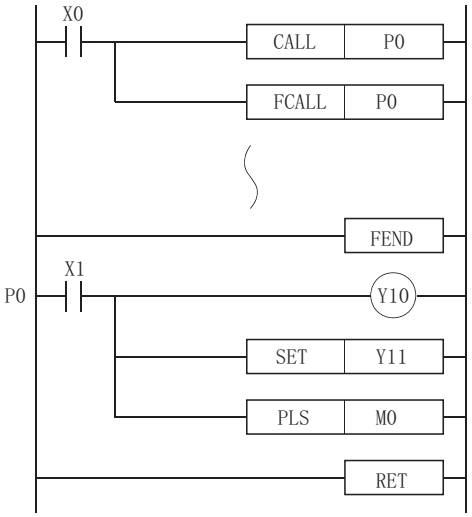


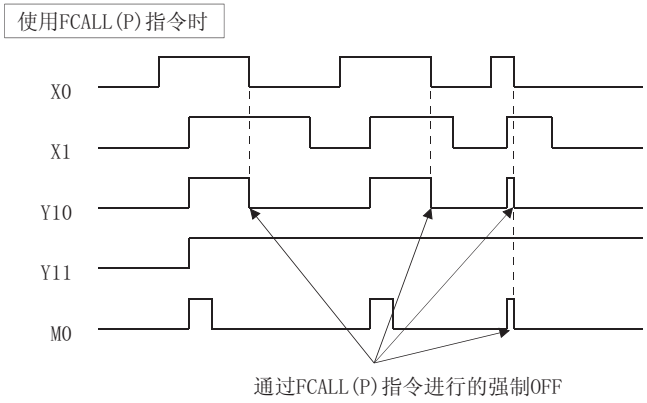
(b) 非执行处理后的各线圈指令的运算结果与条件触点的ON/OFF状态无关，其情况如下所示：

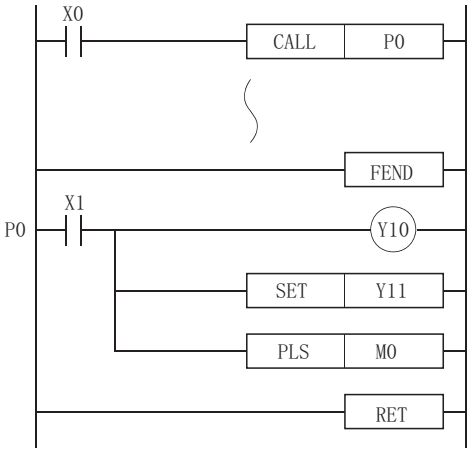
OUT 指令..... 强制 OFF
SET 指令
RST 指令
SFT 指令
基本指令
应用指令 ... 状态保持
PLS 指令
脉冲化指令 (□ P) ... 与条件触点 OFF 时相同的处理
低速 / 高速定时器的当前值 0
累计定时器的当前值
计数器的当前值 ... 保持

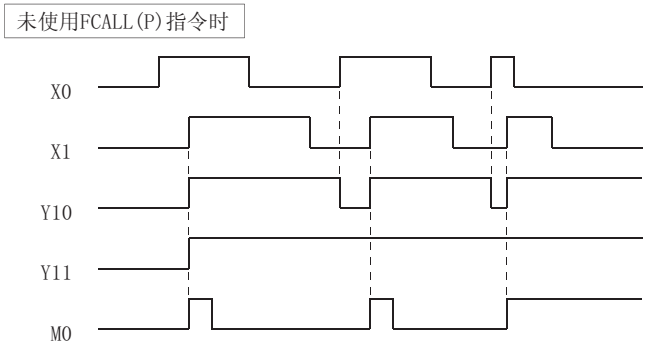
(2) FCALL(P) 指令是与 CALL(P) 指令组合使用。

(3) 将 FCALL(P) 指令与 CALL(P) 指令组合执行时，如果执行指令处于 OFF 状态则进行子程序的非执行处理，因此可以对 OUT 指令、PLS 指令 (包括 □ P 指令) 进行强制 OFF。
未将 FCALL(P) 指令与 CALL(P) 指令组合执行时，即使执行指令处于 OFF 状态也不进行子程序的非执行处理，因此各线圈指令的输出状态将被保持。

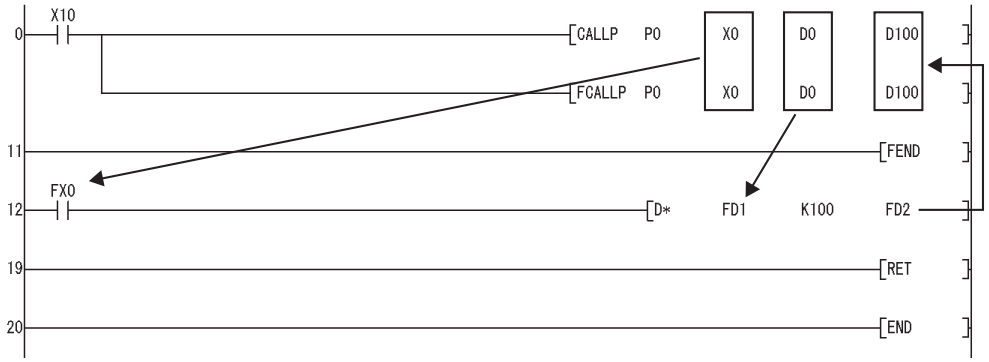








(4) 在子程序中使用功能软元件 (FX、FY、FD) 时，在Ⓢ1 ～Ⓢ5 中指定与功能软元件对应的软元件。
在Ⓢ1 ～Ⓢ5 中指定的软元件的内容如下所示。



(a) 在子程序执行之前，将位数据的内容传送到 FX 中，将字数据的内容传送到 FD 中。

(b) 在子程序执行之后，将 FY、FD 的内容传送到对应的软元件中。

(c) 功能软元件的处理单位如下所示。

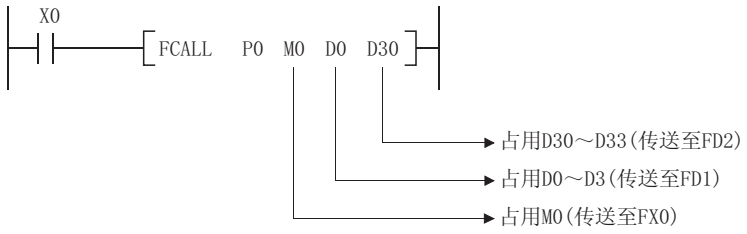
· FX、FY：位单位

· FD：4 字单位
根据变量中指定的软元件的种类，所使用的数据大小有所不同。
功能软元件中指定的软元件应预留出相当于该数据大小的容量。
未能预留出相当于该数据大小的容量时将会变为出错状态。

功能软元件	使用软元件	数据大小	备注
· FX · FY	位软元件	1 点	----
	字软元件的位指定时	1 位	
· FD	位软元件的位数指定时 *1	4 字	FD 的高位 2 字将变为 0。
	字软元件	4 字	----

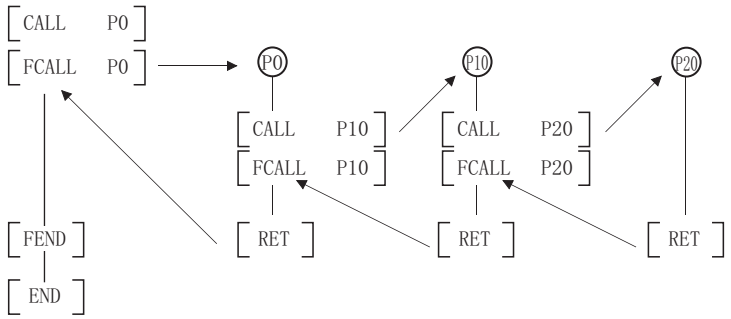
*1：位软元件的位数指定时，即使在Ⓢ1 ～Ⓢ5 中指定的软元件号不是 16 的倍数时，也不会变为出错状态。

[主程序]



(5) 在 FCALL(P) 指令中，可使用范围为Ⓢ1 ～Ⓢ5。

(6) FCALL(P) 指令最多可以有 16 级嵌套。
但是，该 16 级是 CALL(P)、FCALL(P)、ECALL(P)、EFCALL(P)、XCALL 指令中嵌套的总级数。



出 错

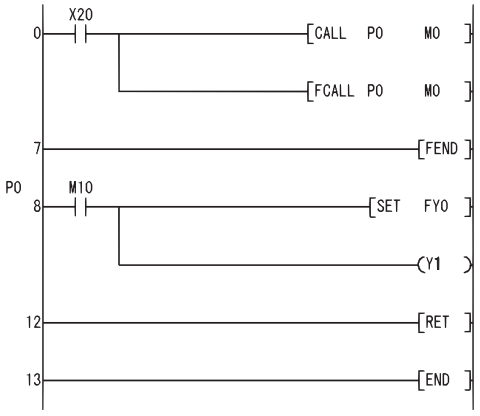
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- 变量中指定的软元件未能预留出相当于数据大小的容量时。 (出错代码： 4101)
- 在 FCALL(P) 指令执行后，RET 指令执行前执行了 END、FEND、GOEND、STOP 指令时。 (出错代码： 4211)
- 在 FCALL(P) 指令执行前执行了 RET 指令时。 (出错代码： 4212)
- 执行了第 17 级嵌套时。 (出错代码： 4213)
- FCALL(P) 指令中指定的指针在子程序中不存在时。 (出错代码： 4210)

程序示例

(1) 以下为 X20 变为 ON 时，执行带变量子程序，X20 由 ON → OFF 时进行强制非执行处理的程序。

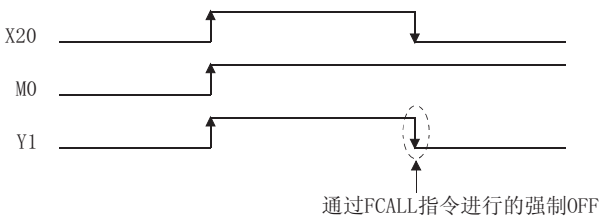
[梯形图模式]



[列表模式]

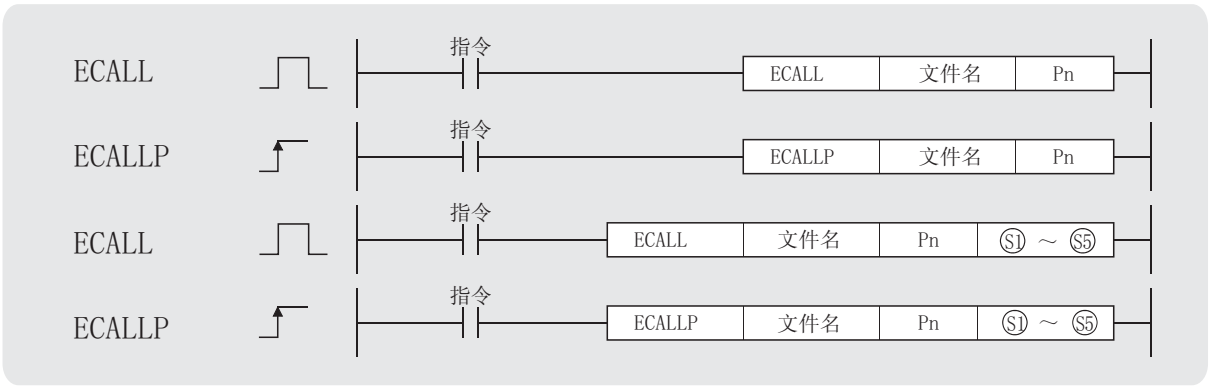
步	指令	软元件	
0	LD	X20	
1	CALL	P0	M0
4	FCALL	P0	M0
7	FEND		
8	P0		
9	LD	M10	
10	SET	FY0	
11	OUT	Y1	
12	RET		
13	END		

[动作]



7.6.6 程序文件之间的子程序调用 (ECALL(P))

Basic | High performance | Process | Redundant | Universal

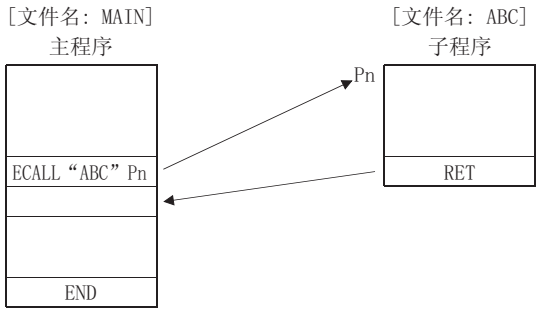


文件名 ：调用的文件名（字符串）。
Pn ：子程序的起始指针编号（软元件名）。
ⓢ1 ～ ⓢ5 ：作为变量传送到子程序中的软元件编号（位、BIN16 位、BIN32 位）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数		其它
	位	字		位	字			K、H	$	P
文件名	--	○		--					○	--
Pn	--	--		--					--	○
ⓢ1～ⓢ5	○（除 F 以外）	○		○					--	--

功 能

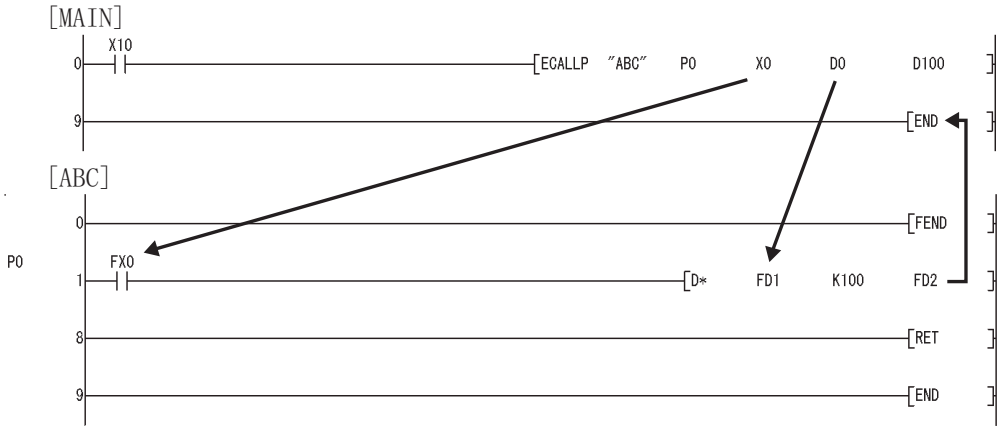
(1) 如果执行 ECALL(P) 指令，将执行指定程序文件名的 Pn 中指定的指针的子程序。
使用 ECALL(P) 指令可从其它程序文件中调用使用局部指针的子程序。



(2) 文件名只能指定驱动器 0(程序存储器 / 内置 RAM) 中存储的程序文件。

(3) 文件名中无需指定扩展名 (.QPG)。
(仅以 .QPG 的文件为对象。)

(4) 在子程序中使用功能软元件 (FX、FY、FD) 时，在Ⓢ1～Ⓢ5 中指定与功能软元件对应的软元件。
在Ⓢ1～Ⓢ5 中指定的软元件的内容如下所示。



(a) 在子程序执行之前，将位数据的内容传送到 FX 中，将字数据的内容传送到 FD 中。

(b) 在子程序执行之后，将 FY、FD 的内容传送到对应的软元件中。

(c) 功能软元件的处理单位如下所示。

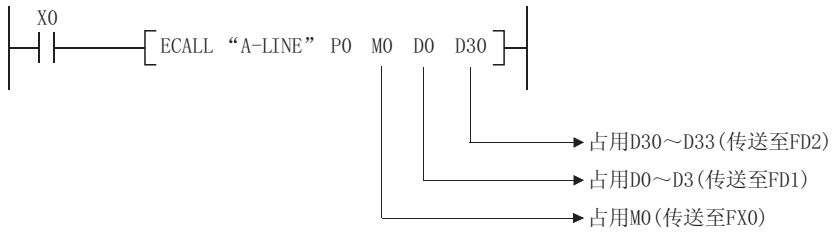
· FX、FY：位单位

· FD：4 字单位
根据变量中指定的软元件的种类，所使用的数据大小有所不同。
功能软元件中指定的软元件应预留出相当于该数据大小的容量。
未能预留出相当于该数据大小的容量时将会变为出错状态。

功能软元件	使用软元件	数据大小	备注
· FX · FY	位软元件	1 点	----
	字软元件的位指定时	1 位	
· FD	位软元件的位数指定时 *1	4 字	数据大小根据所使用的指令而有所不同。
	字软元件	4 字	

*1：位软元件的位数指定时，即使在Ⓢ1～Ⓢ5 中指定的软元件号不是 16 的倍数时，也不会变为出错状态。

[主程序]



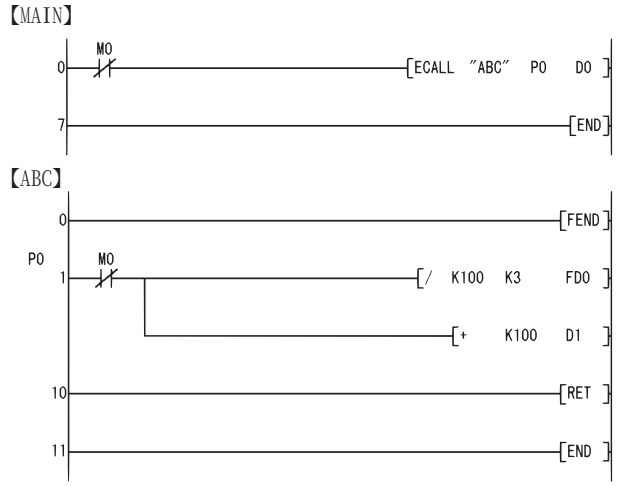
(5) 在 ECALL(P) 指令中，可使用范围为ⓢ1～ⓢ5。

(6) ECALL(P) 指令的变量中使用的软元件不应在子程序中使用。
如果在子程序中使用了 ECALL(P) 指令的变量中使用的软元件，将无法正常进行运算。(参阅以下程序示例。)

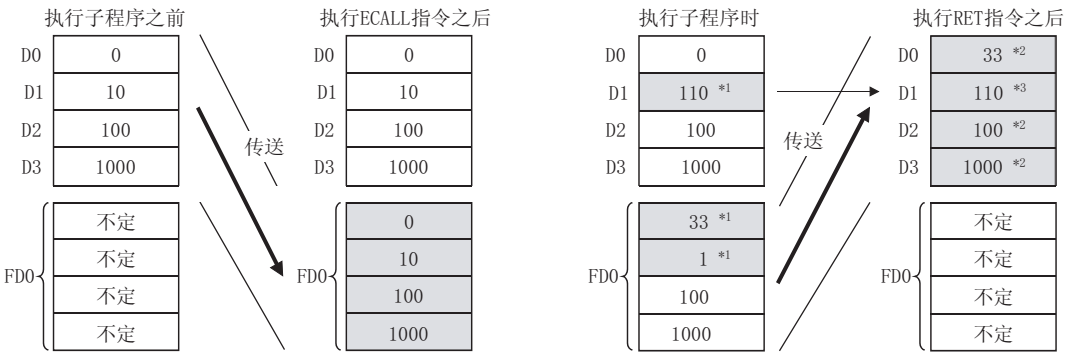
不正常的运算示例

将 D0 指定到子程序的 FD0 中，且在子程序中使用了 D1 时的动作如下所示。

[程序示例]



[执行了子程序后的动作]



*1: 存储子程序的执行结果。

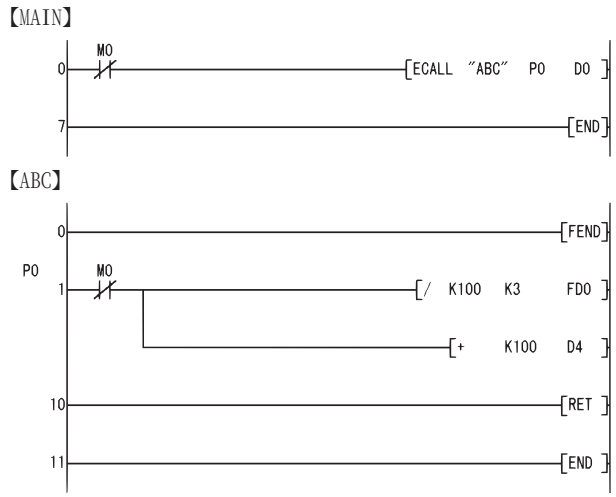
*2: 替换为功能软元件的值。

*3: D1 中不能反映功能软元件的值。

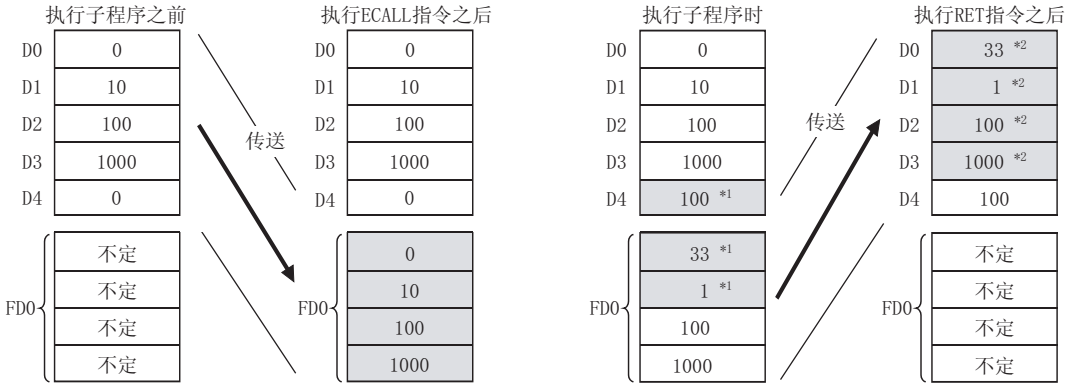
正常的运算示例

将 D0 指定到子程序的 FD0 中，且在子程序中使用了 D4 时的动作如下所示。

[程序示例]



[执行了子程序后的动作]

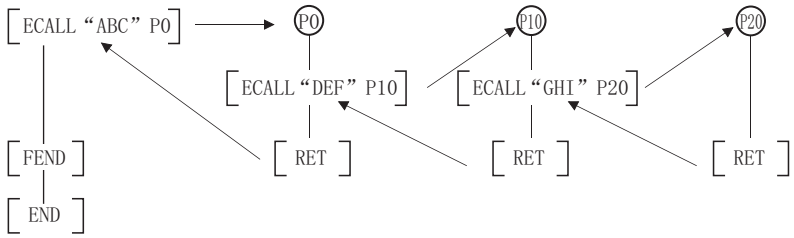


*1：存储子程序的执行结果。

*2：替换为功能软元件的值。

(7) ECALL(P) 指令的变量中指定的软元件号不应重复。
如果重复，有可能导致无法正常运算。

(8) ECALL(P) 指令最多可以有 16 级嵌套。
但是，该 16 级是 CALL(P)、FCALL(P)、ECALL(P)、EFCALL(P)、XCALL 指令中嵌套的总级数。



(9) 对于在子程序内使其变为 ON 的软元件，即使子程序非执行时其 ON 状态也仍将被保持。
对于在执行子程序时变为 ON 的软元件，可以通过 ECALL(P) 指令使其变为 OFF。

出 错

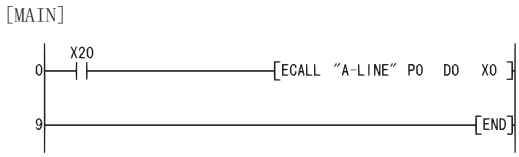
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- 变量中指定的软元件未能预留出相当于数据大小的容量时。 (出错代码： 4101)
- 在 ECALL(P) 指令执行后，RET 指令执行前执行了 END、FEND、GOEND、STOP 指令时。 (出错代码： 4211)
- 在 ECALL(P) 指令执行前执行了 RET 指令时。 (出错代码： 4212)
- 执行了第 17 级嵌套时。 (出错代码： 4213)
- ECALL(P) 指令中指定的指针在子程序中不存在时。 (出错代码： 4210)
- 指定的文件不存在时。 (出错代码： 2410)
- 指定的文件无法执行时。 (出错代码： 2411)

程序示例

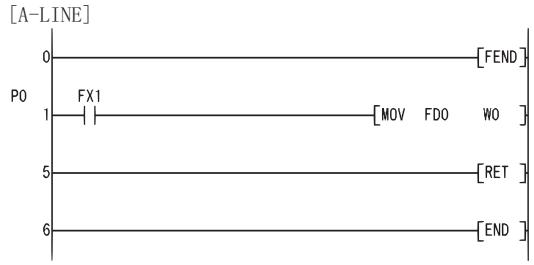
(1) 以下为 X20 变为 ON 时，执行程序名为 A-LINE 的 P0 的程序。

[梯形图模式]



[列表模式]

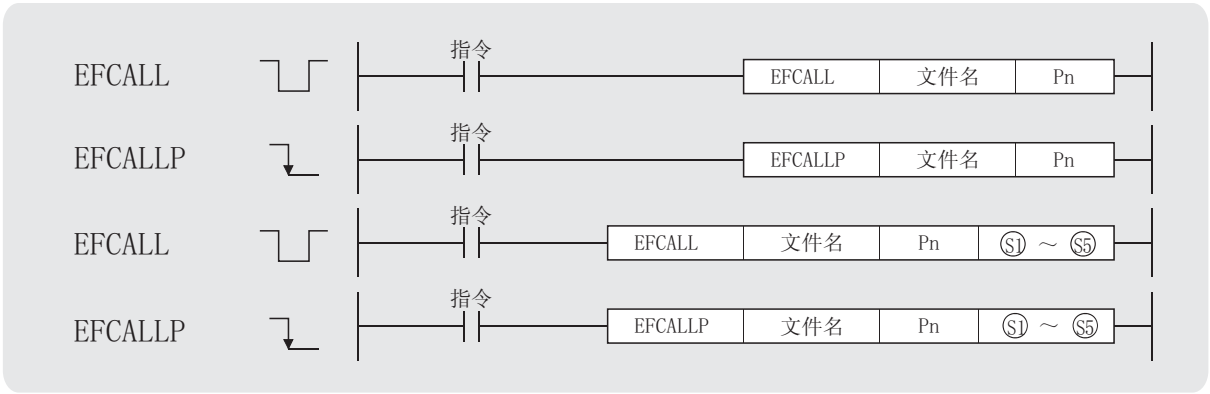
步	指令	软元件
0	LD	X20
1	ECALL	"A-LINE" P0 D0 X0
9	END	



步	指令	软元件
0	FEND	
1	P0	
2	LD	FX1
3	MOV	FD0 W0
5	RET	
6	END	

7.6.7 程序文件之间的子程序输出 OFF 调用 (EFCALL(P))

Basic High performance Process Redundant Universal



文件名 ：调用的文件名（字符串）。
Pn ：子程序的起始指针编号（软元件名）。
(S1) ～ (S5) ：作为变量传送到子程序中的软元件编号（位、BIN16 位、BIN32 位）。

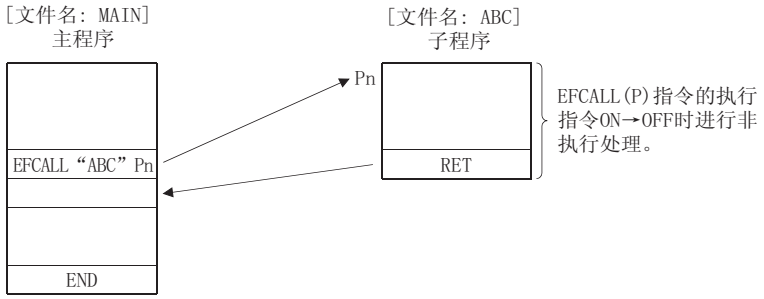
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数		其它
	位	字		位	字			K、H	$	P
文件名	--	○		--					○	--
Pn	--	--		--					--	○
(S1)～(S5)	○（除 F 以外）	○		○					--	--

功 能

(1) 如果执行 EFCALL(P) 指令，将执行指定 Pn 中指定的指针的子程序的非执行处理。

[使用 EFCALL(P) 指令可从其它程序文件中调用使用局部指针的子程序。]

(a) 非执行处理是指，对各线圈指令执行与条件设置处于 OFF 状态时相同的处理。

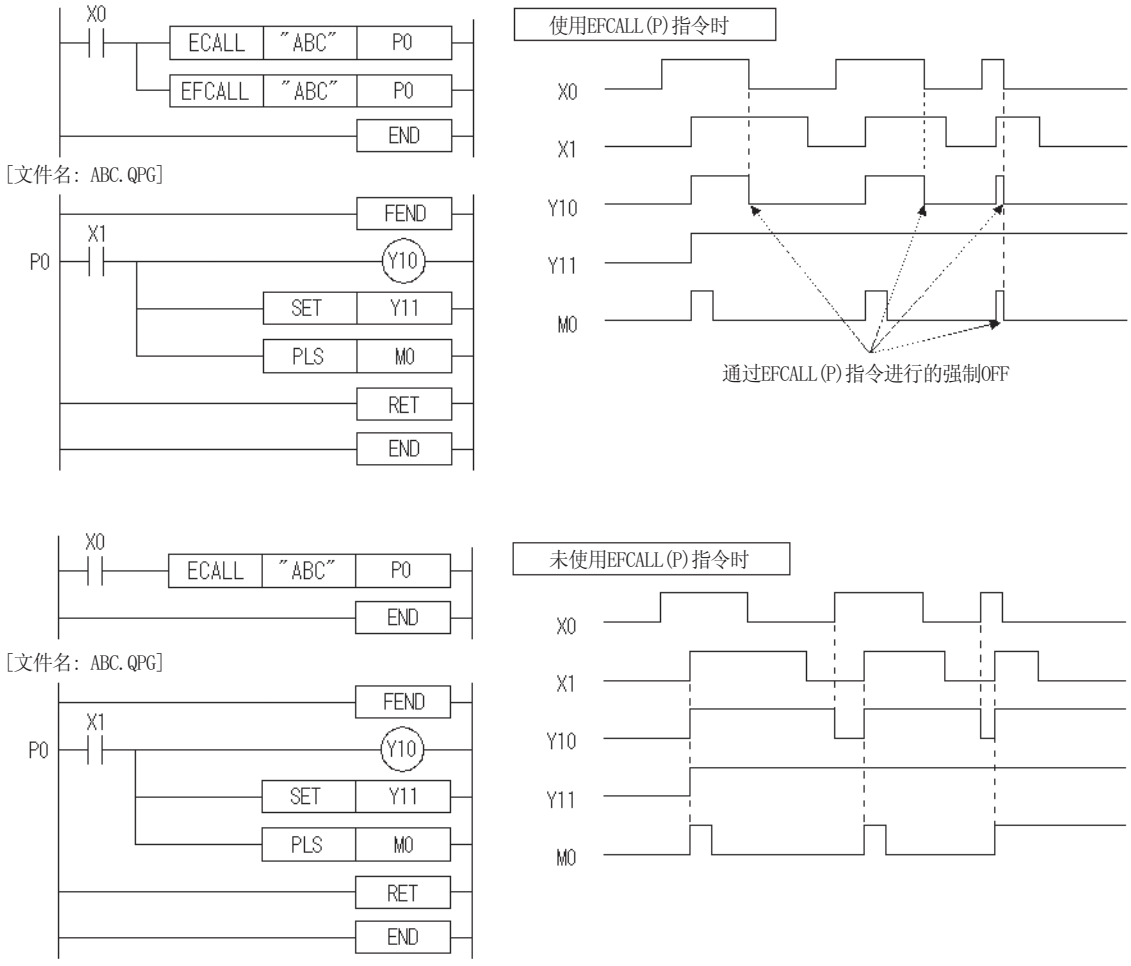


(b) 非执行处理后的各线圈指令的运算结果与条件触点的 ON/OFF 状态无关，其情况如下所示：

OUT 指令 ..... 强制 OFF
SET 指令
RST 指令
SFT 指令
基本指令
应用指令 .... 状态保持
PLS 指令
脉冲化指令 (P) .... 与条件触点 OFF 时相同的处理
低速 / 高速定时器的当前值 0
累计定时器的当前值
计数器的当前值 .... 保持

(2) EFCALL(P) 指令是与 ECALL(P) 指令组合使用。

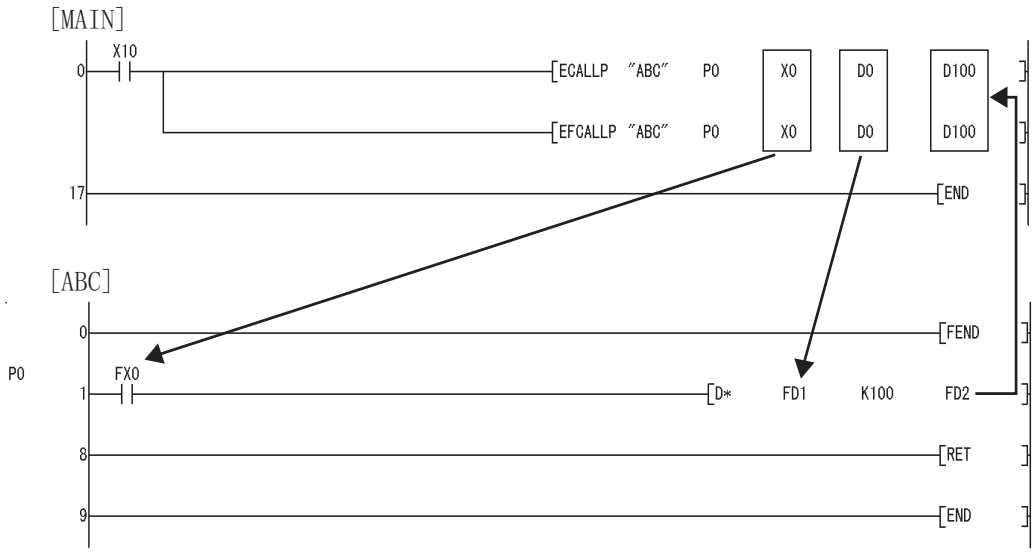
(3) 将 EFCALL(P) 指令与 ECALL(P) 指令组合执行时，如果执行指令处于 OFF 状态则进行子程序的非执行处理，因此可以对 OUT 指令、PLS 指令 (包括 P 指令) 进行强制 OFF。
未将 EFCALL(P) 指令与 ECALL(P) 指令组合执行时，即使执行指令处于 OFF 状态也不进行子程序的非执行处理，因此各线圈指令的输出状态将被保持。



(4) 文件名只能指定驱动器 0(程序存储器 / 内置 RAM) 中存储的程序文件。

(5) 文件名中无需指定扩展名 (.QPG)。
(仅以 .QPG 的文件为对象。)

(6) 在子程序中使用功能软元件 (FX、FY、FD) 时，在Ⓢ1～Ⓢ5 中指定与功能软元件对应的软元件。



(a) 在子程序执行之前，将位数据的内容传送到 FX 中，将字数据的内容传送到 FD 中。

(b) 在子程序执行之后，将 FY、FD 的内容传送到对应的软元件中。

(c) 功能软元件的处理单位如下所示。

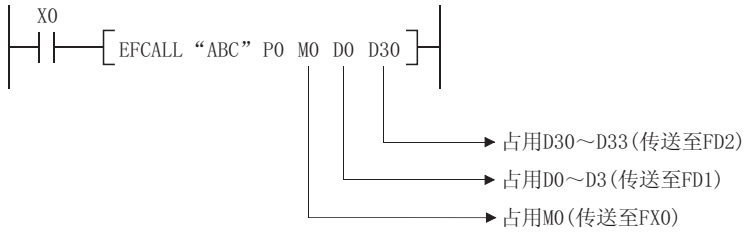
· FX、FY：位单位

· FD：4 字单位
根据变量中指定的软元件的种类，所使用的数据大小有所不同。
功能软元件中指定的软元件应预留出相当于该数据大小的容量。
未能预留出相当于该数据大小的容量时将会变为出错状态。

功能软元件	使用软元件	数据大小	备注
· FX · FY	位软元件	1 点	----
	字软元件的位指定时	1 位	
· FD	位软元件的位数指定时 *1	4 字	FD 的高位 2 字将变为 0。
	字软元件	4 字	----

*1：位软元件的位数指定时，即使在Ⓢ1～Ⓢ5 中指定的软元件号不是 16 的倍数时，也不会变为出错状态。

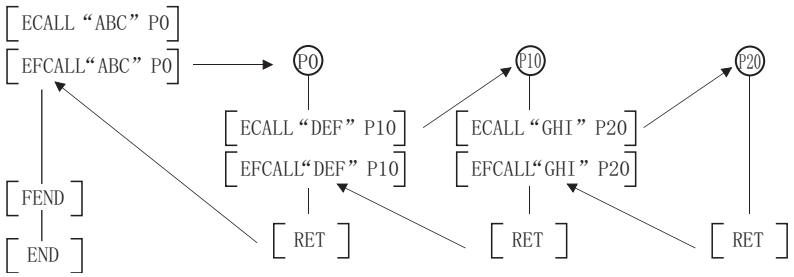
[主程序]



(7) 在 EFCALL(P) 指令中，可使用范围为Ⓢ1～Ⓢ5。

(8) 子程序中使用的功能软元件数必须与 EFCALL(P) 指令中的变量数相同。
此外，功能软元件与 EFCALL(P) 的变量类型应该完全一致。

(9) EFCALL(P) 指令最多可以有 16 级嵌套。
但是，该 16 级是 CALL(P)、FCALL(P)、ECALL(P)、EFCALL(P)、XCALL 指令中嵌套的总级数。



出 错

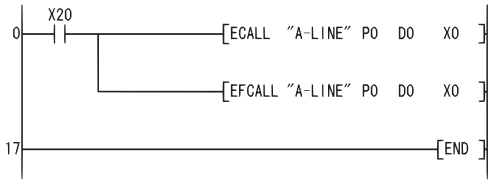
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- 变量中指定的软元件未能预留出相当于数据大小的容量时。 (出错代码 : 4101)
- 在 EFCALL(P) 指令执行后，RET 指令执行前执行了 END、FEND、GOEND、STOP 指令时。 (出错代码 : 4211)
- 在 EFCALL(P) 指令执行前执行了 RET 指令时。 (出错代码 : 4212)
- 执行了第 17 级嵌套时。 (出错代码 : 4213)
- EFCALL(P) 指令中指定的指针在子程序中不存在时。 (出错代码 : 4210)
- 指定的文件不存在时。 (出错代码 : 4210)
- 指定的文件无法执行时。 (出错代码 : 2411)

程序示例

(1) 以下为 X0 变为 ON 时，执行带变量子程序，X20 由 ON → OFF 时进行强制非执行处理的程序。

[梯形图模式]



[列表模式]

步	指令	软元件		
0	LD	X20		
1	ECALL	"A-LINE" P0	D0	X0
9	EFCALL	"A-LINE" P0	D0	X0
17	END			

7.6.8 子程序调用 (XCALL)

Ver. Basic | High performance | Process | Redundant | Universal

基本型 QCPU：序列号的前 5 位数为“ 04122 ”以后

Pn ：子程序的起始指针编号（软元件名）。

S1 ～ S5 ：作为变量传送到子程序中的软元件编号（位、BIN16 位、BIN32 位）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它 P
	位	字		位	字				
P	--	--		--					○
S1～S5	○（除 F 以外）	○		○					--

功 能

(1) XCALL 指令是进行子程序的执行和非执行处理的指令。

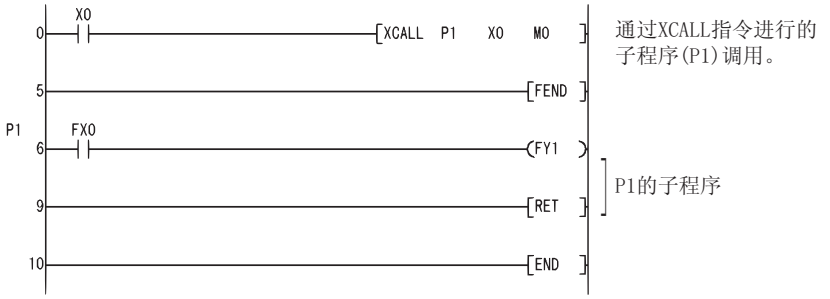
(a) 子程序的执行
在运算过程中根据条件触点的 ON/OFF 状态执行各线圈指令。

(b) 子程序的非执行处理
对各线圈指令执行与条件触点为 OFF 状态时相同的处理。
非执行处理之后的各线圈指令的运算结果与条件触点的 ON/OFF 状态无关，其情况如下所示。

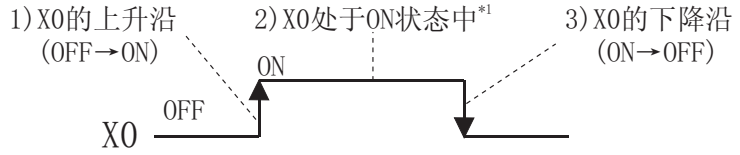
OUT 指令 ..... 强制 OFF
SET 指令
RST 指令
SFT 指令
基本指令
应用指令 ... 状态保持
PLS 指令
脉冲化指令 (□P) ... 与条件触点 OFF 时相同的处理
低速 / 高速定时器的当前值..... 0
累计定时器的当前值
计数器的当前值 ... 保持

(2) XCALL 指令的动作根据 CPU 模块类型而有所不同。
各 CPU 模块的 XCALL 指令的动作如下图的程序示例所示。

[程序示例]



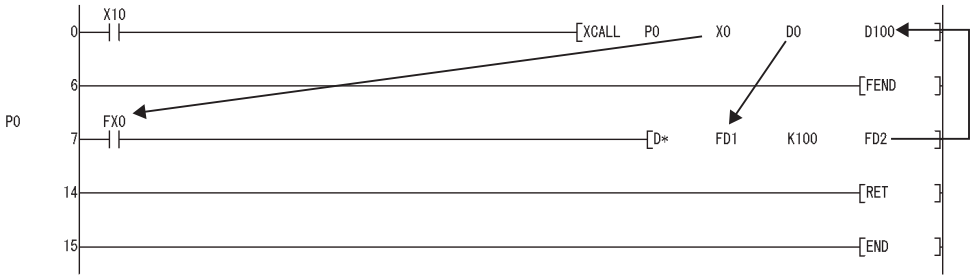
[X0 的 ON/OFF 时机]



*1: X0 处于 ON 状态中 (2)) 不包含 X0 的上升沿 (1))。

CPU 类型	XCALL 指令的动作
· 过程 CPU (序列号的前 5 位数： 07031 或以前) · 高性能型 QCPU (序列号的前 5 位数： 06081 或以前)	1)X0 的上升沿时 ： 无处理 (不执行 “ P1 ” 的子程序。) 2)X0 处于 ON 状态中 ： 执行 “ P1 ” 的子程序。 3)X0 的下降沿时 ： 执行 “ P1 ” 子程序的非执行处理。
· 高性能型 QCPU (序列号的前 5 位数： 06082 或以后。) · 过程 CPU (序列号的前 5 位数： 07032 或以后。)	1) 通过 SM734(XCALL 指令执行条件指定) 选择 X0 的上升沿时的动作。 · SM734 为 OFF 时 ： 无处理 (不执行 “ P1 ” 的子程序。) · SM734 为 ON 时 ： 执行 “ P1 ” 的子程序。 2)X0 处于 ON 状态中 ： 执行 “ P1 ” 的子程序。 3)X0 的下降沿时 ： 执行 “ P1 ” 子程序的非执行处理。
· 冗余 CPU · 基本型 QCPU · 通用型 QCPU	1)X0 的上升沿时 ： 执行 “ P1 ” 的子程序。 2)X0 处于 ON 状态中 ： 执行 “ P1 ” 的子程序。 3)X0 的下降沿时 ： 执行 “ P1 ” 子程序的非执行处理。

(3) 在子程序中使用功能软元件 (FX、FY、FD) 时，在Ⓢ1～Ⓢ5 中指定与功能软元件对应的软元件。
在Ⓢ1～Ⓢ5 中指定的软元件的内容如下所示。



(a) 在子程序执行之前，将位数据的内容传送到 FX 中，将字数据的内容传送到 FD 中。

(b) 在子程序执行之后，将 FY、FD 的内容传送到对应的软元件中。

(c) 功能软元件的处理单位如下所示。

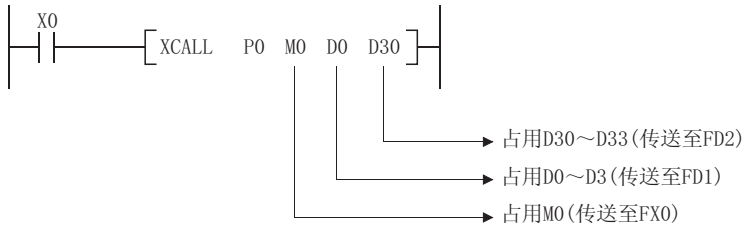
· FX、FY：位单位

· FD：4 字单位
根据变量中指定的软元件的种类，所使用的数据大小有所不同。
功能软元件中指定的软元件应预留出相当于该数据大小的容量。
未能预留出相当于该数据大小的容量时将会变为出错状态。

功能软元件	使用软元件	数据大小	备注
· FX · FY	位软元件	1 点	----
	字软元件的位指定时	1 位	
· FD	位软元件的位数指定时 *2	4 字	数据大小根据所使用的指令而有所不同。
	字软元件	4 字	

*2：位软元件的位数指定时，即使在Ⓢ1～Ⓢ5 中指定的软元件号不是 16 的倍数时，也不会变为出错状态。

[主程序]

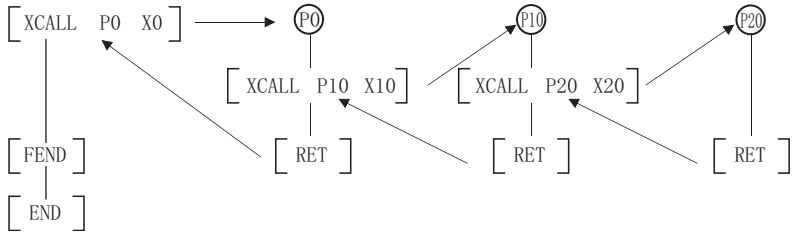


(4) 在 XCALL 指令中，可使用范围为Ⓢ1～Ⓢ5。

(5) 子程序中使用的功能软元件数必须与 XCALL 指令中的变量数相同。
此外，功能软元件与 XCALL 的变量类型应该完全一致。

(6) XCALL 指令的变量中指定的软元件号不应重复。
如果重复，有可能导致无法正常运算。

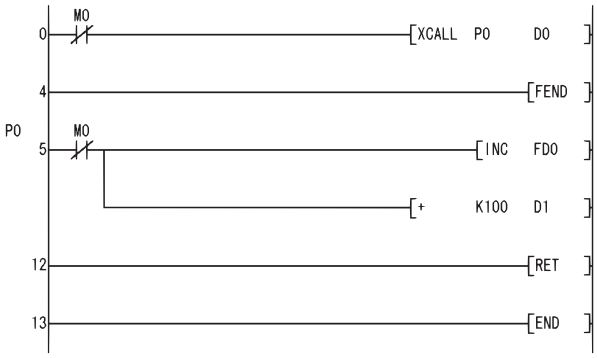
(7) XCALL 指令最多可以有 16 级嵌套。
但是，该 16 级是 CALL(P)、FCALL(P)、ECALL(P)、EFCALL(P)、XCALL 指令中嵌套的总级数。



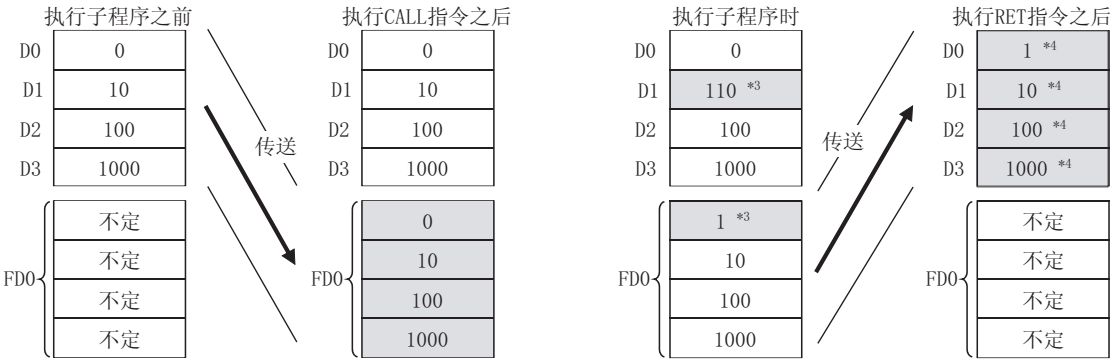
(8) XCALL 指令的变量中使用的软元件不应在子程序中使用。
如果在子程序中使用了 XCALL 指令的变量中使用的软元件，将无法正常进行运算。(参阅以下程序示例。)

将 D0 指定到子程序的 FD0 中，且在子程序中使用了 D1 时的动作如下所示。

[程序示例]



[执行了子程序后的动作]



*3: 存储子程序的执行结果。

*4: 替换为功能软元件的值。
D1 中不能反映子程序中的运算结果。

出 错

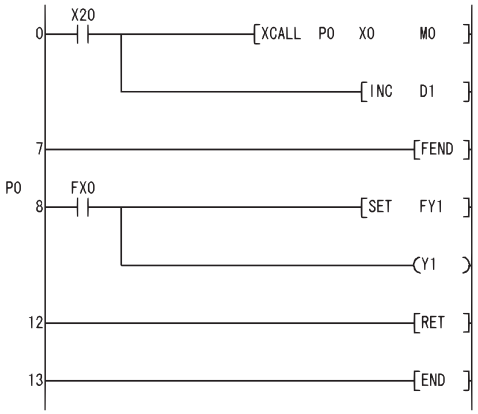
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- 变量中指定的软元件未能预留出相当于数据大小的容量时。 (出错代码： 4101)
- 在 XCALL(P) 指令执行后，RET 指令执行前执行了 END、FEND、GOEND、STOP 指令时。 (出错代码： 4211)
- 在 XCALL(P) 指令执行前执行了 RET 指令时。 (出错代码： 4212)
- 执行了第 17 级嵌套时。 (出错代码： 4213)
- 在 XCALL(P) 指令中指定的指针在子程序中不存在时。 (出错代码： 4210)

程序示例

(1) 以下为 X20 变为 ON 时，执行带变量子程序的程序。

[梯形图模式]



[列表模式]

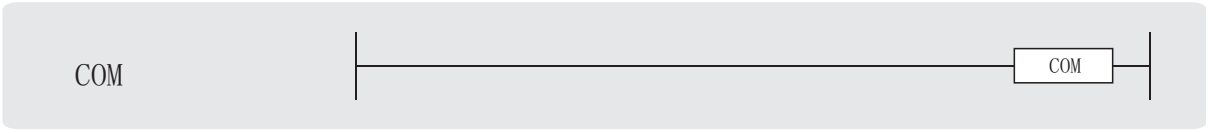
步	指令	软元件
0	LD	X20
1	XCALL	P0 X0 M0
5	INC	D1
7	FEND	
8	P0	
9	LD	FX0
10	SET	FY1
11	OUT	Y1
12	RET	
13	END	

7.6.9 刷新指令 (COM)

Basic | High performance | Process | Redundant | Universal

以下 CPU 模块的 COM 指令请参阅 9.13 节。

- 序列号为 04122 或以后的基本型 QCPU
- 序列号为 04012 或以后的高性能型 QCPU
- 序列号为 07032 或以后的过程 CPU
- 冗余 CPU
- 通用型 QCPU



设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
--	--								

功 能

(1) COM 指令用于以下场合：

(a) 希望提高与远程 I/O 站的发送 / 接收处理速度时。

(b) 在数据链接执行过程中，希望与使用不同扫描时间的其它站之间进行可靠的数据发送 / 接收时。

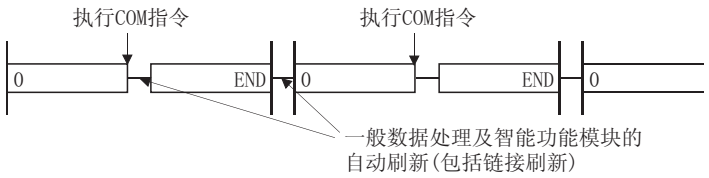
(2) 根据特殊继电器 SM775 的 ON/OFF 状态，COM 指令的处理有所不同。

- SM775 为 OFF 时： 进行自动刷新及与外围设备的通信。 *1*2
- SM775 为 ON 时： 仅进行与外围设备的通信。 *1

*1: 在与外围设备的通信中也进行以下处理。
- 其它站监视处理。
- 通过串行通信模块对其它的智能功能模块的缓冲存储器进行读取处理。

*2: 在自动刷新处理中进行以下处理。
- MELSECNET/10、MELSECNET/H 的刷新
- CC-Link 的刷新
- 智能功能模块的自动刷新

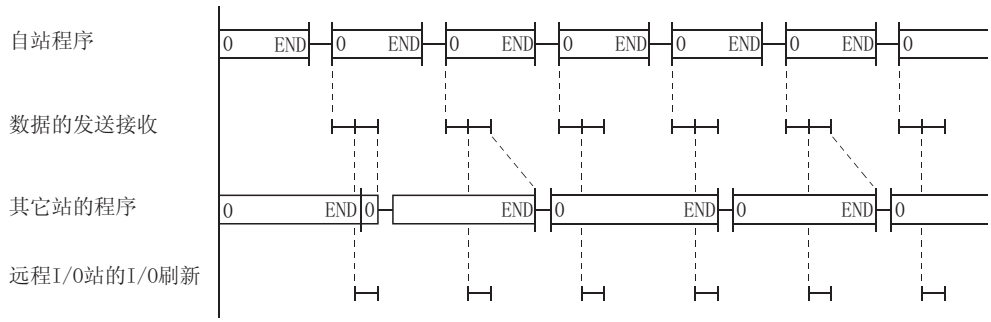
(3) 在执行了 COM 指令的时点，CPU 模块将暂停顺控程序的处理，而进行与 END 处理时的一般数据处理及智能功能模块的自动刷新（包括链接刷新）相同的处理。
但是，不进行 MELSECNET/10、MELSECNET/H 的低速循环刷新。



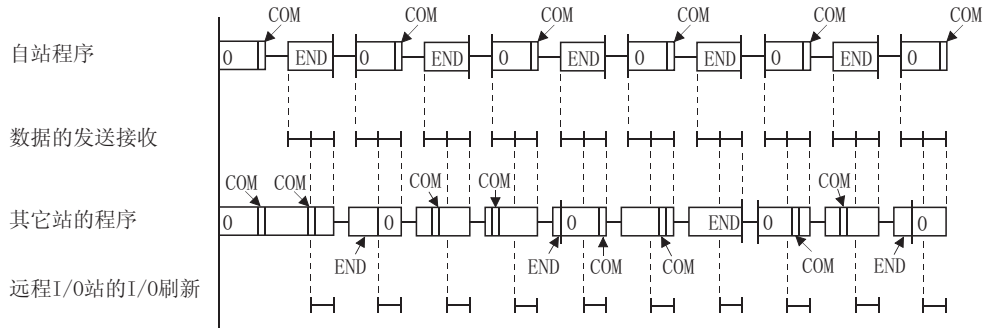
(4) COM 指令可在顺控程序中使用任意次。
但是，由于与外围设备的通信、智能功能模块的自动刷新（包括链接刷新）需要耗费一定的时间，因此顺控程序的扫描时间将会相应延长，应加以注意。

(5) 使用 COM 指令时的数据发送接收

(a) 未使用 COM 指令时的数据发送接收示例



(b) 使用了 COM 指令时的数据发送接收示例

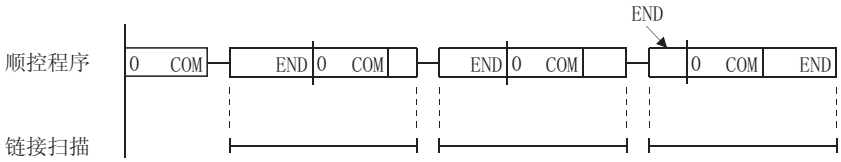


1) 自站中使用了 COM 指令时，如上述 (b) 项所示，与远程 I/O 站的数据发送接收次数可无条件增多，可以提高数据的发送接收速度。
2) 在其它站的扫描时间长于自站的扫描时间的情况下，通过在其它站中使用 COM 指令，可以防止发生如上述 (a) 项所示的无法获取数据的现象。
3) 如果对其它站使用 COM 指令，将在下述期间接收到来自于主站的指令时各进行 1 次链接刷新。

- 第0步 ～COM指令
- COM指令～COM指令
- COM指令～END指令

在此期间可各进行1次链接刷新。

(6) 在链接扫描时间长于自站的顺控程序的扫描时间的情况下，即使对自站指定 COM 指令，也不能加快数据的发送接收速度。



要 点

不能对以下程序使用 COM 指令：

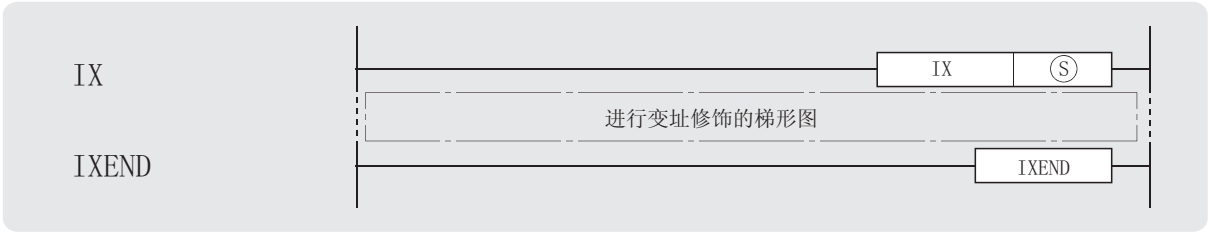
- 低速执行型程序
- 中断程序
- 恒定周期执行型程序

出 错

(1) 在 COM 指令中无运算出错。

7.6.10 整个梯形图的变址修饰 (IX、IXEND)

Basic　High performance　Process　Redundant　Universal



Ⓢ : 存储变址修饰数据的软元件的起始编号 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
Ⓢ	--	○		--					

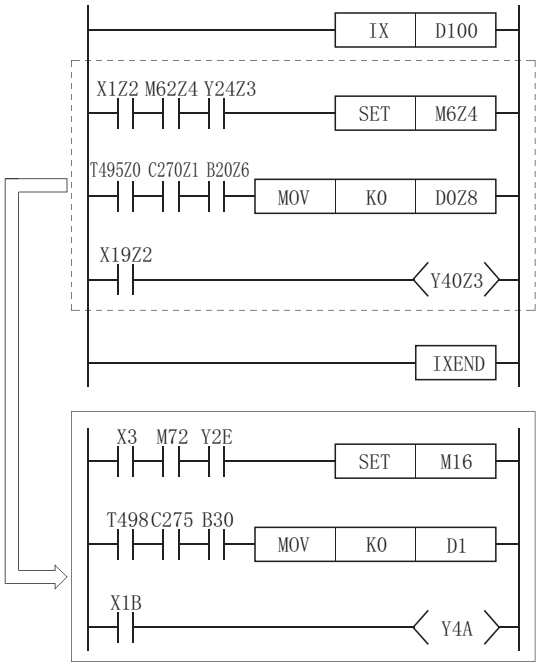
功　能

(1) 使用在变址修饰表中设置的变址修饰值，对梯形图内从 IX 指令起到 IXEND 指令为止的所有软元件进行软元件号的变址修饰。
关于变址修饰表的创建方法，请参阅 7.6.11 项。
变址修改表的构成以及相应的变址寄存器编号如下所示：

	软元件名	变址寄存器编号		软元件名	变址寄存器编号	
Ⓢ	定时器(T)的修饰值	Z0	Ⓢ +8	数据寄存器(D)的修饰值	Z8	
Ⓢ +1	计数器(C)的修饰值	Z1	Ⓢ +9	链接寄存器(W)的修饰值	Z9	
Ⓢ +2	输入(X)的修饰值	Z2	Ⓢ +10	文件寄存器(R)的修饰值	Z10	
Ⓢ +3	输出(Y)的修饰值	Z3	Ⓢ +11	缓冲寄存器I/O号(U)的修饰值	Z11	
Ⓢ +4	内部继电器(M)的修饰值	Z4	Ⓢ +12	缓冲寄存器(G)的修饰值	Z12	
Ⓢ +5	锁存继电器(L)的修饰值	Z5	Ⓢ +13	链接直接软元件网络号(J)的修饰值	Z13	*1
Ⓢ +6	链接继电器(B)的修饰值	Z6	Ⓢ +14	文件寄存器(ZR)的修饰值	Z14	
Ⓢ +7	变址继电器(V)的修饰值	Z7	Ⓢ +15	指针(P)的修饰值	Z15	

* 1:　使用基本型 QCPU 时，不支持 Z10 以后的变址寄存器。

(2) 进行软元件号的变址修饰时，通过对各个软元件预先设置修饰，对 IX ～ IXEND 指令之间的梯形图中所使用的全部软元件进行修饰值的加法运算，并根据加法运算后的软元件号执行程序处理。



修饰值

D100	8	T	(Z0)
D101	5	C	(Z1)
D102	2	X	(Z2)
D103	10	Y	(Z3)
D104	10	M	(Z4)
D105	20	L	(Z5)
D106	16	B	(Z6)
D107	20	V	(Z7)
D108	1	D	(Z8)

- 对X1、X19进行“2”的加法运算 → 作为X3、X1B进行处理。
- 对Y24、Y40进行“10”(AH)的加法运算 → 作为Y2E、Y4A进行处理。
- 对M6、M62进行“10”的加法运算 → 作为M16、M72进行处理。
- 对B20进行“16”(10H)的加法运算 → 作为B30进行处理。
- 对T495进行“8”的加法运算 → 作为T498进行处理。
- 对C270进行“5”的加法运算 → 作为C275进行处理。
- 对D0进行“1”的加法运算 → 作为D1进行处理。

(3) 对于 PLS、PLF、□□□P 的指令之类的，通过输入条件的上升沿仅执行 1 次的指令，不能通过 IX ～ IXEND 指令进行变址修饰。

(4) 由于修饰值的加法运算，软元件号超出了软元件范围时，将不能进行正常处理。

(5) 在进行顺控程序的 RUN 中写入时，不要执行 IX 指令、IXEND 指令。否则将不能进行正常处理。

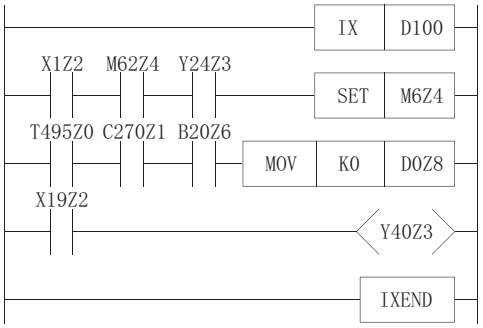
(6) 将修饰值以 BIN 值格式预先设置到任意的字软元件中后，在Ⓢ中对已设置了修饰值的软元件的起始编号进行指定。

(7) 在 IX ～ IXEND 指令中，不要同时执行扫描执行型程序与中断程序。

(8) 根据所使用的 GPP 功能软件包，有时需要进行程序展开或由用户创建程序。

(a) 需要由用户创建程序时 (使用 GX Developer 创建时)

对于通过 IX、IXEND 指令进行变址修饰的梯形图，应由用户进行变址寄存器的附加。*2

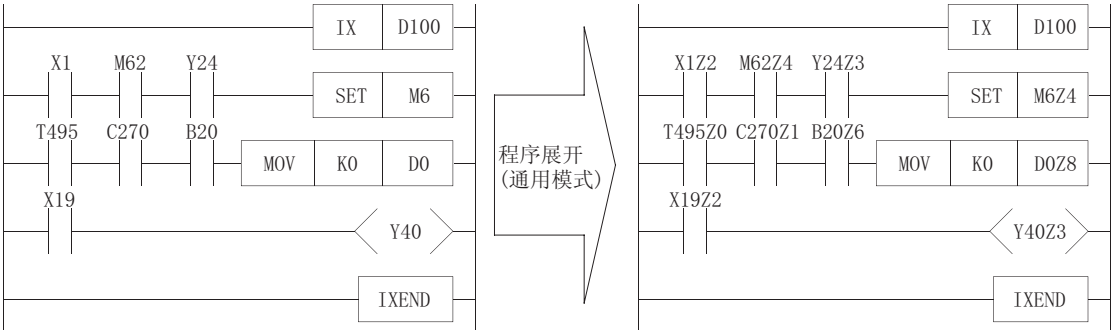


*2: 在执行 IXEND 指令之后，将 Zn 的值恢复为执行 IX 指令之前的原来的值。

(b) 进行程序的展开时 (使用 SW □ -GPPQ 创建时)

对于通过 IX、IXEND 指令进行变址修饰的梯形图，在进行了程序展开时，将被转换为使用了变址寄存器 (Zn) 的梯形图。*3

不能对 IX ～ IXEND 指令内的程序进行变址修饰。



*3: 在执行 IXEND 指令之后，将 Zn 的值恢复为执行 IX 指令之前的原来的值。

要 点

1. 将 IX ～ IXEND 指令用于普通的顺控程序及中断程序中时，应设置互锁以防止同时执行。互锁是指，将普通顺控程序的 IX ～ IXEND 指令之间设置 DI 以执行中断禁止。
2. 可以使用 IXDEV ～ IXSET 指令进行修饰值指定。
   有关详细内容请参阅 7.6.11 项。

出　错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· IX 指令、IXEND 指令未成对使用时。　　(出错代码： 4231)

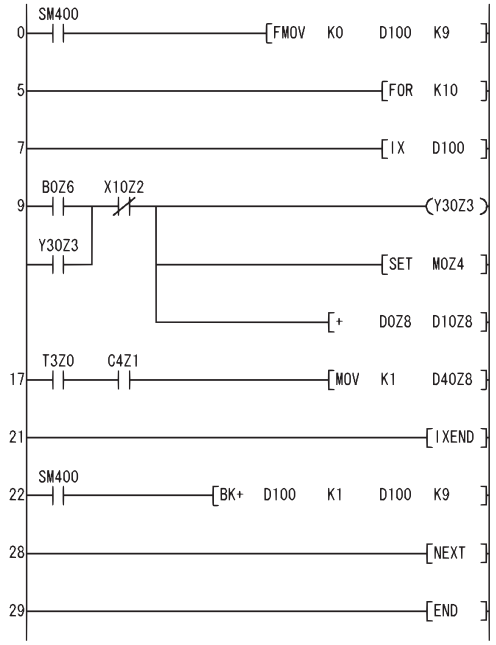
· 在执行 IX 指令后，执行 IXEND 指令之前执行了 END、FEND、GOEND、STOP 指令时。

(出错代码： 4231)

程序示例

(1) 以下为在变更软元件号的同时将同一梯形图执行 10 次的程序。

[梯形图模式]



[列表模式]

步	指令	软元件			
0	LD	SM400			
1	FMOV	K0	D100	K9	
5	FOR	K10			
7	IX	D100			
9	LD	B0Z6			
10	OR	Y30Z3			
11	ANI	X10Z2			
12	OUT	Y30Z3			
13	SET	M0Z4			
14	+	D0Z8	D10Z8		
17	LD	T3Z0			
18	AND	C4Z1			
19	MOV	K1	D40Z8		
21	IXEND				
22	LD	SM400			
23	BK+	D100	K1	D100	K9
28	NEXT				
29	END				

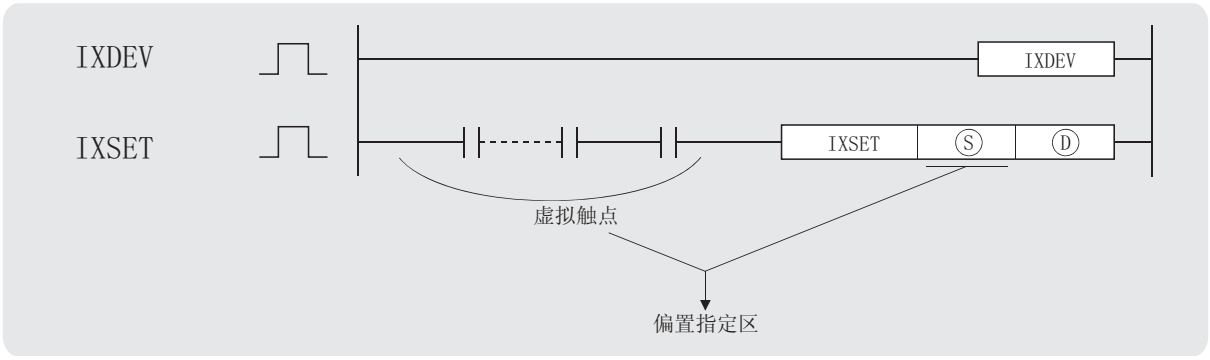
[动作]

	修饰值
D100	T的修饰值
D101	C的修饰值
D102	X的修饰值
D103	Y的修饰值
D104	M的修饰值
D105	L的修饰值
D106	B的修饰值
D107	V的修饰值
D108	D的修饰值

第1次	第2次	第3次		第10次
B0 →	B1 →	B2	------→	B9
X10 →	X11 →	X12	------→	X19
Y30 →	Y31 →	Y32	------→	Y39
M0 →	M1 →	M2	------→	M9
D0 →	D1 →	D2	------→	D9
D10 →	D11 →	D12	------→	D19
T3 →	T4 →	T5	------→	T12
C4 →	C5 →	C6	------→	C13
D40 →	D41 →	D42	------→	D49

7.6.11 整个梯形图的变址修饰中修饰值的指定 (IXDEV、IXSET)

Basic | High performance | Process | Redundant | Universal



Ⓢ : 存储变址修饰数据的软元件的起始编号 (仅指针)P□ (指针)。

Ⓓ : 存储变址修饰数据的软元件的起始编号 (除指针以外)(BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				P
Ⓢ	--	--		--					○
Ⓓ	--	○		--					--

功 能

(1) 该指令是用于创建 IX、IXEND 指令中使用的变址修饰表的指令。

(2) 将通过偏置指定区指定的软元件偏置值设置到Ⓓ中指定的变址修饰表中。

(3) 未指定的情况下将输入 0。

(4) 字软元件也以触点 (字软元件的位指定) 表示。
数据寄存器 10(D10) 是通过 D10.0 指定。
(位号可使用 0 ～ F 中的任意值。)

(5) 指定方法如下所示。*1(□为偏置值。XX 为任意值。)

软元件	T	C	X	Y	M	L	V	B
指定方法	T□	C□	X□	Y□	M□	L□	V□	B□
软元件	D	W	R	U/G		J		ZR
指定方法	D□.XX	W□.XX	R□.XX	U□\G□.XX		J□\B□ *2		ZR□.XX
软元件	P							
指定方法	IXSET Ⓢ Ⓓ *3							

*1: 使用基本型 QCPU 时，不能使用 R、U/G、J、ZR、P 的软元件。

*2: 将 J□\后面的软元件指定为 B、W、X、Y 后，与其对应的偏置值也将被设置。

*3: 使用基本型 QCPU 时，应指定虚拟的软元件号。
Ⓢ变为 P□。

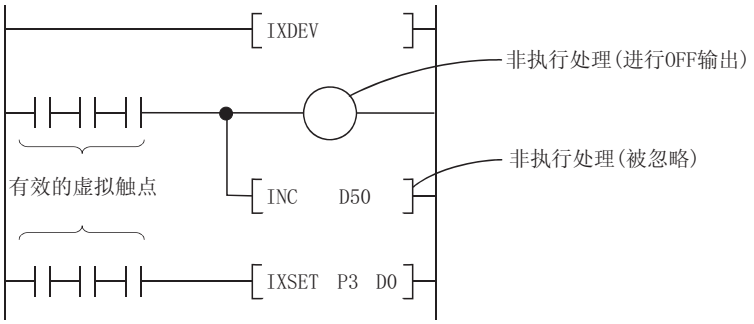
(6) 在偏置指定区记述了相同类型的软元件时，最后设置的软元件有效。

(7) 应将 IXDEV 与 IXSET 指令成对使用。

(8) ZR 时的有效范围为 0 ～ 32767。(将指定的软元件号除以 32768 所得的余数作为偏置值。)

(9) 在偏置指定区的虚拟触点只对于 IXDEV 指令～ IXSET 指令范围内的 LD 或者 AND 有效。如果记述了其它指令，IXDEV 指令～ IXSET 指令将不会执行。

例



出 错

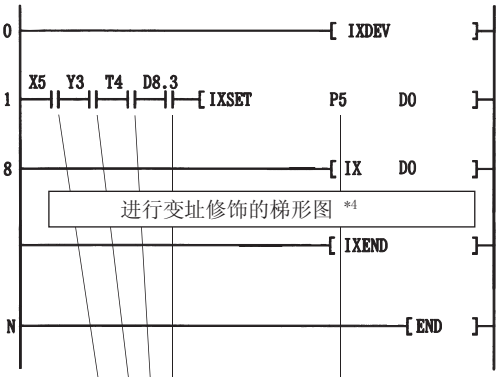
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· IXDEV 与 IXSET 指令未成对使用。 (出错代码： 4231)

程序示例

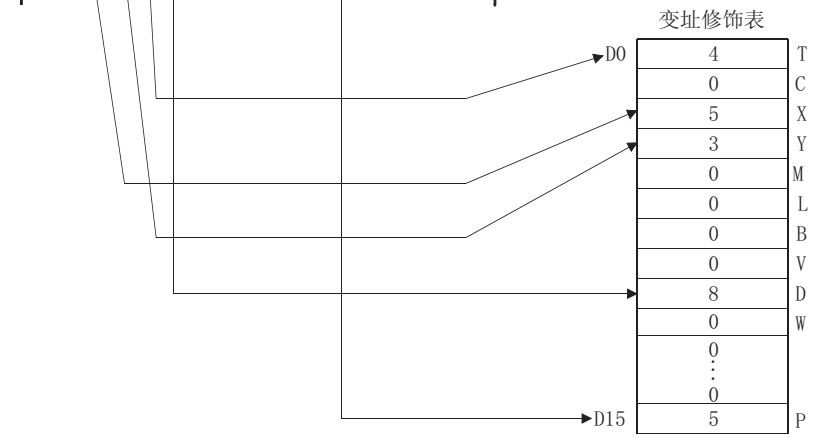
(1) 以下为对输入 (X)、输出 (Y)、数据寄存器 (D)、指针 (P) 的修饰值进行变更的程序。
使用基本型 QCPU 时，不能使用 R、U/G、J、ZR、P 的软元件。

[梯形图模式]



[列表模式]

步	指令	软元件
0	IXDEV	
1	LD	X5
2	AND	Y3
3	AND	T4
4	AND	D8.3
5	IXSET	P5
		D0
8	IX	D0
⋮	IXEND	
N	END	



*4：关于使用了 IX ～ IXEND 指令变址修饰，请参阅 7.6.10 项。

7.7 数据表操作指令

7.7.1 将数据写入数据表 (FIFW(P))

Basic | High performance | Process | Redundant | Universal

Ⓢ : 写入表的数据或者存储数据的软元件编号 (BIN16 位)。

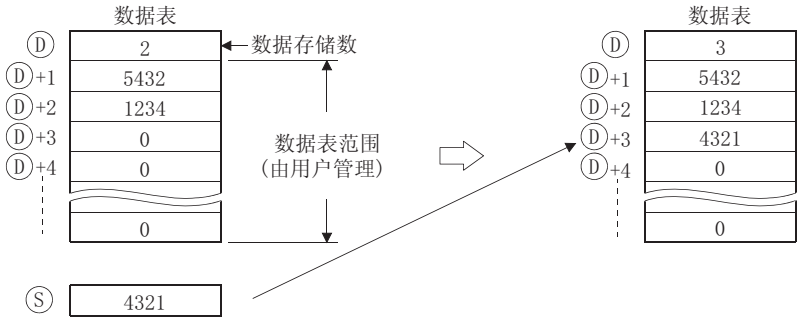
Ⓓ : 表的起始编号 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○	○		○				○	--
Ⓓ	--	○		--				--	--

功 能

(1) 将Ⓢ中指定的 16 位数据存储到Ⓓ中指定的数据表中。

将表中存储的数据数量存储到Ⓓ中，将Ⓢ中指定的数据依次存储到Ⓓ+1 的后面。



(2) 初次执行 FIFW 指令时，应预先清除Ⓓ中指定的软元件的值。

(3) 写入到数据表中的数据数量及数据表范围应由用户管理。
(参阅程序示例 (2))

出 错

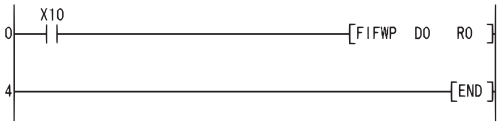
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 执行了 FIFW 指令时，数据表范围超出了相应软元件的范围时。 (出错代码： 4101)

程序示例

(1) 以下为 X10 变为 ON 时，将 D0 的数据存储到 R0 后面的数据表中的程序。

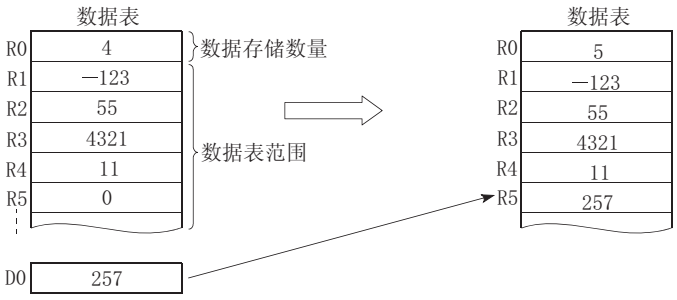
[梯形图模式]



[列表模式]

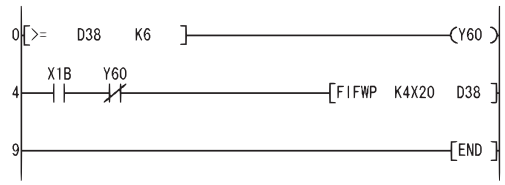
步	指令	软元件	
0	LD	X10	
1	FIFWP	D0	R0
4	END		

[动作]



(2) 以下为 X1B 变为 ON 时，将 X20 ～ X2F 的数据存储到 D38 ～ D44 的数据表中，数据存储数量超过了 6 时，将 Y60 置为 ON 使 FIFW 指令变为禁用状态的程序。

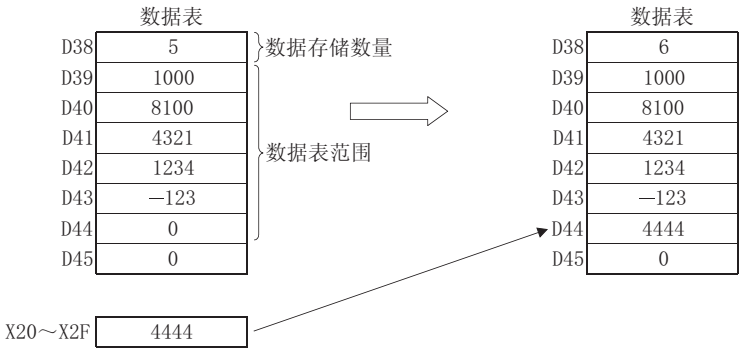
[梯形图模式]



[列表模式]

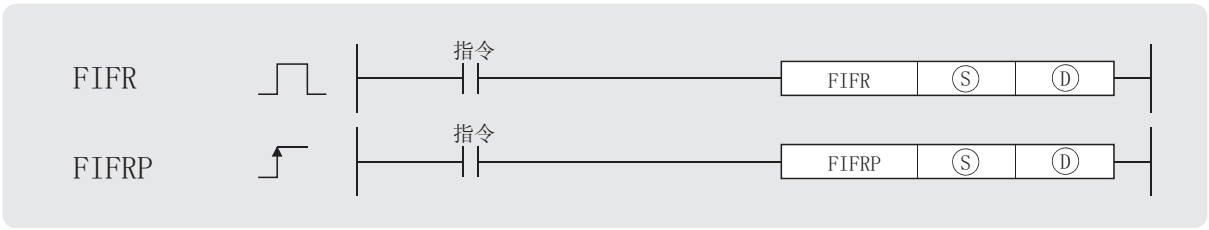
步	指令	软元件	
0	LD>=	D38	K6
3	OUT	Y60	
4	LD	X1B	
5	ANI	Y60	
6	FIFWP	K4X20	D38
9	END		

[动作]



7.7.2 从表中读取最旧的数据 (FIFR(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ : 存储从表中读取的数据的软元件的起始编号 (BIN16 位)。

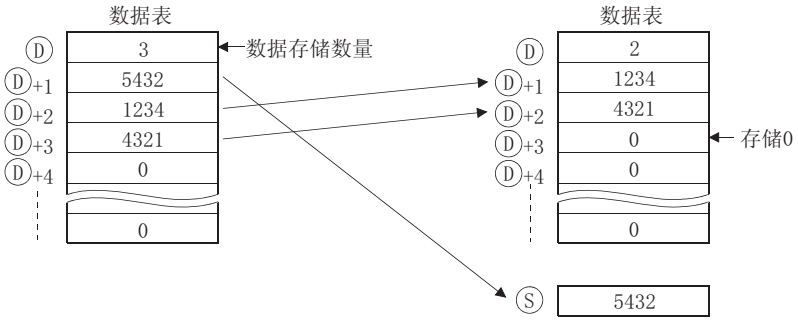
Ⓓ : 表的起始编号 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
Ⓢ	○	○		○				--	
Ⓓ	--	○		--				--	

7

功 能

(1) 将Ⓓ中指定的表的最旧数据 (Ⓓ+1) 存储到Ⓢ中指定的软元件中。
执行 FIFR 指令后数据表的数据逐个向前填充对齐。



(2) Ⓓ中存储的值为 0 时，应由用户设置互锁以防止对其执行 FIFR 指令。(参阅程序示例 (1))

出 错

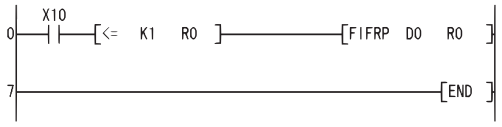
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- Ⓓ中的值为 0 的情况下执行了 FIFR 指令时。 (出错代码： 4100)
- 执行了 FIFR 指令时，数据表范围超出了相应软元件范围时。 (出错代码： 4101)

程序示例

(1) 以下为 X10 变为 ON 时，从 R0 ～ R7 的数据表中将 R1 的数据存储到 D0 中的程序。

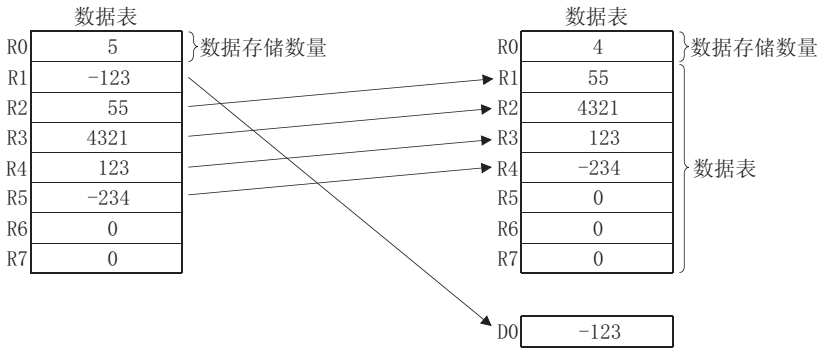
[梯形图模式]



[列表模式]

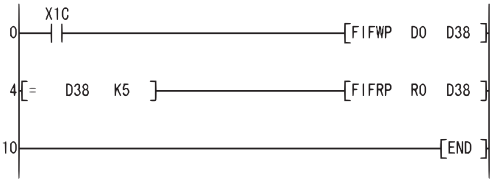
步	指令	软元件	
0	LD	X10	
1	AND<=	K1	R0
4	FIFRP	D0	R0
7	END		

[动作]



(2) 以下为 X1C 变为 ON 时，将 D0 的数据存储到 D38 ～ D43 的数据表中，数据存储数量达到 5 时，将数据表 D39 的数据存储到 R0 中的程序。

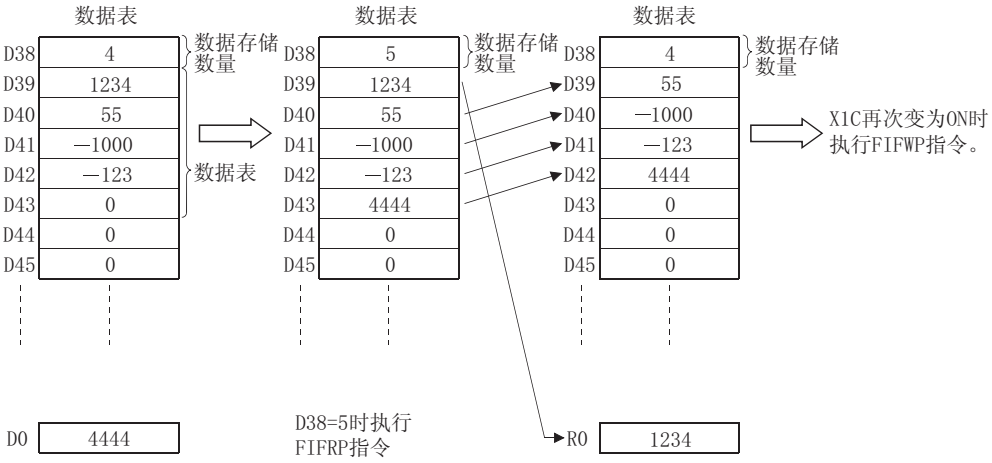
[梯形图模式]



[列表模式]

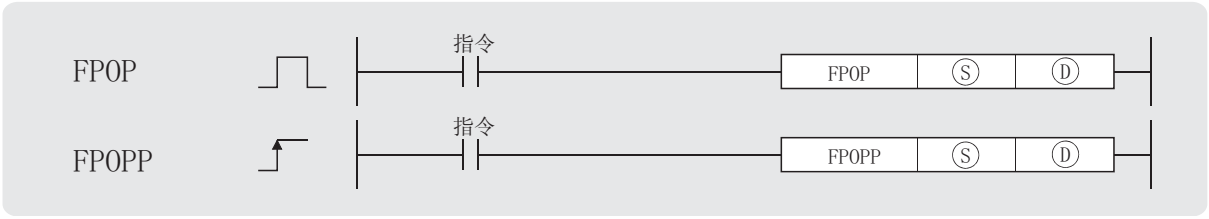
步	指令	软元件	
0	LD	X1C	
1	FIFWP	D0	D38
4	LD=	D38	K5
7	FIFRP	R0	D38
10	END		

[动作]



7.7.3 从数据表中读取最新数据 (FPOP(P))

Basic | High performance | Process | Redundant | Universal



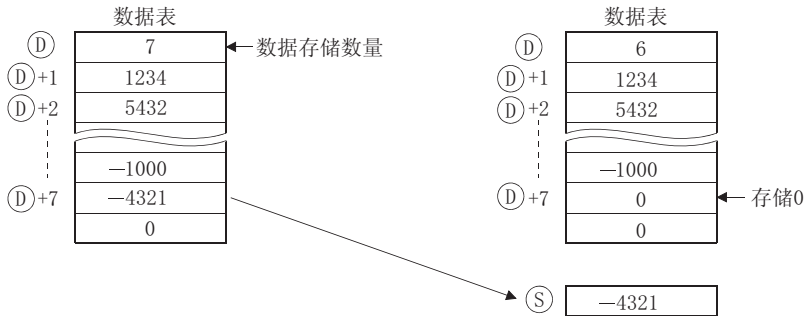
Ⓢ : 存储从表中读取的数据的软元件的起始编号 (BIN16 位)。

Ⓓ : 表的起始编号 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
Ⓢ	○	○		○				--	
Ⓓ	--	○		--				--	

功 能

(1) 将Ⓓ中指定的表的最新数据存储到Ⓢ中指定的软元件中。
执行 FPOP 指令后，存储通过 FPOP 指令读取的数据的软元件将变为 0。



(2) Ⓓ中存储的值为 0 时，应由用户设置互锁以防止对其执行 FPOP 指令。(参阅程序示例 (1))

出 错

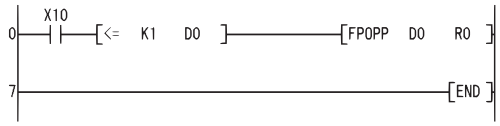
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- Ⓓ中的值为 0 的情况下执行了 FPOP 指令时。 (出错代码： 4100)
- 执行了 FPOP 指令时，数据表范围超出了相应软元件范围时。 (出错代码： 4101)

程序示例

(1) 以下为 X10 变为 ON 时，从 R0 ～ R7 的数据表中将最后存储的数据存储到 D0 中的程序。

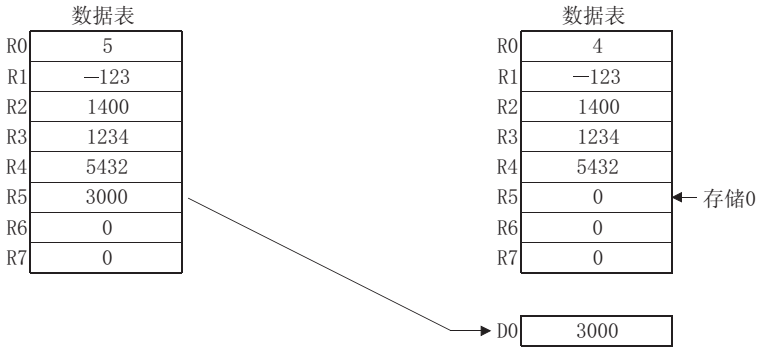
[梯形图模式]



[列表模式]

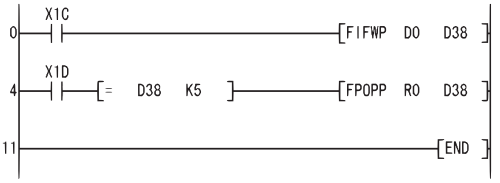
步	指令	软元件	
0	LD	X10	
1	AND<=	K1	D0
4	FPOPP	D0	R0
7	END		

[动作]



(2) 以下为 X1C 变为 ON 时，将 D0 的数据存储到 D38 ～ D43 的数据表中，数据存储数量达到 5 时将 X1D 置为 ON，将数据表中最后存储的数据存储到 R0 中的程序。

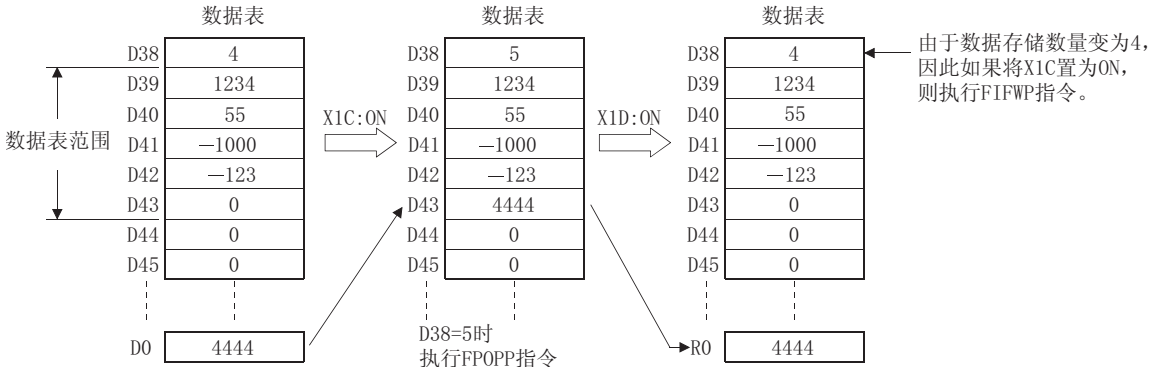
[梯形图模式]



[列表模式]

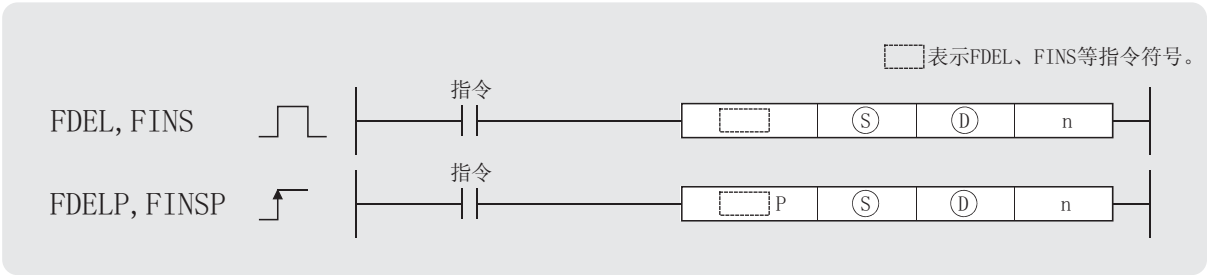
步	指令	软元件	
0	LD	X1C	
1	FIFWP	D0	D38
4	LD	X1D	
5	AND=	D38	K5
8	FPOPP	R0	D38
11	END		

[动作]



7.7.4 数据表的数据删除和插入 (FDEL(P)、FINS(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ ：存储插入数据的软元件的起始编号 (BIN16 位)。
存储删除数据的软元件的起始编号 (BIN16 位)。
Ⓓ ：表的起始编号 (BIN16 位)。
n ：进行插入 / 删除的表位置 (BIN16 位)。

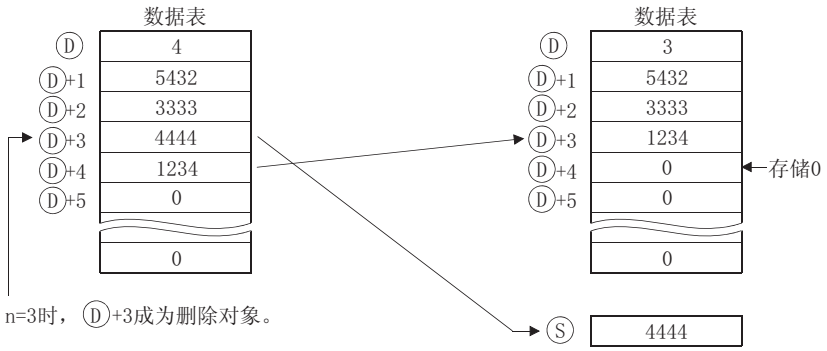
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○	○		○				--	--
Ⓓ	--	○		--				--	--
n	○	○		○				○	--

功 能

FDEL

将Ⓓ中指定的数据表的第 n 个数据删除后，存储到Ⓢ中指定的软元件中。

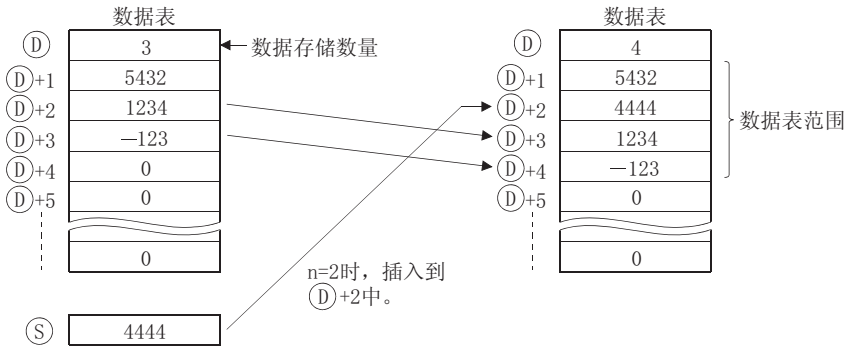
执行 FDEL 指令后，数据表的第 n+1 后面的数据将逐个向前填充对齐。



FINS

将Ⓢ中指定的 16 位数据插入到Ⓓ中指定的数据表的第 n 号中。

执行 FINS 指令后，数据表的第 n 号开始的数据将逐个向下顺延。



出　错

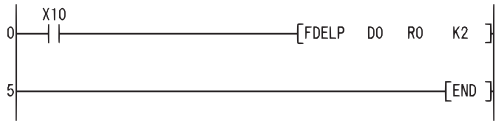
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- 使用 FDEL 指令时，从Ⓓ开始的第 n 号大于数据存储数量时。 (出错代码： 4101)
- 使用 FINS 指令时，从Ⓓ开始的第 n 号大于数据存储数量 +1 时。 (出错代码： 4101)
- 使用 FDEL、FINS 指令时，n 的值超出了Ⓓ的表的软元件范围时。 (出错代码： 4101)
- 在 n=0 的情况下执行了 FDEL、FINS 指令时。 (出错代码： 4100)
- 在Ⓓ的值为 0 的情况下执行了 FDEL 指令时。 (出错代码： 4100)
- 执行了 FDEL、FINS 指令时，数据表范围超出了相应软元件范围时。 (出错代码： 4101)

程序示例

(1) 以下为 X10 变为 ON 时，从 R0 ～ R7 的数据表中将第 2 个数据删除后，存储到 D0 中的程序。

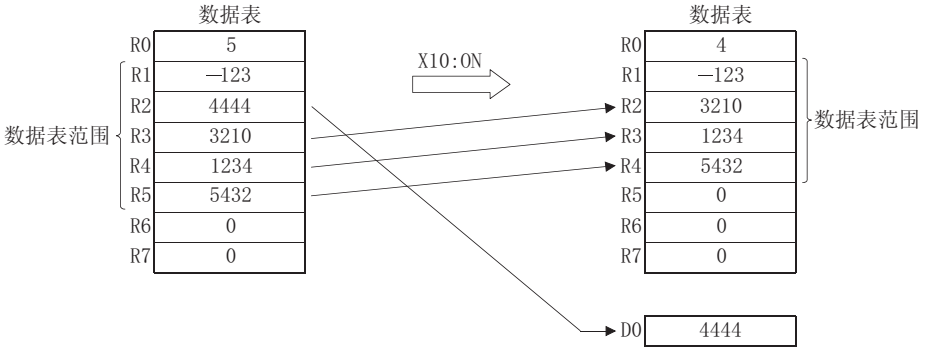
[梯形图模式]



[列表模式]

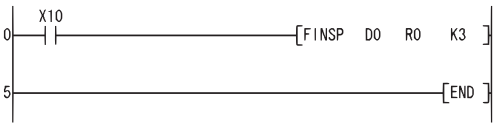
步	指令	软元件		
0	LD	X10		
1	FDELP	D0	R0	K2
5	END			

[动作]



(2) 以下为 X10 变为 ON 时，将 D0 的数据插入到 R0 ～ R7 的数据表的第 3 号中的程序。

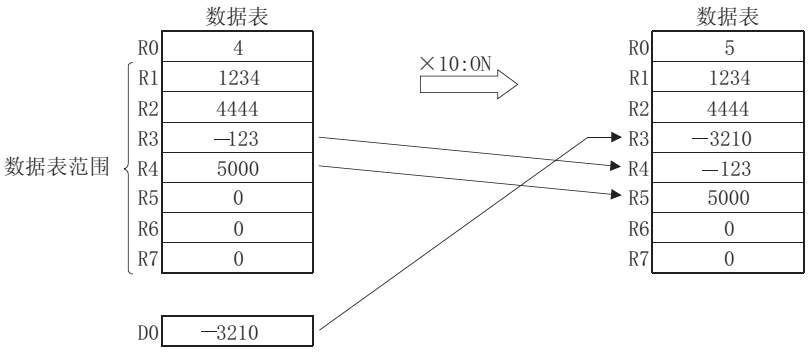
[梯形图模式]



[列表模式]

步	指令	软元件		
0	LD	X10		
1	FINSP	D0	R0	K3
5	END			

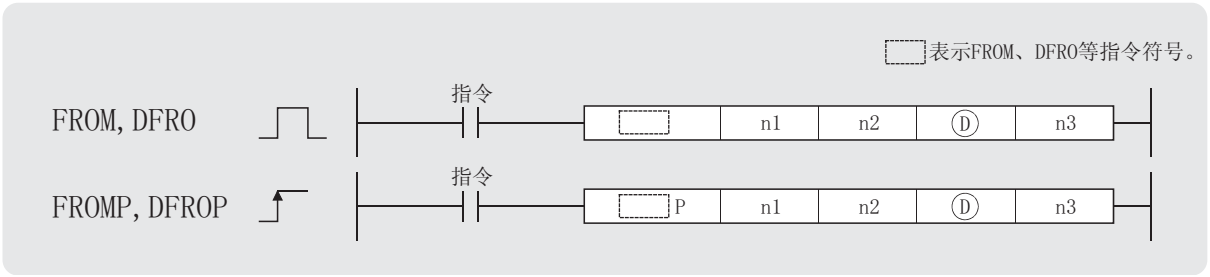
[动作]



7.8 缓冲存储器访问指令

7.8.1 从智能功能模块中读取 1 字 /2 字数据 (FROM(P)、DFRO(P))

Basic | High performance | Process | Redundant | Universal



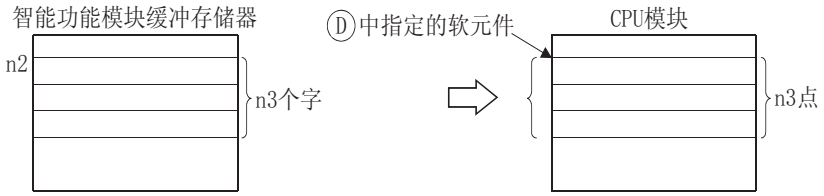
n1 ： 智能功能模块的起始 I/O 编号 (BIN16 位)。
n2 ： 读取数据的起始地址 (BIN16 位)。
Ⓓ ： 存储读取数据的软元件起始编号 (BIN16/32 位)。
n3 ： 读取的数据数 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它 U
	位	字		位	字				
n1		○			○				○
n2		○			○				--
Ⓓ		○			--				--
n3		○			○				--

功 能

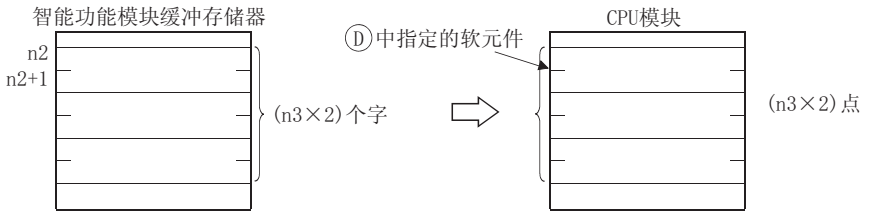
FROM

(1) 从 n1 中指定的智能功能模块的 n2 中指定的缓冲存储器地址开始，读取 n3 个字的数据后，存储到Ⓓ中指定的软元件后面。



DFRO

(1) 从 n1 中指定的智能功能模块的 n2 中指定的缓冲存储器地址开始，读取 (n3 × 2) 个字的数据后，存储到Ⓓ中指定的软元件后面。



要 点

智能功能模块的数据读取也可使用智能功能模块软元件进行。
关于智能功能模块软元件，请参阅所使用的 CPU 模块的用户手册 (功能解说 / 程序基础篇)。

出 错

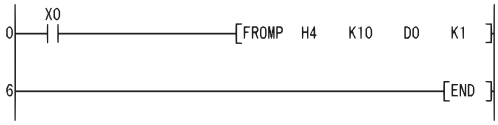
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- 执行指令时无法与智能功能模块进行通信时。 (出错代码： 1412)
- 执行指令时检测出智能功能模块异常时。 (出错代码： 1402)
- n1 中指定的 I/O 号不是智能功能模块时。 (出错代码： 2110)
- 从Ⓓ中指定的软元件开始的 n3 点 (DFRO 时为 2 × n3 点) 超出了指定软元件范围时。 (出错代码： 4101)
- n2 中指定的地址超出缓冲存储器范围时。 (出错代码： 4101)

程序示例

(1) 以下为 X0 变为 ON 时，在 I/O 号为 040 ～ 05F 的 A68AD 中，将 CH1 的数字值读取到 D0 中的程序。(从缓冲存储器的地址 10 开始读取 1 个字的数据。)

[梯形图模式]

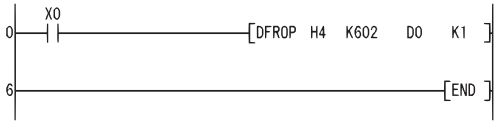


[列表模式]

步	指令	软元件			
0	LD	X0			
1	FROMP	H4	K10	D0	K1
6	END				

(2) 以下为 X0 变为 ON 时，在 I/O 号为 040 ～ 05F 的 AD71 中，将 X 轴的当前值读取到 D0、D1 中的程序。(从缓冲存储器的地址 602，603 开始读取 2 个字的数据。)

[梯形图模式]

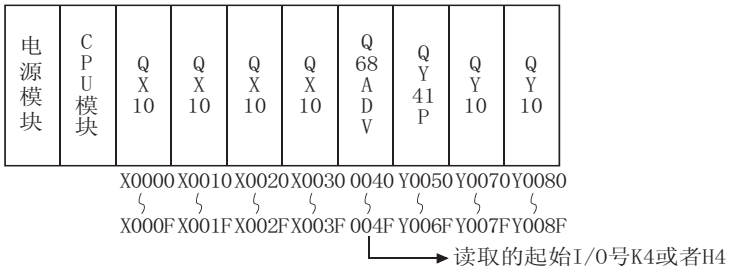


[列表模式]

步	指令	软元件			
0	LD	X0			
1	DFROP	H4	K602	D0	K1
6	END				

备 注

1. 当以 16 进制 4 位数表示安装智能功能模块的插槽的起始 I/O 号时，n1 的值由高 3 位指定。

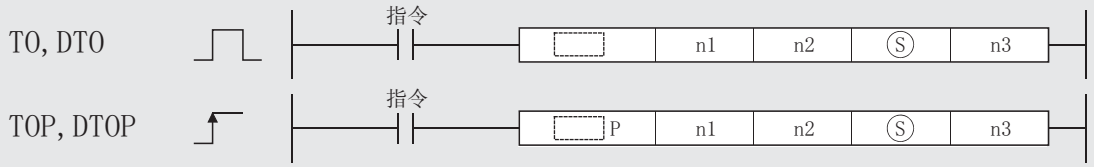


2. 在 QCPU 中建立了 FROM/DFRO 指令的自动互锁。

7.8.2 将 1 字 /2 字数据写入智能功能模块 (TO(P)、DTO(P))

Basic | High performance | Process | Redundant | Universal

表示TO、DTO等指令符号。



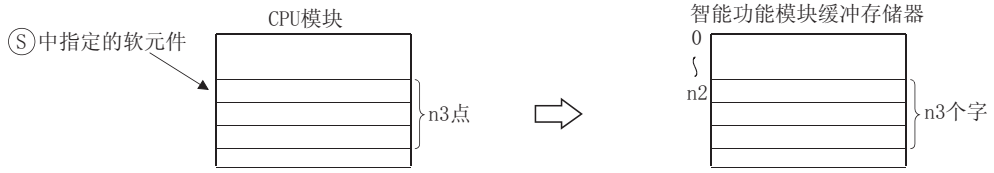
n1 ： 智能功能模块的起始编号 (BIN16 位)。
n2 ： 用于数据写入的起始地址 (BIN16 位)。
Ⓢ： 写入数据或者存储写入数据的软元件的起始编号 (BIN16/32 位)。
n3 ： 写入的数据数 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			K、H	U
n1		○			○			○	○
n2		○			○			○	--
Ⓢ		○			--			○	--
n3		○			○			○	--

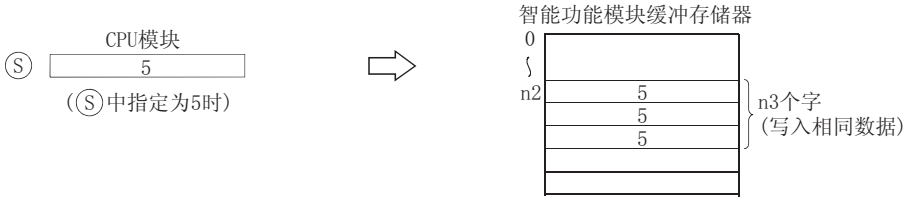
功 能

TO

将从Ⓢ中指定的软元件开始的 n3 点的数据，写入到 n1 中指定的智能功能模块的 n2 中指定的缓冲存储器地址的后面。

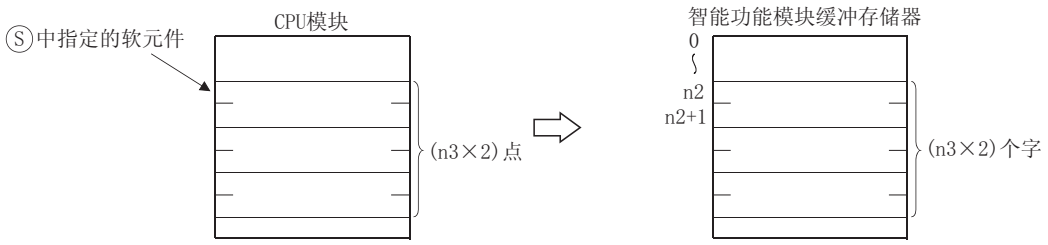


在Ⓢ中指定了常数时，将相同数据 (Ⓢ中指定的值) 写入到从指定缓冲存储器开始的 n3 个字中。
(Ⓢ中的可指定范围为 -32768 ～ 32767 或者 0_H ～ $FFFF_H$。)

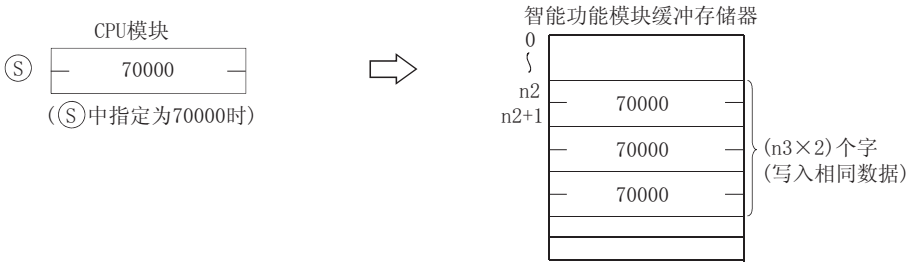


DTO

将从Ⓢ中指定的软元件开始的 (n3 × 2) 点的数据，写入到 n1 中指定的智能功能模块的 n2 中指定的缓冲存储器地址的后面。



在Ⓢ中指定了常数时，将相同数据 (Ⓢ中指定的值) 写入到从指定缓冲存储器开始的 (n3 × 2) 个字中。
(Ⓢ中的可指定范围为 -2147483648 ～ 2147483647 或者 0_H ～ $FFFFFFFF_H$。)



要 点

智能功能模块的数据写入也可使用智能功能模块软元件进行。
关于智能功能模块软元件，请参阅所使用的 CPU 模块的用户手册 (功能解说 / 程序基础篇)。

出 错

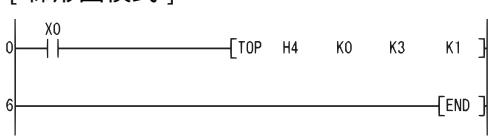
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- 执行指令时无法与智能功能模块进行通信时。 (出错代码： 1412)
- 执行指令时检测出智能功能模块异常时。 (出错代码： 1402)
- n1 中指定的 I/O 号不是智能功能模块时。 (出错代码： 2110)
- 从Ⓢ中指定的软元件开始的 n3 点 (DTO 时为 2 × n3 点) 超出了指定软元件范围时。 (出错代码： 4101)
- n2 中指定的地址超出缓冲存储器范围时。 (出错代码： 4101)

程序示例

(1) 以下为 X0 变为 ON 时，在 I/O 号为 040 ～ 04F 的 Q68ADV 中，将 CH1 及 CH2 设置为 “ 进行 A/D 转换 ” 的程序。(在缓冲存储器的地址 0 中写入 “ 3 ”。)

[梯形图模式]

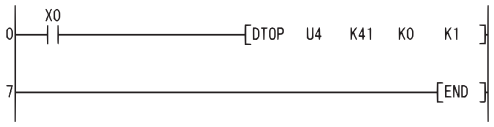


[列表模式]

步	指令	软元件
0	LD	X0
1	TOP	H4 K0 K3 K1
6	END	

(2) 以下为 X0 变为 ON 时，在 I/O 号为 040 ～ 05F 的 AD71 中，将 X 轴的当前值置为 0 的程序。
(缓冲存储器的地址 41、42 中写入 0。)

[梯形图模式]

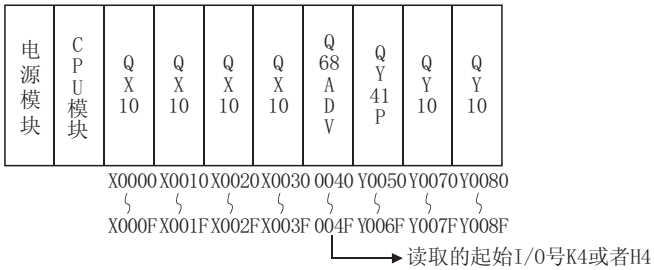


[列表模式]

步	指令	软元件
0	LD	X0
1	DTOP	U4 K41 K0 K1
7	END	

备 注

1. 当以 16 进制 4 位数表示安装智能功能模块的插槽的起始 I/O 号时，n1 的值由高 3 位指定。

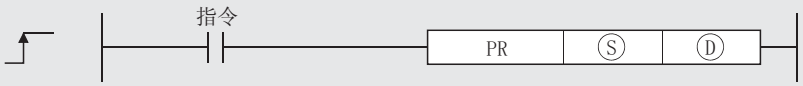


2. 在 QCPU 中建立了 TO/DTO 指令的自动互锁。

7.9 显示指令

7.9.1 ASCII 码打印指令 (PR)

Basic | High performance | Process | Redundant | Universal

PR 

Ⓢ : ASCII 码或者存储 ASCII 码的软元件的起始编号 (字符串)。

Ⓓ : 输出 ASCII 码的输出模块的起始编号 (位)。

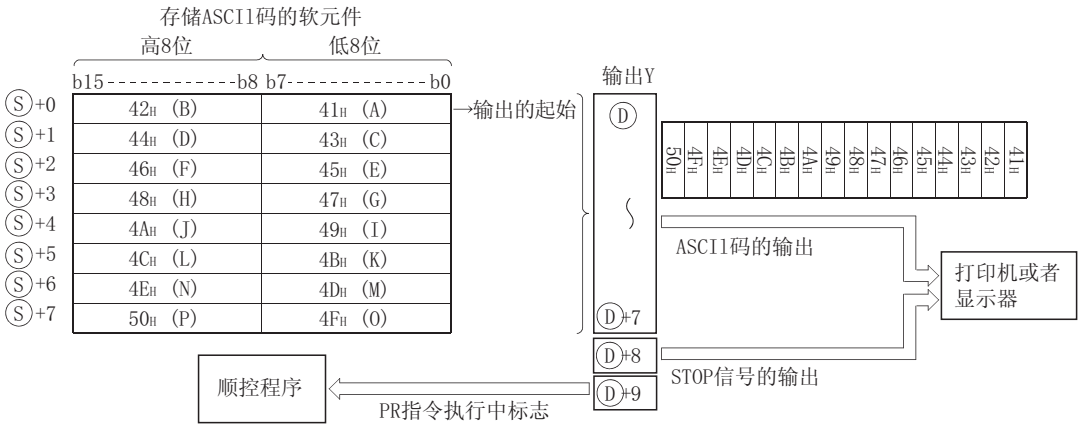
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
Ⓢ	--	△*1		--			○	○	--
Ⓓ	○ (仅Y)	--		--			○	--	--

＊1: 局部软元件以及各程序中设置的文件寄存器不能使用。

功 能

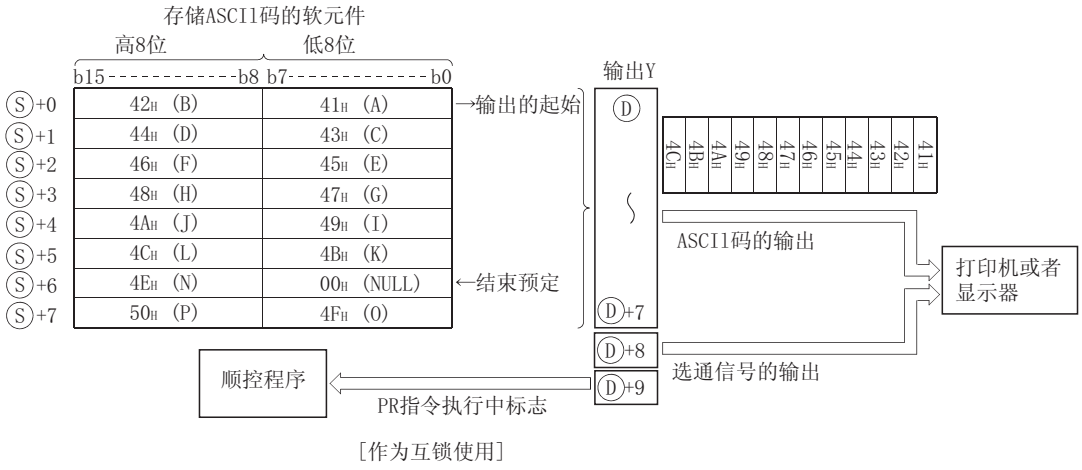
(1) 将Ⓢ中指定的 ASCII 码或者存储在软元件号后面的 ASCII 码输出到Ⓓ中指定的输出模块中。输出的字符数根据 SM701(输出字符数切换) 的 ON/OFF 状态而不同。

(a) SM701 为 ON 时，从Ⓢ中指定的软元件开始的 8 点 (16 个字符) 将成为输出对象。



[作为互锁使用]

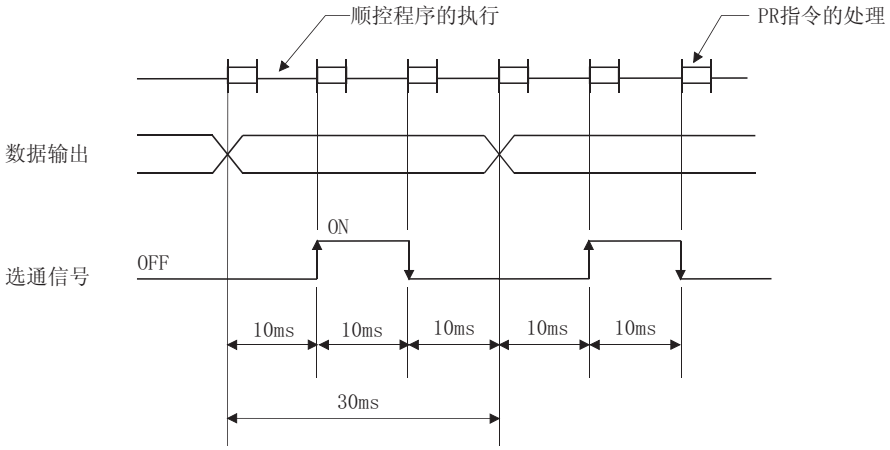
(b) SM701 为 OFF 时，从Ⓢ中指定的软元件开始至 00H 码为止将成为输出对象。



[作为互锁使用]

(2) 输出模块中使用的点数为Ⓓ中指定的 Y 地址号开始的 10 点。

(3) 来自于输出模块的输出信号以每 30ms 一个字符的速率进行发送。
因此，完成 (n) 个指定字符数的传送所需要的时间为 30ms × n(ms)。
PR 指令通过 10ms 中断，执行数据输出→选通信号 ON →选通信号 OFF。在以上处理之间的时间，继续执行其它指令。



(4) 除通过输出模块输出 ASCII 码以外，还通过Ⓓ+8 的软元件输出选通信号 (10ms ON，20ms OFF)。

(5) 执行 PR 指令后，PR 指令执行中标志 (Ⓓ+9 的软元件) 将变为 ON 状态，直至指定字符数的 ASCII 码发送完毕。

(6) 虽然可以多次使用 PR、PRC 指令，但应通过 PR 指令执行中标志 (Ⓓ+9 的软元件) 采取互锁，以防止二个指令同时变为 ON。

(7) 如果在 ASCII 码的输出过程中更改了存储 ASCII 码的软元件的内容，则将输出更改后的数据。

出　错

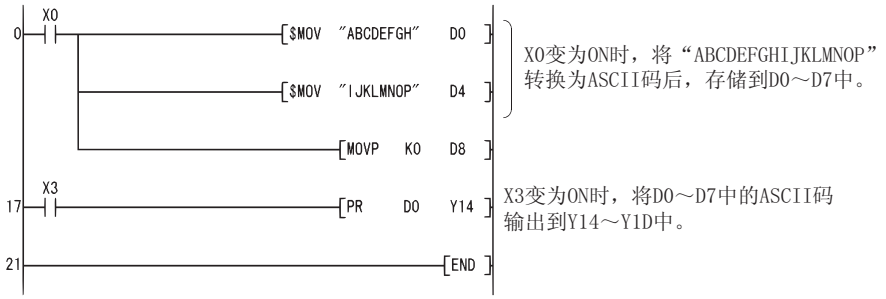
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· SM701 为 OFF 时，Ⓢ中指定的软元件的范围内没有 00H 码时。　　(出错代码 : 4101)

程序示例

(1) 以下为 X0 变为 ON 时，将 “ ABCDEFGHIJKLMNOP ” 转换为 ASCII 码后存储到 D0 ～ D7 中，X3 变为 ON 时，将 D0 ～ D7 中的 ASCII 码输出到 Y14 ～ Y1D 中的程序。

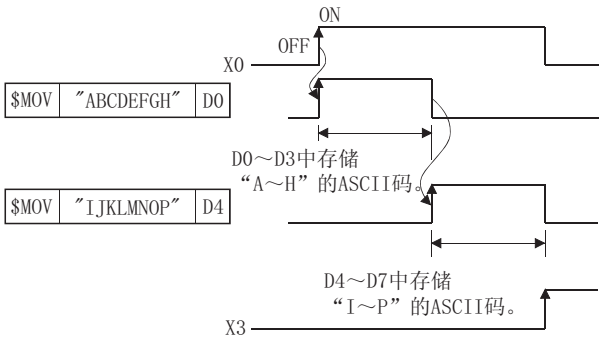
[梯形图模式]

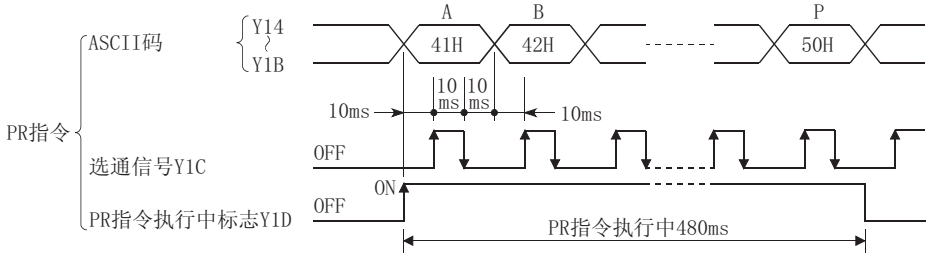


[列表模式]

步	指令	软元件	
0	LD	X0	
1	$MOV	"ABCDEFGH"	D0
8	$MOV	"IJKLMNOP"	D4
15	MOVP	K0	D8
17	LD	X3	
18	PR	D0	Y14
21	END		

[时序图]

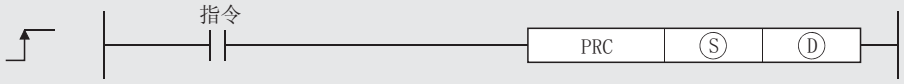




7.9.2 注释打印指令 (PRC)

Basic | High performance | Process | Redundant | Universal

PRC



Ⓢ : 进行注释打印的软元件的起始编号 (软元件名)。

Ⓓ : 进行注释输出的输出模块的起始编号 (位)。

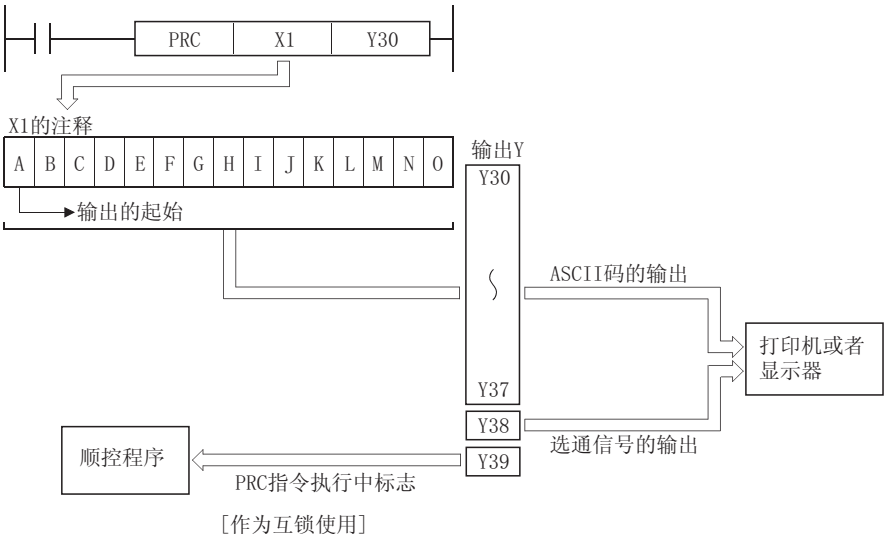
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它 P、I、J、U
	位	字		位	字				
Ⓢ	○	○		○			--	--	○
Ⓓ	○(仅Y)	--		--			--	--	--

功 能

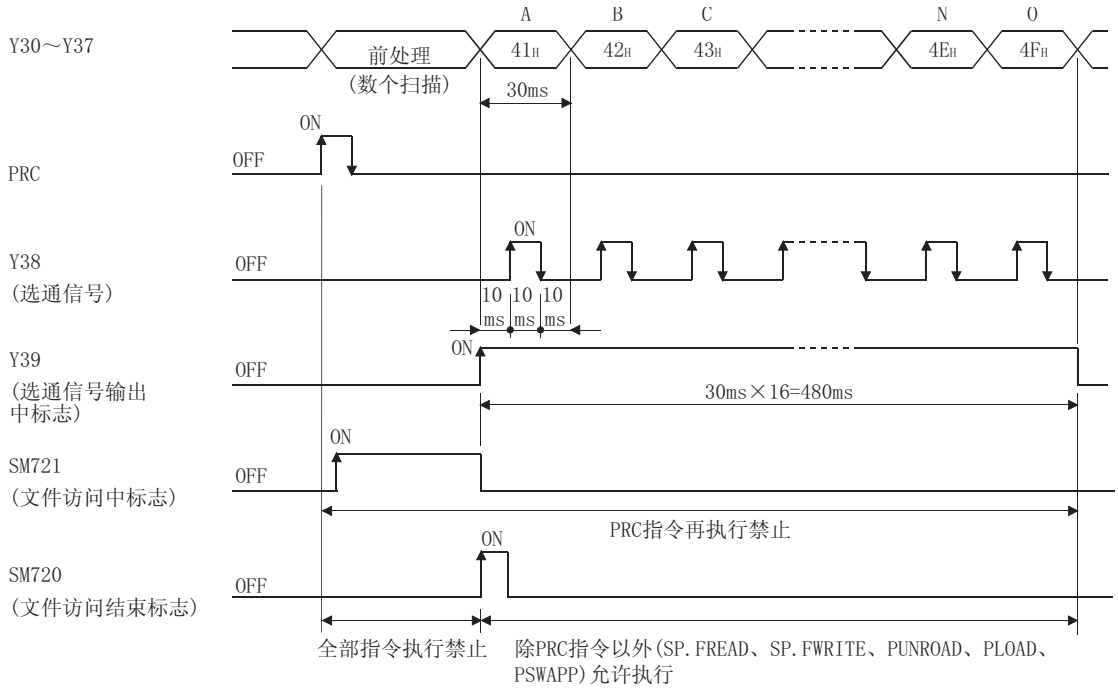
(1) 将Ⓢ中指定的软元件注释 (ASCII 码) 输出到Ⓓ中指定的输出模块中。
输出的字符数根据 SM701 的 ON/OFF 状态而不同。

- SM701 为 OFF 时 : 32 个字符的注释
- SM701 为 ON 时 : 注释的高位 16 个字符

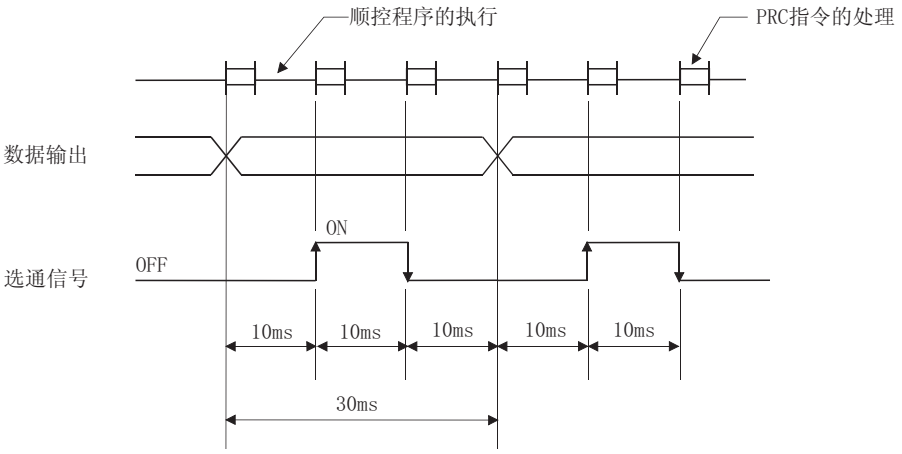
输出模块中使用的点数为Ⓓ中指定的 Y 地址号开始的 10 点。



[时序图]



(2) 来自于输出模块的输出信号以每 30ms 一个字符的速率进行发送。
因此，完成指定字符数的传送所需要的时间为 30ms × n(ms)。
PRC 指令通过 10ms 中断，执行数据输出→选通信号 ON →选通信号 OFF。在以上处理之间的时间，继续执行其它指令。



(3) 除通过输出模块输出 ASCII 码以外，还通过Ⓓ+8 的软元件输出选通信号 (10ms ON，20ms OFF)。

(4) 执行 PRC 指令后，PRC 指令执行中标志 (Ⓓ+9 的软元件) 将变为 ON 状态，直至指定字符数的 ASCII 码发送完毕。

(5) 虽然可以多次使用 PRC 指令，但应通过 PRC 指令执行中标志 (Ⓓ+9 的软元件) 采取互锁，以防止指令同时变为 ON。

(6) 如果未将注释登录到Ⓢ中指定的软元件中，将不执行处理。

(7) 读取注释时，指令结束后 SM720 将 ON 1 个扫描。
此外，在指令执行过程中 SM721 将变为 ON。
在 SM721 处于 ON 状态时，不能执行 PRC 指令。即使执行也将变为无处理。

要 点

1．PRC 指令中使用的软元件注释使用存放在存储卡上的注释文件。
不能使用内存中存储的注释文件。
2．PRC 指令中使用的注释文件是在可编程控制器参数的“ 可编程控制器文件设置 ”中进行设置。
如果未在可编程控制器文件设置中对所使用的注释文件进行设置，将无法通过 PRC 指令进行软元件注释输出。
3．不要在中断程序期间执行 PRC 指令。
如果执行将会导致误动作。

出 错

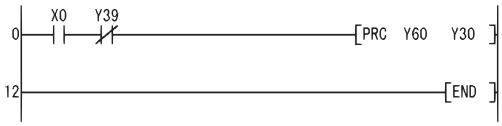
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 在对软元件注释进行 RUN 中写入的过程中，执行了 PRC 指令时。 (出错代码 :4100)

程序示例

(1) 以下为 X0 变为 ON 时，将 Y60 的注释输出到 Y30 ～ Y39 中的程序。

[梯形图模式]

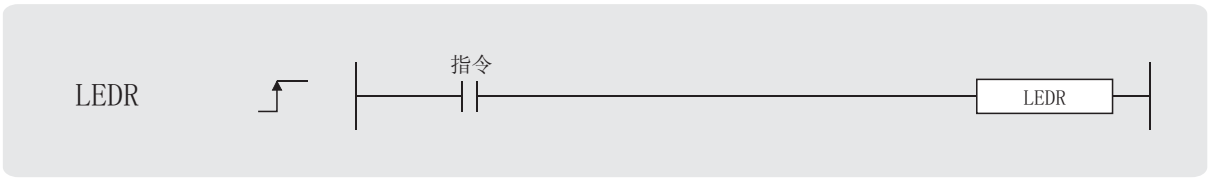


[列表模式]

步	指令	软元件
0	LD	X0
1	ANI	Y39
2	PRC	Y60 Y30
5		<:y60="aa">
12	END	

7.9.3 出错显示或报警器复位指令 (LEDR)

Basic High performance Process Redundant Universal



设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
--	--								

功 能

对 CPU 模块的报警器显示及继续运行型自诊断出错显示进行复位。
在一次指令执行中，只能对出错显示或者报警器二者之一进行复位。

(1) 发生自诊断出错时的动作

(a) 发生了继续运行型自诊断出错时
在 CPU 模块处于继续运行型自诊断出错的出错显示时，如果执行了 LEDR 指令，则只对 CPU 模块前面的“ ERROR/ERR. ”LED 或者出错显示进行复位。此时 SM0、SM1、SD0 的内容不能被复位，因此应由用户通过程序进行复位。
此外，此时出错显示的出错原因的优先顺序高于报警器，因此不能执行报警器的复位的相关处理。

(b) 发生了电池出错时
如果在更换了电池后执行了 LEDR 指令，则对 CPU 模块前面的“ BAT.ARM/BAT. ”LED 或者出错显示进行复位。
此时 SM51 也将变为 OFF。

(2) 报警器 (F) 处于 ON 状态时的动作

(a) 对于前面无 LED 显示器的 CPU 模块

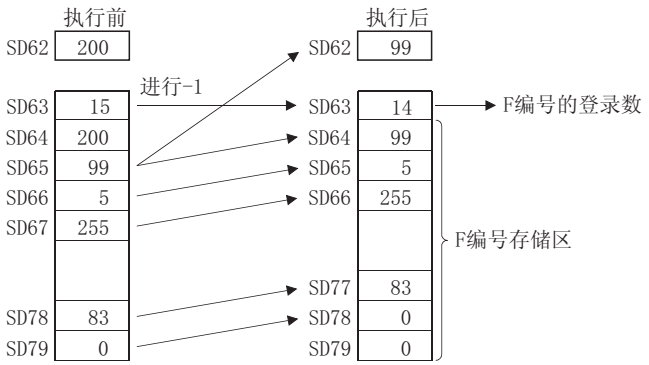
如果执行了 LEDR 指令，将执行以下动作：

1) “ USER ”LED 闪烁、熄灯。

2) 对 SD62、SD64 中存储的报警器 (F) 进行复位后，SD65 ～ SD79 的 F 编号向前填充对齐。

3) 将 SD64 中新存储的数据传送至 SD62 中。

4) 对 SD63 的数据进行 -1。但是 SD63 为 0 时保持为 0 不变。



(b) 对于前面有 LED 显示器的 CPU 模块

如果执行了 LEDR 指令，将执行以下动作：

1) 对 CPU 模块显示的 F 编号进行复位。

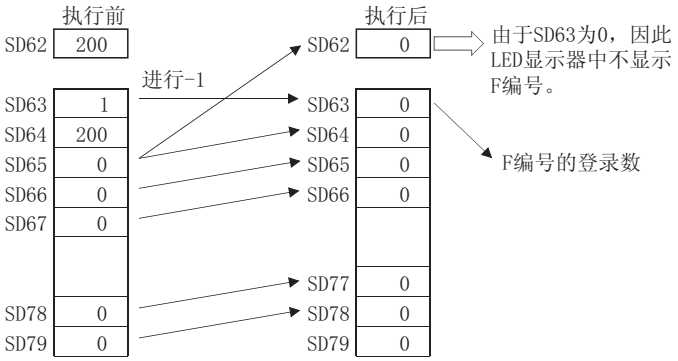
2) “ USER ”LED 闪烁、熄灯。

3) 对 SD62、SD64 中存储的报警器 (F) 进行复位后，SD65 ～ SD79 的 F 编号向前填充对齐。

4) 将 SD64 中新存储的数据传送至 SD62 中。

5) 对 SD63 的数据进行 -1。但是 SD63 为 0 时保持为 0 不变。

6) 在 LED 显示器上显示 SD62 中存储的 F 编号。但是，SD63 为 0 时不显示任何内容。



备 注

1. 特殊寄存器 SD207 ～ SD209 中设置的原因编号的内容及优先顺序的默认值如下所示。

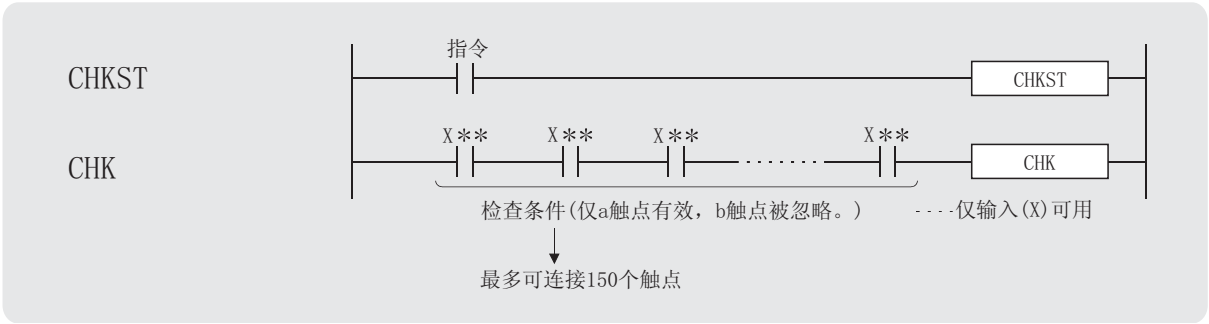
优先顺序	原因编号 (16 进制数)	内容	备注
1	1	AC DOWN SINGLE PS. DOWN SINGLE PS. ERROR	电源断开 冗余基板电源电压过低 冗余电源模块异常
2	2	UNIT VERIFY ERR. FUSE BREAK OFF SP. UNIT ERROR	I/O 模块校验出错 保险丝熔断 特殊功能模块校验出错
3	3	OPERATION ERROR LINK PARA. ERROR SFCP OPE. ERROR SFCP EXE. ERROR	运算出错 链接参数出错 SFC 指令运算出错 SFC 程序执行出错
4	4	ICM. OPE. ERROR FILE OPE. ERROR EXTEND INST. ERROR OPE. MODE DIFF. CAN'T EXE. MODE TRK.TRANS. ERR. TRK.SIZE ERROR TRK.DISCONNECT	存储卡操作出错 文件访问出错 扩展指令出错 运行状态、开关不匹配 当前模式下功能无法执行 热备数据通信出错 热备容量超出出错 热备电缆未连接・故障
5	5	PRG.TIME OVER	恒定扫描设置时间超时 低速执行监视时间超时
6	6	CHK 指令	--
7	7	报警器	--
8	8	LED 指令	--
9	9	BATTERY ERR.	--
10	A	时钟数据	--
11	B	CAN'T SWITCH STANDBY SYS. DOWN MEM. COPY EXE.	系统切换出错 待机系统未启动 / 停止出错 内存拷贝功能实施

2. 如果将报警器设置为最高优先顺序，可通过 LEDR 指令对报警器进行优先复位。

7.10 调试和故障诊断指令

7.10.1 特殊格式故障检查 (CHKST、CHK)

Basic High performance Process Redundant Universal



设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
--	--								

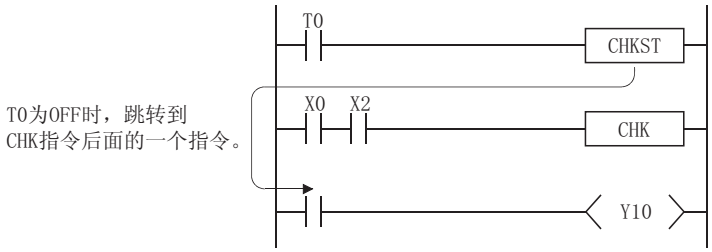
功 能

CHKST

CHKST 指令是启动 CHK 指令的指令。

如果 CHKST 指令的执行指令为 OFF，则跳转到 CHK 指令后面的一个指令。

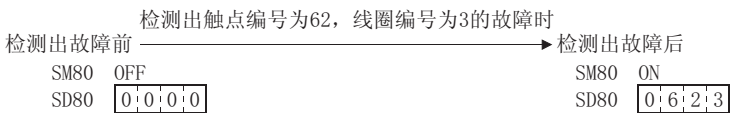
如果 CHKST 指令的执行指令为 ON，则执行 CHK 指令。



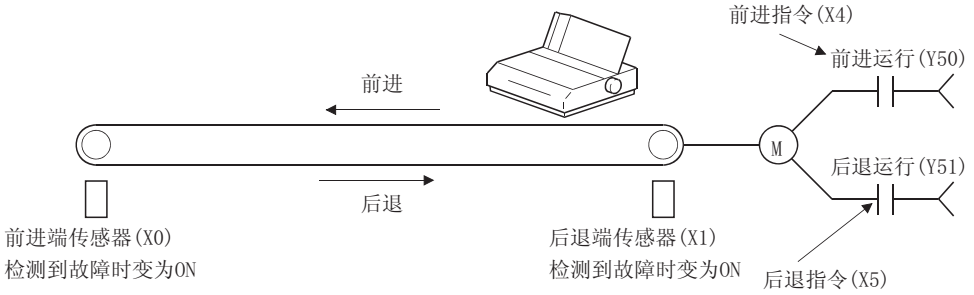
CHK

(1) CHK 指令是用于对下页所示的进行往复运动的系统的故障内容进行确认的指令。

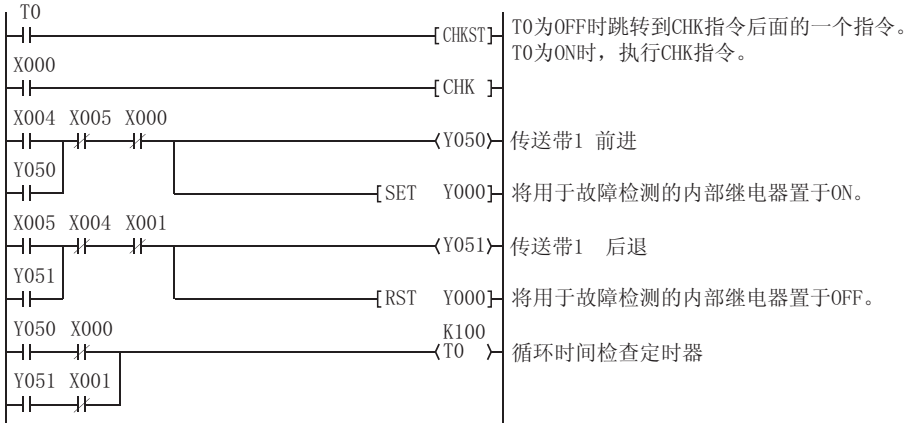
(a) 当执行 CHK 指令时，通过指定的检查条件进行故障诊断检查，如果检测出故障，则 SM80 变为 ON，且以 BCD 值将故障编号存储到 SD80 中。
此外，检测出故障时将发生出错代码为 9010 的出错。
检测出故障的触点编号 (参阅 (3)) 将被存储到 SD80 的高 3 位中，检测出故障的线圈编号 (参见 (2)) 将被存储到 SD80 的低 1 位中。



(b) 位于 CHK 指令之前的触点指令不是用于对 CHK 指令的执行进行控制，而是用于进行检查条件设置。



(c) 对上图的系统进行循环时间超时检查时，创建如下图所示的梯形图。



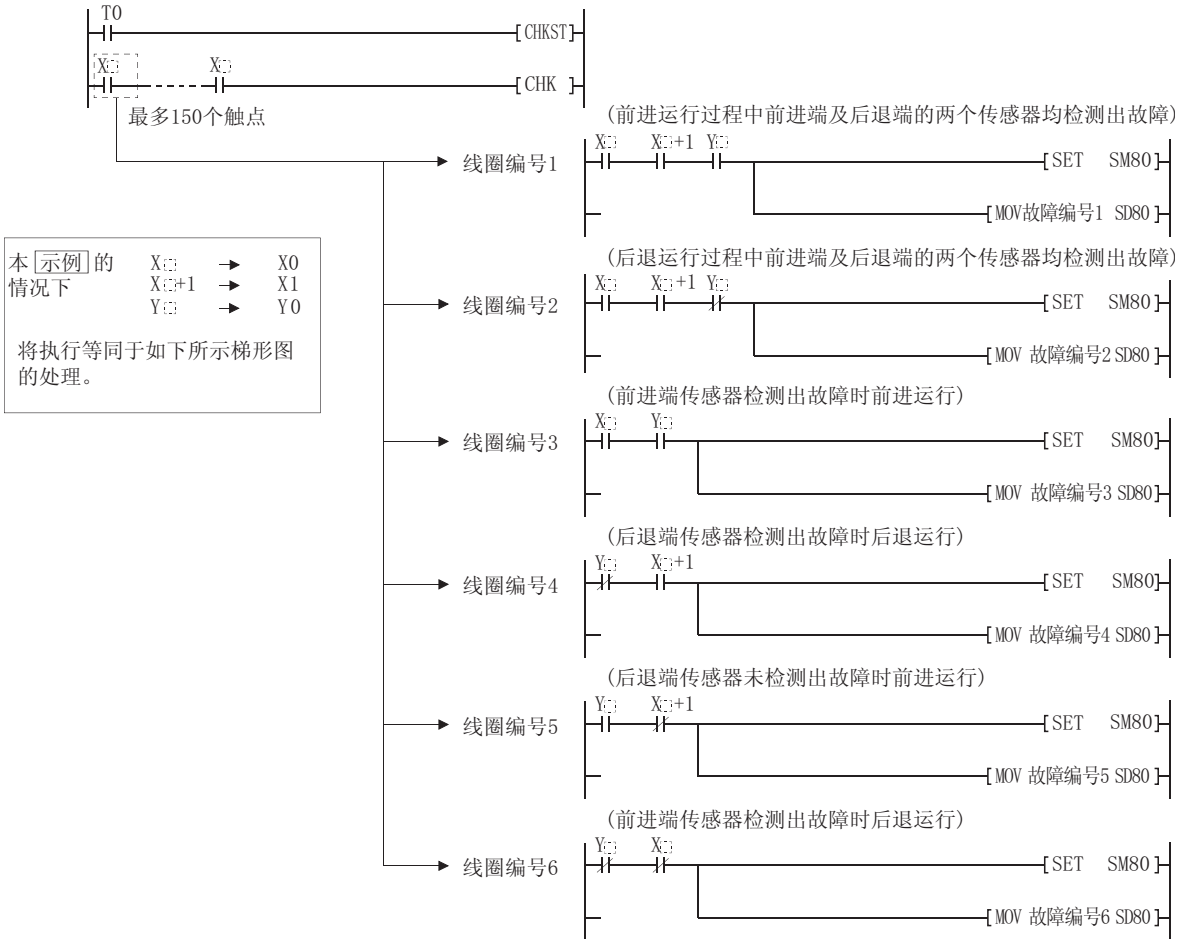
(d) 创建使用 CHK 指令的梯形图时的注意事项如下所示：

1) 前进端传感器与后退端传感器的触点编号 (X□) 必须为连续编号。此外，前进端传感器的触点编号 (X□) 应该小于后退端传感器的触点编号。

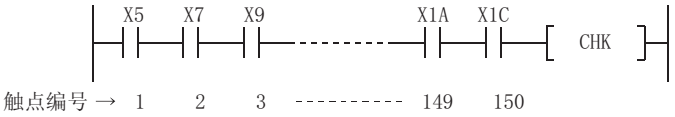
2) 对与前进端传感器触点编号 (X□) 与相同编号的输出 (Y□)*1 应进行如下所示的控制：
进行前进运行时 ..... 变为 ON
进行后退运行时 ..... 变为 OFF

*1: 输出 (Y□) 被作为内部继电器使用，不能对外部进行输出。

(2) CHK 指令根据指定的触点，执行等同于如下所示梯形图的处理。



(3) 检测出故障时的触点编号从左侧的纵母线开始按顺序被分配为 1 ～ 150。



(4) 执行 CHK 指令时，应预先对 SM80、SD80 进行复位。
此外，执行 CHK 指令后，如果未对 SM80、SD80 进行复位，将不能再次执行 CHK 指令。
(SM80、SD80 的内容在用户执行复位之前将被保持。)

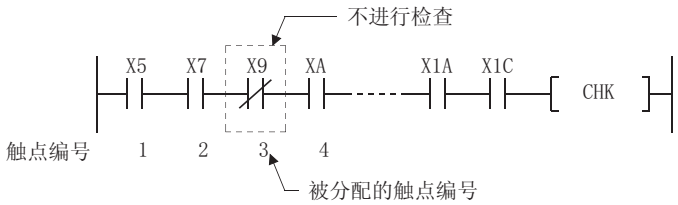
(5) 在 CHK 指令的前面需要设置 CHKST 指令。
在 CHK 指令与 CHKST 指令之间如果存在有除 LD、LDI、AND、ANI 指令以外的指令，将会变为出错状态。 (出错代码： 4235)

(6) CHK 指令可以被写入到程序的任意步中。
但是，使用 CHK 指令的数量有如下限制。

· 在所有正被执行的程序文件中最多可在两个位置使用 CHK 指令。

· 在一个程序文件中只能在一个位置使用 CHK 指令。
如果 CHK 指令的使用超出上述限制，将会变为出错状态。 (出错代码： 4235)

(7) CHK 指令的前面应通过 LD、AND 指令设置检查条件。
不能通过其它触点指令设置检查条件。
通过 LDI、ANI 指令设置了检查条件时，不执行与检查条件相关的处理。
但是，LDI、ANI 指令也将被分配检测出故障时的触点编号。



(8) 故障检测方法根据 SM701 的 ON/OFF 状态而有所不同。

(a) SM701 为 OFF 时，按触点顺序进行线圈编号 1 ～ 6 的检查。
执行 CHK 指令时，对触点编号 1 的线圈编号 1 ～ 6 按编号顺序进行检查。触点编号 1 的线圈编号 6 检查完毕后，对触点编号 2 的线圈编号 1 ～ 6 按编号顺序进行检查。
触点编号 n 的线圈编号 6 检查完毕后，结束 CHK 指令。

(b) SM701 为 ON 时，按线圈顺序进行触点编号 1 ～ n 的检查。
执行 CHK 指令时，对线圈编号 1 的梯形图按触点编号 1 ～ n 的编号顺序进行检查。线圈编号 1 的梯形图的触点编号 n 检查完毕后，对线圈编号 2 的梯形图按触点编号 1 ～ n 的编号顺序进行检查。
线圈编号 6 的触点编号 n 检查完毕后，结束 CHK 指令。

(9) 检测出多个故障时，最先检测出的故障编号将被存储。
后检测出的故障编号将被忽略。

(10) CHK 指令不能用于低速执行型程序。
如果将包含有 CHK 指令的程序文件设置到低速执行型程序中，将变为运算出错状态，CPU 模块运算将停止。

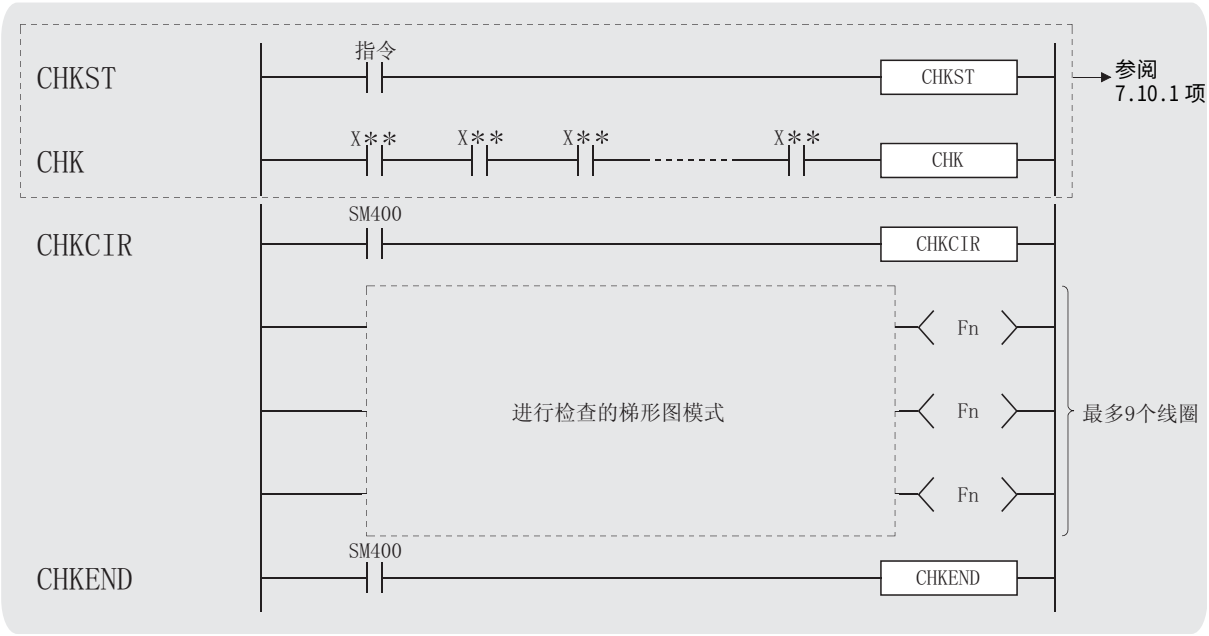
出　错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- 存在有并行梯形图时。 (出错代码： 4235)
- 存在有 NOP 指令时。 (出错代码： 4235)
- 触点指令超过了 150 个时。 (出错代码： 4235)
- 执行 CHKST 指令后，未执行 CHK 指令时。 (出错代码： 4235)
- 在未执行 CHKST 指令的情况下执行了 CHK 指令时。 (出错代码： 4235)
- CHKST 指令、CHK 指令被用于低速执行型程序中时。 (出错代码： 4235)
- 在 CHK 指令与 CHKST 指令之间存在有除 LD、LDI、AND、ANI 指令以外的指令时。 (出错代码： 4235)
- 在处于执行状态的所有程序文件中在 3 个或以上位置处使用了 CHK 指令时。 (出错代码： 4235)
- 在一个程序文件中的 2 个或以上位置处使用了 CHK 指令时。 (出错代码： 4235)

7.10.2 改变 CHK 指令的检查格式 (CHKCIR、CHKEND)

Basic High performance Process Redundant Universal



设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
--	--								

功 能

CHKCIR、CHKEND

(1) 将 CHK 指令中使用的检查梯形图模式更改为任意的格式。
实际的故障检查是通过 CHKST、CHK 指令进行的。

(2) 根据 CHK 指令中指定的检查条件及 CHKCIR ～ CHKEND 指令之间记述的梯形图模式进行故障检查。

备 注

关于 CHKCIR、CHKEND 指令，请参阅 7.10.1 项。

要 点

使用 CHKCIR ～ CHKEND 指令对 CHK 指令的检查格式进行变更时，应由用户创建附加了变址修饰 (Z0) 的梯形图。

(a) 检查条件（下图的 X2、X8）中所示的软元件编号将成为梯形图模式中记述的各软元件编号（报警器 (F) 除外）的变址修饰值。

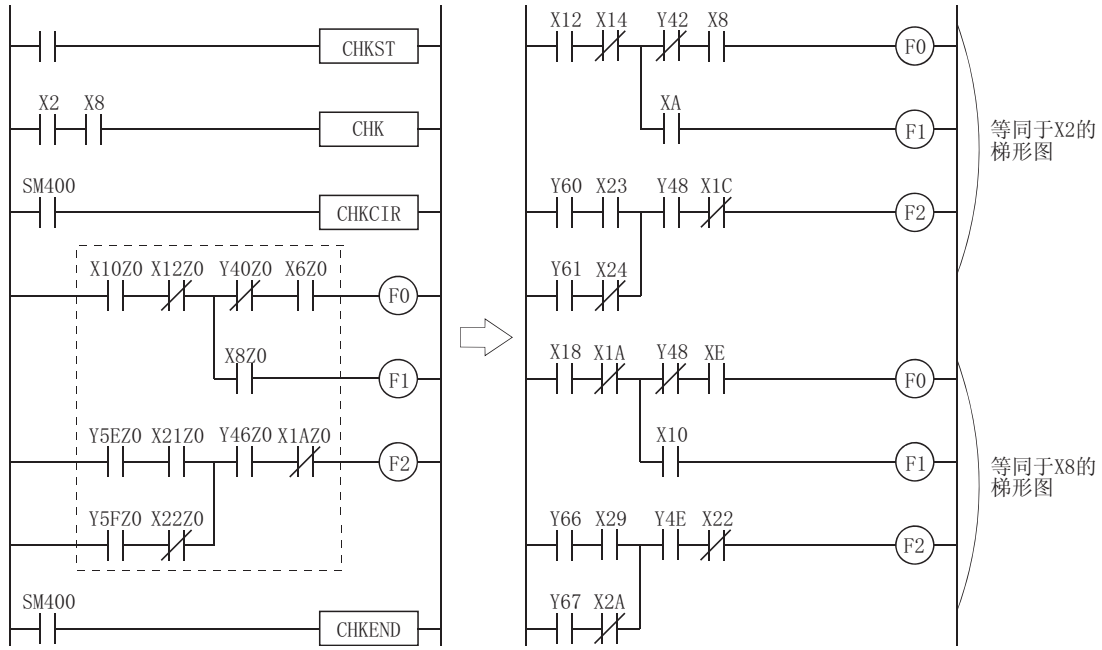
例 下图⬚中的 X10 的情况如下：

对应于检查条件 X2 时 X12
对应于检查条件 X8 时 X18
} 进行处理。

但是，根据 SM701 的 ON/OFF 状态，故障检查的顺序有所不同。

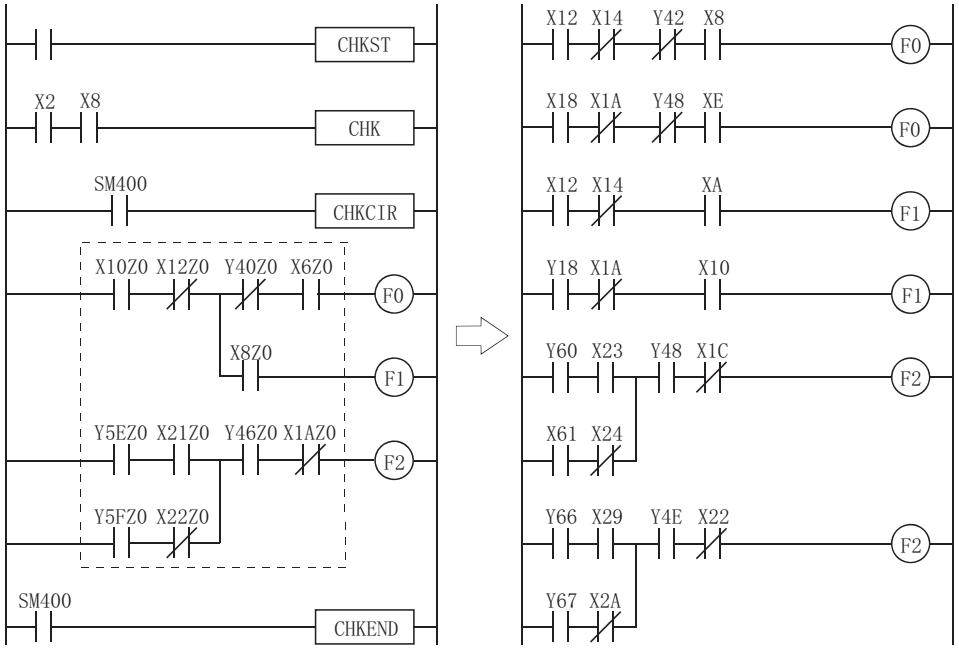
1)SM701 为 OFF 时，按触点顺序从线圈编号 1 开始进行检查。

[CHKCIR ～ CHKEND 指定的梯形图]　　[CPU 模块的检查顺序]



2)SM701 为 ON 时，按线圈顺序从触点编号 1 开始进行检查。

[CHKCIR ～ CHKEND 指定的梯形图]　　[CPU 模块的检查顺序]



(b) 进行故障检查时，根据各检查条件中的梯形图模式，对 OUT F□的 ON/OFF 状态进行检查。
在所有的检查条件中，只要梯形图模式中有一个 OUT F□变为 ON，则 SM80 将变为 ON。
此外，将变为 ON 状态的 OUT F□所对应的出错编号（触点编号及线圈编号）按 BCD 顺序存储到 SD80 中。

(c) 梯形图模式中可使用的指令如下所示：
触点 .....LD、LDI、AND、ANI、OR、ORI、ANB、ORB、MPS、MPP、MRD 比较运算指令
线圈 .....OUT F□

(d) 梯形图模式的触点中可使用的软元件如下所示：
输入 (X)、输出 (Y)

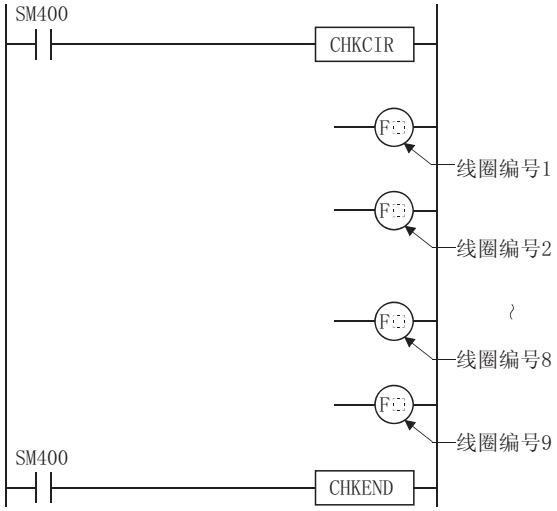
(e) 梯形图模式的线圈中可使用的软元件仅为报警器 (F)。
但是，由于报警器 (F) 为虚拟部件，因此可以设置任意的值。
此外，即使重复也没有关系。

(f) 即使将与 CHK 指令中使用的报警器 (F) 具有相同编号的报警器 (F) 用于除 CHK 指令以外的其它位置，也可正常地进行 ON/OFF 控制。在 CHK 指令中及除 CHK 指令以外的其它位置将分别进行处理。

(g) 由于 CHK 指令中使用的报警器 (F) 不进行实际的 ON/OFF，因此即使通过外围设备进行监控也不进行 ON/OFF。

(h) 可创建最多为 256 步的梯形图模式。
此外，OUT F□最多可使用 9 个线圈。

(3) CHKCIR ～ CHKEND 中指定的梯形图的线圈编号按从上至下的顺序被分配为 1 ～ 9 号。



(4) CHKCIR、CHKEND 指令可被写入到程序中的任意步中。
在所有处于执行状态的程序文件中最多可在 2 个位置处使用。
但是，在一个程序文件中只能在一个位置处使用 CHK 指令。

(5) CHKCIR、CHKEND 指令不能用于低速执行型程序。
将记述了 CHKCIR、CHKEND 指令的程序文件设置到低速执行型程序中时，将变为运算出错状态。此外，在高性能型 QCPU/ 过程 CPU/ 冗余 CPU 中将停止运算。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

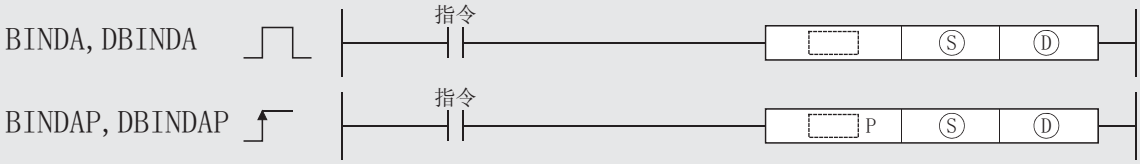
- 在所有程序文件中在 3 个或以上位置处使用了 CHKCIR ～ CHKEND 指令时。 (出错代码： 4235)
- 在 1 个程序文件中的 2 个或以上位置处使用了 CHKCIR ～ CHKEND 指令时。 (出错代码： 4235)
- 执行了 CHKCIR 指令后，未执行 CHKEND 指令时。 (出错代码： 4230)
- 在未执行 CHKCIR 指令的情况下，使用了 CHKEND 指令时。 (出错代码： 4230)
- 将 CHKCIR、CHKEND 指令用于低速执行型程序中时。 (出错代码： 4235)
- 梯形图模式中存在有 10 个或以上的 F 时。 (出错代码： 4235)
- 梯形图模式为 257 步或以上时。 (出错代码： 4235)
- 存在有梯形图模式中不能使用的软元件时。 (出错代码： 4235)
- 对梯形图模式的软元件进行了变址修饰时。 (出错代码： 4235)

7.11 字符串处理指令

7.11.1 BIN16 位 /32 位→ 10 进制 ASCII 码的转换 (BINDA(P)、DBINDA(P))

Basic | High performance | Process | Redundant | Universal

□表示BINDA、DBINDA等指令符号。



Ⓢ : 进行 ASCII 码转换的 BIN 数据 (BIN16/32 位)。

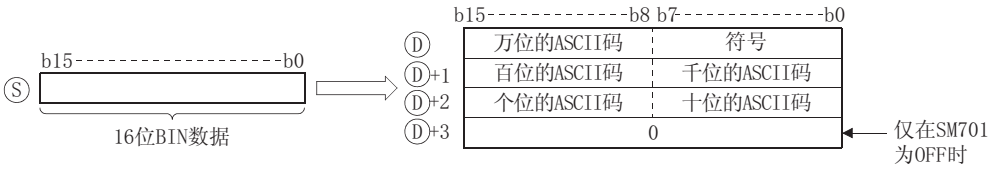
Ⓓ : 存储转换结果的软元件的起始编号 (字符串)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○	○		○					--
Ⓓ	--	○		--					--

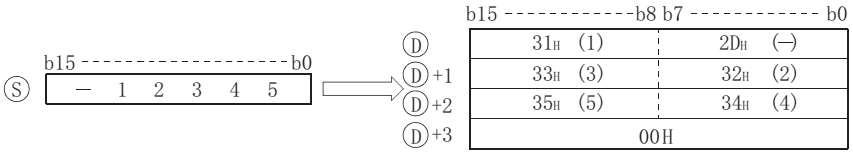
功 能

BINDA

(1) 将Ⓢ中指定的 BIN16 位数据以 10 进制数表示时的各个位数值转换为 ASCII 码后，存储到Ⓓ中指定的软元件编号的后面。



例如，在Ⓢ中指定了 -12345 时，按以下方式存储到Ⓓ的后面。

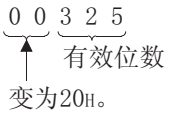


(2) 在Ⓢ中可指定的范围为 -32768 ～ 32767。

(3) Ⓓ中存储的运算结果如下所示。

(a) BIN 数据为正数时在 “ 符号 ” 中存储 “ 20H ”，为负数时在 “ 符号 ” 中存储 “ 2DH ”。

(b) 在有效位数的左侧 “ 0 ” 中存储 “ 20H ”。(进行 0 删除。)

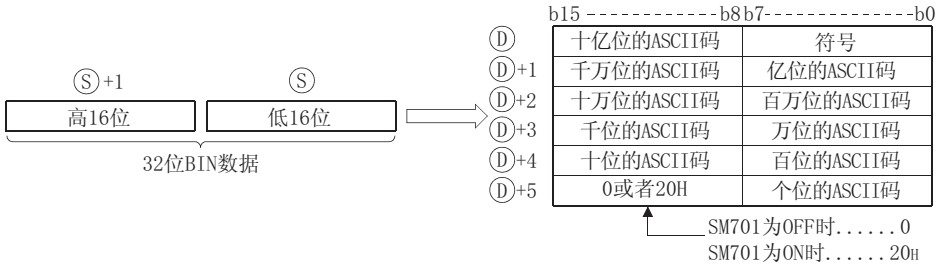


(c) 至Ⓓ+3 中指定的软元件的数据存储根据 SM701(输出字符数切换信号) 的 ON/OFF 状态而有所不同。

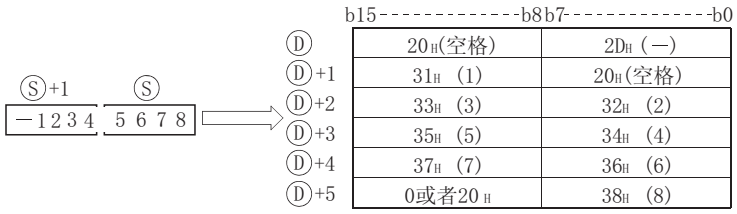
SM701 为 OFF 时 ... 存储 “ 0 ”。
SM701 为 ON 时 .... 无变化。

DBINDA

(1) 将Ⓢ中指定的 BIN32 位数据以 10 进制数表示时的各个位数值转换为 ASCII 码后，存储到Ⓓ中指定的软元件编号的后面。



例如，在Ⓢ中指定了 -12345678 时，按以下方式存储到Ⓓ的后面。

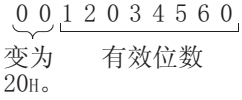


(2) 在Ⓢ中可指定的 BIN 数据的范围为 -2147483648 ～ 2147483647。

(3) Ⓓ中存储的结果如下所示。

(a) BIN 数据为正数时在 “ 符号 ” 中存储 “ 20H ”，为负数时在 “ 符号 ” 中存储 “ 2DH ”。

(b) 在有效位数的左侧 “ 0 ” 中存储 “ 20H ”。(进行 0 删除。)



(c) Ⓓ+5 中指定的软元件的高 8 位中存储的数据根据 SM701(输出字符数切换信号) 的 ON/OFF 状态而有所不同。

SM701 为 OFF 时 ... 存储 “ 0 ”。
SM701 为 ON 时 .... 存储 “ 20H ”。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

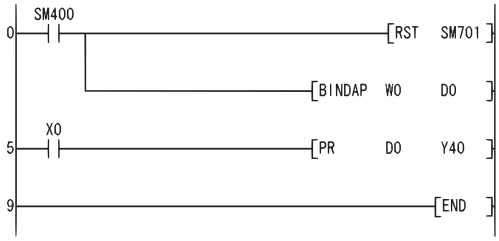
· ⓓ中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)

(出错代码： 4101)

程序示例

(1) 以下为将 16 位 BIN 数据的 W0 值通过 PR 指令转换为 10 进制数的 ASCII 码后输出到 Y40 ～ Y48 中的程序。

[梯形图模式]



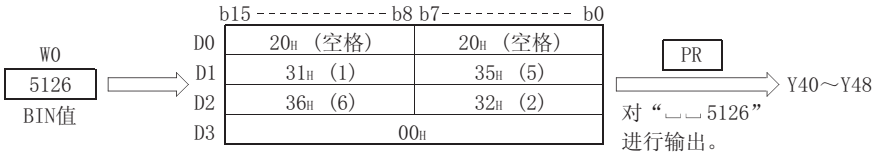
[列表模式]

步	指令	软元件	
0	LD	SM400	
1	RST	SM701	
2	BINDAP	W0	D0
5	LD	X0	
6	PR	D0	Y40
9	END		

[动作]

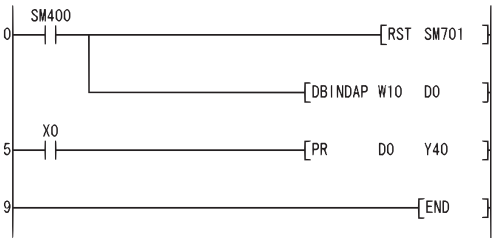
X0 变为 ON 时，通过 PR 指令对 Y40 ～ Y48 进行 ASCII 码输出。

由于 SM701 处于 OFF 状态，因此 PR 指令执行输出直至 ASCII 码 00H 为止。



(2) 以下为将 32 位 BIN 数据 W10、W11 的值通过 PR 指令转换为 10 进制数的 ASCII 码后输出到 Y40 ～ Y48 中的程序。

[梯形图模式]



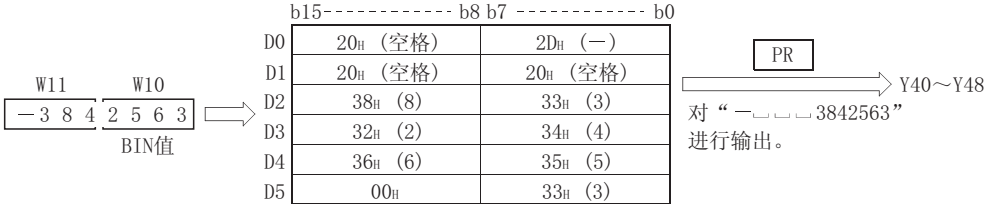
[列表模式]

步	指令	软元件	
0	LD	SM400	
1	RST	SM701	
2	DBINDAP	W10	D0
5	LD	X0	
6	PR	D0	Y40
9	END		

[动作]

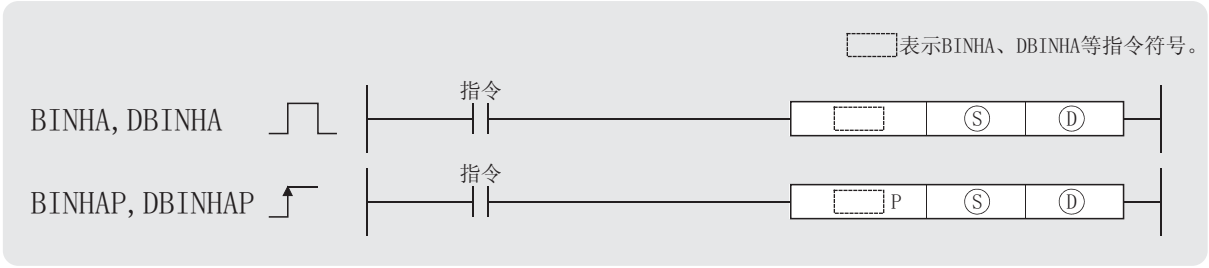
X0 变为 ON 时，通过 PR 指令对 Y40 ～ Y48 进行 ASCII 码输出。

由于 SM701 处于 OFF 状态，因此 PR 指令执行输出直至 ASCII 码 00H 为止。



7.11.2 BIN16 位 /32 位数据→ 16 进制 ASCII 码的转换 (BINHA(P)、DBINHA(P))

Basic High performance Process Redundant Universal



Ⓢ : 进行 ASCII 码转换的 BIN 数据 (BIN16/32 位)。

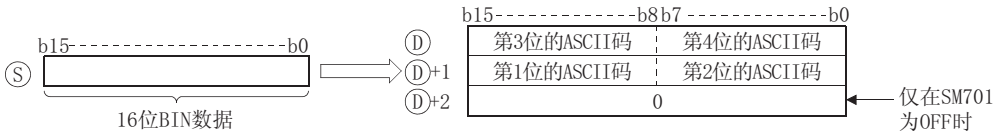
Ⓓ : 存储转换结果的软元件的起始编号 (字符串)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○	○		○					--
Ⓓ	--	○		--					--

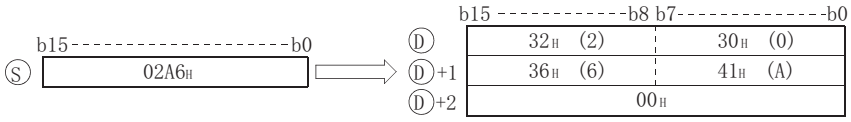
功 能

BINHA

(1) 将Ⓢ中指定的 BIN16 位数据以 16 进制数表示时的各个位数值转换为 ASCII 码后，存储到Ⓓ中指定的软元件编号的后面。



例如，在Ⓢ中指定了 02A6H 时，按以下方式存储到Ⓓ的后面。



(2) 在Ⓢ中可指定的范围为 0H ～ FFFFH。

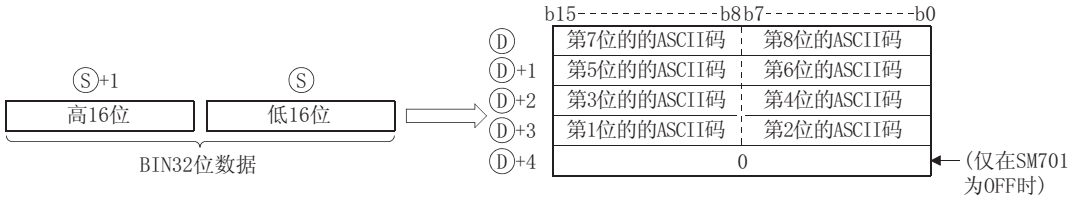
(3) Ⓓ中存储的运算结果被处理为 4 位数的 16 进制数。
因此，有效位数的左侧的 “ 0 ” 将被处理为 “ 0 ”。(不进行 0 删除。)

(4) 至Ⓓ+2 中指定的软元件的数据存储根据 SM701(输出字符数切换信号) 的 ON/OFF 状态而有所不同。

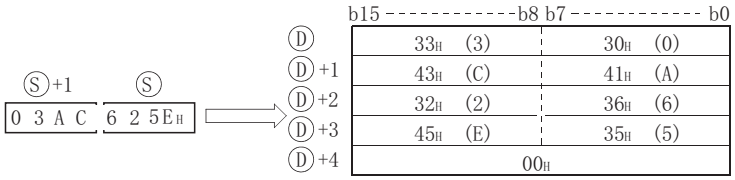
SM701 为 OFF 时 .... 存储 “ 0 ”。
SM701 为 ON 时 ..... 无变化。

DBINHA

(1) 将Ⓢ中指定的 BIN32 位数据以 16 进制数表示时的各个位数值转换为 ASCII 码后，存储到Ⓓ中指定的软元件编号的后面。



例如，在Ⓢ中指定了 03AC625EH 时，按以下方式存储到Ⓓ的后面。



(2) 在Ⓢ中可指定的范围为 0H ～ FFFFFFFFH。

(3) Ⓓ中存储的运算结果被处理为 8 位数的 16 进制数。
因此，有效位数的左侧的“ 0 ”将被处理为“ 0 ”。(不进行 0 删除。)

(4) 至Ⓓ+2 中指定的软元件的数据存储根据 SM701(输出字符数切换信号) 的 ON/OFF 状态而有所不同。

SM701 为 OFF 时 .... 存储“ 0 ”。
SM701 为 ON 时 ..... 无变化。

出 错

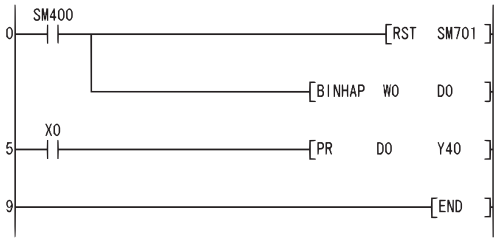
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· Ⓓ中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)
(出错代码： 4101)

程序示例

(1) 以下为将 16 位 BIN 数据的 W0 值通过 PR 指令转换为 16 进制数的 ASCII 码后输出到 Y40 ～ Y48 中的程序。

[梯形图模式]

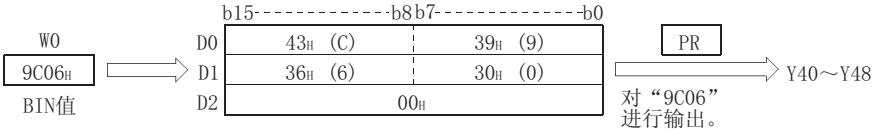


[列表模式]

步	指令	软元件	
0	LD	SM400	
1	RST	SM701	
2	BINHAP	W0	D0
5	LD	X0	
6	PR	D0	Y40
9	END		

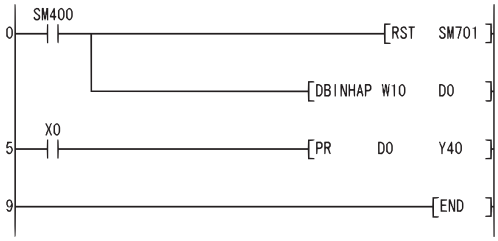
[动作]

X0 变为 ON 时，通过 PR 指令对 Y40 ～ Y48 进行 ASCII 码输出。
由于 SM701 处于 OFF 状态，因此 PR 指令执行输出直至 ASCII 码 00H 为止。



(2) 以下为将 32 位 BIN 数据 W10、W11 的值通过 PR 指令转换为 16 进制数的 ASCII 码后输出到 Y40 ～ Y48 中的程序。

[梯形图模式]

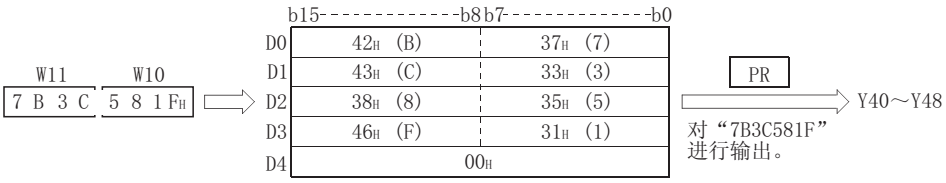


[列表模式]

步	指令	软元件	
0	LD	SM400	
1	RST	SM701	
2	DBINHAP	W10	D0
5	LD	X0	
6	PR	D0	Y40
9	END		

[动作]

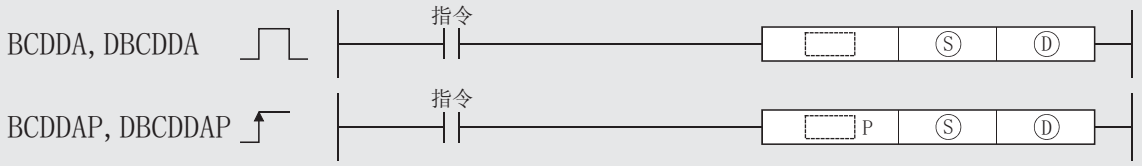
X0 变为 ON 时，通过 PR 指令对 Y40 ～ Y48 进行 ASCII 码输出。
由于 SM701 处于 OFF 状态，因此 PR 指令执行输出直至 ASCII 码 00H 为止。



7.11.3 BCD4 位 /8 位数据→ 10 进制 ASCII 码的转换 (BCDDA(P)、DBCDDA(P))

Basic | High performance | Process | Redundant | Universal

表示BCDDA、DBCDDA等指令符号。



Ⓢ : 进行 ASCII 码转换的 BCD 数据 (BCD4 位 /8 位)。

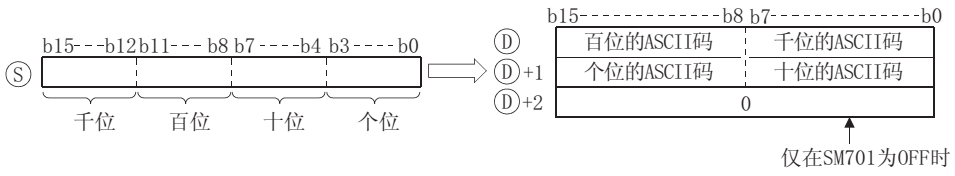
Ⓓ : 存储转换结果的软元件的起始编号 (字符串)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○	○		○					--
Ⓓ	--	○		--					--

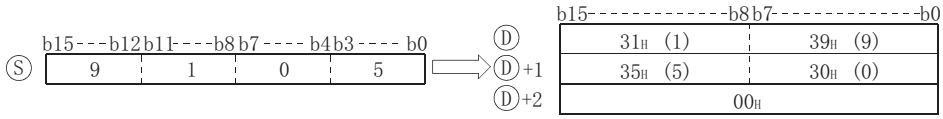
功 能

BCDDA

(1) 将Ⓢ中指定的 BCD4 位数据的各个位数值转换为 ASCII 码后，存储到Ⓓ中指定的软元件编号的后面

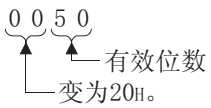


例如，在Ⓢ中指定了 9105 时，按以下方式存储到Ⓓ的后面。



(2) 在Ⓢ中可指定的 BCD 数据范围为 0 ～ 9999。

(3) 在Ⓓ中存储的运算结果中，有效位数的左侧的 “ 0 ” 将被进行 0 删除。

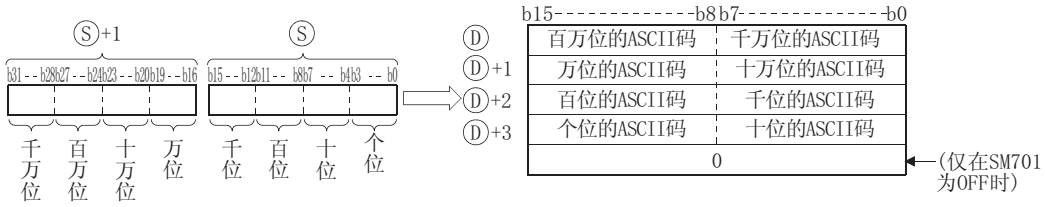


(4) 至Ⓓ+2 中指定的软元件的数据存储根据 SM701(输出字符数切换信号) 的 ON/OFF 状态而有所不同。

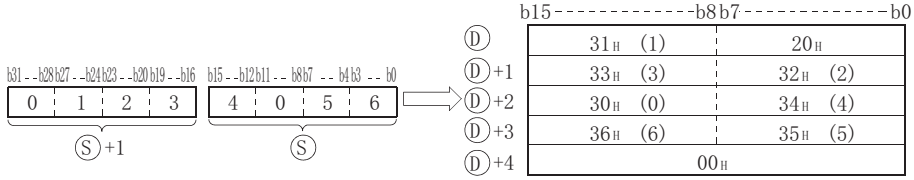
SM701 为 OFF 时 ...存储 “ 0 ”。
SM701 为 ON 时 ....无变化。

DBCDDA

(1) 将Ⓢ中指定的 BCD8 位数据的各个位数值转换为 ASCII 码后，存储到Ⓓ中指定的软元件编号的后面。

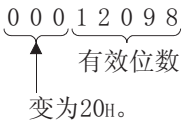


例如，在Ⓢ中指定了 01234056 时，按以下方式存储到Ⓓ的后面。



(2) 在Ⓢ中可指定的 BCD 数据范围为 0 ～ 99999999。

(3) 在Ⓓ中存储的运算结果中，有效位数的左侧的 “ 0 ” 将被进行 0 删除。



(4) 至Ⓓ+4 中指定的软元件的数据存储根据 SM701(输出字符数切换信号) 的 ON/OFF 状态而有所不同。

SM701 为 OFF 时 ... 存储 “ 0 ”。
SM701 为 ON 时 .... 无变化。

出 错

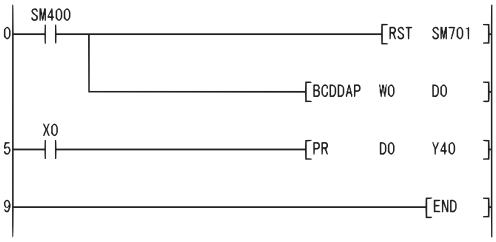
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- 使用 BCDDA 指令时，Ⓢ的数据超出了 0 ～ 9999 的范围时。 (出错代码： 4100)
- 使用 DBCDDA 指令时，Ⓢ的数据超出了 0 ～ 99999999 的范围时。 (出错代码： 4100)
- Ⓓ中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)
  (出错代码： 4101)

程序示例

(1) 以下为将 BCD4 位数据 (W0 值) 通过 PR 指令转换为 10 进制数的 ASCII 码后输出到 Y40 ～ Y48 中的程序。

[梯形图模式]



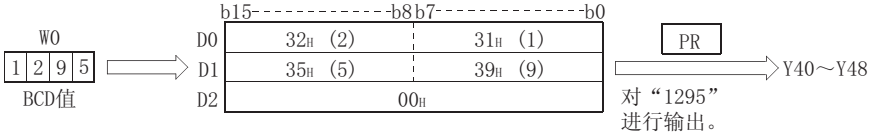
[列表模式]

步	指令	软元件	
0	LD	SM400	
1	RST	SM701	
2	BCDDAP	W0	D0
5	LD	X0	
6	PR	D0	Y40
9	END		

[动作]

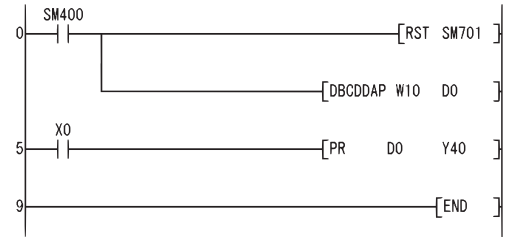
X0 变为 ON 时，通过 PR 指令对 Y40 ～ Y48 进行 ASCII 码输出。

由于 SM701 处于 OFF 状态，因此 PR 指令执行输出直至 ASCII 码 00H 为止。



(2) 以下为将 BCD8 位数据 (W10、W11 的值) 通过 PR 指令转换为 10 进制数的 ASCII 码后输出到 Y40 ～ Y48 中的程序。

[梯形图模式]



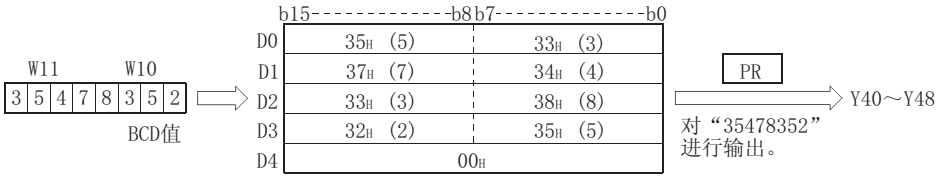
[列表模式]

步	指令	软元件	
0	LD	SM400	
1	RST	SM701	
2	DBCDDAP	W10	D0
5	LD	X0	
6	PR	D0	Y40
9	END		

[动作]

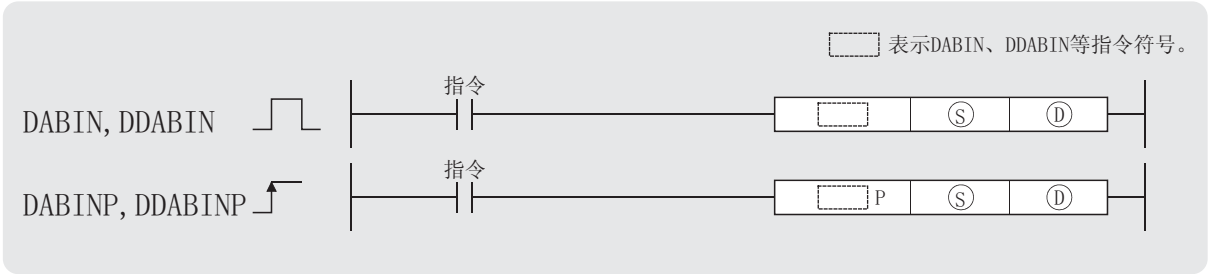
X0 变为 ON 时，通过 PR 指令对 Y40 ～ Y48 进行 ASCII 码输出。

由于 SM701 处于 OFF 状态，因此 PR 指令执行输出直至 ASCII 码 00H 为止。



7.11.4 10 进制 ASCII 码→ BIN16 位 /32 位数据的转换 (DABIN(P)、DDABIN(P))

Basic High performance Process Redundant Universal



Ⓢ : 进行 BIN 值转换的 ASCII 码数据或者存储 ASCII 码数据的软元件的起始编号 (字符串)。

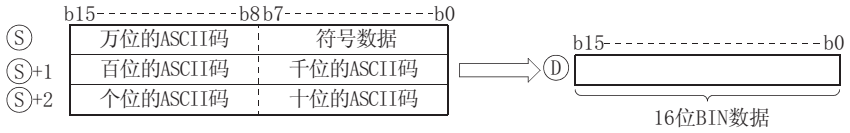
Ⓓ : 存储转换结果的软元件的起始编号 (BIN16/32 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
Ⓢ	--	○		--				○	--
Ⓓ	○	○		○				--	--

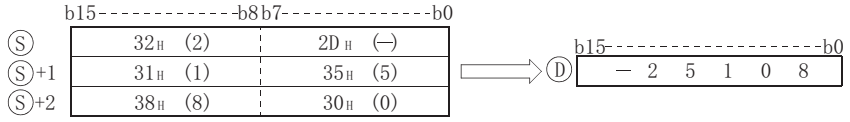
功 能

DABIN

(1) 将存储在Ⓢ中指定的软元件编号后面的 10 进制 ASCII 数据转换为 BIN16 位数据后，存储到Ⓓ中指定编号的软元件中。



例如，Ⓢ后面指定了 -25108H 的 ASCII 码时，按以下方式存储到Ⓓ中。



(2) 在Ⓢ～Ⓢ+2 中可指定的 ASCI 数据的范围为 -32768 ～ 32767。

(3) 转换的数据为正数时在 “ 符号 ” 数据中设置 “ 20H ”，为负数时在 “ 符号 ” 数据中设置 “ 2DH ”。
(设置了除 “ 20H ”、 “ 2DH ” 以外时，将作为正的数据处理。)

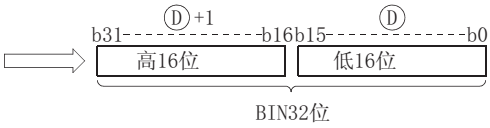
(4) 各位数中可设置的 ASCII 码的范围为 “ 30H ” ～ “ 39H ”。

(5) 各位数中设置的 ASCII 码为 “ 20H ”、 “ 00H ” 时，将作为 “ 30H ” 处理。

DDABIN

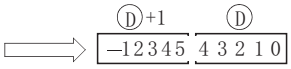
(1) 将存储在Ⓢ中指定的软元件编号后面的 10 进制 ASCII 数据转换为 BIN32 位数据后，存储到Ⓓ中指定编号的软元件中。

	b15----------b8	b7----------b0
Ⓢ	十亿位的ASCII码	符号数据
Ⓢ+1	千万位的ASCII码	亿位的ASCII码
Ⓢ+2	十万位的ASCII码	百万位的ASCII码
Ⓢ+3	千位的ASCII码	万位的ASCII码
Ⓢ+4	十位的ASCII码	百位的ASCII码
Ⓢ+5	(忽略)	个位的ASCII码



例如，Ⓢ后面指定了 -1234543210H 的 ASCII 码时，按以下方式存储到Ⓓ+1、Ⓓ中。

	b15----------b8	b7----------b0
Ⓢ	31H (1)	2DH (−)
Ⓢ+1	33H (3)	32H (2)
Ⓢ+2	35H (5)	34H (4)
Ⓢ+3	33H (3)	34H (4)
Ⓢ+4	31H (1)	32H (2)
Ⓢ+5		30H (0)



(2) 在Ⓢ～Ⓢ+5 中可指定的 ASCI 数据的范围为 -2147483648 ～ 2147483647。

此外，Ⓢ+5 的高位字节中存储的数据将被忽略。

(3) 转换的数据为正数时在 “ 符号 ” 数据中设置 “ 20H ”，为负数时在 “ 符号 ” 数据中设置 “ 2DH ”。

(设置了除 “ 20H ”、 “ 2DH ” 以外时，将作为正的数据处理。)

(4) 各位中可设置的 ASCII 码的范围为 “ 30H ” ～ “ 39H ”。

(5) 各位中设置的 ASCII 码为 “ 20H ”、 “ 00H ” 时，将作为 “ 30H ” 处理。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 在Ⓢ～Ⓢ+5 中指定的 ASCI 码为 “ 30H ” ～ “ 39H ”、“ 20H ”、“ 00H ” 以外时。
(出错代码： 4100)

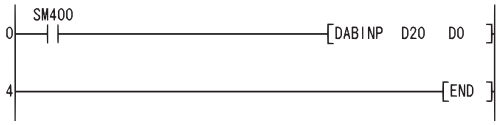
· 在Ⓢ～Ⓢ+5 中指定的 ASCI 数据超出了以下范围时。 (出错代码： 4100)
使用 DABIN 指令时 . . . . . -32768 ～ 32767
使用 DDABIN 指令时 . . . . . -2147483648 ～ 2147483647

· Ⓢ中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)
(出错代码： 4101)

程序示例

(1) 以下为将 D20 ～ D22 中设置的符号及 10 进制 5 位数的 ASCII 数据转换为 BIN 值后，存储到 D0 中的程序。

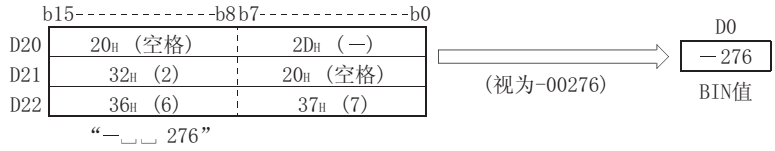
[梯形图模式]



[列表模式]

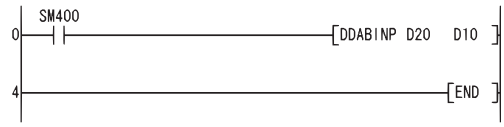
步	指令	软元件	
0	LD	SM400	
1	DABINP	D20	D0
4	END		

[动作]



(2) 以下为将 D20 ～ D25 中设置的符号及 10 进制 10 位数的 ASCII 数据转换为 BIN 值后，存储到 D10、D11 中的程序。

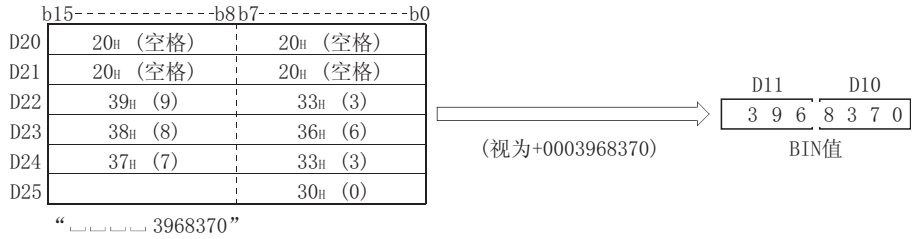
[梯形图模式]



[列表模式]

步	指令	软元件	
0	LD	SM400	
1	DDABINP	D20	D10
4	END		

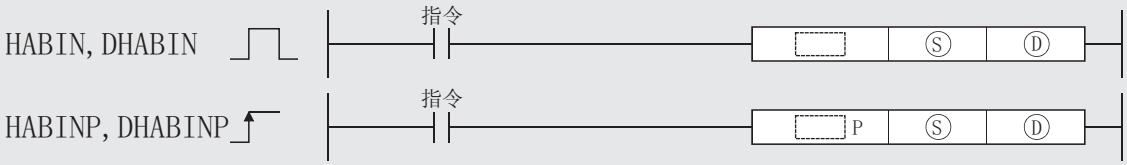
[动作]



7.11.5 16 进制 ASCII → BIN16 位 /32 位数据的转换 (HABIN(P)、DHABIN(P))

Basic High performance Process Redundant Universal

表示HABIN、DHABIN等指令符号。



Ⓢ ：转换为 BIN 值的 ASCII 数据或者存储 ASCII 数据的软元件的起始编号 (字符串)。

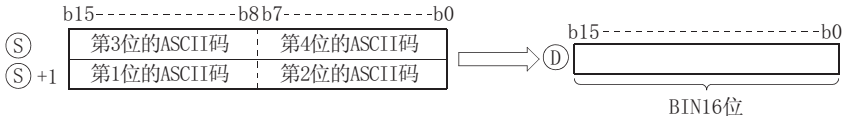
Ⓓ ：存储转换结果的软元件的起始编号 (BIN16/32 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
Ⓢ	--	○		--				○	--
Ⓓ	○	○		○				--	--

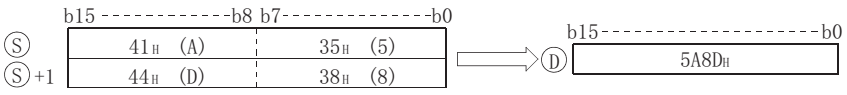
功 能

HABIN

(1) 将Ⓢ中指定的软元件编号后面存储 16 进制 ASCII 数据转换为 BIN16 位数据后，存储到Ⓓ中指定编号的软元件中。



例如，在Ⓢ后面指定了 5A8DH 的 ASCII 码时，按以下方式存储到Ⓓ中。

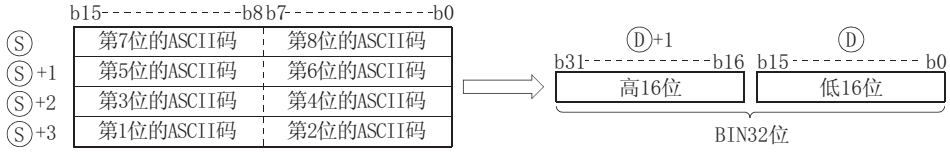


(2) 在Ⓢ～Ⓢ+1 中可指定的 ASCII 数据范围为 0000H ～ FFFFH。

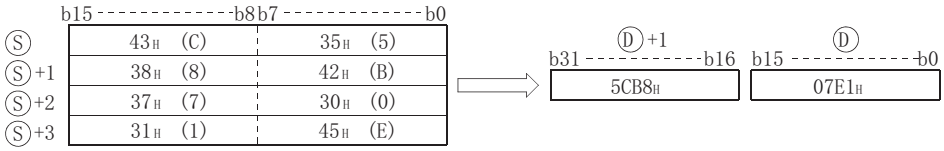
(3) 各位中可设置的 ASCII 码的范围为 “ 30H ” ～ “ 39H ” 及 “ 41H ” ～ “ 46H ”。

DHABIN

(1) 将Ⓢ中指定的软元件编号后面存储 16 进制 ASCII 数据转换为 BIN32 位数据后，存储到Ⓓ中指定编号的软元



例如，在Ⓢ后面指定了 5CB807E1H 的 ASCII 码时，按以下方式存储到Ⓓ+1、Ⓓ中。



(2) 在Ⓢ～Ⓢ+3 中可指定的 ASCII 数据范围为 00000000H ～ FFFFFFFFH。

(3) 各位中可设置的 ASCII 码的范围为“ 30H ”～“ 39H ”及“ 41H ”～“ 46H ”。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

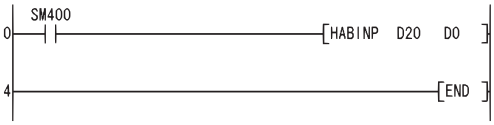
· 在Ⓢ～Ⓢ+3 中指定的各位的 ASCI 码为“ 30H ”～“ 39H ”、“ 41H ”～“ 46H ”以外时。
(出错代码： 4100)

· Ⓢ中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)
(出错代码： 4101)

程序示例

(1) 以下为将 D20、D21 中设置的 16 进制 4 位数的 ASCII 码转换为 BIN 值后，输出到 D0 中的程序。

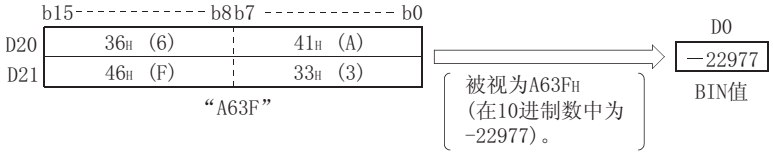
[梯形图模式]



[列表模式]

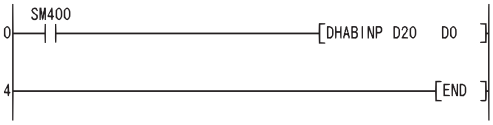
步	指令	软元件	
0	LD	SM400	
1	HABINP	D20	D0
4	END		

[动作]



(2) 以下为将 D20 ～ D23 中设置的 16 进制 8 位数的 ASCII 码转换为 BIN 值后，输出到 D10、D11 中的程序。

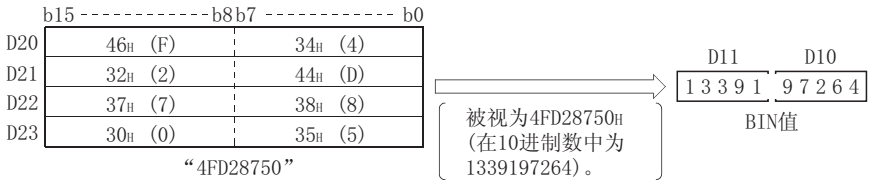
[梯形图模式]



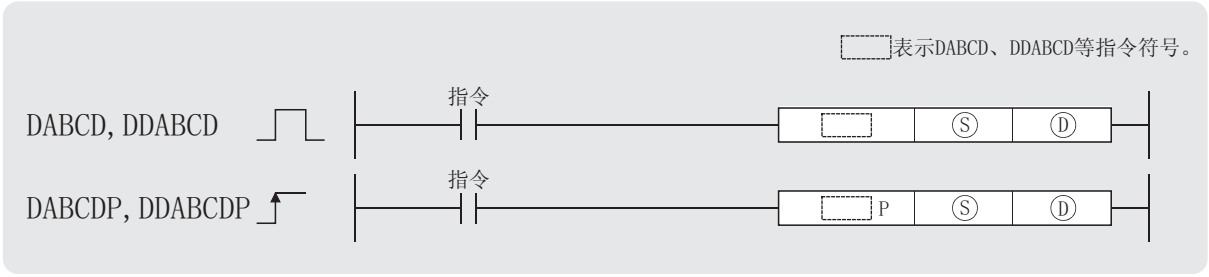
[列表模式]

步	指令	软元件	
0	LD	SM400	
1	DHABINP	D20	D0
4	END		

[动作]



7.11.6 10 进制 ASCII 码→ BCD4 位 /8 位数据的转换 (DABCD(P)、DDABCD(P))



Ⓢ ：转换为 BCD 值的 ASCII 数据或者存储 ASCII 数据的软元件的起始编号 (字符串)。

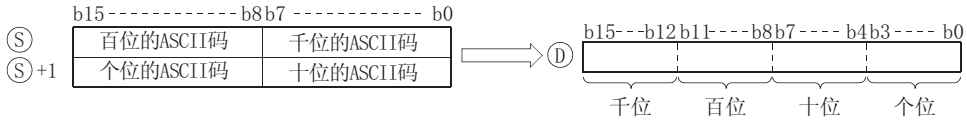
Ⓓ ：存储转换结果的软元件的起始编号 (BCD4 位 /8 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
Ⓢ	--	○		--				○	--
Ⓓ	○	○		○				--	--

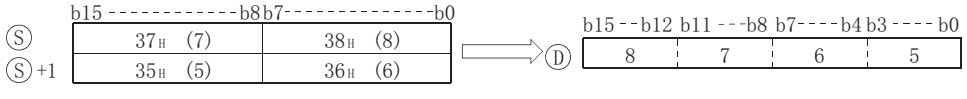
功 能

DABCD

(1) 将Ⓢ中指定的软元件编号后面存储的 10 进制 ASCII 数据转换为 BCD4 位数数据后，存储到Ⓓ中指定编号的软元件中。



例如，在Ⓢ后面指定了 8765H 的 ASCII 码时，按以下方式存储到Ⓓ中。



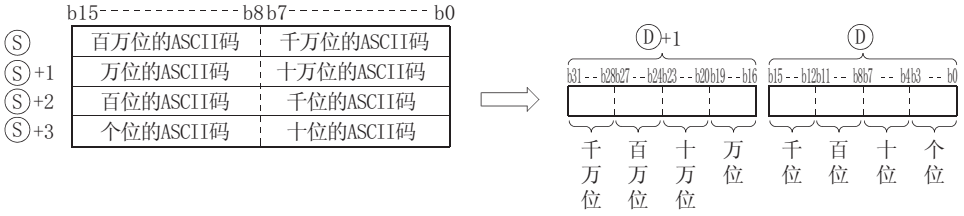
(2) 在Ⓢ～Ⓢ+1 中可指定的 ASCII 数据范围为 0 ～ 9999。

(3) 在各位数中可设置的 ASCII 码的范围为 “ 30H ” ～ “ 39H ”。

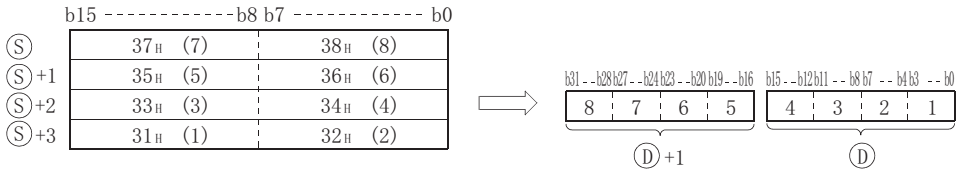
(4) 在各位数中设置的 ASCII 码为 “ 20H ”、 “ 00H ” 时，将作为 “ 30H ” 处理。

DDABCD

(1) 将Ⓢ中指定的软元件编号后面存储的 10 进制 ASCII 数据转换为 BCD8 位数数据后，存储到Ⓓ中指定编号的软元件后面。



例如，在Ⓢ后面指定了 87654321H 的 ASCII 码时，按以下方式存储到Ⓓ+1、Ⓓ中。



(2) 在Ⓢ～Ⓢ+3 中可指定的 ASCII 数据范围为 0 ～ 99999999。

(3) 在各位数中可设置的 ASCII 码的范围为 “ 30H ” ～ “ 39H ”。

(4) 在各位数中设置的 ASCII 码为 “ 20H ” ～ “ 00H ” 时，将作为 “ 30H ” 处理。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

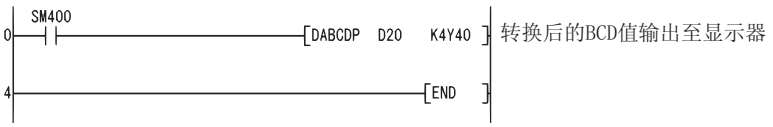
· Ⓢ的数据中存在有除 “ 0 ” ～ “ 9 ” 以外的字符时。 (出错代码： 4100)

· Ⓢ中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)
(出错代码： 4101)

程序示例

(1) 以下为将 D20 ～ D22 中设置的 10 进制数的 ASCII 数据转换为 BCD4 位数数据后，输出到 Y40 ～ Y4F 中的程序。

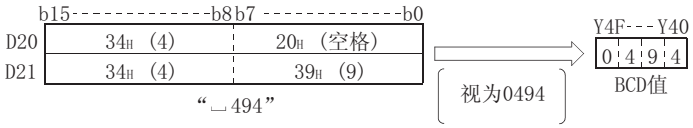
[梯形图模式]



[列表模式]

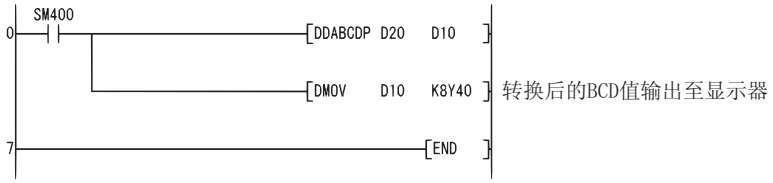
步	指令	软元件	
0	LD	SM400	
1	DABCDP	D20	K4Y40
4	END		

[动作]



(2) 以下为将 D20 ～ D23 中设置的 10 进制数的 ASCII 数据转换为 BCD8 位数数据后，在存储到 D10、D11 中的同时，输出到 Y40 ～ Y5F 中的程序。

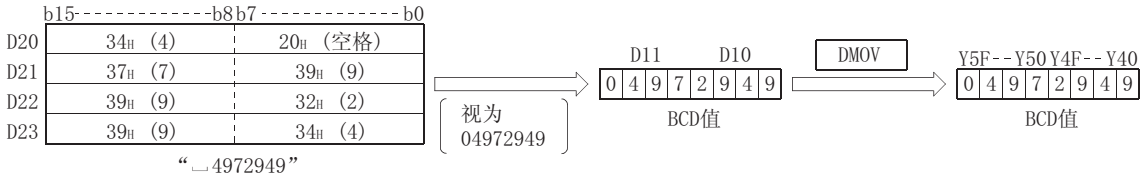
[梯形图模式]



[列表模式]

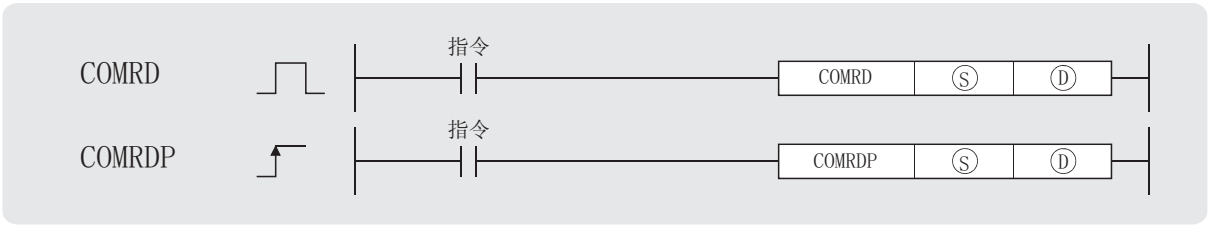
步	指令	软元件	
0	LD	SM400	
1	DDABCDP	D20	D10
4	DMOV	D10	K8Y40
7	END		

[动作]



7.11.7 读取软元件注释数据 (COMRD(P))

Basic | High performance | Process | Redundant | Universal



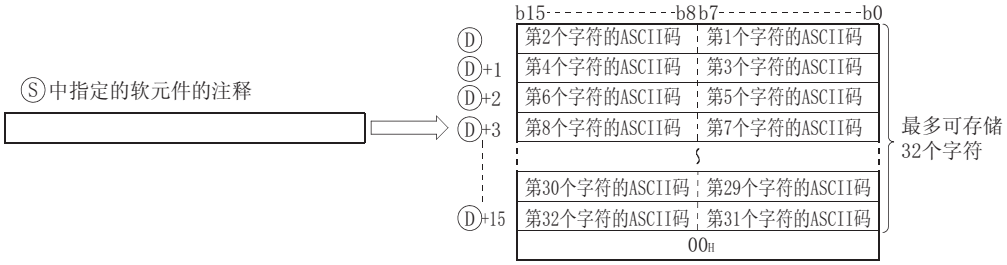
Ⓢ ：登录了读取的注释的软元件的起始编号（软元件名）。

Ⓓ ：存储读取的注释的软元件的起始编号（字符串）。

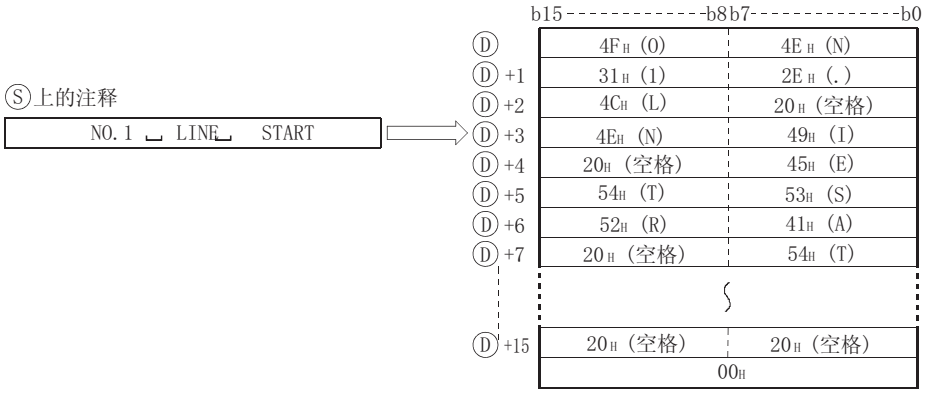
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				BL\S、BL\TR、BL、P、I、J、U
Ⓢ	○	○		○			--		○
Ⓓ	--	○		--			--		--

功 能

(1) 对Ⓢ中指定的软元件编号的注释进行读取后，以 ASCII 码格式存储到Ⓓ中指定编号的软元件后面。



例如，在Ⓢ中指定的软元件注释为“NO.1␣LINE␣START␣␣”时，按以下方式存储到，Ⓓ中。



(2) 在Ⓢ中指定的软元件号未进行注释范围设置，注释未被登录的情况下，注释的字符将全部被处理为“ 20H ”(空格)。

(3) 存储Ⓓ的最后字符的软元件号 +1 根据 SM701(输出字符数切换信号) 的 ON/OFF 状态而不同。

SM701 为 OFF 时 ： 无变化。
SM701 为 ON 时 ： 存储“ 0 ”。

(4) 读取注释时，指令结束后 SM702 将 ON 一个扫描。
此外，指令执行过程中 SM721 处于 ON 状态。
在 SM721 处于 ON 状态时，不能执行 COMRD(P) 指令。执行的情况下将进行无处理。

要 点

1. COMRD(P) 指令中使用的软元件注释使用存储卡及标准 ROM 中存储的注释文件。
   不能使用程序存储器中存储的注释文件。
2. COMRD(P) 指令中使用的注释文件是在可编程控制器参数的“ 可编程控制器文件设置 ”中设置。
   如果未在可编程控制器文件设置中对所使用的注释文件进行设置，将不能通过 COMRD(P) 指令输出软元件注释。
3. COMRD(P) 指令不能在中断程序中执行。
   如果执行将变为无处理。

出 错

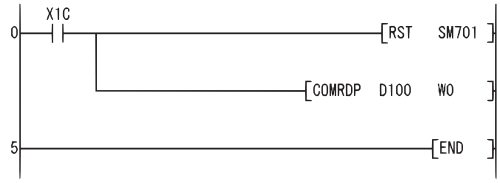
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- Ⓢ中指定的软元件编号的注释未进行登录时。 (出错代码： 4100)
- Ⓓ中指定编号的软元件不是字软元件时。 (出错代码： 4101)
- Ⓓ中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU) (出错代码： 4101)

程序示例

(1) 以下为 X1C 变为 ON 时，将 D100 中设置的注释以 ASCII 码格式存储到 W0 的后面的程序。

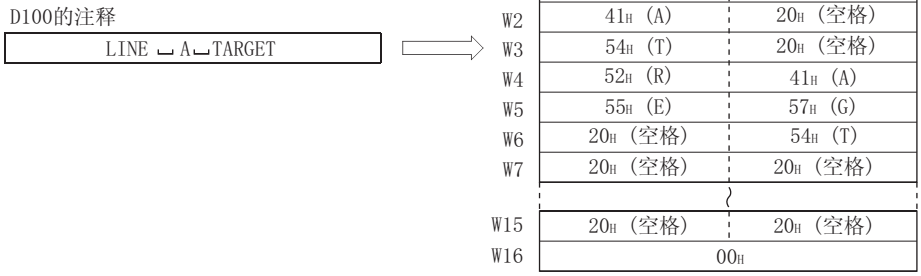
[梯形图模式]



[列表模式]

步	指令	软元件	
0	LD	X1C	
1	RST	SM701	
2	COMRDP	D100	W0
5	END		

[动作]



注意事项

(1) QCPU 在数次扫描后结束处理。

(2) 在指令结束前 (SM721 处于 ON 状态) 即使将 COMRD(P)/PRC 指令的启动信号 (执行指令) 置为 ON，也不能执行 COMRD(P)/PRC 指令。应将程序设置为在 SM721 处于 OFF 状态时执行 COMRD(P)/PRC 指令。

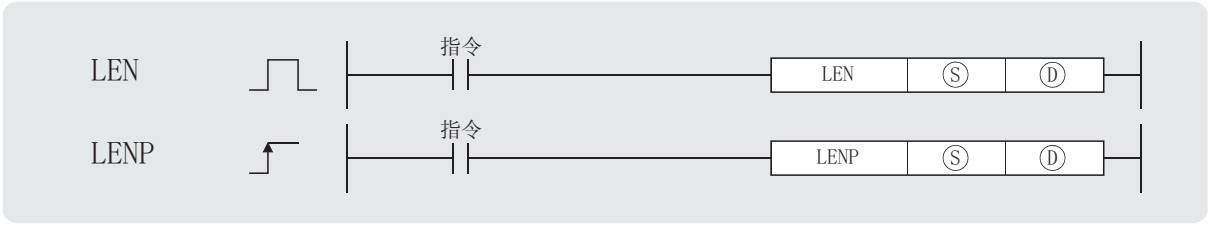
(3) 不能同时访问两个或两个以上的文件注释。

(4) 以下指令由于共用 SM721，所以不能同时执行。

指令名	执行过程中 ON	结束后 ON 一个扫描	异常结束时 ON
SP.FREAD SP.FWRITE	SM721	通过指令指定	(通过指令指定的软元件)+1
PRC COMRD		SM720	无

7.11.8 字符串长度检测 (LEN(P))

Basic High performance Process Redundant Universal



Ⓢ ：字符串或者存储字符串的软元件起始编号（字符串）。

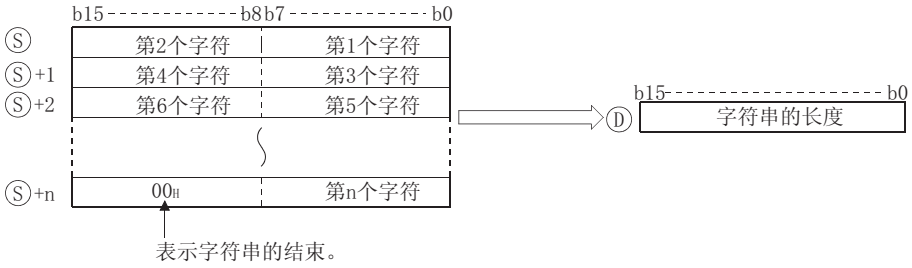
Ⓓ ：存储检测的字符串长度的软元件起始编号 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
Ⓢ	--	○		--				○	--
Ⓓ	○	○		○				--	--

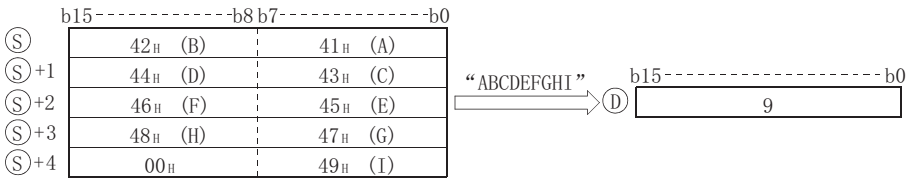
功 能

(1) 对Ⓢ中指定的字符串的长度进行检测后，存储到Ⓓ中指定编号的软元件后面。

将从Ⓢ中指定的软元件编号开始，至存储了“ 00H ”的软元件编号为止的数据作为字符串处理。



例如，在Ⓢ后面存储了“ ABCDEFGHI ”时，在Ⓓ中将存储 9。



出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

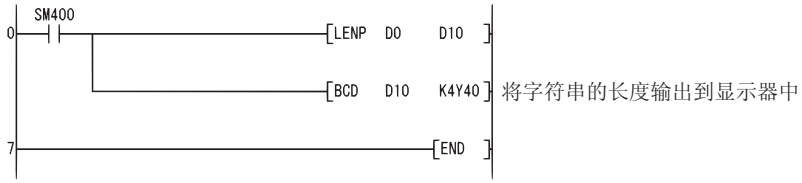
· ⓢ中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)

(出错代码： 4101)

程序示例

(1) 以下为将从 D0 开始的字符串长度以 BCD4 位数格式输出到 Y40 ～ Y4F 中的程序。

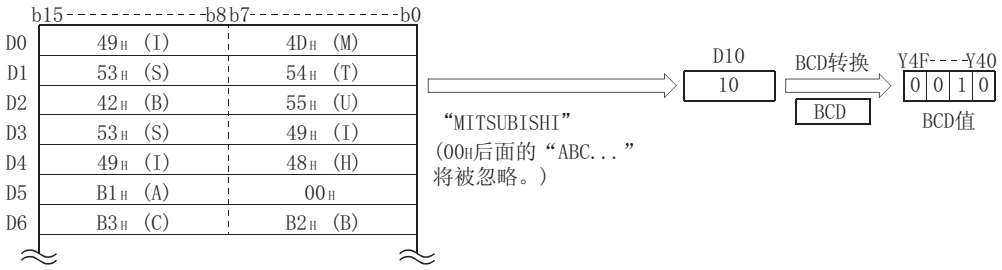
[梯形图模式]



[列表模式]

步	指令	软元件	
0	LD	SM400	
1	LENP	D0	D10
4	BCD	D10	K4Y40
7	END		

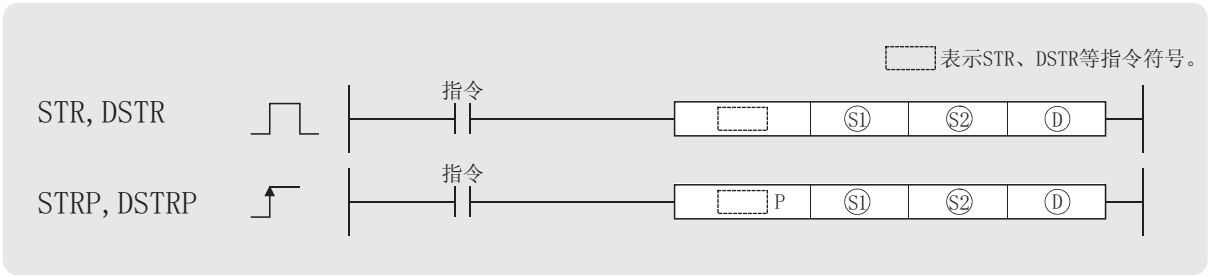
[动作]



7.11.9 BIN16 位 /32 位→字符串的转换 (STR(P)、DSTR(P))

Ver. Basic | High performance | Process | Redundant | Universal

基本型 QCPU：序列号的前 5 位数为 “ 04122 ” 以后



(S1) ：转换数值位数的软元件的起始编号 (BIN16 位)。

(S2) ：进行转换的 BIN 数据 (BIN16/32 位)。

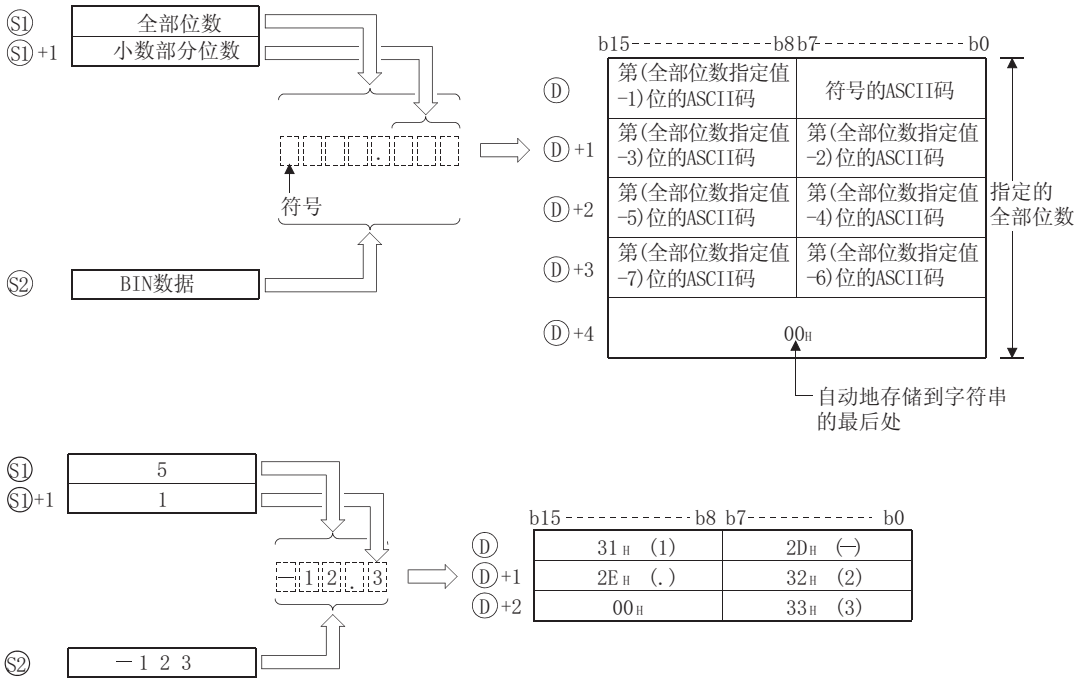
(D) ：存储转换后的字符串的软元件的起始编号 (字符串)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
(S1)	○	○		○				--	--
(S2)	○	○		○				○	--
(D)	--	○		--				--	--

功 能

STR

(1) 将(S2)中指定的 BIN16 位数据转换为在(S1)中指定位置附加了小数点的字符串后，存储到(D)中指定编号的软元件的后面。



(2) ⓢ1 中可指定的全部位数为 2 ～ 8 位。

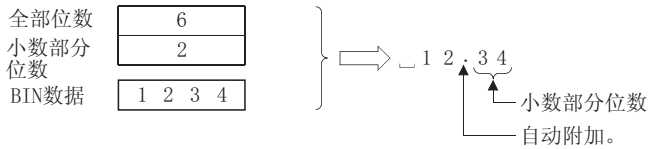
(3) ⓢ1 +1 中可指定的小数部分位数为 0 ～ 5 位。
但是，应设置为小数部分位数≤ (全部位数 -3)。

(4) ⓢ2 中可指定的 BIN 数据范围为 -32768 ～ 32767。

(5) 转换后的字符串数据按以下方式存储到ⓓ的后面的软元件编号中。

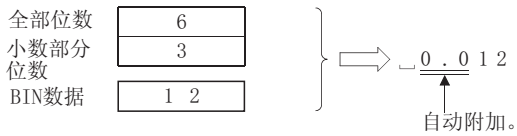
(a) BIN 数据为正数时在 “ 符号 ” 中存储 “ 20H ” (空格)，为负数时在 “ 符号 ” 中存储 “ 2DH ” (-)。

(b) 小数部分位数被设置为 “ 0 ” 以外时，第 (指定位数 +1) 位中将自动存储 “ 2EH ” (.)。

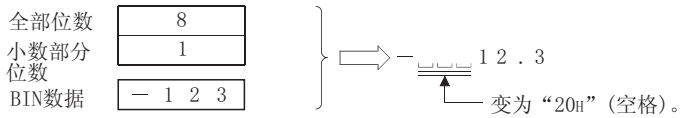


小数部分位数为 “ 0 ” 时，不存储 “ 2EH ” (.)。

(c) 在小数部分位数的值大于 BIN 数据的位数的情况下，在转换时将自动地附加 0 后向右填充对齐，变为 “ 0.□□□□□ ”。



(d) 如果全部位数的值减去符号、小数点后的位数仍然大于 BIN 数据的位数时，将在符号与数值之间存储 “ 20H ” (空格)。

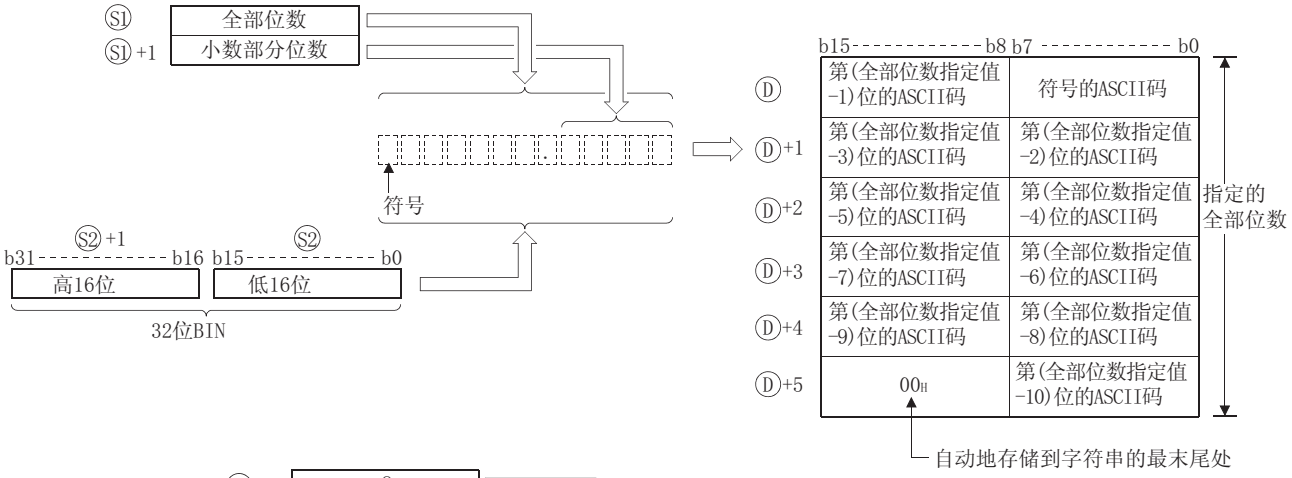


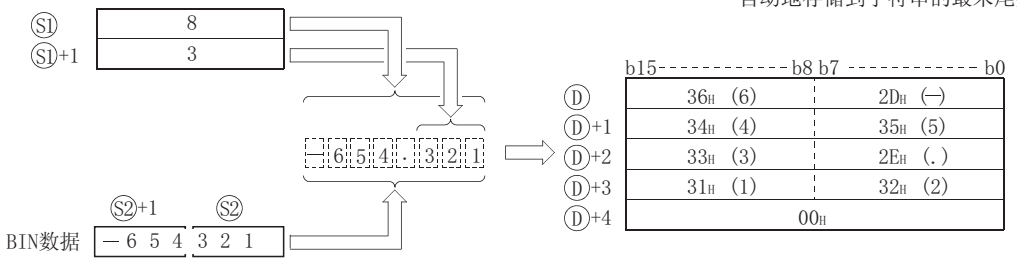
如果 BIN 数据的位数大于全部位数，将变为出错状态。

(e) 在转换后的字符串的最末尾处将自动存储 “ 00H ”。

DSTR

(1) 将Ⓢ2中指定的 BIN32 位数据转换为在Ⓢ1中指定位置附加了小数点的字符串后，存储到Ⓓ中指定编号的软元件的后面。





(2) Ⓢ1中可指定的全部位数为 2 ～ 13 位。

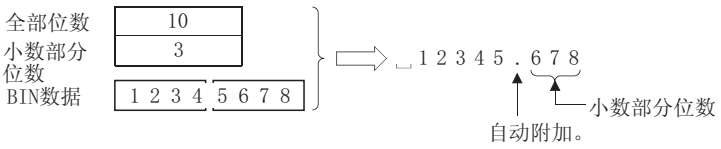
(3) Ⓢ1+1 中可指定的小数部分位数为 0 ～ 10 位。
但是，应设置为小数部分位数≤ (全部位数 -3)。

(4) Ⓢ1、Ⓢ2+1 中可指定的 BIN 数据范围为 -2147483648 ～ 2147483647。

(5) 转换后的字符串数据按以下方式存储到Ⓓ的后面的软元件编号中。

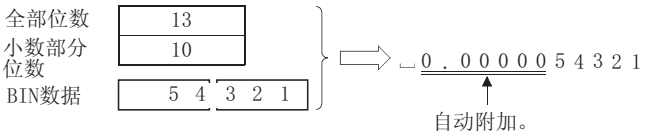
(a) BIN 数据为正数时在 “ 符号 ” 中存储 “ 20H ” (空格)，为负数时在 “ 符号 ” 中存储 “ 2DH ” (-)。

(b) 小数部分位数被设置为 “ 0 ” 以外时，第 (指定位数 +1) 位中将自动存储 “ 2EH ” (.)。

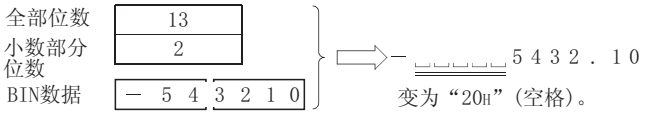


小数部分位数为 “ 0 ” 时，不存储 “ 2EH ” (.)。

(c) 在小数部分位数的值大于 BIN 数据的位数的情况下，在转换时将自动地附加 0 后向右填充对齐，变为“ 0.□□□□□ ”。



(d) 如果全部位数的值减去符号、小数点后的位数仍然大于 BIN 数据的位数时，将在符号与数值之间存储“ 20H ”(空格)。



如果 BIN 数据的位数大于全部位数，将变为出错状态。

(e) 在转换后的字符串的最末尾处将自动存储“ 00H ”。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· ⓢ1 中指定的全部位数指定超出了以下范围时。 (出错代码： 4100)
使用 STR 指令时 .... 2 ～ 8
使用 DSTR 指令时 ... 2 ～ 13

· ⓢ1 +1 中指定的小数部分位数指定超出了以下范围时。 (出错代码： 4100)
使用 STR 指令时 .... 0 ～ 5
使用 DSTR 指令时 ... 0 ～ 10

· ⓢ1 中指定的全部位数与ⓢ1 +1 中指定的小数部分位数指定值的关系不符合下述条件时。 (出错代码： 4100)
全部位数 -3 ≥小数部分位数

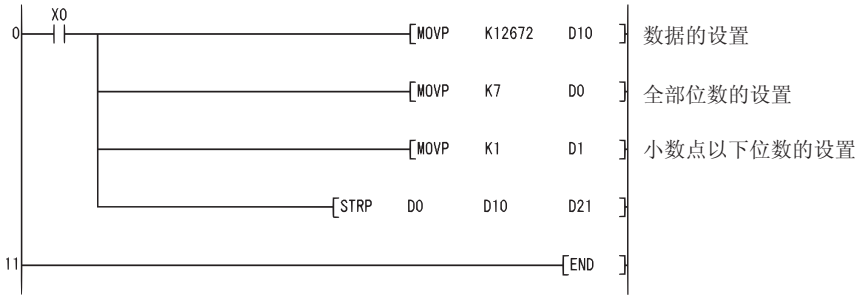
· ⓢ1 中指定的位数小于“ ⓢ2 中指定的 BIN 数据的位数 +2 ”时。
(ⓢ1 的位数＜ⓢ2 的不含符号的 BIN 数据的位数 + 符号 (+ 或 -) 的位数 + 小数点 (.) 的位数时) (出错代码： 4100)

· 存储ⓓ 中指定的字符串的软元件超出了相应软元件的范围时。 (出错代码： 4101)

程序示例

(1) 以下为 X0 变为 ON 时，将 D10 中存储的 BIN16 位数据按照 D0、D1 的位数指定转换为字符串后，存储到 D20 ～ D23 中的程序。

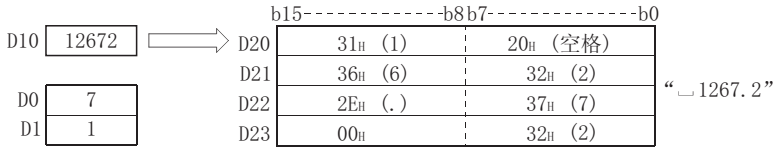
[梯形图模式]



[列表模式]

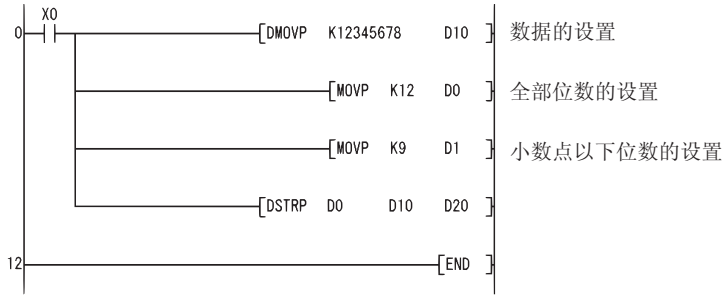
步	指令	软元件		
0	LD	X0		
1	MOVP	K12672	D10	
3	MOVP	K7	D0	
5	MOVP	K1	D1	
7	STRP	D0	D10	D21
11	END			

[动作]



(2) 以下为 X0 变为 ON 时，将 D10、D11 中存储的 BIN32 位数据按照 D0、D1 的位数指定转换为字符串后，存储到 D20 ～ D26 中的程序。

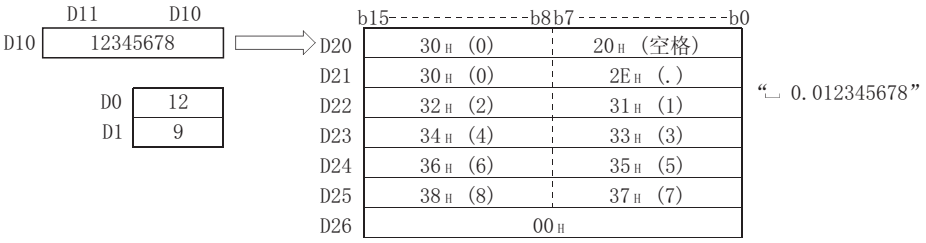
[梯形图模式]



[列表模式]

步	指令	软元件		
0	LD	X0		
1	DMOVP	K12345678		D10
4	MOVP	K12	D0	
6	MOVP	K9	D1	
8	DSTRP	D0	D10	D20
12	END			

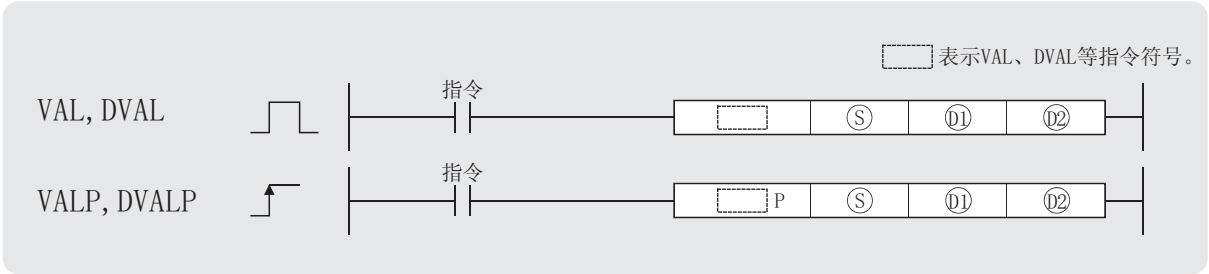
[动作]



7.11.10 字符串→ BIN16 位 /32 位数据的转换 (VAL(P)、DVAL(P))

Ver. Basic | High performance | Process | Redundant | Universal

基本型 QCPU：序列号的前 5 位数为 “ 04122 ” 以后
(支持的 GX Developer：Version8.00A 以后)



Ⓢ ：转换为 BIN 数据的字符串或者存储字符串的软元件的起始编号 (字符串)。
Ⓓ1 ：存储转换后的 BIN 数据的位数的软元件的起始编号 (BIN16 位)。
Ⓓ2 ：存储转换后的 BIN 数据的软元件的起始编号 (BIN16/32 位)。

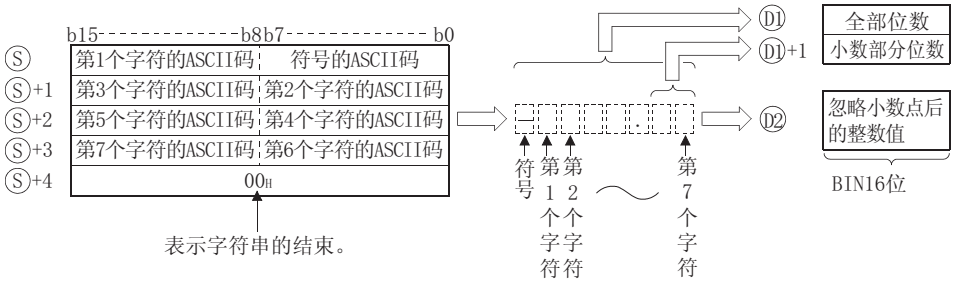
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
Ⓢ	--	○		--				○	--
Ⓓ1	○	○		--				--	--
Ⓓ2	○	○		○				--	--

功 能

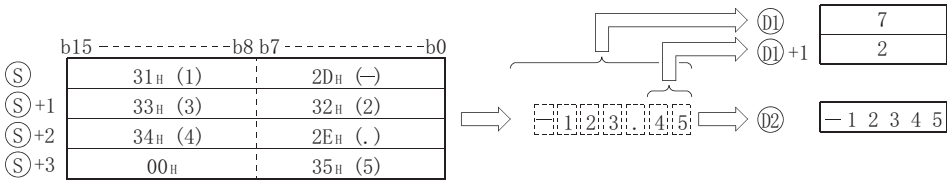
VAL

(1) 将Ⓢ中指定的软元件编号后面存储的字符串转换为 BIN16 位数据后，将位数及 BIN 数据存储到Ⓓ1、Ⓓ2中。

在字符串→ BIN 转换中，将Ⓢ中指定的软元件编号开始至存储了 “ 00H ” 软元件编号为止的数据作为字符串处理。



例如，在Ⓢ后面指定了 “ -123.45 ” 的字符串时，将按以下方式存储到Ⓓ1、Ⓓ2中。



(2) 在Ⓢ中可指定的字符串的全部字符数为 2 ～ 8 个字符。

(3) 在Ⓢ中指定的字符串中的小数部分的字符数范围为 0 ～ 5 个字符。
但是，应在 (全部位数 -3) 以下。

(4) 可转换为 BIN 值的数值字符串的范围 (小数点被忽略后的值) 为 -32768 ～ 32767。
此外，符号及小数点除外的数值字符串只能在 “ 30H ” ～ “ 39H ” 的范围内进行指定。
小数点被忽略后的值如下所示。

例 “ -12345.6 ” → “ -123456 ”

(5) 表示正的数值时在 “ 符号 ” 中存储 “ 20H ”，表示负的数值时在 “ 符号 ” 中存储 “ 2DH ”。

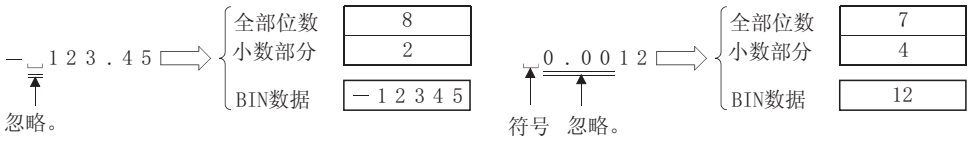
(6) 小数点中设置 “ 2EH ”。

(7) Ⓓ1中存储的全部位数是表示数值的字符 (包含符号、小数点) 的所有的字符数。

Ⓓ1+1 中存储的小数部分位数是表示 “ 2EH ” (.) 后面的小数部分的字符数。

Ⓓ2中存储的 BIN 数据是由忽略了小数点后的字符串所转换成的 BIN 值。

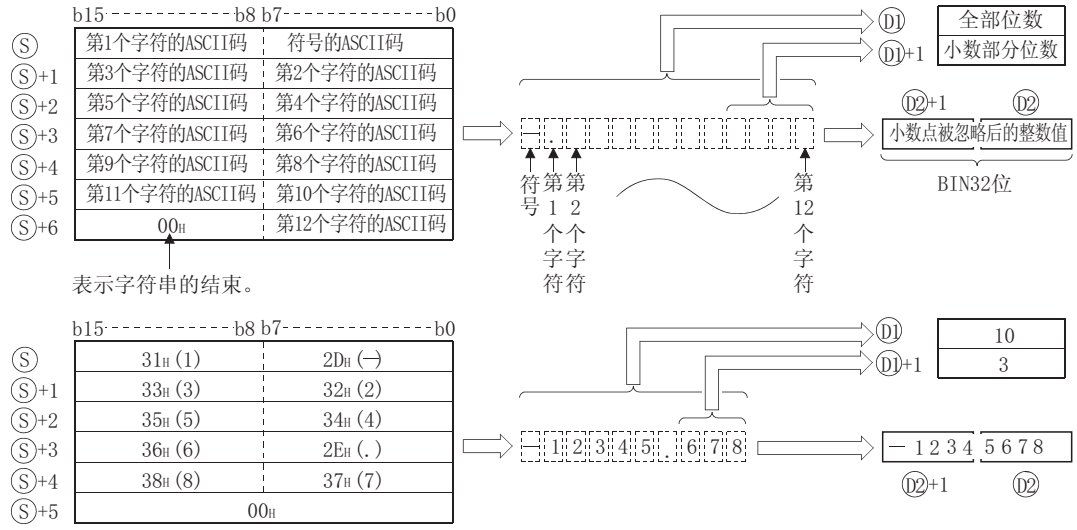
(8) 在Ⓢ中指定的字符串中，在符号和第一个非零的数值之间存在有 “ 20H ” (空格) 或 “ 30H ” (0) 时，则在忽略 “ 20H ”、“ 30H ” 的状况下将其转换为 BIN 值。



DVAL

(1) 将Ⓢ中指定的软元件编号后面存储的字符串转换为 BIN32 位数据后，将位数及 BIN 数据存储到Ⓓ1、Ⓓ2中。

在字符串→ BIN 转换中，将Ⓢ中指定的软元件编号开始至存储了 “ 00H ” 软元件编号为止的数据作为字符串处理。



(2) 在Ⓢ中可指定的字符串的全部字符数为 2 ～ 13 个字符。

(3) 在Ⓢ中指定的字符串中的小数部分的字符数范围为 0 ～ 10 个字符。
但是，应在 (全部位数 -3) 以下。

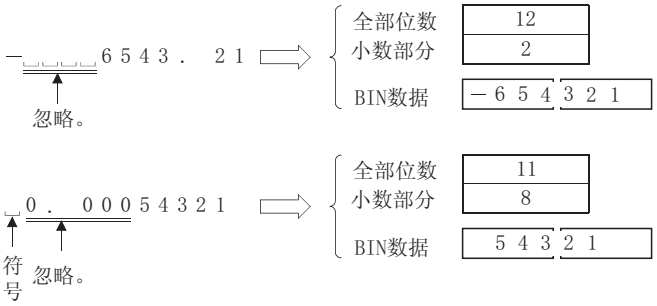
(4) 可转换为 BIN 值的数值字符串的范围 (小数点被忽略后的值) 为 - 2147483648 ～ 2147483647。
此外，符号及小数点除外的数值字符串只能在 “ 30H ” ～ “ 39H ” 的范围内进行指定。

(5) 表示正的数值时在 “ 符号 ” 中存储 “ 20H ”，表示负的数值时在 “ 符号 ” 中存储 “ 2DH ”。

(6) 小数点中设置 “ 2EH ”。

(7) Ⓓ1中存储的全部位数是表示数值的字符 (包含符号、小数点) 的所有的字符数。
Ⓓ1+1 中存储的小数部分位数是表示 “ 2EH ” (.) 后面的小数部分的字符数。
Ⓓ2中存储的 BIN 数据是由忽略了小数点后的字符串所转换成的 BIN 值。

(8) 在Ⓢ中指定的字符串中，在符号与第一个非零的数值之间存在有 “ 20H ” (空格) 或 “ 30H ” (0) 时，则在忽略 “ 20H ”、“ 30H ” 的状况下将其转换为 BIN 值。



出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- Ⓢ1中指定的字符串的字符数超出了下述范围时。 (出错代码： 4100)
  使用 VAL 指令时 . . . . . 2 ～ 8 个字符
  使用 DVAL 指令时 . . . . . 2 ～ 13 个字符
- Ⓢ中指定的小数部分字符数超出了下述范围时。 (出错代码： 4100)
  使用 VAL 指令时 . . . . . 2 ～ 5 个字符
  使用 DVAL 指令时 . . . . . 2 ～ 10 个字符
- Ⓢ中指定的全部字符数与小数部分字符数的关系超出了下述范围时。 (出错代码： 4100)
  全部字符数 -3 ≥小数部分字符数
- 符号中设置了除 “ 20H ”、“ 2DH ” 以外的 ASCII 码时。 (出错代码： 4100)
- 各数字的位数中设置了除 “ 30H ” ～ “ 39H ” 以及 “ 2EH ” (小数点) 以外的 ASCII 码时。 (出错代码： 4100)
- 设置了多个小数点时。 (出错代码： 4100)

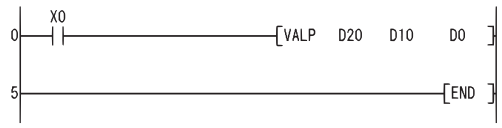
· 转换后的 BIN 值超出了下述范围时。 (出错代码： 4100)
使用 VAL 指令时 ..... —32768 ～ 32767
使用 DVAL 指令时 ..... —2147483648 ～ 2147483647

· 在从Ⓢ中指定的软元件开始至相应软元件的最终软元件编号为止之间未设置“ 00H ”时。
(出错代码： 4101)

程序示例

(1) 以下为 X0 变为 ON 时，将 D20 ～ D22 中存储的字符串数据视为整数值转换为 BIN 值后，存储到 D0 中的程序。

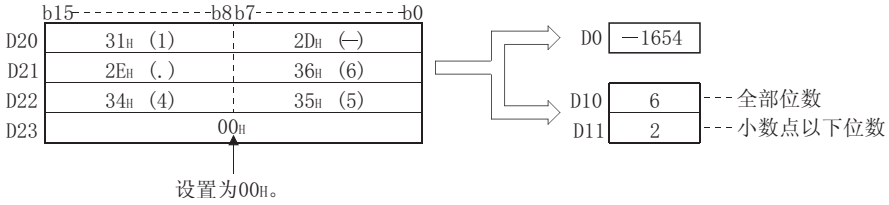
[梯形图模式]



[列表模式]

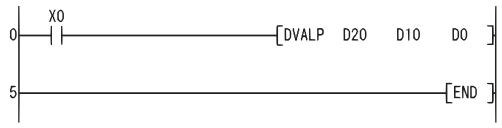
步	指令	软元件		
0	LD	X0		
1	VALP	D20	D10	D0
5	END			

[动作]



(2) 以下为 X0 变为 ON 时，将 D20 ～ D24 中存储的字符串数据视为整数值转换为 BIN 值后，存储到 D0 中的程序。

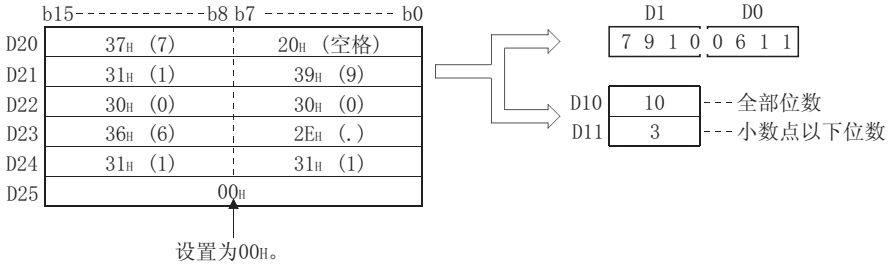
[梯形图模式]



[列表模式]

步	指令	软元件		
0	LD	X0		
1	DVALP	D20	D10	D0
5	END			

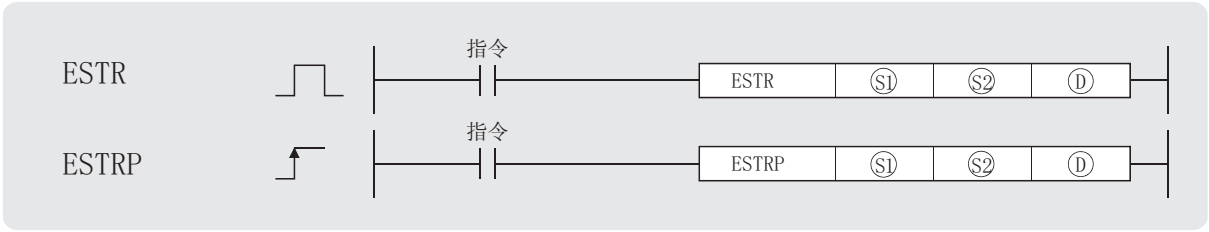
[动作]



7.11.11 浮点数→字符串的转换 (ESTR(P))

Ver. Basic | High performance | Process | Redundant | Universal

基本型 QCPU：序列号的前 5 位数为“ 04122 ”以后



Ⓢ1 ： 进行转换的 32 位浮点数据或者存储数据的软元件的起始编号 (实数)。

Ⓢ2 ： 存储进行转换的数值的显示指定的软元件的起始编号 (BIN16 位)。

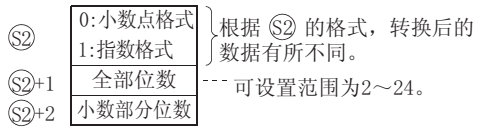
Ⓓ ： 存储转换后的字符串的软元件的起始编号 (字符串)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
Ⓢ1	--	○		--	○		--	○	--
Ⓢ2	--	○		--	--		--	--	--
Ⓓ	--	○		--	--		--	--	--

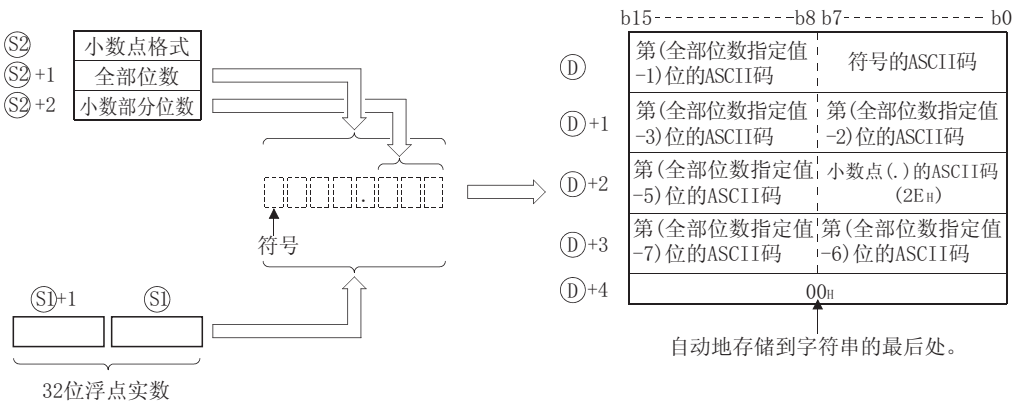
功 能

(1) 将Ⓢ1中指定的 32 位浮点实数数据按照Ⓢ2中指定的显示指定转换为字符串后，存储到Ⓓ中指定编号的软元件的后面。

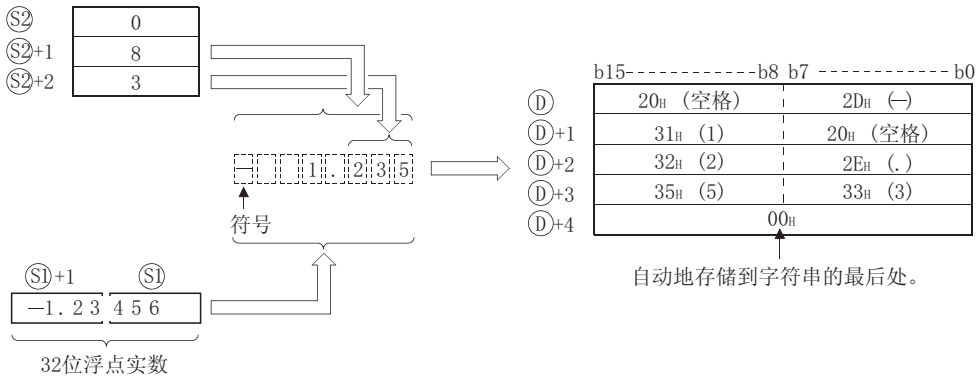
(2) 根据Ⓢ2中指定的显示指定，转换后的数据有所不同。



小数点格式时



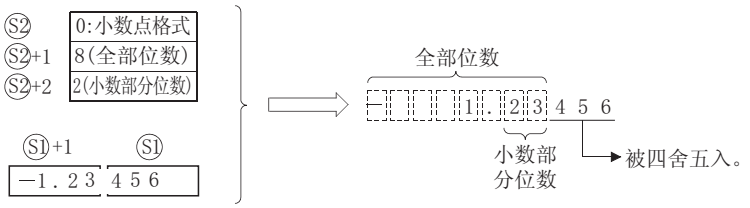
例如，全部位数为 8，小数部分位数为 3 时，指定了 -1.23456 的情况下，按以下方式储存到Ⓓ的后面。



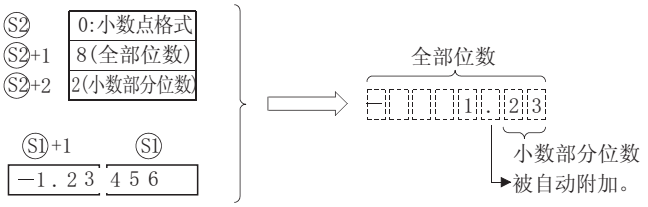
(a) Ⓢ2+1 中可指定的全部位数如下所示：
小数部分位数为“ 0 ”时 . . . . . 位数 (最大： 24) ≥ 2
小数部分位数为“ 0 ”以外时 . . . . . 位数 (最大： 24) ≥ (小数部分位数 +3)

(b) Ⓢ2+2 中可指定的小数部分位数为 0 ～ 7 位。
但是，设置时应满足以下条件： 小数部分位数≤ (全部位数 -3)。

(c) 转换后的字符串数据按以下方式存储到Ⓓ后面的软元件编号中。
1) 32 位浮点实数数据为正数时在“ 符号 ”中存储“ 20H ”(空格)，为负数时在“ 符号 ”中存储“ 2DH ”(-)。
2) 小数部分位数的范围中，不能容纳 32 位浮点实数数据的小数部分时，低位小数部分将被四舍五入。

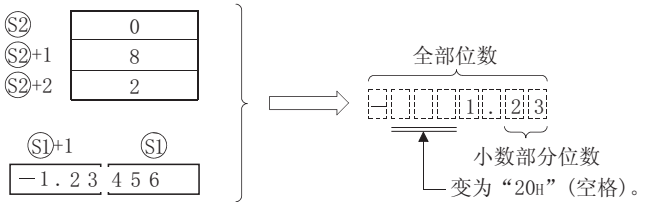


3) 将小数部分位数设置为“ 0 ”以外时，在指定的小数部分位数 +1 位中将自动存储“ 2EH ”(.)。



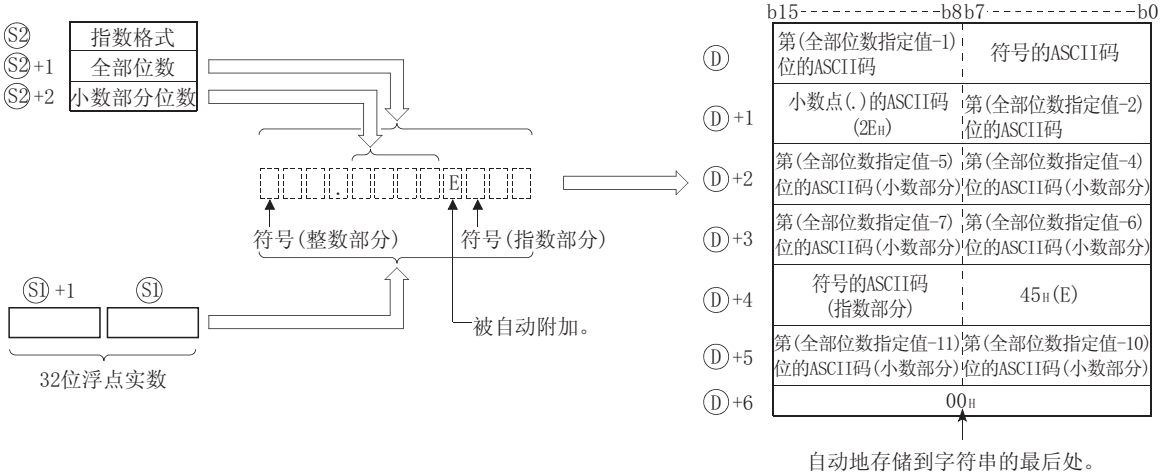
小数部分位数为“ 0 ”时，不存储“ 2EH ”(.)。

4) 从全部位数中减去符号、小数点、小数部分后的位数大于 32 位浮点实数数据的整数部分时，在符号与整数部分之间将存储“ 20H ”(空格)。

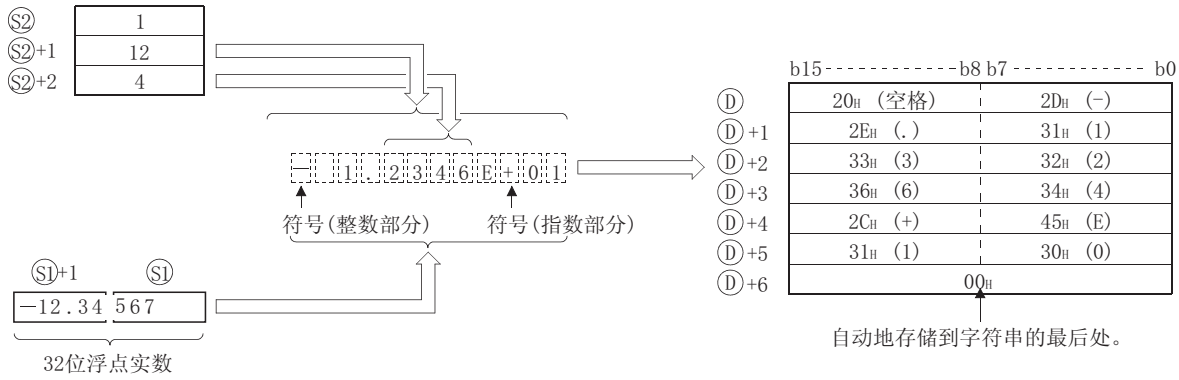


5) 转换后的字符串的末尾处将自动存储“ 00H ”。

指数格式时



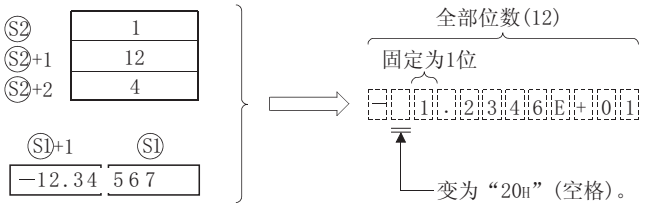
例如，全部位数为 12，小数部分位数为 4 时，指定了 -12.34567 的情况下，按以下方式储存到(D)的后面。



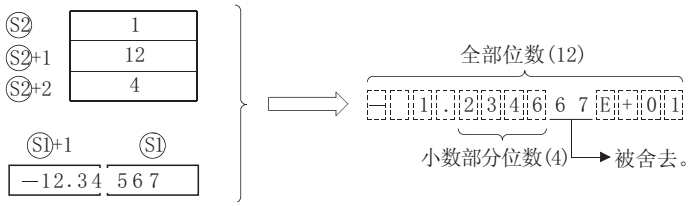
(a) ⓢ2+1 中可指定的全部位数如下所示：
小数部分位数为“ 0 ”时 . . . . . 位数 (最大： 24) ≥ 2
小数部分位数为“ 0 ”以外时 . . . . . 位数 (最大： 24) ≥ (小数部分位数 +7)

(b) ⓢ2+2 中可指定的小数部分位数为 0 ～ 7 位。
但是，设置时应满足以下条件： 小数部分位数≤ (全部位数 -7)。

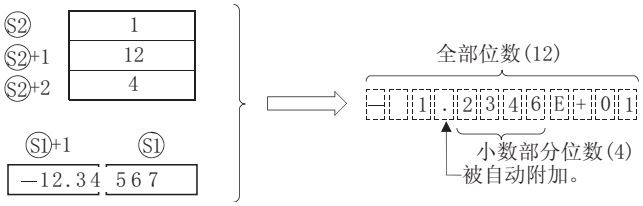
(c) 转换后的字符串数据按以下方式存储到ⓓ后面的软元件编号中。
1) 32 位浮点实数数据为正数时，在整数部分的“ 符号 ”中存储“ 20H ”(空格)，为负数时在整数部分的“ 符号 ”中存储“ 2DH ”(－)。
2) 整数部分固定为 1 位。
在整数部分与符号之间存储“ 20H ”(空格)。



3) 小数部分位数的范围中，不能容纳 32 位浮点实数数据的小数部分时，低位小数部分将被四舍五入。



4) 将小数部分位数设置为“ 0 ”以外时，在指定的小数部分位数 +1 位中将自动存储“ 2EH ”(.)。



小数部分位数为“ 0 ”时，不存储“ 2EH ”(.)。

5) 指数为正数时，在指数部分的“ 符号 ”中存储“ 2CH ”(+)，为负数时在“ 符号 ”中存储“ 2DH ”(-)。

6) 指数部分固定为 2 位。
指数部分为 1 位时，在与指数部分的符号之间存储“ 30H ”(0)。

7) 转换后的字符串的末尾处将自动存储“ 00H ”。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· (S1)超出了下述范围时。 (出错代码： 4100)
0，$2^{-126} \leqq |(S1)| < 2^{128}$

· (S2)中指定的格式指定为 0、1 以外时。 (出错代码： 4100)

· (S2)+1 中指定的全部位数指定超出了下述范围时。 (出错代码： 4100)
小数点格式时
小数部分位数为“ 0 ”时 ..... 全部位数≥ 2
小数部分位数为“ 0 ”以外时 ..... 全部位数≥ (小数部分位数 +3)
指数格式时
小数部分位数为“ 0 ”时 ..... 全部位数≥ 6
小数部分位数为“ 0 ”以外时 ..... 全部位数≥ (小数部分位数 +7)

· (S2)+2 中指定的小数部分位数指定超出了下述范围时。 (出错代码： 4100)
小数点格式时 ..... 小数部分位数≤ (全部位数 -3)
指数格式时 ..... 小数部分位数≤ (全部位数 -7)

· 全部位数中指定了超出“ 24 ”的值时。 (出错代码： 4100)

· 存储(D)中指定的字符串的软元件范围超出了相应软元件的范围时。
(出错代码： 4101)

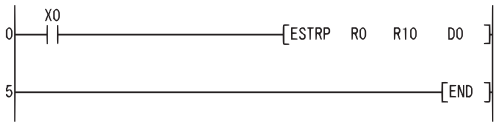
· (S2)中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)
(出错代码： 4101)

· 指定软元件的内容为 -0、非正规数、非数、±∞时。(仅通用型 QCPU)
(出错代码： 4140)

程序示例

(1) 以下为 X0 变为 ON 时，将 R0、R1 中存储的 32 位浮点实数数据，按照 R10 ～ R12 中存储的转换指定进行转换后，存储到 D0 后面的程序。

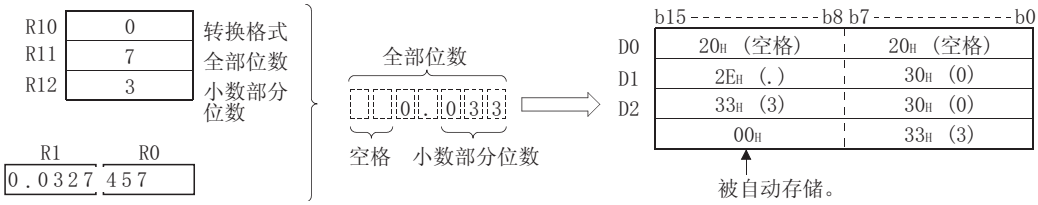
[梯形图模式]



[列表模式]

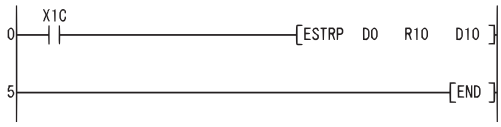
步	指令	软元件		
0	LD	X0		
1	ESTRP	R0	R10	D0
5	END			

[动作]



(2) 以下为 X1C 变为 ON 时，将 D0、D1 中存储的 32 位浮点实数数据，按照 R10 ～ R12 中存储的转换指定进行转换后，存储到 D10 后面的程序。

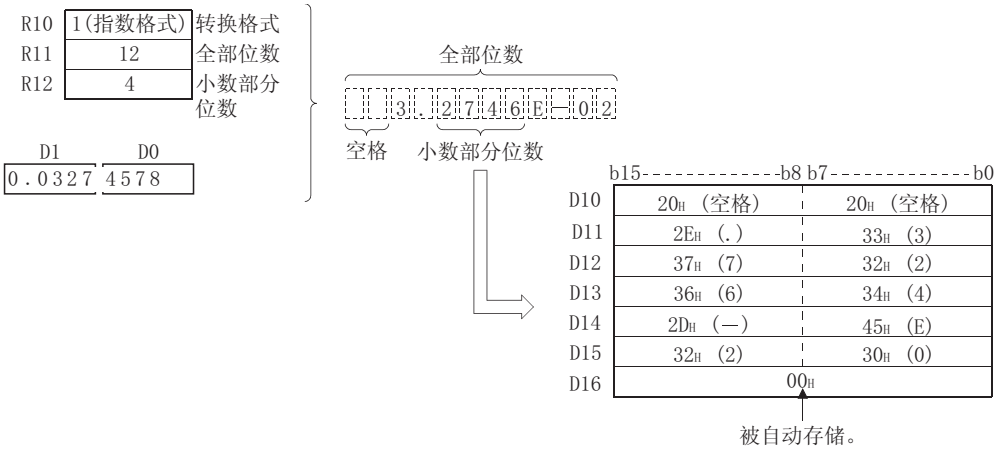
[梯形图模式]



[列表模式]

步	指令	软元件		
0	LD	X1C		
1	ESTRP	D0	R10	D10
5	END			

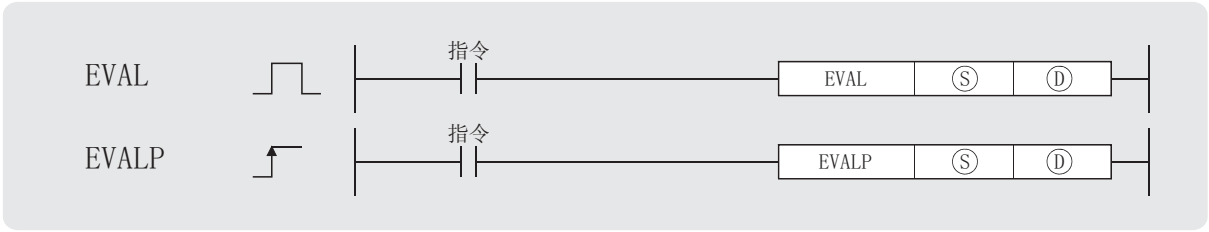
[动作]



7.11.12 字符串→浮点数的转换 (EVAL(P))

Ver. Basic | High performance | Process | Redundant | Universal

基本型 QCPU：序列号的前 5 位数为“ 04122 ”以后



Ⓢ ：要转换为 32 位浮点实数数据的字符串数据或者存储字符串数据的软元件的起始编号（字符串）。

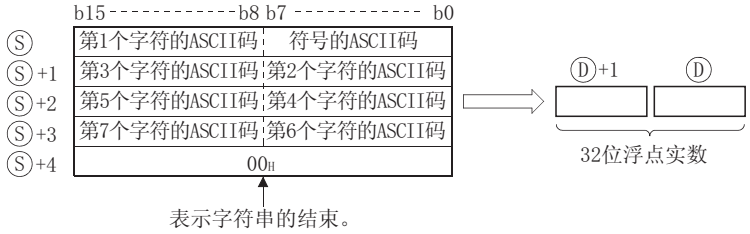
Ⓓ ：存储转换后的 32 位浮点实数数据的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
Ⓢ	--	○		--	--		--	○	--
Ⓓ	--	○		--	○		--	--	--

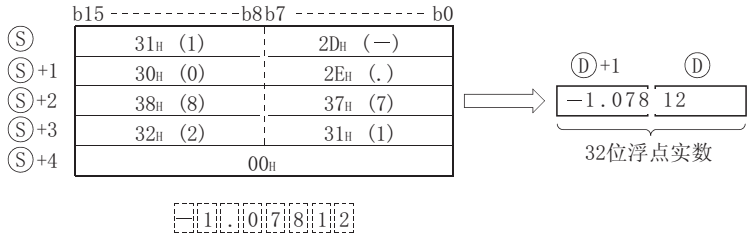
功 能

(1) 将Ⓢ中指定的软元件编号后面存储的字符串转换为 32 位浮点实数后，存储到Ⓓ中指定的软元件中。

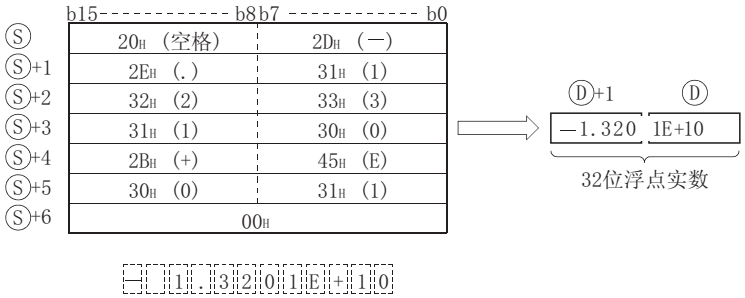
(2) 无论指定的字符串是小数点格式还是指数格式，均可转换为 32 位浮点实数数据。



(a) 小数点格式时

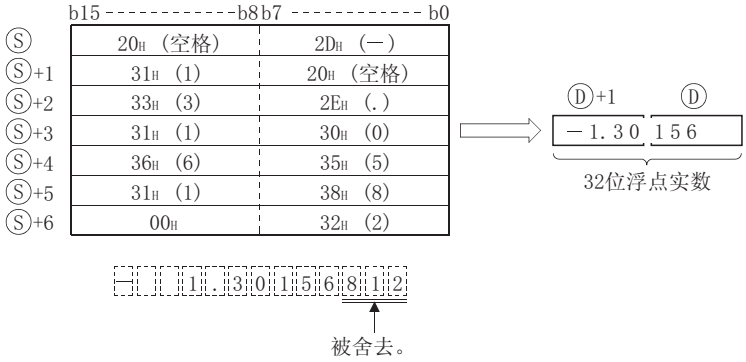


(b) 指数格式时

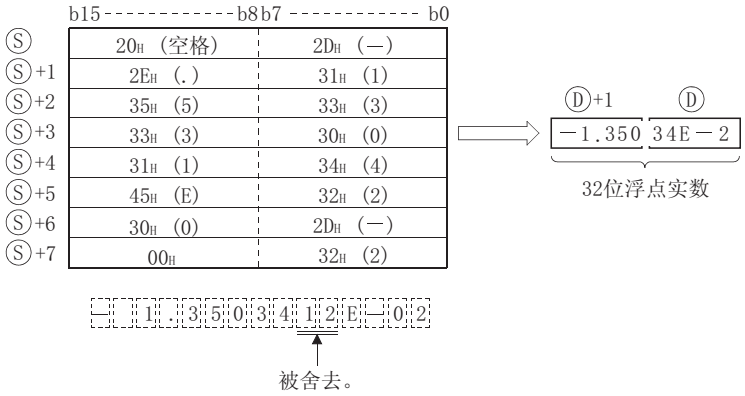


(3) 在Ⓢ中指定的字符串中，要转换为 32 位浮点实数的字符串在除符号、小数点、指数部分外的 6 位数有效，在转换时从第 7 位以后将被舍去。

(a) 小数点格式时



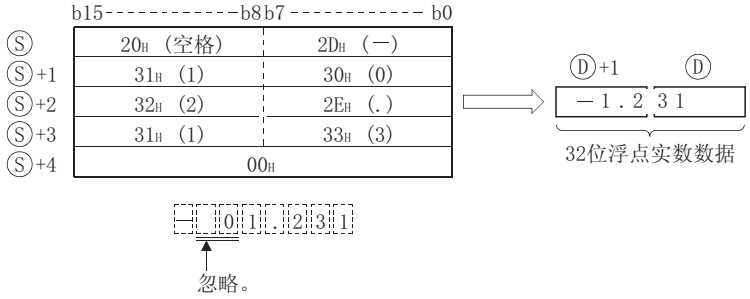
(b) 指数格式时



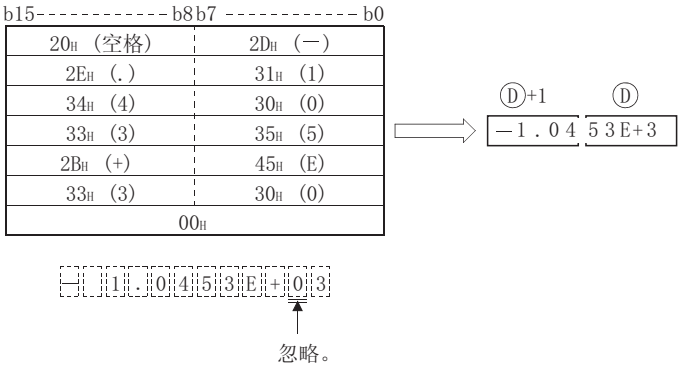
(4) 小数点格式中如果符号中指定了“ 2BH ”(+)，或者符号被省略，则作为正值进行转换。此外，如果符号中指定了“ 2DH ”(-)，则作为负值进行转换。

(5) 指数格式中如果符号中指定了“ 2BH ”(+)，或者符号被省略，则作为正值进行转换。如果符号中指定了“ 2DH ”(-)，则作为负值进行转换。

(6) 在Ⓢ中指定的字符串中，在符号和第一个非零的数值之间存在有“ 20H ”(空格) 或“ 30H ” (0) 时，则在忽略“ 20H ”、“ 30H ”的状况下进行转换。



(7) 在指数格式的字符串中，在“ E ”与数值之间存在有，“ 30H ” (0) 时，则在忽略“ 30H ”的状况下进行转换。



(8) 在字符串中包含有“ 20H ”(空格) 时，将在忽略“ 20H ”的状况下进行转换。

(9) 字符串最多可设置 24 个字符。
字符串中“ 20H ”(空格)、“ 30H ” (0) 也作为 1 个字符计数。

出 错

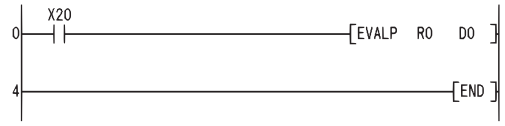
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- 整数部分、小数部分中存在有除“ 30H ” (0) ～“ 39H ” (9) 以外的字符时。 (出错代码： 4100)
- Ⓓ中指定的字符串中存在有 2 个或以上的“ 2EH ” (.) 时。 (出错代码： 4100)
- 指数部分中存在有除“ 45H ” (E)、“ 2BH ” (+)、“ 45H ” (E)、“ 2DH ” (-) 以外的字符时，或者有多个指数部分时。 (出错代码： 4100)
- 转换后的数据不在下述范围内时。 (出错代码： 4100)
  0，$2^{-126} \leqq$ ｜转换后的数据｜ $< 2^{128}$
- 从Ⓢ开始的相应软元件范围内没有“ 00H ”时。 (出错代码： 4101)
- Ⓢ后面的字符数为 0 或者超过 24 个字符时。 (出错代码： 4100)

程序示例

(1) 以下为 X20 变为 ON 时，将 R0 后面存储的字符串转换为 32 位浮点实数后，存储到 D0、D1 中的程序。

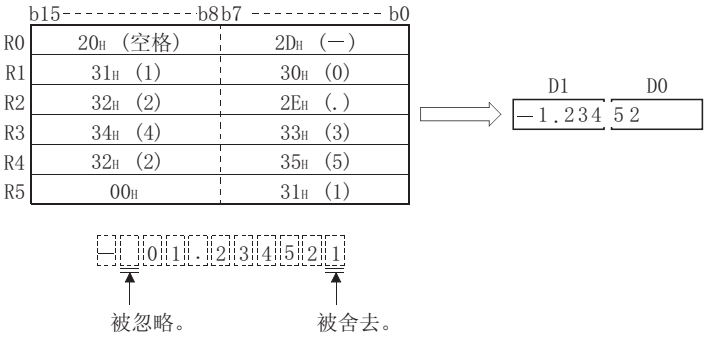
[梯形图模式]



[列表模式]

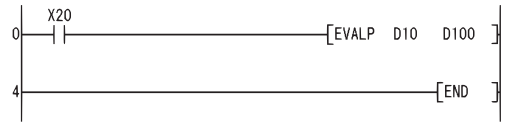
步	指令	软元件	
0	LD	X20	
1	EVALP	R0	D0
4	END		

[动作]



(2) 以下为 X20 变为 ON 时，将 D10 后面存储的字符串转换为 32 位浮点实数后，存储到 D100、D101 中的程序。

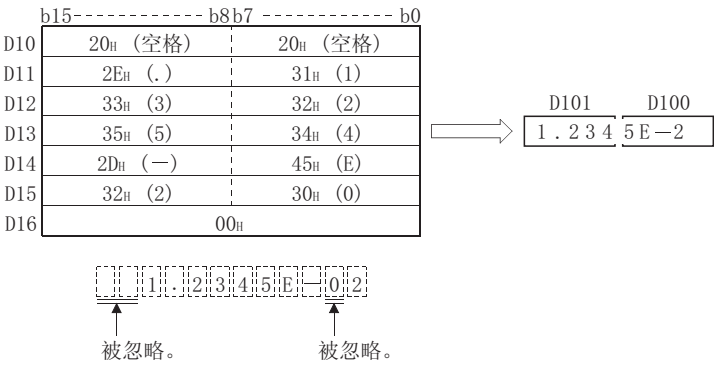
[梯形图模式]



[列表模式]

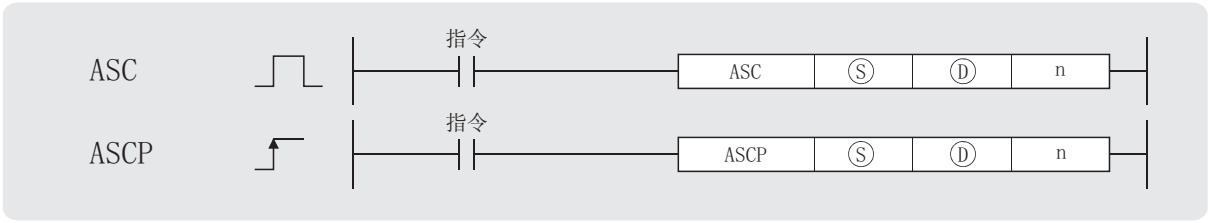
步	指令	软元件	
0	LD	X20	
1	EVALP	D10	D100
4	END		

[动作]



7.11.13 16 进制 BIN → ASCII 码的转换 (ASC(P))

Basic High performance Process Redundant Universal

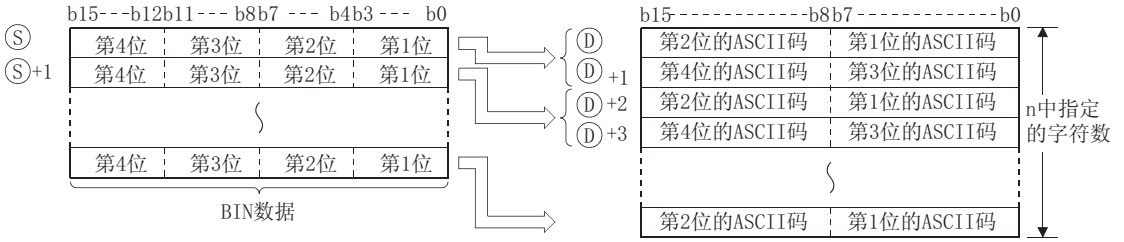


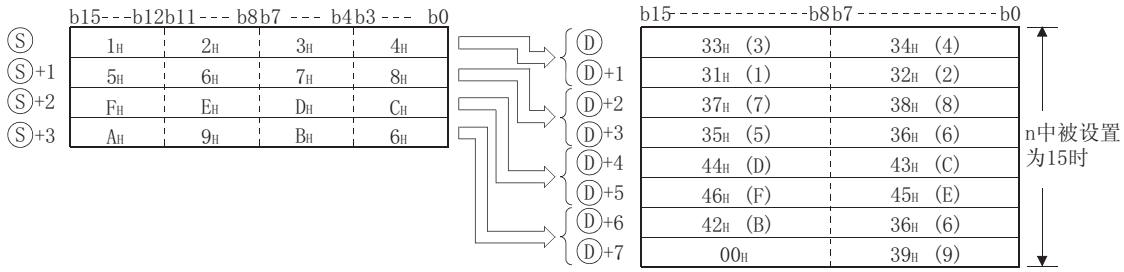
Ⓢ ： 存储要转换为字符串的 BIN 数据的软元件的起始编号 (BIN16 位)。
Ⓓ ： 存储转换后的字符串的软元件的起始编号 (字符串)。
n ： 存储的字符数 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	--	○		--					--
Ⓓ	--	○		--					--
n	○	○		○					--

功 能

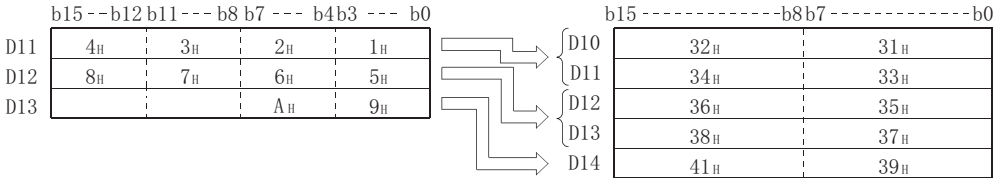
(1) 将Ⓢ中指定的软元件编号后面存储的 BIN16 位数据，通过 16 进制数处理转换为 ASCII 后，以 n 中指定的字符数范围存储到Ⓓ中指定的软元件编号后面。





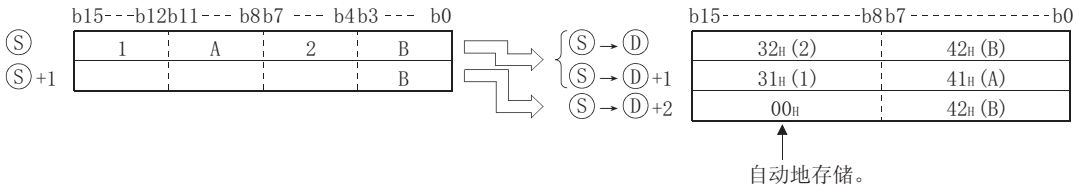
(2) 通过在 n 中设置字符数，Ⓢ中指定的 BIN 数据的范围以及Ⓓ中指定的字符串的存储软元件的范围将被自动确定。

(3) 即使存储要转换的 BIN 数据的软元件范围与存储转换后的 ASCII 数据软元件范围重复时，也可正常进行处理。



(4) n 中指定的字符数为奇数时，存储字符串的软元件范围的最终软元件编号的高 8 位中将自动存储“ 00H ”。

n 的字符数为 5 时



(5) n 中指定的字符数为“ 0 ”时，不执行转换处理。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

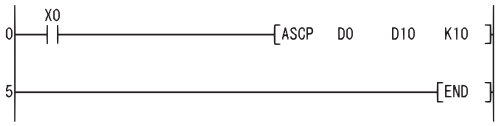
· Ⓢ中指定的软元件编号后面的 n 中指定的字符数范围超出了相应软元件的范围时。
(出错代码： 4101)

· Ⓓ中指定的软元件编号后面的 n 中指定的字符数范围超出了相应软元件的范围时。
(出错代码： 4101)

程序示例

(1) 以下为 X0 变为 ON 时，将 D0 中存储的 BIN 数据视为 16 进制数转换为字符串后，存储到 D10 ～ D14 中的程序。

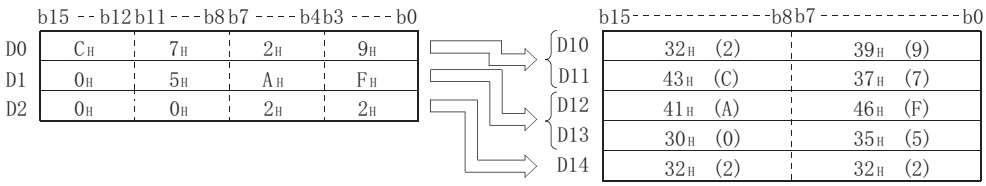
[梯形图模式]



[列表模式]

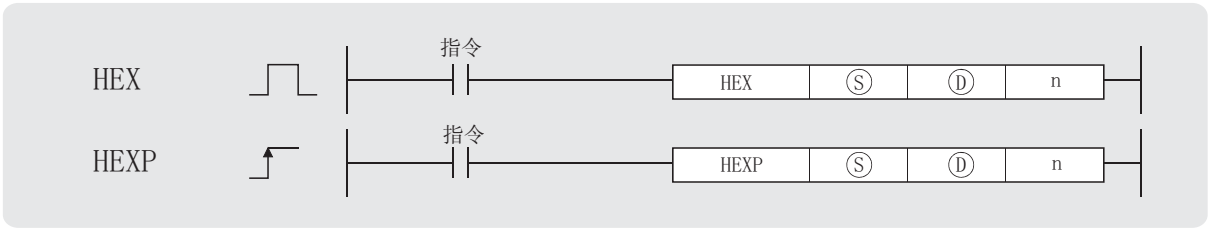
步	指令	软元件		
0	LD	X0		
1	ASCP	D0	D10	K10
5	END			

[动作]



7.11.14 ASCII 码→ 16 进制 BIN 数据的转换 (HEX(P))

Basic | High performance | Process | Redundant | Universal

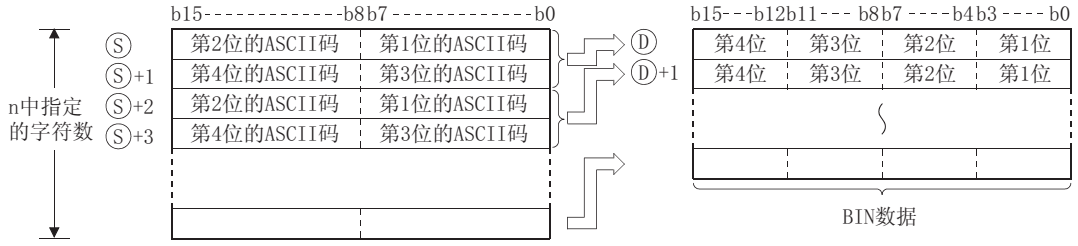


Ⓢ ：存储要转换为 BIN 数据的字符串的软元件的起始编号 (字符串)。
Ⓓ ：存储转换后的 BIN 数据的软元件的起始编号 (BIN16 位)。
n ：存储的字符数 (BIN16 位)。

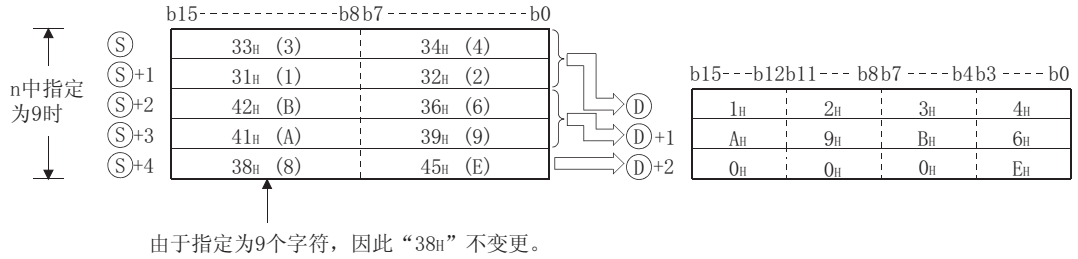
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	--	○		--					--
Ⓓ	--	○		--					--
n	○	○		○					--

功 能

(1) 将存储在Ⓢ中指定的软元件编号后面的 n 中指定的字符数的 16 进制 ASCII 数据转换为 BIN 值后，存储到Ⓓ中指定的软元件编号的后面。

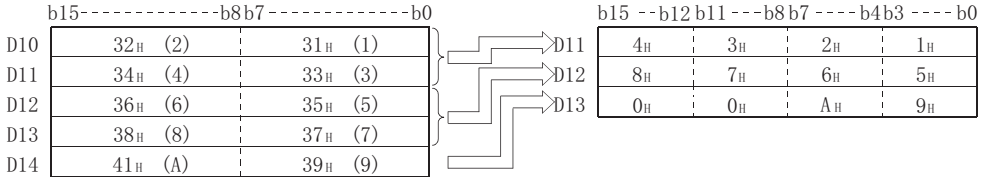


例如，n 中指定了 9 时，其情况如下所示。

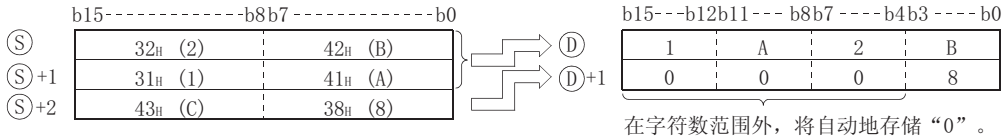


(2) 在 n 中设置了字符数时，存储Ⓢ中指定的字符串的范围以及Ⓓ中指定的 BIN 数据的软元件范围将被自动确定。

(3) 即使存储要转换的 ASCII 数据的软元件范围与存储转换后的 BIN 数据软元件范围重复时，也可正常进行处理。



(4) n 中指定的字符数不是 4 的倍数时，在存储转换后的 BIN 值的软元件编号中，最终软元件编号的指定字符数后面的位数中将自动地存储“ 0 ”。



(5) n 中指定的字符数为“ 0 ”时，不执行转换处理。

(6) Ⓢ中可指定的 ASCII 码在“ 30H ”～“ 39H ”、“ 41H ”～“ 46H ”的范围内。

出 错

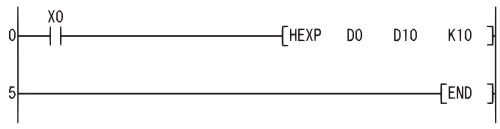
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- Ⓢ中设置了除 16 进制数值的字符串以外的字符 (“ 30H ”～“ 39H ”、“ 41H ”～“ 46H ”以外的 ASCII 码) 时。 (出错代码： 4100)
- 在Ⓢ中指定的软元件编号后面的 n 中指定字符数的范围超出了相应软元件的范围时。 (出错代码： 4101)
- 在Ⓓ中指定的软元件编号后面的 n 中指定字符数的范围超出了相应软元件的范围时。 (出错代码： 4101)
- n 中指定的字符数为负数时。 (出错代码： 4101)

程序示例

(1) 以下为 X0 变为 ON 时，将 D0 ～ D4 中存储的字符串数据转换为 BIN 数据后，存储到 D10 ～ D14 中的程序。

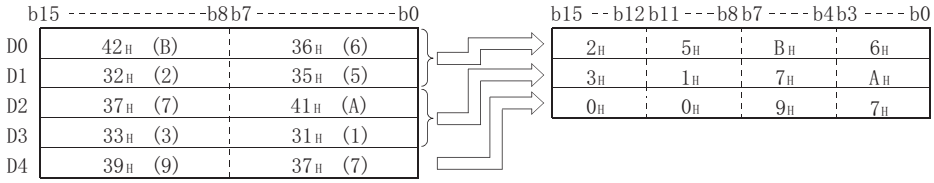
[梯形图模式]



[列表模式]

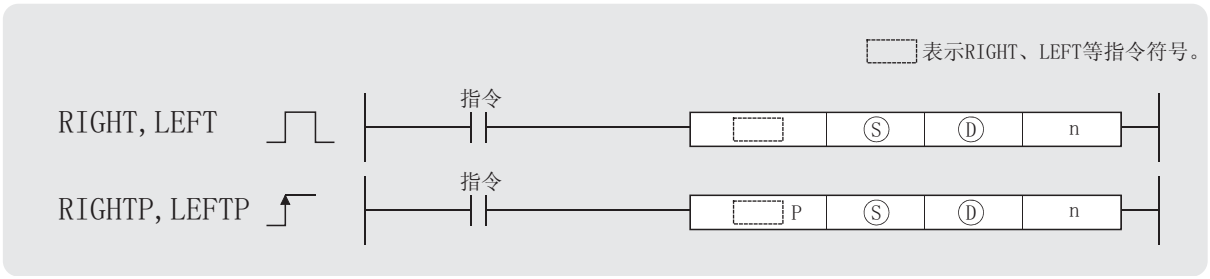
步	指令	软元件		
0	LD	X0		
1	HEXP	D0	D10	K10
5	END			

[动作]



7.11.15 从字符串的右侧或左侧提取数据 (RIGHT(P)、LEFT(P))

Basic High performance Process Redundant Universal



Ⓢ ： 字符串或者存储字符串的软元件的起始编号（字符串）。

Ⓓ ： 存储从Ⓢ的右侧或者左侧开始的 n 个字符的字符串的软元件的起始编号（字符串）。

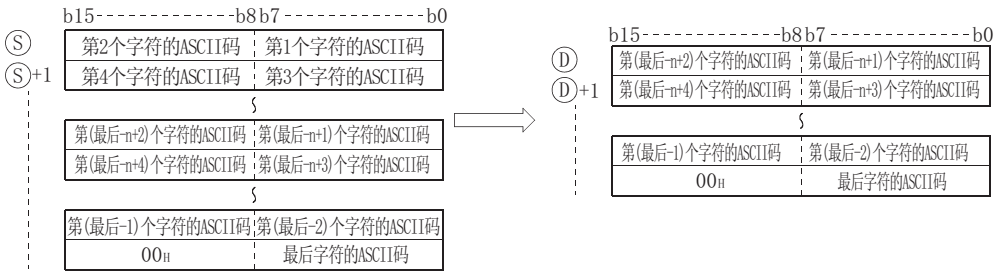
n ： 提取的字符数 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数		其它
	位	字		位	字			K、H	$	
Ⓢ	--	○		--				--	○	--
Ⓓ	--	○		--				--	--	--
n	○	○		○				○	--	--

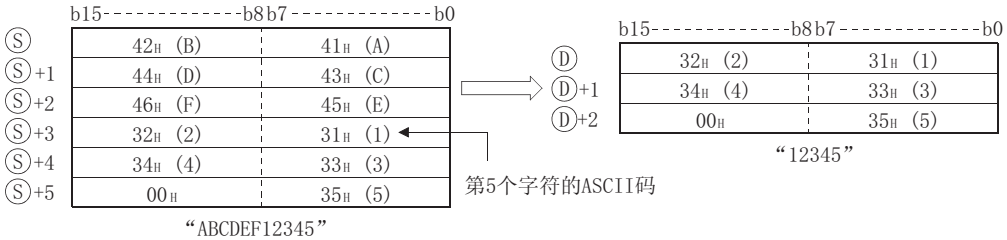
功 能

RIGHT

(1) 从Ⓢ中指定的软元件编号后面存储的字符串数据的右（字符串的最后）侧开始，将 n 个字符的数据存储到Ⓓ中指定的软元件编号的后面。



n=5 时

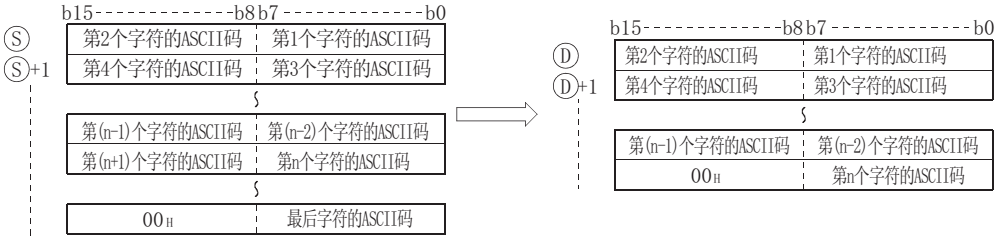


(2) 表示字符串的最后的 NULL 码 (00H) 将被自动附加到字符串的最后。
关于字符串数据的格式，请参阅 3.2.5 项。

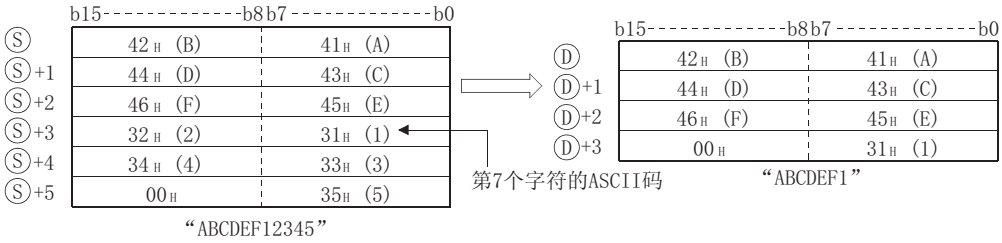
(3) n 中指定的字符数为 “ 0 ” 时，则Ⓓ中将存储 NULL 码 (00H)。

LEFT

(1) 从Ⓢ中指定的软元件编号后面存储的字符串数据的左（字符串的起始）侧开始，将 n 个字符的数据存储到Ⓓ中指定的软元件编号的后面。



n=7 时



(2) 表示字符串的最后的 NULL 码 (00H) 将被自动附加到字符串的最后。
关于字符串数据的格式，请参阅 3.2.5 项。

(3) n 中指定的字符数为 " 0 " 时，则Ⓓ中将存储 NULL 码 (00H)。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

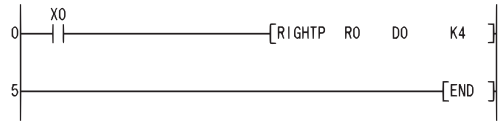
· n 超过了Ⓢ中指定的字符数时。 (出错代码 : 4101)

· 从Ⓓ开始的 n 个字符的范围超出了相应软元件的范围时。 (出错代码 : 4101)

程序示例

(1) 以下为 X0 变为 ON 时，将存储在 R0 后面的字符串数据的右侧开始的 4 个字符的数据存储到 D0 的后面。

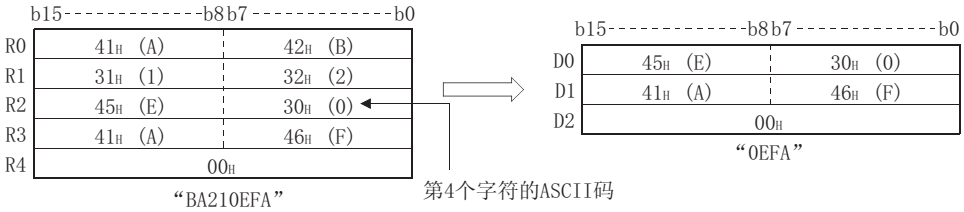
[梯形图模式]



[列表模式]

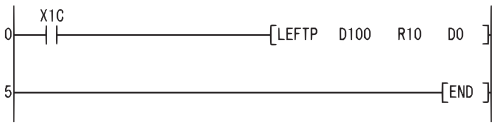
步	指令	软元件		
0	LD	X0		
1	RIGHTP	R0	D0	K4
5	END			

[动作]



(2) 以下为 X1C 变为 ON 时，从存储在 D100 后面的字符串数据的左侧开始，按 D0 中存储的字符数将数据存储到 R10 后面的程序。

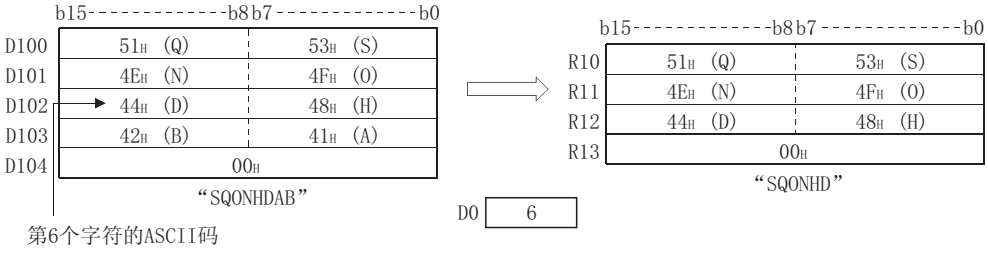
[梯形图模式]



[列表模式]

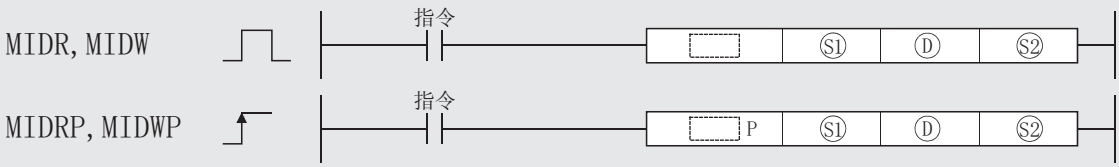
步	指令	软元件		
0	LD	X1C		
1	LEFTP	D100	R10	D0
5	END			

[动作]



7.11.16 字符串的任意提取和置换 (MIDR(P)、MIDW(P))

Basic | High performance | Process | Redundant | Universal



(S1) ：字符串或者存储字符串的软元件的起始编号（字符串）。

(D) ：存储运算结果的字符串数据的软元件的起始编号（字符串）。

(S2) ：存储起始字符的位置以及字符数的软元件的起始编号 (BIN16 位)。

·(S2)：起始字符的位置

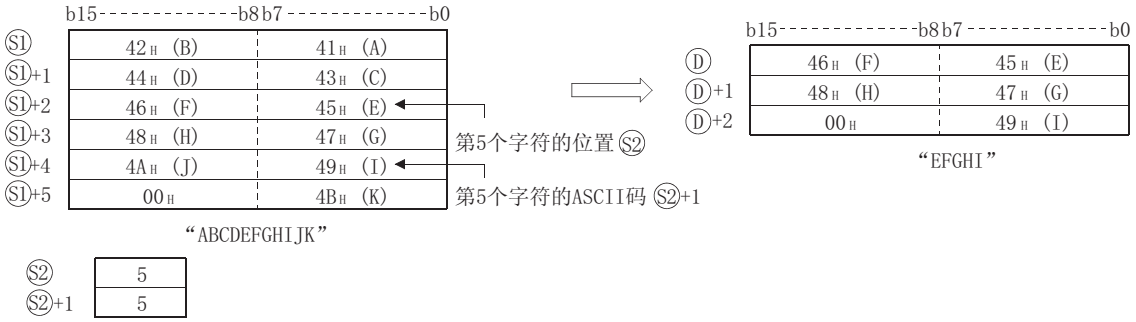
·(S2)+1：字符数

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
(S1)	--	○		--				○	--
(D)	--	○		--				--	--
(S2)	○	○		○				--	--

功 能

MIDR

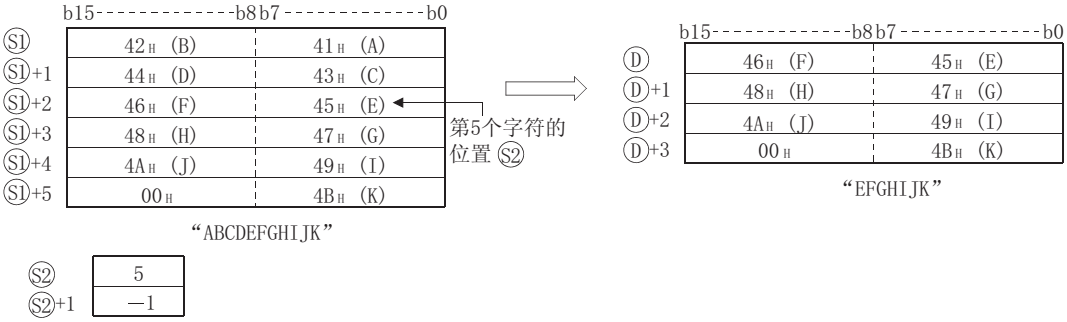
(1) 从(S1)中指定的字符串数据的左侧开始，从(S2)中指定的位置开始将(S2)+1 中指定的字符数的数据存储到(D)中指定的软元件编号的后面。



(2) 表示字符串的结尾的 NULL 码 (00H) 将被自动附加到字符串的结尾处。
关于字符串数据的格式，请参阅 3.2.5 项。

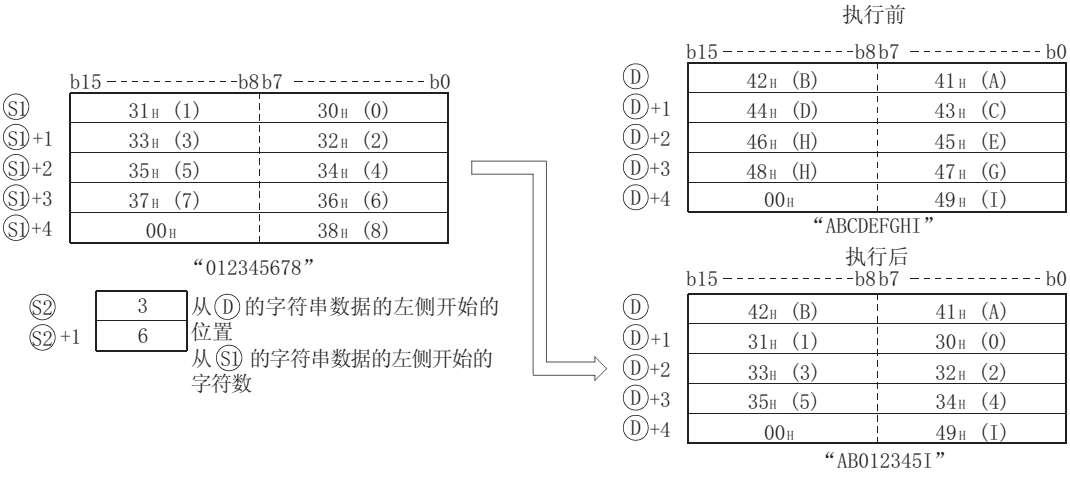
(3) (S2)+1 中指定的字符数为“0”时，不执行处理。

(4) ⓢ2+1 中指定的字符数为“ -1 ”时，将至ⓢ中指定的最后字符数据为止的数据存储到ⓓ中指定的软元件后面。



MIDW

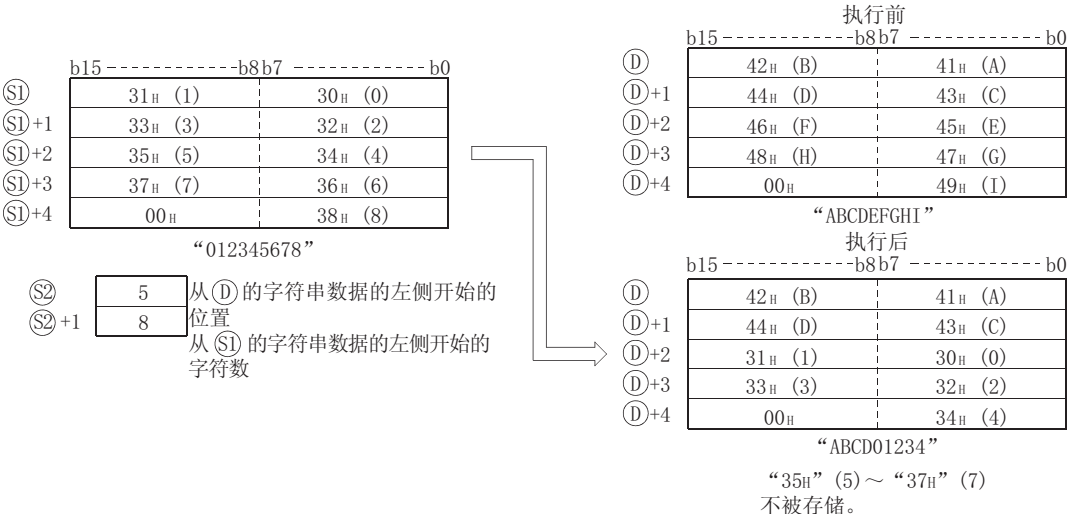
(1) 从ⓢ1 中指定的字符串数据的左侧开始，将ⓢ2+1 中指定字符数的数据，存储到从ⓓ中指定的字符串数据左侧开始的ⓢ2 中指定的位置后面。



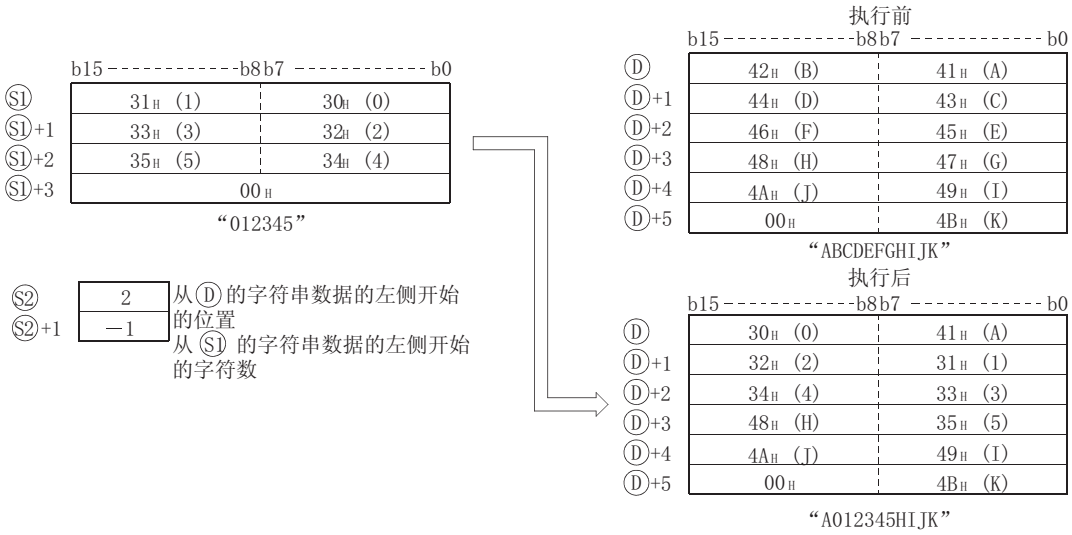
(2) 表示字符串的结尾的 NULL 码 (00H) 将被自动附加到字符串的结尾处。
关于字符串数据的格式，请参阅 3.2.5 项。

(3) ⓢ2+1 中指定的字符数为“ 0 ”时，不执行处理。

(4) ⓢ2+1 中指定的字符数超过了ⓓ中指定的字符串数据的最后字符时，存储至最后字符为止的数据。



(5) ⓢ2+1 中指定的字符数为“ -1 ”时，将至ⓢ1 中指定的最后字符数据为止的数据存储到ⓓ中指定的软元件后面。



出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

MIDR 指令时

· ⓢ2的值超过了ⓢ1的字符数时。 (出错代码： 4101)

· 从ⓓ位置开始的ⓢ2+1 的字符数超出了ⓓ的软元件的范围时。 (出错代码： 4101)

· ⓢ2+0 的值为 0 时。 (出错代码： 4101)

· ⓢ1中指定的软元件编号后面相应软元件的范围中“ 00H ”不存在时。 (出错代码： 4101)

MIDW 指令时

· ⓢ2的值超过了ⓓ的字符数时。 (出错代码： 4101)

· ⓢ2+1 的值超出了ⓢ1的字符数时。 (出错代码： 4101)

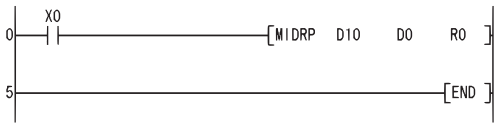
· ⓢ2+0 的值为 0 时。 (出错代码： 4101)

· ⓢ1中指定的软元件编号后面相应软元件的范围中“ 00H ”不存在时。 (出错代码： 4101)

程序示例

(1) 以下为 X0 变为 ON 时，将存储在 D10 后面的字符串数据左侧开始的第 3 个字符至第 6 个字符的数据，存储到 D0 后面的程序。

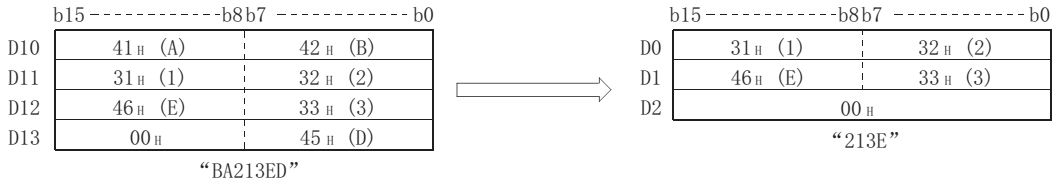
[梯形图模式]



[列表模式]

步	指令	软元件		
0	LD	X0		
1	MIDRP	D10	D0	R0
5	END			

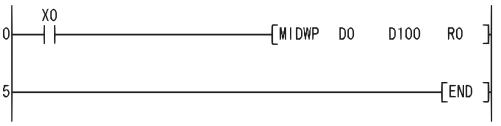
[动作]



R0	3
R1	4

(2) 以下为 X0 变为 ON 时，将存储在 D0 后面的 4 个字符的字符串数据，存储到 D100 后面的字符串数据左起第 3 个字符的后面的程序。

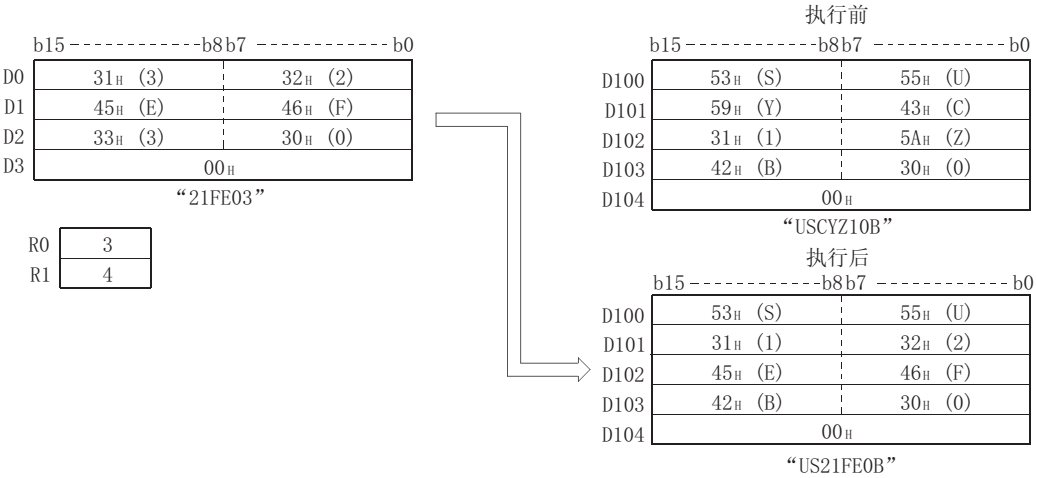
[梯形图模式]



[列表模式]

步	指令	软元件		
0	LD	X0		
1	MIDWP	D0	D100	R0
5	END			

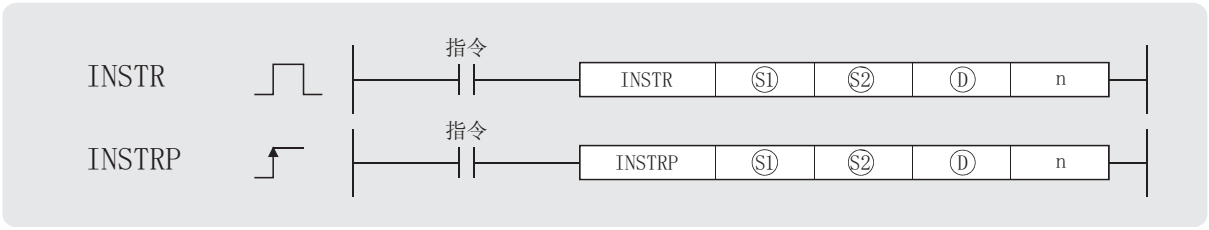
[动作]



R0	3
R1	4

7.11.17 字符串搜索 (INSTR(P))

Basic | High performance | Process | Redundant | Universal



(S1) ：要搜索的字符串或者存储搜索的字符串的软元件的起始编号（字符串）。
(S2) ：被搜索的字符串或者存储被搜索的字符串的软元件的起始编号（字符串）。
(D) ：存储搜索结果的软元件的起始编号 (BIN16 位)。
n ：搜索开始位置 (BIN16 位)。

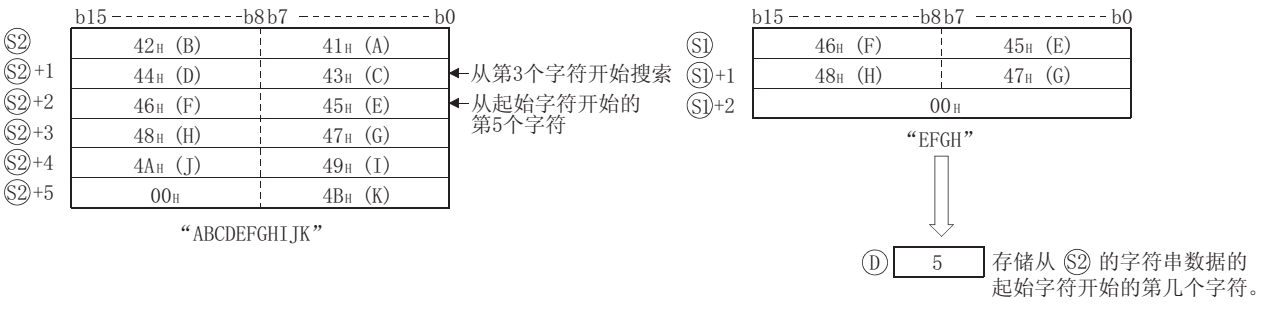
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数		其它
	位	字		位	字			K、H	$	
(S1)	--	○		--				--	○	--
(S2)	--	○		--				--	○	--
(D)	○	○		○				--	--	--
n	○	○		○				○	--	--

功 能

(1) 从(S2)中指定的字符串数据左起第 n 个字符开始，对(S1)中指定的字符串数据进行搜索，并将搜索结果存储到(D)中指定的软元件中。

搜索结果中存储的是(S2)中指定的字符串数据的起始字符开始的第几个字符。

n=3 时



(2) 如果没有一致的字符串数据，将在(D)中存储“0”。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· n 的值超过了Ⓢ2的字符数时。 (出错代码： 4100)

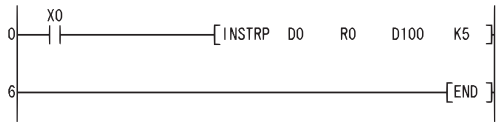
· Ⓢ1、Ⓢ2 中指定的软元件后面相应软元件范围内没有 00H(NULL) 时。
(出错代码： 4100)

· n 的值为负数或者 “ 0 ” 时。 (出错代码： 4100)

程序示例

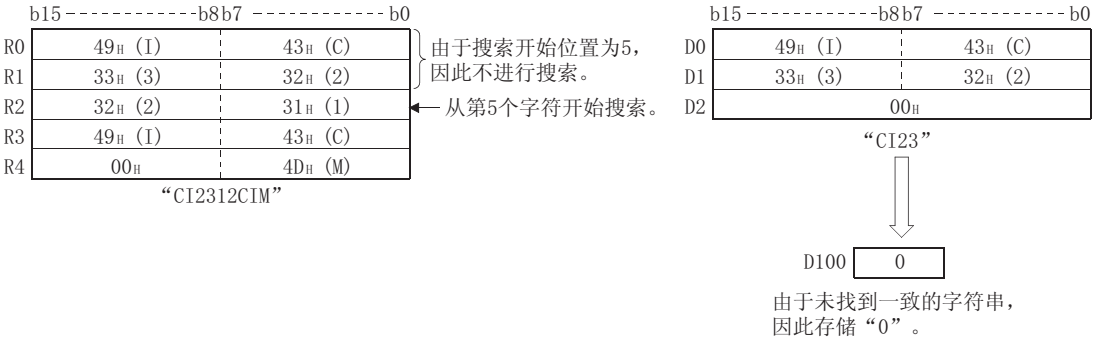
(1) 以下为 X0 变为 ON 时，将 D0 后面存储的字符串数据，从 R0 后面存储的字符串数据的左起第 5 个字符开始进行搜索，并将搜索结果存储到 D100 中的程序。

[梯形图模式] [列表模式]



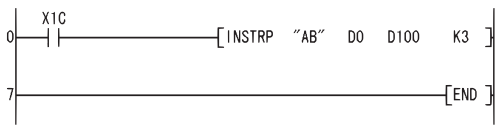
步	指令	软元件			
0	LD	X0			
1	INSTRP	D0	R0	D100	K5
6	END				

[动作]



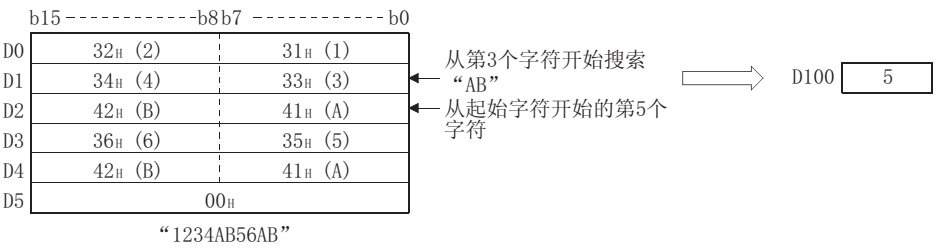
(2) 以下为 X1C 变为 ON 时，从存储在 D0 后面的字符串数据左起第 3 个字符开始对字符串数据 “ AB ” 进行搜索，并将结果存储到 D100 中的程序。

[梯形图模式] [列表模式]



步	指令	软元件			
0	LD	X1C			
1	INSTRP	“AB”	D0	D100	K3
7	END				

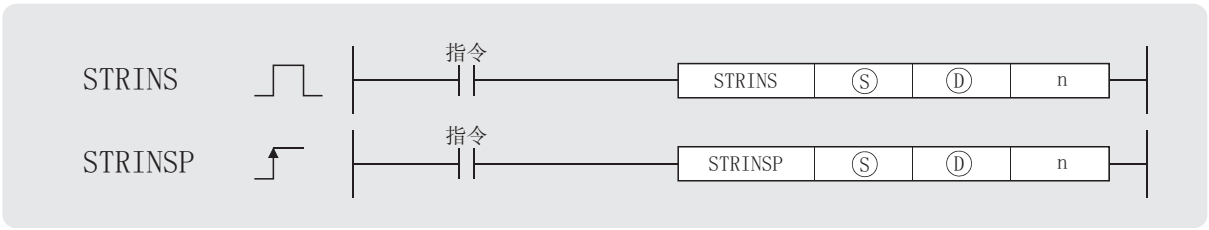
[动作]



7.11.18 字符串插入 (STRINS(P))

Basic / High performance / Process / Redundant / Universal (Ver.)

· QnU(D)(H)CPU：序列号的前 5 位数为 “ 10102 ” 以后
· QnUDE(H)CPU：序列号的前 5 位数为 “ 10102 ” 以后



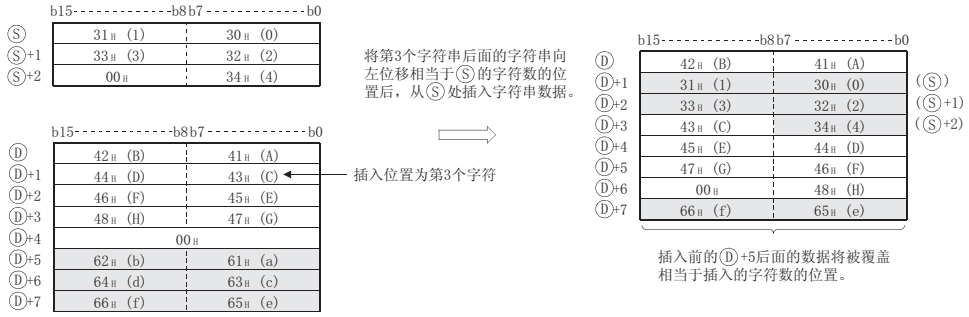
Ⓢ ：插入的字符串或者存储插入字符串的软元件的起始编号（字符串）。
Ⓓ ：存储被插入字符串的软元件的起始编号（字符串）。
n ：插入位置（设置范围 1 ≤ n ≤ 16383)(BIN16 位）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数		其它
	位	字		位	字			K、H	$	
Ⓢ	--	○		--				--	○	--
Ⓓ	--	○		--				--	--	--
n	--	○		○				○	--	--

功 能

(1) 将Ⓢ中指定的字符串数据，插入到Ⓓ中指定的字符数据从起始字符算起的第 n 个字符（插入位置）处。

插入位置 n=3 时



(2) 插入后的字符串（Ⓢ+Ⓓ）为偶数时，字符串的末尾的下一个软元件（1 字）中将存储 NULL 码 (00H)。

(3) 插入后的字符串（Ⓢ+Ⓓ）为奇数时，在字符串的末尾软元件（高 8 位）中将存储 NULL 码 (00H)。

(4) n 中指定了Ⓓ的字符数 +1 时，Ⓢ的字符串将被合并到Ⓓ的字符串的末尾处。

出 错

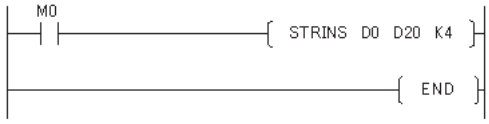
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- 字符串Ⓢ、字符串Ⓓ或者插入后的字符串 (Ⓢ+Ⓓ) 的字符数超过了 16383 个字符时。 (出错代码： 4100)
- n 超出了范围时。(1 ≤ n ≤ 16383) (出错代码： 4100)
- n 中指定的值超过了字符串Ⓓ的字符数 +1 时。 (出错代码： 4100)
- 字符串Ⓢ与字符串Ⓓ的软元件有部分重复时。 (出错代码： 4101)
- 插入后的字符串 (Ⓢ+Ⓓ) 超出了指定软元件的范围时。 (出错代码： 4101)
- Ⓢ、Ⓓ中指定的软元件后面，指定软元件范围内没有 NULL 码 (00H) 时。 (出错代码： 4101)
- 插入后的字符串 (Ⓢ+Ⓓ) 与存储Ⓢ的字符串的软元件重复时。 (出错代码： 4101)

程序示例

(1) 以下为 M0 变为 ON 时，将软元件 D0 后面存储的字符串数据插入到软元件 D20 后面的字符串数据从起始算起的第 4 个字符处的程序。

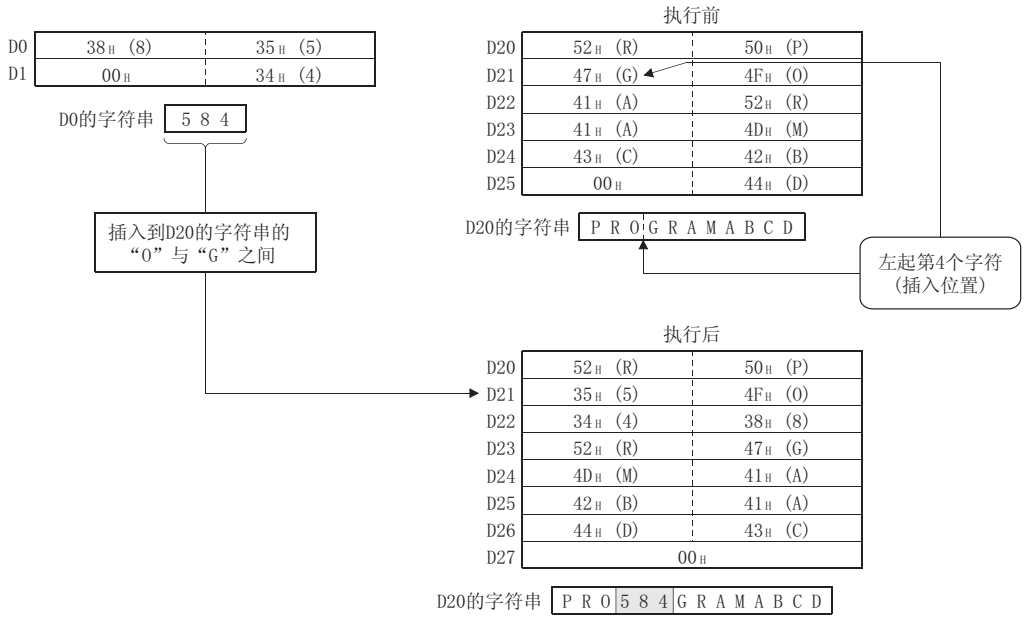
[梯形图模式]



[列表模式]

步	指令	软元件
0	LD	M0
1	STRINS	D0 D20 K4
5	END	

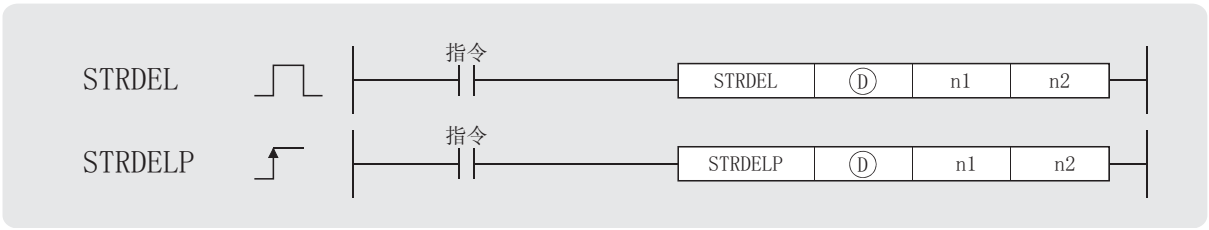
[动作]



7.11.19 字符串删除 (STRDEL(P))

Basic　High performance　Process　Redundant　Universal

· QnU(D)(H)CPU：序列号的前 5 位数为“ 10102 ”以后
· QnUDE(H)CPU：序列号的前 5 位数为“ 10102 ”以后

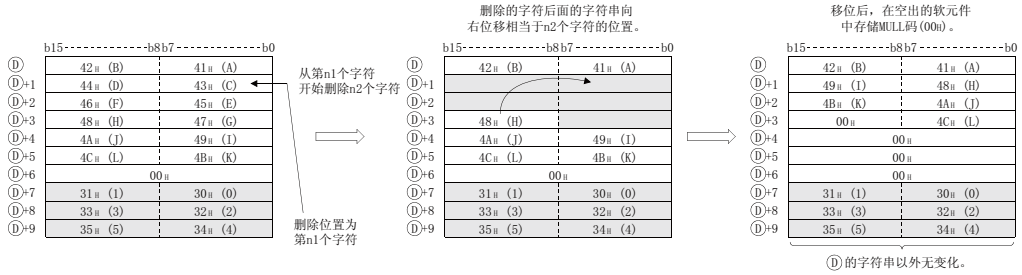


Ⓓ ：存储删除字符串的软元件的起始编号（字符串）。
n1 ：删除开始位置（设置范围 1 ≤ n1 ≤ 16383)(BIN16 位）。
n2 ：删除字符数（设置范围 0 ≤ n2 ≤ 16384-n1)(BIN16 位）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			K、H	
Ⓓ	--	○		--				--	--
n1	--	○		○				○	--
n2	--	○		○				○	--

功　能

(1) 从Ⓓ中指定的字符串数据从起始算起的第 n1 个字符位置（删除开始位置）开始，删除 n2 个字符。
删除位置： n1=3
删除字符数： n2=5 时



(2) 删除后，字符串Ⓓ为偶数时，在字符串的末尾的下一个软元件 (1 字）中将存储 NULL 码 (00H)。

(3) 删除后，字符串Ⓓ为奇数时，在字符串的末尾软元件（高 8 位）中将存储 NULL 码 (00H)。

(4) 删除后的字符串后面的字符串向右移位 n2 个字符后，在空出的软软元件中存储 MULL 码 (00H)。

出 错

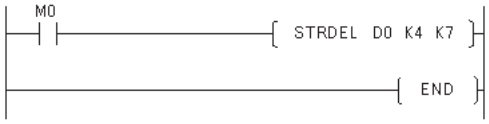
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- 字符串ⓓ的字符数超过了 16383 个字符时。 (出错代码： 4100)
- n1 超出了范围时。(1 ≤ n1 ≤ 16383) (出错代码： 4100)
- n1 中指定的值超过了字符串ⓓ的字符数时。 (出错代码： 4100)
- n2 中指定的值超过了从字符串ⓓ的 n1 开始至最后字符为止的字符数时。 (出错代码： 4100)
- n2 中指定的值为负数时。 (出错代码： 4100)

程序示例

(1) 以下为 M0 变为 ON 时，从软元件 D0 后面存储的字符串数据的第 4 个字符开始删除 7 个字符数据的程序。

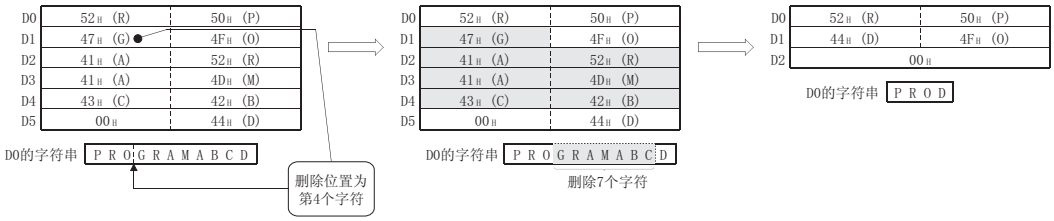
[梯形图模式]



[列表模式]

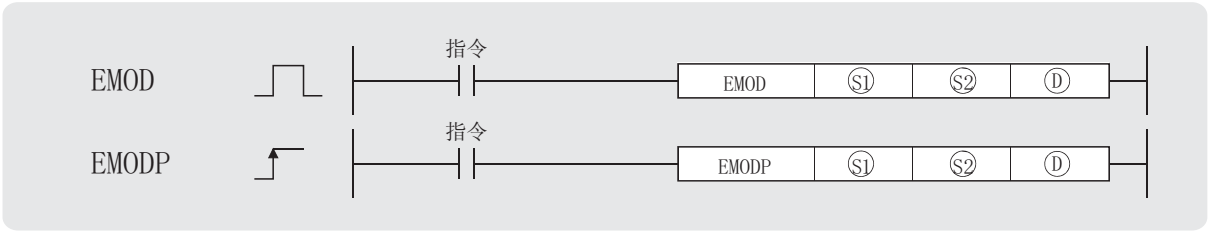
步	指令	软元件
0	LD	M0
1	STRDEL	D0 K4 K7
5	END	

[动作]



7.11.20 浮点数→ BCD 的分解 (EMOD(P))

Basic | High performance | Process | Redundant | Universal



S1 ：32 位浮点实数数据或者存储浮点实数数据的软元件的起始编号 (实数)。

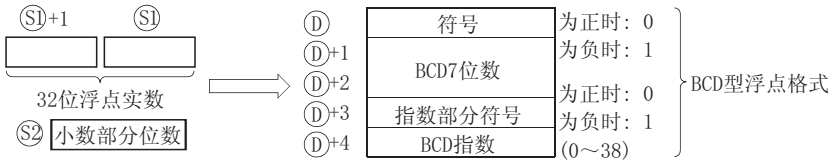
S2 ：小数部分位数数据 (BIN16 位)。

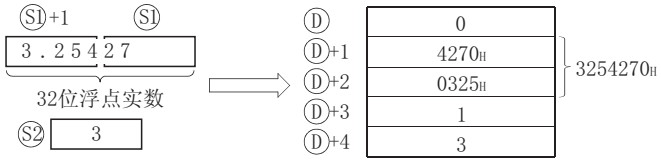
D ：存储进行了 BCD 分解的数据的软元件的起始编号 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数		其它
	位	字		位	字			K、H	E	
S1	--	○		--	○		--		○	--
S2	○	○		○	○		○		--	--
D	--	○		--	--		--		--	--

功 能

(1) 将S1中指定的 32 位浮点实数数据，根据S2中指定的小数部分位数分解为 BCD 型浮点格式后，存储到D中指定的软元件编号后面。

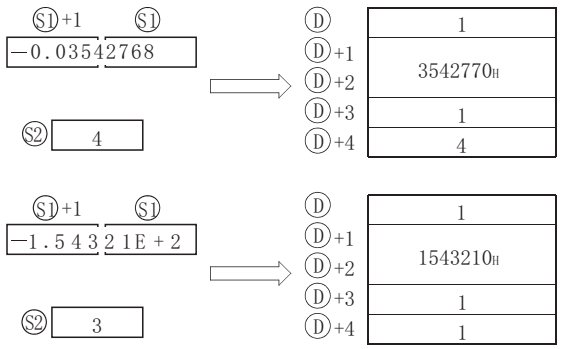




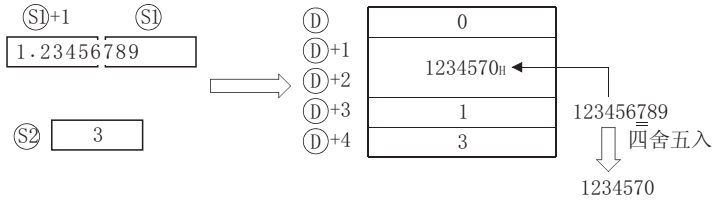
S2是用于指定S1的 32 位浮点实数数据的小数部分位数。在上图示例中的情况如下所示。

3.25427

S2=3



(2) (D)+1、(D)+2 中存储的 BCD 的有效位数为将第 7 位进行四舍五入后的 6 位数。



出　错

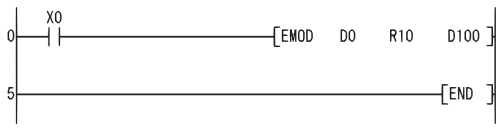
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- (S2) 中指定的小数部分位数超出了 0 ～ 7 的范围时。 (出错代码： 4100)
- (D) 中指定的软元件范围超出了相应软元件的范围时。 (出错代码： 4101)
- (S1) 中指定的 32 位浮点实数不在下述范围内时。
  0，$2^{-126} \leqq$ | 软元件 | $< 2^{128}$ (出错代码： 4100)
- (D) 中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)
  (出错代码： 4101)
- 指定的软元件的内容为 -0、非正规数、非数、±∞时。(仅通用型 QCPU)
  (出错代码： 4140)

程序示例

(1) 以下为 X0 变为 ON 时，将 D0、D1 中存储的 32 位浮点实数数据，根据 R10 中存储的值的小数部分位数分解为 BCD 后，存储到 D100 后面的程序。

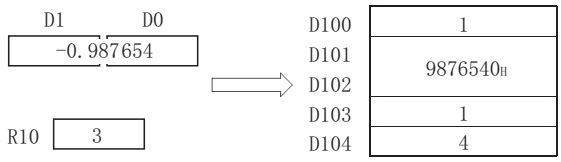
[梯形图模式]



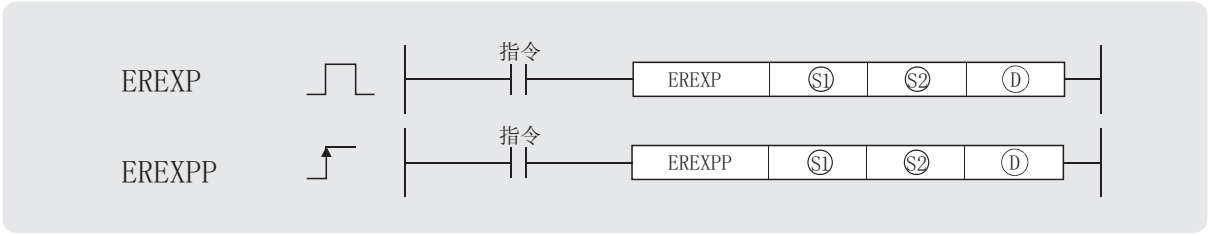
[列表模式]

步	指令	软元件		
0	LD	X0		
1	EMOD	D0	R10	D100
5	END			

[动作]



7.11.21 BCD 格式数据→浮点数的转换 (EREXP(P))



S1 ：存储 BCD 型浮点格式数据的软元件的起始编号 (BIN16 位)。

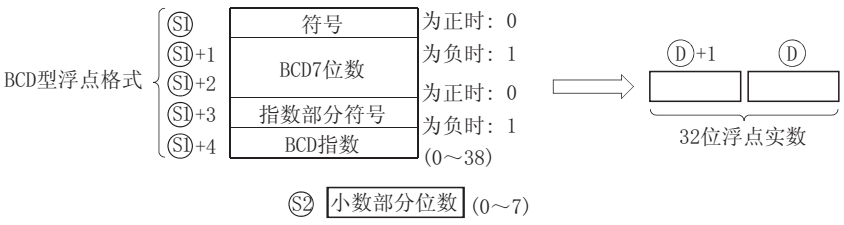
S2 ：小数部分位数数据 (BIN16 位)。

D ：存储 32 位浮点实数数据的软元件 (实数)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
S1	--	○		--	--		--		--
S2	○	○		○	○		○		--
D	--	○		--	○		--		--

功 能

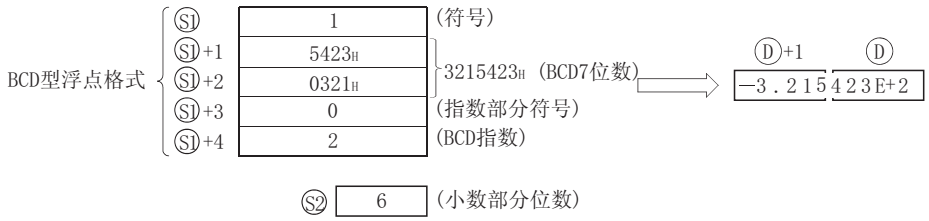
(1) 将S1中指定的 BCD 型浮点格式数据，根据S2中指定的小数部分位数转换为 32 位浮点实数数据后，存储到D中指定的软元件编号后面。



(2) 在S1的符号及S1+3 的指数部分符号中，为正时设置为 0，为负时设置为 1。

(3) S1+4 的 BCD 指数中，可设置范围为 0 ～ 38。

(4) 在S2的小数部分位数中，可设置范围为 0 ～ 7。



出 错

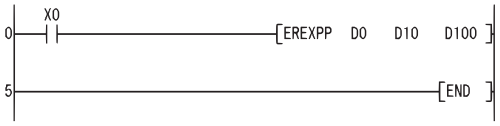
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- ⑤1 中指定的格式指定为 0 及 1 以外时。 (出错代码： 4100)
- ⑤1+1、⑤1+2 的各位数中存在有除 0 ～ 9 以外的值时。 (出错代码： 4100)
- ⑤1+3 中指定的格式指定为 0 及 1 以外时。 (出错代码： 4100)
- ⑤1+4 中指定的指数数据超出了 0 ～ 38 的范围时。 (出错代码： 4100)
- ⑤2 中指定小数部分位数超出了 0 ～ 7 的范围时。 (出错代码： 4100)
- ⑤1 中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)
  (出错代码： 4101)

程序示例

(1) 以下为 X0 变为 ON 时，将 D0 后面存储的 BCD 型浮点格式数据，根据 D10 中存储的小数部分位数转换为 32 位浮点实数数据后，存储到 D100、D101 中的程序。

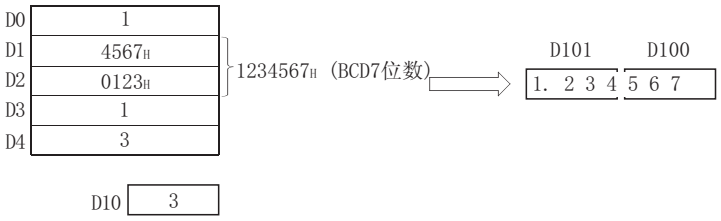
[梯形图模式]



[列表模式]

步	指令	软元件		
0	LD	X0		
1	EREXPP	D0	D10	D100
5	END			

[动作]

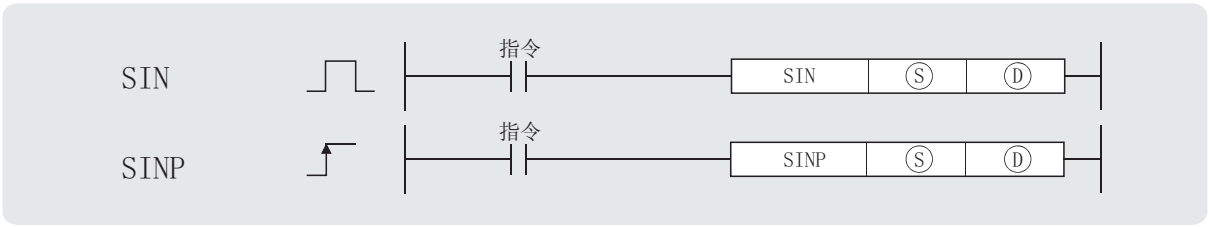


7.12 特殊函数指令

7.12.1 浮点数的 SIN 运算 (单精度)(SIN(P))

Ver.
Basic | High performance | Process | Redundant | Universal

基本型 QCPU：序列号的前 5 位数为 “ 04122 ” 以后



Ⓢ ： 进行 SIN(正弦) 运算的角度数据或者存储角度数据的软元件的起始编号 (实数)。

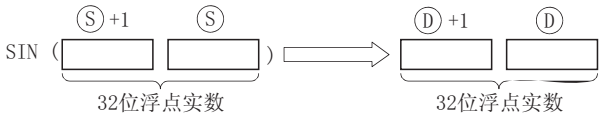
Ⓓ ： 存储运算结果的软元件的起始编号 (实数)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
Ⓢ	--	○		--	○		○ *1	○	--
Ⓓ	--	○		--	○		○ *1	--	--

*1：仅通用型 QCPU 可以使用。

功 能

(1) 对Ⓢ中指定的角度值进行 SIN(正弦) 运算后，将运算结果存储到Ⓓ中指定的软元件中。



(2) Ⓢ中指定的角度是以弧度单位 (角度×π÷180) 设置的。
关于角度↔弧度的转换，请参阅 RAD 指令、DEG 指令。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 指定软元件的内容为 -0 时。*2
(基本型 QCPU、高性能型 QCPU、过程 CPU、冗余 CPU 时) (出错代码： 4100)

*2:有的 CPU 模块即使指定为 -0 也不会变为运算出错状态。
有关详细内容请参阅 3.2.4 项。

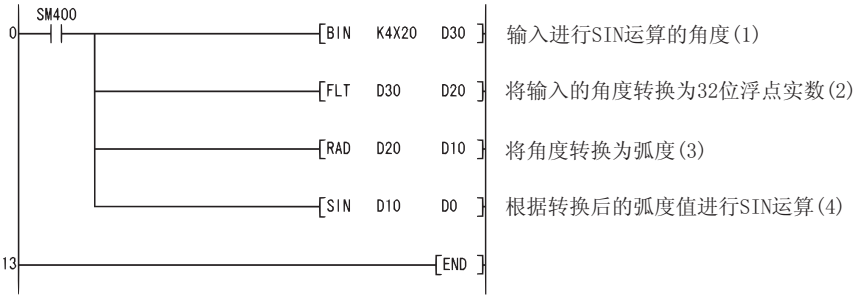
· 运算结果超出了下述范围时。(发生了溢出时。)(仅通用型 QCPU)
$2^{128} \leqq$ | 运算结果 | (出错代码： 4141)

· 指定软元件的内容为 -0、非正规数、非数、±∞时。(仅通用型 QCPU)
(出错代码： 4140)

程序示例

(1) 以下为对 X20 ～ X2F 中以 BCD4 位数设置的角度进行 SIN 运算后，以 32 位浮点实数格式存储到 D0、D1 中的程序。

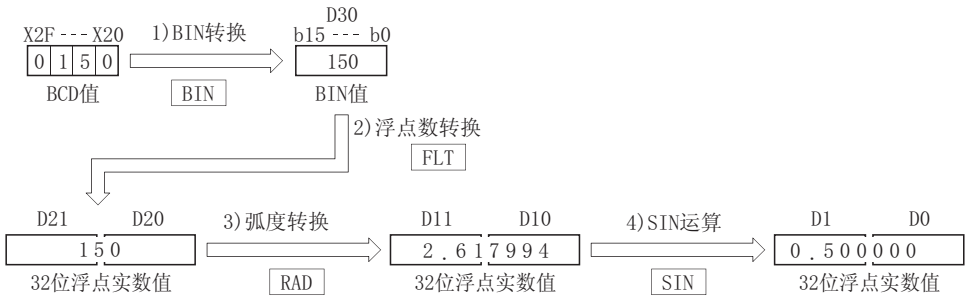
[梯形图模式]



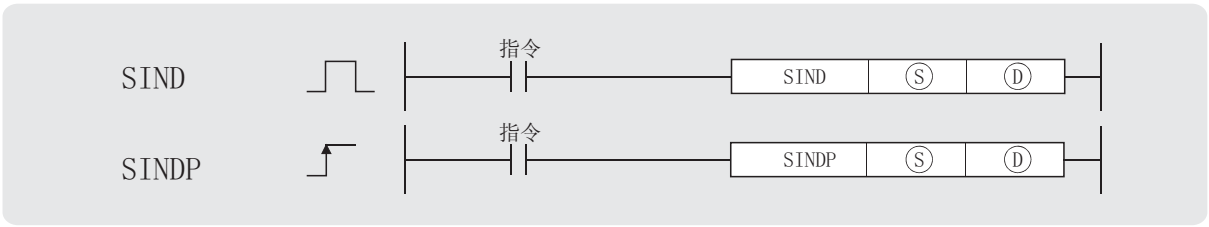
[列表模式]

步	指令	软元件	
0	LD	SM400	
1	BIN	K4X20	D30
4	FLT	D30	D20
7	RAD	D20	D10
10	SIN	D10	D0
13	END		

[在 X20 ～ X2F 中指定了 150 时的动作]



7.12.2 浮点数的 SIN 运算 (双精度)(SIND(P))



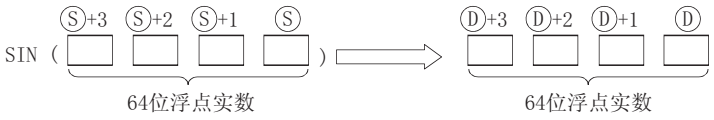
(S) : 进行 SIN(正弦) 运算的角度数据或者存储角度数据的软元件的起始编号 (实数)。

(D) : 存储运算结果的软元件的起始编号 (实数)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
(S)	--	○		--				○	--
(D)	--	○		--				--	--

功 能

(1) 对(S)中指定的角度值进行 SIN(正弦) 运算后，将运算结果存储到(D)中指定的软元件中。



(2) (S)中指定的角度是以弧度单位 (角度×π÷180) 设置的。
关于角度↔弧度的转换，请参阅 RADD 指令、DEGD 指令。

(3) 运算结果为 -0 或者发生了下溢时，将运算结果作为 0 进行处理。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 指定软元件的内容不在下述范围内时。 (出错代码 : 4140)

$0, 2^{-1022} \leqq |$ 指定软元件的内容 $| < 2^{1024}$

· 指定软元件的内容为 -0 时。 (出错代码 : 4140)

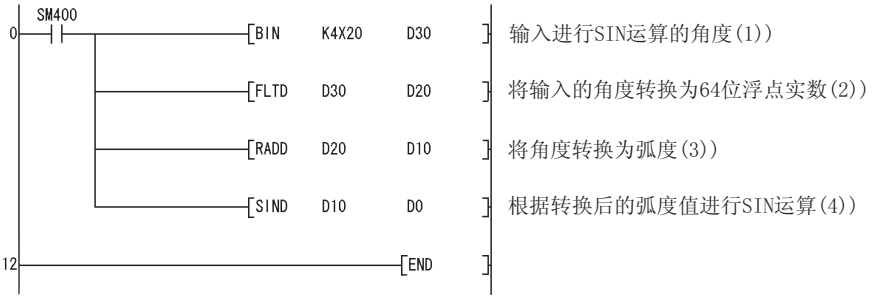
· 运算结果超出了下述范围时。(发生了溢出时。)(仅通用型 QCPU)

$2^{1024} \leqq |$ 运算结果 $|$ (出错代码 : 4141)

程序示例

(1) 以下为对 X20 ～ X2F 中以 BCD4 位数设置的角度进行 SIN 运算后，以 64 位浮点实数格式存储到 D0 ～ D3 中的程序。

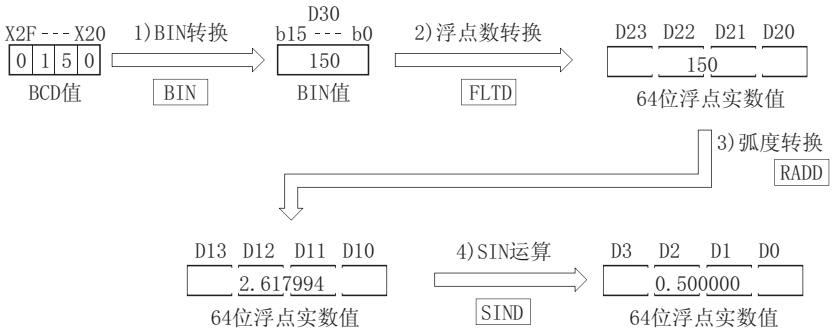
[梯形图模式]



[列表模式]

步	指令	软元件	
0	LD	SM400	
1	BIN	K4X20	D30
3	FLTD	D30	D20
6	RADD	D20	D10
9	SIND	D10	D0
12	END		

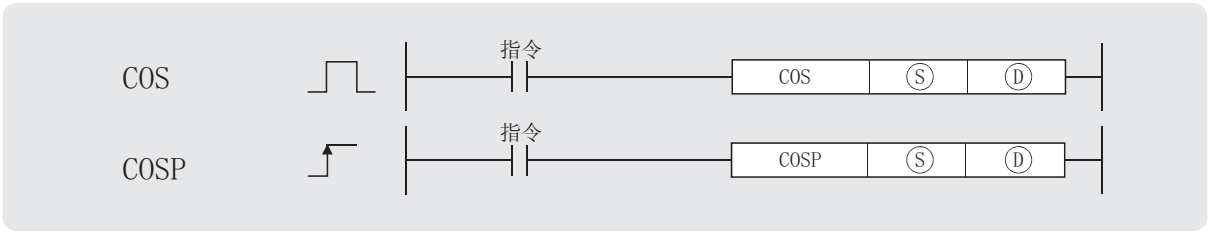
[在 X20 ～ X2F 中指定了 150 时的动作]



7.12.3 浮点数的 COS 运算（单精度）(COS(P))

Ver. Basic | High performance | Process | Redundant | Universal

基本型 QCPU：序列号的前 5 位数为“ 04122 ”以后



Ⓢ ：进行 COS(余弦) 运算的角度数据或者存储角度数据的软元件的起始编号 (实数)。

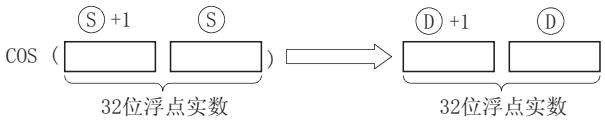
Ⓓ ：存储运算结果的软元件的起始编号 (实数)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
Ⓢ	--	○		--	○		○*1	○	--
Ⓓ	--	○		--	○		○*1	--	--

*1：仅通用型 QCPU 可以使用。

功 能

(1) 对Ⓢ中指定的角度值进行 COS(余弦) 运算后，将运算结果存储到Ⓓ中指定的软元件中。



(2) Ⓢ中指定的角度是以弧度单位 (角度×π÷ 180) 设置的。

关于角度↔弧度的转换，请参阅 RAD 指令、DEG 指令。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 指定软元件的内容为 -0 时。*2

(基本型 QCPU、高性能型 QCPU、过程 CPU、冗余 CPU 时) (出错代码： 4100)

*2:有的 CPU 模块即使指定为 -0 也不会变为运算出错状态。
有关详细内容请参阅 3.2.4 项。

· 运算结果超出了下述范围时。(发生了溢出时。)(仅通用型 QCPU)

$2^{128} \leqq |$ 运算结果 $|$ (出错代码： 4141)

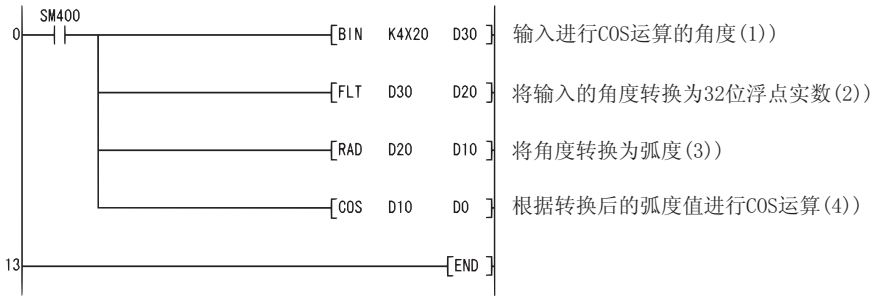
· 指定软元件的内容为 -0、非正规数、非数、±∞时。(仅通用型 QCPU)

(出错代码： 4140)

程序示例

(1) 以下为对 X20 ～ X2F 中以 BCD4 位数设置的角度进行 COS 运算后，以 32 位浮点实数格式存储到 D0、D1 中的程序。

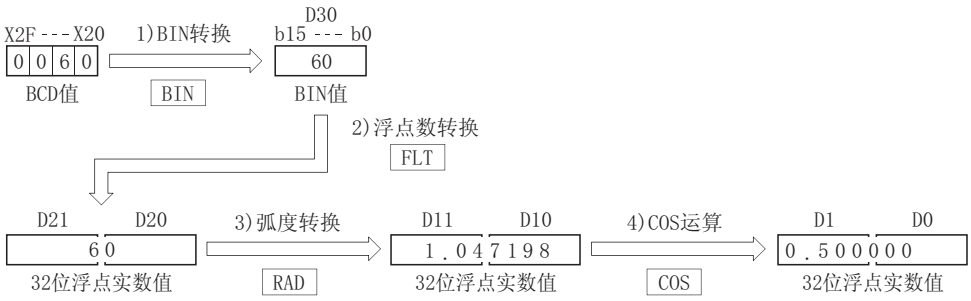
[梯形图模式]



[列表模式]

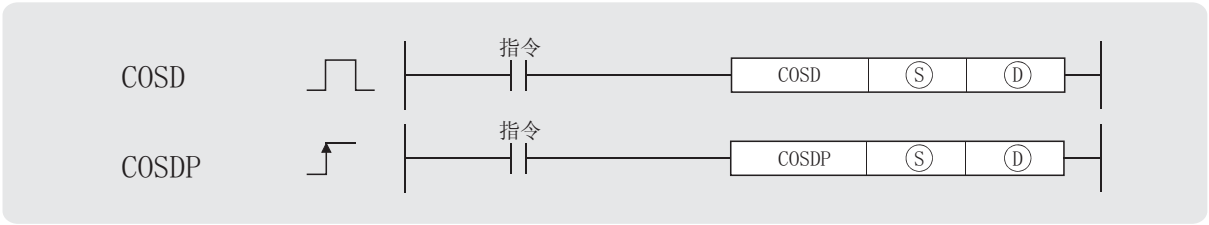
步	指令	软元件	
0	LD	SM400	
1	BIN	K4X20	D30
4	FLT	D30	D20
7	RAD	D20	D10
10	COS	D10	D0
13	END		

[在 X20 ～ X2F 中指定了 60 时的动作]



7.12.4 浮点数的 COS 运算（双精度）(COSD(P))

Basic　High performance　Process　Redundant　Universal



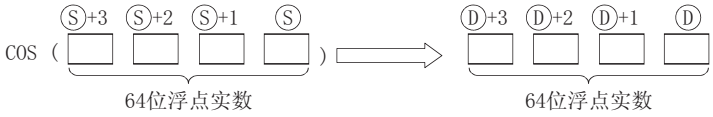
Ⓢ：进行 COS（余弦）运算的角度数据或者存储角度数据的软元件的起始编号（实数）。

Ⓓ：存储运算结果的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
Ⓢ	--	○		--				○	--
Ⓓ	--	○		--				--	--

功　能

(1) 对Ⓢ中指定的角度值进行 COS（余弦）运算后，将运算结果存储到Ⓓ中指定的软元件中。



(2) Ⓢ中指定的角度是以弧度单位（角度×π÷180）设置的。
关于角度↔弧度的转换，请参阅 RADD 指令、DEGD 指令。

(3) 运算结果为 -0 或者发生了下溢时，将运算结果作为 0 进行处理。

出　错

(1) 在以下情况下将变为运算出错状态，出错标志（SM0）将变为 ON，出错代码将被存储到 SD0 中。

· 指定软元件的内容不在下述范围内时。（出错代码：4140）
$0，2^{-1022} \leqq |指定软元件的内容| < 2^{1024}$

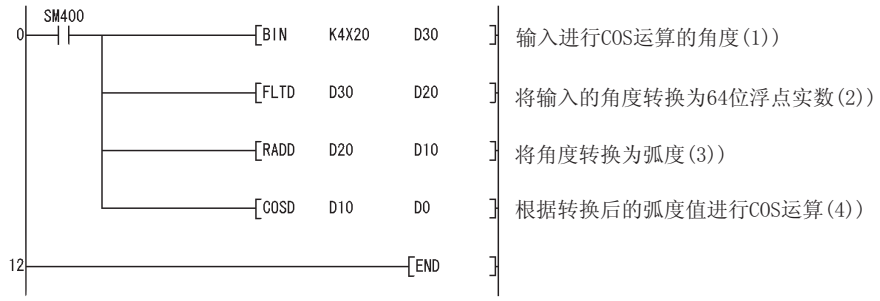
· 指定软元件的内容为 -0 时。（出错代码：4140）

· 运算结果超出了下述范围时。（发生了溢出时。）（仅通用型 QCPU）
$2^{1024} \leqq |运算结果|$（出错代码：4141）

程序示例

(1) 以下为对 X20 ～ X2F 中以 BCD4 位数设置的角度进行 COS 运算后，以 64 位浮点实数格式存储到 D0 ～ D3 中的程序。

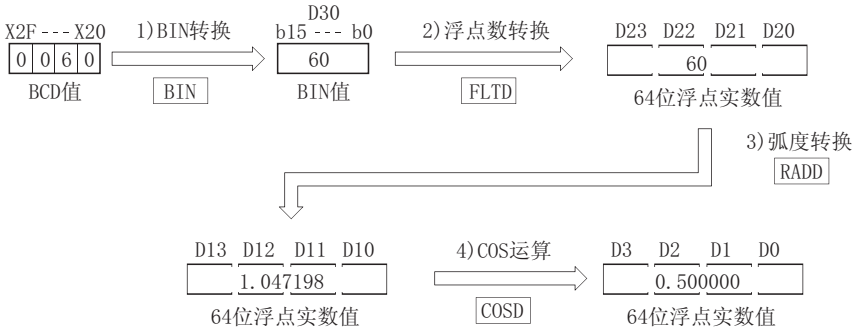
[梯形图模式]



[列表模式]

步	指令	软元件	
0	LD	SM400	
1	BIN	K4X20	D30
3	FLTD	D30	D20
6	RADD	D20	D10
9	COSD	D10	D0
12	END		

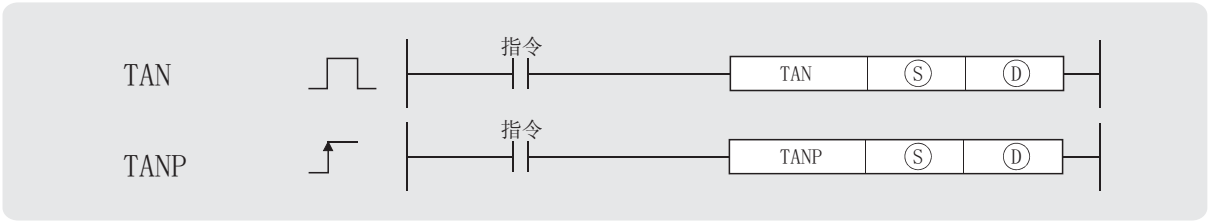
[在 X20 ～ X2F 中指定了 60 时的动作]



7.12.5 浮点数的 TAN 运算（单精度）(TAN(P))

Ver. Basic | High performance | Process | Redundant | Universal

基本型 QCPU：序列号的前 5 位数为“ 04122 ”以后



Ⓢ ：进行 TAN（正切）运算的角度数据或者存储角度数据的软元件的起始编号（实数）。

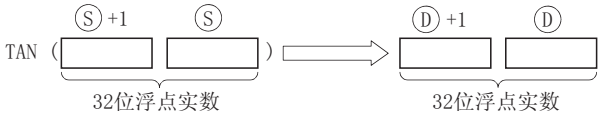
Ⓓ ：存储运算结果的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
Ⓢ	--	○		--	○		○*1	○	--
Ⓓ	--	○		--	○		○*1	--	--

*1：仅通用型 QCPU 可以使用。

功 能

(1) 对Ⓢ中指定的角度值进行 TAN（正切）运算后，将运算结果存储到Ⓓ中指定的软元件中。



(2) Ⓢ中指定的角度是以弧度单位（角度×π÷180）设置的。
关于角度↔弧度的转换，请参阅 RAD 指令、DEG 指令。

(3) Ⓢ中指定的角度为π /2 弧度、(3/2) π弧度时，将会产生弧度值运算误差，且不会变为出错状态，应加以注意。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 指定的软元件的内容不在以下范围内时。

$0, 2^{-126} \leqq |$指定软元件的内容$| < 2^{128}$

(对于基本型 QCPU、高性能型 QCPU、过程 CPU、冗余 CPU) (出错代码： 4100)

· 指定软元件的内容为 -0 时。*2

(对于基本型 QCPU、高性能型 QCPU、过程 CPU、冗余 CPU) (出错代码： 4100)

*2:有的 CPU 模块即使指定了 -0 也不会变为运算出错状态。
有关详细内容请参阅 3.2.4 项。

· 运算结果超出以下范围时。(发生了溢出时。) (仅通用型 QCPU)

$2^{128} \leqq |$运算结果$|$ (出错代码： 4141)

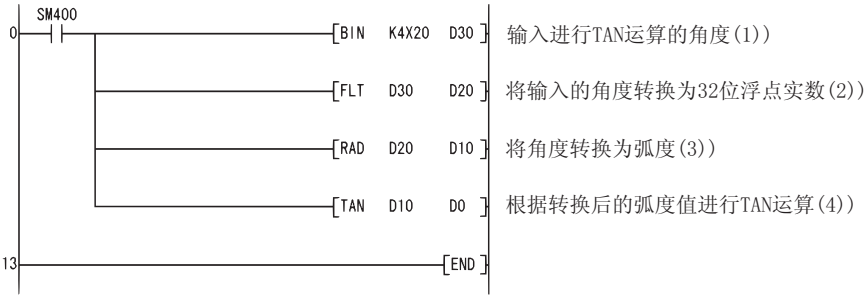
· 指定软元件的内容为 -0、非正规数、非数、±∞时。

(仅通用型 QCPU) (出错代码： 4140)

程序示例

(1) 以下为对 X20 ～ X2F 中以 BCD4 位数设置的角度进行 TAN 运算后，以 32 位浮点实数格式存储到 D0、D1 中的程序。

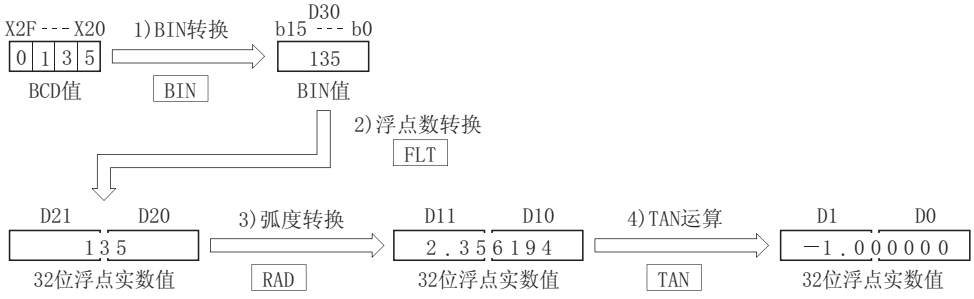
[梯形图模式]



[列表模式]

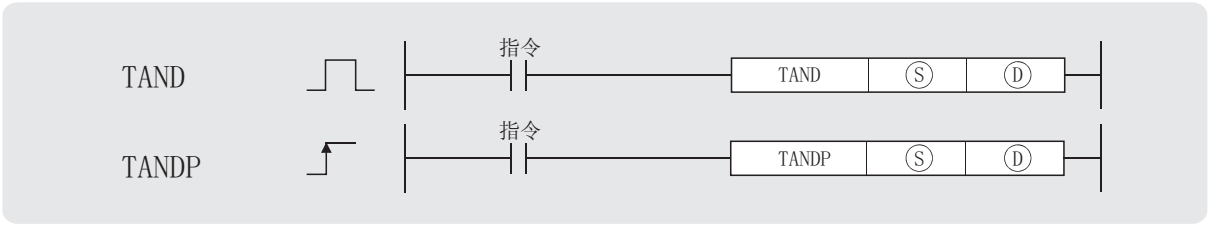
步	指令	软元件	
0	LD	SM400	
1	BIN	K4X20	D30
4	FLT	D30	D20
7	RAD	D20	D10
10	TAN	D10	D0
13	END		

[在 X20 ～ X2F 中指定了 135 时的动作]



7.12.6 浮点数的 TAN 运算（双精度）(TAND(P))

Basic　High performance　Process　Redundant　Universal



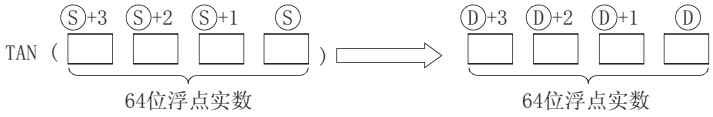
(S)：进行 TAN（正切）运算的角度数据或者存储角度数据的软元件的起始编号（实数）。

(D)：存储运算结果的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
(S)	--	○		--				○	--
(D)	--	○		--				--	--

功　能

(1) 对(S)中指定的角度值进行 TAN（正切）运算后，将运算结果存储到(D)中指定的软元件中。



(2) (S)中指定的角度是以弧度单位（角度×π÷180）设置的。
关于角度↔弧度的转换，请参阅 RADD 指令、DEGD 指令。

(3) (S)中指定的角度为π/2 弧度、(3/2)π弧度时，将会产生弧度值运算误差，且不会变为出错状态，应加以注意。

(4) 运算结果为 -0 或者发生了下溢时，将运算结果作为 0 进行处理。

出　错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 指定的软元件的内容不在以下范围内时。（出错代码：4140）
$0, 2^{-1022} \leqq |\text{指定软元件的内容}| < 2^{1024}$

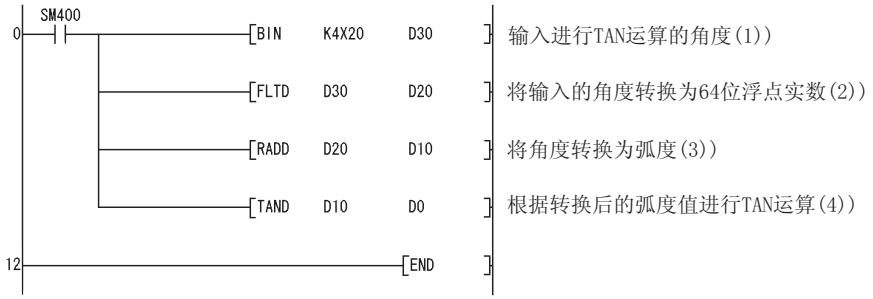
· 指定软元件的内容为 -0 时。（出错代码：4140）

· 运算结果超出了下述范围时。（发生了溢出时。）（仅通用型 QCPU）
$2^{1024} \leqq |\text{运算结果}|$　（出错代码：4141）

程序示例

(1) 以下为对 X20 ～ X2F 中以 BCD4 位数设置的角度进行 TAN 运算后，以 64 位浮点实数格式存储到 D0 ～ D3 中的程序。

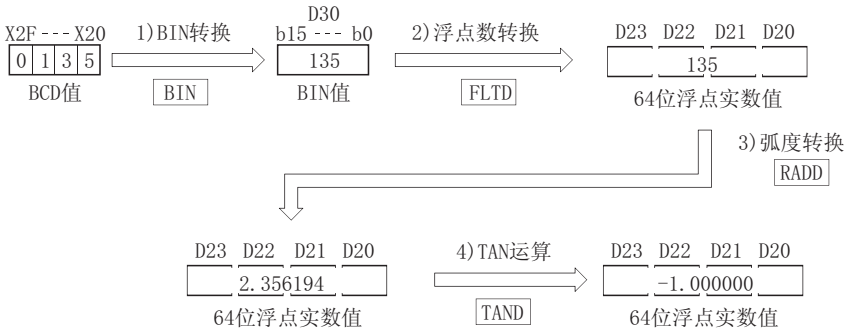
[梯形图模式]



[列表模式]

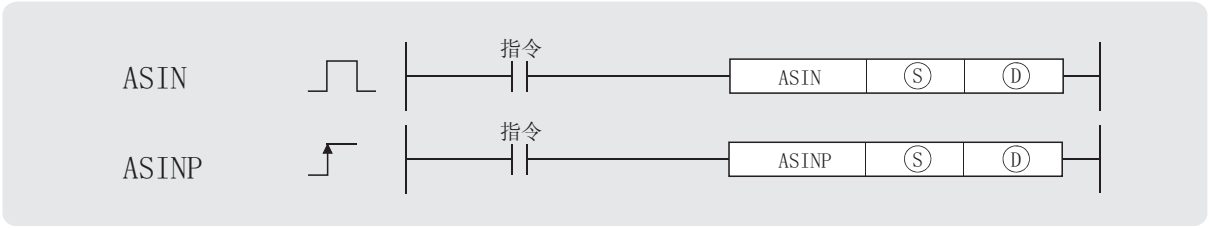
步	指令	软元件	
0	LD	SM400	
1	BIN	K4X20	D30
3	FLTD	D30	D20
6	RADD	D20	D10
9	TAND	D10	D0
12	END		

[在 X20 ～ X2F 中指定了 135 时的动作]



7.12.7 浮点数的 SIN^{-1} 运算（单精度）(ASIN(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ：进行 SIN^{-1}（反正弦）运算的 SIN 值或者存储 SIN 值的软元件的起始编号（实数）。

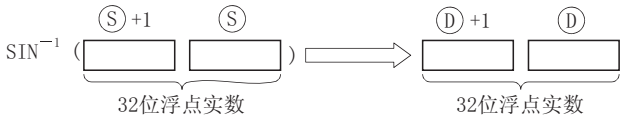
Ⓓ：存储运算结果的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
Ⓢ	--	○		--	○		○*1	○	--
Ⓓ	--	○		--	○		○*1	--	--

*1：仅通用型 QCPU 可以使用。

功 能

(1) 通过Ⓢ中指定的 SIN 值进行角度运算后，将运算结果存储到Ⓓ中指定的字软元件中。



(2) Ⓢ中指定的 SIN 值的可设置范围为 -1.0 ～ 1.0。

(3) Ⓓ中存储的角度（运算结果）是以弧度单位存储的。
关于角度↔弧度的转换，请参阅 RAD 指令、DEG 指令。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· ⓢ中指定的值超过了 -1.0 ～ 1.0 的范围时。 (出错代码： 4100)

· 指定的软元件的内容不在以下范围内时。(仅通用型 QCPU)

0，2^{-126} ≦｜指定软元件的内容｜＜ 2^{128} (出错代码： 4140)

· 指定软元件的内容为 -0 时。*2

(基本型 QCPU、高性能型 QCPU、过程 CPU、冗余 CPU 时) (出错代码： 4100)

*2: 有的 CPU 模块即使指定为 -0 也不会变为运算出错状态。
有关详细内容请参阅 3.2.4 项。

· 运算结果超出了下述范围时。(发生了溢出时。)(仅通用型 QCPU)

2^{128} ≦｜运算结果｜ (出错代码： 4141)

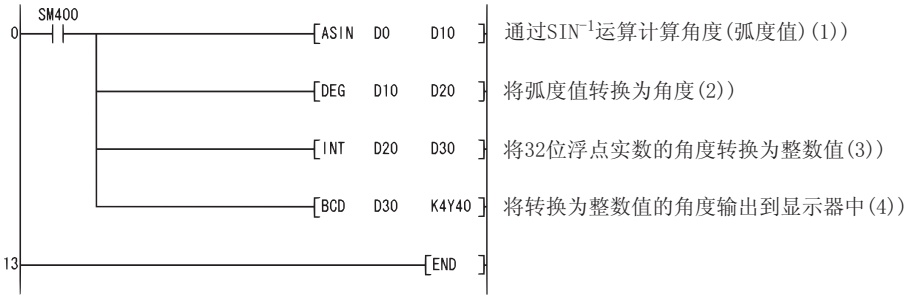
· 指定软元件的内容为 -0、非正规数、非数、±∞时。(仅通用型 QCPU)

(出错代码： 4140)

程序示例

(1) 以下为求出 D0、D1 的 32 位浮点实数的 SIN^{-1} 后，将该角度以 BCD4 位数格式输出到 Y40 ～ Y4F 中的程序。

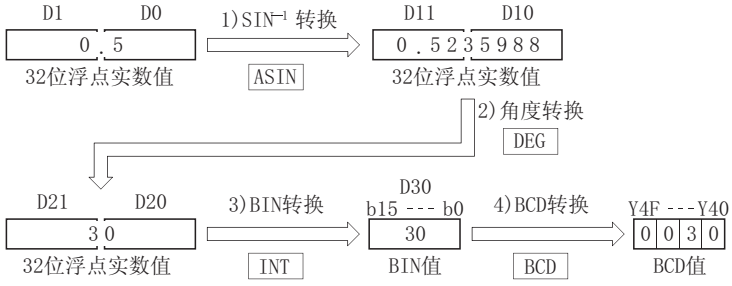
[梯形图模式]



[列表模式]

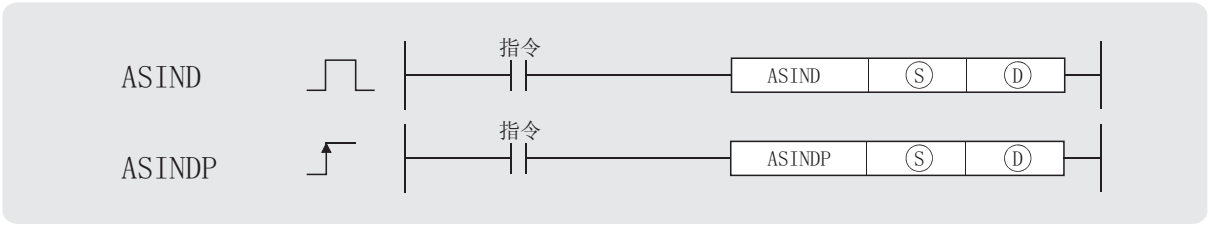
步	指令	软元件	
0	LD	SM400	
1	ASIN	D0	D10
4	DEG	D10	D20
7	INT	D20	D30
10	BCD	D30	K4Y40
13	END		

[D0、D1 的值为 0.5 时的动作]



7.12.8 浮点数的 SIN^{-1} 运算（双精度）(ASIND(P))

Basic　High performance　Process　Redundant　Universal



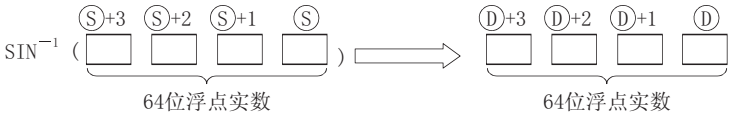
Ⓢ ：进行 SIN^{-1}（反正弦）运算的 SIN 值或者存储 SIN 值的软元件的起始编号（实数）。

Ⓓ ：存储运算结果的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
Ⓢ	--	○		--				○	--
Ⓓ	--	○		--				--	--

功　能

(1) 通过Ⓢ中指定的 SIN 值进行角度运算后，将运算结果存储到Ⓓ中指定的字软元件中。



(2) Ⓢ中指定的 SIN 值的可设置范围为 -1.0 ～ 1.0。

(3) Ⓓ中存储的角度（运算结果）是以弧度单位存储的。
关于角度↔弧度的转换，请参阅 RADD 指令、DEGD 指令。

(4) 运算结果为 -0 或者发生了下溢时，将运算结果作为 0 进行处理。

出　错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 指定的软元件的内容不在以下范围内时。　（出错代码：4140）
$0, 2^{-1022} \leqq |$指定软元件的内容$| < 2^{1024}$

· 指定软元件的内容为 -0 时。　（出错代码：4140）

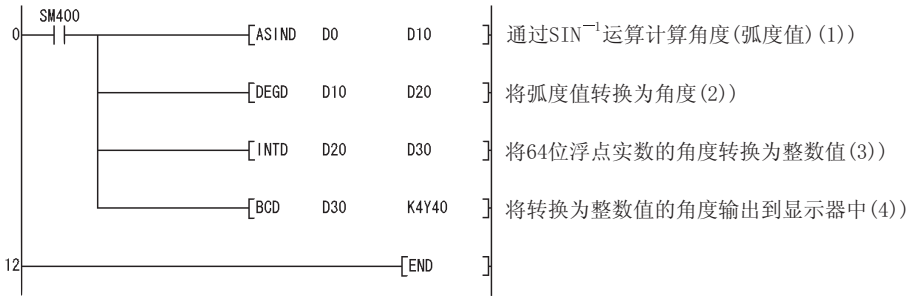
· Ⓢ中指定的值在双精度浮点数范围内超过了 -1.0 ～ 1.0 的范围时。　（出错代码：4100）

· 运算结果超出了下述范围时。（发生了溢出时。）
$2^{1024} \leqq |$运算结果$|$　（出错代码：4141）

程序示例

(1) 以下为求出 D0 ～ D3 的 64 位浮点实数的 SIN^{-1} 后，将该角度以 BCD4 位数格式输出到 Y40 ～ Y4F 中的程序。

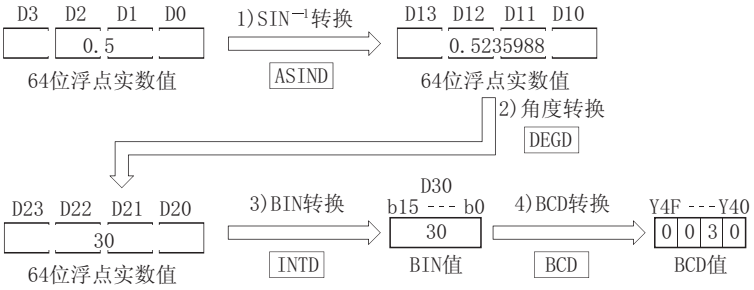
[梯形图模式]



[列表模式]

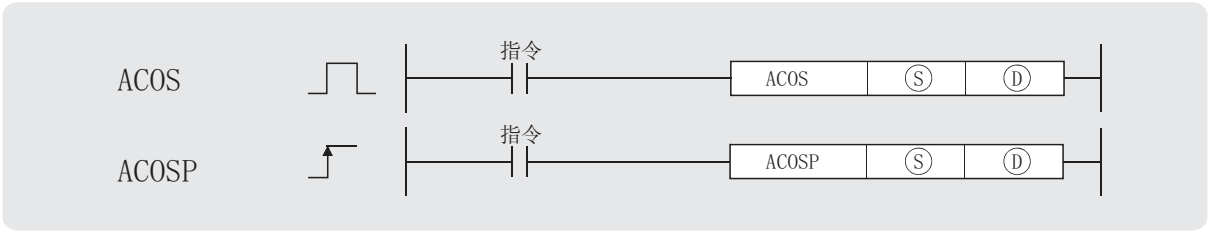
步	指令	软元件	
0	LD	SM400	
1	ASIND	D0	D10
4	DEGD	D10	D20
7	INTD	D20	D30
10	BCD	D30	K4Y40
12	END		

[D0 ～ D3 的值为 0.5 时的动作]



7.12.9 浮点数的 COS $^{-1}$ 运算（单精度）(ACOS(P))

Basic | High performance | Process | Redundant | Universal



(S) ：进行 COS^{-1}（反正弦）运算的 COS 值或者存储 COS 值的软元件的起始编号（实数）。

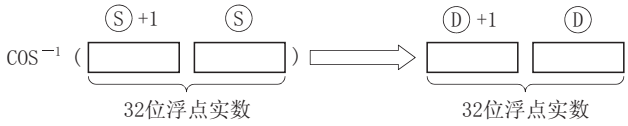
(D) ：存储运算结果的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
(S)	--	○		--	○		○*1	○	--
(D)	--	○		--	○		○*1	--	--

*1：仅通用型 QCPU 可以使用。

功 能

(1) 通过(S)中指定的 COS 值进行角度运算后，将运算结果存储到(D)中指定的字软元件中。



(2) (S)中指定的 COS 值的可设置范围为 -1.0 ～ 1.0。

(3) (D)中存储的角度（运算结果）是以弧度单位存储的。
关于角度↔弧度的转换，请参阅 RAD 指令、DEG 指令。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· ⓢ中指定的值超过了 -1.0 ～ 1.0 的范围时。 (出错代码： 4100)

· 指定的软元件的内容不在以下范围内时。(仅通用型 QCPU)

$0, 2^{-126} \leqq |$指定软元件的内容$| < 2^{128}$ (出错代码： 4140)

· 指定软元件的内容为 -0 时。*2

(基本型 QCPU、高性能型 QCPU、过程 CPU、冗余 CPU 时) (出错代码： 4100)

*2:有的 CPU 模块即使指定为 -0 也不会变为运算出错状态。
有关详细内容请参阅 3.2.4 项。

· 运算结果超出了下述范围时。(发生了溢出时。)(仅通用型 QCPU)

$2^{128} \leqq |$运算结果$|$ (出错代码： 4141)

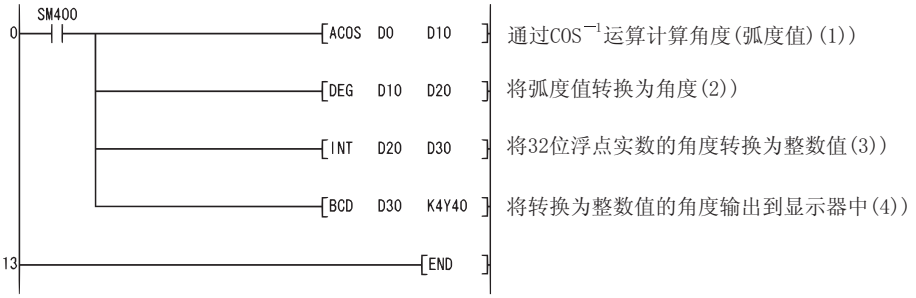
· 指定软元件的内容为 -0、非正规数、非数、±∞时。(仅通用型 QCPU)

(出错代码： 4140)

程序示例

(1) 以下为求出 D0、D1 的 32 位浮点实数的 COS^{-1} 后，将该角度以 BCD4 位数格式输出到 Y40 ～ Y4F 中的程序。

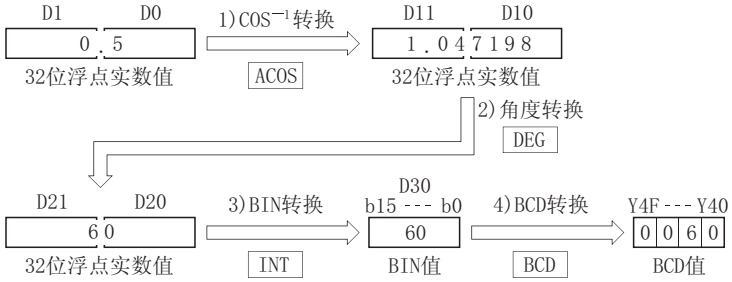
[梯形图模式]



[列表模式]

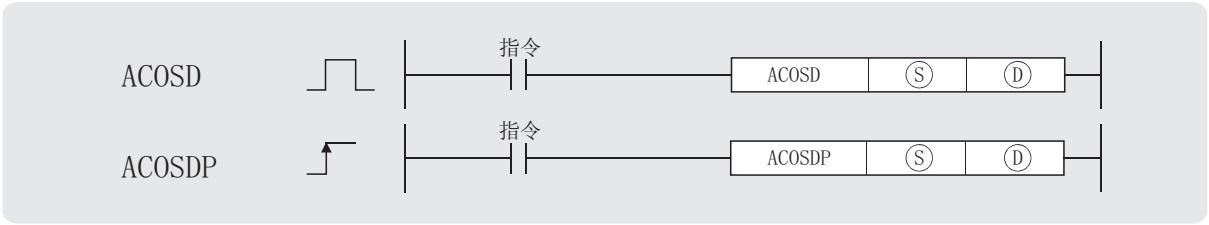
步	指令	软元件	
0	LD	SM400	
1	ACOS	D0	D10
4	DEG	D10	D20
7	INT	D20	D30
10	BCD	D30	K4Y40
13	END		

[D0、D1 的值为 0.5 时的动作]



7.12.10 浮点数的 COS $^{-1}$ 运算（双精度）(ACOSD(P))

Basic　High performance　Process　Redundant　Universal



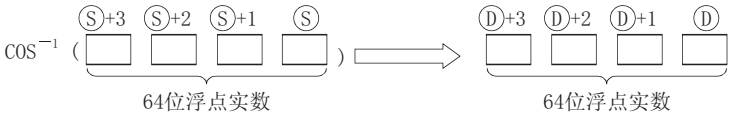
Ⓢ：进行 COS^{-1}（反余弦）运算的 COS 值或者存储 COS 值的软元件的起始编号（实数）。

Ⓓ：存储运算结果的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
Ⓢ	--	○		--				○	--
Ⓓ	--	○		--				--	--

功　能

(1) 通过Ⓢ中指定的 COS 值进行角度运算后，将运算结果存储到Ⓓ中指定的字软元件中。



(2) Ⓢ中指定的 COS 值的可设置范围为 -1.0 ～ 1.0。

(3) Ⓓ中存储的角度（运算结果）是以弧度单位存储的。
关于角度↔弧度的转换，请参阅 RADD 指令、DEGD 指令。

(4) 运算结果为 -0 或者发生了下溢时，将运算结果作为 0 进行处理。

出　错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 指定的软元件的内容不在以下范围内时。　（出错代码：4140）
$0, 2^{-1022} \leqq |$指定软元件的内容$| < 2^{1024}$

· 指定软元件的内容为 -0 时。　（出错代码：4140）

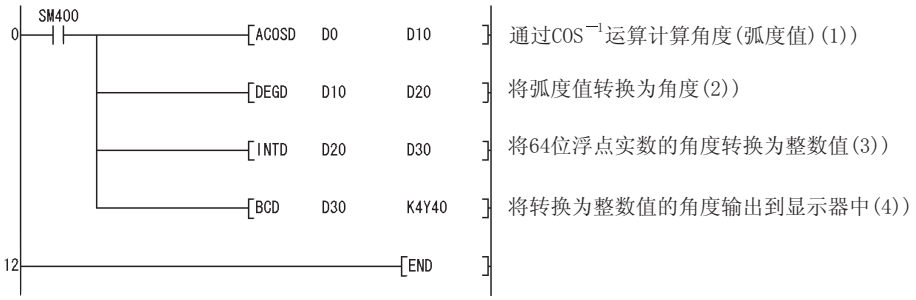
· Ⓢ中指定的值在双精度浮点数范围内超过了 -1.0 ～ 1.0 的范围时。　（出错代码：4100）

· 运算结果超出了下述范围时。（发生了溢出时。）
$2^{1024} \leqq |$运算结果$|$　（出错代码：4141）

程序示例

(1) 以下为求出 D0 ～ D3 的 64 位浮点实数的 COS^{-1} 后，将该角度以 BCD4 位数格式输出到 Y40 ～ Y4F 中的程序。

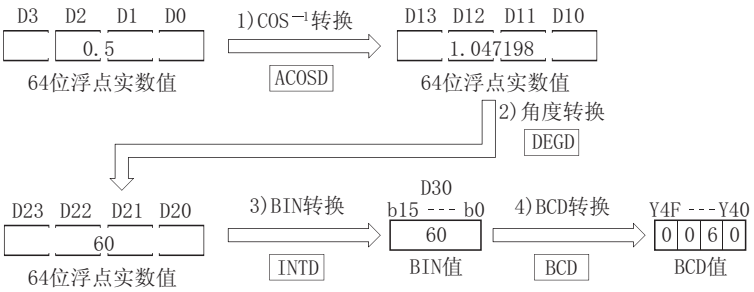
[梯形图模式]



[列表模式]

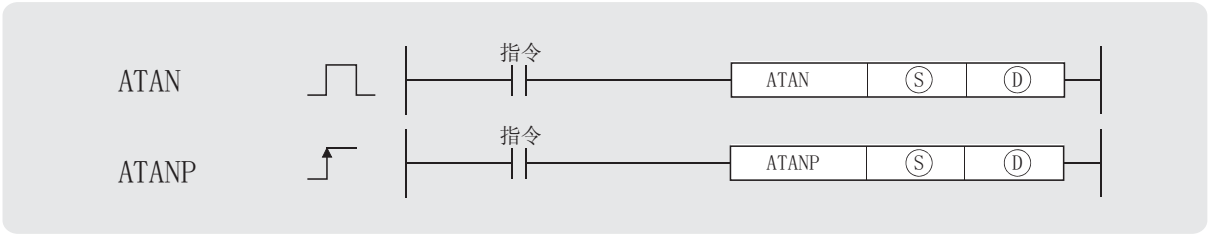
步	指令	软元件	
0	LD	SM400	
1	ACOSD	D0	D10
4	DEGD	D10	D20
7	INTD	D20	D30
10	BCD	D30	K4Y40
12	END		

[D0 ～ D3 的值为 0.5 时的动作]



7.12.11 浮点数的 TAN $^{-1}$ 运算（单精度）(ATAN(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ：进行 TAN^{-1}（反正切）运算的 TAN 值或者存储 TAN 值的软元件的起始编号（实数）。

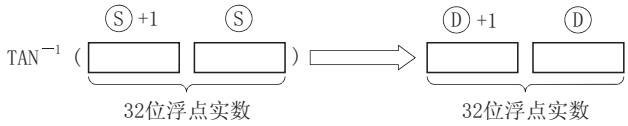
Ⓓ：存储运算结果的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
Ⓢ	--	○		--	○		○*1	○	--
Ⓓ	--	○		--	○		○*1	--	--

*1：仅通用型 QCPU 可以使用。

功 能

(1) 通过Ⓢ中指定的 TAN 值进行角度运算后，将运算结果存储到Ⓓ中指定的字软元件中。



(2) Ⓓ中存储的角度（运算结果）是以弧度单位存储的。

关于角度↔弧度的转换，请参阅 RAD 指令、DEG 指令。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 指定的软元件的内容不在以下范围内时。（仅通用型 QCPU）

$0, 2^{-126} \leqq |$指定软元件的内容$| < 2^{128}$ （出错代码：4140）

· 指定软元件的内容为 -0 时。*2

（基本型 QCPU、高性能型 QCPU、过程 CPU、冗余 CPU 时） （出错代码：4100）

*2:有的 CPU 模块即使指定为 -0 也不会变为运算出错状态。
有关详细内容请参阅 3.2.4 项。

· 运算结果超出了下述范围时。（发生了溢出时。）（仅通用型 QCPU）

$2^{128} \leqq |$运算结果$|$ （出错代码：4141）

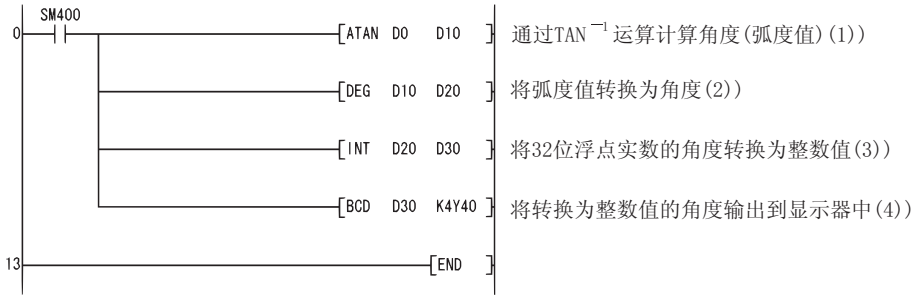
· 指定软元件的内容为 -0、非正规数、非数、±∞时。（仅通用型 QCPU）

（出错代码：4140）

程序示例

(1) 以下为求出 D0、D1 的 32 位浮点实数的 TAN^{-1} 后，将该角度以 BCD4 位数格式输出到 Y40 ～ Y4F 中的程序。

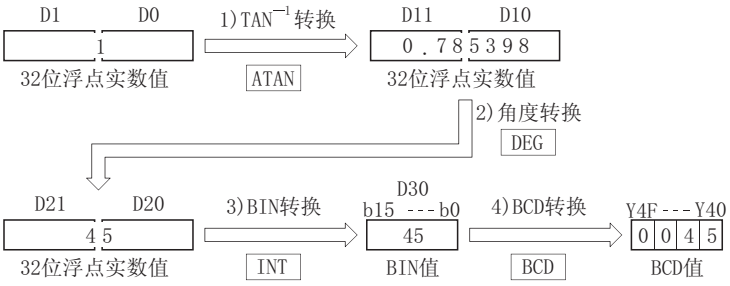
[梯形图模式]



[列表模式]

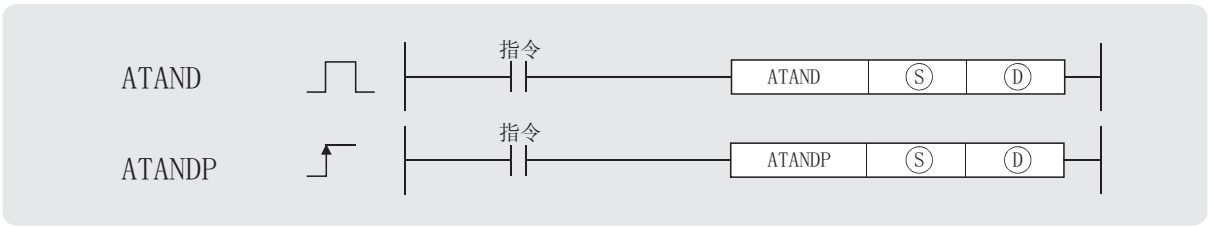
步	指令	软元件	
0	LD	SM400	
1	ATAN	D0	D10
4	DEG	D10	D20
7	INT	D20	D30
10	BCD	D30	K4Y40
13	END		

[D0、D1 的值为 1 时的动作]



7.12.12 浮点数的 TAN $^{-1}$ 运算（双精度）(ATAND(P))

Basic　High performance　Process　Redundant　Universal



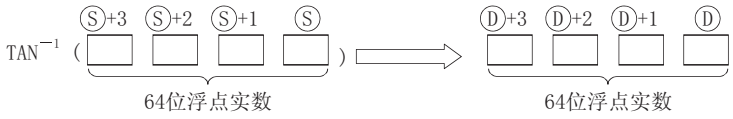
Ⓢ ：进行 TAN^{-1}（反正切）运算的 TAN 值或者存储 TAN 值的软元件的起始编号（实数）。

Ⓓ ：存储运算结果的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
Ⓢ	--	○		--				○	--
Ⓓ	--	○		--				--	--

功　能

(1) 通过Ⓢ中指定的 TAN 值进行角度运算后，将运算结果存储到Ⓓ中指定的字软元件中。



(2) Ⓓ中存储的角度（运算结果）是以弧度单位存储的。
关于角度↔弧度的转换，请参阅 RADD 指令、DEGD 指令。

(3) 运算结果为 -0 或者发生了下溢时，将运算结果作为 0 进行处理。

出　错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 指定的软元件的内容不在以下范围内时。（出错代码：4140）
$0, 2^{-1022} \leqq |$指定软元件的内容$| < 2^{1024}$

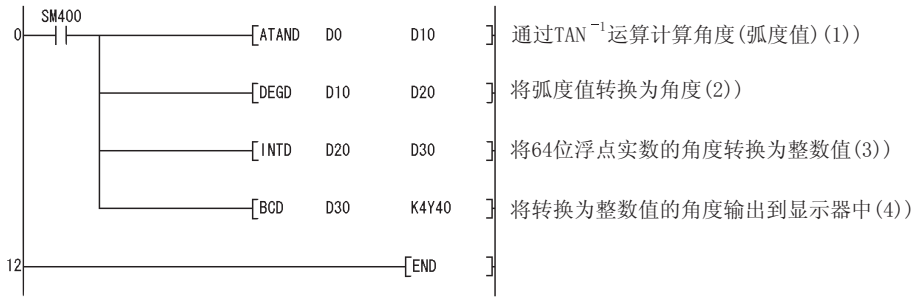
· 指定软元件的内容为 -0 时。（出错代码：4140）

· 运算结果超出了下述范围时。（发生了溢出时。）
$2^{1024} \leqq |$运算结果$|$（出错代码：4141）

程序示例

(1) 以下为求出 D0 ～ D3 的 64 位浮点实数的 TAN^{-1} 后，将该角度以 BCD4 位数格式输出到 Y40 ～ Y4F 中的程序。

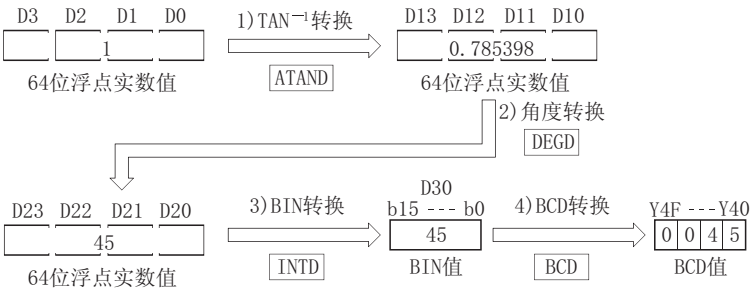
[梯形图模式]



[列表模式]

步	指令	软元件	
0	LD	SM400	
1	ATAND	D0	D10
4	DEGD	D10	D20
7	INTD	D20	D30
10	BCD	D30	K4Y40
12	END		

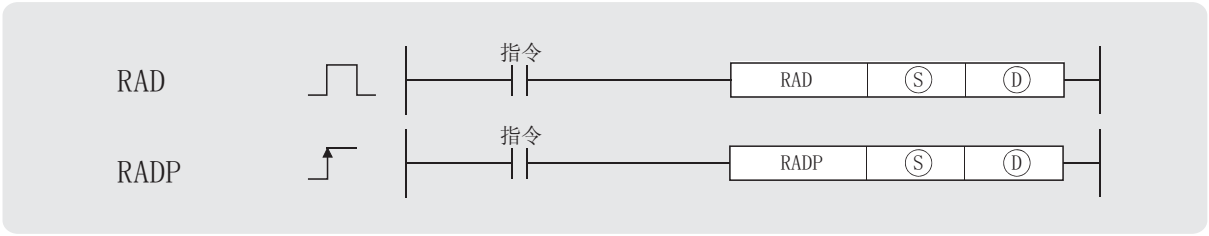
[D0 ～ D3 的值为 1 时的动作]



7.12.13 浮点数角度→弧度的转换（单精度）(RAD(P))

Ver. Basic | High performance | Process | Redundant | Universal

基本型 QCPU：序列号的前 5 位数为“ 04122 ”以后



Ⓢ：要转换为弧度单位的角度或者存储角度的软元件的起始编号（实数）。

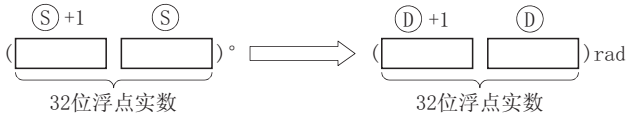
Ⓓ：存储转换为弧度单位的值的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
Ⓢ	--	○		--	○		○*1	○	--
Ⓓ	--	○		--	○		○*1	--	--

*1：仅通用型 QCPU 可以使用。

功 能

(1) 将角度的大小单位从Ⓢ中指定的度单位转换为弧度单位后，将运算结果存储到Ⓓ中指定编号的软元件中。



(2) 度单位→弧度单位的转换公式如下所示。

$$\text{弧度单位} = \text{度单位} \times \frac{\pi}{180}$$

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 指定的软元件的内容不在以下范围内时。(仅通用型 QCPU)

$0, 2^{-126} \leqq |$指定软元件的内容$| < 2^{128}$ (出错代码： 4140)

· 指定软元件的内容为 -0 时。*2

(基本型 QCPU、高性能型 QCPU、过程 CPU、冗余 CPU 时) (出错代码： 4100)

*2:有的 CPU 模块即使指定为 -0 也不会变为运算出错状态。
有关详细内容请参阅 3.2.4 项。

· 运算结果超出了下述范围时。(发生了溢出时。)(仅通用型 QCPU)

$2^{128} \leqq |$运算结果$|$ (出错代码： 4141)

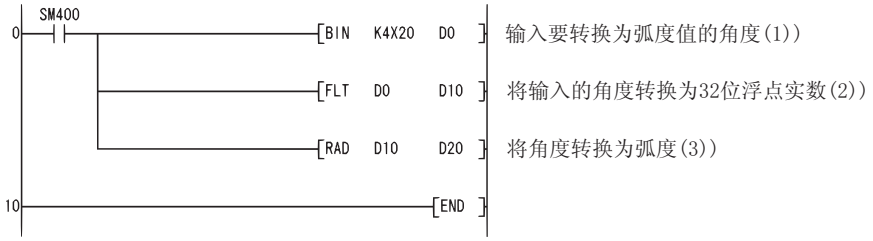
· 指定软元件的内容为 -0、非正规数、非数、±∞时。(仅通用型 QCPU)

(出错代码： 4140)

程序示例

(1) 以下为将 X20 ～ X2F 中以 BCD4 位数设置的角度转换为弧度后，以 32 位浮点实数存储到 D20、D21 中的程序。

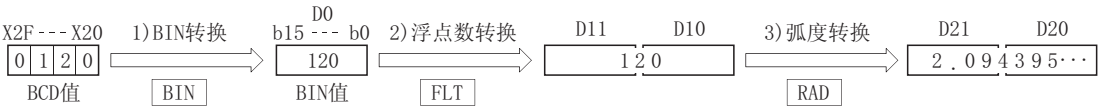
[梯形图模式]



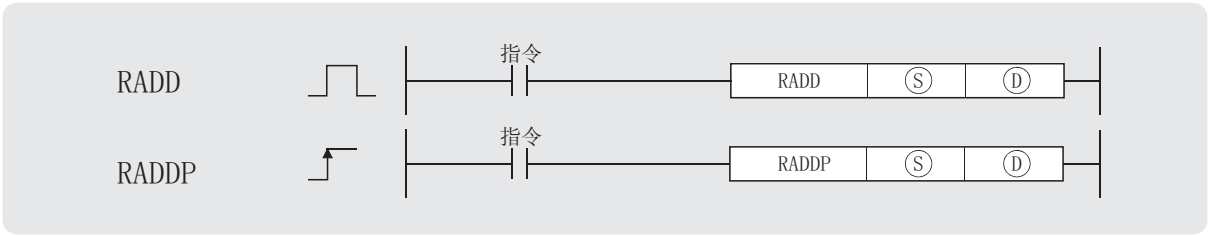
[列表模式]

步	指令	软元件	
0	LD	SM400	
1	BIN	K4X20	D0
4	FLT	D0	D10
7	RAD	D10	D20
10	END		

[在 X20 ～ X2F 中指定了 120 时的动作]



7.12.14 浮点数角度→弧度的转换（双精度）(RADD(P))



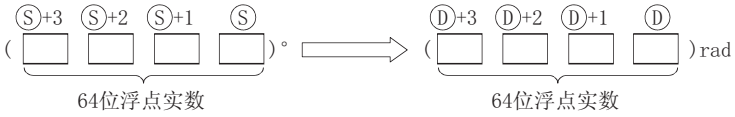
Ⓢ：要转换为弧度单位的角度或者存储角度的软元件的起始编号（实数）。

Ⓓ：存储转换为弧度单位的值的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
Ⓢ	--	○		--				○	--
Ⓓ	--	○		--				--	--

功 能

(1) 将角度的大小单位从Ⓢ中指定的度单位转换为弧度单位后，将运算结果存储到Ⓓ中指定编号的软元件中。



(2) 度单位→弧度单位的转换公式如下所示。

$$\text{弧度单位} = \text{度单位} \times \frac{\pi}{180}$$

(3) 运算结果为 -0 或者发生了下溢时，将运算结果作为 0 进行处理。

出 错

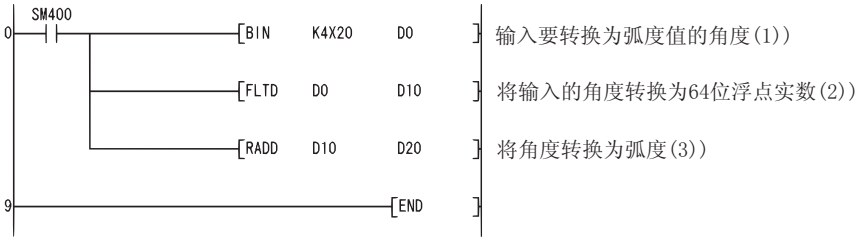
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- 指定的软元件的内容不在以下范围内时。（出错代码：4140）
  $0, 2^{-1022} \leqq |\text{指定软元件的内容}| < 2^{1024}$
- 指定软元件的内容为 -0 时。（出错代码：4140）
- 运算结果超出了下述范围时。（发生了溢出时。）
  $2^{1024} \leqq |\text{运算结果}|$（出错代码：4141）

程序示例

(1) 以下为将 X20 ～ X2F 中以 BCD4 位数设置的角度转换为弧度后，以 64 位浮点实数存储到 D20 ～ D23 中的程序。

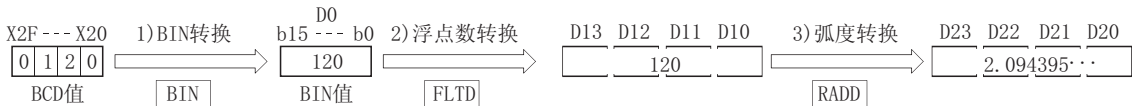
[梯形图模式]



[列表模式]

步	指令	软元件	
0	LD	SM400	
1	BIN	K4X20	D0
3	FLTD	D0	D10
6	RADD	D10	D20
9	END		

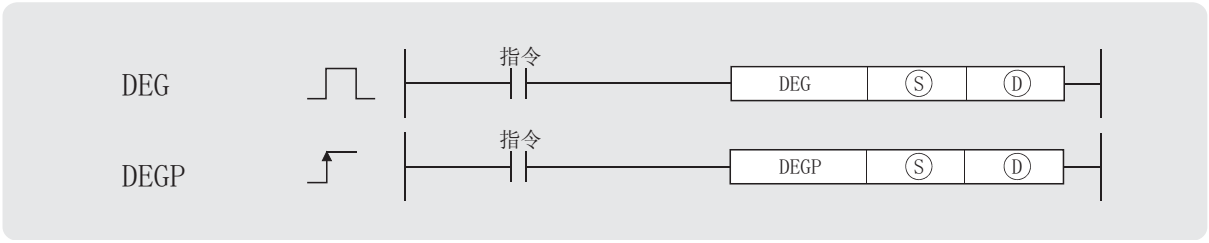
[在 X20 ～ X2F 中指定了 120 时的动作]



7.12.15 浮点数弧度→角度的转换（单精度）(DEG(P))

Ver. Basic | High performance | Process | Redundant | Universal

基本型 QCPU：序列号的前 5 位数为“ 04122 ”以后



Ⓢ：要转换为度单位的弧度角度或者存储弧度角度的软元件的起始编号（实数）。

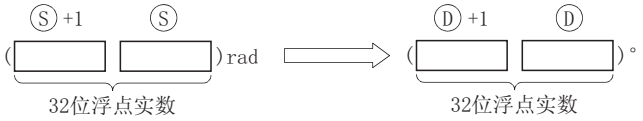
Ⓓ：存储转换为度单位的值的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
Ⓢ	--	○		--	○		○*1	○	--
Ⓓ	--	○		--	○		○*1	--	--

*1：仅通用型 QCPU 可以使用。

功 能

(1) 将角度的大小单位从Ⓢ中指定的弧度单位转换为度单位后，将运算结果存储到Ⓓ中指定编号的软元件中。



(2) 弧度单位→度单位的转换公式如下所示。

$$\text{度单位} = \text{弧度单位} \times \frac{180}{\pi}$$

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 指定软元件的内容为 -0 时。*2
(对于基本型 QCPU、高性能型 QCPU、过程 CPU、冗余 CPU) (出错代码： 4100)

*2: 有的 CPU 模块即使指定了 -0 也不会变为运算出错状态。
有关详细内容请参阅 3.2.4 项。

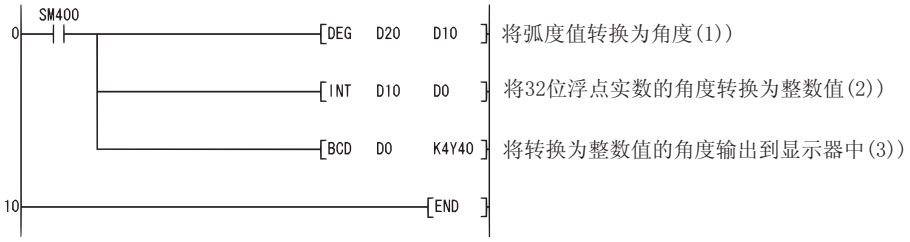
· 运算结果超出以下范围时。(发生了溢出时。) (仅通用型 QCPU)
$2^{128} \leqq$ | 运算结果 | (出错代码： 4141)

· 指定软元件的内容为 -0、非正规数、非数、±∞时。(仅通用型 QCPU)
(出错代码： 4140)

程序示例

(1) 以下为将 D20、D21 中以 32 位浮点实数设置的弧度值转换为角度后，以 BCD 值格式输出到 Y40 ～ Y4F 中的程序。

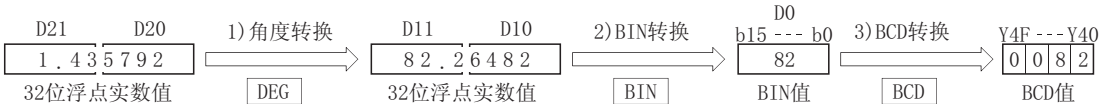
[梯形图模式]



[列表模式]

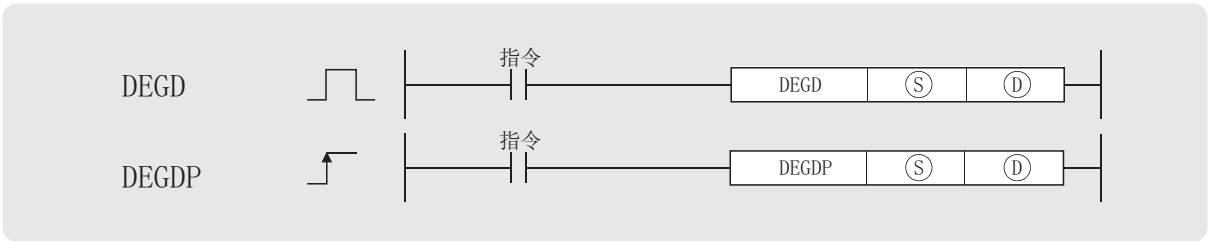
步	指令	软元件	
0	LD	SM400	
1	DEG	D20	D10
4	INT	D10	D0
7	BCD	D0	K4Y40
10	END		

[D20、D21 的值为 1.435792 时的动作]



7.12.16 浮点数弧度→角度的转换（双精度）(DEGD(P))

Basic　High performance　Process　Redundant　Universal



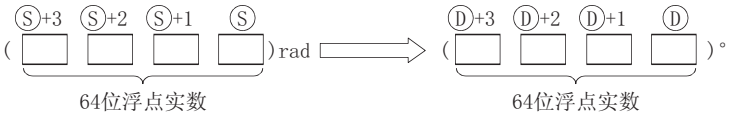
Ⓢ：要转换为度单位的弧度角度或者存储弧度角度的软元件的起始编号（实数）。

Ⓓ：存储转换为度单位的值的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
Ⓢ	--	○		--				○	--
Ⓓ	--	○		--				--	--

功　能

(1) 将角度的大小单位从Ⓢ中指定的弧度单位转换为度单位后，将运算结果存储到Ⓓ中指定编号的软元件中。



(2) 弧度单位→度单位的转换公式如下所示。

$$\text{度单位} = \text{弧度单位} \times \frac{180}{\pi}$$

(3) 运算结果为 -0 或者发生了下溢时，将运算结果作为 0 处理。

出　错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 指定的软元件的内容不在以下范围内时。　（出错代码：4140）

$0, 2^{-1022} \leqq |\text{指定软元件的内容}| < 2^{1024}$

· 指定软元件的内容为 -0 时。　（出错代码：4140）

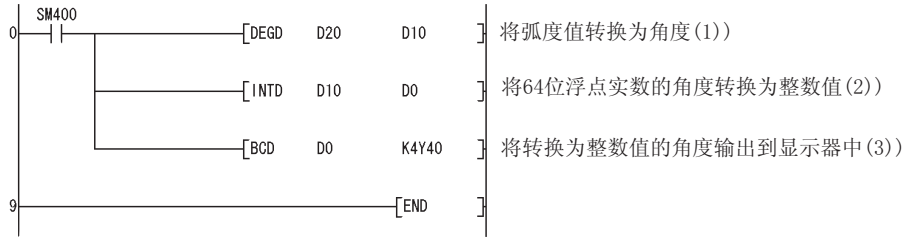
· 运算结果超出了下述范围时。（发生了溢出时。）

$2^{1024} \leqq |\text{运算结果}|$　（出错代码：4141）

程序示例

(1) 以下为将 D20、D21 中以 64 位浮点实数设置的弧度值转换为角度后，以 BCD 值格式输出到 Y40 ～ Y4F 中的程序。

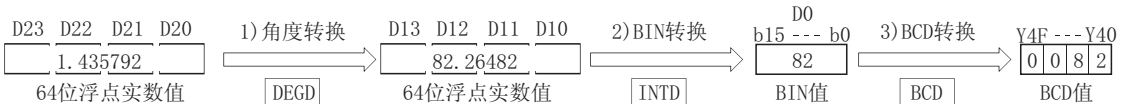
[梯形图模式]



[列表模式]

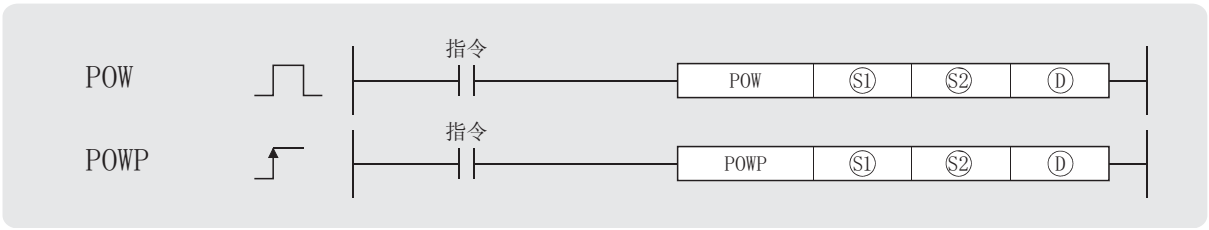
步	指令	软元件	
0	LD	SM400	
1	DEGD	D20	D10
4	INTD	D10	D0
7	BCD	D0	K4Y40
9	END		

[D20 ～ D23 的值为 1.435792 时的动作]



7.12.17 浮点数的幂运算（单精度）(POW(P))

· QnU(D)(H)CPU：序列号的前 5 位数为“ 10102 ”以后
· QnUDE(H)CPU：序列号的前 5 位数为“ 10102 ”以后



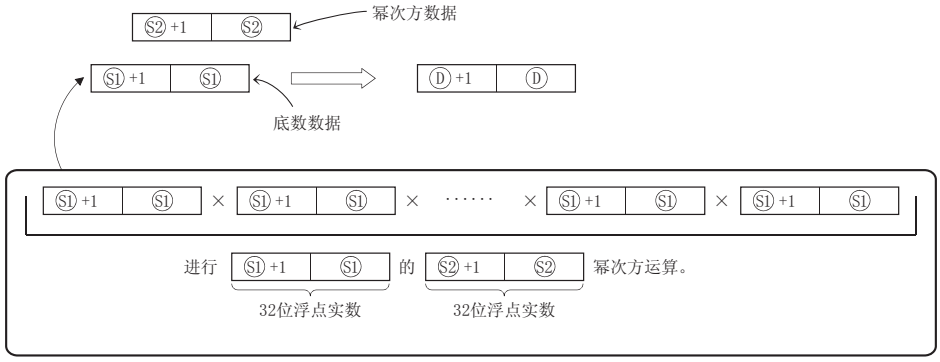
(S1) ：底数数据或者存储底数数据的软元件的起始编号（实数）。
(S2) ：幂次方数据或者存储幂次方数据的软元件的起始编号（实数）。
(D) ：存储运算结果的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
(S1)	--	○		--	○		○	△ *1	--
(S2)	--	○		--	○		○	△ *1	--
(D)	--	○		--	○		○	--	--

*1：仅实数可以使用。

功 能

(1) 以(S1)中指定的 32 位浮点实数作为底数，以(S2)中指定的 32 位浮点实数作为幂次方进行幂运算，并将运算结果存储到(D)中指定的位软元件中。



(2) (S1)、(S2)中可指定的值及可存储的值如下所示。

$0，2^{-126} \leqq |$ 设置值（存储值）$| < 2^{128}$

(3) 运算结果为 -0 或者发生了下溢时，将运算结果作为 0 进行处理。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· ⓢ1 或者ⓢ2 中指定的值不在下述范围内时。 (出错代码： 4140)

$0，2^{-126} \leqq |设置值（存储值）| < 2^{128}$

· ⓢ1 或者ⓢ2 的内容为 -0 时。 (出错代码： 4140)

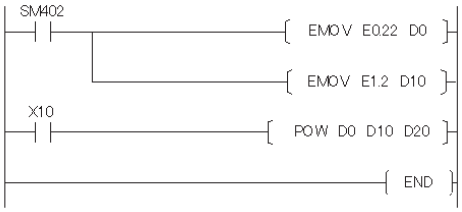
· 运算结果超出了下述范围时。(发生了溢出时。)

$2^{128} \leqq |运算结果|$ (出错代码： 4141)

程序示例

(1) 以下为 X10 变为 ON 时，将 D0、D1 的 32 位浮点实数作为底数，以 D10、D11 的 32 位浮点实数作为幂进行幂运算，并将运算结果存储到 D20、D21 中的程序。

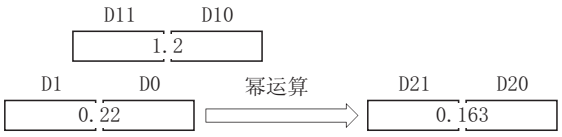
[梯形图模式]



[列表模式]

步	指令	软元件
0	LD	SM402
1	EMOV	E022 D0
4	EMOV	E1.2 D10
7	LD	X10
8	POW	D0 D10 D20
12	END	

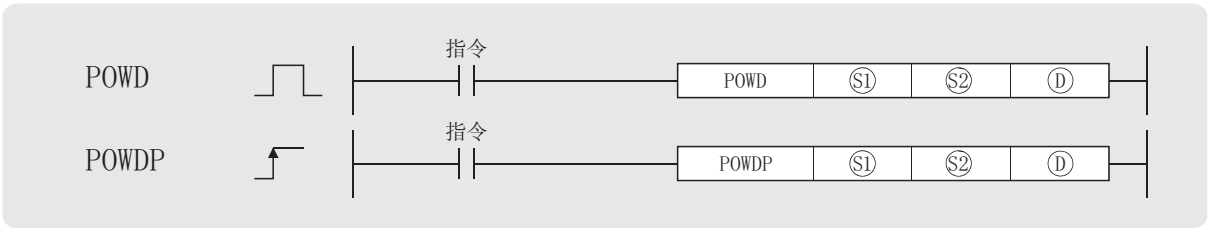
[动作]



7.12.18 浮点数的幂运算（双精度）(POWD(P))

Basic × High performance × Process × Redundant × Universal Ver.

· QnU(D)(H)CPU：序列号的前 5 位数为“ 10102 ”以后
· QnUDE(H)CPU：序列号的前 5 位数为“ 10102 ”以后



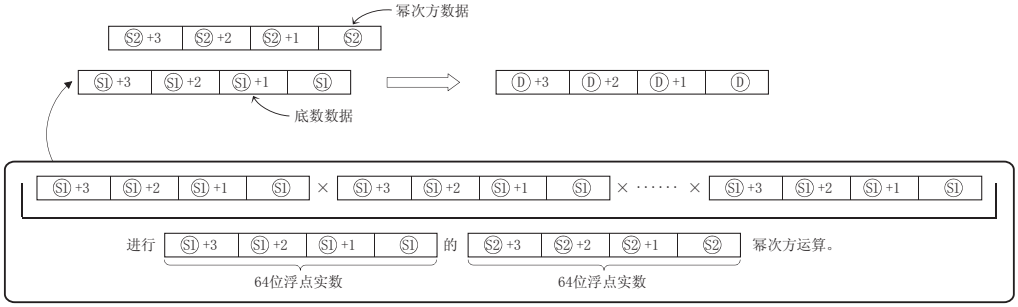
(S1) ：底数数据或者存储底数数据的软元件的起始编号（实数）。
(S2) ：幂次方数据或者存储幂次方数据的软元件的起始编号（实数）。
(D) ：存储运算结果的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
(S1)	--	○		--	○		--	△ *1	--
(S2)	--	○		--	○		--	△ *1	--
(D)	--	○		--	○		--	--	--

*1：仅实数可以使用。

功 能

(1) 以(S1)中指定的 64 位浮点实数作为底数，以(S2)中指定的 64 位浮点实数作为幂次方进行幂运算，并将运算结果存储到(D)中指定的位软元件中。



(2) (S1)、(S2)中可指定的值及可存储的值如下所示。

$0，2^{-1022} \leqq |$ 设置值（存储值）$| < 2^{1024}$

(3) 运算结果为 -0 或者发生了下溢时，将运算结果作为 0 进行处理。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· ⓢ1 或者ⓢ2 中指定的值不在下述范围内时。 (出错代码： 4140)

$0，2^{-1022} \leqq |$ 设置值 (存储值) $| < 2^{1024}$

· ⓢ1 或者ⓢ2 的内容为 -0 时。 (出错代码： 4140)

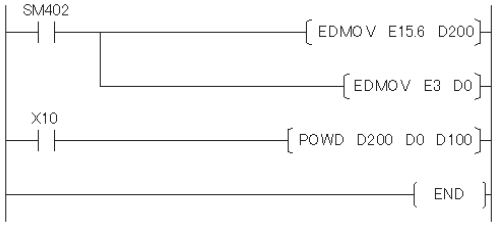
· 运算结果超出了下述范围时。(发生了溢出时。)

$2^{1024} \leqq |$ 运算结果 $|$ (出错代码： 4141)

程序示例

(1) 以下为 X10 变为 ON 时，将 D200 ～ D203 的 64 位浮点实数作为底数，以 D0 ～ D3 的 64 位浮点实数作为幂进行幂运算，并将运算结果存储到 D100 ～ D103 中的程序。

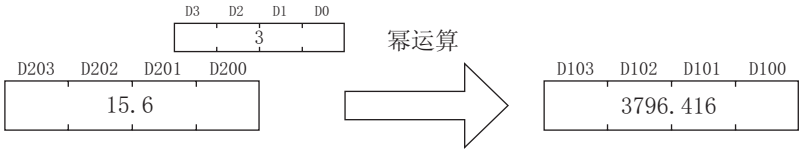
[梯形图模式]



[列表模式]

步	指令	软元件
0	LD	SM402
1	EDMOV	E15.6 D200
4	EDMOV	E3 D0
7	LD	X10
8	POWD	D200 D0 D100
12	END	

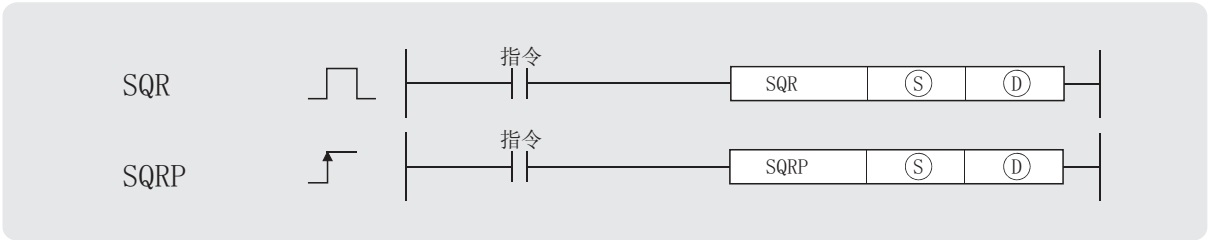
[动作]



7.12.19 浮点数的平方根运算（单精度）(SQR(P))

Ver. Basic | High performance | Process | Redundant | Universal

基本型 QCPU：序列号的前 5 位数为“ 04122 ”以后



Ⓢ ：进行平方根运算的数据或者存储数据的软元件的起始编号（实数）。

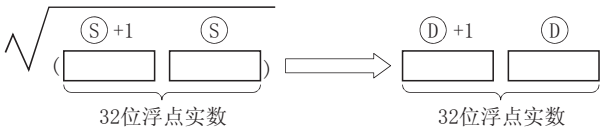
Ⓓ ：存储运算结果的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
Ⓢ	--	○		--	○		○*1	○	--
Ⓓ	--	○		--	○		○*1	--	--

*1：仅通用型 QCPU 可以使用。

功 能

(1) 对Ⓢ中指定的值进行平方根运算后，将运算结果存储到Ⓓ中指定编号的软元件中。



(2) Ⓢ中指定的值只能设置为正数。（为负数时将无法运算。）

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

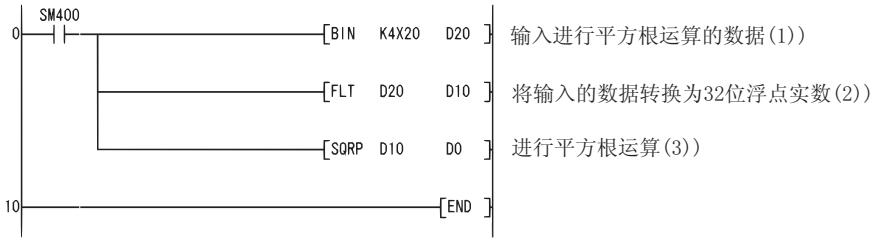
- Ⓢ中指定的值为负数时。（出错代码：4100）
- 指定的软元件的内容不在以下范围内时。（仅通用型 QCPU）
  $0，2^{-126} \leqq |$指定软元件的内容$| < 2^{128}$（出错代码：4140）
- 指定软元件的内容为 -0 时。*2
  （对于基本型 QCPU、高性能型 QCPU、过程 CPU、冗余 CPU）（出错代码：4100）

  *2:有的 CPU 模块即使指定了 -0 也不会变为运算出错状态。
  有关详细内容请参阅 3.2.4 项。
- 运算结果超出以下范围时。（发生了溢出时。）（仅通用型 QCPU）
  $2^{128} \leqq |$运算结果$|$（出错代码：4141）
- 指定软元件的内容为 -0、非正规数、非数、±∞时。（仅通用型 QCPU）
  （出错代码：4140）

程序示例

(1) 以下为对 X20 ～ X2F 中以 BCD4 位数格式设置的值进行平方根运算后，将其结果以 32 位浮点实数存储到 D0、D1 中的程序。

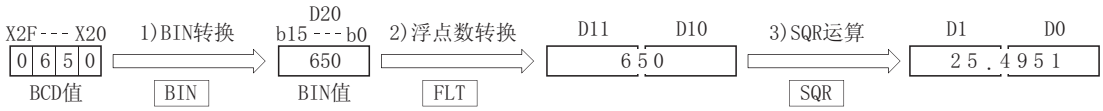
[梯形图模式]



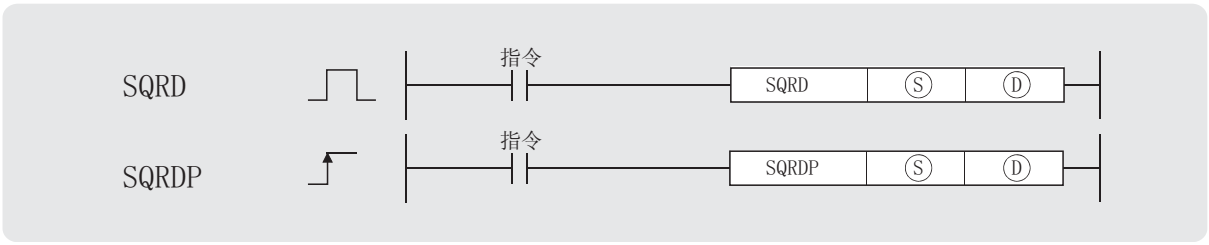
[列表模式]

步	指令	软元件	
0	LD	SM400	
1	BIN	K4X20	D20
4	FLT	D20	D10
7	SQRP	D10	D0
10	END		

[X20 ～ X2F 中指定为 650 时的动作]



7.12.20 浮点数的平方根运算（双精度）(SQRD(P))



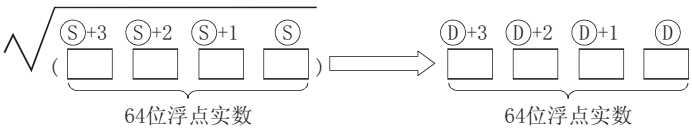
Ⓢ ：进行平方根运算的数据或者存储数据的软元件的起始编号（实数）。

Ⓓ ：存储运算结果的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
Ⓢ	--	○		--				○	--
Ⓓ	--	○		--				--	--

功 能

(1) 对Ⓢ中指定的值进行平方根运算后，将运算结果存储到Ⓓ中指定编号的软元件中。



(2) Ⓢ中指定的值只能设置为正数。（为负数时将无法运算。）

(3) 运算结果为 -0 或者发生了下溢时，将运算结果作为 0 进行处理。

出 错

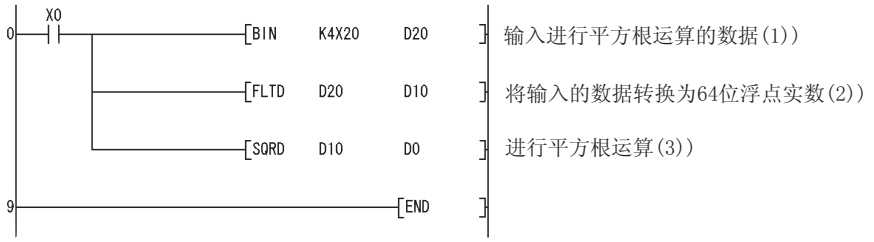
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- Ⓢ中指定的值为负数时。 （出错代码：4100）
- 指定的软元件的内容不在以下范围内时。 （出错代码：4140）
  $0，2^{-1022} \leqq |指定软元件的内容| < 2^{1024}$
- 指定软元件的内容为 -0 时。 （出错代码：4140）
- 运算结果超出以下范围时。（发生了溢出时。）
  $2^{1024} \leqq |运算结果|$ （出错代码：4141）

程序示例

(1) 以下为对 X20 ～ X2F 中以 BCD4 位数格式设置的值进行平方根运算后，将其结果以 64 位浮点实数存储到 D0 ～ D3 中的程序。

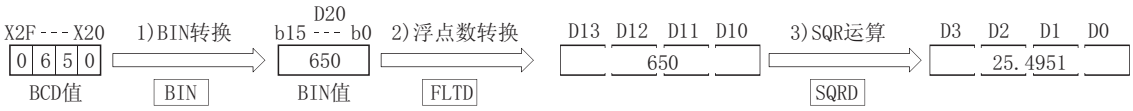
[梯形图模式]



[列表模式]

步	指令	软元件	
0	LD	X0	
1	BIN	K4X20	D20
3	FLTD	D20	D10
6	SQRD	D10	D0
9	END		

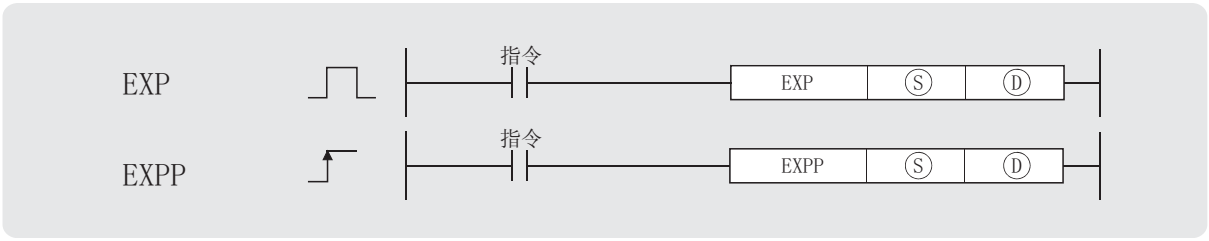
[X20 ～ X2F 中指定为 650 时的动作]



7.12.21 浮点数的指数运算（单精度）(EXP(P))

Ver. Basic | High performance | Process | Redundant | Universal

基本型 QCPU：序列号的前 5 位数为“ 04122 ”以后



Ⓢ ：进行指数运算的数据或者存储数据的软元件的起始编号（实数）。

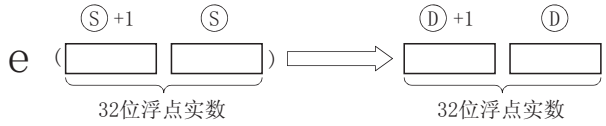
Ⓓ ：存储运算结果的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
Ⓢ	--	○		--	○		○*1	○	--
Ⓓ	--	○		--	○		○*1	--	--

*1：仅通用型 QCPU 可以使用。

功 能

(1) 对Ⓢ中指定的值进行指数运算，并将运算结果存储到Ⓓ中指定编号的位软元件中。



(2) 在指数运算中，将底 (e) 作为“ 2.71828 ”进行运算。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 运算结果不在下述范围内时。 (出错代码： 4100)

$2^{-126} \leqq$ | 运算结果 | $\leqq 2^{128}$ (高性能型 QCPU 时)

$2^{-126} \leqq$ | 运算结果 | $< 2^{128}$ (基本型 QCPU、过程 CPU、冗余 CPU 时)

· 指定软元件的内容为 -0 时。*2
(对于基本型 QCPU、高性能型 QCPU、过程 CPU、冗余 CPU) (出错代码： 4100)

*2:有的 CPU 模块即使指定了 -0 也不会变为运算出错状态。
有关详细内容请参阅 3.2.4 项。

· 运算结果超出以下范围时。(发生了溢出时。) (仅通用型 QCPU)

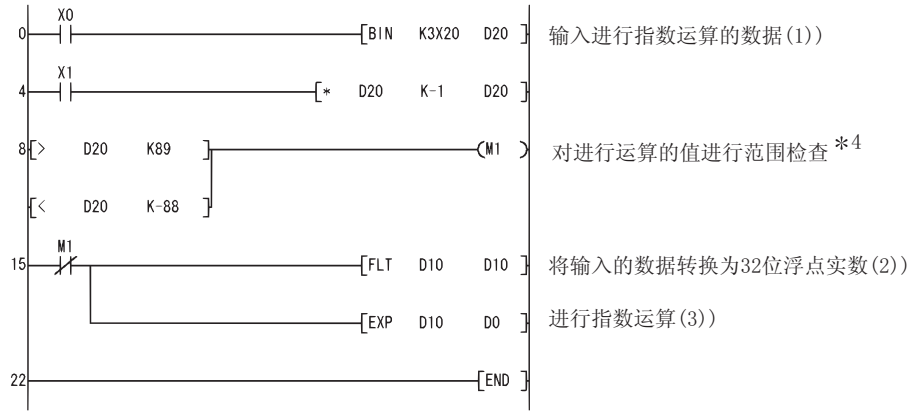
$2^{128} \leqq$ | 运算结果 | (出错代码： 4141)

· 指定软元件的内容为 -0、非正规数、非数、±∞时。(仅通用型 QCPU)
(出错代码： 4140)

程序示例

(1) 以下为以 X20 ～ X27 中以 BCD2 位数格式设置的值进行指数运算后，将其结果以 32 位浮点实数存储到 D0、D1 中的程序。

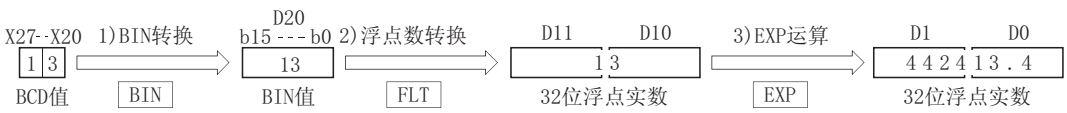
[梯形图模式]



[列表模式]

步	指令	软元件		
0	LD	X0		
1	BIN	K3X20	D20	
4	LD	X1		
5	*	D20	K-1	D20
8	LD>	D20	K89	
11	OR<	D20	K-88	
14	OUT	M1		
15	LDI	M1		
16	FLT	D10	D10	
19	EXP	D10	D0	
22	END			

[X20 ～ X27 中指定为 13 时的动作]



*4: 由于 loge 2^{129}=89.4，为了让运算结果低于 2^{129}，X20 ～ X27 的 BCD 值应为 89 以下。
由于设置值超过 90 以上时将会变为运算出错状态，因此应将程序设置为，如果设置了 90 以上的值，则 M0 将变为 ON，不执行运算。

要 点

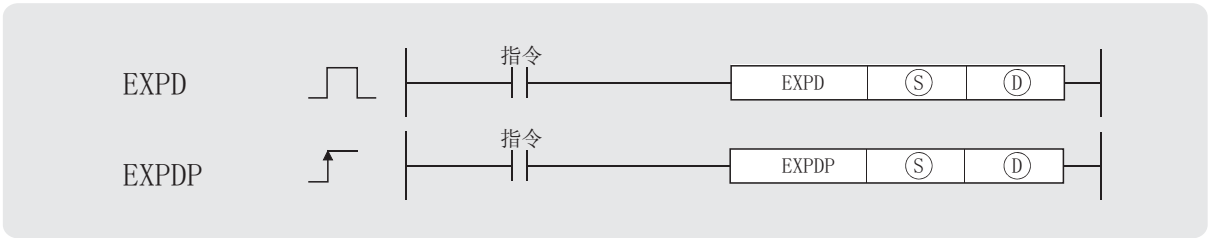
自然对数至常用对数的转换

在 CPU 中是使用自然对数进行运算的。

若要对常用对数值进行计算，应将常用对数值用 0.43429 相除后的值指定到Ⓢ中。

$$10^{X}=e^{\frac{X}{0.43429}}$$

7.12.22 浮点数的指数运算（双精度）(EXPD(P))



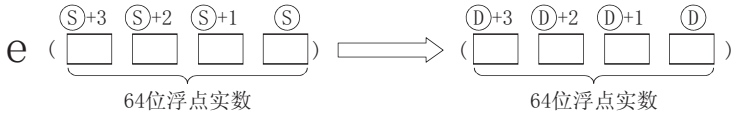
Ⓢ：进行指数运算的数据或者存储数据的软元件的起始编号（实数）。

Ⓓ：存储运算结果的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			E	
Ⓢ	--	○		--				○	--
Ⓓ	--	○		--				--	--

功 能

(1) 以Ⓢ中指定的值进行指数运算，并将运算结果存储到Ⓓ中指定编号的位软元件中。



(2) 在指数运算中，将底 (e) 作为“2.71828”进行运算。

(3) 运算结果为 -0 或者发生了下溢时，将运算结果作为 0 进行处理。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 指定的软元件的内容不在以下范围内时。（出错代码：4140）

$0，2^{-1022} \leqq |指定软元件的内容| < 2^{1024}$

· 指定软元件的内容为 -0 时。（出错代码：4140）

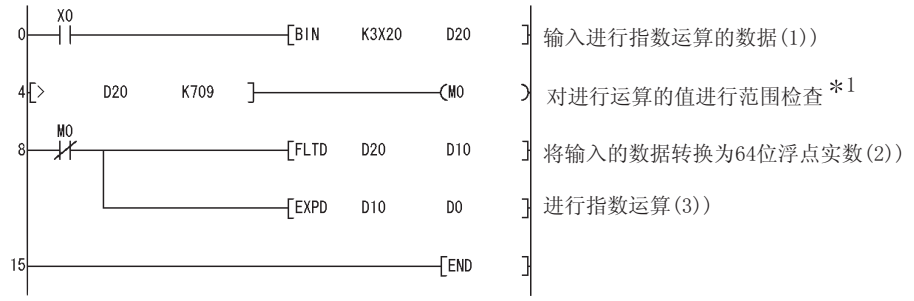
· 运算结果超出以下范围时。（发生了溢出时。）（仅通用型 QCPU）

$2^{1024} \leqq |运算结果|$（出错代码：4141）

程序示例

(1) 以下为以 X20 ～ X31 中以 BCD2 位数格式设置的值进行指数运算后，将其结果以 64 位浮点实数存储到 D0 ～ D3 中的程序。

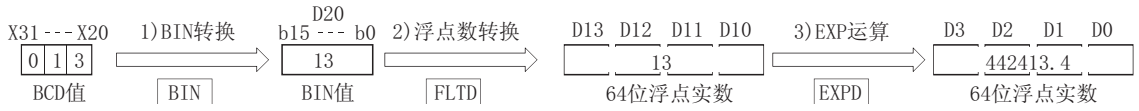
[梯形图模式]



[列表模式]

步	指令	软元件	
0	LD	X0	
1	BIN	K3X20	D20
4	LD>	D20	K709
7	OUT	M0	
8	LDI	M0	
9	FLTD	D20	D10
12	EXPD	D10	D0
15	END		

[X20 ～ X31 中指定为 13 时的动作]



*1: 由于 loge 2^{1024}=709.7832，为了让运算结果低于 2^{1024}，X20 ～ X31 的 BCD 值应为 709 以下。
由于设置值超过 710 以上时将会变为运算出错状态，因此应将程序设置为，如果设置了 710 以上的值，则 M0 将变为 ON，不执行运算。

要 点

自然对数至常用对数的转换

在 CPU 中是使用自然对数进行运算的。

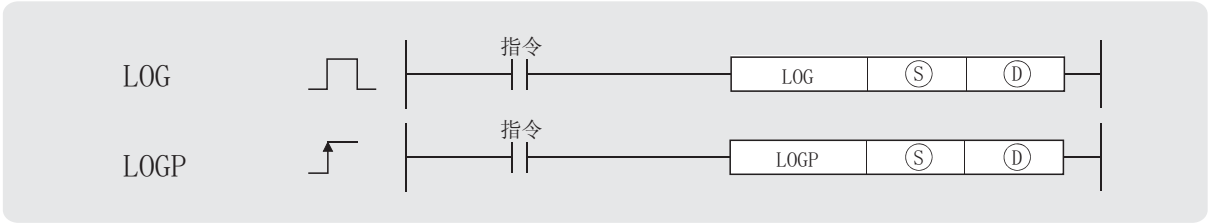
若要对常用对数值进行计算，应将常用对数值用 0.43429 相除后的值指定到Ⓢ中。

$$10^{x}=e^{\frac{x}{0.43429}}$$

7.12.23 浮点数的自然对数运算（单精度）(LOG(P))

Ver. Basic | High performance | Process | Redundant | Universal

基本型 QCPU：序列号的前 5 位数为“ 04122 ”以后



Ⓢ ：进行自然对数运算的数据或者存储数据的软元件的起始编号（实数）。

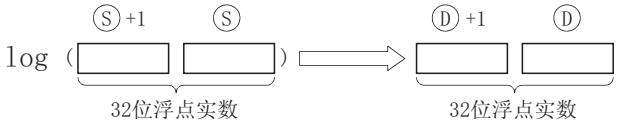
Ⓓ ：存储运算结果的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
Ⓢ	--	○		--	○		○ *1	○	--
Ⓓ	--	○		--	○		○ *1	--	--

*1：仅通用型 QCPU 可以使用。

功 能

(1) 对Ⓢ中指定值的以自然对数 (e) 为底时的对数进行运算，并将运算结果存储到Ⓓ中指定编号的位软元件中。



(2) Ⓢ中指定的值只能设置为正数。（设置为负数时不能执行运算。）

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

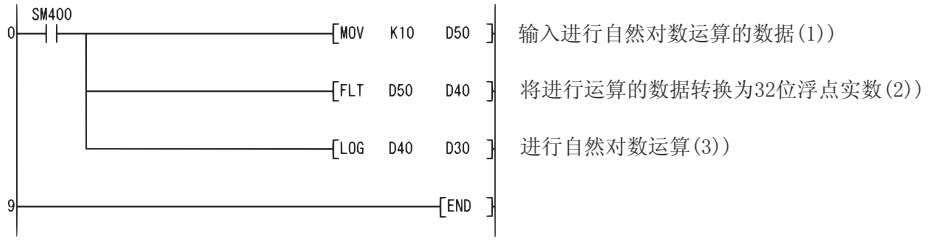
- Ⓢ中指定的值为负数时。 (出错代码： 4100)
- Ⓢ中指定的值为 “ 0 ” 时。 (出错代码： 4100)
- 指定的软元件的内容不在以下范围内时。(仅通用型 QCPU)
  0，2^{-126} ≦ | 指定软元件的内容 | ＜ 2^{128} (出错代码： 4140)
- 指定软元件的内容为 -0 时。*2
  (对于基本型 QCPU、高性能型 QCPU、过程 CPU、冗余 CPU) (出错代码： 4100)

  *2: 有的 CPU 模块即使指定了 -0 也不会变为运算出错状态。
  有关详细内容请参阅 3.2.4 项。
- 运算结果超出以下范围时。(发生了溢出时。) (仅通用型 QCPU)
  2^{128} ≦ | 运算结果 | (出错代码： 4141)
- 指定软元件的内容为 -0、非正规数、非数、±∞时。
  (仅通用型 QCPU) (出错代码： 4140)

程序示例

(1) 以下为求出 D50 中设置的 “ 10 ” 的自然对数后，存储到 D30、D31 中的程序。

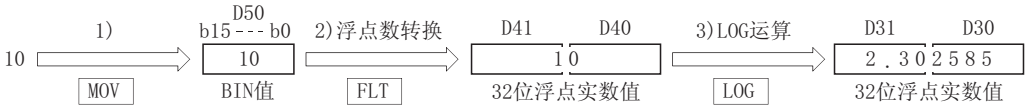
[梯形图模式]



[列表模式]

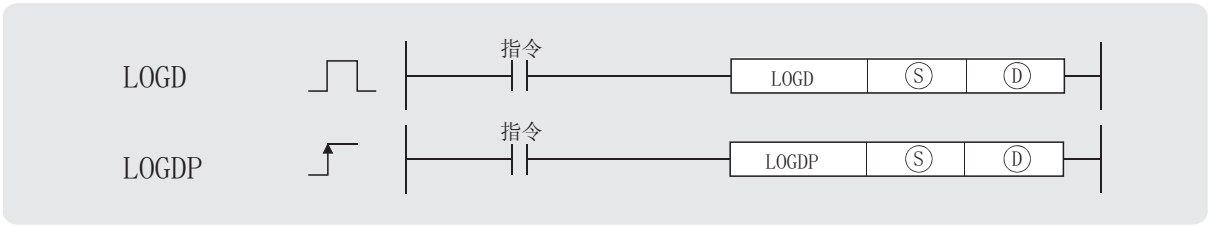
步	指令	软元件	
0	LD	SM400	
1	MOV	K10	D50
3	FLT	D50	D40
6	LOG	D40	D30
9	END		

[动作]



7.12.24 浮点数的自然对数运算（双精度）(LOGD(P))

Basic High performance Process Redundant Universal



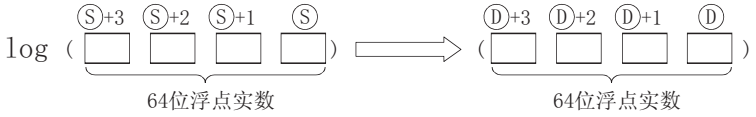
Ⓢ ：进行自然对数运算的数据或者存储数据的软元件的起始编号（实数）。

Ⓓ ：存储运算结果的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
Ⓢ	--	○		--				○	--
Ⓓ	--	○		--				--	--

功 能

(1) 对Ⓢ中指定值的以自然对数 (e) 为底时的对数进行运算，并将运算结果存储到Ⓓ中指定编号的位软元件中。



(2) Ⓢ中指定的值只能设置为正数。（设置为负数时不能执行运算。）

(3) 运算结果为 -0 或者发生了下溢时，将运算结果作为 0 进行处理。

出 错

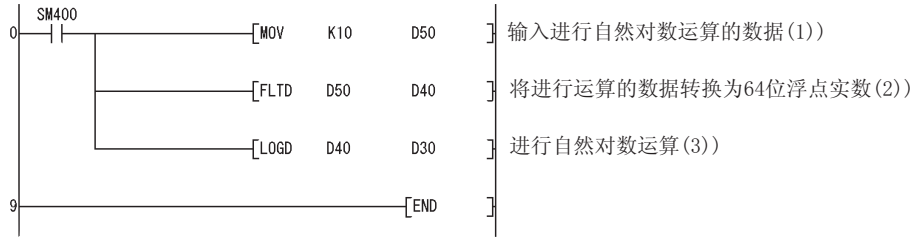
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- Ⓢ中指定的值为负数时。（出错代码：4100）
- Ⓢ中指定的值为“0”时。（出错代码：4100）
- 指定的软元件的内容不在以下范围内时。（出错代码：4140）
  $0，2^{-1022} \leqq |指定软元件的内容| < 2^{1024}$
- 指定软元件的内容为 -0 时。（出错代码：4140）
- 运算结果超出以下范围时。（发生了溢出时。）
  $2^{1024} \leqq |运算结果|$（出错代码：4141）

程序示例

(1) 以下为求出 D50 中设置的 “ 10 ” 的自然对数后，存储到 D30 ～ D33 中的程序。

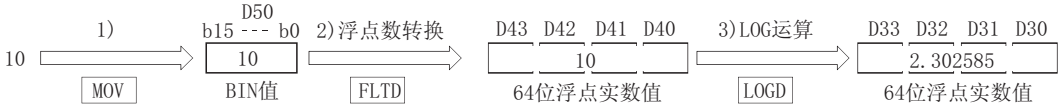
[梯形图模式]



[列表模式]

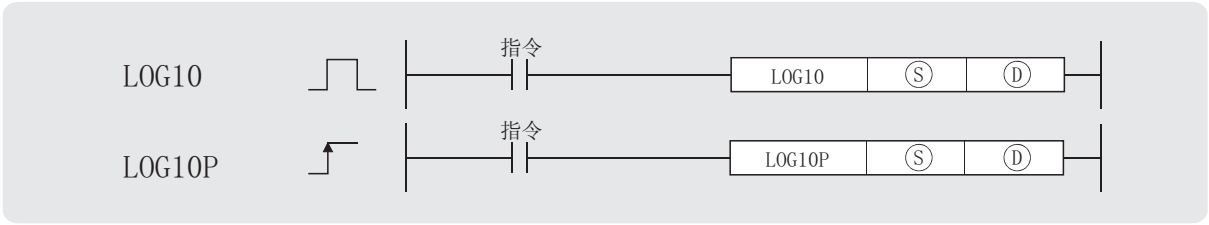
步	指令	软元件	
0	LD	SM400	
1	MOV	K10	D50
3	FLTD	D50	D40
6	LOGD	D40	D30
9	END		

[动作]



7.12.25 浮点数的常用对数运算（单精度）(LOG10(P))

· QnU(D)(H)CPU：序列号的前 5 位数为“ 10102 ”以后
· QnUDE(H)CPU：序列号的前 5 位数为“ 10102 ”以后



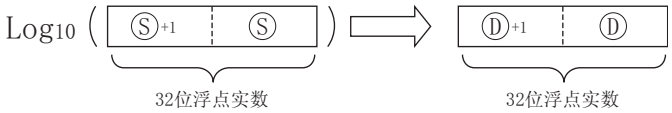
(S) ：进行常用对数运算的数据或者存储数据的软元件的起始编号（实数）。
(D) ：存储运算结果的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
(S)	--	○		--	○		--	△ *1	--
(D)	--	○		--	○		--	--	--

*1：仅实数可以使用。

功 能

(1) 对(S)中指定的值进行常用对数（以 10 为底的对数）运算，并将运算结果存储到(D)中指定编号的位软元件中。



(2) (S)中指定的值只能设置为正数。（设置为负数时不能执行运算。）

(3) 运算结果为 -0 或者发生了下溢时，将运算结果作为 0 处理。

出 错

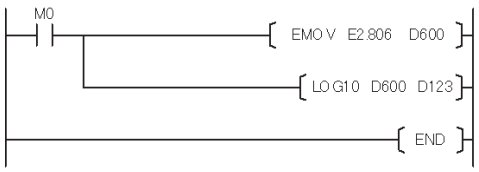
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- ⓢ中指定的值为负数时。 (出错代码： 4100)
- ⓢ中指定的值为 “ 0 ” 时。 (出错代码： 4100)
- 指定的软元件的内容不在以下范围内时。 (出错代码： 4140)
  0，2^{-126} ≦ | 指定软元件的内容 | ＜ 2^{128}
- ⓢ中指定的值为 “ -0 ” 时。 (出错代码： 4140)
- 运算结果超出以下范围时。(发生了溢出时。)
  2^{128} ≦ | 运算结果 | (出错代码： 4141)

程序示例

(1) 以下为 M0 变为 ON 时，对 D600、D601 中存储的 32 位浮点实数的常用对数进行计算后，将运算结果存储到 D123、D124 中的程序。

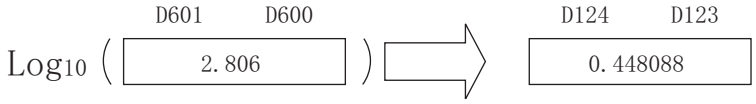
[梯形图模式]



[列表模式]

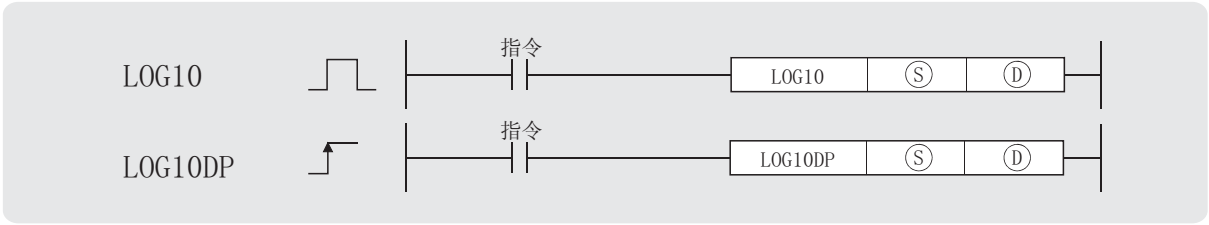
步	指令	软元件	
0	LD	M0	
1	EMOV	E2.806	D600
4	LOG10	D600	D123
7	END		

[动作]



7.12.26 浮点数的常用对数运算（双精度）(LOG10D(P))

· QnU(D)(H)CPU：序列号的前 5 位数为“ 10102 ”以后
· QnUDE(H)CPU：序列号的前 5 位数为“ 10102 ”以后



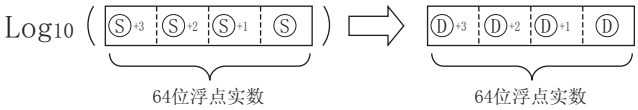
(S) ：进行常用对数运算的数据或者存储数据的软元件的起始编号（实数）。
(D) ：存储运算结果的软元件的起始编号（实数）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
(S)	--	○		--				△ *1	--
(D)	--	○		--				--	--

*1：仅实数可以使用。

功 能

(1) 对(S)中指定的值进行常用对数（以 10 为底的对数）运算，并将运算结果存储到(D)中指定编号的位软元件中。



(2) (S)中指定的值只能设置为正数。（设置为负数时不能执行运算。）

(3) 运算结果为 -0 或者发生了下溢时，将运算结果作为 0 处理。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· (S)中指定的值为负数时。（出错代码： 4100）

· (S)中指定的值为“ 0 ”时。（出错代码： 4100）

· 指定的软元件的内容不在以下范围内时。（出错代码： 4140）
  $0，2^{-1022} \leqq |$指定软元件的内容$| < 2^{1024}$

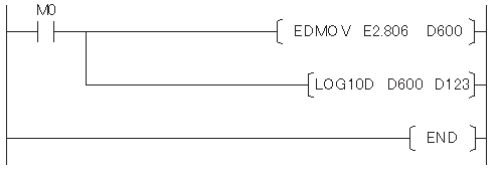
· (S)中指定的值为“ -0 ”时。（出错代码： 4140）

· 运算结果超出以下范围时。（发生了溢出时。）
  $2^{1024} \leqq |$运算结果$|$（出错代码： 4141）

程序示例

(1) 以下为 M0 变为 ON 时，对 D600 ～ D603 中存储的 64 位浮点实数的常用对数进行计算后，将运算结果存储到 D123 ～ D126 中的程序。

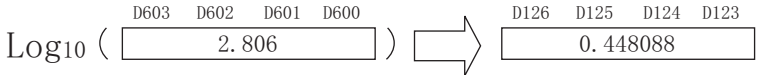
[梯形图模式]



[列表模式]

步	指令	软元件	
0	LD	M0	
1	EDMOV	E2.806	D600
4	LOG10D	D600	D123
7	END		

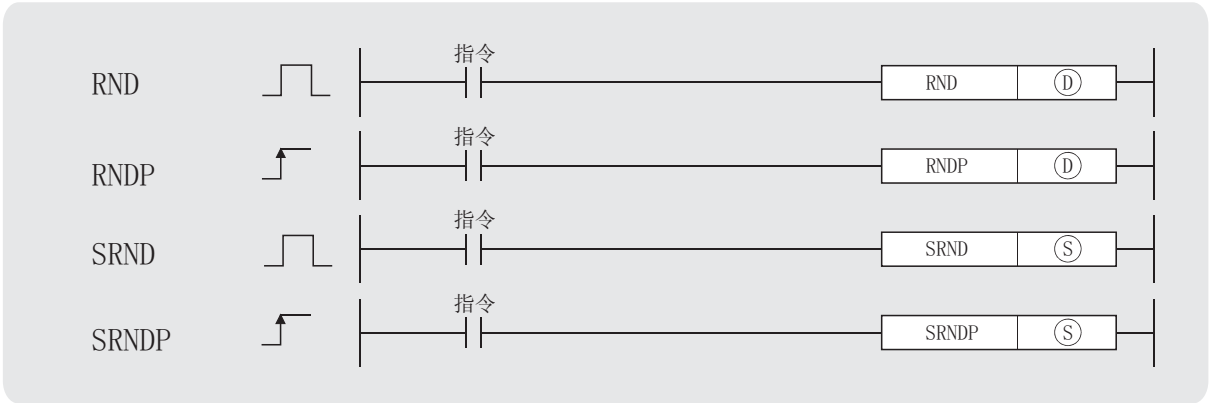
[动作]



7.12.27 随机数的产生和系列变更 (RND(P)、SRND(P))

Ver. Basic | High performance | Process | Redundant | Universal

基本型 QCPU：序列号的前 5 位数为“ 04122 ”以后



Ⓓ ：存储随机数的软元件的起始编号 (BIN16 位)。

Ⓢ ：随机数系列数据或者存储随机系列数据的软元件的起始编号 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓓ				○				--	--
Ⓢ				○				○	--

功 能

随机数产生指令用于按照某个计算公式生成随机数。在计算公式中，将上次的计算结果作为系数使用。
通过系列变更指令可以更改随机数生成模式。

RND

生成 0 ～ 32767 的随机数后，存储到Ⓓ中指定的软元件中。

SRND

按照Ⓢ中指定的软元件中存储的 16 位 BIN 数据的内容对随机数系列进行变更。

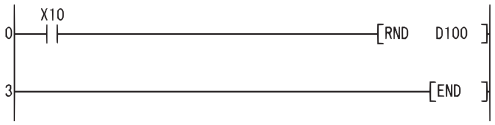
出　错

(1) 在 RND(P)、SRND(P) 指令中无运算出错。

程序示例

(1) 以下为 X10 变为 ON 时，将随机数存储到 D100 中的程序。

[梯形图模式]

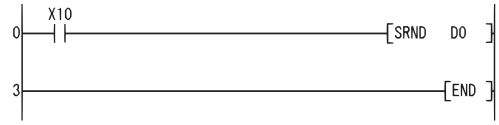


[列表模式]

步	指令	软元件
0	LD	X10
1	RND	D100
3	END	

(2) 以下为 X10 变为 ON 时，按照 D0 中的内容对随机数系列进行变更的程序。

[梯形图模式]

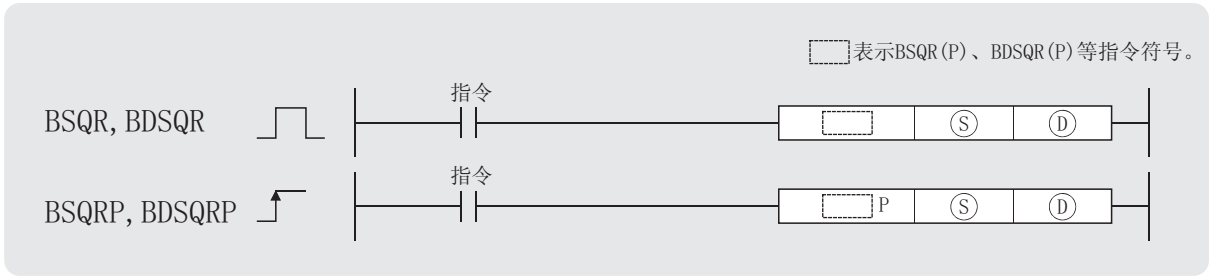


[列表模式]

步	指令	软元件
0	LD	X10
1	SRND	D0
3	END	

7.12.28 BCD4 位和 8 位平方根 (BSQR(P)、BDSQR(P))

Basic High performance Process Redundant Universal



Ⓢ ：进行平方根运算的数据或者存储数据的软元件的起始编号 (BSQR(P)：BCD4 位；BDSQR(P)：BCD8 位)。

Ⓓ ：存储运算结果的软元件的起始编号 (BCD4 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○							○	--
Ⓓ	○							--	--

功 能

BSQR

(1) 进行Ⓢ中指定值的平方根运算后，将运算结果存储到Ⓓ中指定的位软元件中。

$$\sqrt{Ⓢ} = \underset{Ⓓ}{\text{整数部分}}.\underset{Ⓓ+1}{\text{小数部分}}$$

(2) Ⓢ中指定值的范围为最多 4 位数的 BCD 值 (0 ～ 9999)。

(3) Ⓓ、Ⓓ+1 的运算结果分别以 0 ～ 9999 的 BCD 值进行存储。

(4) 运算结果为小数部分第 5 位数被四舍五入后的值。
因此，小数部分第 4 位数将产生 ± 1 的误差。

BDSQR

(1) 进行Ⓢ、Ⓢ+1 中指定值的平方根运算后，将运算结果存储到Ⓓ中指定的位软元件中。

$$\sqrt{(\underbrace{Ⓢ+1 \quad Ⓢ}_{\text{2字数据}})} = \underset{Ⓓ}{\text{整数部分}}.\underset{Ⓓ+1}{\text{小数部分}}$$

(2) Ⓢ、Ⓢ+1 中指定值的范围为最多 8 位数的 BCD 值 (0 ～ 99999999)。

(3) Ⓓ、Ⓓ+1 的运算结果分别以 0 ～ 9999 的 BCD 值进行存储。

(4) 运算结果为小数部分第 5 位数被四舍五入后的值。
因此，小数部分第 4 位数将产生 ± 1 的误差。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· ⓢ中指定的数据不是 BCD 值时。 (出错代码： 4100)

程序示例

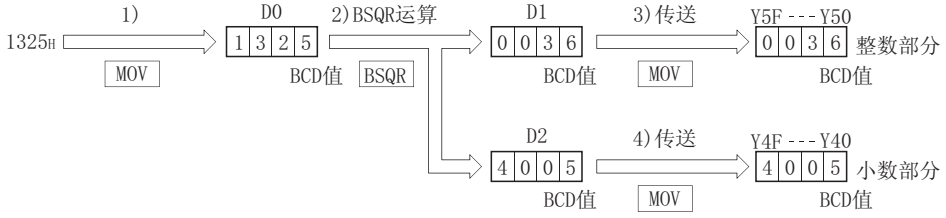
(1) 以下为对 BCD 值 1325 进行平方根运算后，将整数部分以 BCD4 位数输出到 Y50 ～ Y5F 中，将小数部分以 BCD4 位数输出到 Y40 ～ Y4F 中的程序。

[梯形图模式]

[列表模式]

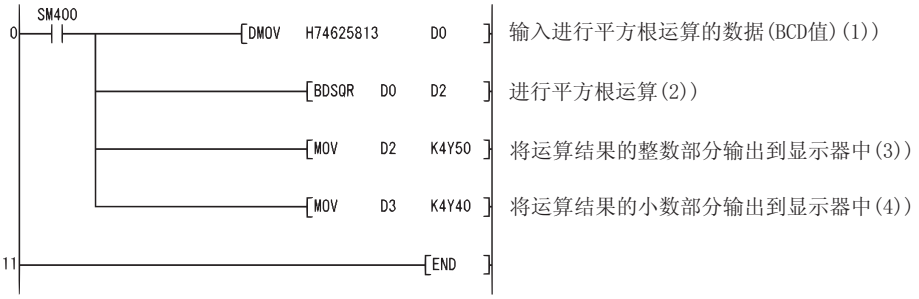
步	指令	软元件	
0	LD	SM400	
1	MOV	H1325	D0
3	BSQR	D0	D1
6	MOV	D1	K4Y50
8	MOV	D2	K4Y40
10	END		

[动作]



(2) 以下为对 BCD 值 74625813 进行平方根运算后，将整数部分以 BCD4 位数输出到 Y50 ～ Y5F 中，将小数部分以 BCD4 位数输出到 Y40 ～ Y4F 中的程序。

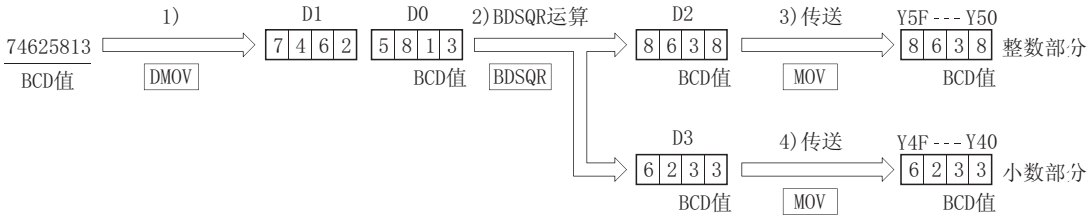
[梯形图模式]



[列表模式]

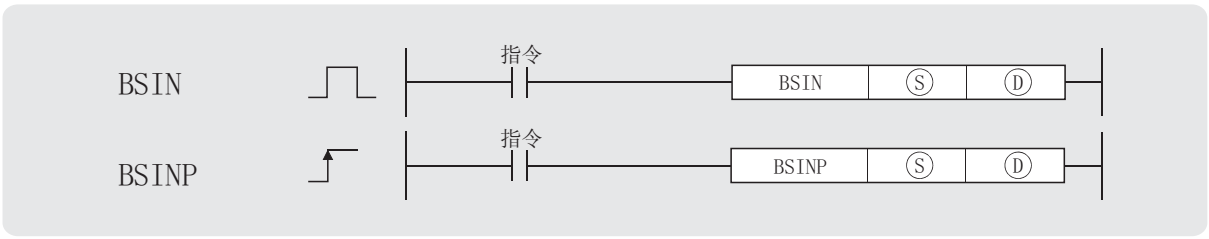
步	指令	软元件
0	LD	SM400
1	DMOV	H74625813 D0
4	BDSQR	D0 D2
7	MOV	D2 K4Y50
9	MOV	D3 K4Y40
11	END	

[动作]



7.12.29 BCD 型 SIN 运算 (BSIN(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ : 进行 SIN(正弦) 运算的数据或者存储数据的软元件的起始编号 (BCD4 位数)。

Ⓓ : 存储运算结果的软元件的起始编号 (BCD4 位数)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○	○		○					--
Ⓓ	--	○		--					--

功 能

(1) 对Ⓢ中指定的值 (角度) 进行 SIN(正弦) 运算，并将运算结果的符号存储到Ⓓ中指定的软元件中，将运算结果存储到Ⓓ+1、Ⓓ+2 中指定的软元件中。

SIN Ⓢ = [Ⓓ 符号] [Ⓓ+1 整数部分] . [Ⓓ+2 小数部分]

(2) Ⓢ中指定的值是以 0 ～ 360 ° (DEG. 单位) 的 BCD 值进行设置的。

(3) 对于Ⓓ中存储的运算结果的符号，在运算结果为正时存储 “ 0 ”，为负时存储 “ 1 ”。

(4) 存储到Ⓓ+1，Ⓓ+2 中的运算结果的范围为 -1.000 ～ 1.000 的 BCD 值。

(5) 运算结果为小数部分第 5 位数被四舍五入后的值。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

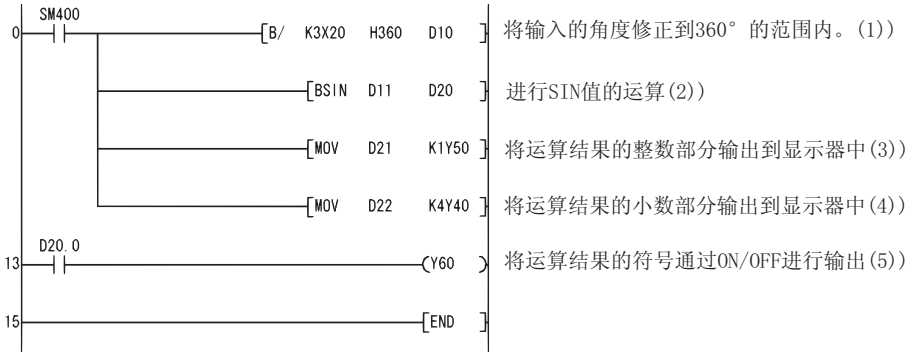
- Ⓢ中指定的数据不是 BCD 值时。 (出错代码 : 4100)
- Ⓢ中指定的数据超出了 0 ～ 360 的范围时。 (出错代码 : 4100)
- Ⓓ中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)
  (出错代码 : 4101)

程序示例

(1) 以下为对 X20 ～ X2B 中以 BCD3 位数指定的数据进行 SIN 运算后，将整数部分以 BCD1 位数输出到 Y50 ～ Y53 中，将小数部分以 BCD4 位数输出到 Y40 ～ Y4F 中的程序。

运算结果为负数时将 Y60 置为 ON。(在 X20 ～ X2F 中设置了 360 以上的值时，将所设置的值修正到 0 ～ 360 的范围内。)

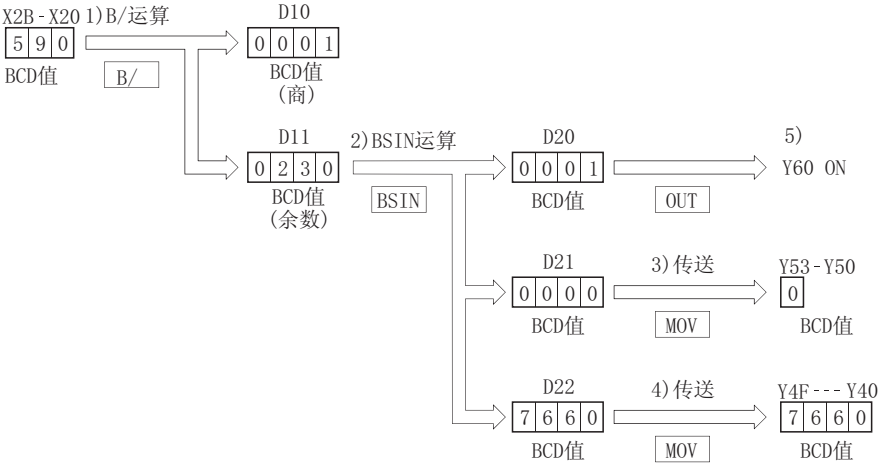
[梯形图模式]



[列表模式]

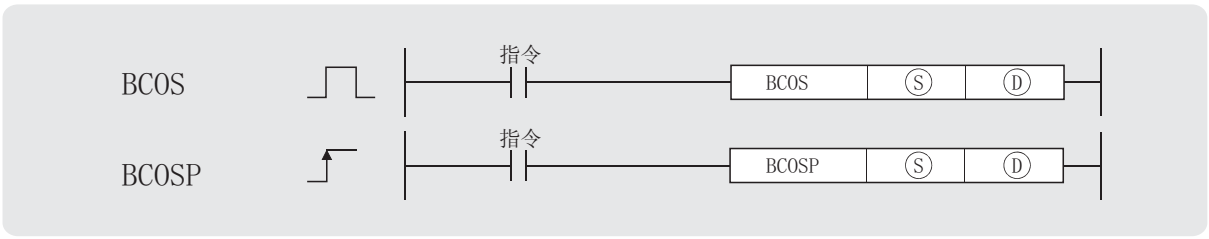
步	指令	软元件		
0	LD	SM400		
1	B/	K3X20	H360	D10
5	BSIN	D11	D20	
8	MOV	D21	K1Y50	
11	MOV	D22	K4Y40	
13	LD	D20.0		
14	OUT	Y60		
15	END			

[在 X20 ～ X2B 中指定了 590 时的动作]



7.12.30 BCD 型 COS 运算 (BCOS(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ : 进行 COS(余弦) 运算的数据或者存储数据的软元件的起始编号 (BCD4 位数)。

Ⓓ : 存储运算结果的软元件的起始编号 (BCD4 位数)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○	○		○					--
Ⓓ	--	○		--					--

功 能

(1) 对Ⓢ中指定的值 (角度) 进行 COS(余弦) 运算，并将运算结果的符号存储到Ⓓ中指定的字软元件中，将运算结果存储到Ⓓ+1、Ⓓ+2 中指定的字软元件中。

COS Ⓢ = Ⓓ 符号 | Ⓓ+1 整数部分 . Ⓓ+2 小数部分

(2) Ⓢ中指定的值是以 0 ～ 360 ° (DEG. 单位) 的 BCD 值进行设置的。

(3) 对于Ⓓ中存储的运算结果的符号，在运算结果为正时存储 “ 0 ”，为负时存储 “ 1 ”。

(4) 存储到Ⓓ+1、Ⓓ+2 中的运算结果的范围为 -1.000 ～ 1.000 的 BCD 值。

(5) 运算结果为小数部分第 5 位数被四舍五入后的值。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

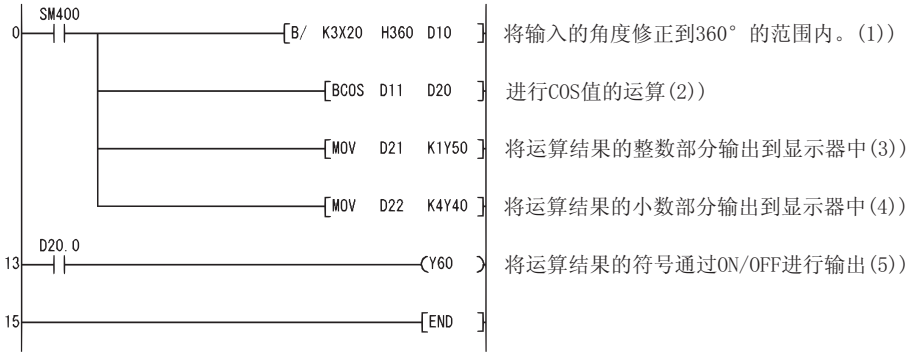
- Ⓢ中指定的数据不是 BCD 值时。 (出错代码 : 4100)
- Ⓢ中指定的数据超出了 0 ～ 360 的范围时。 (出错代码 : 4100)
- Ⓓ中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)
  (出错代码 : 4101)

程序示例

(1) 以下为对 X20 ～ X2B 中以 BCD3 位数指定的数据进行 COS 运算后，将整数部分以 BCD1 位数输出到 Y50 ～ Y53 中，将小数部分以 BCD4 位数输出到 Y40 ～ Y4F 中的程序。

运算结果为负数时将 Y60 置为 ON。

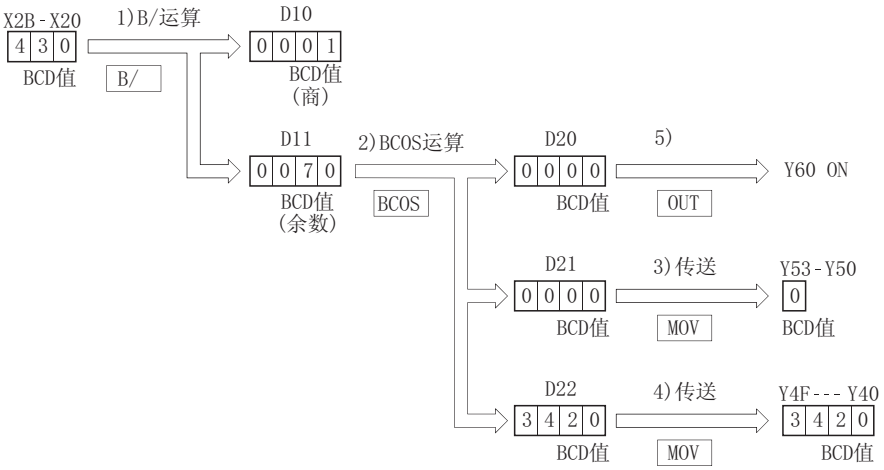
[梯形图模式]



[列表模式]

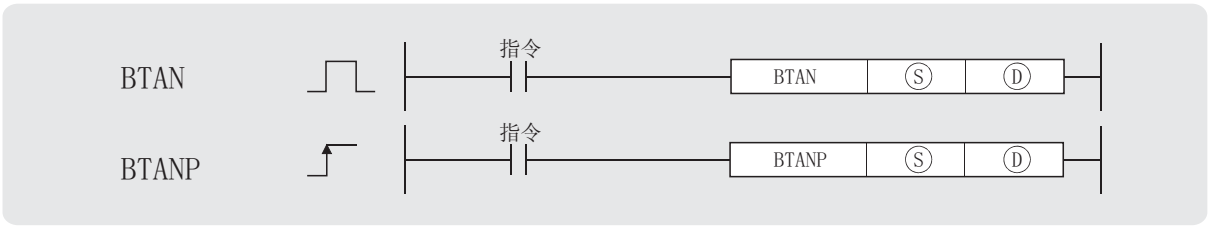
步	指令	软元件		
0	LD	SM400		
1	B/	K3X20	H360	D10
5	BCOS	D11	D20	
8	MOV	D21	K1Y50	
11	MOV	D22	K4Y40	
13	LD	D20.0		
14	OUT	Y60		
15	END			

[在 X20 ～ X2B 中指定了 430 时的动作]



7.12.31 BCD 型 TAN 运算 (BTAN(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ : 进行 TAN(正切) 运算的数据或者存储数据的软元件的起始编号 (BCD4 位数)。

Ⓓ : 存储运算结果的软元件的起始编号 (BCD4 位数)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○	○		○					--
Ⓓ	--	○		--					--

功 能

(1) 对Ⓢ中指定的值 (角度) 进行 TAN(正切) 运算，并将运算结果的符号存储到Ⓓ中指定的软元件中，将运算结果存储到Ⓓ+1、Ⓓ+2 中指定的软元件中。

TAN Ⓢ = Ⓓ 符号 | Ⓓ+1 整数部分 . Ⓓ+2 小数部分

(2) Ⓢ中指定的值是以 0 ～ 360 ° (DEG. 单位) 的 BCD 值进行设置的。

(3) 对于Ⓓ中存储的运算结果的符号，在运算结果为正时存储 “ 0 ”，为负时存储 “ 1 ”。

(4) 存储到Ⓓ+1、Ⓓ+2 中的运算结果的范围为 -57.2901 ～ 57.2902 的 BCD 值。

(5) 运算结果为小数部分第 5 位数被四舍五入后的值。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

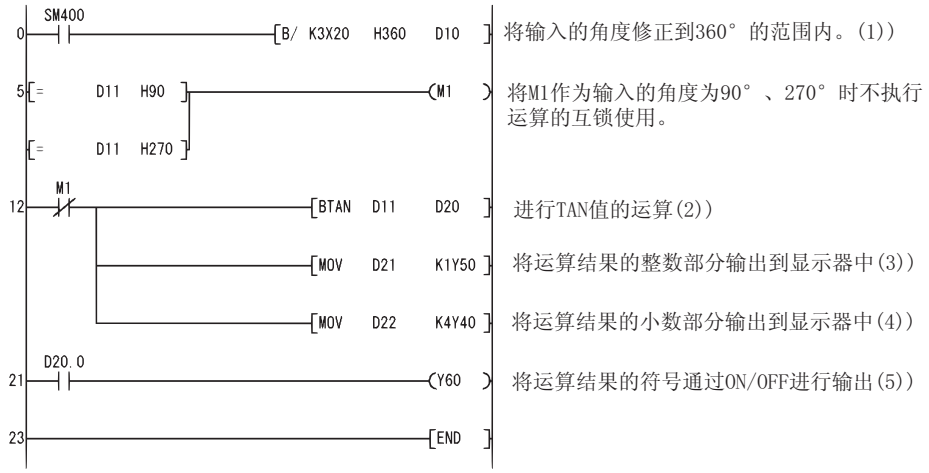
- Ⓢ中指定的数据不是 BCD 值时。 (出错代码 : 4100)
- Ⓢ中指定的数据超出了 0 ～ 360 的范围时。 (出错代码 : 4100)
- Ⓢ中指定的数据为 90 °、270 ° 时。 (出错代码 : 4100)
- Ⓓ中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)
  (出错代码 : 4101)

程序示例

(1) 以下为对 X20 ～ X2B 中以 BCD3 位数指定的数据进行 TAN 运算后，将整数部分以 BCD4 位数输出到 Y50 ～ Y53 中，将小数部分以 BCD4 位数输出到 Y40 ～ Y4F 中的程序。

运算结果为负数时将 Y60 置为 ON。

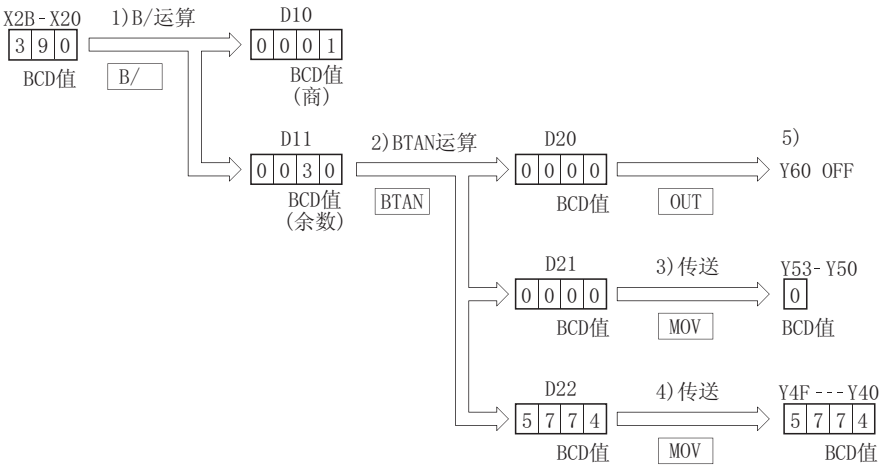
[梯形图模式]



[列表模式]

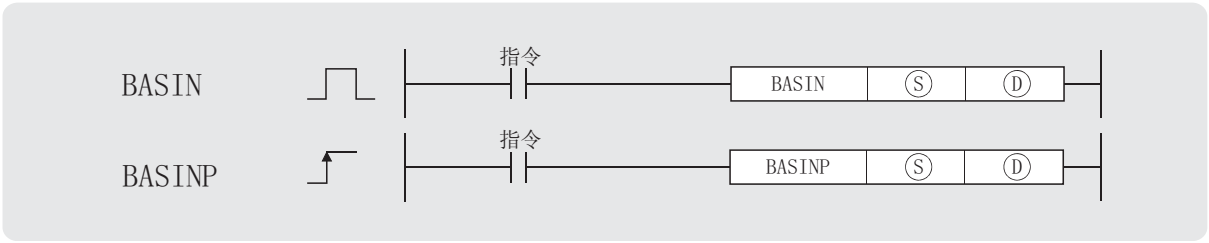
步	指令	软元件		
0	LD	SM400		
1	B/	K3X20	H360	D10
5	LD=	D11	H90	
8	OR=	D11	H270	
11	OUT	M1		
12	LDI	M1		
13	BTAN	D11	D20	
16	MOV	D21	K1Y50	
19	MOV	D22	K4Y40	
21	LD	D20.0		
22	OUT	Y60		
23	END			

[在 X20 ～ X2B 中指定了 390 时的动作]



7.12.32 BCD 型 SIN $^{-1}$ 运算 (BASIN(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ : 进行 SIN^{-1}(反正弦) 运算的数据或者存储数据的软元件的起始编号 (BCD4 位数)。

Ⓓ : 存储运算结果的软元件的起始编号 (BCD4 位数)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
Ⓢ	--	○		--				--	
Ⓓ	○	○		○				--	

功 能

(1) 对Ⓢ中指定的值 (角度) 进行 SIN^{-1}(反正弦) 运算，并将运算结果 (角度) 存储到Ⓓ中指定的软元件中。

$$SIN^{-1}(\underset{\text{符号}}{\boxed{\text{Ⓢ}}}\ \underset{\text{整数部分}}{\boxed{\text{Ⓢ}+1}}.\underset{\text{小数部分}}{\boxed{\text{Ⓢ}+2}}) = \text{Ⓓ}$$

(2) 在Ⓢ中设置进行运算的数据的符号。
运算数据为正时存储 “ 0 ”，为负时存储 “ 1 ”。

(3) 在Ⓢ+1、Ⓢ+2 中，分别以 BCD 值存储运算数据的整数部分及小数部分。(可设置范围为 0 ～ 1.0000。)

(4) 存储到Ⓓ中的运算结果为 0 ～ 90 °、270 ～ 360 ° (DEG. 单位) 范围内的 BCD 值。

(5) 运算结果为小数部分被四舍五入后的值。

出 错

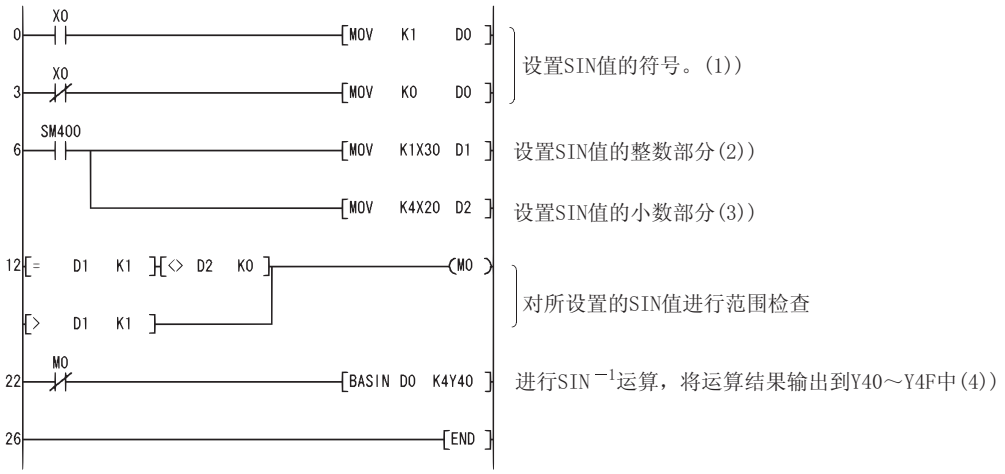
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- Ⓢ中指定的数据不是 BCD 值时。 (出错代码 : 4100)
- Ⓢ中指定的数据超出了 -1.0000 ～ 1.0000 的范围时。 (出错代码 : 4100)
- Ⓢ中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)
  (出错代码 : 4101)

程序示例

(1) 以下为对 X0 中设置了正负符号 (OFF 时为正，ON 时为负)，X30 ～ X33 中以 BCD1 位数设置了整数部分，X20 ～ X2F 中以 BCD4 位数设置了小数部分的值进行 SIN^{-1} 运算后，将求出的角度以 BCD4 位数输出到 Y40 ～ Y4F 中的程序。

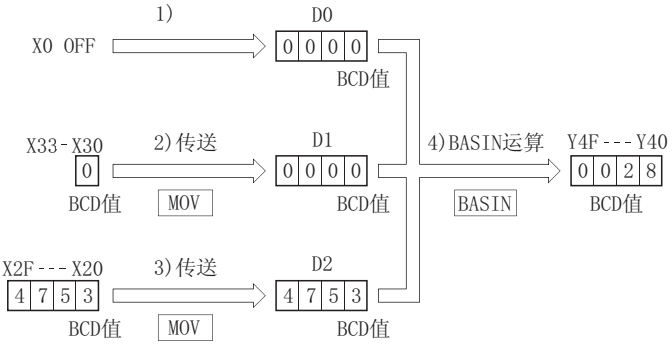
[梯形图模式]



[列表模式]

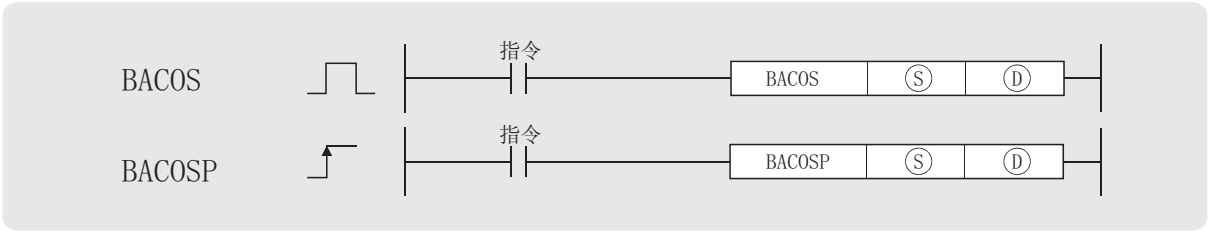
步	指令	软元件	
0	LD	X0	
1	MOV	K1	D0
3	LDI	X0	
4	MOV	K0	D0
6	LD	SM400	
7	MOV	K1X30	D1
10	MOV	K4X20	D2
12	LD=	D1	K1
15	AND<>	D2	K0
18	OR>	D1	K1
21	OUT	M0	
22	LDI	M0	
23	BASIN	D0	K4Y40
26	END		

[在 X20 ～ X33 中指定了 0.4753 时的动作]



7.12.33 BCD 型 COS $^{-1}$ 运算 (BACOS(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ : 进行 COS^{-1}(反余弦) 运算的数据或者存储数据的软元件的起始编号 (BCD4 位数)。

Ⓓ : 存储运算结果的软元件的起始编号 (BCD4 位数)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
Ⓢ	--	○		--				--	
Ⓓ	○	○		○				--	

功 能

(1) 对Ⓢ中指定的值 (角度) 进行 COS^{-1}(反余弦) 运算，并将运算结果 (角度) 存储到Ⓓ中指定的软元件中。

$$\cos^{-1}\left(\underset{\text{Ⓢ}}{\boxed{\text{符号}}}\ \underset{\text{Ⓢ}+1}{\boxed{\text{整数部分}}}.\ \underset{\text{Ⓢ}+2}{\boxed{\text{整数部分}}}\right)=\text{Ⓓ}$$

(2) 在Ⓢ中设置进行运算的数据的符号。
运算数据为正时存储 “ 0 ”，为负时存储 “ 1 ”。

(3) 在Ⓢ+1、Ⓢ+2 中，分别以 BCD 值存储运算数据的整数部分及小数部分。
(可设置范围为 0 ～ 1.0000。)

(4) 存储到Ⓓ中的运算结果为 0 ～ 180 ° (DEG. 单位) 范围内的 BCD 值。

(5) 运算结果为小数部分被四舍五入后的值。

出 错

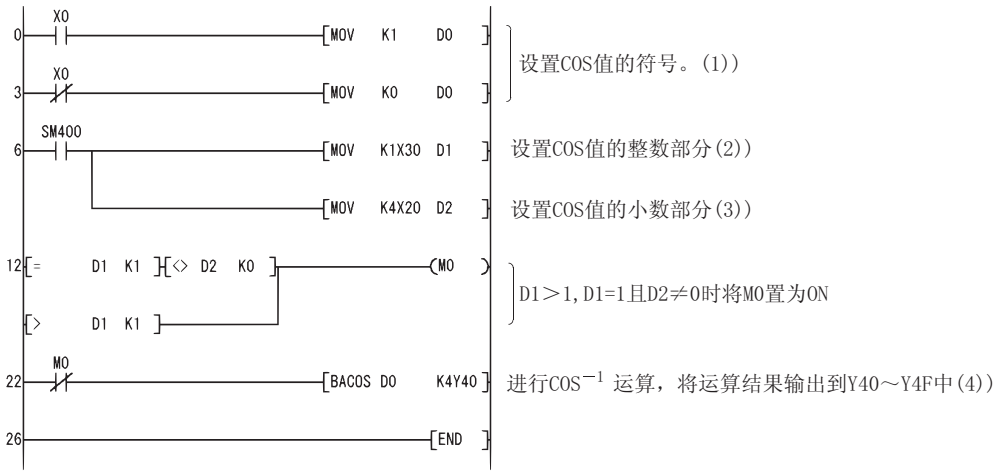
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- Ⓢ中指定的数据不是 BCD 值时。 (出错代码 : 4100)
- Ⓢ中指定的数据超出了 -1.0000 ～ 1.0000 的范围时。 (出错代码 : 4100)
- Ⓢ中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)
  (出错代码 : 4101)

程序示例

(1) 以下为对 X0 中设置了正负符号 (OFF 时为正，ON 时为负)，X30 ～ X33 中以 BCD1 位数设置了整数部分，X20 ～ X2F 中以 BCD4 位数设置了小数部分的值进行 COS^{-1} 运算后，将求出的角度以 BCD4 位数输出到 Y40 ～ Y4F 中的程序。

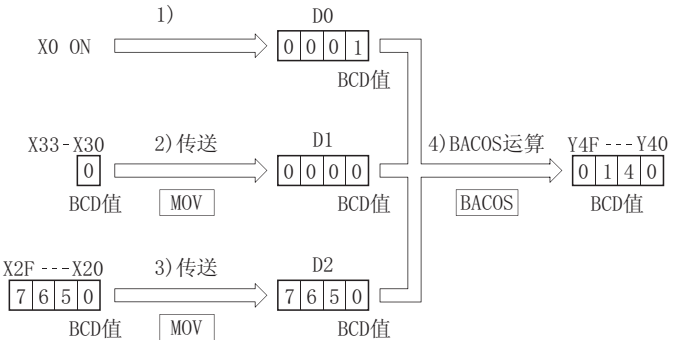
[梯形图模式]



[列表模式]

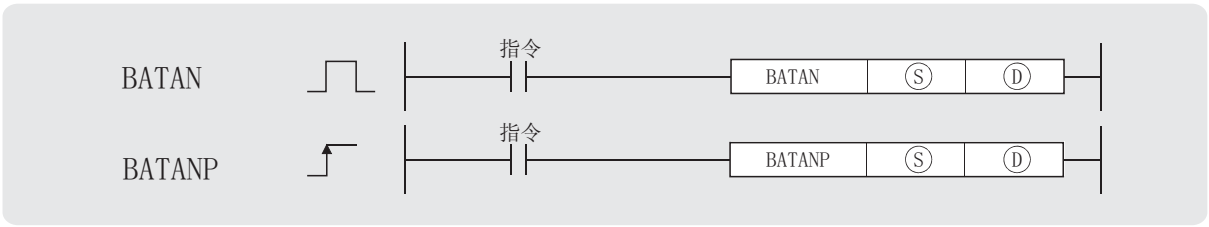
步	指令	软元件	
0	LD	X0	
1	MOV	K1	D0
3	LDI	X0	
4	MOV	K0	D0
6	LD	SM400	
7	MOV	K1X30	D1
10	MOV	K4X20	D2
12	LD=	D1	K1
15	AND<>	D2	K0
18	OR>	D1	K1
21	OUT	M0	
22	LDI	M0	
23	BACOS	D0	K4Y40
26	END		

[在 X0、X20 ～ X33 中指定了 -0.7650 时的动作]



7.12.34 BCD 型 TAN $^{-1}$ 运算 (BATAN(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ : 进行 TAN^{-1}(反正切) 运算的数据或者存储数据的软元件的起始编号 (BCD4 位数)。

Ⓓ : 存储运算结果的软元件的起始编号 (BCD4 位数)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
Ⓢ	--	○		--				--	
Ⓓ	○	○		○				--	

功 能

(1) 对Ⓢ中指定的值进行 TAN^{-1}(反正切) 运算，并将运算结果 (角度) 存储到Ⓓ中指定的软元件中。

TAN^{-1} (Ⓢ 符号 | Ⓢ+1 整数部分 . Ⓢ+2 小数部分) = Ⓓ

(2) 在Ⓢ中设置进行运算的数据的符号。
运算数据为正时存储 “ 0 ”，为负时存储 “ 1 ”。

(3) 在Ⓢ+1、Ⓢ+2 中，分别以 BCD 值存储运算数据的整数部分及小数部分。
(可设置范围为 0 ～ 9999.9999。)

(4) 存储到Ⓓ中的运算结果为 0 ～ 90 °、270 ～ 360 ° (DEG. 单位) 范围内的 BCD 值。

(5) 运算结果为小数部分被四舍五入后的值。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

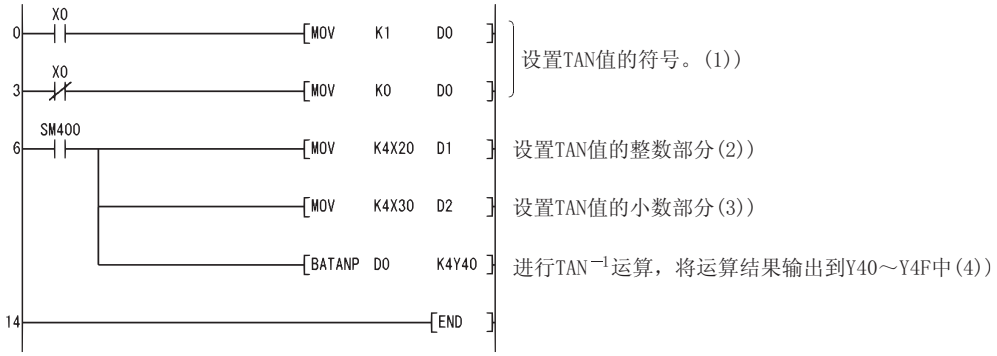
· Ⓢ中指定的数据不是 BCD 值时。 (出错代码 : 4100)

· Ⓢ中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)
(出错代码 : 4101)

程序示例

(1) 以下为对 X0 中设置了正负符号 (OFF 时为正，ON 时为负)，X20 ～ X2F 中以 BCD4 位数设置了整数部分，X30 ～ X3F 中以 BCD4 位数设置了小数部分的值进行 TAN^{-1} 运算后，将求出的角度以 BCD4 位数输出到 Y40 ～ Y4F 中的程序。

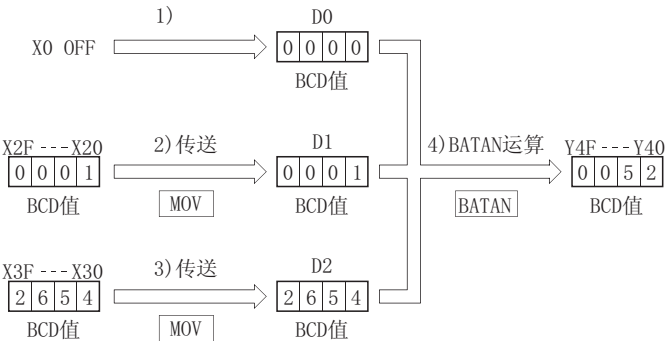
[梯形图模式]



[列表模式]

步	指令	软元件	
0	LD	X0	
1	MOV	K1	D0
3	LDI	X0	
4	MOV	K0	D0
6	LD	SM400	
7	MOV	K4X20	D1
9	MOV	K4X30	D2
11	BATANP	D0	K4Y40
14	END		

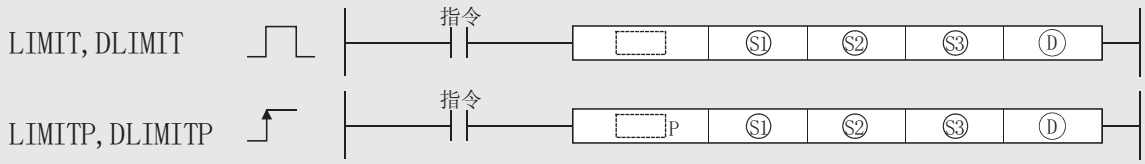
[在 X0、X20 ～ X2F 中指定了 1.2654 时的动作]



7.13 数据控制指令

7.13.1 BIN16 位 /BIN32 位数据的上下限控制 (LIMIT(P)、DLIMIT(P))

Basic | High performance | Process | Redundant | Universal



S1 ：下限值 (最小输出界限值) (BIN16/BIN32 位)。
S2 ：上限值 (最大输出界限值) (BIN16/BIN32 位)。
S3 ：通过上下限控制进行控制的输入值 (BIN16/BIN32 位)。
D ：存储通过上下限控制进行控制的输入值的软元件的起始编号 (BIN16/BIN32 位)。

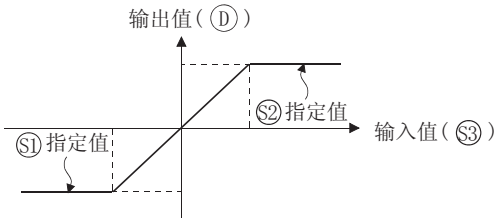
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
S1				○				○	--
S2				○				○	--
S3				○				○	--
D				○				--	--

功 能

LIMIT

(1) 根据S3中指定的输入值 (BIN16 位值) 是否在S1、S2中指定的上下限值的范围内，对存储到D中指定的软元件中的输出值进行控制。

- S1 [下限值] > S3 [输入值] 时 ..... S1 [下限值] → D [输出值]
- S2 [上限值] < S3 [输入值] 时 ..... S2 [上限值] → D [输出值]
- S1 [下限值] ≦ S3 [输入值] ≦ S2 [上限值] 时 ..... S3 [输入值] → D [输出值]



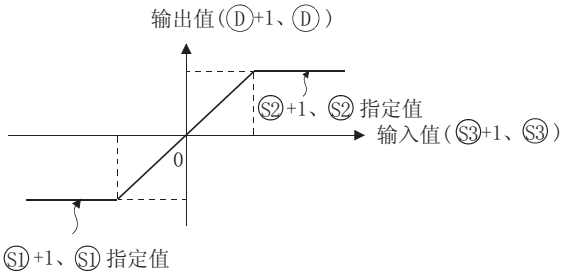
(2) S1、S2、S3 中可指定的值的范围为 -32768 ～ 32767。

(3) 仅根据上限值进行控制时，在Ⓢ1中指定的下限值中设置“ -32768 ”。

(4) 仅根据下限值进行控制时，在Ⓢ2中指定的上限值中设置“ 32767 ”。

DLIMIT

(1) 根据 (Ⓢ3、Ⓢ3+1) 中指定的输入值 (BIN32 位值) 是否在 (Ⓢ1、Ⓢ1+1)、(Ⓢ2、Ⓢ2+1) 中指定的上下限值的范围内，对存储到 (Ⓓ、Ⓓ+1) 中指定的软元件中的输出值进行控制。



(2) (Ⓢ1、Ⓢ1+1)、(Ⓢ2、Ⓢ2+1)、(Ⓢ3、Ⓢ3+1) 中可指定的值的范围为 -2147483648 ～ 2147483647。

(3) 仅根据上限值进行控制时，在 (Ⓢ1、Ⓢ1+1) 中指定的下限值中设置“ -2147483648 ”。

(4) 仅根据下限值进行控制时，在 (Ⓢ2、Ⓢ2+1) 中指定的上限值中设置“ 2147483647 ”。

出 错

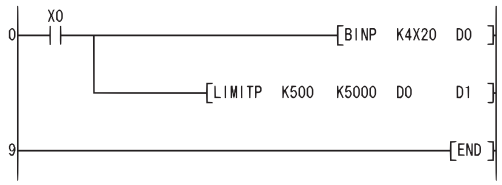
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· Ⓢ1中指定的下限值大于Ⓢ2中指定的上限值时。 (出错代码： 4100)

程序示例

(1) 以下为 X0 变为 ON 时，对 X20 ～ X2F 中以 BCD 值设置的数据进行 500 ～ 5000 的限制控制，并将结果存储到 D1 中的程序。

[梯形图模式]



[列表模式]

步	指令	软元件			
0	LD	X0			
1	BINP	K4X20	D0		
4	LIMITP	K500	K5000	D0	D1
9	END				

[动作]

· D0 ＜ 500 时，D1 变为 500。

例 D0＝400→D1＝500

· 500 ≤ D0 ≤ 5000 时，D1 变为 D0 的值。

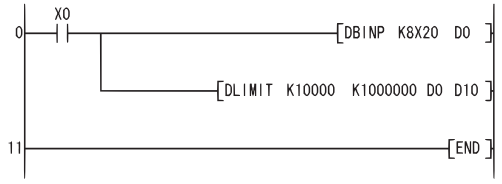
例 D0＝1300→D1＝1300

· 5000 ＜ D0 时，D1 变为 5000。

例 D0＝9600→D1＝5000

(2) 以下为 X0 变为 ON 时，对 X20 ～ X3F 中以 BCD 值设置的数据进行 10000 ～ 1000000 的限制控制，并将结果存储到 D10、D11 中的程序。

[梯形图模式]



[列表模式]

步	指令	软元件			
0	LD	X0			
1	DBINP	K8X20	D0		
4	DLIMIT	K10000	K1000000	D0	D10
11	END				

[动作]

· (D1、D0) ＜ 10000 时，(D11、D10) 变为 10000。

例 (D1、D0)＝400→((D11、D10)＝10000

· 10000 ≤ (D1、D0) ≤ 1000000 时，(D11、D10) 变为 (D1、D0) 的值。

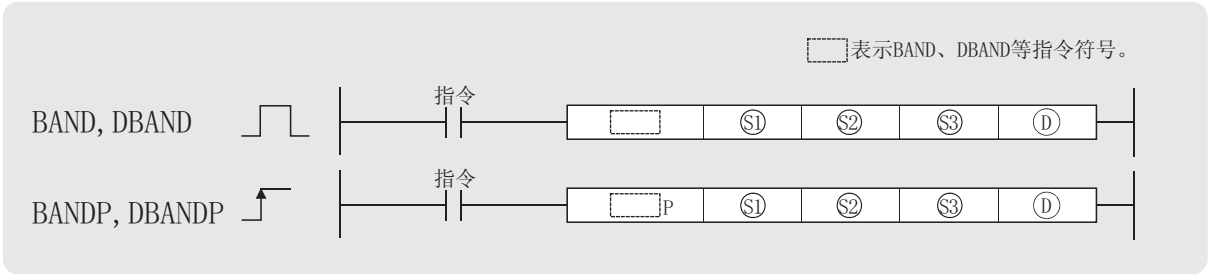
例 (D1、D0)＝345678→(D11、D10)＝345678

· 1000000 ＜ (D1、D0) 时，(D11、D10) 变为 1000000。

例 (D1、D0)＝9876543→(D11、D10)＝1000000

7.13.2 BIN16 位 /32 位死区控制 (BAND(P)、DBAND(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ1 ：死区 (无输出区域) 的下限值 (BIN16/BIN32 位)。
Ⓢ2 ：死区 (无输出区域) 的上限值 (BIN16/BIN32 位)。
Ⓢ3 ：通过死区控制进行控制的输入值 (BIN16/BIN32 位)。
Ⓓ ：存储通过死区控制进行控制的输入值的软元件的起始编号 (BIN16/BIN32 位)。

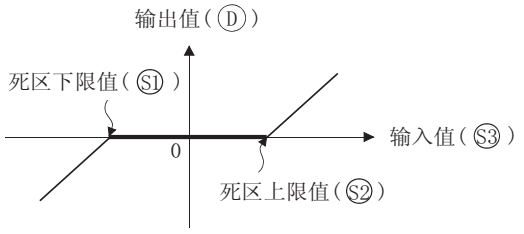
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ1		○						○	--
Ⓢ2		○						○	--
Ⓢ3		○						○	--
Ⓓ		○						--	--

功 能

BAND

(1) 根据Ⓢ3 中指定的输入值 (BIN16 位值) 是否在Ⓢ1、Ⓢ2 循中指定的死区的上下限范围内，对存储到Ⓓ中指定的软元件中的输出值进行控制。
输出值的控制如下所示。

- Ⓢ1 [下限值] > Ⓢ3 [输入值] 时..... Ⓢ3 [输入值] - Ⓢ1 [下限值] → Ⓓ [输出值]
- Ⓢ2 [上限值] < Ⓢ3 [输入值] 时..... Ⓢ3 [输入值] - Ⓢ2 [上限值] → Ⓓ [输出值]
- Ⓢ1 [下限值] ≦ Ⓢ3 [输入值] ≦ Ⓢ2 [上限值] 时.....0 → Ⓓ [输出值]



(2) Ⓢ1、Ⓢ2、Ⓢ3 中可指定的值的范围为 -32768 ～ 32767。

(3) Ⓓ中存储的输出值为带符号的 16 位 BIN 值。因此运算结果超出了 -32768 ～ 32767 的范围时的情况如下所示。

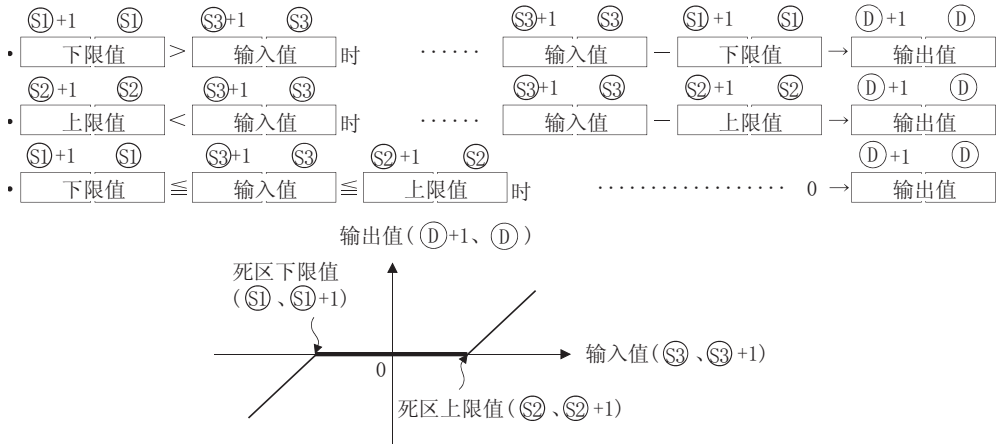
死区下限值 (Ⓢ1) 10
输入值 (Ⓢ3) ..... －32768
} 时

输出值＝－32768－10＝8000H－AH＝7FF6H＝32758。

DBAND

(1) 根据 (Ⓢ3、Ⓢ3+1) 中指定的输入值 (BIN32 位值) 是否在 (Ⓢ1、Ⓢ1+1)、(Ⓢ2、Ⓢ2+1) 中指定的死区上下限值的范围内，对存储到Ⓓ中指定的软元件中的输出值进行控制。

输出值的控制如下所示。



(2) (Ⓢ1、Ⓢ1+1)、(Ⓢ2、Ⓢ2+1)、(Ⓢ3、Ⓢ3+1) 中可指定的值的范围为 -2147483648 ～ 2147483647。

(3) Ⓓ、Ⓓ+1 中存储的输出值为带符号的 32 位 BIN 值。因此运算结果超出了 -2147483648 ～ 2147483647 的范围时的情况如下所示。

死区下限值 (Ⓢ1、Ⓢ1+1) 1000
输入值 (Ⓢ3、Ⓢ3+1) ..... －2147483648
} 时

输出值＝－2147483648－1000＝80000000H－000003E8H
＝7FFFFC18H＝2147482648。

出　错

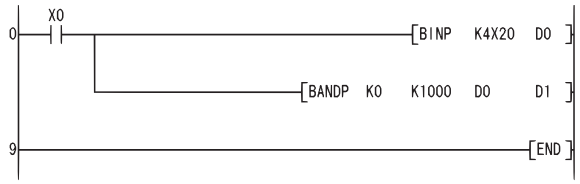
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· Ⓢ1中指定的下限值大于Ⓢ2中指定的上限值时。 (出错代码： 4100)

程序示例

(1) 以下为 X0 变为 ON 时，对 X20 ～ X2F 中以 BCD 值设置的数据进行 0 ～ 1000 的死区控制，并将结果存储到 D1 中的程序。

[梯形图模式]



[列表模式]

步	指令	软元件			
0	LD	X0			
1	BINP	K4X20	D0		
4	BANDP	K0	K1000	D0	D1
9	END				

[动作]

· 0≦D0≦1000 时，D1 中存储 0。

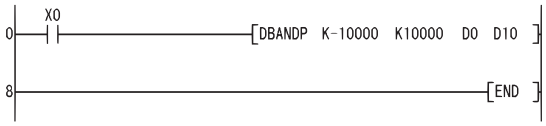
例 D0＝500→D1＝0

· 1000＜D0 时，D1 中存储 (D0)-1000 的值。

例 D0＝7000→D1＝6000

(2) 以下为 X0 变为 ON 时，对 D0、D1 设置的数据进行 -10000 ～ 10000 的死区控制，并将结果存储到 D10、D11 中的程序。

[梯形图模式]



[列表模式]

步	指令	软元件			
0	LD	X0			
1	DBANDP	K-10000	K10000	D0	D10
8	END				

[动作]

· (D1、D0) ＜ (-10000) 时，(D11、D10) 中存储 (D1、D0)-(-10000) 的值。

例 (D1、D0)＝－12345→(D11、D10)＝－2345

· －10000≦(D1、D0)≦10000 时，(D11、D10) 中存储 0。

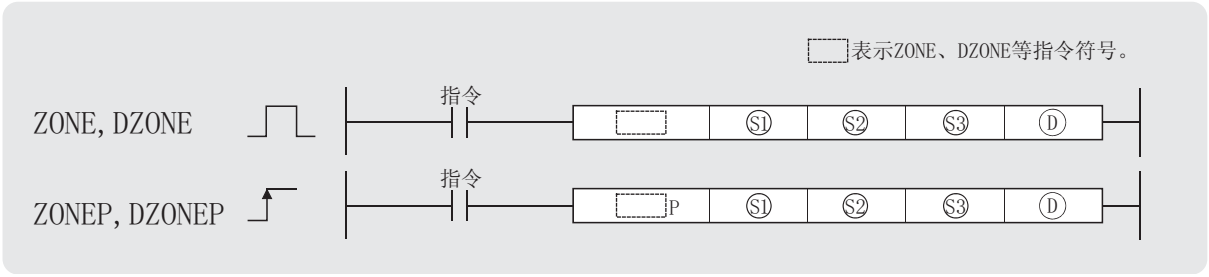
例 (D1、D0)＝6789→(D11、D10)＝0

· 10000＜(D1、D0) 时，(D11、D10) 中存储 (D1、D0)-10000 的值。

例 (D1、D0)＝50000→(D11、D10)＝40000

7.13.3 BIN16 位 /BIN32 位数据的区域控制 (ZONE(P)、DZONE(P))

Basic | High performance | Process | Redundant | Universal



(S1) ：对输入值进行加法运算的负偏置值 (BIN16/BIN32 位)。
(S2) ：对输入值进行加法运算的正偏置值 (BIN16/BIN32 位)。
(S3) ：用于进行区域控制的输入值 (BIN16/BIN32 位)。
(D) ：存储通过区域控制进行控制的输入值的软元件的起始编号 (BIN16/BIN32 位)。

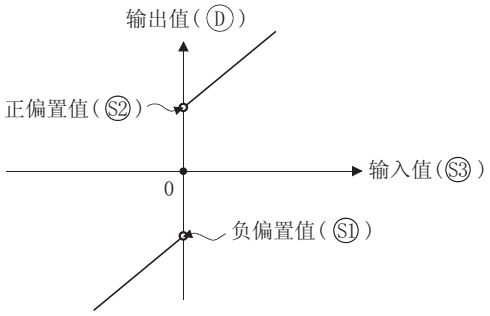
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
(S1)		○						○	--
(S2)		○						○	--
(S3)		○						○	--
(D)		○						--	--

功 能

ZONE

(1) 在(S3)中指定的输入值上附加(S1)或者(S2)中指定的偏置值后，存储到(D)中指定编号的软元件中。偏置值的附加情况如下所示。

- (S3) 输入值 < 0 时 ..... (S3) 输入值 + (S1) 负偏置值 → (D) 输出值
- (S3) 输入值 = 0 时 0 → (D) 输出值
- (S3) 输入值 > 0 时 ..... (S3) 输入值 + (S2) 正偏置值 → (D) 输出值



(2) (S1)、(S2)、(S3)中可指定的值的范围为 -32768 ～ 32767。

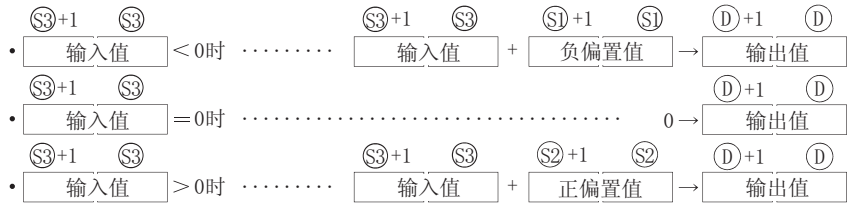
(3) Ⓓ中存储的输出值为带符号的 16 位 BIN 值。因此运算结果超出了 -32768 ～ 32767 的范围时的情况如下所示。

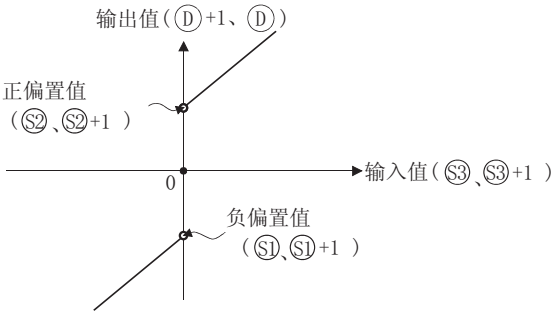
负偏置值 (Ⓢ1) ..... －100
输入值 (Ⓢ3) ..... －32768
} 时

输出值＝－32768+(－100)＝8000H+FF9C＝7F9CH＝32668。

DZONE

(1) 在 (Ⓢ3、Ⓢ3+1) 中指定的输入值中，附加了 (Ⓢ1、Ⓢ1+1) 或者 (Ⓢ2、Ⓢ2+1) 中指定的偏置值后，存储到 (Ⓓ、Ⓓ+1) 中指定编号的软元件中。
偏置值的附加情况如下所示。





(2) (Ⓢ1、Ⓢ1+1)、(Ⓢ2、Ⓢ2+1)、(Ⓢ3、Ⓢ3+1) 中可指定的值的范围为 -2147483648 ～ 2147483647。

(3) (Ⓓ、Ⓓ+1) 中存储的值为带符号的 32 位 BIN 值。
因此运算结果超出了 -2147483648 ～ 2147483647 的范围时的情况如下所示。

负偏置值 (Ⓢ1、Ⓢ1+1) ..... －1000
输入值 (Ⓢ3、Ⓢ3+1) ..... －2147483648
} 时

输出值＝－2147483648+(－1000)＝80000000H+FFFFFC18H
＝7FFFFC18＝2147482648。

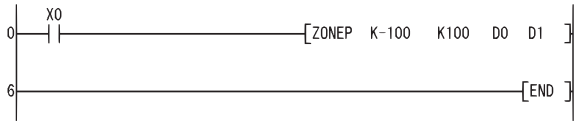
出　错

(1) ZONE(P)、DZONE(P) 指令中无运算出错。

程序示例

(1) 以下为 X0 变为 ON 时，对 D0 中设置的数据进行 -100 ～ 100 的区域控制，并将结果存储到 D1 中的程序。

[梯形图模式]



[列表模式]

步	指令	软元件			
0	LD	X0			
1	ZONEP	K-100	K100	D0	D1
6	END				

[动作]

· D0＜0 时，在 D1 中存储 (D0)+(-100) 的值。

例 D0＝－200→D1＝－300

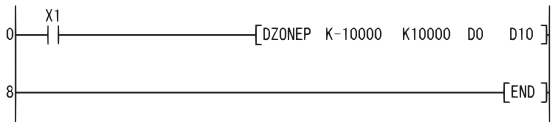
· D0＝0 时，D1 中存储 0。

· 0＜D0 时，D1 中存储 (D0)+100 的值。

例 D0＝700→D1＝800

(2) 以下为 X0 变为 ON 时，对 D0、D1 设置的数据进行 -10000 ～ 10000 的区域控制，并将结果存储到 D10、D11 中的程序。

[梯形图模式]



[列表模式]

步	指令	软元件			
0	LD	X1			
1	DZONEP	K-10000	K10000	D0	D10
8	END				

[动作]

· (D1、D0)＜0 时，(D11、D10) 中存储 (D1、D0)+(-10000) 的值。

例 (D1、D0)＝－12345→(D11、D10)＝－22345

· (D1、D0)＝0 时，(D11、D10) 中存储 0。

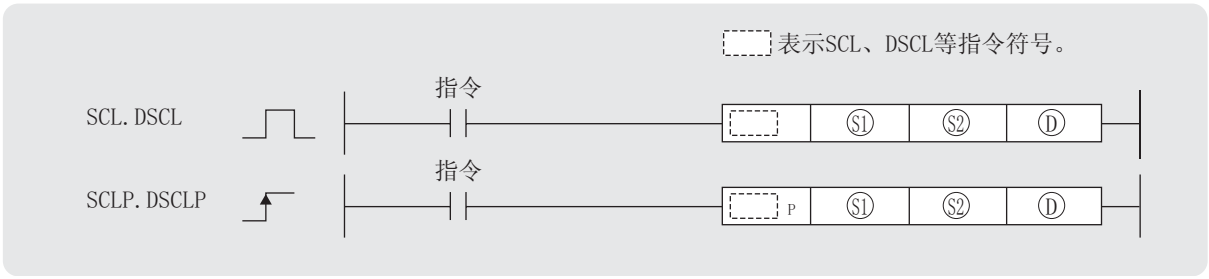
· 0＜(D1、D0) 时，(D11、D10) 中存储 (D11、D10)+10000 的值。

例 (D1、D0)＝50000→(D11、D10)＝60000

7.13.4 标度（点坐标数据）(SCL(P)、DSCL(P))

Basic ✗ High performance ✗ Process ✗ Redundant ✗ Ver. Universal

· QnU(D)(H)CPU：序列号的前 5 位数为“ 10102 ”以后
· QnUDE(H)CPU：序列号的前 5 位数为“ 10102 ”以后



(S1) ：进行标度的输入值或者存储输入值的软元件的起始编号 (BIN16/32 位)。
(S2) ：存储标度用转换数据的软元件的起始编号 (BIN16/32 位)。
(D) ：存储通过标度进行控制的输出值的软元件的起始编号 (BIN16/BIN32 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
(S1)	--	○	○	○				○	--
(S2)	--	○	○	--				--	--
(D)	--	○	○	○				--	--

功 能

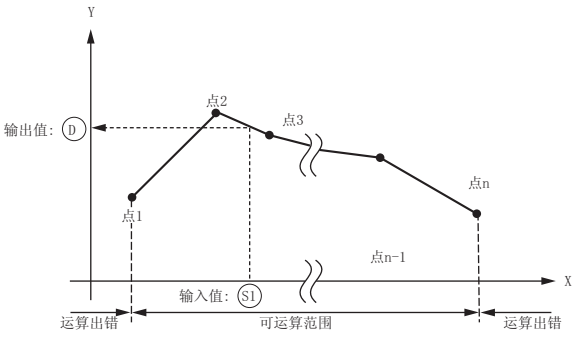
SCL(P)

(1) 对(S2)中指定的标度用转换数据 (16 位数据单位)，通过(S1)中指定的输入值进行标度，并将运算结果存储到(D)中指定编号的软元件中。
标度转换是基于(S2)中指定的软元件后面存储的标度用转换数据进行的。

标度用转换数据的构成

设置项目		软元件分配
坐标点数		(S2)
点1	X坐标	(S2)+1
	Y坐标	(S2)+2
点2	X坐标	(S2)+3
	Y坐标	(S2)+4
点n	X坐标	(S2)+2n-1
	Y坐标	(S2)+2n

※n表示(S2)中指定的坐标点数



(2) 运算结果不是整数值时，对小数点以下第 1 位进行四舍五入。

(3) 标度用转换数据的 X 坐标数据应按升序进行设置。

(4) ⓢ1 应在标度用转换数据范围内 (ⓢ2 的软元件值) 进行设置。

(5) 将多个点指定为相同的 X 坐标时，将对最大点号的点的 Y 坐标值进行输出。

(6) 标度用转换数据的坐标点数范围为 1 ～ 32767。

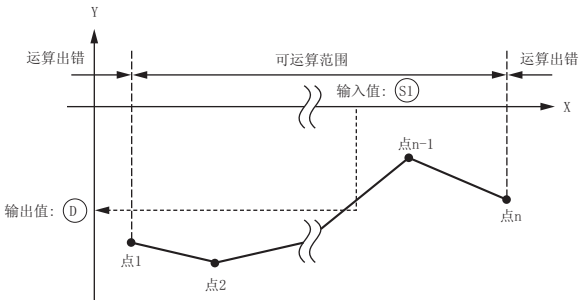
DSCL(P)

(1) 对ⓢ2 中指定的标度用转换数据 (32 位数据单位)，通过ⓢ1 中指定的输入值进行标度，并将运算结果存储到ⓓ中指定编号的软元件中。
标度转换是基于ⓢ2 中指定的软元件后面存储的标度用转换数据进行的。

标度用转换数据的构成

设置项目		软元件分配
坐标点数		(S2)+1，(S2)
点1	X坐标	(S2)+3，(S2)+2
	Y坐标	(S2)+5，(S2)+4
点2	X坐标	(S2)+7，(S2)+6
	Y坐标	(S2)+9，(S2)+8
⋮		
点n	X坐标	(S2)+4n-1，(S2)+4n-2
	Y坐标	(S2)+4n+1，(S2)+4n

※n表示(S2)中指定的坐标点数。



(2) 运算结果不是整数值时，对小数点以下第 1 位进行四舍五入。

(3) 标度用转换数据的 X 坐标数据应按升序进行设置。

(4) ⓢ1 应在标度用转换数据范围内 (ⓢ2、ⓢ2+1 的软元件值) 进行设置。

(5) 将多个点指定为相同的 X 坐标时，将对最大点号的点的 Y 坐标值进行输出。

(6) 标度用转换数据的坐标点数范围为 1 ～ 32767。

要 点

(1) 根据 SM750 的 ON/OFF 状态其探索方法有所不同。

SM750	探索方法	探索次数范围
OFF	逐次探索	1 ≤次数≤ 32767
ON	二分探索	1 ≤次数≤ 15

(2) 标度用转换数据按升序排序时，根据 SM750 的状态其探索方法有所不同，因此处理速度也不相同。

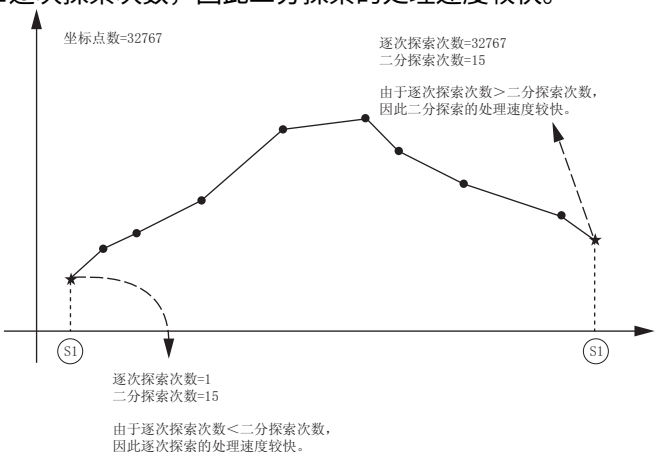
处理速度取决于探索次数，探索次数越少其处理速度越快。

(a) 逐次探索的处理速度较快的情况

Ⓢ1 的最大坐标点数为 1 ～ 15 之间时，由于逐次探索次数≤ 15，因此逐次探索的处理速度将较快。

(b) 二分探索的处理速度较快的情况

由于最大探索次数为 15 次，在Ⓢ1 的坐标点为 16 点以上时，二分探索次数≤逐次探索次数，因此二分探索的处理速度较快。



出　错

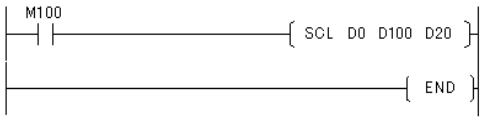
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- 位于标度用转换数据的(S1)前面的点的 X 坐标数据未按升序设置时。(但是，SM750 为 ON 时，不进行此出错检查。)　(出错代码： 4100)
- (S1)中指定的输入值超出了所设置的标度用转换数据的范围时。　(出错代码： 4100)
- 从(S2)的软元件开始的坐标点数超出了 1 ～ 32767 的范围时。　(出错代码： 4100)
- 从(S2)的软元件开始的坐标点数超出了指定软元件的范围时。　(出错代码： 4101)

程序示例

(1) 以下为 M100 变为 ON 时，通过 D0 中指定的输入值对 D100 后面设置的标度用转换数据进行标度后，输出到 D20 中的程序。

[梯形图模式]



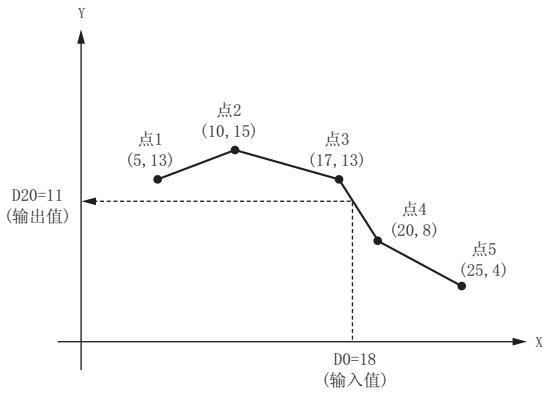
[列表模式]

步	指令	软元件
0	LD	M100
1	SCL	D0 D100 D20
5	END	

[动作]

标度用转换数据的构成

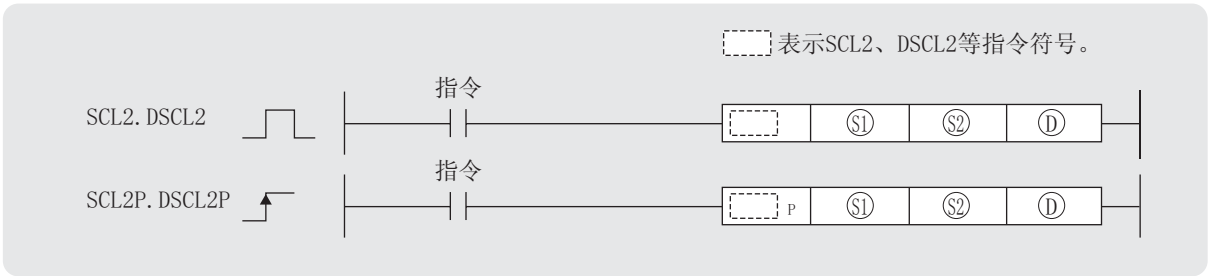
设置项目		软元件	设置内容
坐标点数		D100	K5
点1	X坐标	D101	K5
	Y坐标	D102	K13
点2	X坐标	D103	K10
	Y坐标	D104	K15
点3	X坐标	D105	K17
	Y坐标	D106	K13
点4	X坐标	D107	K20
	Y坐标	D108	K8
点5	X坐标	D109	K25
	Y坐标	D110	K22



7.13.5 标度 (X/Y 坐标数据)(SCL2(P)、DSCL2(P))

Basic High performance Process Redundant Ver. Universal

· QnU(D)(H)CPU：序列号的前 5 位数为 “ 10102 ” 以后
· QnUDE(H)CPU：序列号的前 5 位数为 “ 10102 ” 以后



(S1) ：进行标度的输入值或者存储输入值的软元件的起始编号 (BIN16/32 位)。
(S2) ：存储标度用转换数据的软元件的起始编号 (BIN16/32 位)。
(D) ：存储通过标度进行控制的输出值的软元件的起始编号 (BIN16/BIN32 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
(S1)	--	○	○		○			○	--
(S2)	--	○	○		--			--	--
(D)	--	○	○		○			--	--

功 能

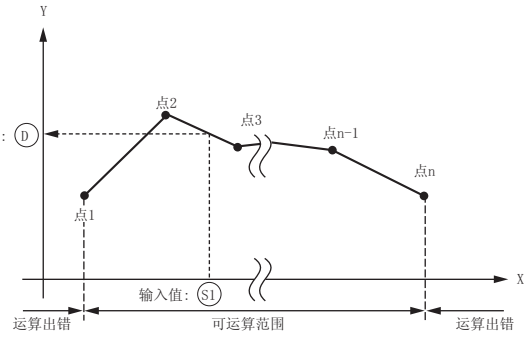
SCL2(P)

(1) 对(S2)中指定的标度用转换数据 (16 位数据单位)，通过(S1)中指定的输入值进行标度，并将运算结果存储到(D)中指定编号的软元件中。
标度转换是基于(S2)中指定的软元件后面存储的标度用转换数据进行的。

标度用转换数据的构成

设置项目		软元件分配
坐标点数		(S2)
X坐标	点1	(S2)+1
	点2	(S2)+2
	⋮	⋮
	点n	(S2)+n
Y坐标	点1	(S2)+n+1
	点2	(S2)+n+2
	⋮	⋮
	点n	(S2)+2n

※n表示(S2)中指定的坐标点数。



(2) 运算结果不是整数值时，对小数点以下第 1 位进行四舍五入。

(3) 标度用转换数据的 X 坐标数据应按升序进行设置。

(4) ⓢ1 应在标度用转换数据范围内 (ⓢ2 的软元件值) 进行设置。

(5) 将多个点指定为相同的 X 坐标时，将对最大点号的点的 Y 坐标值进行输出。

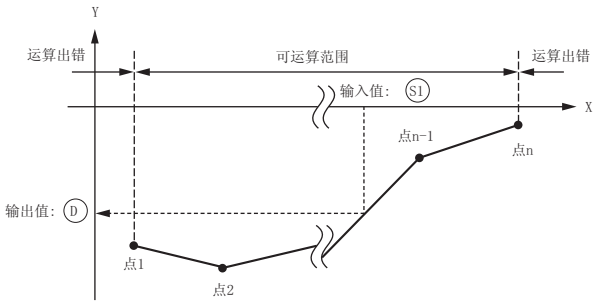
DSCL2(P)

(1) 对ⓢ2 中指定的标度用转换数据 (32 位数据单位)，通过ⓢ1 中指定的输入值进行标度，并将运算结果存储到ⓓ中指定编号的软元件中。
标度转换是基于ⓢ2 中指定的软元件后面存储的标度用转换数据进行的。

标度用转换数据的构成

设置项目		软元件分配
坐标点数		(S2)+1 , (S2)
X坐标	点1	(S2)+3 , (S2)+2
	点2	(S2)+5 , (S2)+4
	⋮	⋮
	点n	(S2)+2n+1 , (S2)+2n
Y坐标	点1	(S2)+2n+3 , (S2)+2n+2
	点2	(S2)+2n+5 , (S2)+2n+4
	⋮	⋮
	点n	(S2)+4n+1 , (S2)+4n

※n表示(S2)中指定的坐标点数。



(2) 运算结果不是整数值时，对小数点以下第 1 位进行四舍五入。

(3) 标度用转换数据的 X 坐标数据应按升序进行设置。
但是，ⓢ1 后面的点即使未按升序进行设置，也不会变为运算出错状态，将继续执行指令。

(4) ⓢ1 应在标度用转换数据范围内 (ⓢ2、ⓢ2+1 的软元件值) 进行设置。

(5) 将多个点指定为相同的 X 坐标时，将对最大点号的点的 Y 坐标值进行输出。

(6) 标度用转换数据的坐标点数范围为 1 ～ 32767。

要 点

标度用转换数据按升序排序时，根据 SM750 的状态其探索方法有所不同，因此处理速度也不相同。
详细内容请参阅 7.13.4 项。

出 错

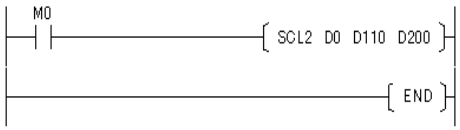
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- X 坐标数据未按升序设置时。 (出错代码： 4100)
- ⑤1 中指定的输入值超出了所设置的标度用转换数据的范围时。 (出错代码： 4100)
- ⑤2 中指定的坐标点数超出了 1 ～ 32767 的范围时。 (出错代码： 4100)
- 从⑤2 的软元件开始的坐标点数超出了指定软元件的范围时。 (出错代码： 4101)

程序示例

(1) 以下为 M0 变为 ON 时，将 D0 中指定的输入值通过 D110 后面设置的标度用转换数据进行标度后，输出到 D200 中的程序。

[梯形图模式]



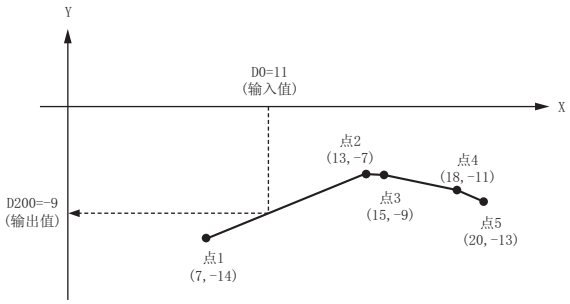
[列表模式]

步	指令	软元件
0	LD	M0
1	SCL2	D0 D110 D200
5	END	

[动作]

标度用转换数据的构成

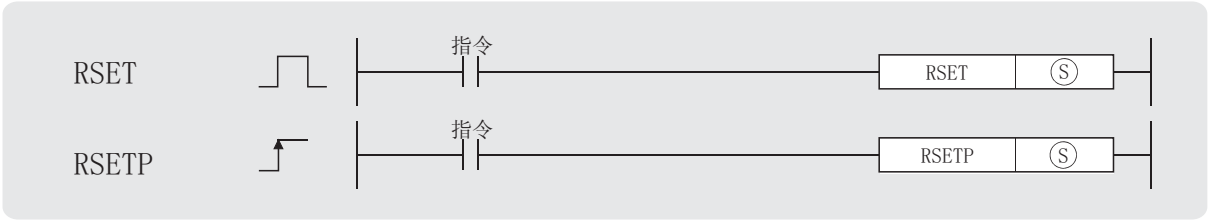
设置项目		软元件	设置内容
坐标点数		D110	K5
X坐标	点1	D111	K7
	点2	D112	K13
	点3	D113	K15
	点4	D114	K18
	点5	D115	K20
Y坐标	点1	D116	K-14
	点2	D117	K-7
	点3	D118	K-15
	点4	D119	K-11
	点5	D120	K-18



7.14 文件寄存器切换指令

7.14.1 文件寄存器的块号切换 (RSET(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ : 进行切换的块号数据或者存储块号数据的软元件的起始编号 (BIN16 位)。

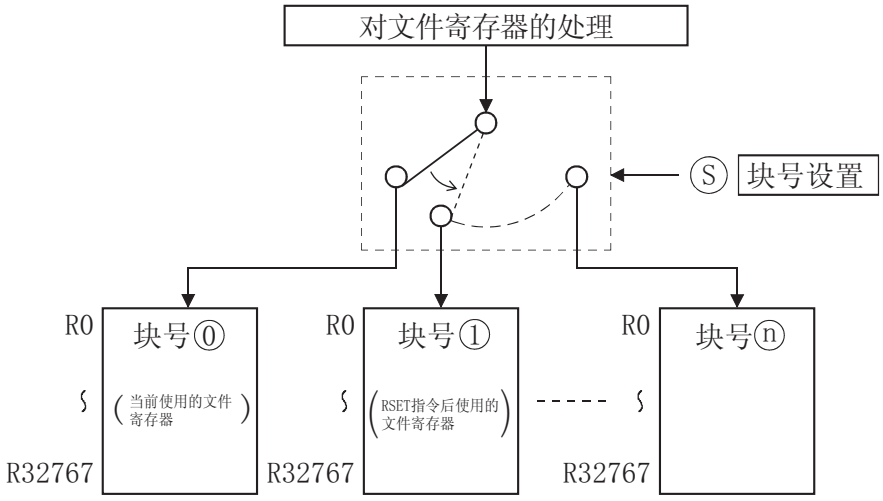
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○								--

功 能

将程序中使用的文件寄存器的块号变更为Ⓢ中指定的软元件中存储的块号。

进行块号变更后，顺控程序中使用的所有文件寄存器均按变更后的块号的文件寄存器进行处理。

例 从块号 0 切换为块号 1 时



要 点

将文件寄存器 (R) 指定为刷新软元件，并通过 RSET 指令切换文件寄存器 (R) 的块号时，应加以注意。

关于文件寄存器的限制事项，请参阅 3.10 节。

出　错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

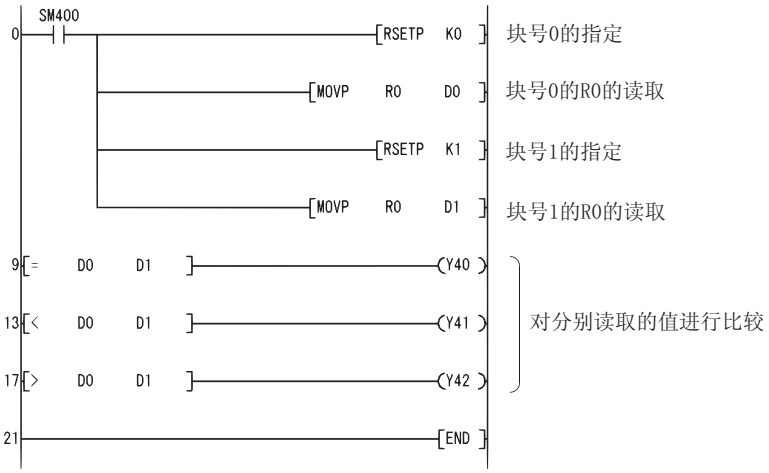
· ⓢ中指定的块号不存在时。　　(出错代码： 4100)

· 文件寄存器不存在时。　　(出错代码： 4101)

程序示例

(1) 以下为将块号 0 与块号 1 的 R0 进行比较的程序。

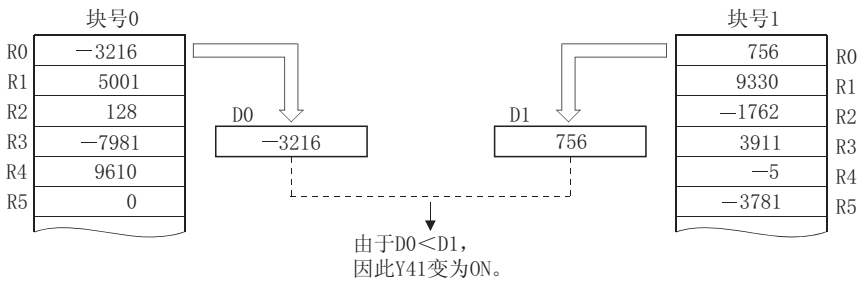
[梯形图模式]



[列表模式]

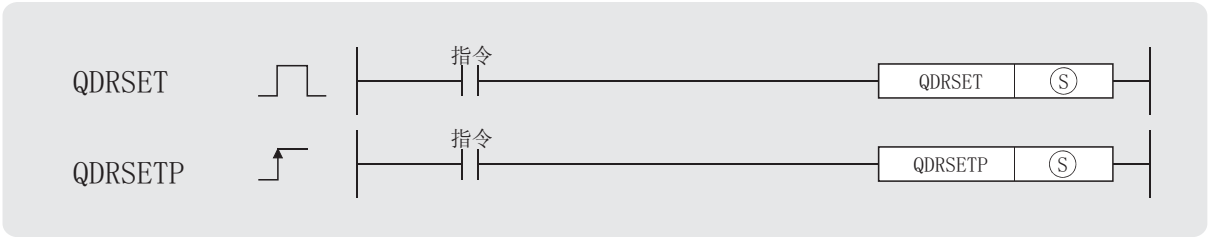
步	指令	软元件	
0	LD	SM400	
1	RSETP	K0	
3	MOVP	R0	D0
5	RSETP	K1	
7	MOVP	R0	D1
9	LD=	D0	D1
12	OUT	Y40	
13	LD<	D0	D1
16	OUT	Y41	
17	LD>	D0	D1
20	OUT	Y42	
21	END		

[动作]



7.14.2 文件寄存器用文件的设置 (QDRSET(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ : 进行设置的文件寄存器的驱动器号及文件名的字符串数据或者存储字符串数据的软元件的起始编号 (字符串)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
Ⓢ	--	○		--				○	--

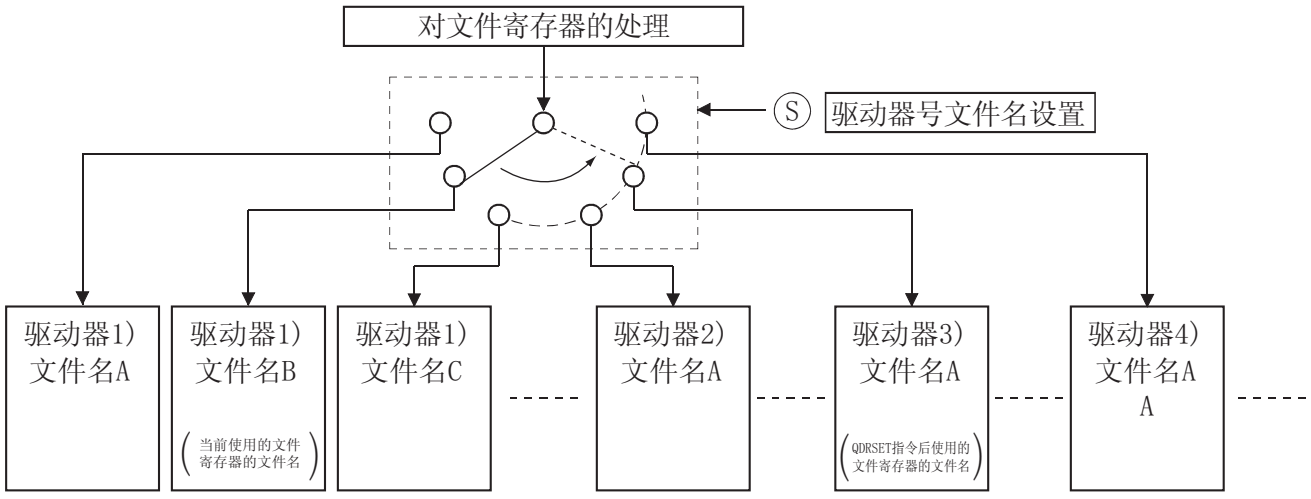
功 能

(1) 将程序中使用的文件寄存器的文件名变更为Ⓢ中指定的软元件中存储的文件名。

进行了文件名变更后，顺控程序中使用的所有文件寄存器均按变更后文件名的块号 0 的文件寄存器进行处理。

块号的切换是通过 RSET 指令进行。

例 从驱动器号 1 的文件名 B 切换为驱动器号 3 的文件名 A 时



(2) 驱动器号可在 1 ～ 4 的范围内指定。
(驱动器号不能指定为驱动器 0(程序存储器 / 内置存储器)。)
但是，根据 CPU 模块的不同可指定的驱动器也有所不同。
请通过所使用的 CPU 模块的手册对可指定的驱动器进行确认。

(3) 文件名中无需指定扩展名 (.QDR)。

(4) 通过在文件名中指定 NULL 字符 (00H)，可以对文件名的设置进行解除。

(5) 即使在参数中对驱动器号、文件名进行了指定，本指令中指定的文件名也将优先。

要 点

1. 即使通过 QDRSET 指令对文件名进行了变更，如果进行了 CPU 模块的 STOP → RUN 操作，将恢复为参数中设置的文件名。
如果希望在进行了 CPU 模块的 STOP → RUN 操作时，仍保持为通过 QDRSET 指令变更后的文件名，应使用 STOP → RUN 时 1 个扫描 ON 的特殊继电器后，执行 QDRSET 指令。
2. 将文件寄存器指定为刷新软元件时，不要通过 QDRSET 指令对文件寄存器的文件名进行变更。
关于文件寄存器的限制事项，请参阅 3.10 节。

出 错

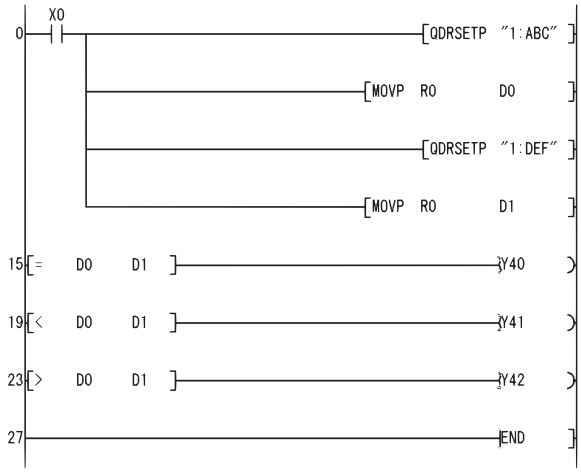
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· ⓢ中指定的驱动器的文件名不存在时。 (出错代码： 2410)

程序示例

(1) 以下为将驱动器号 1 的 ABC 与驱动器号 1 的 DEF 的 R0 进行比较的程序。

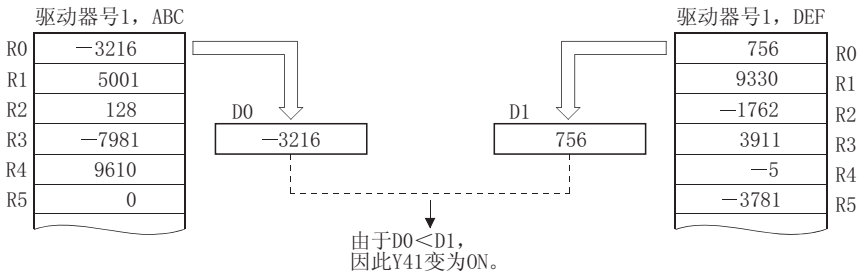
[梯形图模式]



[列表模式]

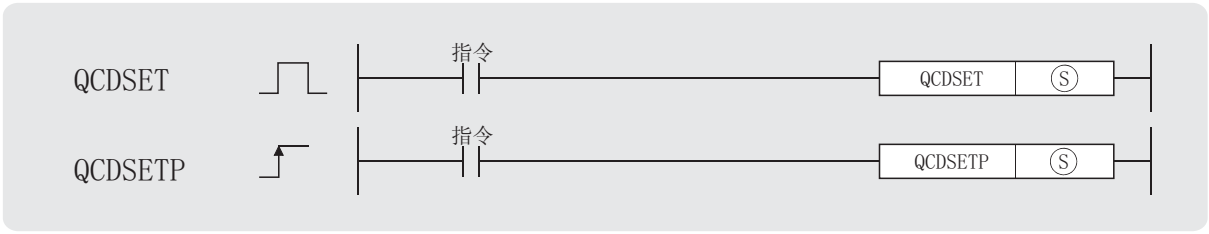
步	指令	软元件	
0	LD	X0	
1	QDRSETP	"1:ABC"	
6	MOVP	R0	D0
8	QDRSETP	"1:DEF"	
13	MOVP	R0	D1
15	LD=	D0	D1
18	OUT	Y40	
19	LD<	D0	D1
22	OUT	Y41	
23	LD>	D0	D1
26	OUT	Y42	
27	END		

[动作]



7.14.3 注释用文件的设置 (QCDSET(P))

Basic High performance Process Redundant Universal



Ⓢ ：进行设置的注释文件的驱动器号及文件名的字符串数据或者存储字符串数据的软元件的起始编号（字符串）。

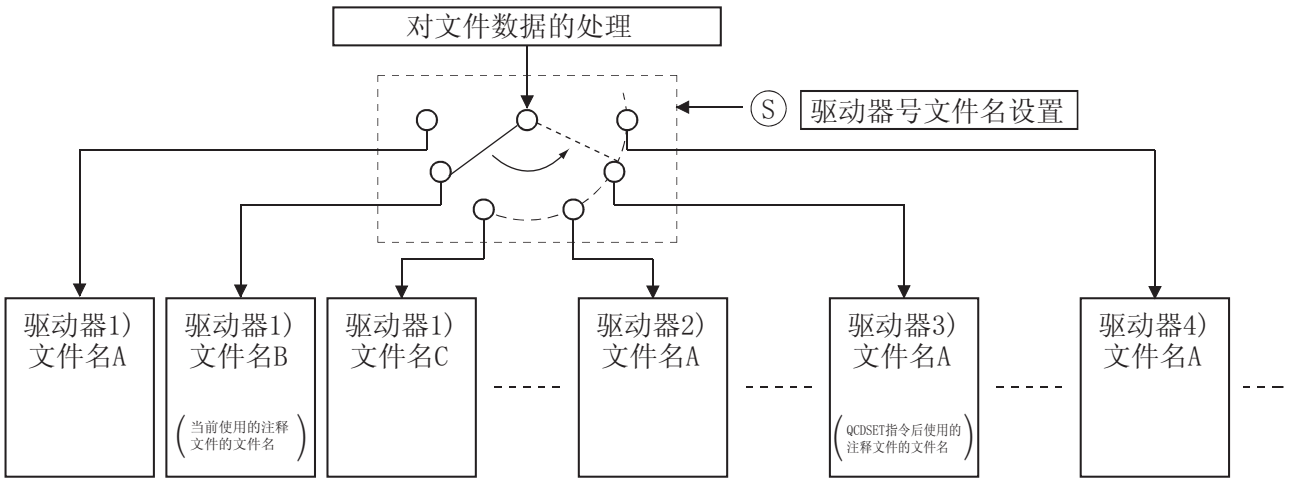
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			$	
Ⓢ	--	○		--				○	--

功 能

(1) 将程序中使用的注释文件的文件名变更为Ⓢ中指定的软元件中存储的文件名。

进行了文件名变更后，顺控程序中使用的所有注释数据均按变更后文件名的注释数据进行处理。

例 从驱动器号 1 的文件 B 切换为驱动器号 3 的文件 A 时



(2) 驱动器号可在 1 ～ 4 的范围内指定。
(驱动器号不能指定为驱动器 0(程序存储器 / 内置存储器)。)
但是，根据 CPU 模块的不同可指定的驱动器也有所不同。
请通过所使用的 CPU 模块的手册对可指定的驱动器进行确认。

(3) 文件名中无需指定扩展名 (.QCD)。

(4) 通过在文件名中指定 NULL 字符 (00_H)，可以对文件名的设置进行解除。

(5) 即使在参数中对驱动器号、文件名进行了指定，本指令中指定的文件名也将优先。

(6) 在通用型 QCPU 中，SM721 为 ON 状态时不能执行本指令。如果执行也将无处理。

要 点

即使通过 QCDSET 指令对文件名进行了变更，如果进行了 CPU 模块的 STOP → RUN 操作，将恢复为参数中设置的文件名。
如果希望在进行了 CPU 模块的 STOP → RUN 操作时，仍保持为通过 QCDSET 指令变更后的文件名，应使用 STOP → RUN 时 1 个扫描 ON 的特殊继电器 SM402 后，执行 QCDSET 指令。

出 错

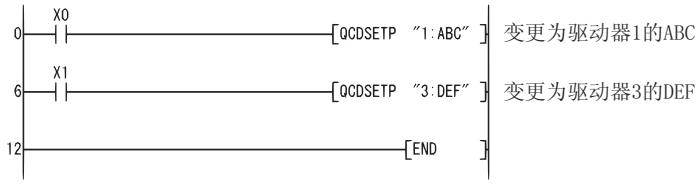
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· Ⓢ中指定的驱动器号的文件名不存在时。 (出错代码 : 2410)

程序示例

(1) 以下为 X0 变为 ON 时，将注释对象文件切换为驱动器 1 的 ABC.QCD，X1 变为 ON 时，将注释对象文件切换为驱动器 3 的 DEF.QCD 的程序。

[梯形图模式]



[列表模式]

步	指令	软元件
0	LD	X0
1	QCDSETP	"1:ABC"
6	LD	X1
7	QCDSETP	"3:DEF"
12	END	

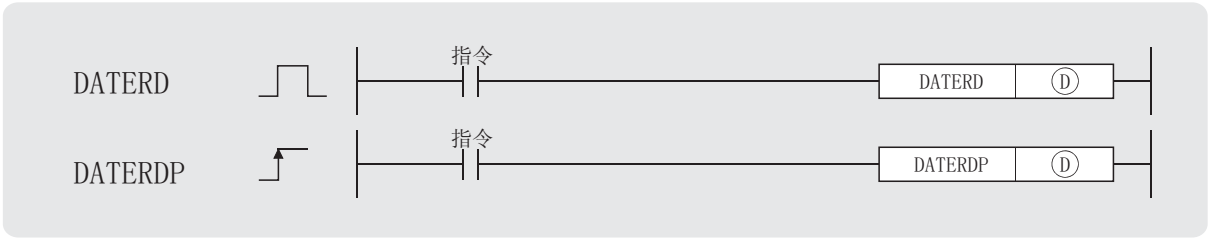
注意事项

(1) 在通用型 QCPU 中，SM721(文件访问中) 为 ON 状态时，即使本指令的执行指令变为 ON，也不能执行本指令。应将程序设置为在 SM721 处于 OFF 状态时执行本指令。

7.15 时钟指令

7.15.1 时钟数据的读取 (DATERD(P))

Basic | High performance | Process | Redundant | Universal

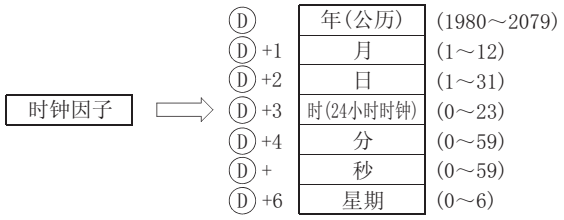


Ⓓ : 存储读取的时钟数据的软元件的起始编号 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
Ⓓ	--	○		--					

功 能

(1) 根据 CPU 模块的时钟因子读取 “ 年、月、日、时、分、秒、星期 ” 后，以 BIN 值存储到Ⓓ中指定的软元件的后面。



(2) Ⓓ的 “ 年 ” 以 公历 4 位数存储。

(3) Ⓓ+6 的 “ 星期 ” 中以 “ 0 ～ 6 ” 存储 “ 星期日～星期六 ”。

星期	星期日	星期一	星期二	星期三	星期四	星期五	星期六
存储数据	0	1	2	3	4	5	6

(4) 闰年时将进行自动修正。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· Ⓓ中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)

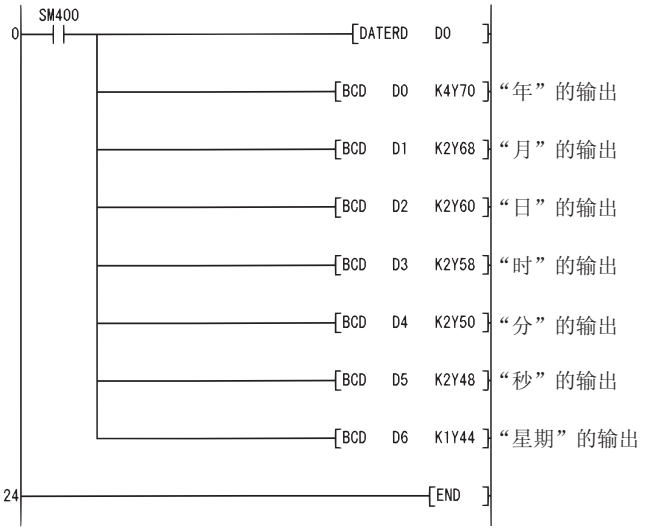
(出错代码： 4101)

程序示例

(1) 以下为将时钟数据以 BCD 值进行输出的程序。

年 ..... Y70 ～ Y7F
月 ..... Y68 ～ Y6F
日 ..... Y60 ～ Y67
时 ..... Y58 ～ Y5F
分 ..... Y50 ～ Y57
秒 ..... Y48 ～ Y4F
星期 ..... Y44 ～ Y47

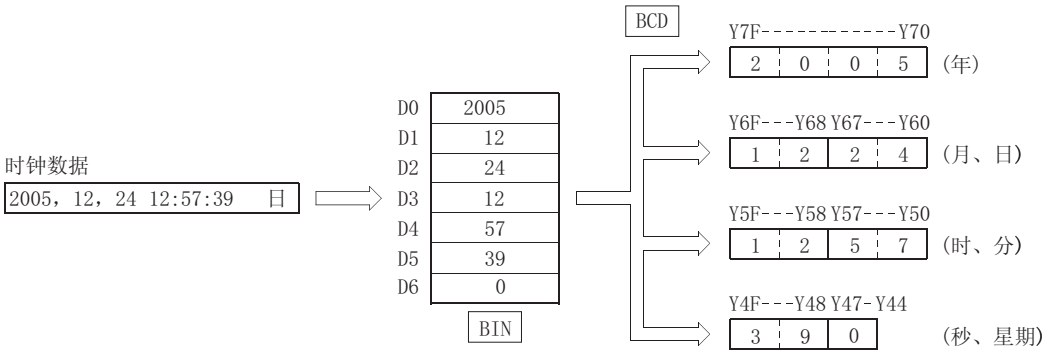
[梯形图模式]



[列表模式]

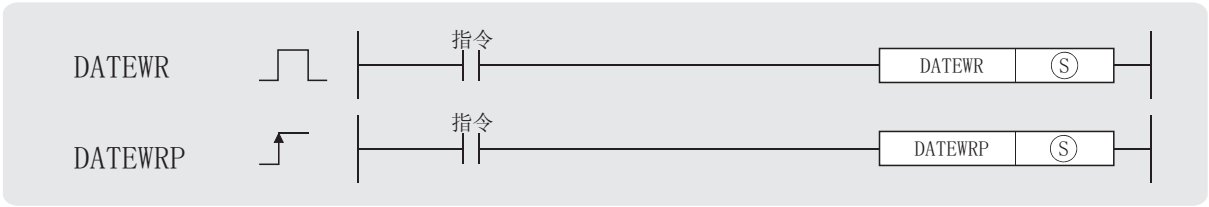
步	指令	软元件	
0	LD	SM400	
1	DATERD	D0	
3	BCD	D0	K4Y70
6	BCD	D1	K2Y68
9	BCD	D2	K2Y60
12	BCD	D3	K2Y58
15	BCD	D4	K2Y50
18	BCD	D5	K2Y48
21	BCD	D6	K1Y44
24	END		

[动作]



7.15.2 时钟数据的写入 (DATEWR(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ：存储写入到时钟因子中的时钟数据的软元件的起始编号 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
Ⓢ	--	○		--					

功 能

(1) 将Ⓢ中指定的软元件编号后面存储的时钟数据，写入到 CPU 模块的时钟因子中。

Ⓢ	年
Ⓢ+1	月
Ⓢ+2	日
Ⓢ+3	时
Ⓢ+4	分
Ⓢ+5	秒
Ⓢ+6	星期

→ 时钟因子

(2) 各项目的设置是以 BIN 值进行的。

(3) Ⓢ的“ 年 ”是以公历 4 位数在 1980 ～ 2079 的范围内进行设置。

(4) Ⓢ+1 的“ 月 ”是在 1 ～ 12(1 月～ 12 月) 的范围内进行设置。

(5) Ⓢ+2 的“ 日 ”是在 1 ～ 31(1 日～ 31 日) 的范围内进行设置。

(6) Ⓢ+3 的“ 时 ”是在 0 ～ 23(0 时～ 23 时) 的范围内进行设置。(设置为 24 小时时钟。)

(7) Ⓢ+4 的“ 分 ”是在 0 ～ 59(0 分～ 59 分) 的范围内进行设置。

(8) Ⓢ+5 的“ 秒 ”是在 0 ～ 59(0 秒～ 59 秒) 的范围内进行设置。

(9) Ⓢ+6 的“ 星期 ”是以“ 0 ～ 6 ”对“ 星期日～星期六 ”进行设置。

星期	星期日	星期一	星期二	星期三	星期四	星期五	星期六
存储数据	0	1	2	3	4	5	6

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 各项目中设置了超出设置范围的数据时。 (出错代码： 4100)

· ⓢ中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)

(出错代码： 4101)

程序示例

(1) 以下为 X40 变为 ON 时，将以 BCD 值输入的下述时钟数据写入到时钟因子中的程序。

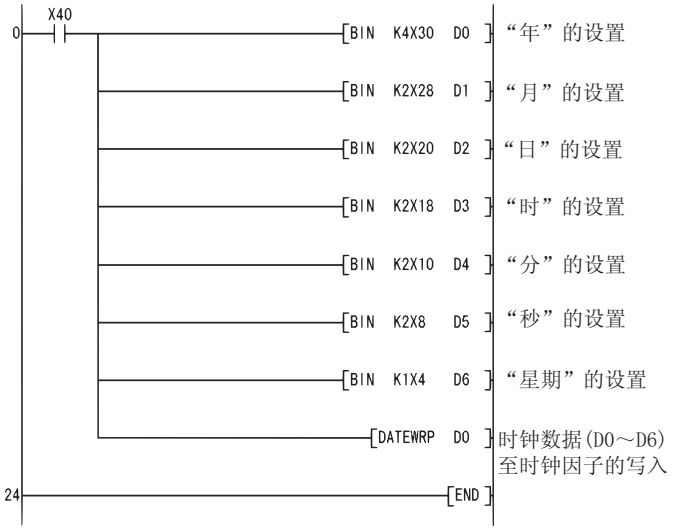
年 .... X30 ～ X3F　　时 .... X18 ～ X1F

月 .... X28 ～ X2F　　分 .... X10 ～ X17

日 .... X20 ～ X27　　秒 .... X8 ～ XF

星期 .. X4 ～ X7

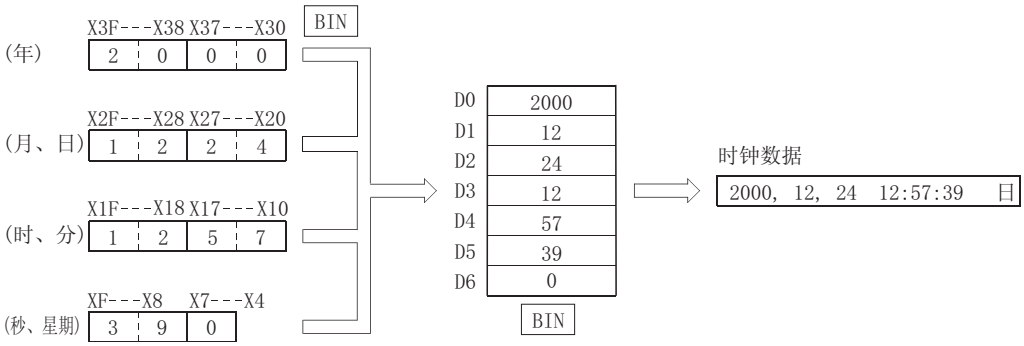
[梯形图模式]



[列表模式]

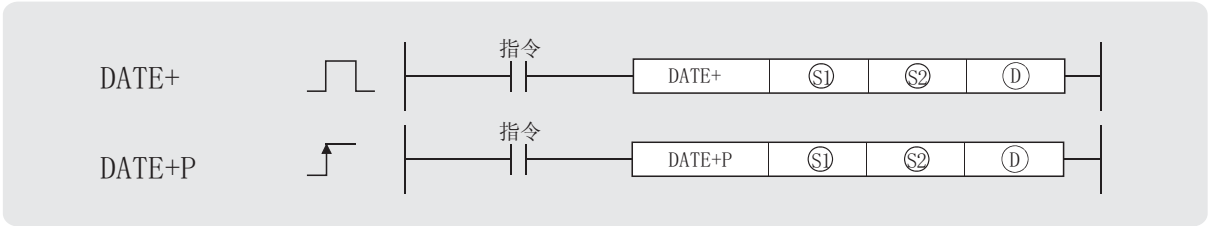
步	指令	软元件	
0	LD	X40	
1	BIN	K4X30	D0
4	BIN	K2X28	D1
7	BIN	K2X20	D2
10	BIN	K2X18	D3
13	BIN	K2X10	D4
16	BIN	K2X8	D5
19	BIN	K1X4	D6
22	DATEWRP	D0	
24	END		

[动作]



7.15.3 时钟数据的加法运算 (DATE+(P))

Basic | High performance | Process | Redundant | Universal



(S1) ：存储被进行加法运算的时间数据的软元件的起始编号 (BIN16 位)。

(S2) ：存储进行加法运算的时间数据的软元件的起始编号 (BIN16 位)。

(D) ：存储加法运算结果时间数据的软元件的起始编号 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
(S1)	--	○		--					
(S2)	--	○		--					
(D)	--	○		--					

功 能

(1) 将(S1)中指定的时间数据与(S2)中指定的时间数据进行加法运算，并将加法运算结果存储到(D)中指定的软元件编号的后面。

		数据范围			数据范围			数据范围
(S1)	时	(0～23)	(S2)	时	(0～23)	(D)	时	(0～23)
(S1)+1	分	(0～59) +	(S2)+1	分	(0～59) ⇨	(D)+1	分	(0～59)
(S1)+2	秒	(0～59)	(S2)+2	秒	(0～59)	(D)+2	秒	(0～59)

例如，将 6 时 32 分 40 秒与 7 时 48 分 10 秒进行加法运算时的情况如下所示。

(S1)	6时		(S2)	7时		(D)	14时
(S1)+1	32分	+	(S2)+1	48分	⇨	(D)+1	20分
(S1)+2	40秒		(S2)+2	10秒		(D)+2	50秒

(2) 运算结果的时间超过了 24 小时时，将该值减去 24 小时后的值作为运算结果。

例如，将 14 时 20 分 30 秒与 20 时 20 分 20 秒进行加法运算时，其结果不是 34 时 40 分 50 秒，而是 10 时 40 分 50 秒。

(S1)	14时		(S2)	20时		(D)	10时
(S1)+1	20分	+	(S2)+1	20分	⇨	(D)+1	40分
(S1)+2	30秒		(S2)+2	20秒		(D)+2	50秒

备 注

关于时、分、秒的可设置数据，请参阅 7.15.2 项。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

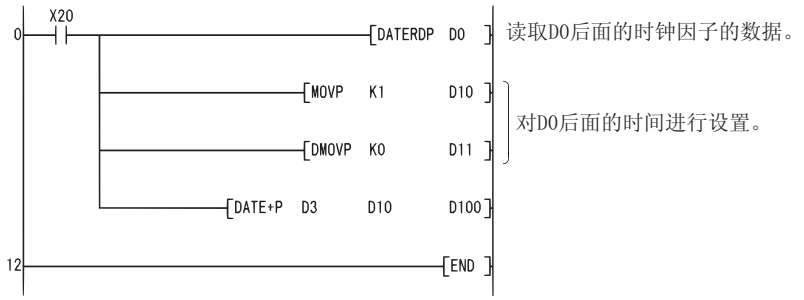
· ⓢ1、ⓢ2 的数据超出了范围时。 (出错代码： 4100)

· ⓢ1、ⓢ2、ⓓ 中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)
(出错代码： 4101)

程序示例

(1) 以下为 X20 变为 ON 时，对从时钟因子中读取的时间数据进行 1 小时的加法运算后，将运算结果存储到 D100 后面的程序。

[梯形图模式]

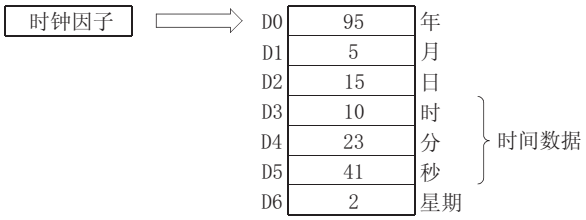


[列表模式]

步	指令	软元件		
0	LD	X20		
1	DATERDP	D0		
3	MOVP	K1	D10	
5	DMOVP	K0	D11	
8	DATE+P	D3	D10	D100
12	END			

[动作]

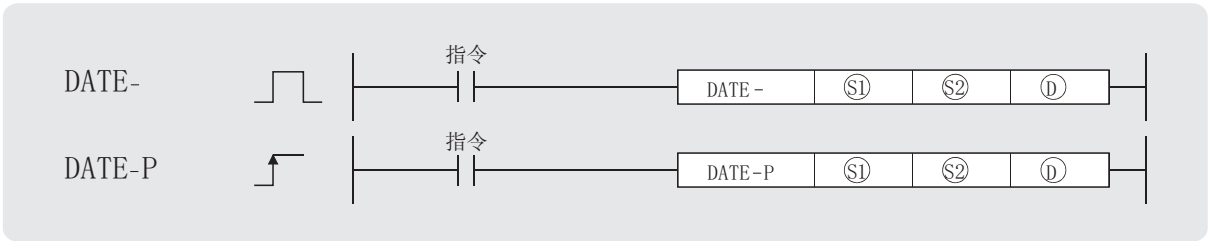
· 通过 DATERDP 指令读取时间数据



· 通过 DATE+P 指令进行加法运算

7.15.4 时钟数据的减法运算 (DATE-(P))

Basic | High performance | Process | Redundant | Universal



S1 ：存储被进行减法运算的时间数据的软元件的起始编号 (BIN16 位)。

S2 ：存储进行减法运算的时间数据的软元件的起始编号 (BIN16 位)。

D ：存储减法运算结果时间数据的软元件的起始编号 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
S1	--	○		--					
S2	--	○		--					
D	--	○		--					

功 能

(1) 将S1中指定的时间数据与S2中指定的时间数据进行减法运算，并将减法运算结果存储到D中指定的软元件编号的后面。

		数据范围			数据范围			数据范围
S1	时	(0～23)	S2	时	(0～23)	D	时	(0～23)
S1+1	分	(0～59)	S2+1	分	(0～59)	D+1	分	(0～59)
S1+2	秒	(0～59)	S2+2	秒	(0～59)	D+2	秒	(0～59)

例如，将 10 时 40 分 20 秒与 3 时 50 分 10 秒进行减法运算时的情况如下所示。

S1	10时		S2	3时		D	6时
S1 +1	40分	—	S2 +1	50分	⇨	D+1	50分
S1 +2	20秒		S2 +2	10秒		D+2	10秒

(2) 运算结果的时间为负数时，将该值加上 24 小时后的值作为运算结果。

例如，将 4 时 50 分 32 秒与 10 时 42 分 12 秒进行减法运算时，其结果不是 -6 时 8 分 20 秒，而是 18 时 8 分 20 秒。

S1	4时		S2	10时		D	18时
S1+1	50分	—	S2+1	42分	⇨	D+1	8分
S1+2	32秒		S2+2	12秒		D+2	20秒

备 注

关于时、分、秒的可设置数据，请参阅 7.15.2 项。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

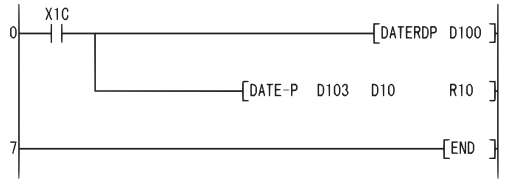
· ⓢ1、ⓢ2 的数据超出了范围时。 (出错代码： 4100)

· ⓢ1、ⓢ2、ⓓ 中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)
(出错代码： 4101)

程序示例

(1) 以下为 X1C 变为 ON 时，将从时钟因子中读取的时间数据与 D10 后面存储的时间数据进行减法运算后，将运算结果存储到 R10 后面的程序。

[梯形图模式]

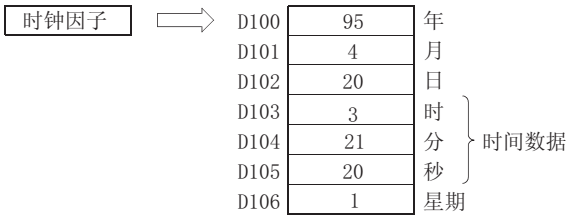


[列表模式]

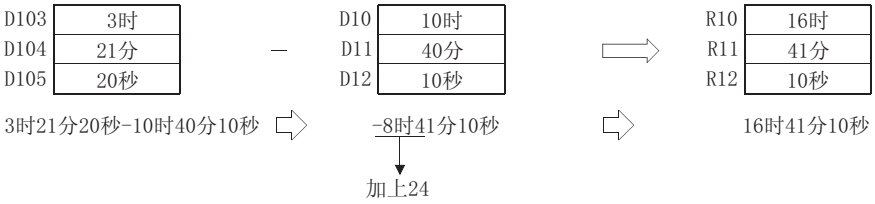
步	指令	软元件		
0	LD	X1C		
1	DATERDP	D100		
3	DATE-P	D103	D10	R10
7	END			

[动作]

· 通过 DATERDP 指令读取时间数据



· 通过 DATE-P 指令进行减法运算 (D10 ～ D12 中指定了 10 时 40 分 10 秒时)



7.15.5 时间数据的转换（小时、分、秒→秒）(SECOND(P))

Basic | High performance | Process | Redundant | Universal

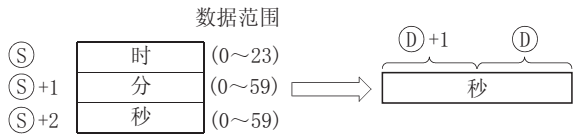
Ⓢ ：存储转换前的时钟数据的软元件的起始编号 (BIN16 位)。

Ⓓ ：存储转换后的时钟数据的软元件的起始编号 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
Ⓢ	--	○		--				--	
Ⓓ	○	○		○				--	

功 能

(1) 将Ⓢ中指定的软元件编号后面存储的时间数据换算为秒后，将换算结果存储到Ⓓ中指定的软元件中。



例如，指定为 4 时 29 分 31 秒时的情况如下所示。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

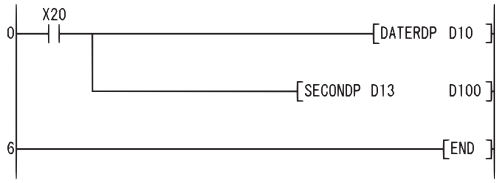
· Ⓢ的数据超出了范围时。 (出错代码： 4100)

· Ⓢ中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)
(出错代码： 4101)

程序示例

(1) 以下为 X20 变为 ON 时，将从时钟因子中读取的时间数据换算为秒后，存储到 D100、D101 中的程序。

[梯形图模式]

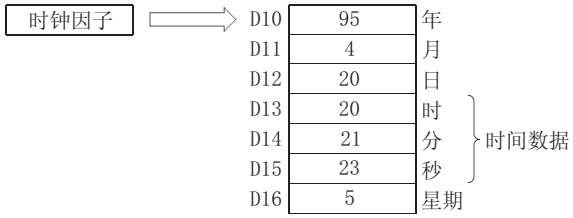


[列表模式]

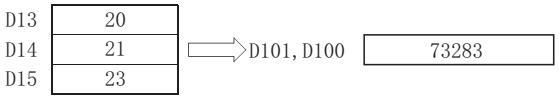
步	指令	软元件	
0	LD	X20	
1	DATERDP	D10	
3	SECONDP	D13	D100
6	END		

[动作]

· 通过 DATERDP 指令读取时间数据

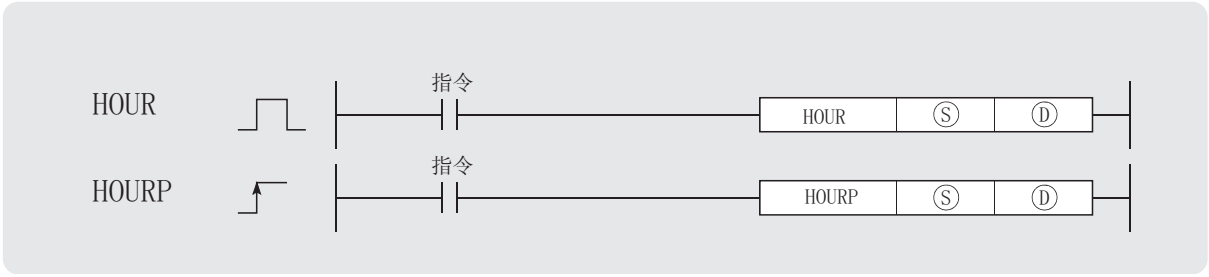


· 通过 SECONDP 指令换算为秒



7.15.6 时间数据的转换（秒→小时、分、秒）(HOUR(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ：存储转换前的时钟数据的软元件的起始编号（BIN16 位）。

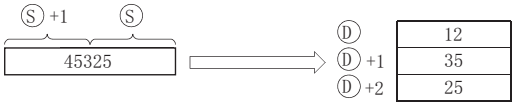
Ⓓ：存储转换后的时钟数据的软元件的起始编号（BIN16 位）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○	○		○				○	--
Ⓓ	--	○		--				--	--

功 能

(1) 将Ⓢ中指定的软元件编号后面存储的秒数据换算为时、分、秒后，将换算结果存储到Ⓓ中指定的软元件后面。

例如，指定了 45325 秒时的情况如下所示。



出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· Ⓢ的数据超出了范围时。 （出错代码： 4100）

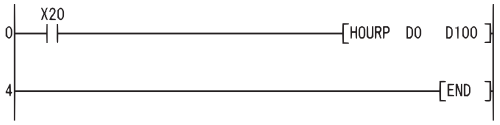
· Ⓓ中指定的软元件超出了相应软元件的范围时。（仅通用型 QCPU）

（出错代码： 4101）

程序示例

(1) 以下为 X20 变为 ON 时，将 D0、D1 中存储的秒数据换算为时、分、秒后，存储到 D100 后面的程序。

[梯形图模式]

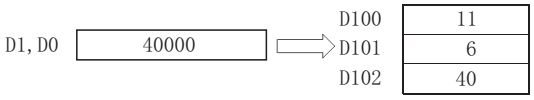


[列表模式]

步	指令	软元件	
0	LD	X20	
1	HOURP	D0	D100
4	END		

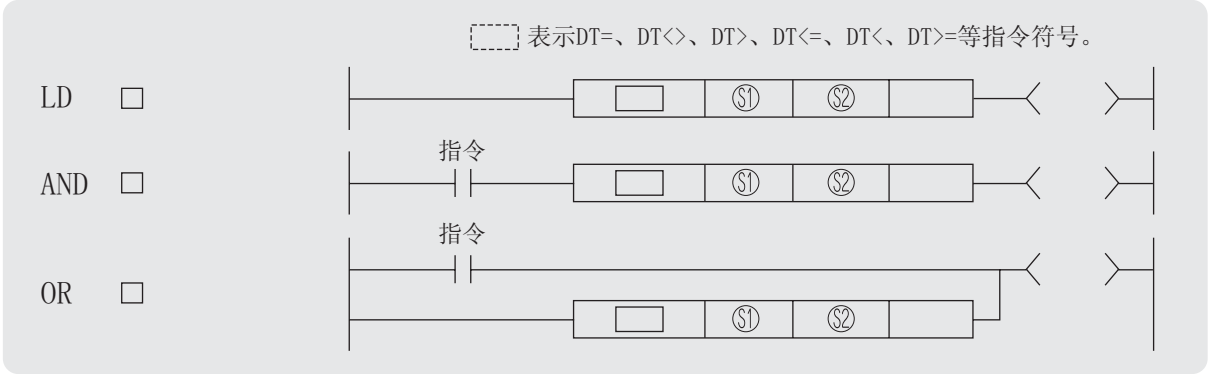
[动作]

· 通过 HOURP 指令进行至时、分、秒的转换 (D0、D1 中指定了 40000 秒时)



7.15.7 日期比较 (DT=、DT<>、DT>、DT<=、DT<、DT>=)

· QnU(D)(H)CPU：序列号的前 5 位数为“ 10102 ”以后
· QnUDE(H)CPU：序列号的前 5 位数为“ 10102 ”以后



(S1) ：存储进行比较的数据的软元件的起始编号 (BIN16 位)。
(S2) ：存储进行比较的数据的软元件的起始编号 (BIN16 位)。
n ：表示比较对象的值或者存储比较对照的数据数 (BIN16 位)。

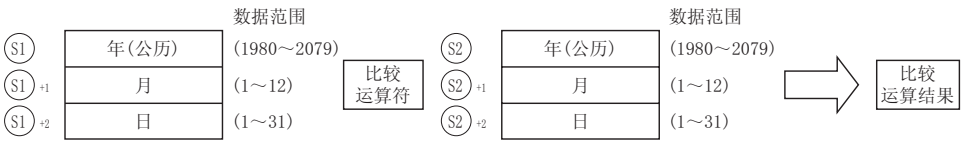
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
(S1)	--	○		--				--	--
(S2)	--	○		--				--	--
n	--	○		○				○	--

功 能

(1) 对(S1)、(S2)中指定的日期数据进行比较。或者，将(S1)中指定的日期数据与当前日期进行比较。通过 n 可以选择比较对象。

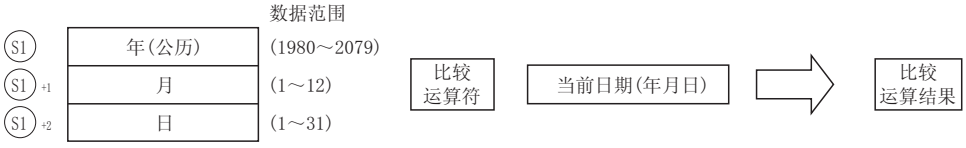
(a) 与任意日期数据的比较

· 将(S1)中指定的日期数据与(S2)中指定的日期数据按照 n 的条件通过 a 触点进行比较。



(b) 与当前日期数据的比较

· 将(S1)中指定的日期数据与当前的日期数据按照 n 的条件通过 a 触点进行比较。



要 点

与任意的日期数据比较时，或者与当前的日期数据比较时，Ⓢ1、Ⓢ2 中的某一个处于以下状况时将变为运算出错状态（出错代码： 4101）或者可能导致误动作。

- 变址修饰目标的指定范围超出了软元件范围时。
- 在未设置文件寄存器的状态下指定了文件寄存器时。

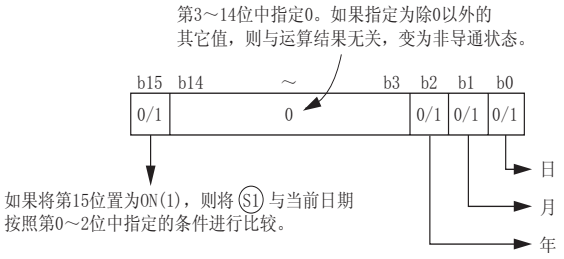
(2) 各项目设置是以 BIN 值格式进行设置的。

(3) Ⓢ1、Ⓢ2 的“ 年 ”是以公历 4 位数在 1980 ～ 2079 的范围内进行设置。

(4) Ⓢ1+1、Ⓢ2+1 的“ 月 ”是在 1 ～ 12(1 月～ 12 月）的范围内进行设置。

(5) Ⓢ1+2、Ⓢ2+2 的“ 日 ”是在 1 ～ 31(1 日～ 31 日）的范围内进行设置。

(6) 通过在 n 中指定下图的值，可以对比较对象进行详细指定。
n 的位构成如下所示。



(a) 比较对象日期（第 0 ～ 2 位）

- 0：不对比较对象的日期数据（年 / 月 / 日）进行比较。
- 1：对比较对象的日期数据（年 / 月 / 日）进行比较。

(b) 比较运算对象（第 15 位）

- 0：将Ⓢ1 中指定的日期数据与Ⓢ2 中指定的日期数据进行比较。
- 1：将Ⓢ1 中指定的日期数据与当前的日期数据进行比较。
  Ⓢ2 中指定的日期数据将被忽略。

(c) 比较对象位的处理内容如下所示。

与任意的日期数据进行比较时的 n 值	与当前的日期数据进行比较时的 n 值	比较对象日期	处理内容
0001H	8001H	日	仅对日 (Ⓢ1+2) 进行比较。
0002H	8002H	月	仅对月 (Ⓢ1+1) 进行比较。
0003H	8003H	月、日	对月 (Ⓢ1+1)、日 (Ⓢ1+2) 进行比较。
0004H	8004H	年	仅对年 (Ⓢ1) 进行比较。
0005H	8005H	年、日	对年 (Ⓢ1)、日 (Ⓢ1+2) 进行比较。
0006H	8006H	年、月	对年 (Ⓢ1)、月 (Ⓢ1+1) 进行比较。
0007H	8007H	年、月、日	对年 (Ⓢ1)、月 (Ⓢ1+1)、日 (Ⓢ1+2) 均进行比较。
除 0001H ～ 0007H、8001H ～ 8007H 以外		无	对年 (Ⓢ1)、月 (Ⓢ1+1)、日 (Ⓢ1+2) 均不进行比较。（变为非导通状态。）

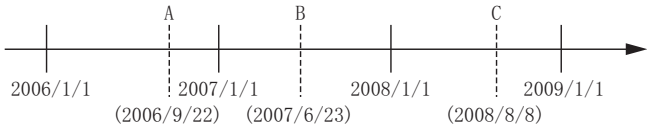
(7) 在比较对象软元件中存储的数据不能被识别为日期数据的情况下，执行指令后将 SM709 置为 ON，变为非导通状态。即使是在不能被识别为日期数据的情况下，只要是在设置范围内，SM709 将不变为 ON。

Ⓢ1～Ⓢ1+2 或者Ⓢ2～Ⓢ2+2 超出了指定软元件范围时，SM709 也将变为 ON，变为非导通状态。一旦 SM709 变为 ON 后，在进行复位 / 电源 OFF 之前将保持 ON 状态不变，因此应根据需要将其置为 OFF。

(8) 各指令的比较运算结果如下所示。

内的指令符号	条件	比较运算结果	内的指令符号	条件	比较运算结果
DT=	Ⓢ1 = Ⓢ2	导通状态	DT=	Ⓢ1 ≠ Ⓢ2	非导通状态
DT<>	Ⓢ1 ≠ Ⓢ2		DT<>	Ⓢ1 = Ⓢ2	
DT>	Ⓢ1 > Ⓢ2		DT>	Ⓢ1 ≤ Ⓢ2	
DT<=	Ⓢ1 ≤ Ⓢ2		DT<=	Ⓢ1 > Ⓢ2	
DT<	Ⓢ1 < Ⓢ2		DT<	Ⓢ1 ≥ Ⓢ2	
DT>=	Ⓢ1 ≥ Ⓢ2		DT>=	Ⓢ1 < Ⓢ2	

(a) 日期的比较示例如下所示。



上述日期 A、B、C 的比较运算结果如下所示。

即使在相同的条件下进行比较，根据选择的比较对象其比较运算结果将有所不同。

比较对象	比较条件		
	A<B	B<C	A<C
日	○	×	×
月	×	○	×
月、日	×	○	×
年	○	○	○
年、日	○	○	○
年、月	○	○	○
年、月、日	○	○	○
无	×	×	×

○：导通　×：非导通

(b) 即使在所比较的日期是实际上并不存在的日期的情况下，只要是在允许设置范围内的日期，将按下述条件进行比较运算。

· 日期 A：2006/02/30（该日期实际上并不存在，但在允许设置范围内。）

· 日期 B：2007/03/29

· 日期 C：2008/02/31（该日期实际上并不存在，但在允许设置范围内。）

比较对象	比较条件		
	A<B	B<C	A<C
日	×	×	○
月	×	×	×
月、日	○	×	○
年	○	○	○
年、日	○	○	○
年、月	○	○	○
年、月、日	○	○	○
无	×	×	×

○：导通　×：非导通

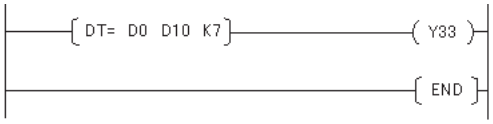
出　错

(1) DT=、DT<>、DT>、DT<=、DT<、DT>= 指令中无运算出错。

程序示例

(1) 以下为将 D0 的数据与 D10 的数据（年、月、日）进行比较，当 D0 的数据与 D10 的数据一致时，将 Y33 变为导通状态的程序。

[梯形图模式]

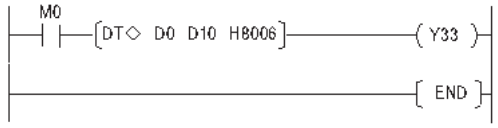


[列表模式]

步	指令	软元件
0	LDDT=	D0 D10 K7
4	OUT	Y33
5	END	

(2) 以下为 M0 变为 ON 时，将 D0 的数据与当前的日期数据（年、月）进行比较，当 D0 的数据与当前的日期数据不一致时，将 Y33 变为导通状态的程序。

[梯形图模式]

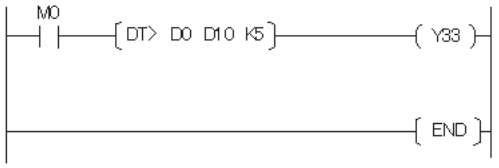


[列表模式]

步	指令	软元件
0	LD	M0
1	ANDDT◇	D0 D10 H8006
5	OUT	Y33
6	END	

(3) 以下为 M0 变为 ON 时，将 D0 的数据与 D10 的数据（年、日）进行比较，当 D10 的数据小于 D0 的数据时，将 Y33 变为导通状态的程序。

[梯形图模式]

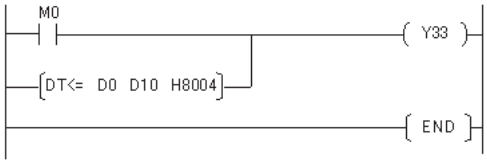


[列表模式]

步	指令	软元件
0	LD	M0
1	ANDDT>	D0 D10 K5
5	OR	M10
6	ANB	
7	OUT	Y33
8	END	

(4) 以下为将 D0 的数据与当前的日期数据（年）进行比较，当当前的日期数据大于 D0 的数据时，将 Y33 变为导通状态的程序。

[梯形图模式]



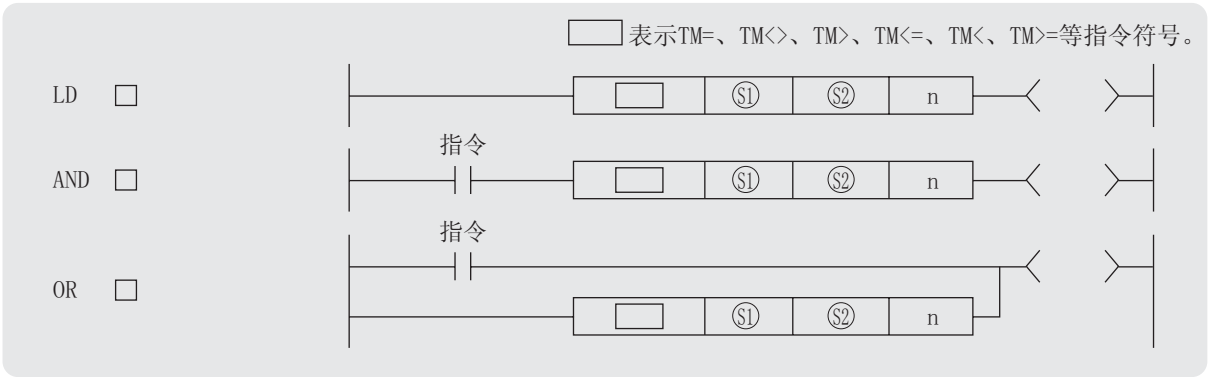
[列表模式]

步	指令	软元件
0	LD	M0
1	AND	M10
2	ORDT<=	D0 D10 H8004
6	OUT	Y33
7	END	

7.15.8 时间比较 (TM=、TM<>、TM>、TM<=、TM<、TM>=)

Basic × | High performance × | Process × | Redundant × | Ver. Universal

· QnU(D)(H)CPU：序列号的前 5 位数为“ 10102 ”以后
· QnUDE(H)CPU：序列号的前 5 位数为“ 10102 ”以后



S1 ：存储进行比较的数据的软元件的起始编号 (BIN16 位)。
S2 ：存储进行比较的数据的软元件的起始编号 (BIN16 位)。
n ：表示比较方法的值或者存储比较方法的数据数 (BIN16 位)。

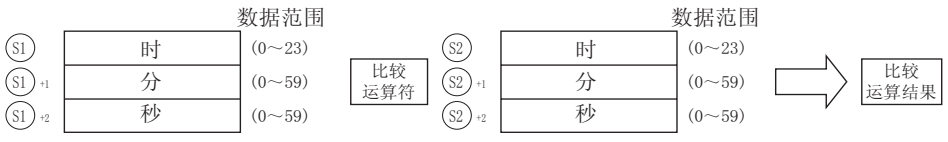
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
S1	--	○		--				--	--
S2	--	○		--				--	--
n	--	○		○				○	--

功 能

(1) 对S1、S2中指定的时间数据进行比较。或者，将S1中指定的时间数据与当前时间进行比较。通过 n 可以选择比较对象。

(a) 与任意时间数据的比较

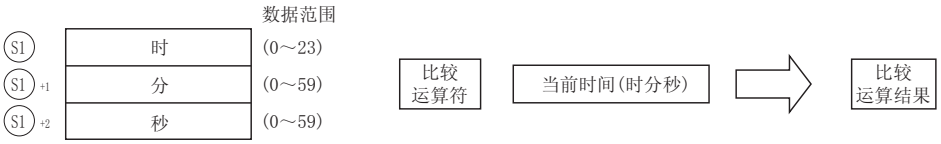
· 将S1中指定的时间数据与S2中指定的时间数据按照 n 的值通过 a 触点进行比较。



(b) 与当前时间数据的比较

· 将Ⓢ1中指定的时间数据与当前的时间数据按照 n 的值通过 a 触点进行比较。

· 将Ⓢ2中指定的时间数据作为虚拟数据处理，将其忽略。



要 点

与任意的时间数据比较时，或者与当前的时间数据比较时，Ⓢ1、Ⓢ2中的某一个处于以下状况时将变为运算出错状态（出错代码： 4101）或者可能导致误动作。

· 变址修饰目标的指定范围超出了软元件范围时。

· 在未设置文件寄存器的状态下指定了文件寄存器时。

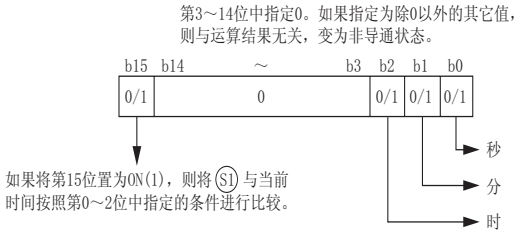
(2) 各项目设置是以 BIN 值格式进行设置的。

(3) Ⓢ1、Ⓢ2的“ 时 ”是在 0 ～ 23(0 时～ 23 时）的范围内进行设置。

(4) Ⓢ1+1、Ⓢ2+1 的“ 分 ”是在 0 ～ 59(0 分～ 59 分）的范围内进行设置。

(5) Ⓢ1+2、Ⓢ2+2 的“ 秒 ”是在 0 ～ 59(0 秒～ 59 秒）的范围内进行设置。

(6) 通过在 n 中指定下图的值，可以对比较对象进行详细指定。
n 的位构成如下所示。



(a) 比较对象时间（第 0 ～ 2 位）

· 0：不对比较对象的时间数据（时 / 分 / 秒）进行比较。

· 1：对比较对象的时间数据（时 / 分 / 秒）进行比较。

(b) 比较运算对象（第 15 位）

· 0：将Ⓢ1中指定的时间数据与Ⓢ2中指定的时间数据进行比较。

· 1：将Ⓢ1中指定的时间数据与当前的时间数据进行比较。
Ⓢ2中指定的时间数据将被忽略。

(c) 比较对象位的处理内容如下所示。

与任意的时间数据进行比较时的 n 值	与当前的时间数据进行比较时的 n 值	比较对象时间	处理内容
0001H	8001H	秒	仅对秒 (Ⓢ1+2) 进行比较。
0002H	8002H	分	仅对分 (Ⓢ1+1) 进行比较。
0003H	8003H	分、秒	对分 (Ⓢ1+1)、秒 (Ⓢ1+2) 进行比较。
0004H	8004H	时	仅对时 (Ⓢ1) 进行比较。
0005H	8005H	时、秒	对时 (Ⓢ1)、秒 (Ⓢ1+2) 进行比较。
0006H	8006H	时、分	对时 (Ⓢ1)、分 (Ⓢ1+1) 进行比较。
0007H	8007H	时、分、秒	对时 (Ⓢ1)、分 (Ⓢ1+1)、秒 (Ⓢ1+2) 均进行比较。
除 0001H ～ 0007H、8001H ～ 8007H 以外		无	对时 (Ⓢ1)、分 (Ⓢ1+1)、秒 (Ⓢ1+2) 均不进行比较。(变为非导通状态。)

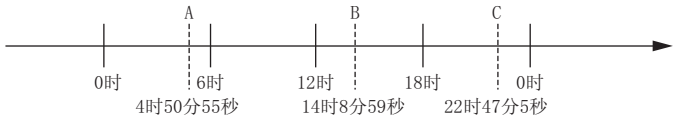
(7) 在比较对象软元件中存储的数据不能被识别为时间数据的情况下，执行指令后将 SM709 置为 ON，变为非导通状态。一旦 SM709 变为 ON 后，在进行复位 / 电源 OFF 之前将保持 ON 状态不变，因此应根据需要将其置为 OFF。

Ⓢ1 ～ Ⓢ1+2 或者Ⓢ2 ～ Ⓢ2+2 超出了指定软元件范围时，SM709 也将变为 ON，变为非导通状态。

(8) 各指令的比较运算结果如下所示。

内的指令符号	条件	比较运算结果	内的指令符号	条件	比较运算结果
TM=	Ⓢ1 = Ⓢ2	导通状态	TM=	Ⓢ1 ≠ Ⓢ2	非导通状态
TM<>	Ⓢ1 ≠ Ⓢ2		TM<>	Ⓢ1 = Ⓢ2	
TM>	Ⓢ1 > Ⓢ2		TM>	Ⓢ1 ≤ Ⓢ2	
TM<=	Ⓢ1 ≤ Ⓢ2		TM<=	Ⓢ1 > Ⓢ2	
TM<	Ⓢ1 < Ⓢ2		TM<	Ⓢ1 ≥ Ⓢ2	
TM>=	Ⓢ1 ≥ Ⓢ2		TM>=	Ⓢ1 < Ⓢ2	

(a) 时间的比较示例如下所示。



上述时间 A、B、C 的比较运算结果如下所示。
即使在相同的条件下进行比较，根据选择的比较对象其比较运算结果将有所不同。

比较对象	比较条件		
	A<B	B<C	A<C
秒	○	×	×
分	×	○	×
分、秒	×	○	×
时	○	○	○
时、秒	○	○	○
时、分	○	○	○
时、分、秒	○	○	○
无	×	×	×

○：导通　×：非导通

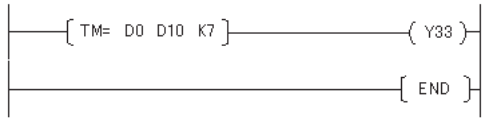
出　错

(1) 在 TM=、TM<>、TM>、TM<=、TM<、TM>= 指令中无运算出错。

程序示例

(1) 以下为将 D0 的数据与 D10 的数据（时、分、秒）进行比较，当 D0 的数据与 D10 的数据一致时，将 Y33 变为导通状态的程序。

[梯形图模式]

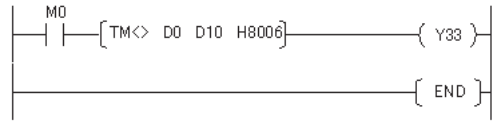


[列表模式]

步	指令	软元件
0	LDTM=	D0 D10 K7
4	OUT	Y33
5	END	

(2) 以下为 M0 变为 ON 时，将 D0 的数据与当前的时间数据（时、分）进行比较，当 D0 的数据与当前的时间数据不一致时，将 Y33 变为导通状态的程序。

[梯形图模式]

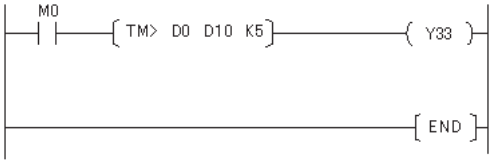


[列表模式]

步	指令	软元件
0	LD	M0
1	ANDTM<>	D0 D10 H8006
5	OUT	Y33
6	END	

(3) 以下为 M0 变为 ON 时，将 D0 的数据与 D10 的数据（时、秒）进行比较，当 D10 的数据小于 D0 的数据时，将 Y33 变为导通状态的程序。

[梯形图模式]

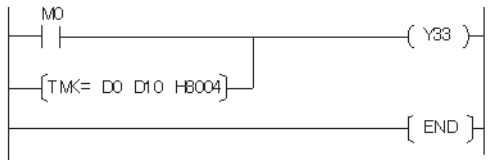


[列表模式]

步	指令	软元件
0	LD	M0
1	ANDTM>	D0 D10 K5
5	OR	M10
6	ANB	
7	OUT	Y33
8	END	

(4) 以下为将 D0 的数据与当前的时间数据（时）进行比较，当当前的时间数据大于 D0 的数据时，将 Y33 变为导通状态的程序。

[梯形图模式]



[列表模式]

步	指令	软元件
0	LD	M0
1	AND	M10
2	ORTMK=	D0 D10 H8004
6	OUT	Y33
7	END	

7.16 程序控制指令

(1) 通过程序控制通指令对执行类型进行了切换时的处理如下表所示。

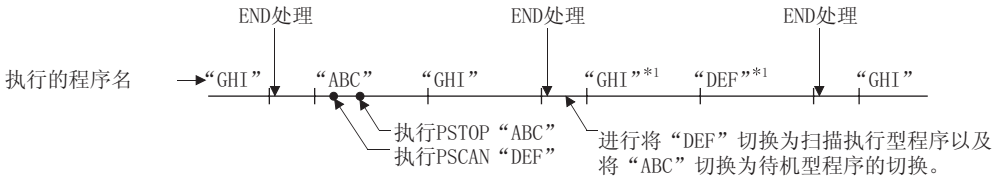
变更前的执行类型	执行指令			
	PSCAN	PSTOP	POFF	PLOW
扫描执行型	保持为扫描执行型不变。	变为待机型。	在下一个扫描中将输出变为 OFF。 从下下个扫描以后变为待机型。	变为低速执行型。
初始执行型	变为扫描执行型。			
待机型		保持为待机型不变。	无处理	
低速执行型	对低速执行型的执行进行中断后，从下一个扫描开始变为扫描执行型。 (从 0 步开始执行)	对低速执行型的执行进行中断后，从下一个扫描开始变为待机型。	对低速执行型的执行进行中断后，在下一个扫描中将输出变为 OFF。 从下下个扫描以后变为待机型。	保持为低速执行型不变。
恒定周期执行型	变为扫描执行型。	变为待机型。	在下一个扫描中将输出变为 OFF。 从下下个扫描以后变为待机型。	变为低速执行型。

要 点

如果将恒定周期执行型程序变更为其它执行类型的程序，将不能恢复为恒定周期执行型程序。

(2) 程序的执行类型切换是通过 END 处理进行的。因此在程序的执行过程中不能切换程序的执行类型。

此外，在同一个扫描中对同一个程序设置了不同的类型时，将变为后执行的执行类型切换指令的执行类型。



*1: “ GHI ”与“ DEF ”的程序执行顺序为参数的程序设置中所设置的顺序。

从恒定执行型程序至其它执行类型的切换时机如下所示。

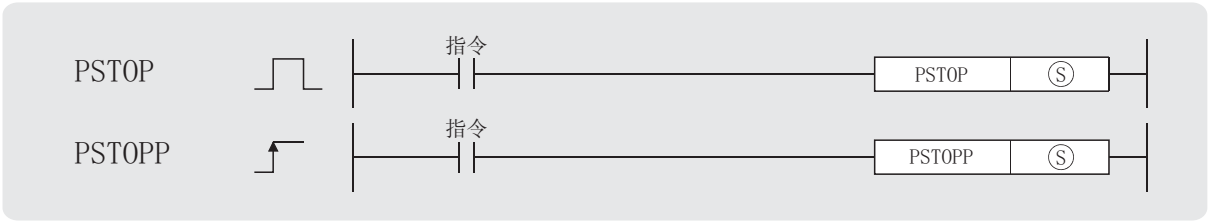
(a) 通用型 QCPU 时
在执行程序控制用指令后的 END 处理中停止恒定周期执行型程序的执行，进行程序类型的变更。

(b) 除通用型 QCPU 以外时
在执行程序控制用指令时停止恒定周期执行型程序的执行，在其 END 处理中进行执行类型的变更。

(3) 如果执行了 POFF 指令，将在下一个扫描中将输出置为 OFF，在下下个扫描后变为待机型。
在执行输出的 OFF 处理之前，即使执行程序控制用指令也将被忽略。

7.16.1 程序待机指令 (PSTOP(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ：置于待机状态的程序文件名的字符串数据或者存储字符串数据的软元件的起始编号（字符串）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
Ⓢ	--	○		--				○	--

功 能

(1) 将Ⓢ中指定的软元件中存储的文件名的程序置为待机状态。

(2) 只有驱动器 0（程序存储器 / 内置 RAM）中存储的程序才可以被置为待机状态。

(3) 指定的程序将在 END 处理中被变为待机状态。

(4) 即使在参数中指定了执行类型，本指令也将优先。

(5) 文件名中无需指定扩展名 (.QPG)。（仅以 .QPG 文件作为对象。）

出 错

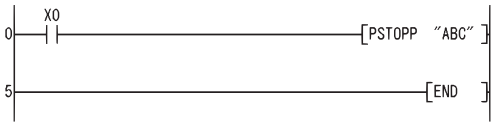
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- Ⓢ中指定的文件名程序不存在时。（出错代码：2410）
- Ⓢ中指定的文件名的程序类型为 SFC 程序时。（出错代码：2412）
- Ⓢ的文件名存储目标软元件超出了相应软元件范围时。（出错代码：4101）

程序示例

(1) 以下为 X0 变为 ON 时，将文件名为 ABC 的程序置为待机状态的程序。

[梯形图模式]

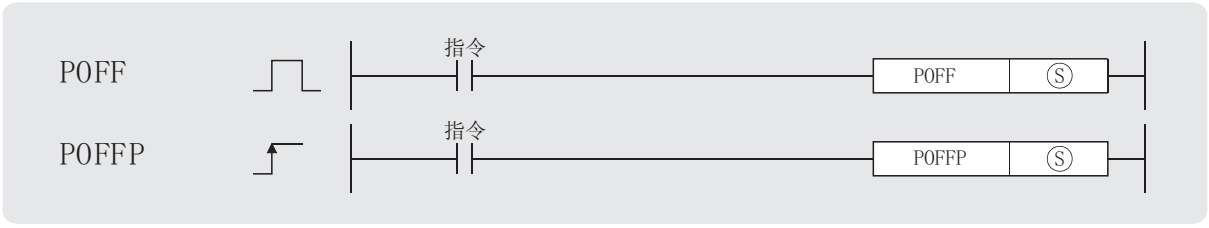


[列表模式]

步	指令	软元件
0	LD	X0
1	PSTOPP	"ABC"
5	END	

7.16.2 程序输出 OFF 待机指令 (POFF(P))

Basic High performance Process Redundant Universal



Ⓢ ：将输出置为 OFF 使之变为待机型程序的文件名或者存储文件名的软元件的起始编号 (字符串)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
Ⓢ	--	○		--				○	--

功 能

(1) 对Ⓢ中指定的软元件中存储的文件名的程序的执行类型进行变更。

· 扫描执行型： 在下一个扫描中将输出置为 OFF(非执行处理)。在下下个扫描后变为待机型。

· 低速执行型： 中断低速执行型程序的执行，在下一个扫描中将输出置为 OFF。在下下个扫描后变为待机型。

(2) 只有驱动器 0(程序存储器) 中存储的程序才可以被置为待机型。

(3) 即使在参数中指定了执行类型，本指令也将优先。

(4) 文件名中无需指定扩展名 (.QPG)。(仅以 .QPG 文件作为对象。)

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· Ⓢ中指定的文件名程序不存在时。 (出错代码： 2410)

· Ⓢ的文件名存储目标软元件超出了相应软元件的范围时。 (出错代码： 4101)

备 注

1. 非执行处理是指，执行与各线圈指令的执行条件为 OFF 状态时相同的处理。

2. 与条件触点的 ON/OFF 状态无关，非执行处理后的各线圈指令的运算结果如下所示。

- OUT 指令 ..... 强制 OFF
- SET 指令
- RST 指令
- SFT 指令
- 基本指令
- 应用指令

.... 状态保持

- PLS 指令
- 脉冲化指令 (□ P)

.... 执行与条件触点 OFF 时相同的处理

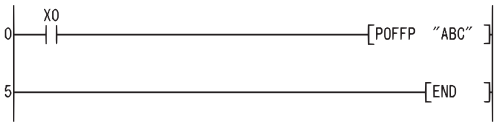
- 低速 / 高速定时器的当前值 . . . 0
- 累计定时器的当前值
- 计数器的当前值

.... 保持

程序示例

(1) 以下为 X0 变为 ON 时，将文件名为 ABC 的程序置为非执行后，变为待机状态的程序。

[梯形图模式]

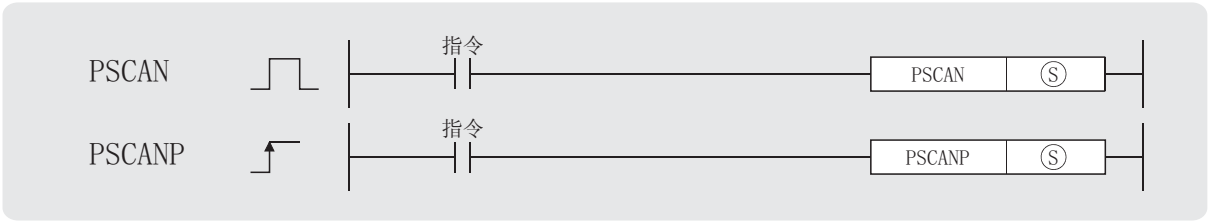


[列表模式]

步	指令	软元件
0	LD	X0
1	POFFP	"ABC"
5	END	

7.16.3 程序扫描执行登录指令 (PSCAN(P))

Basic High performance Process Redundant Universal



Ⓢ：置为扫描执行型的程序的文件名或者存储文件名的软元件的起始编号（字符串）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
Ⓢ	--	○		--				○	--

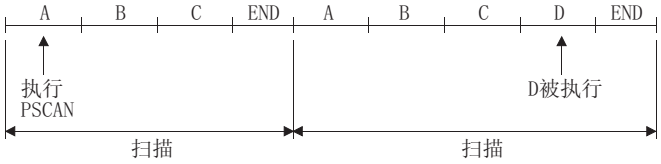
功 能

(1) 将Ⓢ中指定的软元件中存储的文件名的程序置为扫描执行型。

(2) 只有驱动器 0（程序存储器 / 内置 RAM）中存储的程序才可以被置为扫描执行型。

(3) 指定的程序将在 END 处理中被变为扫描执行型。

例 存在程序 A、B、C，通过 A 程序对 D 程序执行了 “ PSCAN ” 时。



(4) 即使在参数中指定了执行类型，本指令也将优先。

(5) 文件名中无需指定扩展名 (.QPG)。（仅以 .QPG 文件作为对象。）

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· Ⓢ中指定的文件名程序不存在时。 (出错代码： 2410)

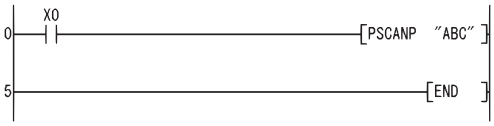
· Ⓢ的文件名存储目标软元件超出了相应软元件范围时。 (出错代码： 4101)

· 指定的文件名的程序类型为 SFC 程序，但已有其它程序名的 SFC 程序处于已启动状态时。
(SFC 程序的重复启动出错)
(通用型 QCPU) (出错代码： 4131)
(高性能型 QCPU、过程 CPU、冗余 CPU) (出错代码： 2504)

程序示例

(1) 以下为 X0 变为 ON 时，将文件名为 ABC 的程序置为扫描执行型的程序。

[梯形图模式]

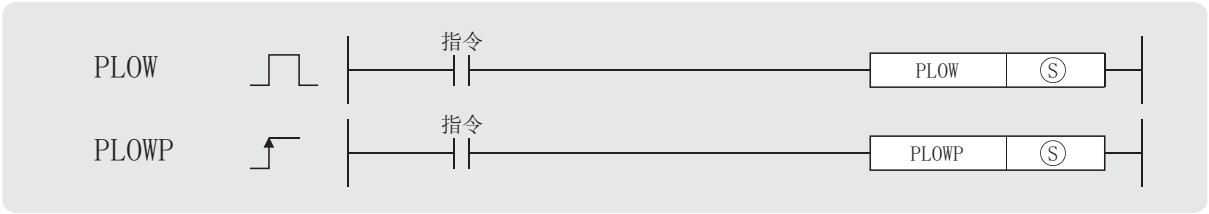


[列表模式]

步	指令	软元件
0	LD	X0
1	PSCANP	"ABC"
5	END	

7.16.4 程序低速执行登录指令 (PLOW(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ：置为低速执行型的程序的文件名或者存储文件名的软元件的起始编号（字符串）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
Ⓢ	--	○		--				○	--

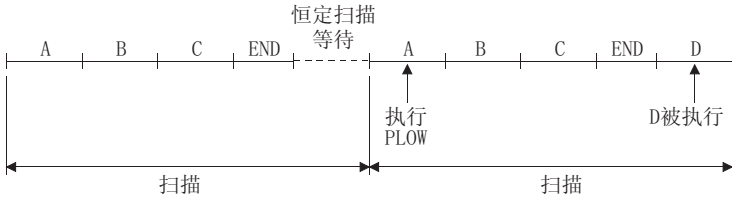
功 能

(1) 将Ⓢ中指定的软元件中存储的文件名的程序置为低速执行型。

(2) 只有驱动器 0（程序存储器 / 内置 RAM）中存储的程序才可以被置为低速执行型。

(3) 指定的程序将在 END 处理中被变为低速执行型。

例 存在程序 A、B、C，通过 A 程序对 D 程序执行了 “ PLOW ” 时。（相当于进行了恒定扫描的设置。）



(4) 即使在参数中指定了执行类型，本指令也将优先。

(5) 文件名中无需指定扩展名 (.QPG)。（仅以 .QPG 文件作为对象。）

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- 指定的文件名程序不存在时。（出错代码：2410）
- 指定的文件名的程序中存在有 CHK 指令时。（出错代码：4235）

程序示例

(1) 以下为 X0 变为 ON 时，将文件名为 ABC 的程序置为低速执行型的程序。

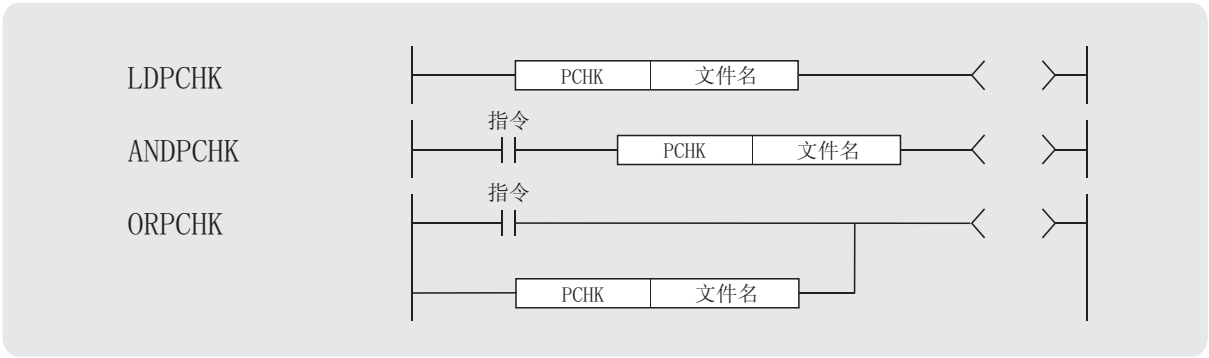
[梯形图模式]

[列表模式]

步	指令	软元件
0	LD	X0
1	PLOWP	"ABC"
5	END	

7.16.5 程序执行状态检查指令 (PCHK)

Basic | High performance | Process | Redundant | Universal



Ⓢ：进行执行状态检查的程序的文件名（字符串）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
Ⓢ	--							○	--

功 能

(1) 检查指定的文件名的程序是处于执行中状态还是未处于执行中（非执行）状态。

(2) 指定的文件名的程序处于执行中状态时，变为导通，处于非执行中状态时变为非导通。

(3) 文件名指定为去除了扩展名 (.QPG) 后的文件名。
例如，文件名为 ABC.QPG 时，指定为“ABC”。

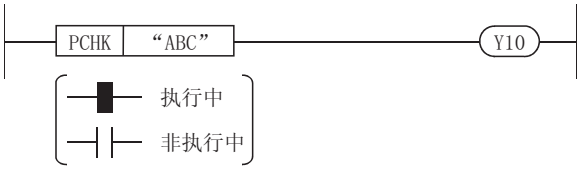
出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 指定的文件名程序不存在时。 （出错代码：2410）

程序示例

(1) 以下为程序文件“ABC.QPG”处于执行中状态时，将 Y10 置为 ON 的程序。



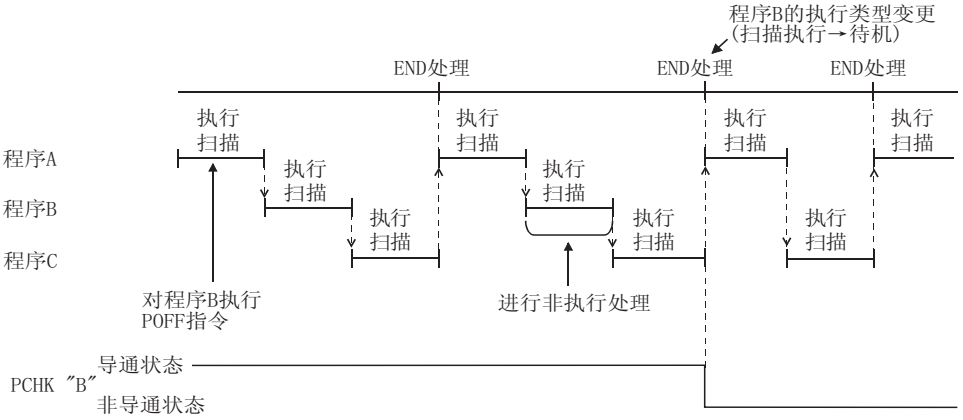
备 注

非执行中表示程序的执行类型为待机型的状态。
执行中表示程序的执行类型为扫描执行型（输出 OFF 中（非执行处理中）也包含在内）、低速执行型、恒定周期执行型的状态。

要 点

指定的文件名的程序（对象程序）处于执行中时，PCHK 指令变为导通状态；指定的文件名的程序（对象程序）处于非执行中时，PCHK 指令变为非导通状态；
通过 POFF 指令将对象程序置为非执行（待机型）状态时，在进行对象程序的非执行处理期间，PCHK 指令变为导通状态。
在非执行处理结束的扫描的 END 处理中，对象程序变为非执行（待机型）状态，PCHK 指令变为非导通状态。
因此，如果对已通过 POFF 指令结束了非执行处理的程序执行了 PCHK 指令，PCHK 指令有时会变为导通状态，应加以注意。

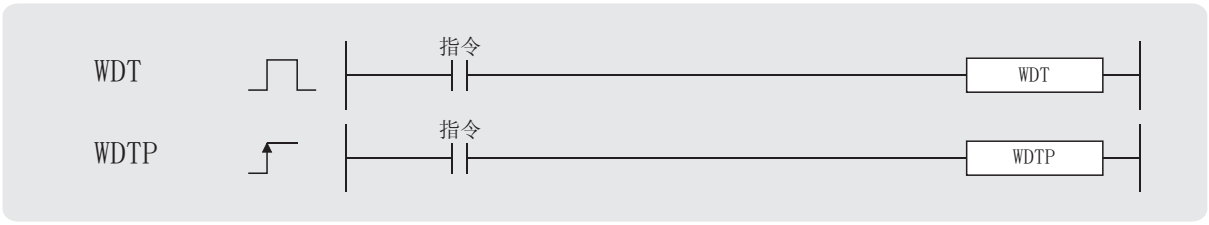
按程序 A →程序 B →程序 C 的顺序执行的情况下，程序 A 对程序 B 执行了 POFF 指令，程序 C 对程序 B 执行了 PCHK 指令时的动作如下图所示。



7.17 其它指令

7.17.1 看门狗定时器复位 (WDT(P))

Basic | High performance | Process | Redundant | Universal



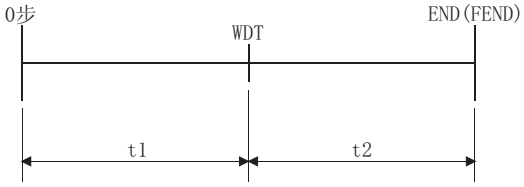
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
--	--								

功 能

(1) 在顺控程序的执行过程中进行看门狗定时器的复位。

(2) 根据条件扫描时间超出了看门狗定时器的设置值时使用本指令。
在扫描时间超出了各扫描的看门狗定时器的设置值时，应通过外围设备的参数设置对看门狗定时器的设置值进行变更。

(3) 应使从 0 步开始至 WDT 指令为止的 t1 及从 WDT 指令开始至 END(FEND) 指令为止的 t2 均不超过看门狗定时器的设置值。



(4) 虽然在 1 个扫描中可以使用 2 次以上 WDT 指令，但发生异常时至输出 OFF 为止需要耗费一定的时间，应加以注意。

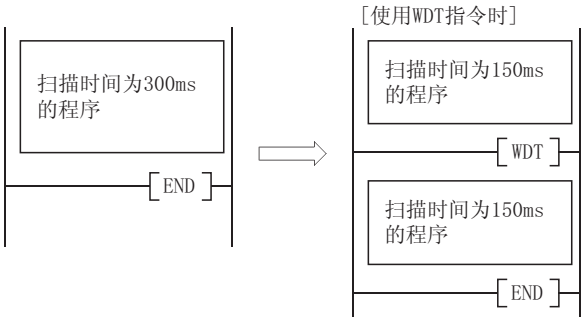
(5) 即使执行了 WDT、WDTP 指令，特殊寄存器中存储的扫描时间值也不会被清除。
因此，各特殊寄存器的扫描时间值有时会大于参数中设置的看门狗定时器的设置值。

出　错

(1) WDT(P) 指令中无运算出错。

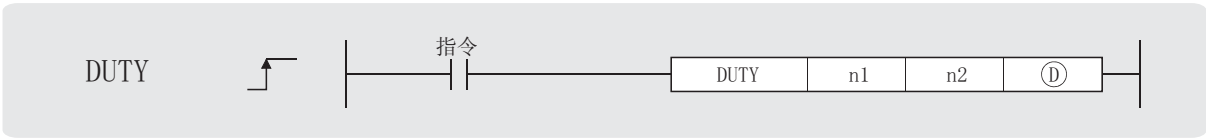
程序示例

(1) 以下为看门狗定时器的设置为 200ms，根据程序的执行条件 0 ～ END(FEND) 指令为止的时间为 300ms 时的程序。



7.17.2 定时脉冲发生 (DUTY)

Basic | High performance | Process | Redundant | Universal



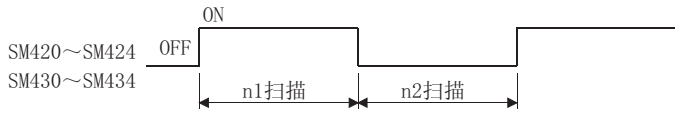
n1 ：置为 ON 的扫描数 (BIN16 位)。
n2 ：置为 OFF 的扫描数 (BIN16 位)。
Ⓓ ：用户用定时时钟 (SM420 ～ SM424、SM430 ～ SM434) (位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
n1	○			○					--
n2	○			○					--
Ⓓ	○ *1			--					--

*1: 仅 SM420 ～ SM424、SM430 ～ SM434 可以使用。

功 能

(1) 将Ⓓ中指定的用户用定时时钟 (SM420 ～ SM424、SM430 ～ SM434) 按 n1 中指定的扫描数置为 ON 后，按 n2 中指定的扫描数置为 OFF。



(2) 在扫描执行型程序中使用 SM420 ～ SM424，在低速执行型程序中使用 SM430 ～ SM434。

(3) n1、n2 被设置为 0 时的情况如下所示。

(a) n1=0， n2 ≥ 0 ... SM420 ～ SM424/SM430 ～ SM434 保持 OFF 状态不变。

(a) n1 < 0，n2=0 ... SM420 ～ SM424/SM430 ～ SM434 保持 ON 状态不变。

(4) 执行 DUTY 指令时，将 n1、n2、Ⓓ中指定的数据登录到系统中后，通过 END 处理进行定时脉冲的 ON/OFF。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

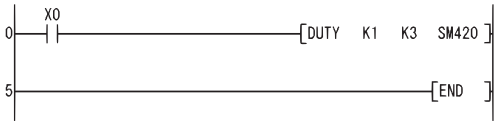
· Ⓓ中指定的软元件为除 SM420 ～ SM424、SM430 ～ SM434 以外时。 (出错代码： 4101)

· n1、n2 小于 0 时。 (出错代码： 4100)

程序示例

(1) 以下为 X0 变为 ON 时，将 SM420 置为 1 个扫描 ON，3 个扫描 OFF 的程序。

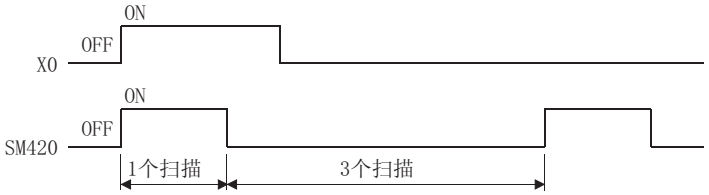
[梯形图模式]



[列表模式]

步	指令	软元件		
0	LD	X0		
1	DUTY	K1	K3	SM420
5	END			

[动作]



7.17.3 时间检查指令 (TIMCHK)

Ver. Basic | High performance | Process | Redundant | Universal

基本型 QCPU：序列号的前 5 位数为“ 04122 ”以后

(S1) ：存储计测的当前值的软元件 (BIN16 位)。
(S2) ：存储计测的设置值的软元件 (BIN16 位)。
(D) ：时间到时变为 ON 的软元件 (位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
(S1)	--	○		--					--
(S2)	○	○		○					--
(D)	○	--		--					--

功 能

(1) 对条件软元件的 ON 时间进行计测，如果其连续 ON 的时间超过了(S2)中指定的软元件中设置的时间，则将(D)中指定的软元件置为 ON。

(2) (S1)中指定的软元件的当前值的清 0 及(D)中指定的软元件的 OFF 是在执行指令的上升沿时执行。

即使执行指令变为 OFF 后(S1)中指定的软元件的当前值及(D)中指定的软元件的 ON 状态仍将被保持。

(3) 计测的设置值是以 100ms 为单位进行设置的。

出 错

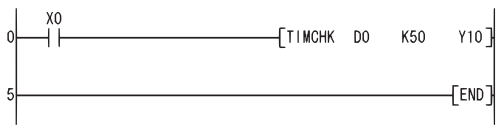
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 指定了不能指定的软元件时。 (出错代码： 4100)

程序示例

(1) 以下为将 X0 的 ON 时间设置为 5 秒，将当前值存储软元件设置为 D0，将时间到时变为 ON 的软元件设置为 Y10 时的程序。

[梯形图模式]

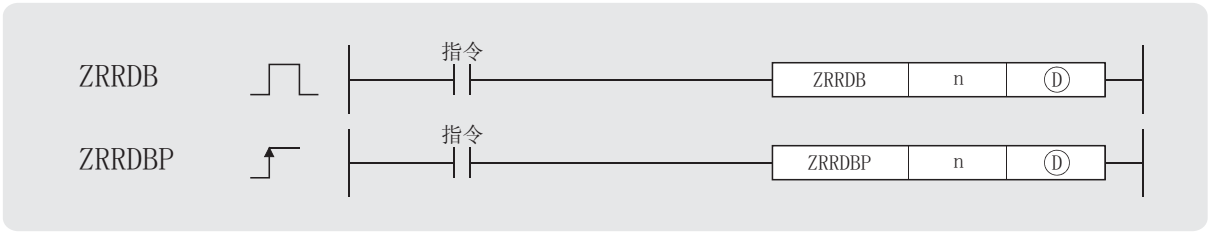


[列表模式]

步	指令	软元件		
0	LD	X0		
1	TIMCHK	D0	K50	Y10
5	END			

7.17.4 文件寄存器的直接 1 字节读取 (ZRRDB(P))

Basic | High performance | Process | Redundant | Universal



n ：读取的文件寄存器的连续字节号 (BIN32 位)。

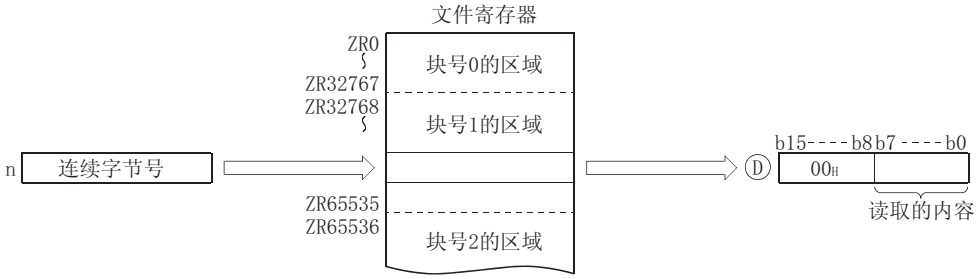
Ⓓ ：存储读取的数据的软元件编号 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			K、H	
n		○						○	--
Ⓓ		○						--	--

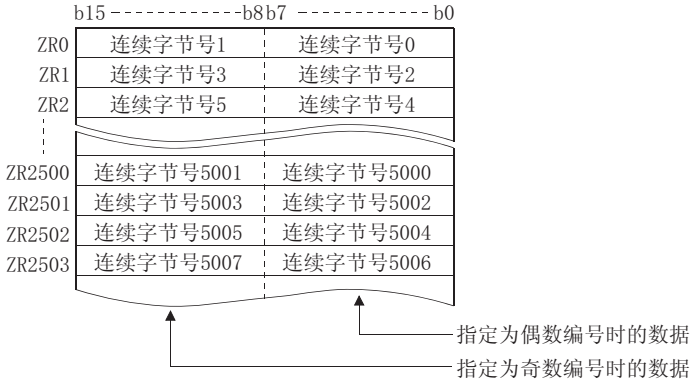
功 能

(1) 在无需理会块号的情况下，读取 n 中指定的连续字节号的文件寄存器的内容，并将其存储到Ⓓ中指定的软元件的低 8 位中。

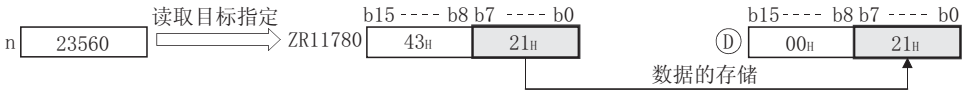
Ⓓ中指定的软元件的高 8 位将变为 00_H。



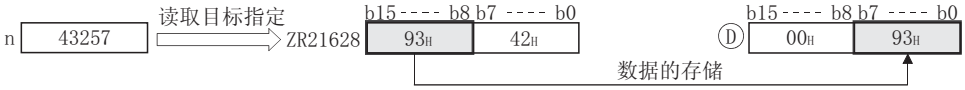
(2) 对应于连续字节号文件寄存器的编号如下所示。



(a) 指定为 n=23560 时，读取 ZR11780 的低 8 位的数据。



(b) 指定为 n=41257 时，读取 ZR21628 的高 8 位的数据。



出　错

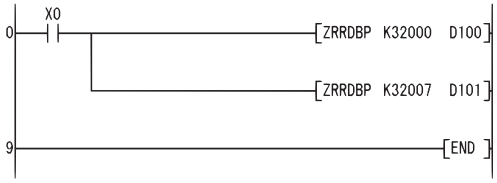
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 指定的软元件编号 (连续字节号) 超出了允许指定范围时。　　(出错代码： 4101)

程序示例

(1) 以下为 X0 变为 ON 时，读取 R16000 的低位及 R16003 的高位的内容后，存储到 D100、D101 中的程序。

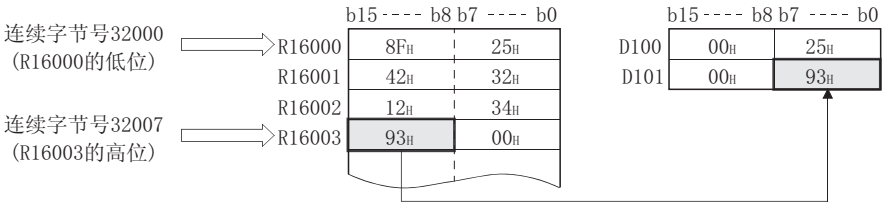
[梯形图模式]



[列表模式]

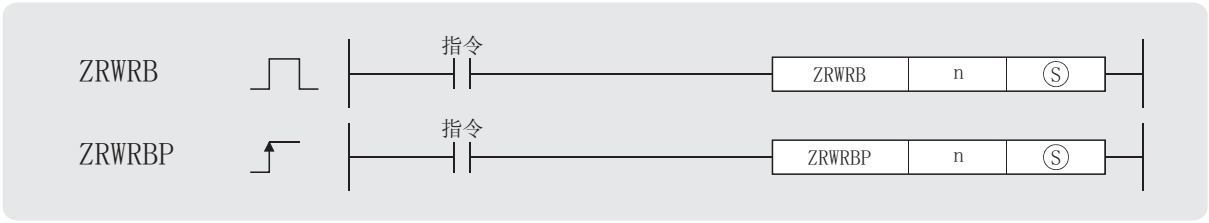
步	指令	软元件	
0	LD	X0	
1	ZRRDBP	K32000	D100
5	ZRRDBP	K32007	D101
9	END		

[动作]



7.17.5 文件寄存器的直接 1 字节写入 (ZRWRB(P))

Basic | High performance | Process | Redundant | Universal



n ：写入的文件寄存器的连续字节号 (BIN32 位)。

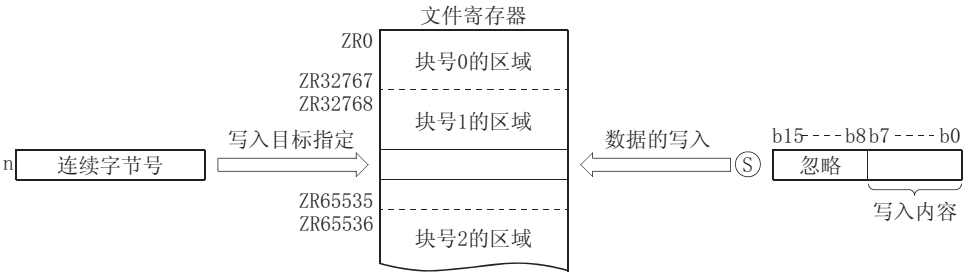
Ⓓ ：存储写入的数据的软元件编号 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
n		○							--
Ⓢ		○							--

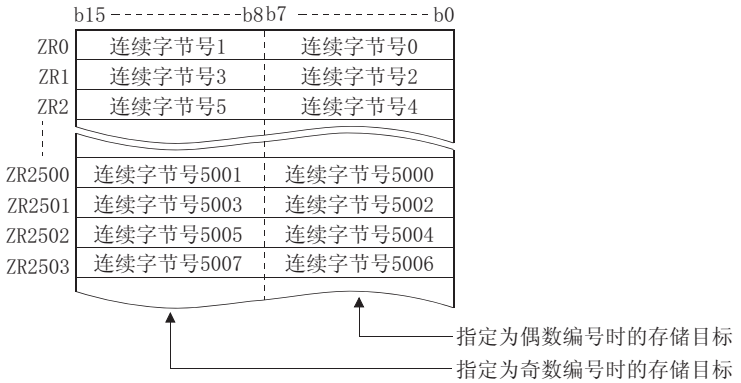
功 能

(1) 在无需理会块号的情况下，将Ⓢ中指定的软元件中存储的低位的内容写入到 n 中指定的连续字节号的文件寄存器中。

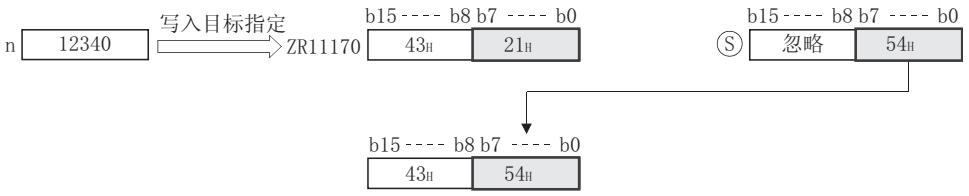
Ⓢ中指定的软元件的高 8 位的数据将被忽略。



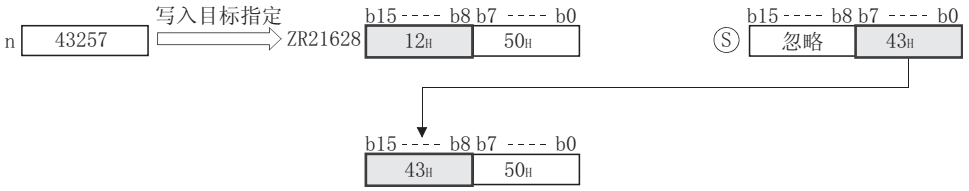
(2) 对应于连续字节号文件寄存器的编号如下所示。



指定为 n=12340 时，写入到 ZR11170 的低 8 位中。



指定为 n=43257 时，写入到 ZR21628 的高 8 位中。



出　错

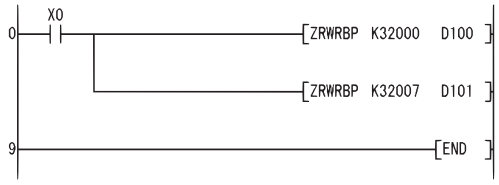
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· 指定的软元件编号（连续字节号）超出了允许指定范围时。　　（出错代码： 4101）

程序示例

(1) 以下为 X0 变为 ON 时，将 D100、D101 中的低位数据写入到 R16000 的低位及 R16003 的高位中的程序。

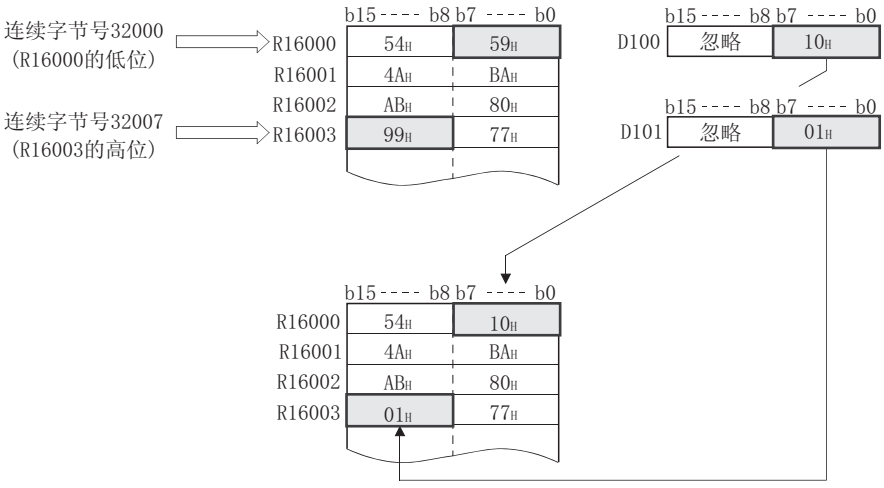
[梯形图模式]



[列表模式]

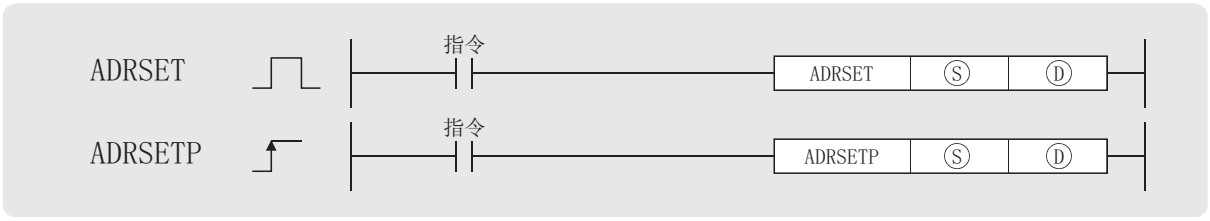
步	指令	软元件	
0	LD	X0	
1	ZRWRBP	K32000	D100
5	ZRWRBP	K32007	D101
9	END		

[动作]



7.17.6 间接地址读取 (ADRSET(P))

Basic | High performance | Process | Redundant | Universal



Ⓢ：读取间接地址的软元件编号（软元件名）。

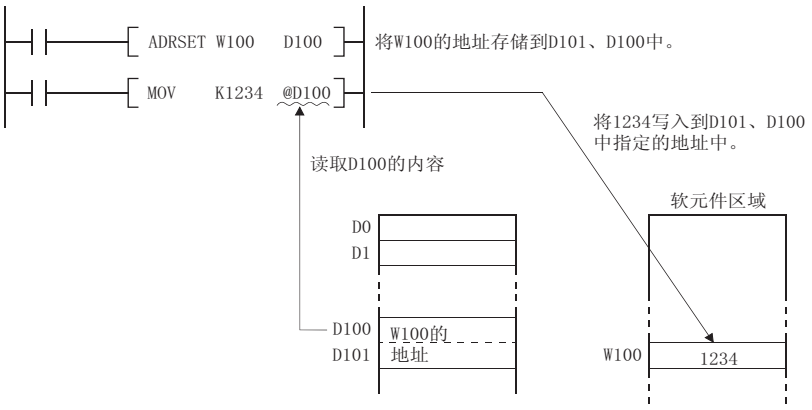
Ⓓ：存储Ⓢ中指定的软元件的间接地址的软元件编号 (BIN32 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
Ⓢ		○			--				
Ⓓ		○			--				

功 能

(1) 将Ⓢ中指定的软元件的间接地址存储到Ⓓ+1、Ⓓ的软元件中。

在顺控程序中执行软元件的间接地址时使用Ⓓ中指定的软元件中存储的地址。



(2) Ⓢ中不能进行位软元件的位数指定。

出 错

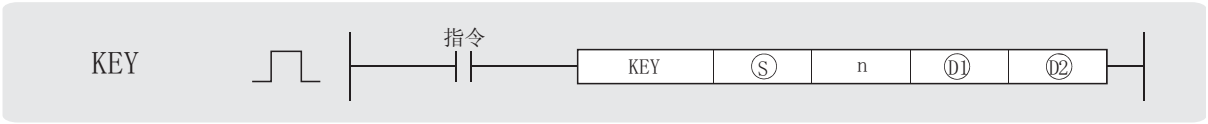
(1) 在 ADRSET(P) 指令中无运算出错。

备 注

关于间接指定，清参阅 3.4 节。

7.17.7 键盘的数字键输入 (KEY)

Basic　High performance　Process　Redundant　Universal

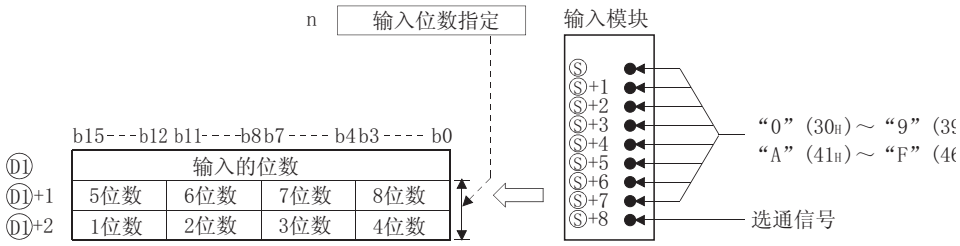


Ⓢ ：进行数字输入 (X) 的软元件的起始编号 (位)。
n ：进行数字输入的位数 (BIN16 位)。
Ⓓ1 ：存储输入的数字的软元件的起始编号 (BIN16 位)。
Ⓓ2 ：输入结束时置为 ON 的位软元件编号 (位)。

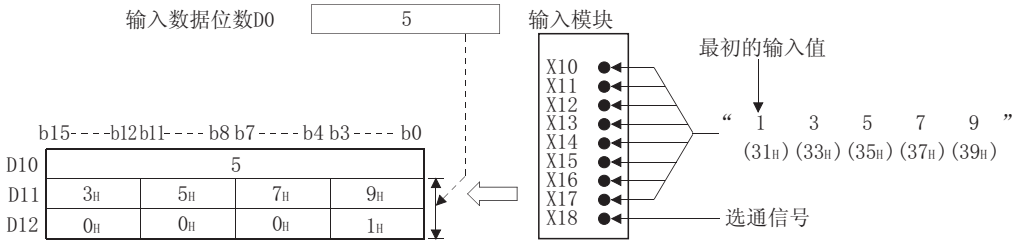
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○(仅X)	--		--			--		--
n	○	○		○			○		--
Ⓓ1	--	○		--			--		--
Ⓓ2	○	○		○			--		--

功　能

(1) 从Ⓢ中指定的输入 (X) 中读取 8 点的 ASCII 数据，转换为 16 进制数值后存储到Ⓓ1中指定的软元件后面。

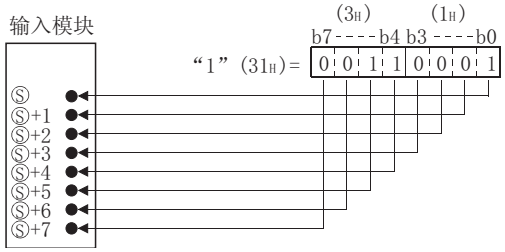


例如，将输入数据的位数 (n) 设置为 5，在输入模块的 X10 ～ X18 中分别输入了“ 31 ”、“ 33 ”、“ 35 ”、“ 37 ”、“ 39 ”时的情况如下所示。



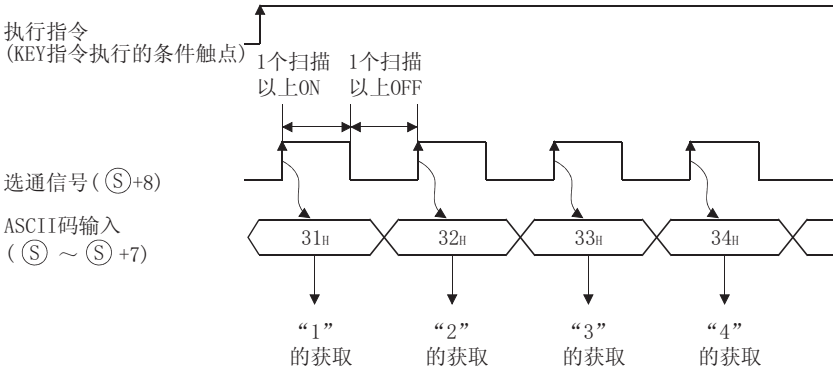
(2) 对Ⓢ中指定的输入 (X) 进行数字输入时，将与数字对应的 ASCII 码执行位展开到Ⓢ～Ⓢ+7 中进行输入。

可输入的 ASCII 码的范围为 30H(0) ～ 39H(9) 以及 41H(A) ～ 46H(F)。



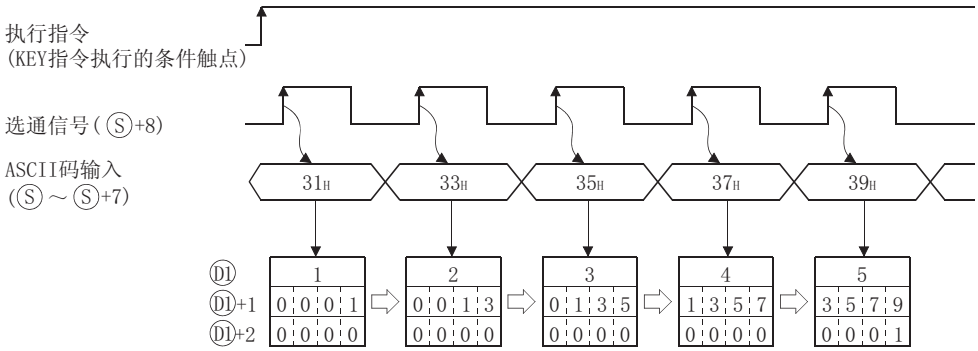
(3) 将 ASCII 码输入到Ⓢ～Ⓢ+7 中后，通过将Ⓢ+8 的选通信号置为 ON，将指定的数字读取到内部。

选通信号的 ON/OFF 状态应保持在顺控程序的 1 个扫描以上。
如果保持在 1 个扫描以下，有可能无法正常地获取数据。



(4) 执行指令 (KEY 指令执行的条件触点) 必须预先置为 ON，直至指定位数的输入结束为止。
如果执行指令为 OFF 则无法执行 KEY 指令。

(5) 存储到Ⓓ1 中指定的软元件中时，将实际获取的数字的位数存储到Ⓓ1 中，将输入的 ASCII 码转换为 16 进制 BIN 值后存储到Ⓓ1+1，Ⓓ1+2 中。



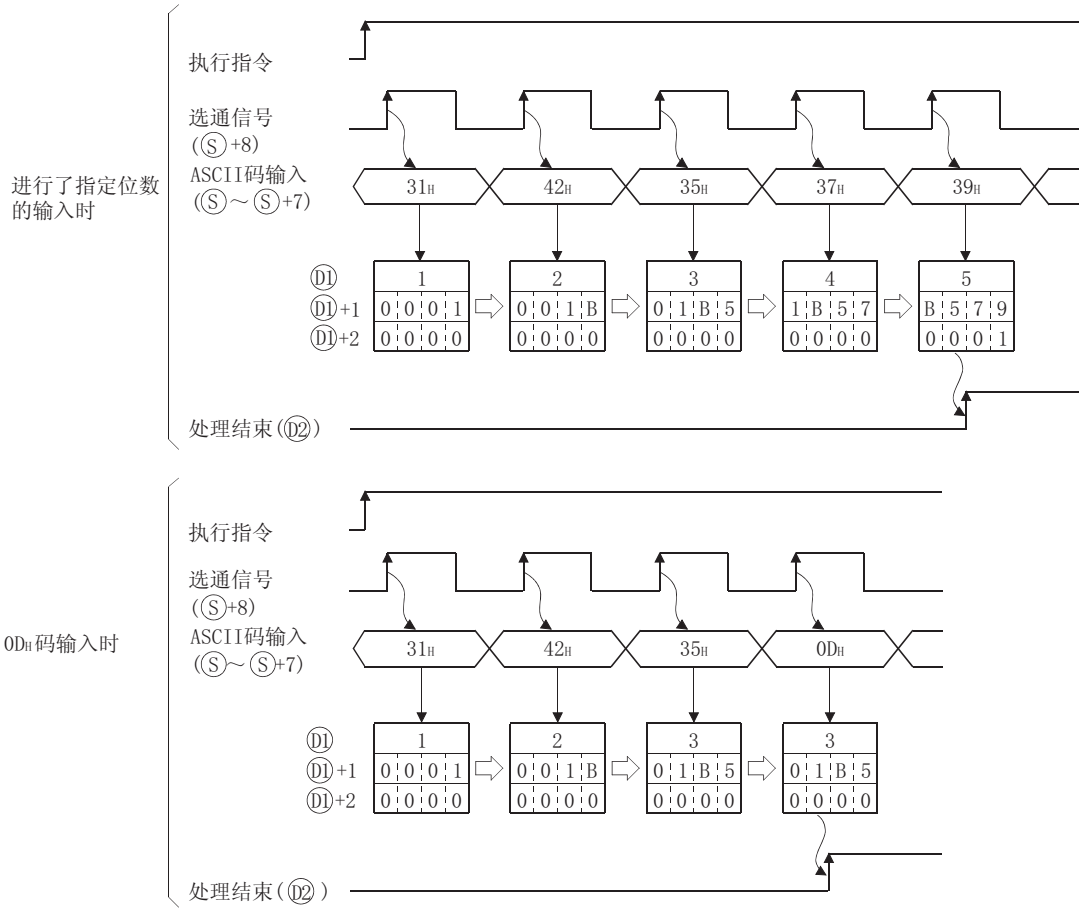
(6) n 中指定的输入位数范围为 1 ～ 8。

(7) 进行输入数据的内部获取时，在进行了下述输入时结束，并将Ⓓ2中指定的位软元件置为 ON。

· 进行了 n 中指定的位数的输入时

· 输入了 “ 0DH ”码时

例如，指定了 n=5 时的动作如下所示。



希望再次进行输入处理的情况下，需要通过用户程序对Ⓓ1中存储的输入位数、输入数据进行清除以及将指定位软元件置为 OFF。

如果未进行Ⓓ1的清除以及Ⓓ2的 OFF，将无法进行下一个输入处理。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

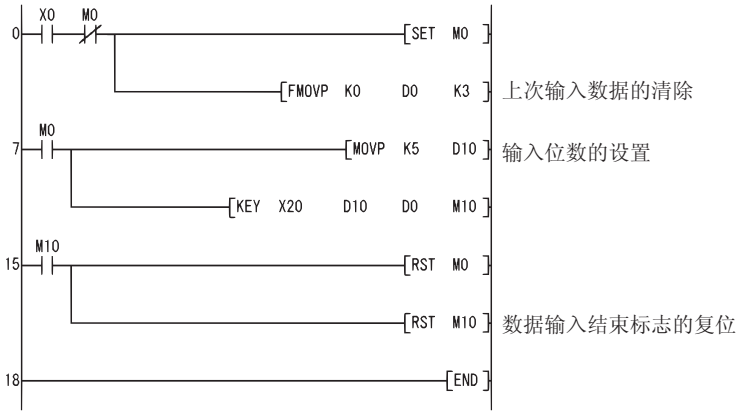
· Ⓢ中指定的软元件不是输入 (X) 时。 (出错代码： 4100)

· n 中指定的位数超出了 1 ～ 8 的范围时。 (出错代码： 4100)

程序示例

(1) 以下为 X0 变为 ON 时，通过 X20 ～ X28 连接的数字键获取 5 位数以内的数据后，存储到 D0 后面的程序。

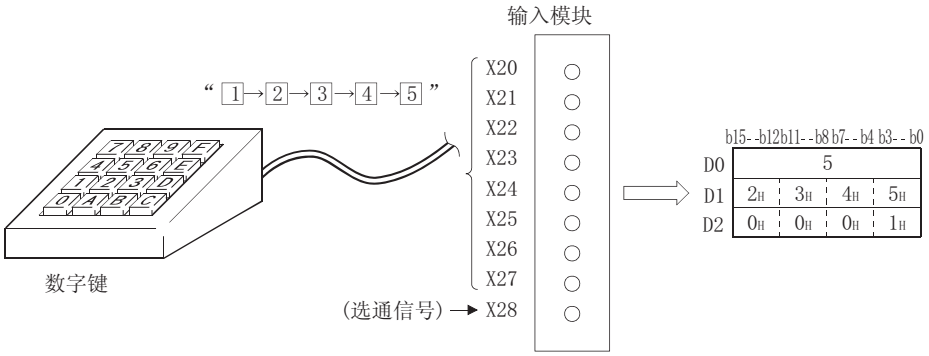
[梯形图模式]



[列表模式]

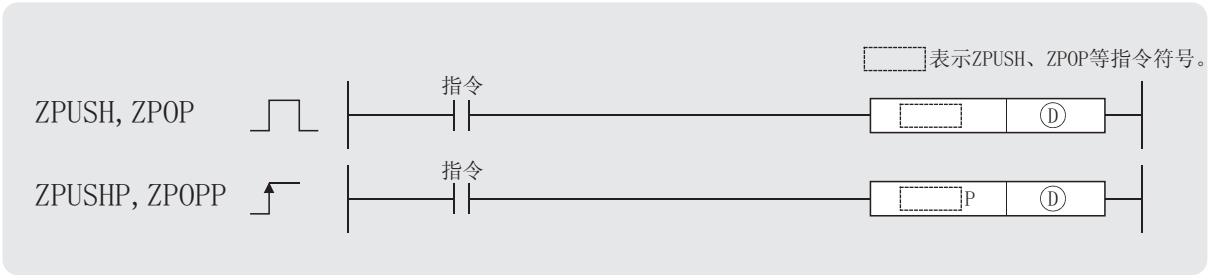
步	指令	软元件		
0	LD	X0		
1	ANI	M0		
2	SET	M0		
3	FMOVP	K0	D0	K3
7	LD	M0		
8	MOVP	K5	D10	
10	KEY	X20	D10	D0 M10
15	LD	M10		
16	RST	M0		
17	RST	M10		
18	END			

[动作]



7.17.8 变址寄存器的批量保存、恢复 (ZPUSH(P)、ZPOP(P))

Basic | High performance | Process | Redundant | Universal



Ⓓ ：进行变址寄存器的保存 / 恢复的软元件的起始编号 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
Ⓓ	--	○		--					

功 能

ZPUSH

(1) 将下述变址寄存器的内容保存到Ⓓ中指定的软元件后面。

(如果对变址寄存器的内容进行保存则Ⓓ+0(保存次数) 将被 +1。)

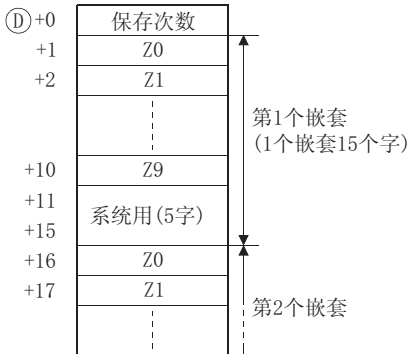
- 基本型 QCPU ..... Z0 ～ Z9
- 高性能型 QCPU、过程 CPU、冗余 CPU ..... Z0 ～ Z15
- 通用型 QCPU ..... Z0 ～ Z19

(2) 进行数据的恢复时，使用 ZPOP 指令。通过成对使用 ZPUSH 与 ZPOP 指令，可以进行嵌套。

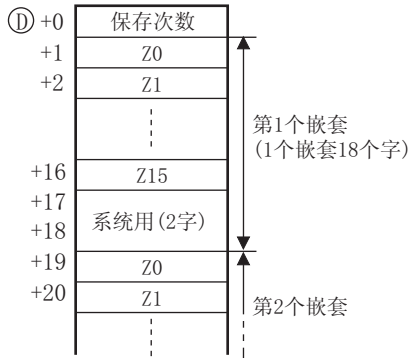
(3) 进行了嵌套时，由于每执行一次 ZPUSH 指令后，将在Ⓓ的后面增加使用的区域，因此应预先预留好与嵌套使用次数相对应的区域。

(4) Ⓓ后面使用的区域的构成如下所示。

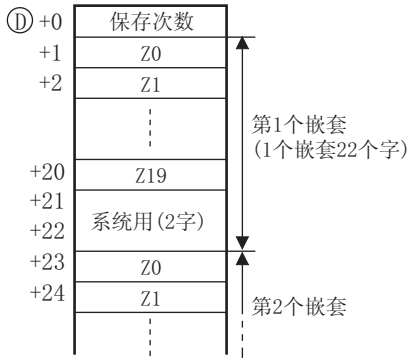
· 使用基本型 QCPU 时



· 使用高性能型 QCPU/ 过程 CPU/ 冗余 CPU 时



· 使用通用型 QCPU 时



ZPOP

(1) 将Ⓓ中指定的软元件后面保存数据读取到变址寄存器中。(对保存的变址寄存器的内容进行读取时则Ⓓ+0(保存次数) 将被 -1。)

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

· ZPUSH(P) 指令中使用的Ⓓ后面的点数范围超出了相应软元件的范围时。
(出错代码： 4101)

· ZPOP(P) 指令中Ⓓ+0 的内容 (保存次数) 为 0 时。 (出错代码： 4100)

索引

索

索

索

索

索

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。

QCPU
编程手册

公共指令篇1/2

三菱电机自动化
400-821-3030
CALL CENTER 技术支持热线
周一至周五 9:00~17:00(法定节假日除外)

三菱电机自动化(上海)有限公司

地址：上海市黄浦区南京西路288号创兴金融中心17楼
邮编：200003
电话：021-23223030　传真：021-23223000
网址：www.meas.cn

书号	SH(NA)-080814CHN(1/2)-A(0903)STC
印号	STC-QCPU-CI(1/2)-PM(0903)

MITSUBISHI

三菱可编程控制器

MELSEC Q 系列

QCPU
编程手册
公共指令篇2/2

● 安全注意事项 ●

（使用之前请务必阅读）

在使用 MELSEC-Q 系列可编程控制器之前，应仔细阅读各产品附带的手册以及附带手册中介绍的关联手册，同时在充分注意安全的前提下正确地操作。

请妥善保管产品附带手册以备需要时阅读，并应将本手册交给最终用户。

修订记录

※本手册号在封底的左下角。

印刷日期	※手册编号	修订记录
2009 年 03 月	SH(NA)-080814CHN-A	第一版

日文手册原稿： SH-080804-A

前言

本手册 “QCPU 编程手册（公共指令篇）”介绍进行 QCPU 编程时需要的公共指令有关内容。

• 公共指令是指，AJ71QC24、AJ71PT32-S3 等特殊功能模块专用指令、AD57 专用指令、PID 控制专用指令、SFC 专用指令、除 ST 专用指令以外的其它指令。

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

■对象 CPU 模块

CPU 模块	型号
基本型 QCPU	Q00JCPU、Q00CPU、Q01CPU
高性能型 QCPU	Q02CPU、Q02HCPU、Q06HCPU、Q12HCPU、Q25HCPU
过程 CPU	Q02PHCPU、Q06PHCPU、Q12PHCPU、Q25PHCPU
冗余 CPU	Q12PRHCPU、Q25PRHCPU
通用型 QCPU	Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU、Q03UDCPU、Q04UDHCPU、Q06UDHCPU、Q10UDHCPU、Q13UDHCPU、Q20UDHCPU、Q26UDHCPU、Q03UDECPU、Q04UDEHCPU、Q06UDEHCPU、Q10UDEHCPU、Q13UDEHCPU、Q20UDEHCPU、Q26UDEHCPU

目录

公共指令篇 1/2

公共指令篇 2/2

手册体系

在基本手册中介绍基本的规格、功能、使用方法有关内容。
其它手册介绍相应的 CPU 模块及功能。
请根据需要参考下表订购各手册。

“CPU 模块”栏中所示的编号及 CPU 模块的对应如下所示：

编号	CPU 模块
1)	基本型 QCPU
2)	高性能型 QCPU
3)	过程 CPU
4)	冗余 CPU
5)	通用型 QCPU

●：基本手册　○：使用相应 CPU 模块 / 功能时参阅。

手册名称 <手册编号>	记载内容	CPU 模块				
		1)	2)	3)	4)	5)
■用户手册						
QCPU 用户手册（硬件设计 / 维护点检篇） <SH-080501CHN>	介绍 CPU 模块、电源模块、基板、扩展电缆及存储卡等的硬件规格及系统维护・点检、故障排除、出错代码等。	●	●	●	●	●
QnUCPU 用户手册（功能解说 / 程序基础篇） <SH-080812CHN>	介绍创建程序所需的功能、编程方法和软元件等。					●
Qn(H)/QnPH/QnPRHCPU 用户手册（功能解说 / 程序基础篇） <SH-080803>	介绍创建程序所需的功能、编程方法和软元件等。	●	●	●	●	
QCPU 用户手册（内置以太网板通信篇） <SH-080813CHN>	介绍 CPU 内置以太网板通信的功能有关内容。					○
■编程手册						
QCPU 编程手册（公共指令篇） <SH-080814CHN>	介绍顺控程序指令、基本指令以及应用指令等的使用方法。	●	●	●	●	●
QCPU(Q 模式)/QnACPU 编程手册 (SFC 篇) <SH-080283CHN>	介绍 MELSAP3 格式的 SFC 程序的创建所需的系统配置、性能规格、功能、编程、调试和出错代码等。	○	○	○	○	○
QCPU(Q 模式）编程手册 (MELSAP-L 篇) <SH-080076>	介绍 MELSAP-L 格式的 SFC 程序的创建所需的编程方法、规格、功能等。	○	○	○	○	○
QCPU(Q 模式）编程手册（结构化文本篇） <SH-080363>	介绍结构化文本语言的编程方法。	○	○	○	○	○
QCPU(Q 模式)/QnACPU 编程手册 (PID 控制指令篇) <SH-080240C>	介绍用于执行 PID 控制的专用指令。	○	○		○	○
QnPHCPU/QnPRHCPU 编程手册（过程控制指令篇） <SH-080449CHN>	介绍用于执行过程控制的专用指令。			○	○	

关联手册

手册名称 <手册编号>	记载内容
CC-Link IE 控制网络参考手册 <SH-080649>	介绍 CC-Link IE 控制网络的控制网络规格、投行前的设置及步骤、参数设置、编程及故障排除等。
Q 系列 MELSECNET/H 网络系统参考手册（可编程控制器网络篇） <SH-080289C>	介绍 MELSECNET/H 网络系统的可编程控制器网络的规格、投运前的设置及步骤、参数设置、编程及故障排除等。
Q 系列 MELSECNET/H 网络系统参考手册（远程 I/O 网络篇） <SH-080290C>	介绍 MELSECNET/H 网络系统的远程 I/O 网络的规格、投运前的设置及步骤、参数设置、编程及故障排除等。
MELSECNET、MELSECNET/B 数据链接系统参考手册 <IB-68277>	介绍 MELSECNET(II) 和 MELSECNET/B 的概要、规格和各部位名称及设置等。
Q 系列以太网接口模块用户手册（应用篇） <SH-080285C>	介绍以太网模块的电子邮件功能、可编程控制器 CPU 的状态监视、经由 LSECNET/H 或 MELSECNET/10 网络的通信功能、通过数据链接用指令的通信功能、文件传输 (FTP 服务器）等功能。

备忘录

8 数据链接指令

分类	处理内容	参阅章节
网络刷新指令	进行指定网络模块的刷新处理。	8.1 节
路由信息读取 / 写入指令	路由参数中设置的数据的读取。	8.2 节
	将路由数据写入到路由参数中指定的区域中。	

备 注

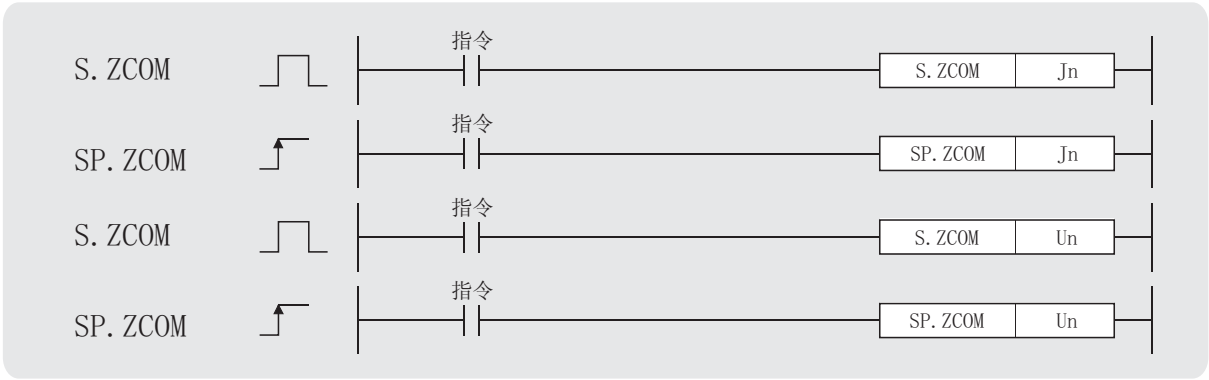
对于本章中记述的各指令的名称，在未特别指定的情况下，将按下述方式进行省略。

- S(P)/J(P)/G(P).ZCOM → ZCOM
- S(P)/Z(P).RTREAD → RTREAD
- S(P)/Z(P).RTWRITE → RTWRITE

8.1 网络刷新指令

8.1.1 对指定模块的刷新指令 (S(P)/J(P)/G(P).ZCOM)

Basic | High performance | Process | Redundant | Universal



Jn ：自站网络号 (BIN16 位)。
Un ：自站网络模块的起始 I/O 编号 (BIN16 位)。

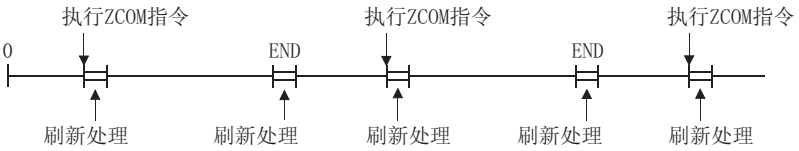
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
--	--								

ZCOM 指令用于在顺控程序的执行过程中，在任意的时机进行刷新。
通过 ZCOM 指令进行刷新的对象如下所示。

- CC-Link IE 控制网络的刷新（刷新参数设置时）
- MELSECNET/H 的刷新（刷新参数设置时）
- CC-Link 的自动刷新（刷新参数设置时）
- 智能功能模块的自动刷新（刷新参数设置时）

功 能

(1) 执行 ZCOM 指令时，CPU 模块将暂时中断顺控程序的处理，进行 Jn/Un 中指定的网络模块的刷新处理。



(2) 在 ZCOM 指令中，不执行下述处理。

(a) CPU 模块与编程工具的通信处理。

(b) 其它站监视处理。

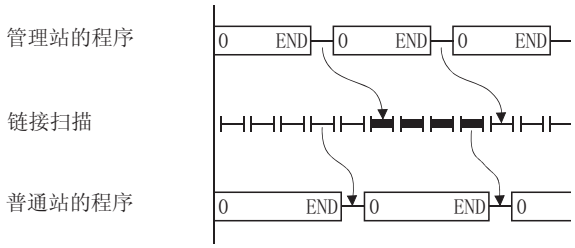
(c) 在串行通信模块中，对其它智能功能模块的缓冲存储器的读取处理。

(d) MELSECNET/H 的低速循环传送数据。

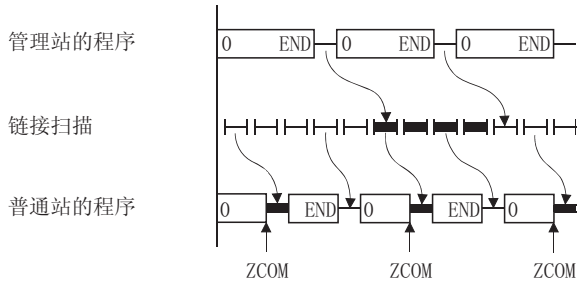
(3) 对于可编程控制器网络 *1

(a) 自站的顺控程序的扫描时间长于其它站的扫描时间时，为了切实地执行从其它站的数据获取，可以使用 ZCOM 指令。

1) 未使用 ZCOM 指令时的数据发送接收示例



2) 使用了 ZCOM 指令时的数据发送接收示例

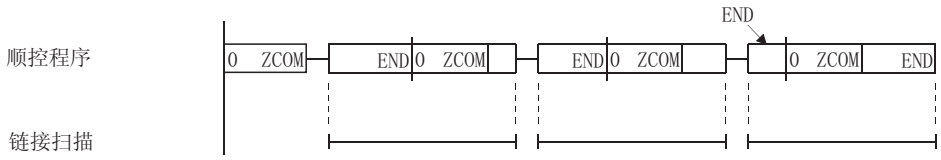


关于可编程控制器网络的传送延迟时间的详细内容，请参阅以下手册。

• CC-Link IE 控制网络参考手册

• Q 系列 MELSECNET/H 网络系统参考手册（可编程控制器网络篇）

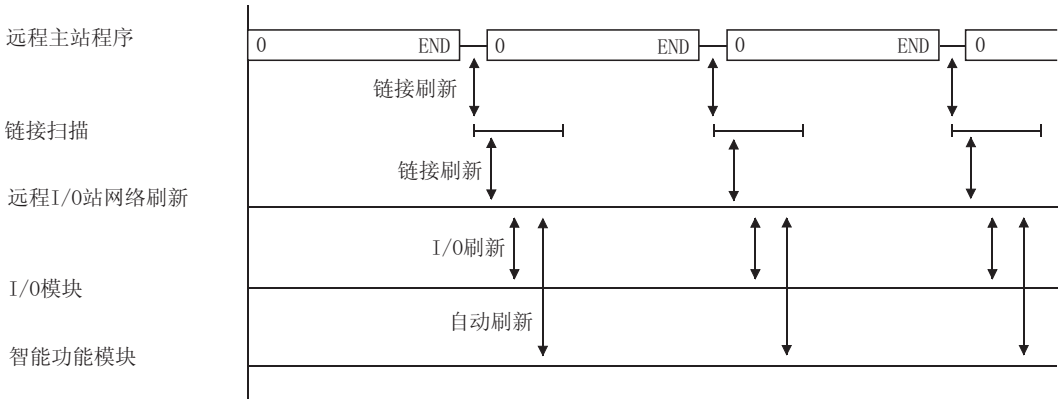
(b) 在链接扫描时间长于顺控程序的扫描时间的情况下，即使使用 ZCOM 指令也不能提高数据的发送接收速度。



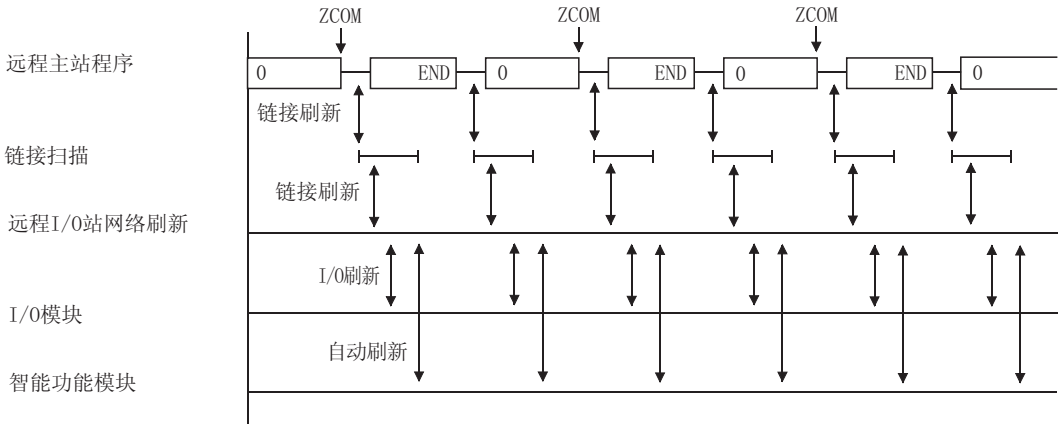
(4) 对于远程 I/O 网络

远程主站的链接刷新是在 CPU 模块的 “END 处理”时进行的。
由于链接刷新结束时进行扫描，因此链接扫描与 CPU 模块的程序 “同步”。
在远程主站中使用 ZCOM 指令时，将在 ZCOM 指令执行的时点进行链接刷新，链接刷新结束时进行链接扫描。
因此，如果在远程主站中使用 ZCOM 指令，可以提高与远程 I/O 站的发送接收处理速度。

1) 未使用 ZCOM 指令时



2) 使用了 ZCOM 指令时



关于远程 I/O 网络的传送延迟时间的详细内容，请参阅以下手册。

• Q 系列 MELSECNET/H 网络系统参考手册（可编程控制器网络篇）

(5) ZCOM 指令在顺控程序中的使用次数无限制。

但是，顺控程序的扫描时间将会有相当于刷新时间的延迟，因此应加以注意。

(6) 通过将 Un 指定为变量，不仅可将网络模块指定为对象，也可将智能功能模块指定为对象。
此时，进行智能功能模块的缓冲存储器的自动刷新。（变为 FROM/TO 指令的替代指令。）

要 点

1. ZCOM 指令不能用于恒定周期执行型程序、中断程序。
2. 在冗余 CPU 中，使用 ZCOM 指令时是有限制的。
有关详细内容请参阅下述手册。
• QnPRHCPU 用户手册（冗余系统篇）

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。
- 指定的网络号未与自站相连接时。（出错代码：4102）
- 起始 I/O 编号中指定的模块不是网络模块 / 链接模块时。（除通用型 QCPU 以外）（出错代码：2111）
- 起始 I/O 编号中指定的模块不是网络模块 / 链接模块时。（仅通用型 QCPU）（出错代码：4102）

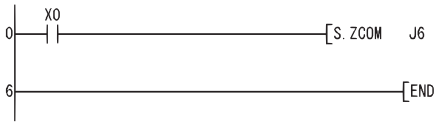
要 点

希望仅与外围设备进行通信时，应使用 COM 指令（参阅 7.6.9 项、9.13 节）

程序示例

(1) 以下为 X0 变为 ON 时，对网络号 6 的网络模块进行链接刷新的程序。

[梯形图模式]

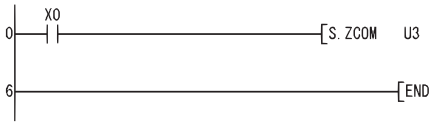


[列表模式]

步	指令	软元件
0	LD	X0
1	S.ZCOM	J6
6	END	

(2) 以下为 X0 变为 ON 时，对安装在起始 I/O 编号为 X/Y30 ～ X/Y4F 位置上的网络模块进行链接刷新的程序。

[梯形图模式]



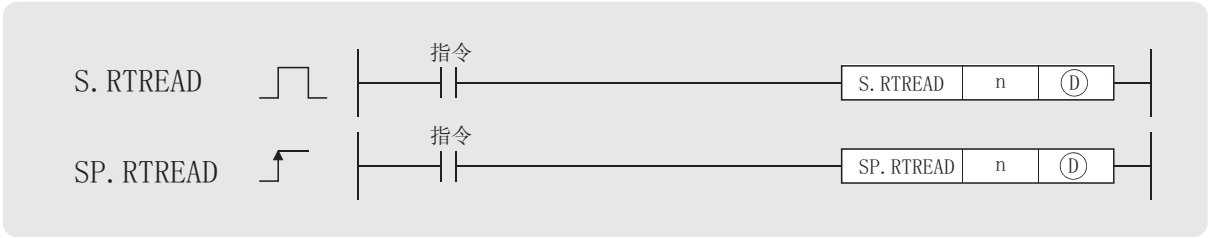
[列表模式]

步	指令	软元件
0	LD	X0
1	S.ZCOM	U3
6	END	

8.2 路由信息的读取 / 写入

8.2.1 路由信息的读取 (S(P)/Z(P).RTREAD)

Basic High performance Process Redundant Universal



n ：传送目标网络号 (1 ～ 239)(BIN16 位)。

Ⓓ ：存储读取的数据的软元件的起始编号 (软元件名)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
n	○	○		--				○	--
Ⓓ	--	○		--				--	--

功 能

(1) 根据路由参数中设置的路由信息，对 n 中指定的传送目标网络号的数据进行读取后，存储到Ⓓ的后面。

(2) n 中指定的传送目标网络号的数据未在路由参数中进行设置时，在Ⓓ的后面将存储 0。

(3) Ⓓ后面存储的数据内容如下所示。

(各数据的范围)

Ⓓ+0	中继目标网络号	(1～239)
+1	中继目标站号	(1～120)
+2	虚拟	

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

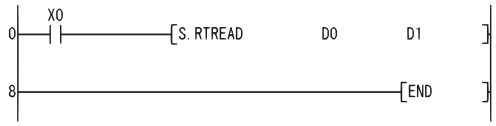
- n 的数据超出了 1 ～ 239 的范围时。 (出错代码 : 4100)
- Ⓓ中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)

(出错代码 : 4101)

程序示例

(1) 以下为 X0 变为 ON 时，对 D0 中指定的网络号的路由信息进行读取的程序。

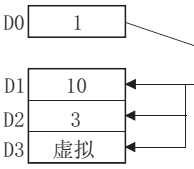
［梯形图模式］



［列表模式］

步	指令	软元件	
0	LD	X0	
1	S.RTREAD	D0	D1
8	END		

［动作］

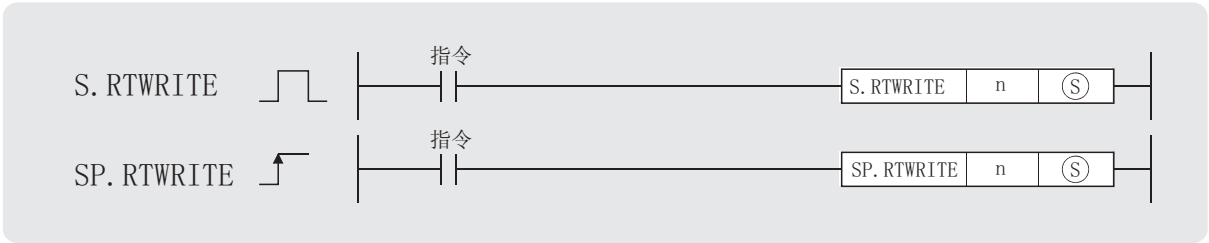


［路由参数的设置内容］

传送目标网络号	中继目标网络号	中继目标站号
1	10	3
2	10	2
3	10	1

8.2.2 路由信息的登录 (S(P)/Z(P).RTWRITE)

Basic High performance Process Redundant Universal



n ：传送目标网络号 (1 ～ 239)(BIN16 位)。

Ⓢ ：存储写入数据的软元件的起始编号 (软元件名)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
n	○	○		--				○	--
Ⓢ	--	○		--				--	--

功 能

(1) 将Ⓢ后面的路由数据登录到路由参数的 n 中指定的传送目标网络号的区域中。

(2) Ⓢ后面设置的数据内容如下所示。

		(各数据的范围)
Ⓢ+0	中继目标网络号	(0～239)
+1	中继目标站号	(0～120)
+2	虚拟	

(3) n 中指定的传送目标网络号的数据未在路由参数中进行设置时，将被变更为Ⓢ后面的数据。

(4) Ⓢ后面的数据 (Ⓢ+0 ～Ⓢ+2) 全部为 0 时，将 n 中指定的传送目标网络号的数据从路由参数中删除。

出 错

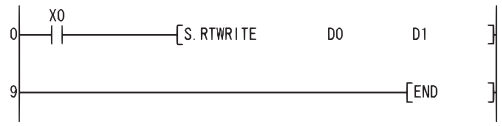
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- n 的数据超出了 1 ～ 239 的范围时。 (出错代码： 4100)
- ⓢ后面的数据超出了各设置范围时。 (出错代码： 4100)
- 网络参数的路由参数中登录的路由信息的登录数与 RTWRITE 指令中登录的路由信息的登录数的合计超过了 64 时。 (出错代码： 4100)
- ⓢ中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)
  (出错代码： 4101)

程序示例

(1) 以下为 X0 变为 ON 时，将 D1 ～ D3 中指定的路由信息写入到 D0 中指定的网络号的网络模块中的程序。

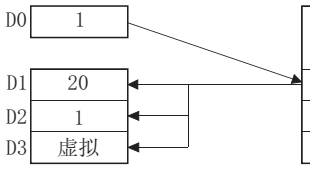
[梯形图模式]



[列表模式]

步	指令	软元件	
0	LD	X0	
1	S.RTWRITE	D0	D1
9	END		

[动作]



[路由参数的设置内容]

传送目标网络号	中继目标网络号	中继目标站号
1	20	1
2	10	2
3	10	1

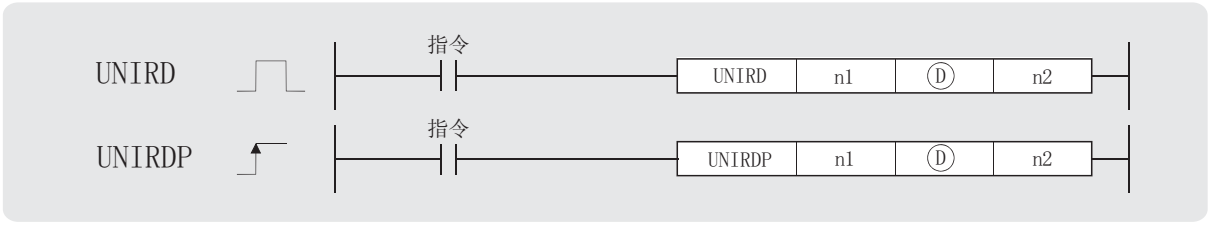
备忘录

9 QCPU 指令

分类	处理内容	参阅章节
模块信息读取	读取指定模块状态的详细信息。	9.1 节
跟踪设置、复位	进行采样跟踪的触发及采样跟踪的复位。	9.2 节
至指定文件的数据写入	对指定的文件进行数据写入。	9.3 节
从指定文件中读取数据	对指定的文件进行数据读取。	9.4 节
至标准 ROM 的数据写入	对标准 ROM 的软元件数据存储用文件进行数据写入。	9.5 节
从标准 ROM 中读取数据	对标准 ROM 的软元件数据存储用文件进行数据读取。	9.6 节
通过存储卡进行的程序装载	将存储卡、标准 ROM 的程序传送到程序存储器中并置于待机状态。	9.7 节
从程序存储器中卸载程序	将程序存储器的待机程序删除。	9.8 节
装载 + 卸载	从程序存储器中卸载程序，通过存储卡进行程序装载。	9.9 节
文件寄存器高速块传送	将指定的数据进行高速批量传送。	9.10 节
至自站 CPU 共享内存的写入	将自站的软元件写入到自站 CPU 模块的 CPU 共享内存中。	9.11 节
从其它站 CPU 共享内存中读取	从其它站 CPU 模块的 CPU 共享内存中，将软元件读取到自站中。	9.12 节
选择刷新指令	对 CPU 模块的共享内存进行自动刷新。	9.13 节
扩展时钟指令	对扩展时钟数据进行读取、加减法运算。	9.15 节

9.1 模块信息读取 (UNIRD(P))

Basic | High performance | Process | Redundant | Universal



n1 ：将模块信息读取源起始 I/O 编号用 16 相除后的值 (0 ～ FFn) (BIN16 位)。

Ⓓ ：存储模块信息的软元件的起始编号 (软元件名)。

n2 ：读取数据点数 (0 ～ 256) (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
n1	○	○		--				○	--
Ⓓ	--	○		--				--	--
n2	○	○		--				○	--

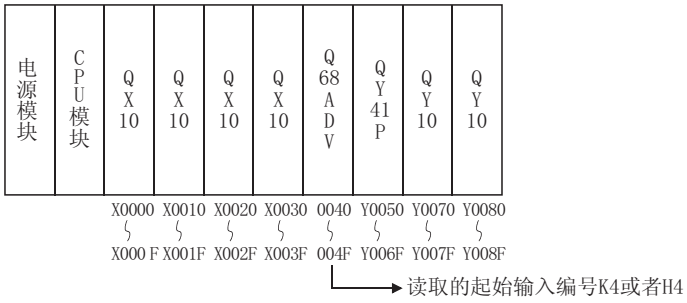
功 能

(1) 从 n1(将起始 I/O 编号用 16 相除后的值) 中指定的模块中，将 n2 中指定点数的模块信息存储到Ⓓ中指定的软元件后面。

(所读取的模块信息不是 I/O 分配中指定的模块类型，而是实际安装的模块状态。)

备 注

进行 n1 指定时，是将进行模块信息读取的插槽的起始 I/O 编号以 4 位的 16 进制数表示时的高 3 位进行指定。



模块信息的详细情况如下所示。

位	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
各模块信息																

位	项目名	内容	
b0	I/O 点数	000:16 点	001:32 点
b1		010:48 点	011:64 点
b2		100:128 点	101:256 点
		110:512 点	111:1024 点
b3	模块类型	000：输入模块　001：输出模块	
b4		010：输入输出混合模块	
b5		011：智能功能模块	
b6	外部供应电源状态（用于将来扩展）	1：处于有外部电源供应状态	0：无外部电源供应
b7	是否发生保险丝熔断	1：有保险丝熔断的模块	0：正常
b8	在线模块更换状态 / 通过待机系统执行	1：处于正在进行在线模块更换状态，或者试图从冗余系统的待机系统读取扩展基板上的模块信息。*1 0：除上述状态以外	
b9	轻・中度出错状态	1：有轻・中度出错	0：正常
b10	模块出错状态	00：无模块出错	01：轻度出错
b11		10：中度出错	11：严重出错
b12	模块准备状态	1：正常	0：有模块出错
b13	空闲	0：固定	
b14	Q 模块	0：Q 系列模块	
b15	模块安装状态	1：安装了模块	0：未安装模块

*1 ： 对于在多 CPU 系统中使用的通用型 QCPU，即使其它站管理的模块处于在线模块更换状态时也仍将继续保持为 ON 状态。

出　错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

[高性能型 QCPU、过程 CPU、冗余 CPU、通用型 QCPU]

- n1 超出了 0 ～ FFH 的范围时。（出错代码：4100）
- n2 超出了 0 ～ 256 的范围时。（出错代码：4100）
- n1 与 n2 的合计为 257 以上时。（出错代码：4100）

[Q00/Q01CPU]

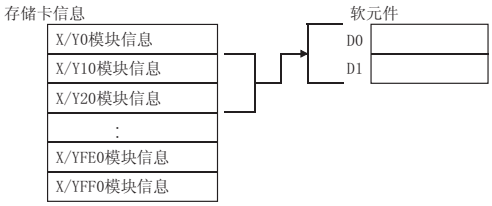
- n1 超出了 0 ～ 3FH 的范围时。（出错代码：4100）
- n2 超出了 0 ～ 64 的范围时。（出错代码：4100）
- n1 与 n2 的合计为 65 以上时。（出错代码：4100）

[Q00JCPU]

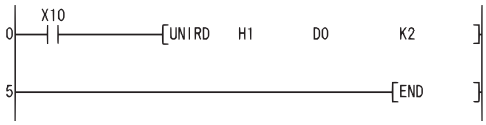
- n1 超出了 0 ～ FH 的范围时。（出错代码：4100）
- n2 超出了 0 ～ 16 的范围时。（出错代码：4100）
- n1 与 n2 的合计为 17 以上时。（出错代码：4100）

程序示例

(1) 以下为 X10 变为 ON 时，将 I/O 编号为 10H ～ 20H 的模块信息存储到 D0 后面的程序。



[梯形图模式]

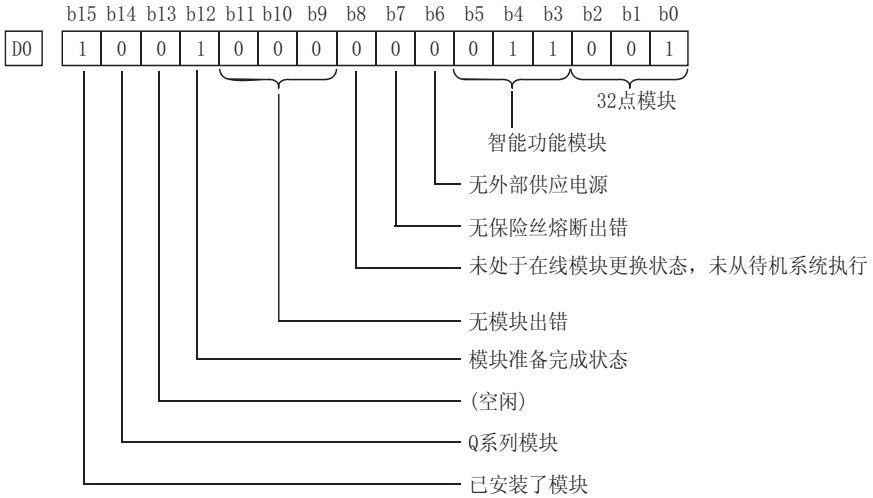


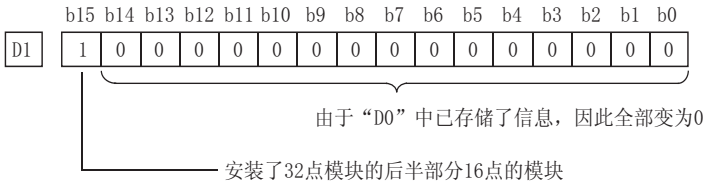
[列表模式]

步	指令	软元件		
0	LD	X10		
1	UNIRD	H1	D0	K2
5	END			

读取结果侧（读取到 D0 时）

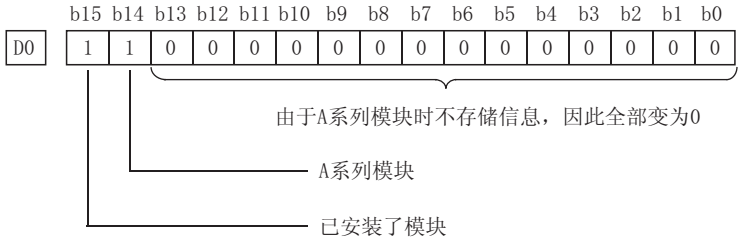
1)Q 系列 32 点智能功能模块的情况下

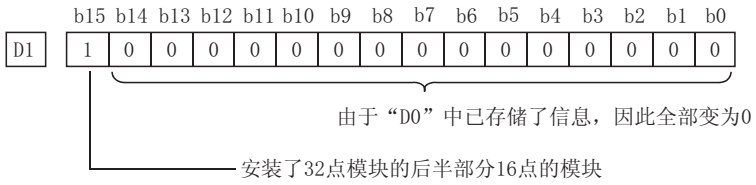




- 48 点模块时在 D2 中存储与 D1 相同的内容，64 点模块时在 D2 及 D3 中分别存储与 D1 相同的内容。

2)A 系列 32 点模块的情况下

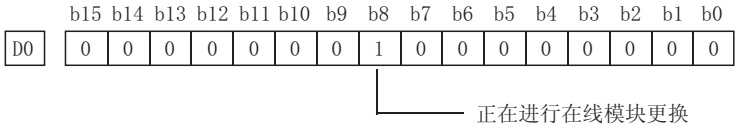




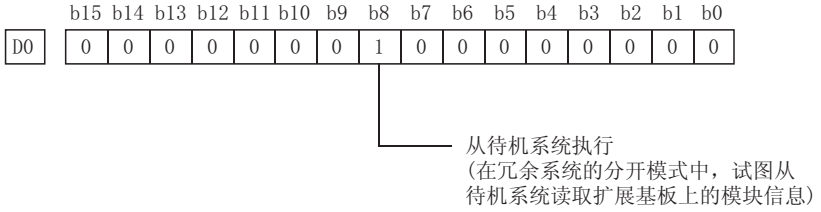
- 48 点模块时在 D2 中存储与 D1 相同的内容，64 点模块时在 D2 及 D3 中分别存储与 D1 相同的内容。

3) 空插槽的情况下

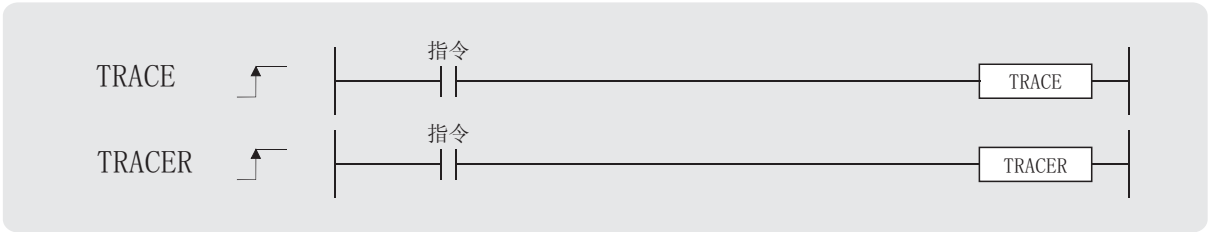
4) 处于在线模块更换状态的情况下



5) 在冗余系统的分开模式中，试图从待机系统读取扩展基板上的模块信息的情况下



9.2 跟踪设置 / 复位 (TRACE、TRACER)

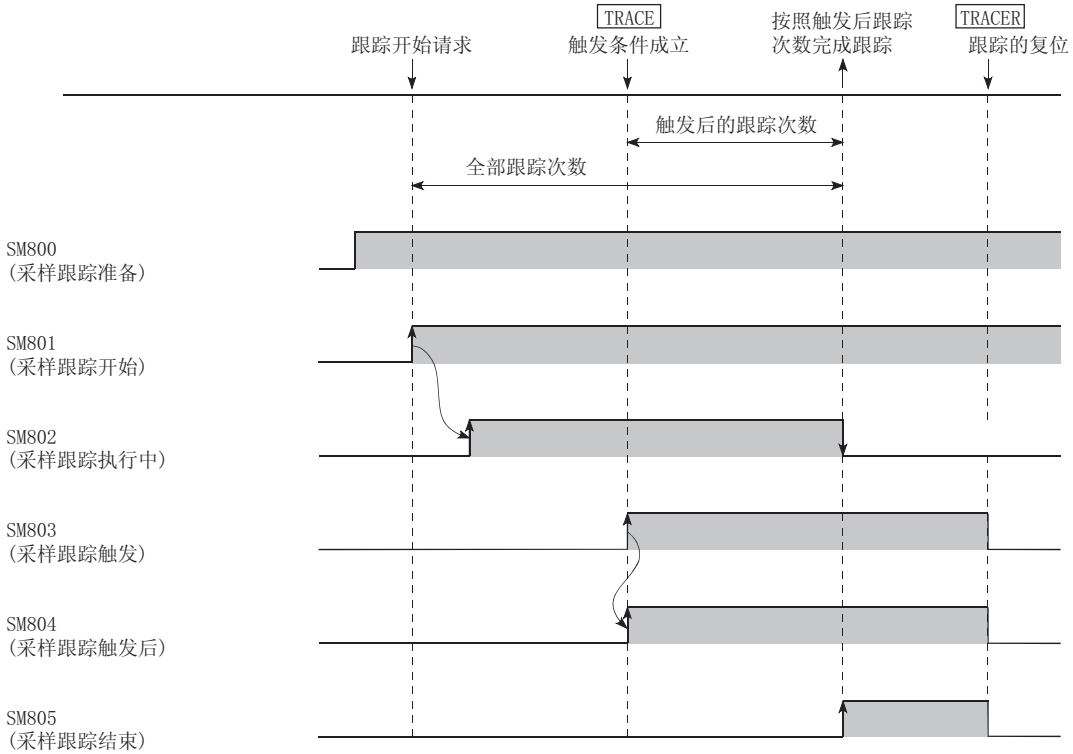


设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
--	--								

功 能

采样跟踪功能是指，在指定的时机对 CPU 模块的指定软元件的内容进行连续采集的功能。

进行采样跟踪时，在 SM800、SM801、SM802 为 ON 的情况下，按照设置的次数将跟踪结果存储到存储卡的采样跟踪用文件中。



TRACE

(1) 执行 TRACE 指令时，将 SM803 置为 ON，按照跟踪条件设置的触发后次数中设置的次数进行了采样后，将数据锁存并停止采样跟踪。

(2) 在跟踪执行过程中如果 SM801 变为 OFF，则采样将停止。

(3) 执行 TRACE 指令后，跟踪结束时将 SM805 置为 ON。

(4) 如果执行了 1 次 TRACE 指令，则第 2 次及以后的 TRACE 指令将被忽略。
如果执行了 TRACER 指令，则 TRACE 指令将再次有效。

TRACER

(1) TRACER 指令是对 TRACE 指令进行复位的指令。
如果执行了 TRACER 指令，则 TRACE 指令将再次有效。

(2) 如果执行了 TRACERE 指令，则 SM803 ～ SM805 将变为 OFF。

备 注

1. 关于采样跟踪的详细内容，请参阅所使用的 CPU 模块的用户手册（功能解说 / 程序基础篇）。
2. 关于通过 GX Developer 执行跟踪的内容，请参阅 GX Developer 的操作手册。

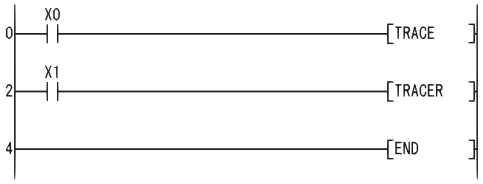
出 错

(1) 在 TRACE、TRACER 指令中无运算出错。

程序示例

(1) 以下为 X0 变为 ON 时执行 TRACE 指令，X1 变为 ON 时通过 TRACER 指令对 TRACE 指令进行复位的程序。

［梯形图模式］

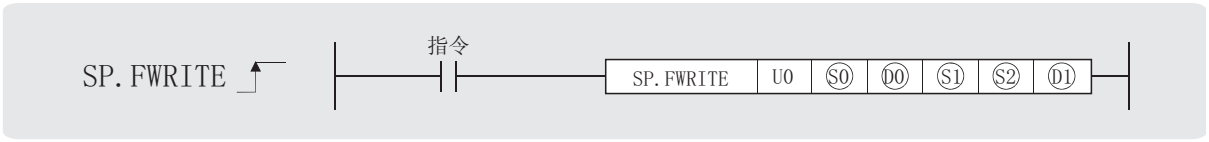


［列表模式］

步	指令	软元件
0	LD	X0
1	TRACE	
2	LD	X1
3	TRACER	
4	END	

9.3 写数据到指定的文件 (SP.FWRITE)

Basic High performance Process Redundant Universal



设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数		其它
	位	字		位	字			K、H	$	
S0	○	○		--				○	--	--
D0	--	○		--				--	--	--
S1	--	○		--				--	--	--
S2	--	○		--				--	○	--
D1	△ *1	△ *1		--				--	--	--

*1: 局部软元件以及各程序中设置的文件寄存器不能使用。

设置数据/控制数据

设置数据	内容			设置范围	设置方	数据类型
U0	虚拟			--	--	BIN16 位
S0	驱动器指定			2	用户	
D0	存储控制数据的软元件的起始编号。 控制数据如下所示。					
	软元件	项目	内容及设置数据	设置范围	设置方	
	D0	执行 / 结束类型	指定执行类型。 0000H : 二进制写入 0100H :CSV 格式转换写入	0000H 0100H	用户	
	D0 +1	(未使用)	系统用	--	系统	
	D0 +2	写入结果 (数据数)	输入S2中指定数据的实际写入数据数。值的单位取决于D0+7 的数据类型指定。	--	系统	
	D0 +3	(未使用)	--	--	--	
	D0 +4 D0 +5	文件位置	D0中指定了二进制写入时，设置文件位置。 00000000H: 从文件的起始开始。 00000001H ～ FFFFFFFEH : 根据指定位置。 单位取决于数据类型指定。 FFFFFFFFH : 添加到文件的最后。 D0中指定了 CSV 格式写入时 • 对于序列号的前 5 位数为 “01111” 以前的高性能型 QCPU，必须设置为文件的起始 (0H)。 • 对于序列号的前 5 位数为 “01112” 以后的高性能型 QCPU/ 过程 CPU/ 冗余 CPU/ 通用型 QCPU，设置文件位置。 00000000H ～ FFFFFFFEH : 从文件的起始开始。 FFFFFFFFH : 添加到文件的最后。	00000000H ～ FFFFFFFFH	用户	

设置数据	内容			设置范围	设置方	数据类型
ⓓ0	ⓓ0+6	列数指定	ⓓ0中指定了二进制写入时必须设置为0。 ⓓ0中指定了CSV格式写入指定时，设置进行写入的列数。 0　　：无列。设置为1行。 0以外　：设置为指定数的列。	0H ～ FFFFH (0 ～ 65535)	用户	BIN16位
	ⓓ0+7	数据类型指定	0: 字 1: 字节	0, 1	用户	
ⓢ1	存储文件名的软元件的起始编号。文件名如下所示。					
	软元件	项目	内容及设置数据	设置范围	设置方	
	ⓢ1 ～ ⓢ1+□	文件名字符串	指定文件名的字符串。 • 省略扩展名的情况下从“.”（点号）开始省略。 • 应在8字符+点号+3字符以内。 • 指定为9字符以上时，即使有扩展名也将被忽略，扩展名将变为“BIN”或者“CSV”。	字符串	用户	
ⓢ2	存储数据的软元件的起始编号。写入的数据如下所示。					
	软元件	项目	内容及设置数据	设置范围	设置方	
	ⓢ2	请求写入的数据数	指定进行写入请求的数据数。（以字为单位） ⓓ0+7中指定为字节时也将进行字换算后以字为单位进行设置。	1 ～ 480 1 ～ 32767*2	用户	
	ⓢ2+1 ～ ⓢ2+□	写入数据	进行写入请求的数据	0000H ～ FFFFH		
ⓓ1	处理结束时变为ON的位软元件（但是，异常结束时ⓓ1+1也变为ON。）					位
	软元件	项目	内容及设置数据	设置范围	设置方	
	ⓓ1	结束信号	表示处理结束。 ON: 结束　　OFF: 未结束	--	系统	
	ⓓ1+1	异常结束信号	表示是正常结束还是异常结束。 ON: 异常结束　OFF: 正常结束	--		

*2 ：仅为通用型QCPU的范围。

注意事项

(1) ⓢ0（驱动器指定）中只能设置为ATA卡的驱动器(2)。
但是，在安装了Flash卡的情况下，不能通过SP.FWRITE指令进行写入。
不能设置为SRAM卡、标准RAM、标准ROM的驱动器。

(2) CSV设置时写入的数据为10进制数的值。

□ 字符“A”(41H) → “65”被写入。
处理范围 :-32768 ～ 32767

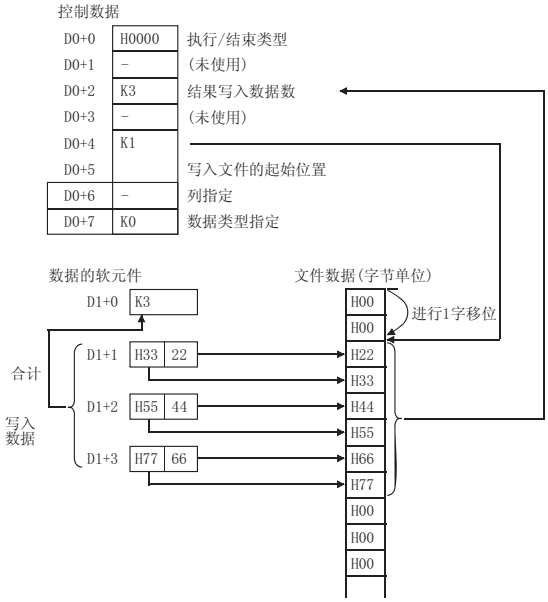
(3) 二进制写入时，字指定中的文件位置的设置范围为00000000H ～ 7FFFFFFFH、FFFFFFFFH。

功 能

(1) 将指定数据数的数据写入到指定的文件中。
根据控制数据的执行 / 结束类型，指定是将二进制数据原样不变地写入，还是将二进制数据转换为 CSV 格式后写入。
(写入对象仅为 ATA 卡。)

(2) 对于处理结束 (ⓓ1) 的位软元件，在检测出本指令的处理结束的 END 指令执行时自动地将其变为 ON，在下一个扫描的 END 指令时将其变为 OFF。
作为本指令的执行结束标志使用。
本指令异常结束时，异常结束 (ⓓ1+1) 软元件与处理结束 (ⓓ1) 软元件在相同的时机 ON/OFF，因此作为本指令的异常结束标志使用。
此外，在指令的执行过程中 SM721 将处于 ON 状态。
在 SM721 处于 ON 的状态下不能执行本指令。(执行时将变为无处理。)
再者，在执行指令前如果检测出出错 (SM721 变为 ON 之前)，处理结束 (ⓓ1) 软元件、异常结束 (ⓓ1+1) 软元件以及 SM721 将不变为 ON。

(3) 应将数据的请求写入数据数 (ⓢ2) 以及文件位置 (ⓓ0+4、ⓓ0+5) 的处理单位设置为字单位。
二进制写入时的请求写入数据数、文件位置指定时的数据的写入方法如下所示。



(4) 二进制写入时

(a) 在省略了对象文件的扩展名的情况下，扩展名将变为 “.BIN”。

(b) 在指定了不存在的文件的情况下，相应文件将被新建，将从起始开始对数据进行添加保存。
此时新建文件的属性将被设置为存档。

(c) 在写入数据的过程中如果超出了现有文件的大小，超出部分的数据将被添加保存。

(d) 指定的文件位置超出了现有文件的大小时的情况如下所示。

- 对于序列号的前 5 位数为 “01111” 及以前的高性能型 QCPU，将变为出错状态。
- 对于序列号的前 5 位数为 “01112” 及以后的高性能型 QCPU/ 过程 CPU/ 冗余 CPU/ 通用型 QCPU，将写入 0 点并正常结束。

(e) 在数据的添加保存过程中，如果存储媒介的可用空间已用尽时将变为出错状态。
此时，已写入或添加保存成功的部分将保持为写入状态不变。
在添加保存结束后出错状态将结束。

(5) CSV 格式转换写入时

(a) 在省略了扩展名的情况下，扩展名将变为 “.CSV”。

(b) 在指定了已存在的文件时的情况如下所示。
[序列号的前 5 位数为 “01111”以前的高性能型 QCPU]
将文件内容全部删除后，从起始开始保存数据。
[序列号的前 5 位数为 “01112”以后的高性能型 QCPU/ 过程 CPU/ 冗余 CPU/ 通用型 QCPU]

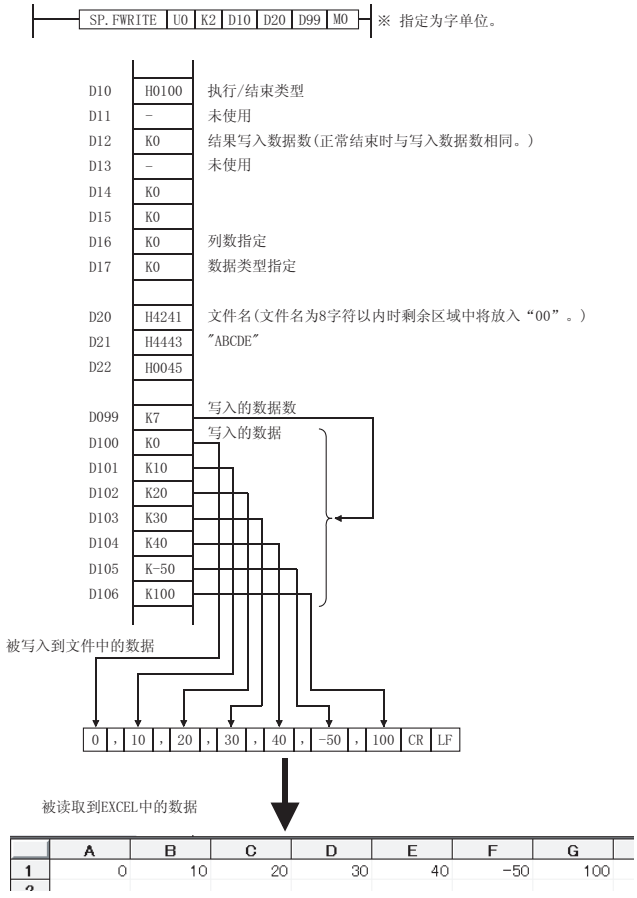
- (Ⓓ0+4、Ⓓ0+5) 中设置为除 FFFFFFFFH 以外时，将文件内容全部删除后，从起始开始保存数据。
- (Ⓓ0+4、Ⓓ0+5) 中设置为 FFFFFFFFH 时，从文件的最后开始保存数据。

(c) 在指定了不存在的文件的情况下，相应文件将被新建，将从起始开始对数据进行添加保存。
此时新建文件的属性将被设置为存档。

(d) 在数据的添加保存过程中，如果存储媒介的可用空间已用尽时将变为出错状态。
此时，已添加保存成功的部分将保持为写入状态不变。
在添加保存结束后出错状态将结束。

(e) 将列数指定为 0 时，作为 1 行的格式文件保存。

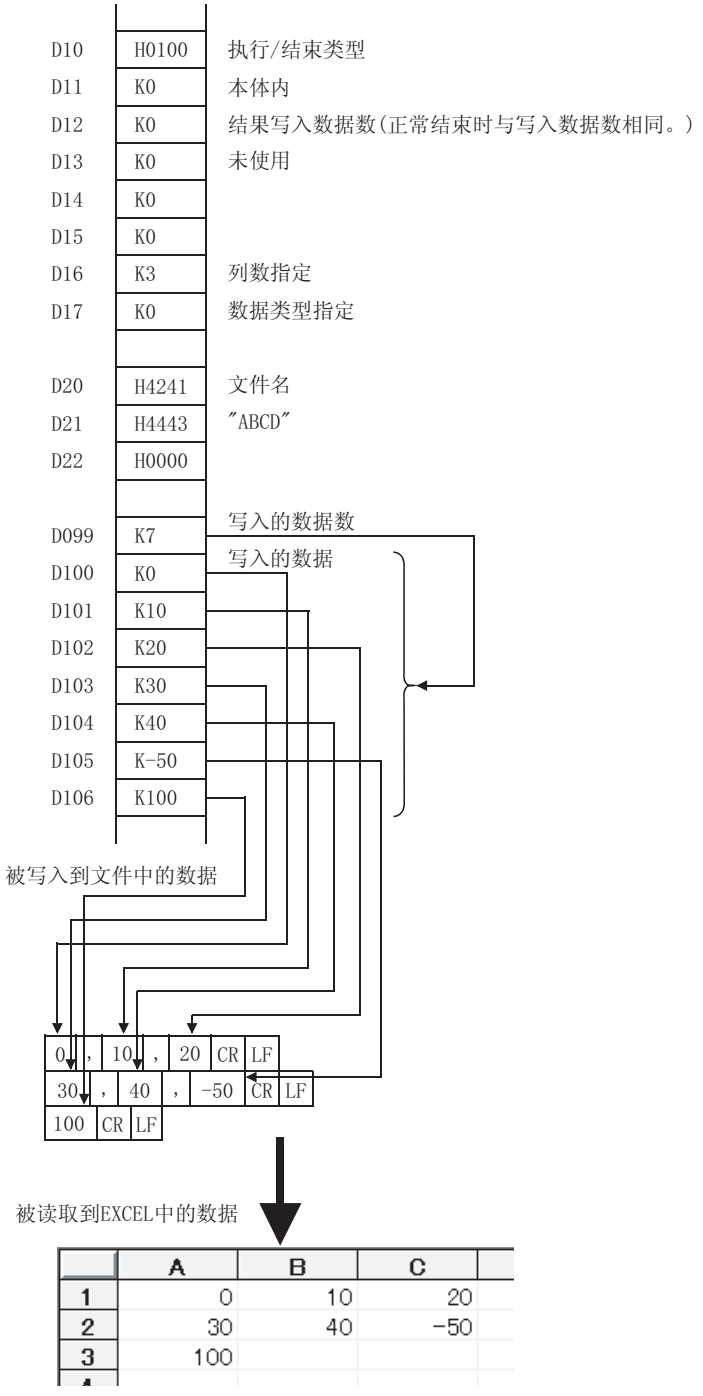
CSV 格式转换写入时将列数指定为 0 的情况下



(f) 在 CSV 格式转换写入中，在指定列数不为 0 的情况下，将作为指定列数的表格被保存为 CSV 格式文件。

CSV 格式转换写入时指定列数不为 0 的情况下

※ 指定为字单位。

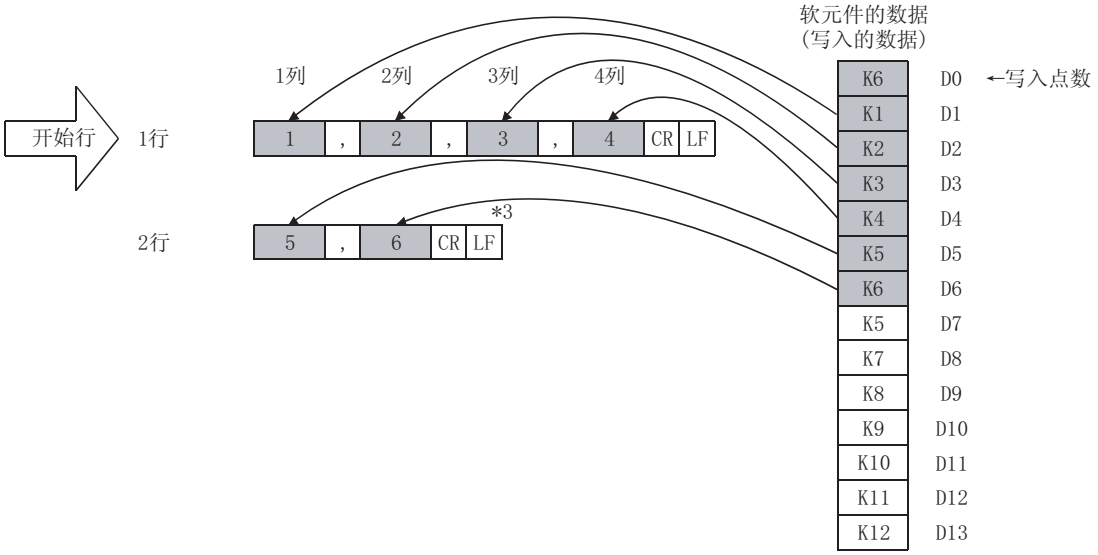


	A	B	C
1	0	10	20
2	30	40	-50
3	100		

(g) 在序列号的前 5 位数为 “01112” 以后的高性能型 QCPU/ 过程 CPU/ 冗余 CPU/ 通用型 QCPU 中，添加数据的情况如下所示。

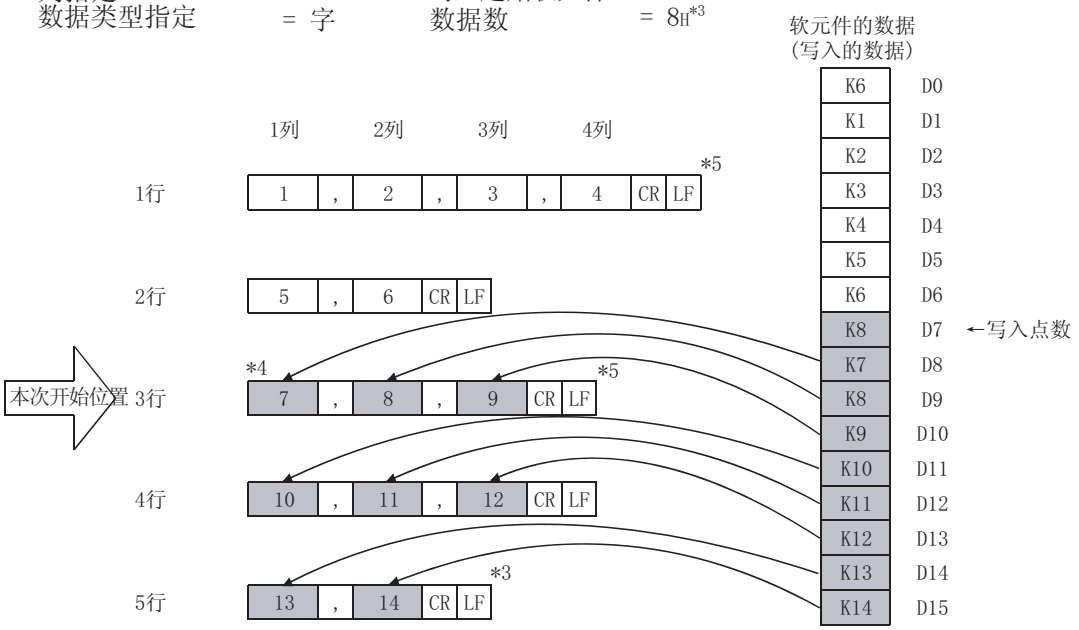
[指定进行写入的文件。](即使文件已存在也将其删除后再次新建。)

执行类型	= CSV格式	文件位置	= 0H(新建文件)
列指定	= 4H *3*5	写入起始软元件	= D0
数据类型指定	= 字	数据数	= 6H *3



[在添加模式中，被添加到文件的最后。]

执行类型	= CSV格式	文件位置	= FFFFFFFFH(继续模式)
列指定	= 3H *3*5	写入起始软元件	= D7
数据类型指定	= 字	数据数	= 8H*3



*3: 如果未将 “写入点数” 设置为 “列指定” 的整数倍，列数将变为随机数。

*4: 由于最后的数据后面必定附有换行码，因此通常添加模式时是从新一行的起始处开始添加。

*5: 添加模式时，如果 “列指定” 与上次写入时不相同，则列数将错乱。

(h) 不要在中断程序中执行本指令。
(如果在中断程序中执行了本指令将无法保证动作正常。)

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

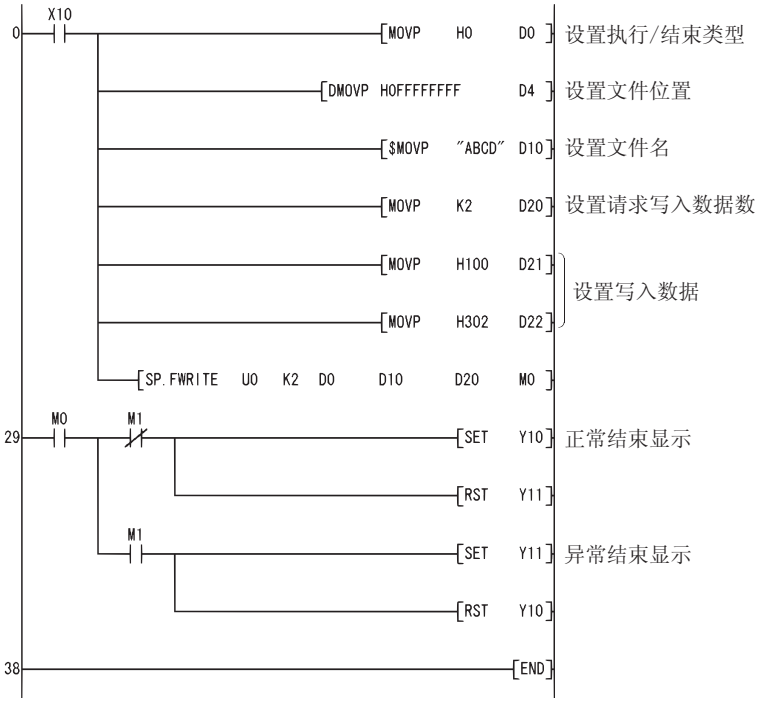
- 驱动器指定(S0)中指定的驱动器不是 ATA 卡时。 (出错代码： 4100)
- 设置到控制数据(D0)后面的值超出了设置范围时。 (出错代码： 4100)
- 请求写入数据数 ((S2)) 中指定的值超出了设置范围时，或者超出了 ((S2)+1) 后面的软元件范围时。 (出错代码： 4101)
- ATA 卡的可用空间不足时。 (出错代码： 4100)
- 新建文件时，可用空间不足时。 (出错代码： 4100)
- 指定了不能指定的软元件时。 (出错代码： 4004)
- ATA 卡内发生了访问异常时。 (出错代码： 4100)
- 文件名 ((S1)) 中设置了不能使用的值时。 (出错代码： 4100)
- 文件名 ((S1)) 的属性为只读时。 (出错代码： 4100)
- (D0)或者(D1)中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU) (出错代码 :4101)

程序示例

(1) 以下为 X10 变为 ON 时，在安装为驱动器 2 的存储卡的文件 “ABCD.BIN”中添加 00H、01H、02H、03H 这 4 个字节的二进制数据的程序。

• 将ⓓ开始的 8 点预留给控制数据用软元件。

[梯形图模式]



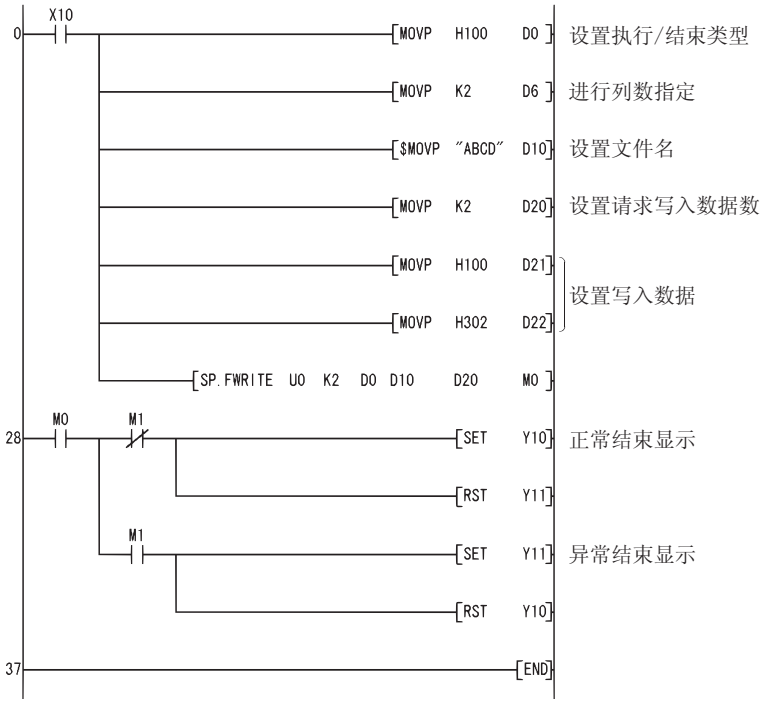
[列表模式]

步	指令	软元件						
0	LD	X10						
1	MOVP	H0	D0					
3	DMOVP	H0FFFFFFFF	D4					
6	$MOVP	"ABCD"	D10					
11	MOVP	K2	D20					
13	MOVP	H100	D21					
15	MOVP	H302	D22					
17	SP.FWRITE		U0	K2	D0	D10	D20	M0
29	LD	M0						
30	MPS							
31	ANI	M1						
32	SET	Y10						
33	RST	Y11						
34	MPP							
35	AND	M1						
36	SET	Y11						
37	RST	Y10						
38	END							

(2) 以下为 X10 变为 ON 时，将 00H、01H、02H、03H 这 4 个字节以 2 列的 CSV 格式文件创建到安装为驱动器 1 的存储卡的文件中，并命名为 “ABCD.CSV” 的文件名的程序。

• 文件的写入情况如下所示。

[梯形图模式]



[列表模式]

步	指令	软元件					
0	LD	X10					
1	MOVP	H100	D0				
3	MOVP	K2	D6				
5	$MOVP	"ABCD"	D10				
10	MOVP	K2	D20				
12	MOVP	H100	D21				
14	MOVP	H302	D22				
16	SP.FWRITE		U0	K2	D0	D10	D20	M0
28	LD	M0					
29	MPS						
30	ANI	M1					
31	SET	Y10					
32	RST	Y11					
33	MPP						
34	AND	M1					
35	SET	Y11					
36	RST	Y10					
37	END						

• 将(D0)开始的 8 点预留给控制数据用软元件。

0	,	1	,	CR	LF
2	,	3	,	CR	LF

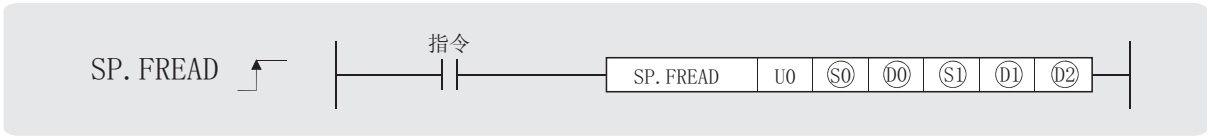
被写入的文件内容

被读取到EXCEL中的数据

	A	B
1	0	1
2	2	3

9.4 从指定文件中读取数据 (SP. FREAD)

Basic | High performance | Process | Redundant | Universal



设置数据	内部软元件		R, RZ	J□\□		U□\G□	Zn	常数		其它
	位	字		位	字			K、H	$	
(S0)	○	○		--				○	--	--
(D0)	--	○		--				--	--	--
(S1)	--	○		--				--	○	--
(D1)	--	○		--				--	--	--
(D2)	△ *1	△ *1		--				--	--	--

*1: 局部软元件以及各程序中设置的文件寄存器不能使用。

设置数据/控制数据

设置数据	内容			设置范围	设置方	数据类型
U0	虚拟			--	--	BIN16 位
(S0)	驱动器指定			2	用户	
(D0)	存储控制数据的软元件的起始编号。 控制数据如下所示。					
	软元件	项目	内容及设置数据	设置范围	设置方	
	(D0)	执行 / 结束类型	指定执行类型。 0000H : 二进制读取 0100H :CSV 格式转换读取	0000H 0100H	用户	
	(D0)+1	（未使用）	系统用	--	系统	
	(D0)+2	请求读取的数据数	指定希望读取的数据数。 （字单位） 在(D0)+7 中将数据类型指定为字节时，也将进行字换算后以字为单位进行设置。	1 ～ 480 1 ～ 32767*2	用户	
	(D0)+3	（未使用）	--	--	--	

*2: 仅为通用型 QCPU 的范围。

设置数据	内容			设置范围	设置方	数据类型
(D0)	(D0)+4 (D0)+5	文件位置	(D0)中指定了二进制读取时，设置文件位置。 00000000H：从文件的起始开始。 00000001H ～ FFFFFFFEH ：根据指定位置。 单位取决于数据类型指定。 FFFFFFFFH ：不能设置 在(D0)中指定了CSV格式读取时 • 对于序列号的前5位数为“01111”以前的高性能型QCPU，必须设置为文件的起始(0H)。 • 对于序列号的前5位数为“01112”以后的高性能型QCPU/过程CPU/冗余CPU/通用型QCPU，设置文件位置（行）。 00000000H：从文件的起始开始读取。 00000001H ～ FFFFFFFEH ：从指定行开始读取。 FFFFFFFFH ：从上次读取的位置开始继续读取。	00000000H ～ FFFFFFFFH	用户	BIN16位
	(D0)+6	列数指定	在(D0)中指定了二进制读取时必须设置为0。 在(D0)中指定了CSV格式读取指定时，设置进行读取的列数。 0 ：无列。视为1行。 0以外 ：视为指定数的列。	0H ～ FFFFH (0 ～ 65535)	用户	
	(D0)+7	数据类型指定	0：字 1：字节	0,1	用户	
(S1)	存储文件名的软元件的起始编号。文件名如下所示。					
	软元件	项目	内容及设置数据	设置范围	设置方	
	(S1) ～ (S1)+□	文件名字符串	指定文件名的字符串。 • 省略扩展名的情况下从“.”（点号）开始省略。 • 应在8字符+点号+3字符以内。 • 指定为9字符以上时，即使有扩展名也将被忽略，扩展名将变为“BIN”或者“CSV”。	字符串	用户	
(D1)	存储读取的数据的软元件的起始编号。					
	软元件	项目	内容及设置数据	设置范围	设置方	
	(D1)	读取结果数据数	设置相对于(D0)+2中指定数据数的实际读取的数据数。值的单位根据数据类型指定。	--	系统	
	(D1)+1 ～ (D1)+□	读取数据	读取的数据	--	系统	
(D2)	处理结束时变为ON的位软元件（但是，异常结束时(D2)+1也变为ON。）					位
	软元件	项目	内容及设置数据	设置范围	设置方	
	(D2)	结束信号	表示处理结束。 ON：结束 OFF：未结束	--	系统	
	(D2)+1	异常结束信号	表示是正常结束还是异常结束。 ON：异常结束 OFF：正常结束	--		

注意事项

(1) ⓢ0（驱动器指定）中只能设置为 ATA 卡的驱动器 (2)。
但是，在安装了 Flash 卡的情况下，不能通过 SP.FREAD 指令进行读取。
不能设置为 SRAM 卡、标准 RAM、标准 ROM 的驱动器。

(2) CSV 设置时写入的数据为 10 进制数的值。
□ 字符 “A”(41H) → “65”被写入。
处理范围 : -32768 ～ 32767

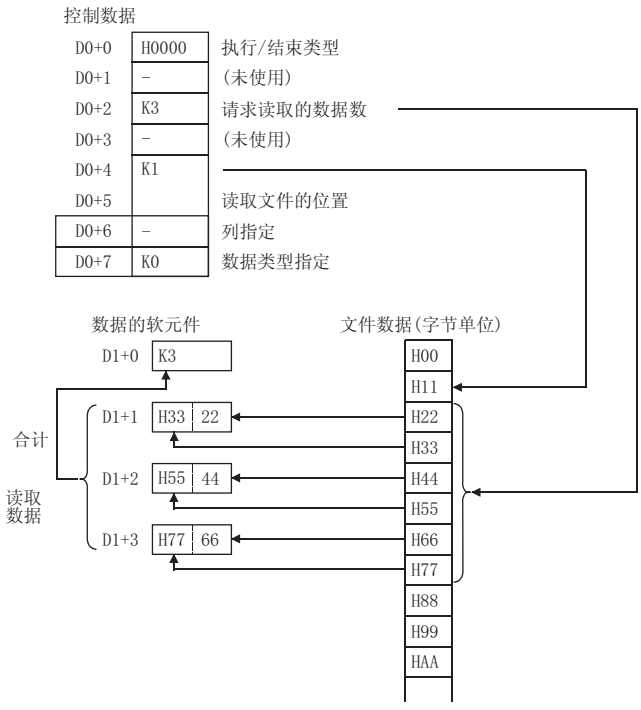
(3) 二进制读取时，字指定中的文件位置的设置范围为 00000000H ～ 7FFFFFFFH。

功 能

(1) 从指定文件中进行数据读取。
根据控制数据的执行 / 结束类型，指定是将文件内容以二进制数据读取，还是将文件内容转换为 CSV 格式后读取。（读取的对象仅为 ATA 卡。）

(2) 对于处理结束（ⓓ2）的位软元件，在检测出本指令的处理结束的 END 指令执行时自动地将其变为 ON，在下一个扫描的 END 指令时将其变为 OFF。
作为本指令的执行结束标志使用。
本指令异常结束时，异常结束（ⓓ1+1）软元件与处理结束（ⓓ1）软元件在相同的时机 ON/OFF，因此作为本指令的异常结束标志使用。
此外，在指令的执行过程中 SM721 将处于 ON 状态。
在 SM721 处于 ON 的状态下不能执行本指令。（执行时将变为无处理。）
再者，在执行指令前如果检测出出错 (SM721 变为 ON 之前)，处理结束（ⓓ1）软元件、异常结束（ⓓ1+1）软元件以及 SM721 将不变为 ON。

(3) 应将数据的请求读取数据数（ⓓ0+2）、文件位置（ⓓ0+4、ⓓ0+5）以及读取的数据软元件大小（ⓓ1）的处理单位设置为字单位。
二进制读取时，各指定情况下的数据读取方法如下所示。



(4) 二进制读取时

(a) 在省略了对象文件的扩展名的情况下，扩展名将变为 “.BIN”。

(b) 在指定了不存在的文件的情况下，将变为出错状态。

(c) 指定的文件位置超出了现有文件的大小时的情况如下所示。

- 对于序列号的前 5 位数为 “01111”及以前的高性能型 QCPU，将变为出错状态。
- 对于序列号的前 5 位数为 “01112”及以后的高性能型 QCPU/ 过程 CPU/ 冗余 CPU/ 通用型 QCPU，将读取 0 点并正常结束。

(5) CSV 格式转换读取时

(a) 将 CSV 格式文件的要素 (EXCEL 的单元格) 按行方向的顺序进行读取，将数值字符串转换为二进制值后，存储到软元件中。

(b) 在省略了扩展名的情况下，扩展名将变为 “.CSV”。

(c) 在指定了不存在的文件的情况下将变为出错状态。

(d) 从文件的起始开始按照请求读取数据数 (Ⓓ+2) 中指定的值进行要素读取。
在读取到文件的最后数据时所读取的数据量尚未达到指定的数据数时的情况如下所示。

- 对于序列号的前 5 位数为 “01111”及以前的高性能型 QCPU，将变为出错状态。
- 对于序列号的前 5 位数为 “01112”及以后的高性能型 QCPU/ 过程 CPU/ 冗余 CPU/ 通用型 QCPU，将读取可以读取的数据。

(e) 在指定列数为 0 的情况下，在读取时将忽略 CSV 格式文件的行区分。

CSV 格式转换读取时将列数指定为 0 的情况下

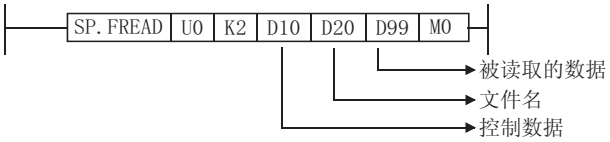
以EXCEL创建的数据

	A	B	C
1	大/小项目		测定值
2	长度	1	3
3	温度	-21	

以CSV格式保存的数据

大/小项目	,	,	测定值	CR	LF	
长度	,	1	,	3	CR	LF
温度	,	-21	,		CR	LF

读取到软元件中的数据



控制数据

D10	H0100	执行/结束类型
D11	-	未使用
D12	K9	请求读取的数据数
D13	-	未使用
D14	K0	
D15	K0	
D16	K0	列数指定
D17	K0	数据类型指定
D20	H4241	文件名
D21	H4443	"ABCDE"
D22	H0045	

被读取的数据

存入读取的数据数 → D099 K9 ····· 读取结果的数据数

读取的数据

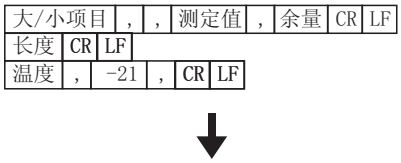
大/小项目 →	D100	K0	····· 由于“大/小项目”为非数值的数据，因此转换数据(0)被存储。
,与, 之间的数据 →	D101	K0	····· 由于“”为非数值的数据，因此转换数据(0)被存储。
测定值 →	D102	K0	····· 由于“测定值”为非数值的数据，因此转换数据(0)被存储。
长度 →	D103	K0	····· 由于“长度”为非数值的数据，因此转换数据(0)被存储。
1 →	D104	K1	····· 由于“1”为数值数据，因此被转换为二进制值。
3 →	D105	K3	····· 由于“3”为数值数据，因此被转换为二进制值。
温度 →	D106	K0	····· 由于“温度”为非数值的数据，因此转换数据(0)被存储。
K-21 →	D107	K-21	····· 由于“-21”为数值数据，因此被转换为二进制值。
,与CR 之间的数据 →	D108	K0	····· 由于“”为非数值的数据，因此转换数据(0)被存储。

即使在各行的列数不相同的情况下，也将在忽略行的状况下进行读取。

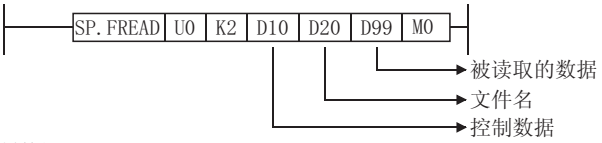
要 点

在 EXCEL 中不会创建出这样的文件。在用户对 CSV 文件进行了修改时有可能发生此现象。

□ 在读取时各行的列数不相同的情况下



被读取到软元件中的数据



控制数据

D10	H0100	执行/结束类型
D11	-	未使用
D12	K7	请求读取的数据数
D13	-	未使用
D14	K0	
D15	K0	
D16	K0	列数指定
D17	K0	数据类型指定
D20	H4241	文件名
D21	H4443	"ABCD"
D22	H0000	

被读取的数据

存入读取的数据数

读取的数据

	D099	K7	读取结果的数据数
大/小项目	D100	K0	由于“大/小项目”为非数值的数据，因此转换数据(0)被存储。
,与, 之间的数据	D101	K0	由于“”为非数值的数据，因此转换数据(0)被存储。
测定值	D102	K0	由于“测定值”为非数值的数据，因此转换数据(0)被存储。
余量	D103	K0	由于“余量”为非数值的数据，因此转换数据(0)被存储。
长度	D104	K0	由于“长度”为非数值的数据，因此转换数据(0)被存储。
温度	D105	K0	由于“温度”为非数值的数据，因此转换数据(0)被存储。
K-21	D106	K-21	由于“-21”为数值数据，因此被转换为二进制值。

(f) 在 CSV 格式转换读取中，在指定列数不为 0 时，将作为指定的列数的表的 CSV 格式文件进行读取。超出了指定列数要素将被忽略。

□ 在 CSV 格式转换读取时指定列数不为 0 的情况下

以EXCEL创建的数据

	A	B	C
1	大/小项目		测定值
2	长度	1	3
3	温度	-21	

以CSV格式保存的数据

大/小项目	,	,	测定值	CR	LF	
长度	,	1	,	3	CR	LF
温度	,	-21	,		CR	LF

超出了指定列数的要素被忽略。

被读取到软元件中的数据

SP. FREAD	U0	K2	D10	D20	D99	M0

被读取的数据
文件名
控制数据

控制数据

D10	H0100	执行/结束类型
D11	-	未使用
D12	K6	请求读取的数据数
D13	-	未使用
D14	K0	
D15	K0	
D16	K2	列数指定
D17	K0	数据类型指定
D20	H4241	文件名
D21	H4443	"ABCD"
D22	H0000	

被读取的数据

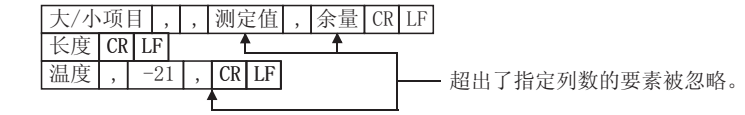
存入读取的数据数

读取的数据

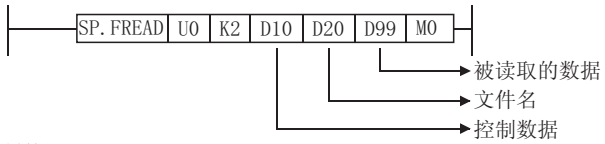
	D099	K6	……读取结果的数据数
大/小项目	D100	K0	……由于“大/小项目”为非数值的数据，因此转换数据(0)被存储。
,与, 之间的数据	D101	K0	……由于“”为非数值的数据，因此转换数据(0)被存储。
长度	D102	K0	……由于“长度”为非数值的数据，因此转换数据(0)被存储。
1	D103	K1	……由于“1”为数值数据，因此被转换为二进制值。
温度	D104	K0	……由于“温度”为非数值的数据，因此转换数据(0)被存储。
K-21	D105	K-21	……由于“-21”为数值数据，因此被转换为二进制值。

即使在各行的列数不相同的情况下，超出了指定列数的要素将被忽略。未超出指定列数的列中将被填补 0。

在读取时各行的列数不相同的情况下



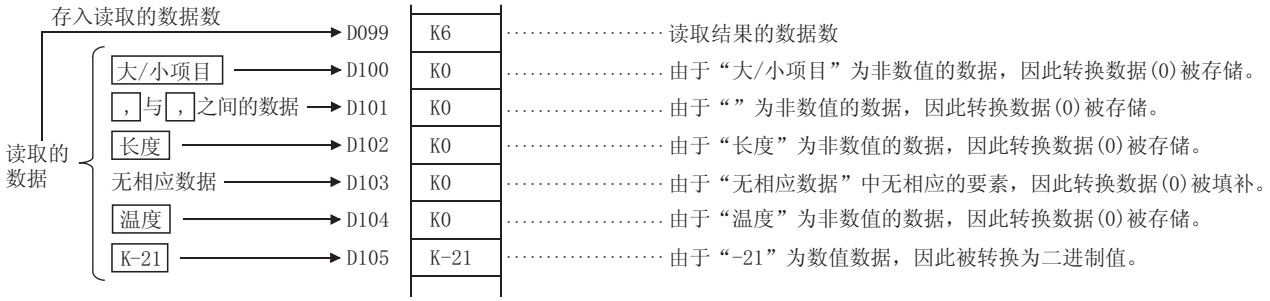
被读取到软元件中的数据



控制数据

D10	H0100	执行/结束类型
D11	-	未使用
D12	K6	请求读取的数据数
D13	-	未使用
D14	K0	
D15	K0	
D16	K2	列数指定
D17	K0	数据类型指定
D20	H4241	文件名
D21	H4443	"ABCD"
D22	H0000	

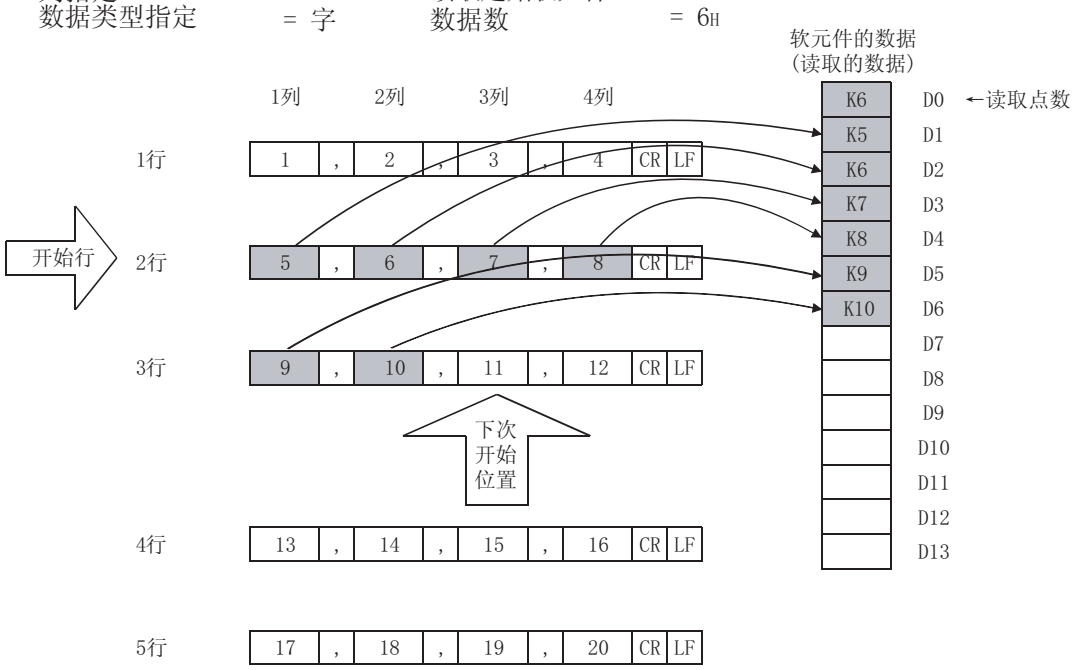
被读取的数据



(g) 在序列号的前 5 位数为 “01112” 及以后的高性能型 QCPU/ 过程 CPU/ 冗余 CPU/ 通用型 QCPU 中，可以分多次进行读取。

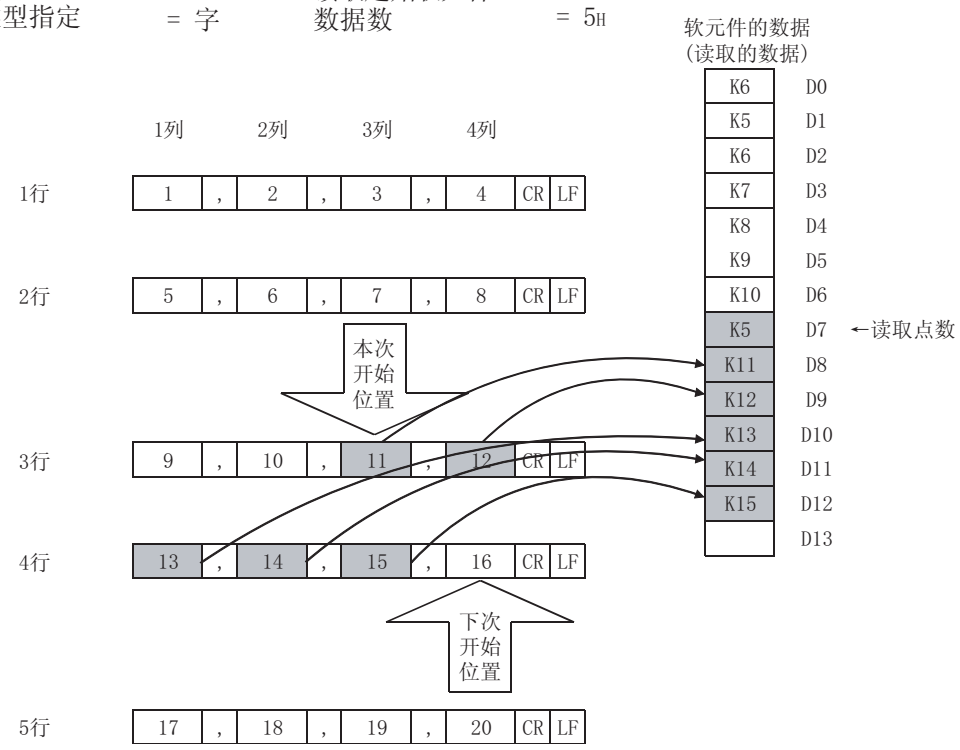
[指定希望开始读取的行。]

执行类型	= CSV格式	开始行数	= 2H
列指定	= 4H	读取起始软元件	= D0
数据类型指定	= 字	数据数	= 6H



[在上次继续模式中，从上次读取的位置开始继续进行读取。]

执行类型	= CSV格式	开始行数	= FFFFFFFFH(继续模式)
列指定	= 4H	读取起始软元件	= D7
数据类型指定	= 字	数据数	= 5H



- 以继续模式进行读取时，如果对 “执行类型”、“列指定”、“数据类型指定” 进行了与上次不同的设置，将无法正常地从上次位置开始添加。
- 在继续模式进行读取的过程中，如果执行了其它设置的 SP.FREAD 指令或 SP.FWRITE 指令，将无法正常地从上次位置开始添加。

(h) 在 CSV 格式转换读取中，如果读取的 CSV 格式文件中存在有超出范围的值或者非数值要素，将被转换为 0H。

(i) 在 CSV 格式转换读取中，读取时的数值转换情况如下所示。

CSV 内数值		-32768 ～ -1	0 ～ 32767	32768 ～ 65535
字软元件	无符号	32768 ～ 65535	0 ～ 32767	32768 ～ 65535
	有符号	-32768 ～ -1	0 ～ 32767	-32768 ～ -1

(j) 不要在中断程序中执行本指令。
（如果在中断程序中执行了本指令，有可能导致误动作。）

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

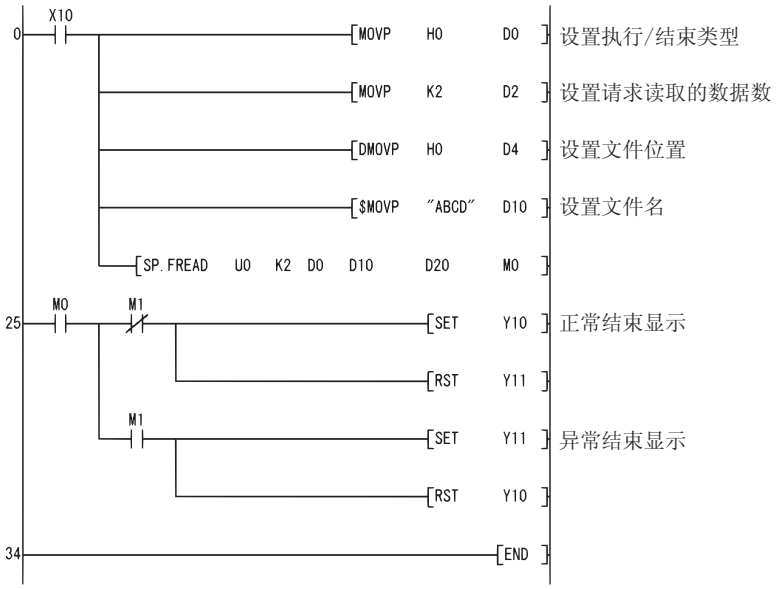
- 驱动器指定（(S0)）中指定的驱动器不是 ATA 卡时。（出错代码： 4100）
- 设置到控制数据（(D0)）后面的值超出了设置范围时。（(D0)+2 除外）（出错代码： 4100）
- 读取的数据数（(D0)+2）中指定的值超出了设置范围时。（出错代码： 4101）
- 指定了不能指定的软元件时。（出错代码： 4004）
- 文件名字符串（(S1)）后面指定的文件名在指定的驱动器中不存在时。（出错代码： 2410）
- 读取的数据大小超出了读取软元件的大小时。（出错代码： 4101）
- 二进制读取时请求读取数据数（(D0)+2）中指定的大小低于文件的数据数时。
（序列号的前 5 位数为“01111”以前的高性能性 QCPU）（出错代码： 4100）
- ATA 卡内发生了访问异常时。（出错代码： 4100）
- (D0)或者(D2)中指定的软元件超出了相应软元件的范围时。（仅通用型 QCPU）（出错代码： 4101）

程序示例

(1) 以下为 X10 变为 ON 时，从安装为驱动器 2 的存储卡的文件 “ABCD.BIN”的起始开始以二进制读取 4 个字节的程序。

- 将(D0)开始的 8 点预留给控制数据用软元件。
- 将 D20 开始的 100 个字节预留给读取用软元件。

[梯形图模式]



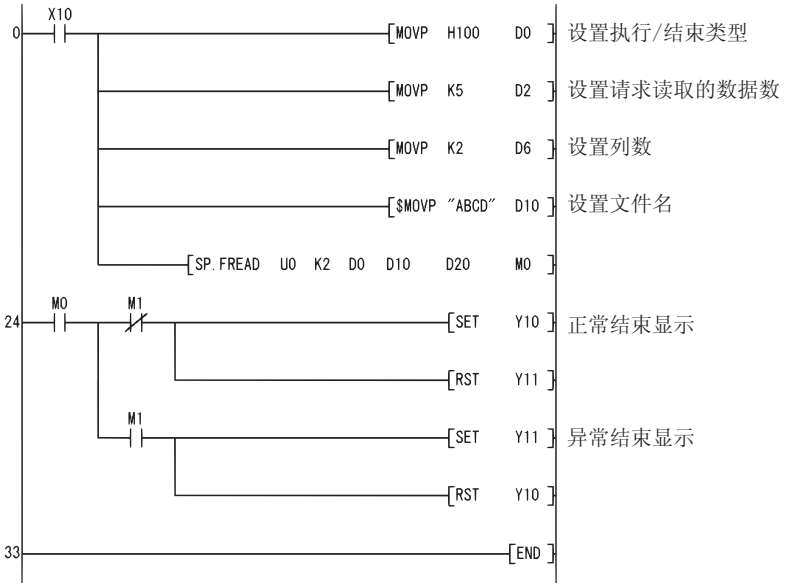
[列表模式]

步	指令	软元件					
0	LD	X10					
1	MOVP	H0	D0				
3	MOVP	K2	D2				
5	DMOVP	H0	D4				
8	$MOVP	"ABCD"	D10				
13	SP. FREAD	U0	K2	D0	D10	D20	M0
25	LD	M0					
26	MPS						
27	AND	M1					
28	SET	Y10					
29	RST	Y11					
30	MPP						
31	AND	M1					
32	SET	Y11					
33	RST	Y10					
34	END						

(2) 以下为 X10 变为 ON 时，从安装在插槽 0 中的 PC 卡中将文件名为 “ABCD.CSV” 的文件以 2 列的 CSV 格式读取的程序。

- 将Ⓓ0开始的 8 点预留给控制数据用软元件。
- 将 D20 开始的 100 个字节预留给读取用软元件。
- 假设读取的 CSV 格式文件中不存在非数值要素。

[梯形图模式]



[列表模式]

步	指令	软元件					
0	LD	X10					
1	MOVP	H100	D0				
3	MOVP	K5	D2				
5	MOVP	K2	D6				
7	$MOVP	"ABCD"	D10				
12	SP.FREAD	U0	K2	D0	D10	D20	M0
24	LD	M0					
25	MPS						
26	ANI	M1					
27	SET	Y10					
28	RST	Y11					
29	MPP						
30	AND	M1					
31	SET	Y11					
32	RST	Y10					
33	END						

9.5 向标准 ROM 中写入数据 (SP.DEVST)

Basic | High performance | Process | Redundant | Universal

n1 ：软元件数据存储用文件的写入偏置（以 1 点 16 位的单位指定）(BIN32 位）。

Ⓢ ：写入到标准 ROM 中的软元件的起始编号（软元件名）。

n2 ：写入点数 (BIN16 位）。

Ⓓ ：Ⓓ+0: 结束软元件（位）

Ⓓ+1: 异常结束软元件（位）。

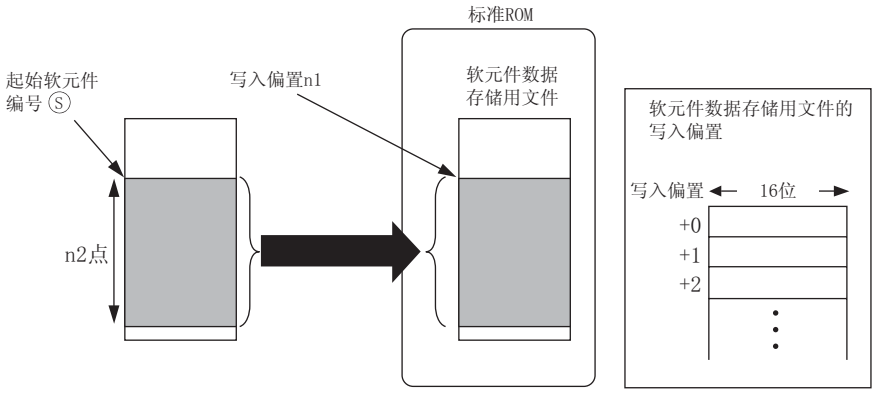
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
n1	--	○	○	--				○	--
Ⓢ	--	○	○	--				--	--
n2	--	○	○	--				○	--
Ⓓ	△*1	--	△*1	--				--	--

*1: 局部软元件以及各程序中设置的文件寄存器不能使用。

功 能

(1) 将Ⓢ中指定的软元件的 n2 中指定的点数的软元件数据，写入到标准 ROM 上的软元件数据存储用文件的 n1 中指定的写入偏置中。

n1 为软元件数据存储用文件从起始开始的偏置，通过字偏置（以每 16 位 +1 为单位）指定。



(2) 对于标准 ROM 的软元件数据写入位置结束软元件（Ⓓ+0）在检测出本指令的处理结束的 END 指令执行时自动地将其变为 ON，在下一个扫描的 END 指令时将其变为 OFF，因此作为本指令的执行结束标志使用。

(3) 本指令异常结束时，异常结束软元件（Ⓓ+1）与结束软元件（Ⓓ+0）在相同的时机 ON/OFF，因此作为本指令的异常结束标志使用。

(4) 在指令的执行过程中 SM721 将处于 ON 状态。

在 SM721 已处于 ON 的状态下不能执行本指令。（执行时将变为无处理。）

(5) 在执行指令时如果检测出出错，结束软元件（ⓓ+0）、异常结束软元件（ⓓ+1）以及 SM721 将不变为 ON。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- n1 中指定的写入偏置超出了软元件数据存储用文件的范围时。（出错代码： 4100）
- 从 n1 中指定的写入偏置开始的 n2 点超出了软元件数据存储用文件的范围时。（出错代码： 4100）
- 从ⓢ中指定的软元件开始的 n2 点的范围超出了相应软元件的范围时。（出错代码： 4101）
- 在可编程控制器参数的可编程控制器文件设置中，未对软元件存储用文件进行设置时。（出错代码： 2410）
- ⓓ中指定的软元件超出了相应软元件的范围时。（出错代码： 4101）

程序示例

(1) 以下为 M0 变为 ON 时，从 D100 中将 10 点数据写入到标准 ROM 的软元件数据存储用文件中的程序。

［梯形图模式］

［列表模式］

步	指令	软元件
0	LD	M0
1	SP.DEVST	K3 D100 K10 M1
12	END	

注意事项

(1) 写入到标准 ROM 中的值为本指令执行时的值。

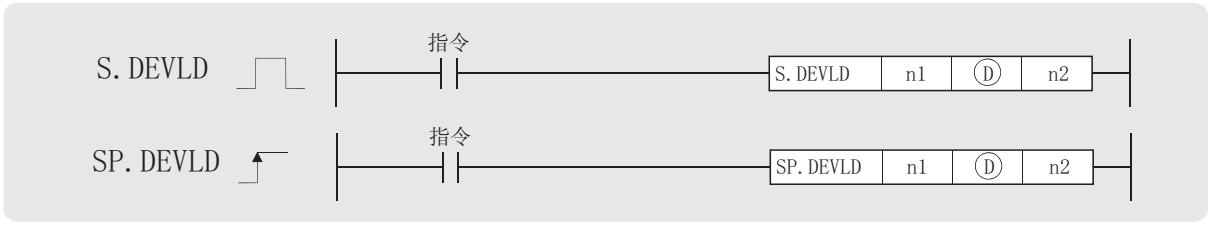
(2) 随着 SP.DEVST 指令的执行，标准 ROM 写入次数指标（SD687、SD688）将增加。标准 ROM 写入次数指标超过了 10 万次时，将发生 FLASH ROM ERROR（出错代码： 1610）的出错。

(3) 为了防止意外指令执行导致 ROM 写入次数的意外增加，对至标准 ROM 的写入指令执行次数指定（SD695）进行设置，对 1 日内的写入次数进行限制。

如果超过了所设置的写入次数（默认值： 36 次），将发生 OPERATION ERROR（出错代码： 4113）的出错。

9.6 从标准 ROM 中读取数据 (S(P).DEVLD)

Basic | High performance | Process | Redundant | Universal



n1 ：从软元件数据存储用文件中读取偏置（以 1 点 16 位的单位指定）(BIN32 位）。

Ⓓ ：从标准 ROM 中读取的软元件的起始编号（软元件名）。

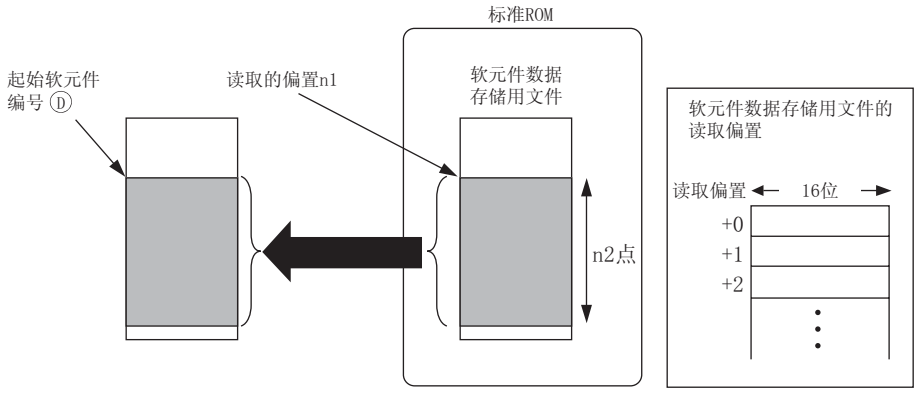
n2 ：读取点数 (BIN16 位）。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
n1	--	○		--				○	--
Ⓓ	--	○		--				--	--
n2	--	○		--				○	--

功 能

(1) 从标准 ROM 上的软元件数据存储用文件的 n1 中指定的读取偏置中，读取 n2 中指定的点数的软元件数据后，存储到 Ⓓ中指定的软元件中。

n1 为软元件数据存储用文件从起始开始的偏置，通过字偏置（以每 16 位 +1 为单位）指定。



出 错

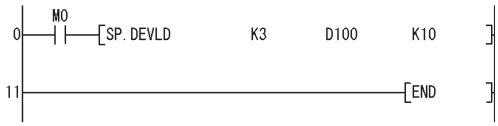
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- n1 中指定的地址超出了标准 ROM 的范围时。 (出错代码： 4100)
- 从 n1 中指定的地址开始的 n2 点超出了标准 ROM 的范围时。 (出错代码： 4100)
- 从Ⓓ中指定的软元件开始的 n2 点的范围超出了相应软元件范围时。
  (出错代码： 4101)
- 在可编程控制器参数的可编程控制器文件设置中，未进行软元件存储用文件的设置时。
  (出错代码： 2410)

程序示例

(1) 以下为 M0 变为 ON 时，从标准 ROM 的软元件数据存储用文件中将 10 点数据读取到 D100 中的程序。

[梯形图模式]



[列表模式]

步	指令	软元件		
0	LD	M0		
1	SP.DEVLD	K3	D100	K10
11	END			

9.7 通过存储卡进行程序装载 (PLOADP)

Basic | High performance | Process | Redundant | Universal

Ⓢ：存储装载的程序的驱动器号、文件名的字符串数据或者存储字符串数据的软元件的起始编号 (BIN16 位)*1

Ⓓ：指令结束时使其 1 个扫描 ON 的软元件 (位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			$	
Ⓢ	--	○		--				○	--
Ⓓ	△*2	--		--				--	--

*1: 在 “(驱动器编号): (文件名)” 中指定。例 1: MAIN

*2: 局部软元件不能使用。

功 能

(1) 将存储卡、标准 ROM 中存储的程序传送到程序存储器 (驱动器 0) 中。
即使传送的程序未在可编程控制器参数的程序设置中登录，也在 CPU 模块内部的程序设置中将其设置为待机类型。
此时可编程控制器参数的程序设置不变化。
(通过 PLOADP 指令传送程序时，程序存储器中需要有连续的可用空间。)

(2) 通过 PLOADP 指令添加的程序将被分配为未使用的程序号中最小的号。
(由用户指定的情况下，应将程序号存储到 SD720 中。)
例如，通过 PLOADP 指令添加 “MAIN6” 的情况如下所示。

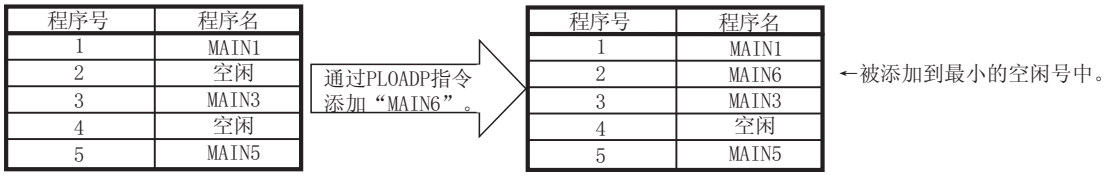
(a) 在程序号以连续方式设置的情况下，被添加到所设置程序号的最后。
程序号 1 ～ 5 已被设置了程序时，将新增程序号。

程序号	程序名
1	MAIN1
2	MAIN2
3	MAIN3
4	MAIN4
5	MAIN5

通过PLOADP指令添加 “MAIN6”。

程序号	程序名	
1	MAIN1	
2	MAIN2	
3	MAIN3	
4	MAIN4	
5	MAIN5	
6	MAIN6	←被添加到最后处。

(b) 在存在有多个未放入程序的空闲程序号的情况下，通过 PLOADP 指令指定的程序将被添加到最小的空闲程序号中。
（通过 PUNLOADP 指令删除了程序时将会产生空闲的程序号。）
在存在有空闲的程序号 2 及 4 的情况下，将被添加到程序号 2 中。



(3) 驱动器号可指定为 1、2、4。（不能指定为驱动器 3）

- 驱动器 1：存储卡 (RAM)
- 驱动器 2：存储卡 (ROM)
- 驱动器 4：标准 ROM

(4) 文件名中无需指定扩展名 (.QPG)。

(5) 在本指令结束的扫描的 END 处理中，将ⓓ中指定的位软元件置为 ON，在下一个 END 处理中置为 OFF。

(6) PLOADP/PUNLOADP/PSWAPP 指令不能同时执行。
如果同时执行了多个上述指令，后执行的指令将变为非执行。
使用上述指令时，用户应设置互锁以防止上述指令同时被执行。

(7) 不要在中断程序中执行本指令。
（如果在中断程序中执行了本指令，有可能导致误动作。）

(8) 希望执行通过本指令传送到程序存储器中的程序时，应通过 PSCAN 指令将其设置为“扫描执行型”。（参阅 7.16.3 项）

(9) 通过本指令传送的程序的可编程控制器文件设置方法如下所示。

(a) 各程序文件的使用方法
通过本指令传送的程序的文件寄存器、软元件初始值、注释、局部软元件的使用方法均为“按照可编程控制器文件设置”。
但是，通过本指令进行程序传送时，如果满足下述两个条件将会变为出错状态。

- 在可编程控制器文件设置中设置了“使用局部软元件”
- 程序存储器的程序个数超过了参数中设置的程序个数时

在通过本指令传送的程序中使用局部软元件时，应在参数中登录一个虚拟的程序文件。
然后通过 PUNLOADP 指令将该虚拟程序文件删除后，通过 PLOADP 指令进行装载。

(b) I/O 刷新设置
在通过本指令传送的程序的 I/O 刷新设置中，输入、输出均为“无设置”。

(10)“PLOADP 指令”与 “RUN 中写入”的处理不能同时执行。

(a) 在 PLOADP 指令的处理过程中，如果发生了 RUN 中写入请求，则 RUN 中写入将被等待。
PLOADP 指令的处理结束后，开始执行 RUN 中写入。

(b) 如果在进行 RUN 中写入的过程中执行 PLOADP 指令，则 PLOADP 指令的处理将被等待。
在 RUN 中写入处理结束后，开始进行 PLOADP 指令的处理。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

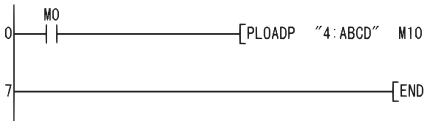
- Ⓢ中指定的驱动器号的文件名不存在时。 (出错代码： 2410)
- Ⓢ中指定的驱动器号为禁止指定的驱动器号时。 (出错代码： 4100)
- 驱动器 0 中的可用空间小于相应程序的装载容量时。 (出错代码： 2413)
- 在程序存储器中登录的程序个数已达到下表的文件个数时。 (出错代码： 4101)
- 存储到 SD720 中的程序号已被使用，或者超出了下表的程序号范围时。 (出错代码： 4101)
- 存在有与要装载的程序文件相同名称的程序文件时。 (出错代码： 2410)
- 可用空间小于局部软元件的文件容量时。 (出错代码： 2401)

CPU 型号	程序存储器 (文件个数)	最大程序号
Q02(H)CPU	28 个	28
Q06HCPU	60 个	60
Q12HCPU	124 个	124
Q25HCPU	124 个	124
Q12PHCPU	124 个	124
Q25PHCPU	124 个	124

程序示例

(1) 以下为 M0 变为 ON 时，将驱动器 4 中存储的 “ABCD.QPG”传送到驱动器 0 中，并置为待机状态的程序。

[梯形图模式]



[列表模式]

步	指令	软元件
0	LD	M0
1	PLOADP	"4:ABCD" M10
7	END	

9.8 从程序存储器中卸载程序 (PUNLOADP)

Basic High performance Process Redundant Universal

Ⓢ：要卸载的程序文件名的字符串数据或者存储字符串数据的软元件的起始编号 (BIN16 位)。

Ⓓ：指令结束时使其 1 个扫描 ON 的软元件 (位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
Ⓢ	--	○		--				○	--
Ⓓ	△*1	--		--				--	--

*1: 局部软元件不能使用。

功 能

(1) 将程序存储器 (驱动器 0) 中存储的待机程序从程序存储器中删除。
(不能删除通过 PSCAN 指令设置为 “扫描执行型” 的程序或者通过 PLOW 指令设置为 “低速执行型” 的程序。)

(2) 通过 PUNLOADP 指令删除的程序的程序号将变为 “空闲”。
在可编程控制器参数的程序设置中设置了程序号 1 ～ 5 时，如果通过本指令删除了程序号 2 的程序，则程序号将变为空闲。

程序号	程序名
1	MAIN1
2	MAIN2
3	MAIN3
4	MAIN4
5	MAIN5

通过PUNLOADP指令删除“MAIN2”。

程序号	程序名
1	MAIN1
2	空闲
3	MAIN3
4	MAIN4
5	MAIN5

←程序号2的程序被删除。

(3) 文件名中无需指定扩展名 (.QPG)。

(4) 在本指令结束的扫描的 END 处理中，将Ⓓ中指定的位软元件置为 ON，在下一个 END 处理中置为 OFF。

(5) PLOADP/PUNLOADP/PSWAPP 指令不能同时执行。
如果同时执行了多个上述指令，后执行的指令将变为非执行。
使用上述指令时，用户应设置互锁以防止上述指令同时被执行。

(6) 执行 PUNLOADP 指令后，进行可编程控制器的电源 OFF → ON 或者 CPU 模块的复位时的情况如下所示。

(a) 在可编程控制器参数中进行了引导设置的情况下，被进行了引导设置的程序将被传送到程序存储器中。
不再执行通过 PUNLOADP 指令删除的程序的情况下，应将相应程序名从可编程控制器参数的引导设置及程序设置中删除。

(b) 未在可编程控制器参数中进行引导设置的情况下，将变为 “FILE SET ERROR(出错代码 2400)”的出错状态。

1) 不再执行通过 PUNLOADP 指令删除的程序的情况下，应将相应程序名从可编程控制器参数的引导设置中删除。

2) 需要再次执行通过 PUNLOADP 指令删除的程序的情况下，应将相应程序写入到 CPU 模块中。

(7) 不要在中断程序中执行本指令。
(如果在中断程序中执行了本指令将无法保证动作正常。)

(8) 通过本指令从程序存储器中删除程序时，应事先通过 PSTOP 指令将其设置为 “待机执行型”。(参阅 7.16.1 项)

(9) “PUNLOADP 指令”与 “RUN 中写入”的处理不能同时执行。

(a) 在 PUNLOADP 指令的处理过程中，如果发生了 RUN 中写入请求，则 RUN 中写入将被等待。
PUNLOADP 指令的处理结束后，开始执行 RUN 中写入。

(b) 如果在进行 RUN 中写入的过程中执行 PUNLOADP 指令，则 PUNLOADP 指令的处理将被等待。
在 RUN 中写入处理结束后，开始进行 PUNLOADP 指令的处理。

出 错

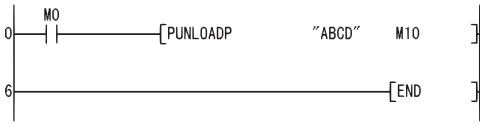
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- Ⓢ中指定的文件名不存在时。 (出错代码 : 2410)
- Ⓢ中指定的程序不是待机程序，或者是当前正在执行中的程序时。 (出错代码 : 4101)

程序示例

(1) 以下为 M0 由 OFF 变为 ON 时，将存储在驱动器 0 中的 “ABCD.QPG”从存储器中删除的程序。

[梯形图模式]

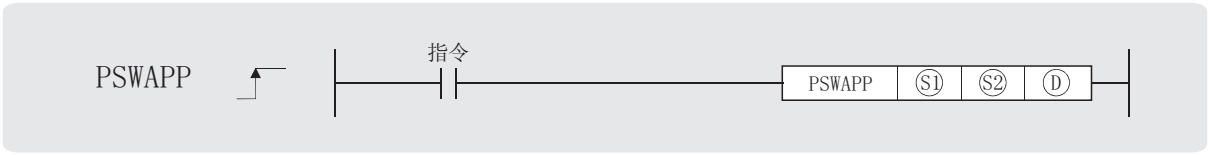


[列表模式]

步	指令	软元件
0	LD	M0
1	PUNLOADP	"ABCD" M10
6	END	

9.9 装载 + 卸载 (PSWAPP)

Basic | High performance | Process | Redundant | Universal



S1 ：要卸载的程序文件名的字符串数据或者存储字符串数据的软元件的起始编号 (BIN16 位)。

S2 ：存储要装载的程序的驱动器号、文件名的字符串数据或者存储字符串数据的软元件的起始编号 (BIN16 位)。*1

D ：指令结束时使其 1 个扫描 ON 的软元件 (位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
S1	--	○		--				○	--
S2	--	○		--				○	--
D	△ *2	--		--				--	--

*1: 在 “(驱动器编号): (文件名)” 中指定。例 1: MAIN

*2: 局部软元件不能使用。

功 能

(1) 将S1中指定的程序存储器 (驱动器 0) 中存储的待机型程序从程序存储器中删除。然后，将S2中指定的存储卡、标准 ROM 中存储的程序传送到程序存储器中，并置为待机状态。

(将程序传送到程序存储器中时，程序存储器中需要有连续的可用空间。)

此外，不能删除通过 PSCAN 指令设置为 “扫描执行型” 的程序或者通过 PLOW 指令设置为 “低速执行型” 的程序。

(2) 通过 PSWAPP 指令传送到程序存储器中的程序的程序号将变为从程序存储器中被删除的程序的程序号。

(即使在从程序存储器中被删除的程序的程序号前面存在有空闲的程序号，也不会分配给传送到程序存储器中的程序。)

例如，程序号 2 为 “空闲” 时，如果通过本指令对程序号 3 的程序进行了交换，传送到程序存储器中的程序将被登录为程序号 3。

程序号	程序名
1	MAIN1
2	空闲
3	MAIN3
4	MAIN4
5	MAIN5

通过PSWAPP指令将 “MAIN3” 与 “MAIN6” 进行交换。

程序号	程序名	
1	MAIN1	
2	空闲	
3	MAIN6	←变为MAIN6。
4	MAIN4	
5	MAIN5	

(3) 驱动器号可指定为 1、2、4。(不能指定为驱动器 3)

- 驱动器 1: 存储卡 (RAM)
- 驱动器 2: 存储卡 (ROM)
- 驱动器 4: 标准 ROM

(4) 文件名中无需指定扩展名 (.QPG)。

(5) 在本指令结束的扫描的 END 处理中，将ⓓ中指定的位软元件置为 ON，在下一个 END 处理中置为 OFF。

(6) PLOADP/PUNLOADP/PSWAPP 指令不能同时执行。
如果同时执行了多个上述指令，后执行的指令将变为非执行。
使用上述指令时，用户应设置互锁以防止上述指令同时被执行。

(7) 执行 PSWAPP 指令后，进行可编程控制器的电源 OFF → ON 或者 CPU 模块的复位时的情况如下所示。

(a) 在可编程控制器参数中进行了引导设置的情况下，被进行了引导设置的程序将被传送到程序存储器中。
执行通过 PSWAPP 指令替换的程序的情况下，应将可编程控制器参数的引导设置及程序设置变更为相应的程序名。

(b) 未在可编程控制器参数中进行引导设置的情况下，将变为 “FILE SET ERROR(出错代码 2400)” 的出错状态。
1) 执行通过 PSWAPP 指令替换的程序的情况下，应将可编程控制器参数的程序设置变更为相应的程序名。
2) 执行可编程控制器参数的程序设置中设置的程序的情况下，应将相应程序再次写入到 CPU 模块中。

(8) 不要在中断程序中执行本指令。
(如果在中断程序中执行了本指令将无法保证动作正常。)

(9) 执行了 PSWAPP 指令的程序的可编程控制器文件设置情况如下所示。

(a) 各程序文件的使用方法
执行了 PSWAPP 指令后的程序的文件寄存器、软元件初始值、注释、局部软元件的使用方法均为 “按照可编程控制器文件设置”。

(b) I/O 刷新设置
在执行了 PSWAPP 指令后的程序的 I/O 刷新设置中，输入、输出均为 “无设置”。

(10)“PSWAPP 指令” 与 “RUN 中写入” 的处理不能同时执行。

(c) 在 PSWAPP 指令的处理过程中，如果发生了 RUN 中写入请求，则 RUN 中写入将被等待。
SWAPP 指令的处理结束后，开始执行 RUN 中写入。

(d) 如果在进行 RUN 中写入的过程中执行 PSWAPP 指令，则 PSWAPP 指令的处理将被等待。
在 RUN 中写入处理结束后，开始进行 PSWAPP 指令的处理。

出 错

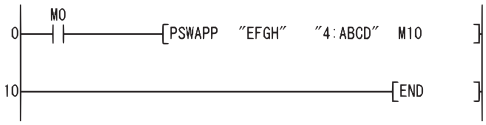
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⑤1、⑤2 中指定的驱动器号的文件名不存在时。 (出错代码：2410)
- ⑤1 中指定的驱动器号为禁止指定的驱动器号时。 (出错代码：4100)
- 驱动器 0 中的可用空间小于相应程序的装载容量时。 (出错代码：2413)
- ⑤1 中指定的程序不是待机程序，或者是当前正在执行中的程序时。 (出错代码：4101)

程序示例

(1) 以下为 M0 由 OFF 变为 ON 时，将存储在驱动器 0 中的 “EFGH.QPG”从存储器中删除的同时，将驱动器 4 中存储的 “ABCD.QPG”传送到驱动器 0 中的程序。

[梯形图模式]

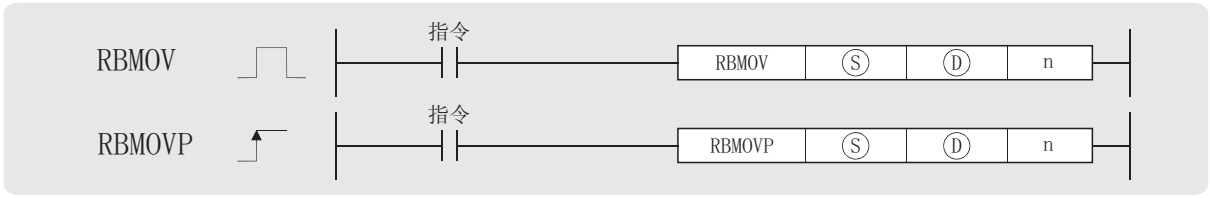


[列表模式]

步	指令	软元件
0	LD	M0
1	PSWAPP	"EFGH" "4:ABCD" M10
10	END	

9.10 文件寄存器的高速块传送(RBMOV(P))

Basic | High performance | Process | Redundant | Universal

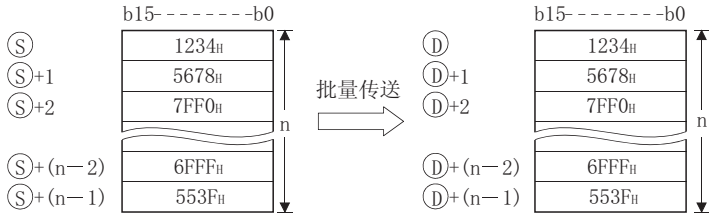


Ⓢ ：存储传送数据的软元件的起始编号(BIN16 位)。
Ⓓ ：传送目标的软元件的起始编号(BIN16 位)。
n ：传送数(BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	○						--		--
Ⓓ	○						--		--
n	○						○		--

功 能

(1) 将从Ⓢ中指定的软元件开始的 n 点的 16 位数据，批量地传送到Ⓓ中指定的软元件开始的 n 点中。



(2) 即使传送源与传送目标的软元件重复时，也可进行传送。
向软元件编号较小的一方传送时，从Ⓢ开始进行传送，向软元件编号较大的一方传送时，从Ⓢ+(n-1) 开始进行传送。

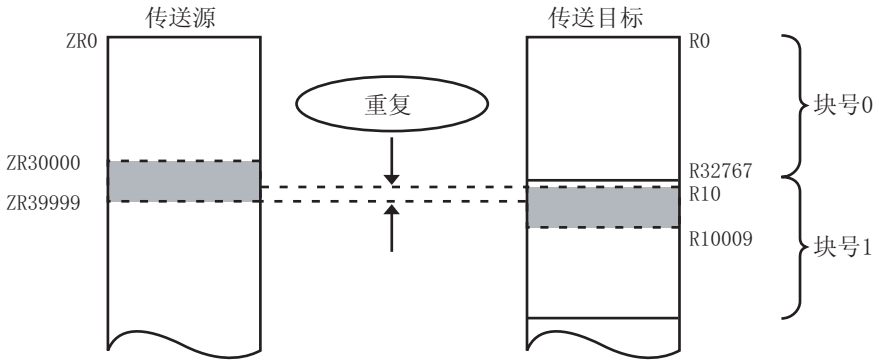
但是，在从 R 传送至 ZR 或者从 ZR 传送至 R 中时，应按如下所示那样注意避免 ZR 与 Z 的各传送范围重复。

- ZR 的传送范围((指定的 ZR 起始号)～(指定的 ZR 起始号 + 传送数 -1))
- R 的传送范围((指定的 R 起始号 + 文件寄存器号× 32768)～
(指定的 R 起始号 + 文件寄存器号× 32768+ 传送数 -1))

例 将 10000 点的数据从传送源的 ZR30000 传送至传送目标的程序号 1 的 R10 中时，传送范围重复。

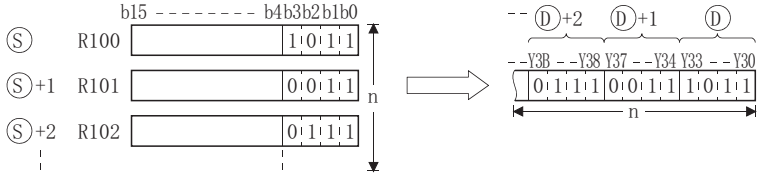
- ZR 的传送范围 → (30000) ～ (30000+10000-1) → (30000) ～ (39999)
- R 的传送范围 → (10+(1 × 32768)) ～ (10+(1 × 32768)+10000-1)
  → (32778) ～ (42777)

因此，从 32778 起至 39999 为止的范围重复。



(3) 在Ⓢ为字软元件而Ⓓ为位软元件的情况下，字软元件将成为位软元件的位数指定中指定的位数对象。

在Ⓓ中指定了 K1Y30 的情况下，Ⓢ中指定的字软元件的低 4 位将成为对象。



出 错

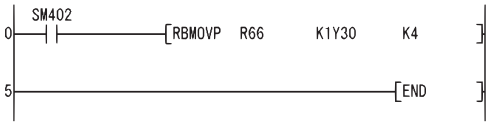
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- Ⓢ、Ⓓ开始的 n 点的软元件范围超出了相应软元件的范围时。 (出错代码： 4101)
- Ⓢ、Ⓓ中的某一个中未指定文件寄存器时。 (出错代码： 4101)

程序示例

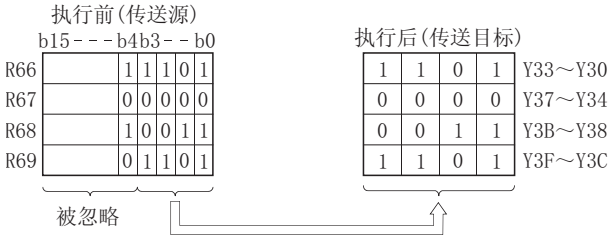
(1) 以下为将 R66 ～ R99 的低 4 位的数据以 4 点为单位从 Y30 输出到 Y3F 中的程序。

[梯形图模式]



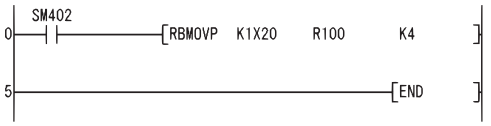
[列表模式]

步	指令	软元件		
0	LD	SM402		
1	RBMOVP	R66	K1Y30	K4
5	END			



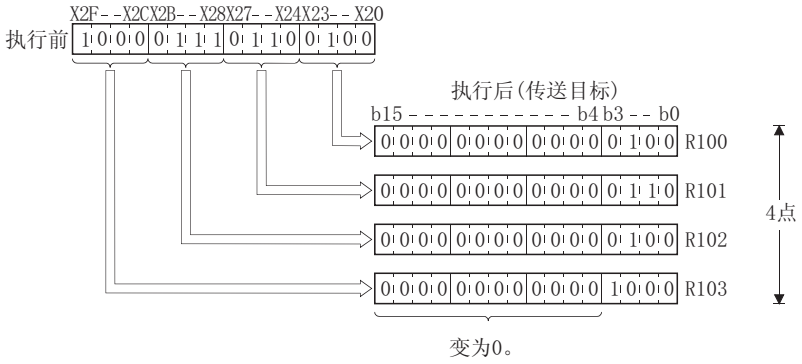
(2) 以下为将 X20 ～ X2F 的数据以 4 点为单位输出到 R100 ～ R103 中的程序。

[梯形图模式]



[列表模式]

步	指令	软元件		
0	LD	SM402		
1	RBMOVP	K1X20	R100	K4
5	END			



要 点

本指令在 QnHCPU/QnPHCPU/QnPRHCPU 中对文件寄存器数据进行大量批量传送时有效。在 QnUCPU 的情况下，处理速度与 BMOV 指令相同。
与 BMOV 指令的处理速度比情况如下所示。

(1) 文件寄存器→内部软元件 / 内部软元件→文件寄存器的情况下。

CPU	文件寄存器存储对象存储器	RBMOV			BMOV	
		1 字	1000 字	10000 字	1 字	10000 字
QnHCPU QnPHCPU QnPRHCPU	标准 RAM	20.0 μs	91.0 μs	775.0 μs	7.5 μs	720.0 μs
	SRAM 卡	22.0 μs	305.0 μs	2900.0 μs	8.0 μs	3900.0 μs
	Flash 卡 *1	22.5 μs	405.0 μs	3950.0 μs		4250.0 μs
QnCPU	标准 RAM	45.5 μs	215.0 μs	1850.0 μs	17.5 μs	1700.0 μs
	SRAM 卡	49.5 μs	540.0 μs	5150.0 μs	18.0 μs	5050.0 μs
	Flash 卡 *1					5800.0 μs

*1 ：将文件寄存器存储到 Flash 卡中时，内部软元件→文件寄存器的模式将无处理。

(2) 文件寄存器→文件寄存器的情况下

CPU	文件寄存器存储对象存储器	RBMOV			BMOV	
		1 字	1000 字	10000 字	1 字	10000 字
QnHCPU QnPHCPU QnPRHCPU	标准 RAM	20.0 μs	91.0 μs	775.0 μs	7.5 μs	720.0 μs
	SRAM 卡	22.5 μs	545.0 μs	5300.0 μs	8.5 μs	7050.0 μs
QnCPU	标准 RAM	45.5 μs	215.0 μs	1850.0 μs	17.5 μs	1700.0 μs
	SRAM 卡	50.0 μs	870.0 μs	8350.0 μs	18.5 μs	8600.0 μs

9.11 写入自站 CPU 共享存储器

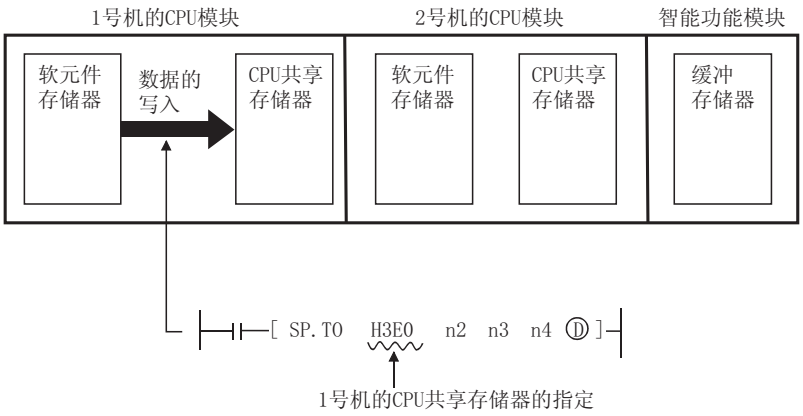
在多 CPU 系统中对自站 CPU 共享存储器进行写入时，通过 S.TO 指令或者 TO 指令进行。

S.TO 指令及 TO 指令的使用可否如下表所示。

CPU 模块型号		S.TO 指令	TO 指令
基本型 QCPU	Q00CPU、Q01CPU	可以使用	可以使用
高性能型 QCPU	Q02CPU、Q02HCPU、 Q06HCPU、Q12HCPU、 Q25HCPU	可以使用	不能使用
过程 CPU	Q02PHCPU、Q06PHCPU、 Q12PHCPU、Q25PHCPU	可以使用	不能使用
冗余 CPU	Q12PRHCPU、Q25PRHCPU	不能使用	不能使用
通用型 QCPU	Q00UCPU、Q01UCPU、Q02UCPU、 Q03UDCPU、Q04UDHCPU、Q06UDHCPU、 Q10UDHCPU, Q13UDHCPU、Q20UDHCPU, Q26UDHCPU、Q03UDECPU、Q04UDEHCPU、 Q06UDEHCPU、Q10UDEHCPU、Q13UDEHCPU、 Q20UDEHCPU、Q26UDEHCPU	可以使用	可以使用

(1) S.TO 指令的动作

通过 S.TO 指令，可以将数据写入到自站 CPU 模块的 CPU 共享存储器中。
在 1 号机的 CPU 模块中，执行 S.TO 指令时的处理如下图所示。



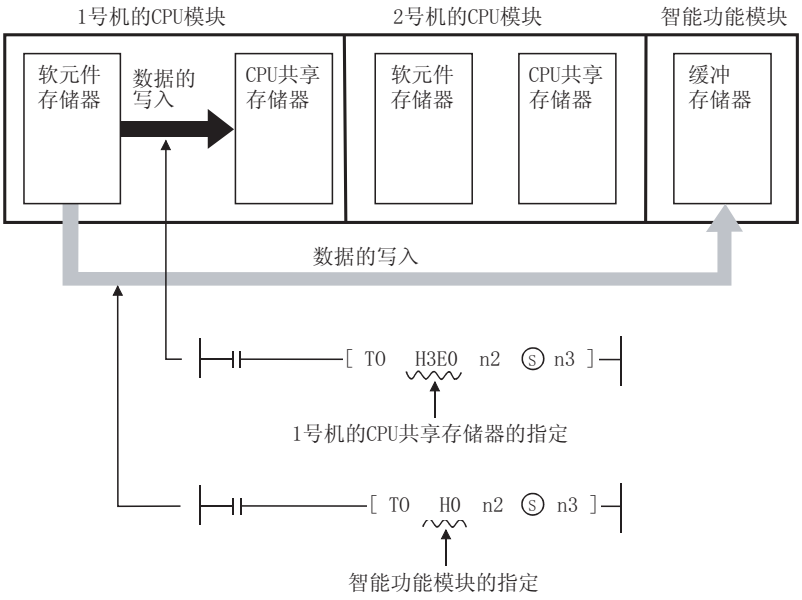
(2) TO 指令的动作

通过 TO 指令，可以将软元件存储器的数据写入到下述存储器中。

• 自站 CPU 模块的 CPU 共享存储器

• 智能功能模块的缓冲存储器

在 1 号机的 CPU 模块中，执行了 TO 指令时的处理如下图所示。



要 点

在基本型 QCPU(Q00CPU、Q01CPU) 与通用型 QCPU 中，使用 S.TO 指令或 TO 指令均可对 CPU 共享存储器进行写入，但对自站 CPU 的共享存储器进行写入时，建议使用 TO 指令，因为于可以减少步数及处理时间。

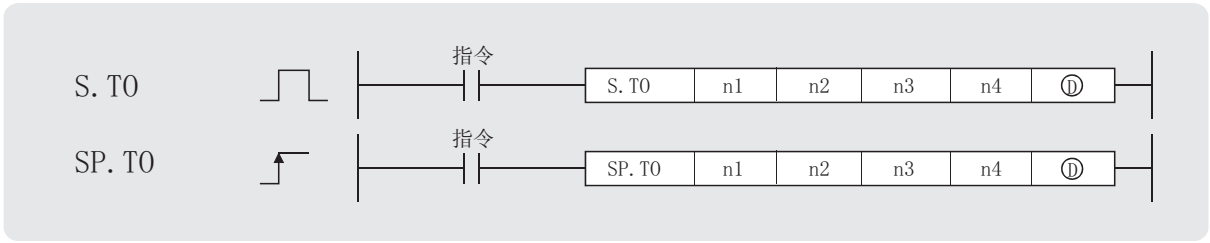
备 注

通过 TO 指令智能功能模块的缓冲存储器进行写入时，请参阅 7.8.2 项。

9.11.1 写入到自站 CPU 共享存储器 (S(P).TO)

Ver. Basic Ver. High performance Process Redundant Universal

基本型 QCPU：序列号的前 5 位数为 “04122” 以后
高性能型 QCPU：功能版本 B 以后



n1 ：自站的起始 I/O 编号 (BIN16 位)。
n2 ：写入目标的自站 CPU 共享存储器地址 (BIN16 位)。
- 基本型 QCPU：0 ～ 511
- 高性能型 QCPU、过程 CPU、通用型 QCPU：0 ～ 4095

n3 ：存储写入数据的软元件的起始编号 (BIN16 位)。
n4 ：写入数据数 (BIN16 位)。
- 基本型 QCPU：1 ～ 320
- 高性能型 QCPU、过程 CPU：1 ～ 256
- 通用型 QCPU：0 ～ 2048

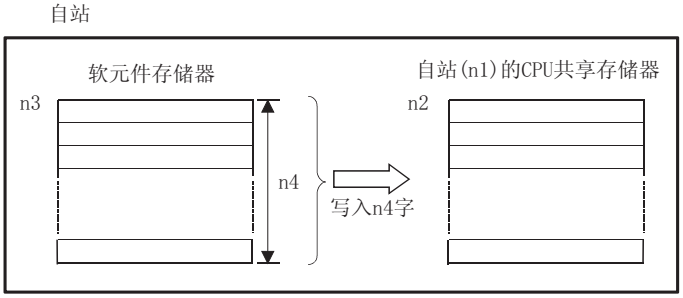
Ⓓ ：写入结束时使其 1 个扫描 ON 的软元件 (位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
n1	--	○		--				○	--
n2	--	○		--				○	--
n3	--	○		--				--	--
n4	--	○		--				○	--
Ⓓ	○	○		--				--	--

功 能

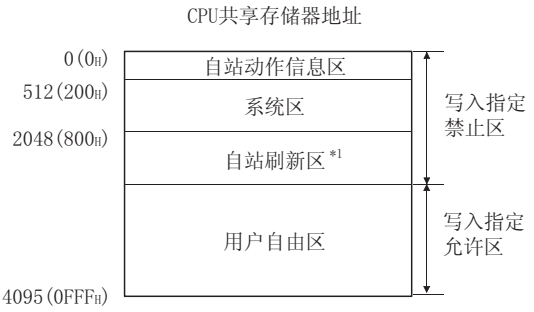
(1) 从自站 CPU 模块的 n3 开始，将 n4 字的软元件数据写入到自站 CPU 模块的 n2 中指定的 CPU 共享存储器地址的后面。

写入结束时，Ⓓ中指定的结束位将变为 ON。



(a) 基本型 QCPU 时的 CPU 共享存储器地址

(b) 高性能型 QCPU、过程 CPU、通用型 QCPU*2 时的 CPU 共享存储器地址



*1: 未进行自动刷新设置的情况下，可以作为用户自由区使用。
此外，即使进行了自动刷新设置，自动刷新发送范围后面的区域也可作为用户自由区使用。

*2: 不能通过 S(P).TO 指令对通用型 QCPU 的多 CPU 高速通信区进行写入。

(2) 写入点数为 0 时，将变为无处理且结束软元件也不变为 ON。

(3) 每个站的 1 个扫描中只能执行一个 S.TO 指令。
在有 2 处以上的执行条件同时成立的情况下，由于将自动进行握手，因此后执行的 S.TO 指令将不进行处理。

(4) 写入数据数根据对象 CPU 模块而有所不同。

CPU 模块	写入点数
基本型 QCPU	1 ～ 320
高性能型 QCPU、过程 CPU	1 ～ 256
通用型 QCPU	1 ～ 2048

要 点

对 CPU 共享存储器进行数据写入时，也可使用智能功能模块软元件进行写入。
关于智能功能模块软元件，请参阅所使用的 CPU 模块的用户手册（功能解说 / 程序基础篇）。

出 错

在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

(1) 指定的数据超出了下述范围时。 (出错代码： 4101)

- 写入点数 (n4) 超出了设置数据的指定范围时。
- 写入目标的自站 CPU 共享存储器地址 (n2) 的起始超出了 CPU 共享存储器地址的范围时。
- 写入目标的自站 CPU 共享存储器地址 (n2)+ 写入点数 (n4) 超出了 CPU 共享存储器地址的范围时。
- 存储写入数据的起始软元件编号 (n3)+ 写入点数 (n4) 超出了软元件的范围时。

(2) 写入目标的自站 CPU 共享存储器地址 (n2) 中指定了自站动作信息区、系统区或者自站刷新区时。

(高性能型 QCPU、过程 CPU) (出错代码： 4101)
(基本型 QCPU、通用型 QCPU) (出错代码： 4111)

(3) 自站的起始 I/O 编号 (n1) 中指定了自站以外时。

(高性能型 QCPU、过程 CPU) (出错代码： 2107)
(基本型 QCPU、通用型 QCPU) (出错代码： 4112)

(4) CPU 模块的起始 I/O 编号中指定的位置上 CPU 模块不存在时。 (出错代码： 2110)

(5) 自站的起始 I/O 编号 (n1) 中，指定了除 3E0H/3E1H/3E2H/3E3H 以外时。 (出错代码： 4100)

(6) 指定的指令不正确时。 (出错代码： 4002)

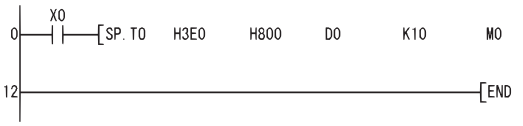
(7) 软元件数有误时。 (出错代码： 4003)

(8) 指定了不能使用的软元件时。 (出错代码： 4004)

程序示例

(1) 以下为 X0 由 OFF 变为 ON 时，将从 D0 开始 10 点的数据存储到 1 号机的 CPU 共享存储器的 800H 地址号中的程序。

[梯形图模式]



[列表模式]

步	指令	软元件				
0	LD	X0				
1	SP.TO	H3E0	H800	D0	K10	M0
12	END					

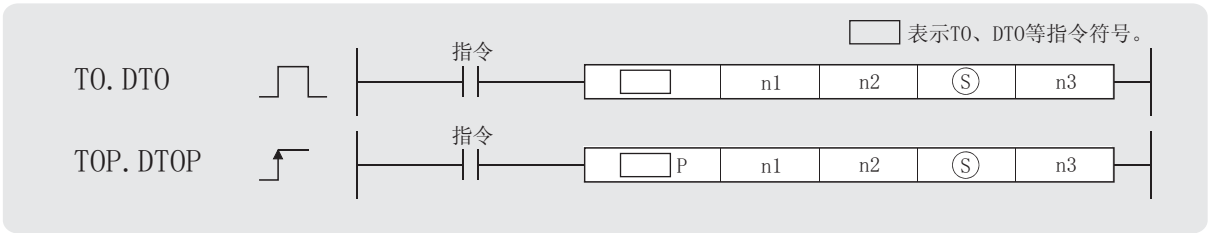
备 注

进行 n1 指定时，是将安装了 CPU 模块的插槽的起始 I/O 编号以 4 位数的 16 进制数表示时的高 3 位进行指定。

	CPU 插槽	插槽 0	插槽 1	插槽 2
起始 I/O 编号	3E00	3E10	3E20	3E30
n1	3E0	3E1	3E2	3E3

9.11.2 写入到自站 CPU 共享存储器 (TO(P)、DTO(P))

基本型 QCPU：序列号的前 5 位数为 “04122” 以后



n1 ：自站的起始 I/O 编号 (BIN16 位)。
- 基本型 QCPU ：3E0H
- 通用型 QCPU:3E0H ～ 3E3H

n2 ：写入目标的自站 CPU 共享存储器地址 (BIN16 位)。
- 基本型 QCPU ：192 ～ 511
- 通用型 QCPU:2048 ～ 4095、10000 ～ 24335*1

Ⓢ ：写入数据或者存储写入数据的软元件的起始编号 (BIN16 位)。

n3 ：写入数据数 (BIN16 位)。
- 基本型 QCPU:TO(P):1 ～ 320、DTO(P):1 ～ 160
- 通用型 QCPU:TO(P):1 ～ 14336*1、DTO(P):1 ～ 7168*1

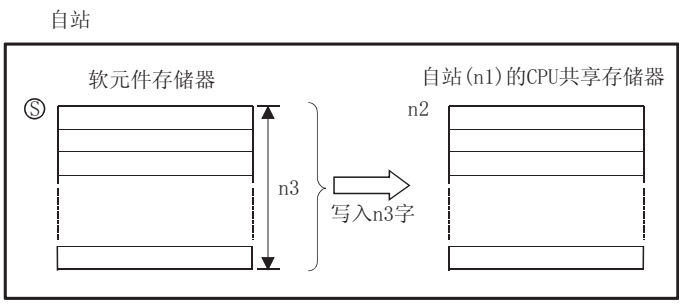
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它 U
	位	字		位	字				
n1		○			○			○	○
n2		○			○			○	--
Ⓢ		○			--			○	--
n3		○			○			○	--

*1: 设置范围取决于多 CPU 高速通信功能的自动刷新设置范围。

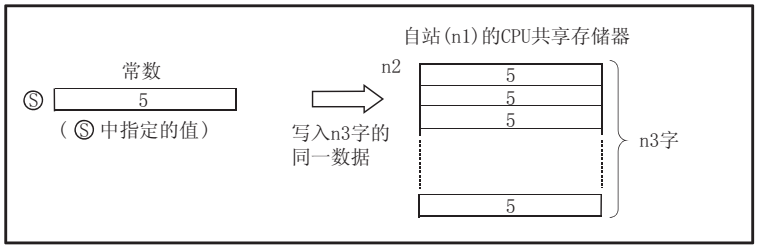
功 能

TO

(1) 将自站 CPU 模块的Ⓢ开始的 n3 字的软元件数据，写入到自站 CPU 模块的 n2 中指定的 CPU 共享存储器地址的后面。

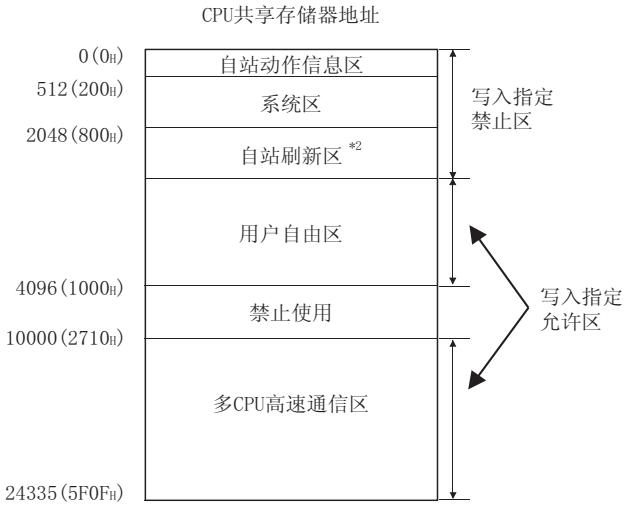


Ⓢ中指定了常数时，将 n3 字的同一数据（Ⓢ中指定的值 ）写入到从指定 CPU 共享存储器开始的 n3 字区域中。



(a) 基本型 QCPU 时的 CPU 共享存储器地址

(b) 高性能型 QCPU 时的 CPU 共享存储器地址 *3



*2: 未进行自动刷新设置的情况下，可以作为用户自由区使用。
此外，即使进行了自动刷新设置，自动刷新发送范围后面的区域也可作为用户自由区使用。

*3: 在 Q02UCPU 中，不能对多 CPU 高速通信区进行写入。

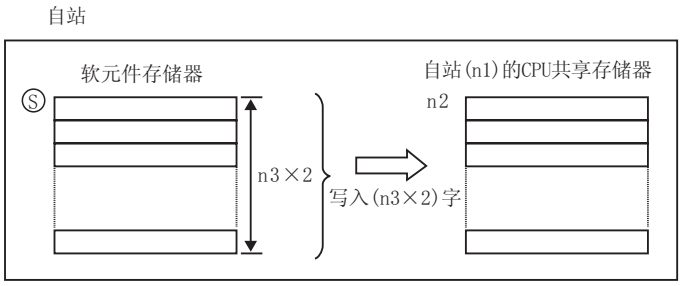
(2) 写入点数为 0 时，将变为无处理。

(3) 写入数据数根据对象 CPU 模块而有所不同。

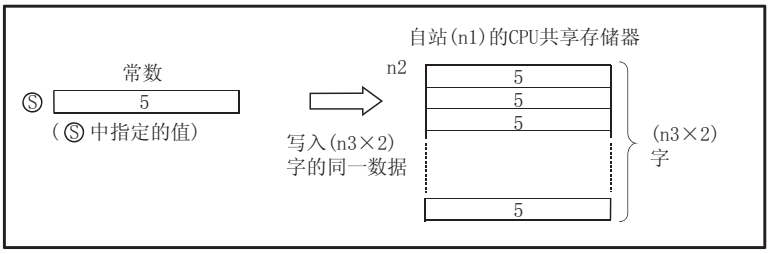
CPU 模块	写入点数
基本型 QCPU	1 ～ 320
通用型 QCPU	1 ～ 14336

DTO

(1) 将自站 CPU 模块的Ⓢ开始的 (n3 × 2) 字的软元件数据，写入到自站 CPU 模块的 n2 中指定的 CPU 共享存储器地址的后面。



Ⓢ中指定了常数时，将 (n3 × 2) 字的同一数据（Ⓢ中指定的值）写入到从指定 CPU 共享存储器开始的 (n3 × 2) 字区域中。



(2) 写入点数为 0 时，将变为无处理。

(3) 写入数据数根据对象 CPU 模块而有所不同。

CPU 模块	写入点数
基本型 QCPU	1 ～ 160
通用型 QCPU	1 ～ 7168

要 点

对 CPU 共享存储器进行数据写入时，也可使用智能功能模块软元件进行写入。
关于智能功能模块软元件，请参阅所使用的 CPU 模块的用户手册（功能解说 / 程序基础篇）。

出 错

在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

(1) 指定的数据超出了下述范围时。 (出错代码 : 4101)

- 写入点数 (n3) 超出了设置数据的指定范围时。
- 写入目标的自站 CPU 共享存储器地址 (n2)+ 写入点数 (n3) 超出了 CPU 共享存储器地址的范围时。
- 存储写入数据的起始软元件编号 (Ⓢ)+ 写入点数 (n3) 超出了软元件的范围时。
- 写入目标的自站 CPU 共享存储器地址 (n2) 的起始值的指定超出了允许写入的区域时。

(2) 写入目标的自站 CPU 共享存储器地址 (n2) 的起始值不是有效值时。 (出错代码 : 4111)

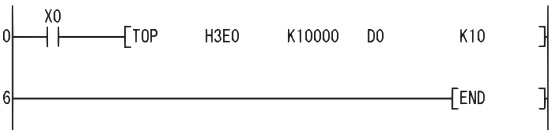
(3) (n1) 中指定了除自站以外时。(但是，指定了其它站的多 CPU 高速通信区时除外。) (出错代码 : 4112)

(4) CPU 模块的起始 I/O 编号中指定的位置上 CPU 模块不存在时。 (出错代码 : 2110)

程序示例

(1) 以下为 X0 变为 ON 时，将从 D0 开始 10 点的数据存储到 1 号机的 CPU 共享存储器的 10000 号后面的程序。

[梯形图模式]

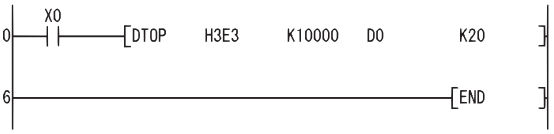


[列表模式]

步	指令	软元件			
0	LD	X0			
1	TOP	H3E0	K10000	D0	K10
6	END				

(2) 以下为 X0 变为 ON 时，将从 D0 开始 20 点的数据存储到 4 号机的 CPU 共享存储器的 10000 号后面的程序。

[梯形图模式]



[列表模式]

步	指令	软元件			
0	LD	X0			
1	DTOP	H3E3	K10000	D0	K20
6	END				

备 注

进行 n1 指定时，是将安装了 CPU 模块的插槽的起始 I/O 编号以 4 位数的 16 进制数表示时的高 3 位进行指定。

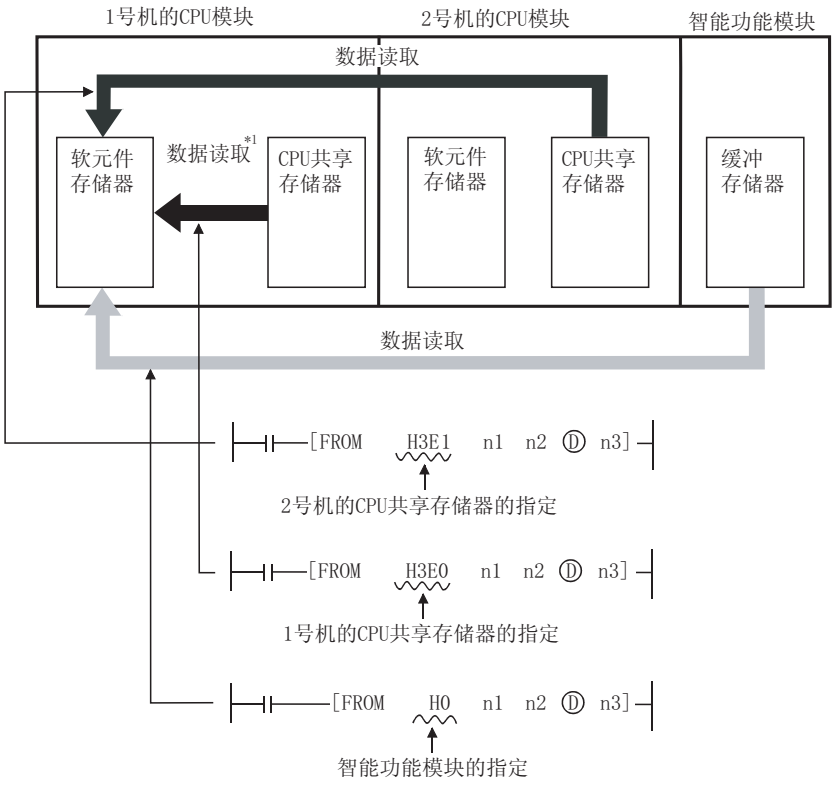
	CPU 插槽	插槽 0	插槽 1	插槽 2
起始 I/O 编号	3E00	3E10	3E20	3E30
n1	3E0	3E1	3E2	3E3

9.12 从其它站 CPU 共享存储器中读取数据

通过在多 CPU 系统中使用 FROM(P)/DFRO(P) 指令，可以从下述存储器中进行读取。

- 智能功能模块的缓冲存储器
- 其它站 CPU 模块的 CPU 共享存储器
- 自站 CPU 模块的 CPU 共享存储器（只在基本型 QCPU、通用型 QCPU 中可以执行）

在 1 号机中执行了 FROM(P) 指令时的处理如下图所示。



*1: 在基本型 QCPU、通用型 QCPU 中可以执行。

备 注

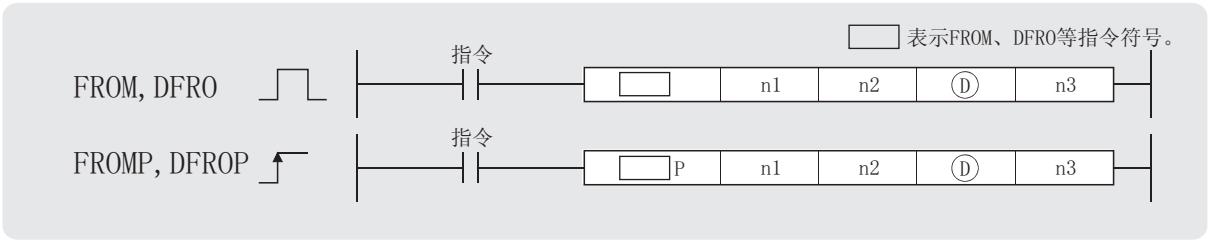
通过 FROM/DFRO 指令对智能功能模块的缓冲存储器进行读取时，请参阅 7.8.1 项。

9.12.1 从其它站共享存储器读取数据(FROM(P)、DFRO(P))

Ver. Basic　Ver. High performance　Process　Redundant　Universal

基本型 QCPU：序列号的前 5 位数为 “04122” 以后
高性能型 QCPU：功能版本 B 以后

1 使用基本型 QCPU、通用型 QCPU 时



n1 ： 读取对象 CPU 模块的起始 I/O 编号 (BIN16 位)。
- 基本型 QCPU：3E0H ～ 3E2H
- 通用型 QCPU：3E0H ～ 3E3H

n2 ： 读取目标的自站 CPU 共享存储器地址 (BIN16 位)。
- 基本型 QCPU：0 ～ 512
- 通用型 QCPU：0 ～ 4095, 10000 ～ 24335*1

Ⓓ ： 存储读取数据的软元件的起始编号 (BIN16 位)。

n3 ： 读取数据数 (BIN16 位)。
- 基本型 QCPU：FROM(P)：1 ～ 512、DFRO(P)：1 ～ 256
- 通用型 QCPU：FROM(P)：1 ～ 14336*1、DFRO(P)：1 ～ 7168*1

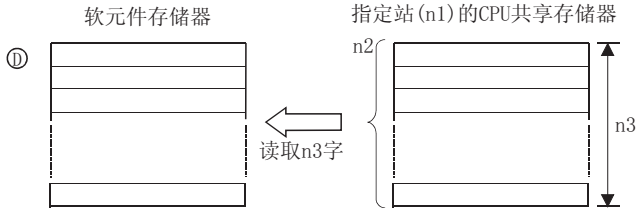
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			K、H	U
n1	--	○		○				○	○
n2	--	○		○				○	--
Ⓓ	--	○		--				--	--
n3	--	○		○				○	--

*1: 设置范围取决于多 CPU 高速通信功能的自动刷新设置范围。

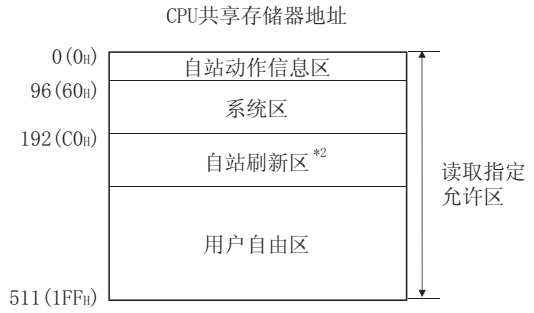
功 能

FROM

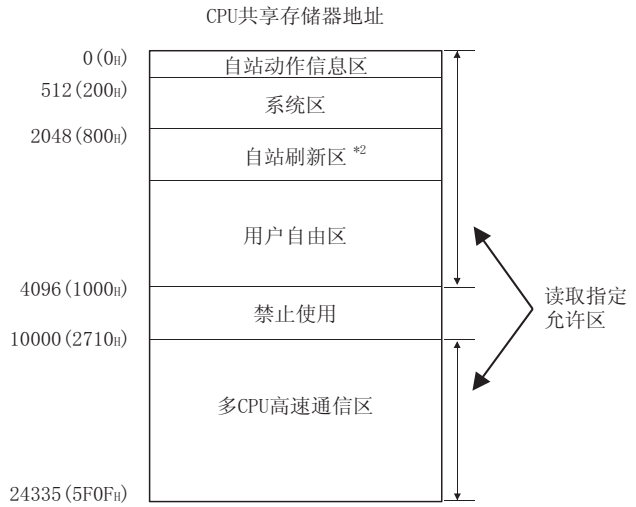
(1) 从 n1 中指定的 CPU 模块的 n2 中指定的 CPU 共享存储器地址开始，读取 n3 字的数据后，存储到Ⓓ中指定的软元件后面。



(a) 基本型 QCPU 时的 CPU 共享存储器地址



(b) 通用型 QCPU 时的 CPU 共享存储器地址 *3



*2: 未进行自动刷新设置的情况下，可以作为用户自由区使用。
此外，即使进行了自动刷新设置，自动刷新发送范围后面的区域也可作为用户自由区使用。

*3: 在 Q02UCPU 中，不能对多 CPU 高速通信区进行读取。

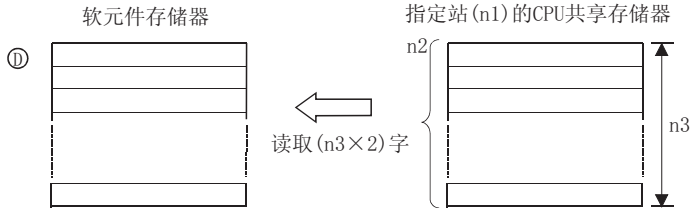
(2) 读取数据 n3 为 0 时，将变为无处理。

(3) 读取数据数根据对象 CPU 模块而有所不同。

CPU 模块	写入点数
基本型 QCPU	1 ～ 512
通用型 QCPU	1 ～ 14336

DFRO

(1) 将 n1 中指定的 CPU 模块的 n2 中指定的 CPU 共享存储器地址中，读取 (n3 × 2) 字的数据后，存储到Ⓓ中指定的软元件后面。



(2) 读取数据 n3 为 0 时，将变为无处理。

(3) 读取数据数根据对象 CPU 模块而有所不同。

CPU 模块	写入点数
基本型 QCPU	1 ～ 256
通用型 QCPU	1 ～ 7168

要 点

对 CPU 共享存储器进行数据写入时，也可使用智能功能模块软元件进行写入。
关于智能功能模块软元件，请参阅所使用的 CPU 模块的用户手册（功能解说 / 程序基础篇）。

出 错

在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

(1) 指定的数据超出了下述范围时。 （出错代码： 4101）

- 进行读取的 CPU 共享存储器地址 (n2) 的起始超出了 CPU 共享存储器的范围时。
- 进行读取的 CPU 共享存储器地址 (n2)+ 读取点数 (n3) 超出了 CPU 共享存储器的范围时。
- 读取数据存储软元件编号 (D)+ 读取点数 (n3) 超出了指定软元件的范围时。

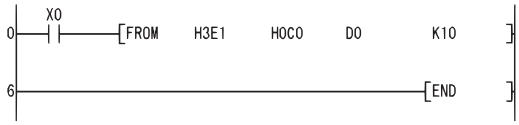
(2) CPU 模块的起始 I/O 编号中指定的位置上 CPU 模块不存在时。 （出错代码： 2110）

(3) 进行读取的 CPU 共享存储器地址 (n2) 的起始值不是有效值时。(4097 ～ 9999)
（出错代码： 4101）

程序示例

(1) 以下为 X0 变为 ON 时，将 2 号机的 CPU 共享存储器的 C0H 地址号开始的 10 点数据存储到 D0 后面的程序。

[梯形图模式]

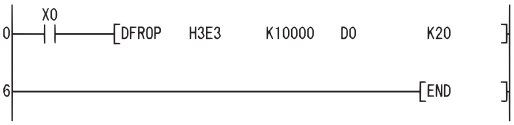


[列表模式]

步	指令	软元件			
0	LD	X0			
1	FROM	H3E1	H0C0	D0	K10
6	END				

(2) 以下为 X0 变为 ON 时，将 4 号机的 CPU 共享存储器的 10000 地址号开始的 20 点数据存储到 D0 后面的程序。

[梯形图模式]



[列表模式]

步	指令	软元件			
0	LD	X0			
1	DFROP	H3E3	K10000	D0	K20
6	END				

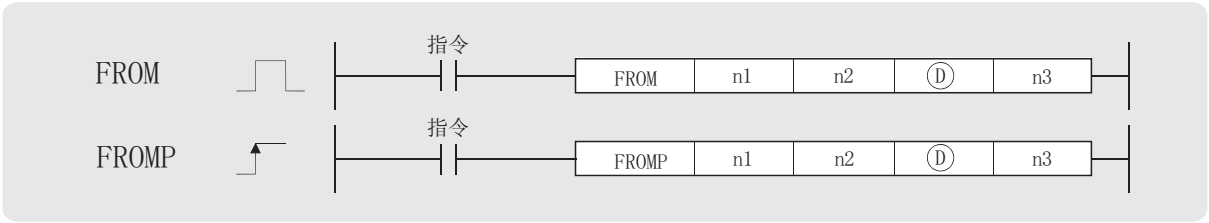
备 注

进行 n1 指定时，是将安装了 CPU 模块的插槽的起始 I/O 编号以 4 位数的 16 进制数表示时的高 3 位进行指定。

	CPU 插槽	插槽 0	插槽 1	插槽 2
起始 I/O 编号	3E00	3E10	3E20	3E30
n1	3E0	3E1	3E2	3E3

对 FROM/TO 指令进行了自动互锁。

2 使用高性能型 QCPU、过程 CPU 时

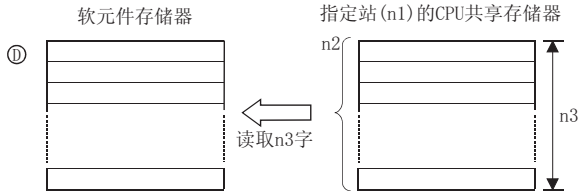


n1 ： 读取对象 CPU 模块的起始 I/O 编号 (3E0H ～ 3E3H)(BIN16 位)。
n2 ： 读取目标的 CPU 共享存储器地址 (0 ～ 4095)(BIN16 位)。
Ⓓ ： 存储读取的数据的软元件的起始编号 (BIN16 位)。
n3 ： 读取数据数 (1 ～ 4096)(BIN16 位)。

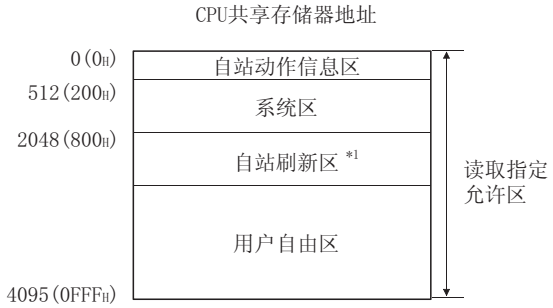
设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			K、H	U
n1	--	○		○				○	○
n2	--	○		○				○	--
Ⓓ	--	○		--				--	--
n3	--	○		○				○	--

功 能

(1) 从 n1 中指定的 CPU 模块的 n2 中指定的 CPU 共享存储器地址开始，读取 n3 字的数据后，存储到Ⓓ中指定的软元件后面。



高性能型 QCPU、过程 CPU 时的 CPU 共享存储器地址



*1: 未进行自动刷新设置的情况下，可以作为用户自由区使用。
此外，即使进行了自动刷新设置，自动刷新发送范围后面的区域也可作为用户自由区使用。

(2) 读取数据 n3 为 0 时，将变为无处理。

(3) 读取数据数根据对象 CPU 模块而有所不同。

CPU 模块	写入点数
高性能型 QCPU 过程 CPU	1 ～ 4096

要 点

对 CPU 共享存储器进行数据读取时，也可使用智能功能模块软元件进行写入。
关于智能功能模块软元件，请参阅所使用的 CPU 模块的用户手册（功能解说 / 程序基础篇）。

出 错

在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

(1) 指定的数据超出了下述范围时。 （出错代码： 4101）

- 进行读取的 CPU 共享存储器地址 (n2) 的起始超出了 CPU 共享存储器的范围时。
- 进行读取的 CPU 共享存储器地址 (n2)+ 读取点数 (n3) 超出了 CPU 共享存储器的范围时。
- 读取数据存储软元件编号 (D)+ 读取点数 (n3) 超出了指定软元件的范围时。

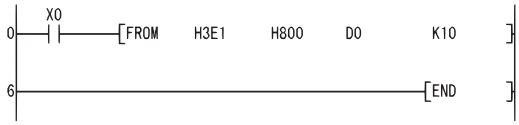
(2) CPU 模块的起始 I/O 编号中指定的位置上 CPU 模块不存在时。 （出错代码： 2110）

(3) 进行读取的 CPU 共享存储器地址 (n2) 的起始值不是有效值时。(4097 ～ 9999)
（出错代码： 4101）

程序示例

(1) 以下为 X0 变为 ON 时，将 2 号机的 CPU 共享存储器的 800H 地址号开始的 10 点数据存储到 D0 后面的程序。

[梯形图模式]



[列表模式]

步	指令	软元件
0	LD	X0
1	FROM	H3E1 H800 D0 K10
6	END	

备 注

进行 n1 指定时，是将安装了 CPU 模块的插槽的起始 I/O 编号以 4 位数的 16 进制数表示时的高 3 位进行指定。

	CPU 插槽	插槽 0	插槽 1	插槽 2
起始 I/O 编号	3E00	3E10	3E20	3E30
n1	3E0	3E1	3E2	3E3

对 FROM/TO 指令进行了自动互锁。

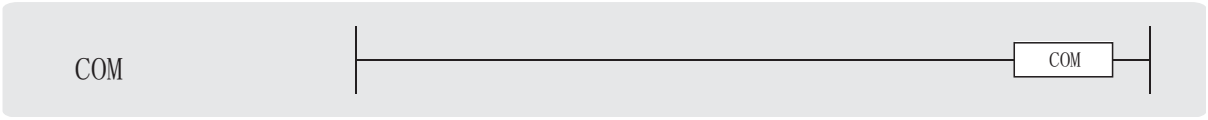
9.13 选择刷新指令 (COM)

关于下述 CPU 模块的 COM 指令的内容，请参阅 7.6.9 项。

- 序列号 04121 以前的基本型 QCPU
- 序列号 04011 以前的的高性能型 QCPU
- 序列号 07031 以前的过程 CPU

Ver. Basic　Ver. High performance　Ver. Process　Redundant　Universal

基本型 QCPU：序列号的前 5 位数为 “04122” 以后
高性能型 QCPU：序列号的前 5 位数为 “04012” 以后
过程 CPU：序列号的前 5 位数为 “07032” 以后



设置数据	内部软元件		Z, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
--	--								

功　能

(1) 如果执行 COM 指令，可以进行下述刷新。

- I/O 刷新
- CC-Link 的刷新
- CC-Link IE 控制网络的刷新
- MELSECNEN/H 的刷新
- 智能功能模块的自动刷新
- 使用了多 CPU 系统的 QCPU 标准区的自动刷新
- 多 CPU 系统的组外的输入 / 输出的获取
- 使用了多 CPU 系统的多 CPU 高速通信区的自动刷新
- 与外围设备的通信

备 注

在与外围设备的通信中也进行如下处理。

- 其它站监视处理
- 在串行通信模块中，对其它智能功能模块的缓冲存储器进行的读取处理。

(2) 如果将 SM775 置为 OFF，将对除 I/O 刷新以外的所有刷新项目进行刷新。

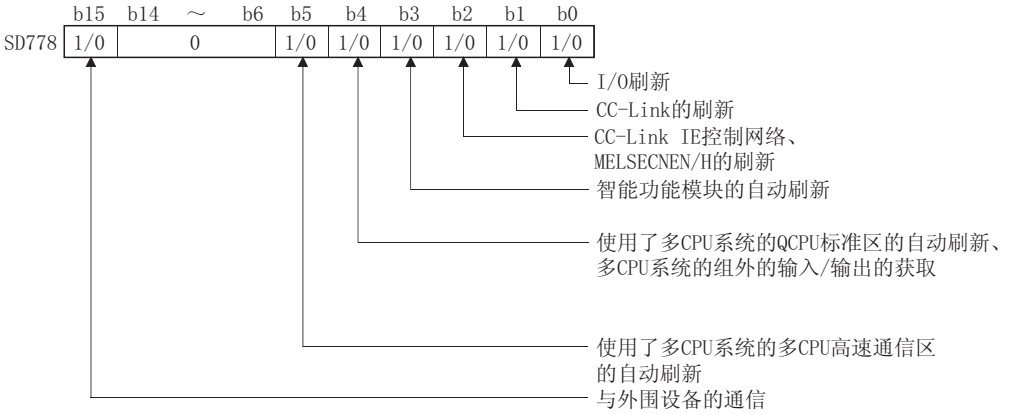
(3) 选择刷新项目时

(a) 通过 SD778 选择刷新项目后，将 SM775 置为 ON。
SM775 的 ON/OFF、SD778 中可指定的刷新项目如下表所示。

刷新项目	SM775 为 OFF 时	SM775 为 ON 时
I/O 刷新	非执行	可以选择执行 / 非执行
CC-Link 的刷新	执行	
CC-Link IE 控制网络的刷新		
MELSECNEN/H 的刷新		
智能功能模块的自动刷新		
使用了多 CPU 系统的 QCPU 标准区的自动刷新		
多 CPU 系统的组外的输入 / 输出的获取		
使用了多 CPU 系统的多 CPU 高速通信区的自动刷新		
与外围设备的通信		

(b) 通过 SD778 的 b0 ～ b5、b15 选择刷新项目。
SD778 的各位的执行 / 非执行的指定如下表所示。

SD778 的位	执行	非执行
b0 ～ b5	1	0
b15	0	1

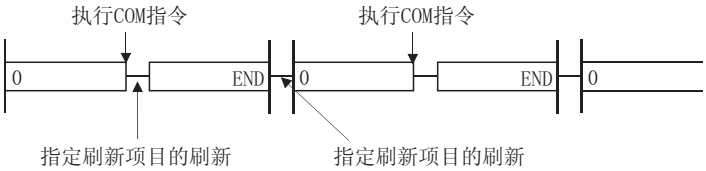


例 希望仅加快与远程 I/O 站的发送接收处理的情况下，则仅指定 MELSECNEN/H 的刷新。
（仅在 SD778 的 b2 及 b15 中写入 1(SD778：8004H)）

要 点

通过 COM 指令进行的多 CPU 间刷新是在下述情况下执行。
- 来自于其它站的接收动作： SD778 的 b4(CPU 共享存储器的自动刷新）为 1 时
- 来自于自站的发送动作： SD778 的 b15（与外围设备的通信的执行 / 非执行）为 0 时

(4) 在执行 COM 指令时 CPU 模块暂时中断顺控程序的处理，进行指定刷新项目的刷新。



(5) COM 指令的使用次数在顺控程序中无限制。

但是，通过 SD778 选择的与外围设备的通信、刷新项目的刷新需要耗费时间，顺控程序的扫描时间将因此而相应延长，应加以注意。

(6) 在通用型 QCPU 中执行 COM 指令的过程中，将变为中断允许状态。但是，通过中断程序等使用刷新数据时，有可能发生数据背离，应加以注意。

(7) COM 指令的刷新项目如下表所示。

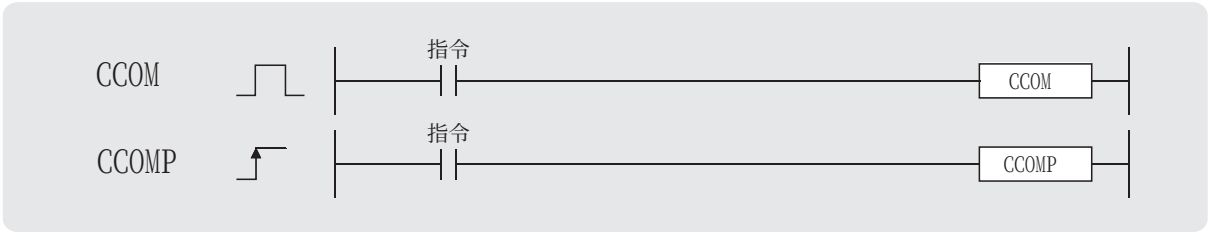
CPU 模块型号	功能版本	序列号	SM775	刷新项目
Q00JCPU、Q00CPU、Q01CPU	A	0421 以前	OFF	刷新所有的刷新项目
			ON	仅与外围设备通信
	B	0422 以后	OFF	刷新所有的刷新项目
			ON	刷新通过 SD778 选择的刷新项目
Q02CPU、Q02HCPU、Q06HCPU、Q12HCPU、Q25HCPU	A	----	ON/OFF	刷新所有的刷新项目
	B	04011 以前	OFF	刷新所有的刷新项目
			ON	仅与外围设备通信
		04012 以后	OFF	刷新所有的刷新项目
			ON	刷新通过 SD778 选择的刷新项目
Q02PHCPU、Q06PHCPU	----	----	OFF	刷新所有的刷新项目
			ON	刷新通过 SD778 选择的刷新项目
Q12PHCPU、Q25PHCPU	C	07031 以前	OFF	刷新所有的刷新项目
			ON	仅与外围设备通信
		07032 以后	OFF	刷新所有的刷新项目
			ON	刷新通过 SD778 选择的刷新项目
Q12PRHCPU、Q25PRHCPU	D	----	OFF	刷新所有的刷新项目
			ON	刷新通过 SD778 选择的刷新项目
Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU、Q03UDCPU、Q04UDHCPU、Q06UDHCPU、Q10UDHCPU、Q13UDHCPU、Q20UDHCPU、Q26UDHCPU、Q03UDECPU、Q04UDEHCPU、Q06UDEHCPU、Q10UDEHCPU、Q13UDEHCPU、Q20UDEHCPU、Q26UDEHCPU	B	----	OFF	刷新所有的刷新项目
			ON	刷新通过 SD778 选择的刷新项目

要 点

1. COM 指令不能用于低速执行型程序、恒定周期执行型程序以及中断程序。
2. 在冗余 CPU 中，使用 COM 指令时是有所限制的。详细内容请参阅下述手册。
   • QnPRHCPU 用户手册（冗余系统篇）

9.14 选择刷新指令 (CCOM(P))

• QnU(D)(H)CPU：序列号的前 5 位数为 “10102”以后
• QnUDE(H)CPU：序列号的前 5 位数为 “10102”以后



设置数据	内部软元件		Z, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
--	--								

功 能

(1) 如果执行 CCOM(P) 指令，可以执行下述刷新。

- I/O 刷新
- CC-Link 的刷新
- CC-Link IE 控制网络的刷新
- MELSECNEN/H 的刷新
- 智能功能模块的自动刷新
- 使用了多 CPU 系统的 QCPU 标准区的自动刷新
- 多 CPU 系统的组外的输入 / 输出的获取
- 使用了多 CPU 系统的多 CPU 高速通信区的自动刷新
- 与外围设备的通信

(2) 如果将 SM775 置为 OFF，将对除 I/O 刷新以外的所有刷新项目进行刷新。

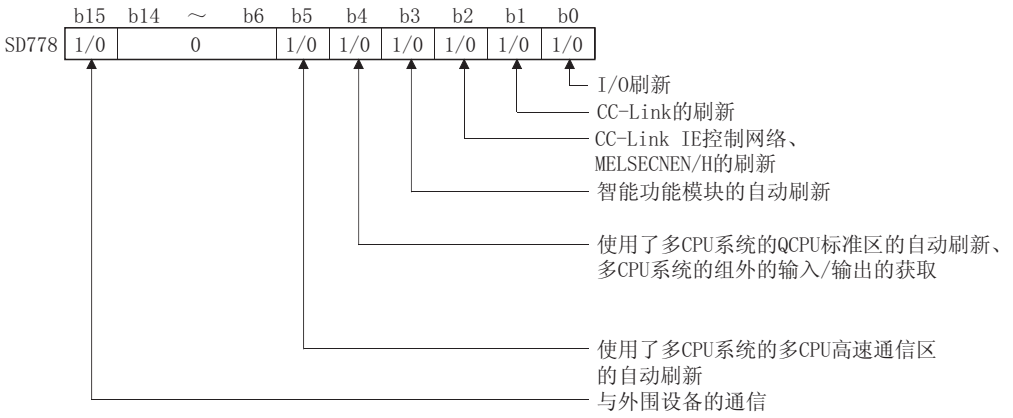
(3) 选择刷新项目时

(a) 通过 SD778 选择刷新项目后，将 SM775 置为 ON。
SM775 的 ON/OFF、SD778 中可指定的刷新项目如下表所示。

刷新项目	SM775 为 OFF 时	SM775 为 ON 时
I/O 刷新	非执行	可以选择执行 / 非执行
CC-Link 的刷新	执行	
CC-Link IE 控制网络的刷新		
MELSECNEN/H 的刷新		
智能功能模块的自动刷新		
使用了多 CPU 系统的 QCPU 标准区的自动刷新		
多 CPU 系统的组外的输入 / 输出的获取		
使用了多 CPU 系统的多 CPU 高速通信区的自动刷新		
与外围设备的通信		

(b) 通过 SD778 的 b0 ～ b5、b15 选择刷新项目。
SD778 的各位的执行 / 非执行的指定如下表所示。

SD778 的位	执行	非执行
b0 ～ b5	1	0
b15	0	1

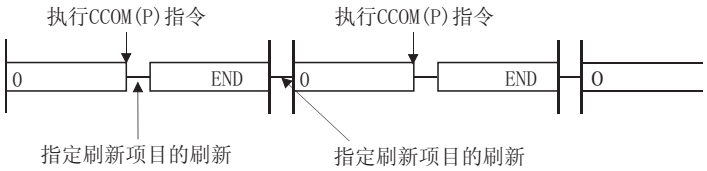


要 点

通过 CCOM(P) 指令进行的多 CPU 间刷新是在下述情况下执行。

- 来自于其它站的接收动作： SD778 的 b4(CPU 共享存储器的自动刷新) 为 1 时
- 来自于自站的发送动作： SD778 的 b15(与外围设备的通信的执行 / 非执行) 为 0 时

(4) 在执行 CCOM(P) 指令时 CPU 模块暂时中断顺控程序的处理，进行指定刷新项目的刷新。



(5) CCOM(P) 指令的使用次数在顺控程序中无限制。

但是，通过 SD778 选择的刷新项目的刷新需要耗费时间，顺控程序的扫描时间将因此而相应延长，应加以注意。

(6) 在通用型 QCPU 中执行 CCOM(P) 指令的过程中，将变为中断允许状态。但是，通过中断程序等使用刷新数据时，有可能发生数据背离，应加以注意。

(7) CCOM(P) 指令不能用于恒定周期执行型程序以及中断程序。

出　错

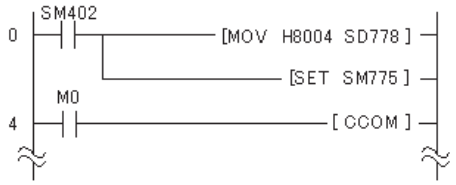
(1) 在序列号的前 5 位数为 “10101” 以前的 QnUD(H)CPU 中执行了 CCOM(P) 指令时。

（出错代码：4100）

程序示例

(1) 以下为根据 M0 的 ON/OFF 状态，可以对选择刷新的执行 / 非执行进行切换的程序。

[梯形图模式]



[列表模式]

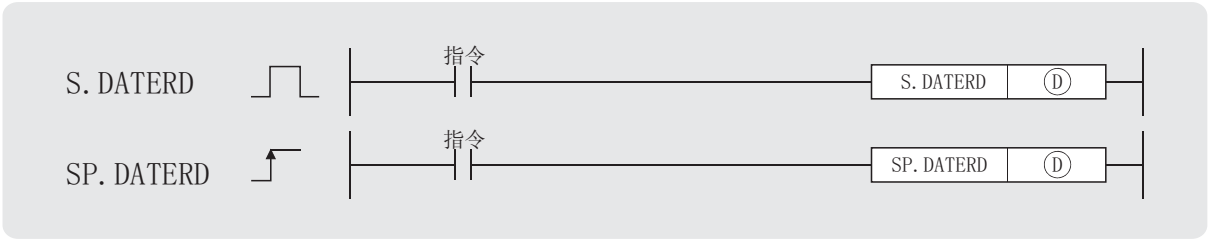
步	指令	软元件	
0	LD	SM402	
1	MOV	H8004	SD778
3	SET	SM775	
4	LD	M0	
5	CCOM		

9.15 扩展时钟指令

9.15.1 扩展时钟数据的读取 (S(P).DATERD)

Basic (×) | Ver. High performance | Ver. Process | Ver. Redundant | Universal

高性能型 QCPU：序列号的前 5 位数为“07032”以后
过程 CPU：序列号的前 5 位数为“07032”以后
冗余 CPU：序列号的前 5 位数为“07032”以后



Ⓓ：存储读取的时钟数据的软元件的起始编号 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
Ⓓ	--	○		--					

功 能

(1) 从 CPU 模块的时钟因子中读取“年、月、日、时、分、秒、星期、1/1000 秒”后，以 BIN 值存储到Ⓓ中指定的软元件的后面。

时钟因子 ⇨

Ⓓ	年(公历)	(1980～2079)
Ⓓ+1	月	(1～12)
Ⓓ+2	日	(1～31)
Ⓓ+3	时(24小时时钟)	(0～23)
Ⓓ+4	分	(0～59)
Ⓓ+5	秒	(0～59)
Ⓓ+6	星期	(0～6)
Ⓓ+7	1/1000秒	(0～999)

(2) Ⓓ的“年”以公历 4 位数存储。

(3) Ⓓ+6 的“星期”中以“0 ～ 6”存储“星期日～星期六”。

星期	星期日	星期一	星期二	星期三	星期四	星期五	星期六
存储数据	0	1	2	3	4	5	6

(4) 闰年时将进行自动修正。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

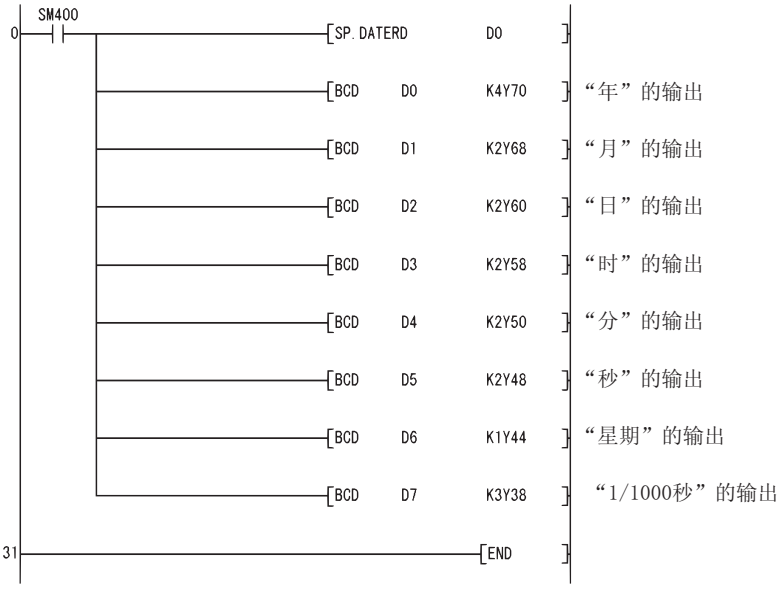
•Ⓓ中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)　　(出错代码：4101)

程序示例

(1) 以下为将时钟数据以 BCD 值进行输出的程序。

年..... Y70 ～ Y7F
月..... Y68 ～ Y6F
日..... Y60 ～ Y67
时..... Y58 ～ Y5F
分..... Y50 ～ Y57
秒..... Y48 ～ Y4F
星期..... Y44 ～ Y47
1/1000 秒 ..... Y38 ～ Y43

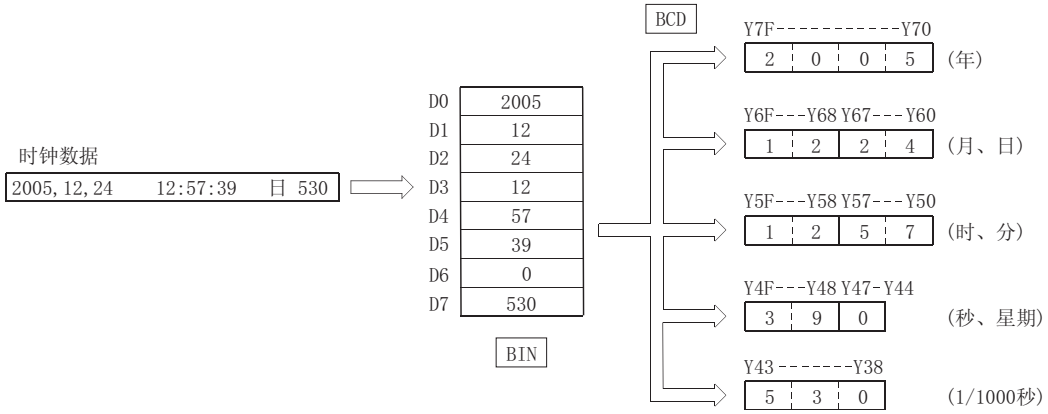
[梯形图模式]



[列表模式]

步	指令	软元件	
0	LD	SM400	
1	SP.DATERD		D0
7	BCD	D0	K4Y70
10	BCD	D1	K2Y68
13	BCD	D2	K2Y60
16	BCD	D3	K2Y58
19	BCD	D4	K2Y50
22	BCD	D5	K2Y48
25	BCD	D6	K1Y44
28	BCD	D7	K3Y38
31	END		

[动作]



注意事项

(1) 对于本指令，即使在 CPU 模块中设置了错误的时钟数据情况下，也将时钟数据读取后存储到软元件中。（例： 2 月 30 日）

通过 DATEWR 指令及 GX Developer 设置时钟数据时，应设置正确的时钟数据。

(2) 读取 1/1000 秒的时钟数据时的误差最大为 2ms。(CPU 模块内部的时钟因子中记忆的数据与通过本指令读取的数据之间的误差。)

(3) 位软元件的位数指定仅在满足下述 (a)、(b) 的条件时才可以使用。

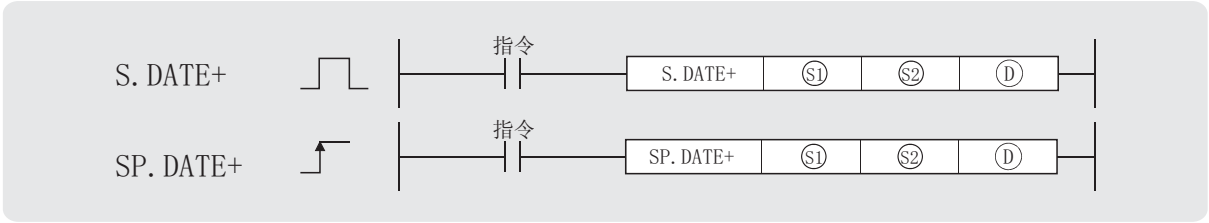
(a) 位数的指定： K4

(b) 软元件的起始： 16 的倍数

如果未满足上述 (a)、(b) 的条件，将变为 INSTRCT CODE ERR.（出错代码： 4004）的出错状态。

9.15.2 扩展时钟数据的加法运算 (S(P).DATE+)

高性能型 QCPU：序列号的前 5 位数为 “07032” 以后
过程 CPU：序列号的前 5 位数为 “07032” 以后
冗余 CPU：序列号的前 5 位数为 “07032” 以后



(S1) ：存储被进行加法运算的时间数据的软元件的起始编号 (BIN16 位)。
(S2) ：存储进行加法运算的时间（时刻）数据的软元件的起始编号 (BIN16 位)。
(D) ：存储加法运算结果（时间）数据的软元件的起始编号 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
(S1)	--	○		--					
(S2)	--	○		--					
(D)	--	○		--					

功 能

(1) 将(S1)中指定的时间数据与(S2)中指定的时间数据进行加法运算后，将加法运算结果存储到(D)中指定的软元件编号的后面。

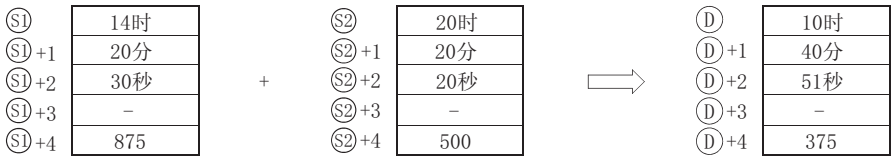
		设置数据			设置数据			设置数据
(S1)	时	(0～23)	(S2)	时	(0～23)	(D)	时	(0～23)
(S1)+1	分	(0～59)	(S2)+1	分	(0～59)	(D)+1	分	(0～59)
(S1)+2	秒	(0～59) +	(S2)+2	秒	(0～59) ⇒	(D)+2	秒	(0～59)
(S1)+3	-		(S2)+3	-		(D)+3	-	
(S1)+4	1/1000秒	(0～999)	(S2)+4	1/1000秒	(0～999)	(D)+4	1/1000秒	(0～999)

例如，将 6 时 32 分 40 秒 875 与 7 时 48 分 10 秒 500 进行加法运算时的情况如下所示。

(S1)	6时		(S2)	7时		(D)	14时
(S1)+1	32分		(S2)+1	48分		(D)+1	20分
(S1)+2	40秒	+	(S2)+2	10秒	⇒	(D)+2	51秒
(S1)+3	-		(S2)+3	-		(D)+3	-
(S1)+4	875		(S2)+4	500		(D)+4	375

(2) 运算结果的时间超过了 24 小时时，将该值减去 24 小时后的值作为运算结果。

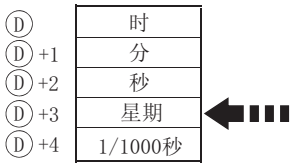
例如，将 14 时 20 分 30 秒 875 与 20 时 20 分 20 秒 500 进行加法运算时，其结果不是 34 时 40 分 51 秒 375，而是 10 时 40 分 51 秒 375。



要 点

(S1)+3、(S2)+3、(D)+3 的软元件不能用于运算。

对于通过 S(P).DATERD 指令读取的时钟数据可以直接进行加法运算。



通过S(P).DATERD指令读取时，在秒与1/1000秒之间加入星期。
由于在S(P).DATE+指令中不对星期进行运算，因此可以直接进行加法运算。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- (S1)、(S2) 的设置数据超出了允许范围时。(参阅功能 (1)) (出错代码： 4100)
- (S1)、(S2)、(D) 中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU) (出错代码： 4101)

注意事项

(1) 位软元件的位数指定仅在满足下述 (a)、(b) 的条件时才可以使用。

(a) 位数的指定 ： K4

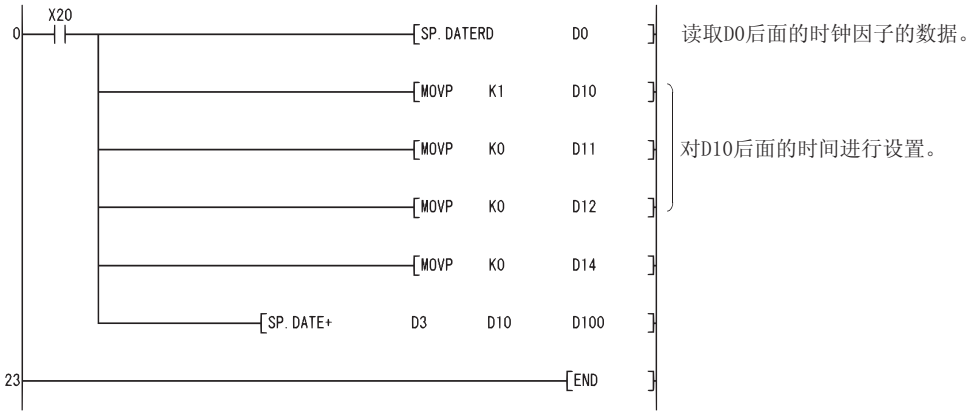
(b) 软元件的起始： 16 的倍数

如果未满足上述 (a)、(b) 的条件，将变为 INSTRCT CODE ERR.(出错代码： 4004) 的出错状态。

程序示例

(1) 以下为 X20 变为 ON 时，对从时钟因子中读取的时间数据进行 1 小时的加法运算后，将运算结果存储到 D100 后面的程序。

[梯形图模式]

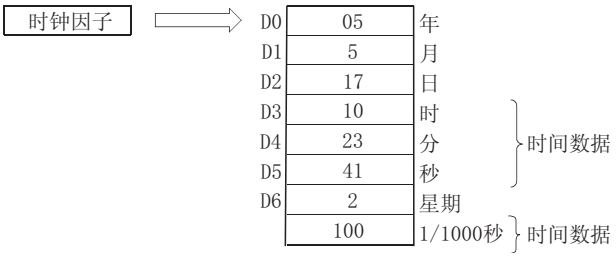


[列表模式]

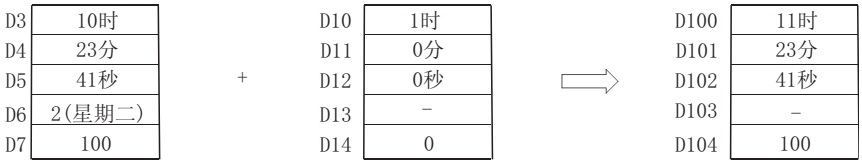
步	指令	软元件		
0	LD	X20		
1	SP.DATERD		D0	
7	MOVP	K1	D10	
9	MOVP	K0	D11	
11	MOVP	K0	D12	
13	MOVP	K0	D14	
15	SP.DATE+	D3	D10	D100
23	END			

[动作]

• 通过 SP.DATERD 指令读取时间数据

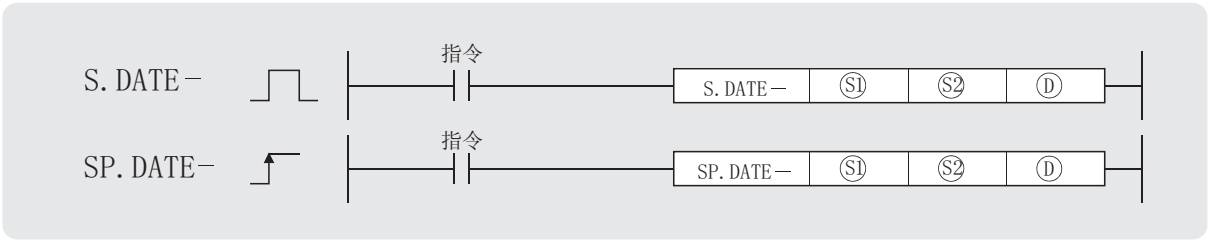


• 通过 SP.DATE+ 指令进行加法运算



9.15.3 扩展时钟数据的减法运算 (S(P).DATE-)

高性能型 QCPU：序列号的前 5 位数为“07032”以后
过程 CPU：序列号的前 5 位数为“07032”以后
冗余 CPU：序列号的前 5 位数为“07032”以后



(S1) ：存储被进行减法运算的时间数据的软元件的起始编号 (BIN16 位)。
(S2) ：存储进行减法运算的时间数据的软元件的起始编号 (BIN16 位)。
(D) ：存储减法运算结果时间数据的软元件的起始编号 (BIN16 位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
(S1)	--	○		--					
(S2)	--	○		--					
(D)	--	○		--					

功 能

(1) 将(S1)中指定的时间数据与(S2)中指定的时间数据进行减法运算，并将减法运算结果存储到(D)中指定的软元件编号的后面。

		设置数据			设置数据			设置数据
(S1)	时	(0～23)	(S2)	时	(0～23)	(D)	时	(0～23)
(S1)+1	分	(0～59)	(S2)+1	分	(0～59)	(D)+1	分	(0～59)
(S1)+2	秒	(0～59) -	(S2)+2	秒	(0～59) ⇨	(D)+2	秒	(0～59)
(S1)+3	-		(S2)+3	-		(D)+3	-	
(S1)+4	1/1000秒	(0～999)	(S2)+4	1/1000秒	(0～999)	(D)+4	1/1000秒	(0～999)

例如，将 10 时 40 分 20 秒 875 与 3 时 50 分 10 秒 500 进行减法运算时的情况如下所示。

(S1)	10时		(S2)	3时		(D)	6时
(S1)+1	40分		(S2)+1	50分		(D)+1	50分
(S1)+2	20秒	-	(S2)+2	10秒	⇨	(D)+2	10秒
(S1)+3	-		(S2)+3	-		(D)+3	-
(S1)+4	875		(S2)+4	500		(D)+4	375

(2) 运算结果的时间为负数时，将该值加上 24 小时后的值作为运算结果。

例如，将 4 时 50 分 32 秒 875 与 10 时 42 分 12 秒 500 进行减法运算时，其结果不是 -6 时 8 分 20 秒 375，而是 18 时 8 分 20 秒 375。

ⓢ1	4时		ⓢ2	10时		ⓓ	18时
ⓢ1+1	50分		ⓢ2+1	42分		ⓓ+1	8分
ⓢ1+2	32秒	-	ⓢ2+2	12秒	⇨	ⓓ+2	20秒
ⓢ1+3	-		ⓢ2+3	-		ⓓ+3	-
ⓢ1+4	875		ⓢ2+4	500		ⓓ+4	375

要 点

ⓢ1+3、ⓢ2+3、ⓓ+3 的软元件不能用于运算。

对于通过 S(P).DATERD 指令读取的时钟数据可以直接进行减法运算。

ⓓ	时
ⓓ+1	分
ⓓ+2	秒
ⓓ+3	星期
ⓓ+4	1/1000秒

通过S(P).DATERD指令读取时，在秒与1/1000秒之间加入星期。
由于在S(P).DATE-指令中不对星期进行运算，因此可以直接进行减法运算。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- ⓢ1、ⓢ2 的设置数据超出了允许范围时。(参阅功能 (1)) (出错代码： 4100)
- ⓢ1、ⓢ2、ⓓ 中指定的软元件超出了相应软元件的范围时。(仅通用型 QCPU)

(出错代码： 4101)

注意事项

(1) 位软元件的位数指定仅在满足下述 (a)、(b) 的条件时才可以使用。

(a) 位数的指定 ： K4

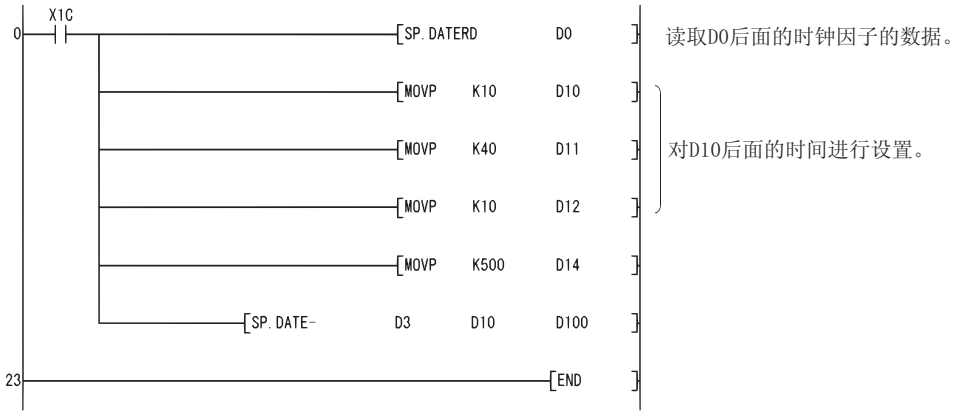
(b) 软元件的起始： 16 的倍数

如果未满足上述 (a)、(b) 的条件，将变为 INSTRCT CODE ERR.(出错代码： 4004) 的出错状态。

程序示例

(1) 以下为 X1C 变为 ON 时，将从时钟因子中读取的时间数据与 D10 后面存储的时间数据进行减法运算后，将运算结果存储到 D100 后面的程序。

[梯形图模式]

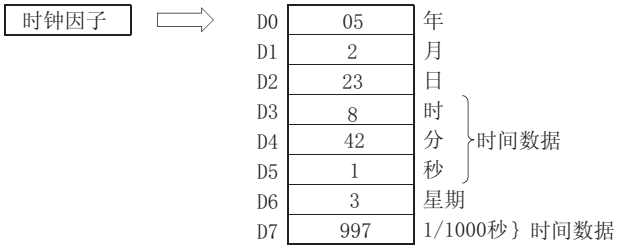


[列表模式]

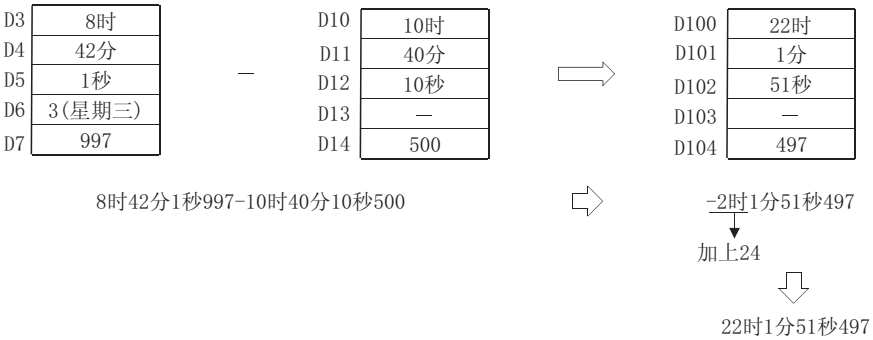
步	指令	软元件		
0	LD	X1C		
1	SP.DATERD		D0	
7	MOVP	K10	D10	
9	MOVP	K40	D11	
11	MOVP	K10	D12	
13	MOVP	K500	D14	
15	SP.DATE-	D3	D10	D100
23	END			

[动作]

• 通过 SP.DATERD 指令读取时间数据



• 通过 SP.DATE- 指令进行减法运算



备忘录

10 冗余系统指令（用于冗余 CPU）

分类	处理内容	参阅章节
系统切换指令	在执行了 SP.CONTSW 指令的扫描的 END 处理时，进行控制系统与待机系统的切换。	10.1 节

10.1 系统切换指令 (SP.CONTSW)

Basic High performance Process Redundant Universal

Ⓢ ：用于指定发出系统切换请求处理的 0 以外的值 (BIN16 位)。

Ⓓ ：异常结束软元件编号 (位)。

设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字		位	字				
Ⓢ	--	○		--				○	--
Ⓓ	○	○ *1		--				--	--

*1: 可以使用字软元件的位指定。

功 能

(1) 在执行了 SP.CONTSW 指令的扫描的 END 处理时，进行控制系统与待机系统的切换。

(2) 通过 SP.CONTSW 指令进行系统切换时，需要预先将 “手动切换允许标志 (SM1592)” 置为 ON (允许)。

(3) 使用了多个 SP.CONTSW 指令时，通过Ⓢ对发生了系统切换的程序的处理块进行掌控。

Ⓢ中的值的允许指定范围为 -32768 ～ -1，1 ～ 32767(1H ～ FFFFH)。

通过 SP.CONTSW 指令指定的Ⓢ的值在系统切换正常结束时将被存储到出错公共信息的 “系统切换指令变量 (SD6)” 中。*2

在同一个扫描中执行了多个 SP.CONTSW 指令时，最先执行的 SP.CONTSW 指令的变量将被存储到系统切换指令变量 (SD6) 中。

(4) SP.CONTSW 指令中指定的Ⓢ值在系统切换正常结束时，将被存储到新控制系统 CPU 模块的 “系统切换指令变量 (SD1602)” 中。*3

通过在新控制系统 CPU 模块中读取 SD1602，可以确认通过哪个 SP.CONTSW 指令进行了系统切换。

*2: 通过 SP.CONTSW 指令指定的Ⓢ值可以在 GX Developer 的可编程控制器诊断的出错公共信息中确认。

*3: 新控制系统 CPU 模块是指，通过 SP.CONTSW 指令进行系统切换后，由待机系统被切换为控制系统的 CPU 模块。

(5) 在通过 SP.CONTSW 指令未能完成系统切换时，在控制系统 CPU 模块中异常结束软元件将变为 ON。

(a) 执行 SP.CONTSW 指令的情况下，如果由于下述原因检测出 OPERATION ERROR，执行 SP.CONTSW 指令时异常结束软元件将变为 ON。

- 执行的 SP.CONTSW 指令的Ⓢ中指定了 0 时。
- “手动切换允许标志 (SM1592)”处于 OFF 状态时。
- 在分开模式下的待机系统中执行了 SP.CONTSW 指令时。
- 在调式模式下执行了 SP.CONTSW 指令时。

(b) 在由于下表中的原因导致系统未能完成切换的情况下，在 END 处理的系统切换执行时异常结束软元件将变为 ON。

系统切换失败原因号	系统切换失败原因
0	正常结束。
1	热备电缆脱落或者热备电缆异常。
2	发生了待机系统硬件异常、电源 OFF、处于复位状态、看门狗定时器出错。
3	控制系统发生了看门狗定时器出错。
4	热备传送准备中。
5	通信超时。
6	待机系统中发生停止出错。（看门狗定时器出错除外）
7	控制系统及待机系统的动作状态异常。
8	正在从控制系统向待机系统进行存储器复制。
9	正在进行 RUN 中写入。
10	待机系统中检测出网络异常。

系统切换未能完成，异常结束软元件变为 ON 时，在 “系统切换原因 (SD1588)”中将存储 16，在 “系统切换失败原因 (SD1589)”中将存储上表中的系统切换失败原因号。

(6) 对于处于 ON 状态的异常结束位，应通过用户程序或者 GX Developer 将其变为 OFF。
在异常结束软元件处于 ON 的状态下，如果执行 SP.CONTSW 指令后正常地进行了系统切换，在新待机系统 CPU 模块的异常结束软元件也将变为 OFF。
但是，在由于 SP.CONTSW 指令以外的原因导致系统切换未能完成时，异常结束软元件将不变为 OFF。

出 错

(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- 执行 SP.CONTSW 指令时，在Ⓢ中指定了 0。 （出错代码： 4100）
- 执行 SP.CONTSW 指令时，手动切换允许标志 (SM1592) 处于 OFF（禁止）状态。 （出错代码： 4120）
- 在分开模式下的待机系统 CPU 模块中执行了 SP.CONTSW 指令时。 （出错代码： 4121）
- 在调试模式下执行了 SP.CONTSW 指令时。 （出错代码： 4121）

(2) 在系统切换未能完成的情况下，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

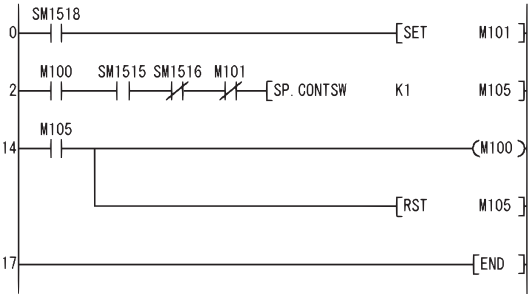
- 热备电缆脱落或者热备电缆异常时。 (出错代码：6220)
- 待机系统中发生了硬件异常、电源 OFF、处于复位状态、看门狗定时器出错时。 (出错代码：6220)
- 控制系统中发生了看门狗定时器出错时。 (出错代码：6220)
- 处于热备传送准备中状态时。 (出错代码：6220)
- 发生了通信超时时。 (出错代码：6220)
- 待机系统中发生了除看门狗定时器出错以外的停止出错时。 (出错代码：6220)
- 控制系统及待机系统的动作状态异常时。 (出错代码：6220)
- 正在从控制系统向待机系统进行存储器复制时。 (出错代码：6220)
- 正在进行 RUN 中写入时。 (出错代码：6220)
- 待机系统中检测出网络异常时。 (出错代码：6220)

程序示例

(1) 以下为在系统切换指令 (M100) 的上升沿进行系统切换的程序。

如果系统切换指令 (M100) 保持为 ON 状态不变，系统切换后新控制系统 CPU 模块中也将执行 SP.CONTSW 指令，因此将 M101 附加到执行条件中作为防止连续切换标志使用。

[梯形图模式]



[列表模式]

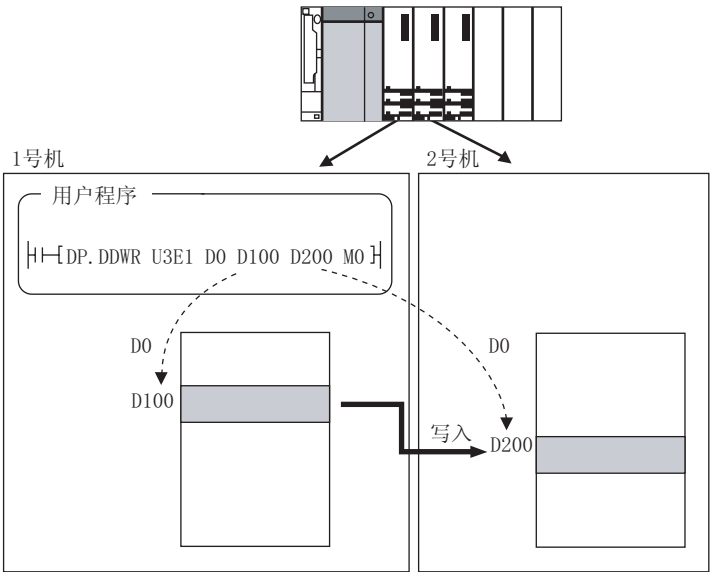
步	指令	软元件		
0	LD	SM1518		
1	SET	M101		
2	LD	M100		
3	AND	SM1515		
4	ANI	SM1516		
5	ANI	M101		
6	SP.CONTSW		K1	M105
14	LD	M105		
15	OUT	M100		
16	RST	M105		
17	END			

11 多 CPU 高速通信专用指令

分类	处理内容	参阅章节
至其它站的写入指令	对其它站进行软元件写入	11.2 节
对其它站进行的读取指令	对其它站进行软元件读取	11.3 节

11.1 概要

多 CPU 高速通信专用指令是从通用型 QCPU 中对其它站的通用型 QCPU 进行软元件数据的写入 / 读取的指令。
根据多 CPU 高速通信专用指令，从 1 号机对 2 号机进行写入时的动作如下图所示。



要 点

使用多 CPU 高速通信专用指令时，自站、其它站（指令的执行对象站）均只能使用下述 CPU 模块。

- 序列号的前 5 位数为 “10012” 以后的 Q03UDCPU、Q04UDHCPU、Q06UDHCPU
- Q10UDHCPU、Q13UDHCPU、Q20UDHCPU、Q26UDHCPU
- QnUDE(H)CPU

(1) 用于执行多 CPU 高速通信专用指令的系统、参数设置

多 CPU 高速通信专用指令功能在如下所示的系统配置、参数设置时可以执行。

- 1 号机中使用的是 QnUD(H)CPU 或者 QnUDE(H)CPU。
- 使用了多 CPU 高速通信主基板 (Q3 □ DB)。
- 可编程控制器参数的多 CPU 设置中设置了 “使用多 CPU 高速通信功能”。

(2) 可进行写入 / 读取的软元件

(a) 可进行写入 / 读取的软元件名

通过多 CPU 高速通信专用指令可对其它站的通用型 QCPU 进行写入 / 读取的软元件如下表所示。

分类	类型	软元件名	对象软元件设置 允许 / 禁止	备注
内部用户软元件	位软元件	X、Y、M、L、B、F、SB	△	设置时的必要条件 • 进行了 16 位 (4 位数) 的位数指定。 • 开始位软元件为 16(10H) 的倍数。
	字软元件	T、ST、C、D、W、SW	○	-
内部系统软元件	位软元件	SM	△	设置时的必要条件 • 进行了 16 位 (4 位数) 的位数指定。 • 开始位软元件为 16(10H) 的倍数。
	字软元件	SD	○	-
文件寄存器	字软元件	R、ZR	○	-

○ : 可以设置 △ : 带条件可设置

要 点

SB、SW、SM、SD 中包含有系统信息区。
通过多 CPU 高速通信专用指令的 D(P).DDWR 指令，对上述软元件进行写入时，应注意防止破坏系统信息。

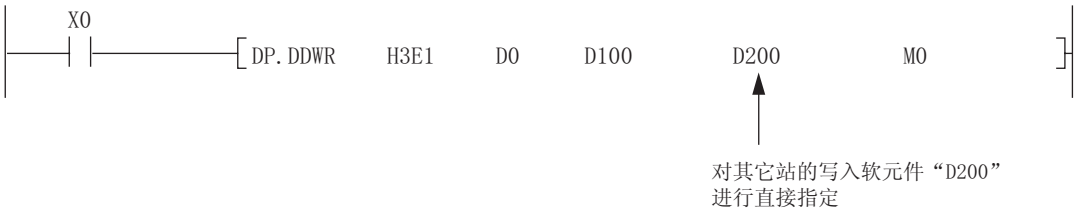
(3) 软元件的指定方法及可进行写入 / 读取的范围

在其它站 CPU 软元件的指定方法中，有软元件指定及字符串指定这 2 种类型。
在软元件指定及字符串指定中，可对其它站进行写入 / 读取的软元件范围是不相同的。

(a) 软元件指定

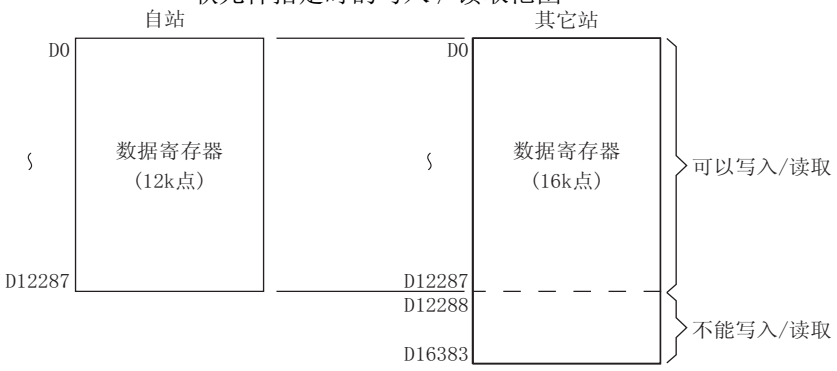
软元件指定是对执行写入 / 读取的其它站软元件进行直接指定的方法。

DP.DDWR 指令的软元件指定程序



在软元件指定中，可在自站的软元件范围内进行写入 / 读取。
例如，自站的数据寄存器为 12k 点，其它站的数据寄存器为 16k 点时，可对其它站的数据寄存器的起始开始的 12k 点的数据进行写入 / 读取。

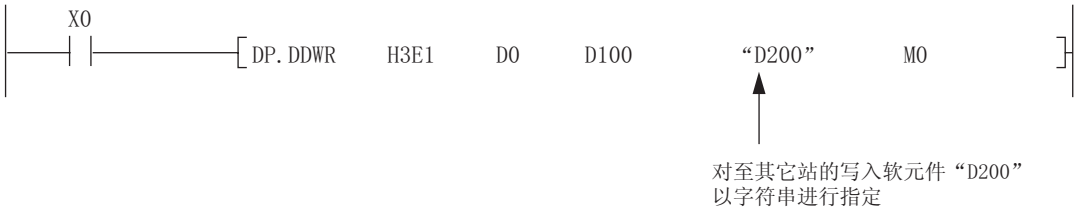
软元件指定时的写入 / 读取范围



(b) 字符串指定

字符串指定是对进行写入 / 读取的其它站的软元件以字符串进行指定的方法。

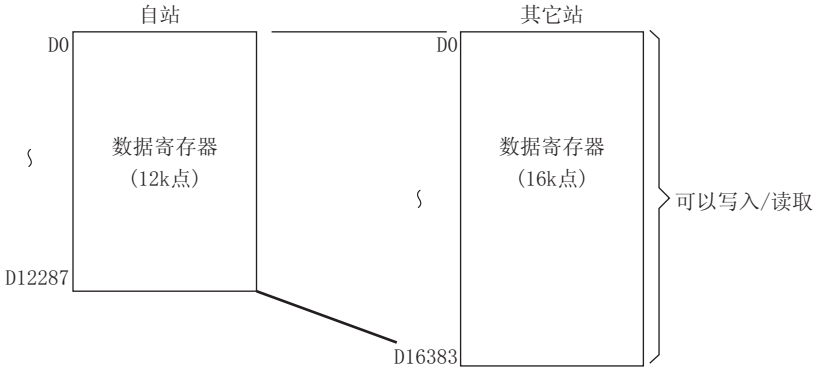
DP.DDWR 指令的字符串指定程序



在字符串指定中，可对其它站软元件的所有范围进行写入 / 读取。

例如，自站的数据寄存器为 12k 点，其它站的数据寄存器为 16k 点时，可对其它站的数据寄存器的起始开始的 16k 点的数据进行写入 / 读取。

字符串指定时的写入 / 读取范围



备 注

字符串指定的注意事项如下所示。

- 字符串指定中最多可指定 32 个字符。
- 无论在软元件号的高位中是否附加了 “0”，均作为相同的软元件进行处理。

  例如，“D1”与 “D0001”均作为 D1 处理。
- 以大写字母指定的软元件与以小写字母字母指定的软元件均作为相同的软元件进行处理。

  例如，“D1”与 “d1”均作为 D1 进行处理。
- 通过字符串对其它站 CPU 中不存在的软元件进行了指定时，指令将变为异常结束。

(4) 多 CPU 高速通信区的管理

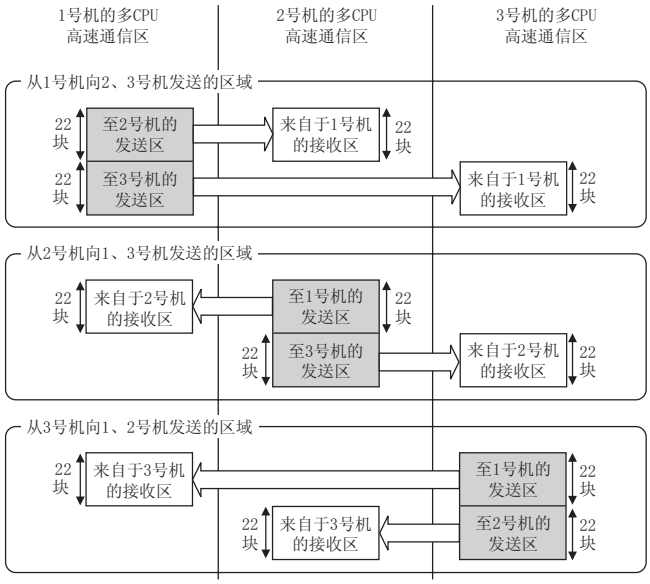
(a) 多 CPU 高速通信区是以 16 字作为最小单位的块进行管理的。

各站中可使用的块数、指令中使用的块数如下表所示。

CPU 个数	系统区 *1	
	1k 点	2k 点
2	46	110
3	22	54
4	14	35

*1: 关于系统区的设置，请参阅 QCPU 用户手册（多 CPU 系统篇）。

(b) 在由 3 个 CPU 模块构成的多 CPU 系统中，系统区大小为 1k 字时的多 CPU 高速通信区的构成如下所示。



(5) 指令中使用的块数

指令中使用的块数根据进行写入的点数而不同。

指令中使用的块数如下表所示。

指令中指定的写入 / 读取点数	D(P).DDWR 指令	D(P).DDRD 指令
1 ～ 4	1	1
5 ～ 20	2	
21 ～ 36	3	
37 ～ 52	4	
53 ～ 68	5	
69 ～ 84	6	
85 ～ 100	7	

(6) 可同时执行的多 CPU 高速通信专用指令

在通用型 CPU 中，可同时执行以下范围内数量的多 CPU 高速通信专用指令。

[各站中可使用的块数] ≧ [同时执行指令使用的块数的合计]

由于执行多 CPU 高速通信专用指令，多 CPU 高速通信专用指令使用的块数超过了多 CPU 高速通信区的总块数时，在该扫描中不执行本指令（变为无处理），在下一个扫描中再次执行本指令。但是，执行了本指令时，多 CPU 高速通信区的空闲块数少于 SD796 ～ SD799（多 CPU 高速通信专用指令最多块数设置）的设置值的情况下，本指令将变为异常结束。

多 CPU 高速通信区的空闲块数少于多 CPU 高速通信专用指令使用的块数时或者少于 SD796 ～ SD799 的设置值时，多 CPU 高速通信专用指令的执行可否情况如下所示。

指令使用块数*1与空闲的大小关系 / SD设置值与空闲块数的大小关系	指令使用块数*1 ≤空闲块数*2	指令使用块数*1＞空闲块数*2
SD设置值*3≤空闲块数*2	执行	不执行(执行无处理)
SD设置值*3＞空闲块数*2	异常结束	

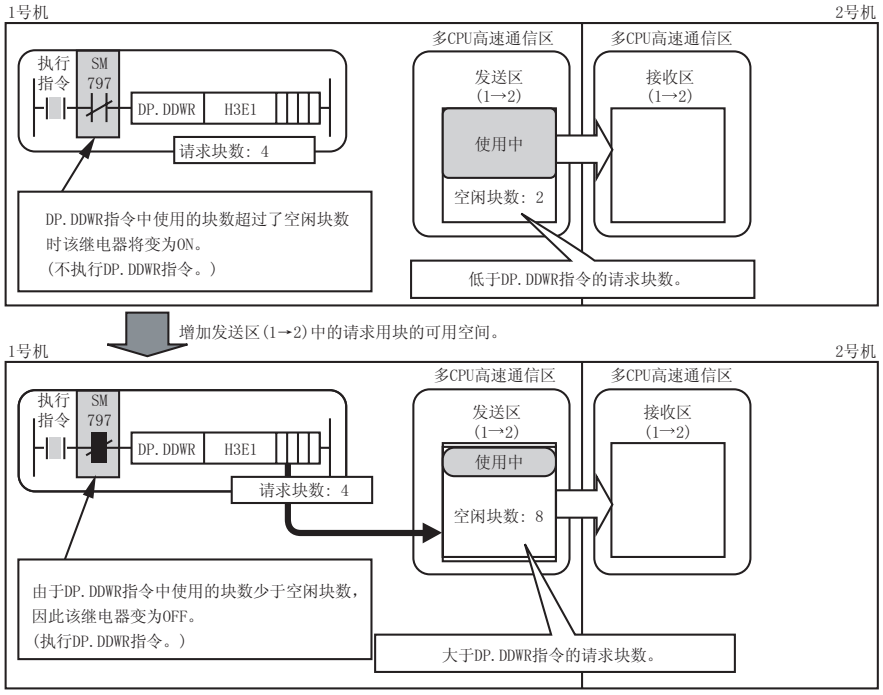
*1: 多CPU高速通信专用指令使用的块数。
*2: 多CPU高速通信区的空闲块数。
*3: SD796～SD799的设置值。

(7) 使用多 CPU 高速通信专用指令时的互锁

(a) 作为多 CPU 高速通信专用指令的互锁，配备了特殊继电器 SM796 ～ SM799（多 CPU 高速通信专用指令使用块信息）。
同时执行多个多 CPU 高速通信专用指令时，应将 SM796 ～ SM799 作为 CPU 高速通信专用指令的互锁使用。

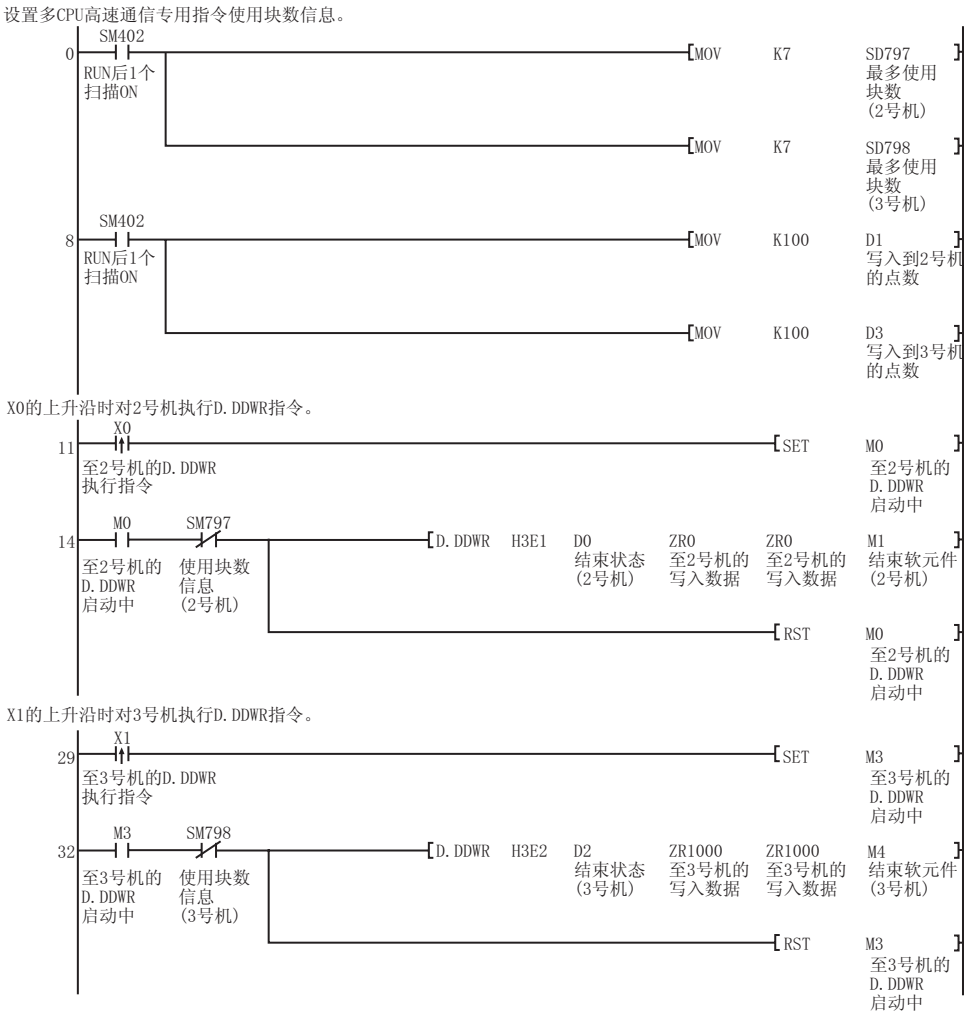
要 点

使用特殊继电器 SM796 ～ SM799 时，应将各站中可使用指令的最多块数设置到特殊寄存器 SD796 ～ SD799 中。（例如，对 3 号机执行多 CPU 高速通信专用指令的块数的最大值为 5 时，在 SD798 中设置 5。）
多 CPU 高速通信区的空闲块数低于 SD796 ～ SD799 中设置的块数时，相应特殊继电器 (SM796 ～ SM799) 将变为 ON。



(b) 将 SM796 ～ SM799 作为互锁使用时的程序示例

X0 的上升沿时对 2 号机执行 D.DDWR 指令，X1 的上升沿时对 3 号机执行 D.DDWR 指令的程序如下所示。



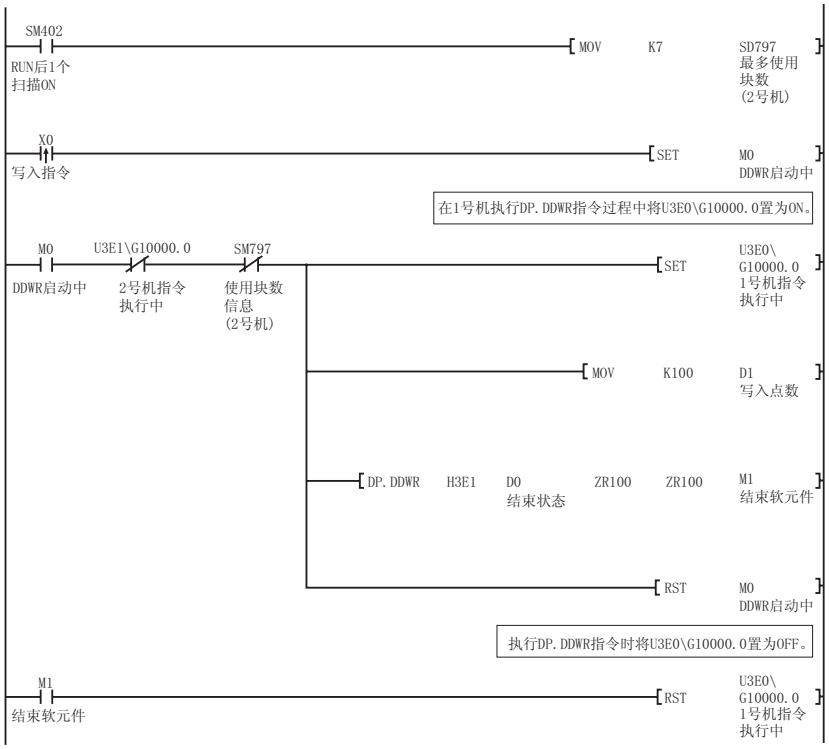
(8) 在多个 CPU 模块中相互执行多 CPU 高速通信专用指令时的程序示例

在通用型 QCPU 之间相互执行多 CPU 高速通信专用指令时，应采取互锁以防止同时执行多 CPU 高速通信专用指令。

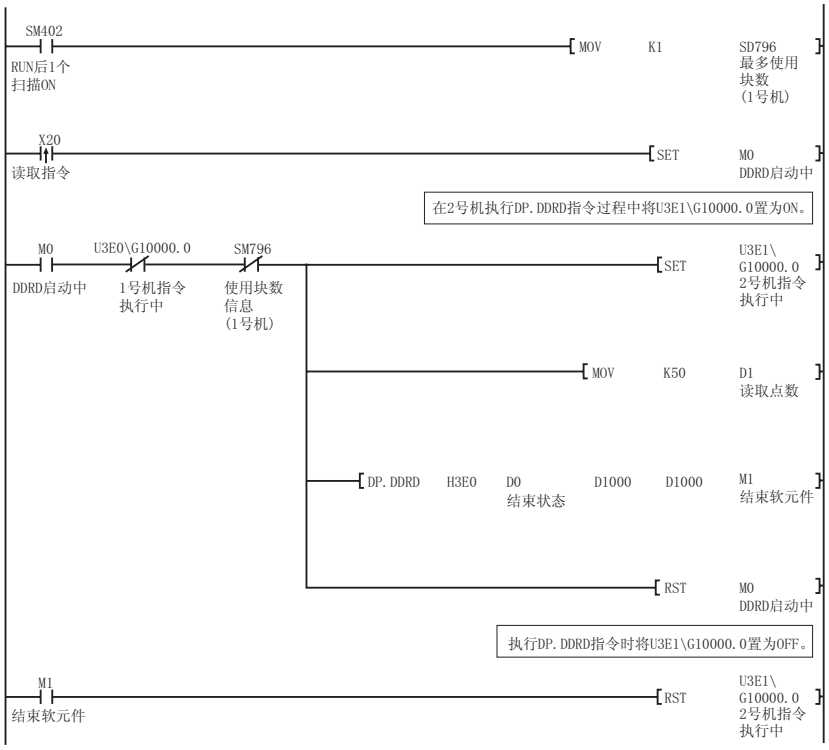
在互锁中使用多 CPU 共享软元件 (U3En\G10000 ～)。

在 1 号机与 2 号机之间相互执行多 CPU 高速通信专用指令时的程序示例如下所示。

在 1 号机中执行多 CPU 高速通信专用指令时的程序示例



在 2 号机中执行多 CPU 高速通信专用指令时的程序示例



(9) 通过多 CPU 高速通信专用指令对超过 100 字的数据进行写入 / 读取时的程序示例

多 CPU 高速通信专用指令可处理的写入 / 读取点数最多为 100 字。对超过 100 字的数据进行写入 / 读取时，通过多次执行多 CPU 高速通信专用指令来实现。

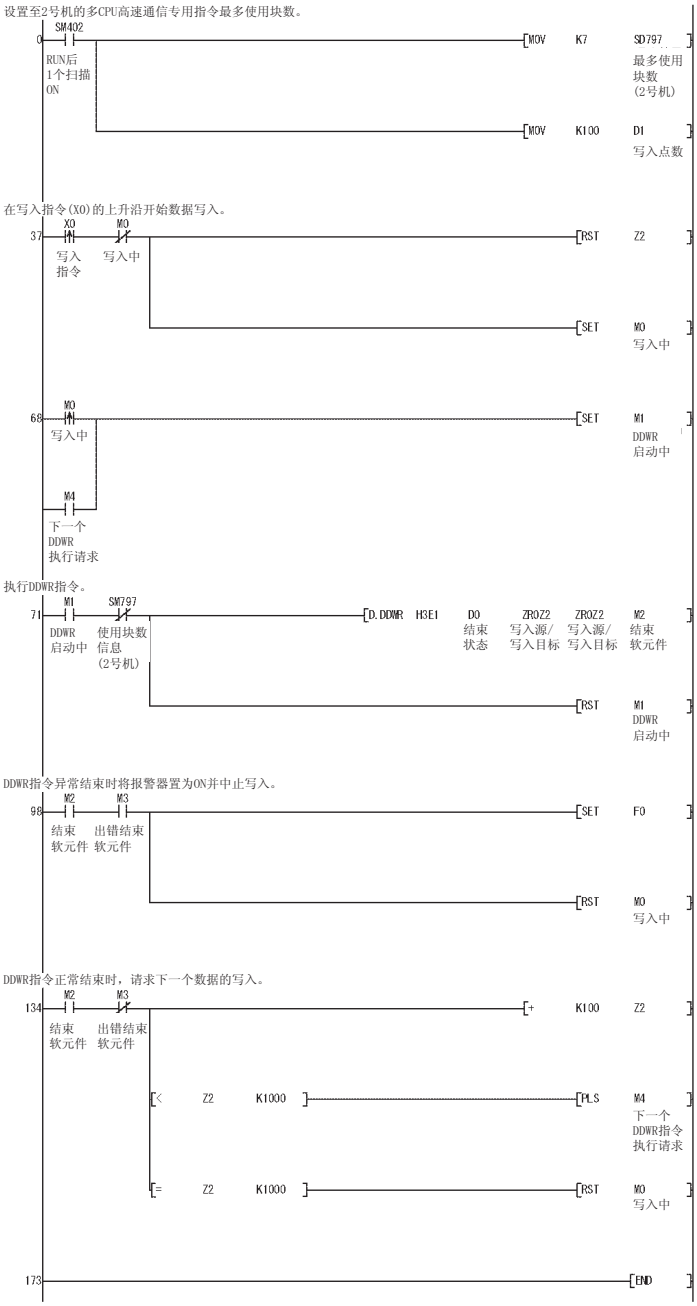
此外，下图所示为使用了多 CPU 高速通信专用指令的 D(P).DDWR 指令的程序示例，使用多 CPU 高速通信专用指令的 D(P).DDRD 指令时，可以使用与下图的程序示例相同构成的程序。

(a) 仅启动 1 个 D(P).DDWR 指令的程序示例

使用 D.DDWR 指令，将 1 号机的 ZR0 ～ ZR999(1000 点) 写入到 2 号机的 ZR0 ～ ZR999 中的程序示例如下所示。

在下图的示例中，通过 D.DDWR 指令的结束软元件 (M2) 的 ON，启动下一个 D.DDWR 指令，以实现每次仅执行 1 个 D.DDWR 指令。

仅启动 1 个 D(P).DDWR 指令的程序示例

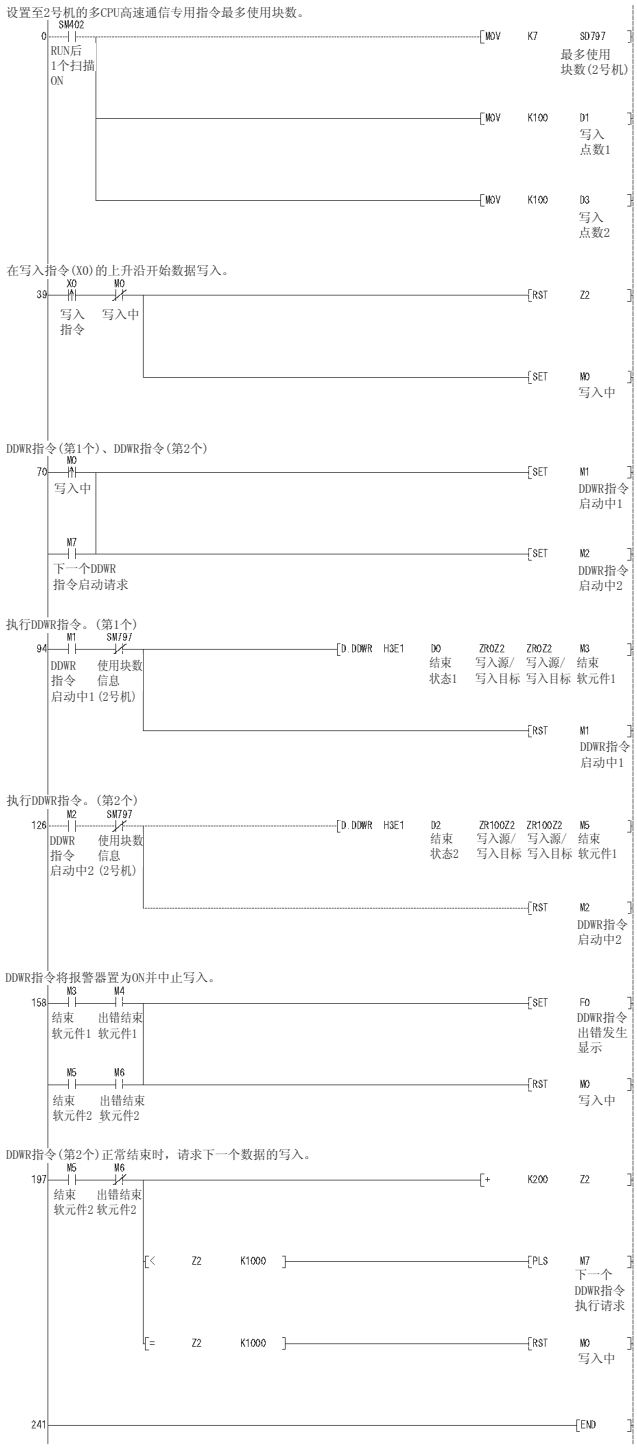


(b) 同时启动 2 个以上 D(P).DDWR 指令的程序示例

使用 D.DDWR 指令，将 1 号机的 ZR0 ～ ZR999(1000 点) 写入到 2 号机的 ZR0 ～ ZR999 中的程序示例如下所示。

如下图程序示例所示，同时启动 2 个以上多 CPU 高速通信专用指令的软元件写入 / 读取时，多 CPU 高速通信区 (发送区) 的总块数越多，则至多 CPU 高速通信专用指令的写入 / 读取结束为止的时间便可越短。

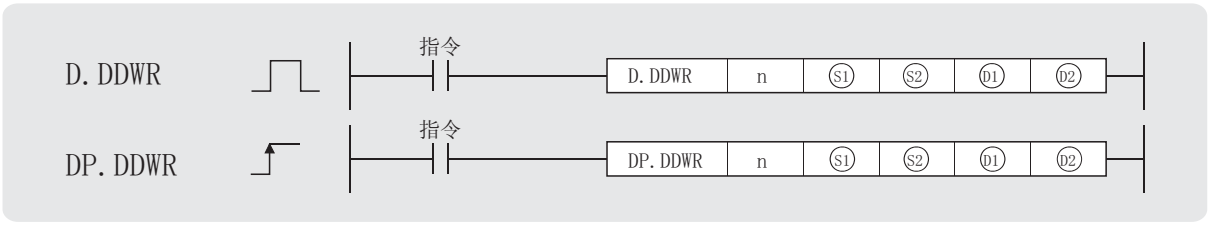
同时启动多个 D(P).DDWR 指令的程序示例



11.2 至其它站的软元件写入 (D(P).DDWR)

Basic High performance Process Redundant Ver. Universal

QO3UDCPU、QO4UDHCPU、QO6UDHCPU：序列号的前 5 位数为“10012”以后
QnUDE(H)CPU



设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字 *5		位	字				
n *1	–	○	○	–				○	–
(S1) *2	–	△ *3	△ *4	–				–	–
(S2) *2	–	○	○	–				–	–
(D1) *2	–	○	○	–				–	–
(D2) *2	△ *6	–	△ *4	–				–	–

*1：设置数据 n 不能进行变址修饰。
*2：设置数据(S1)～(D2)可以进行变址修饰。
*3：不能使用局部软元件。
*4：不能使用各程序的文件寄存器。
*5：不能使用 FD、@□（间接指定）。
*6：不能使用 FX、FY。

设置数据

设置数据	内容	数据类型
n	其它站 CPU 的起始 I/O 编号÷ 16 1 号机 :3E0H、2 号机 :3E1H、3 号机 :3E2H、4 号机 :3E3H	BIN16 位
(S1)	存储控制数据的自站 CPU 的起始软元件	软元件名
(S2)	存储写入数据的自站 CPU 的起始软元件	
(D1)	存储写入数据的其它站 CPU 的起始软元件	软元件名 *7 字符串 *8
(D2)	结束软元件	位

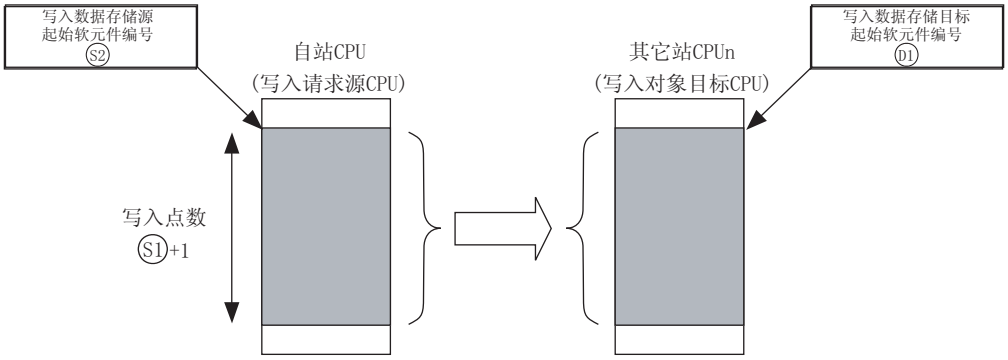
*7: 指定了文件寄存器 (R、ZR) 的情况下，在自站 CPU 中也可向超出范围的其它站 CPU 的软元件进行写入。
*8: 通过用字符串 “” 指定起始软元件，在执行本指令的自站 CPU 中也可向超出范围的其它站 CPU 的软元件进行写入。

控制数据

软元件	项目	设置数据	设置范围	设置方
(S1)+0	结束状态	存储指令结束时的执行结果。 0000(H)：无出错（正常结束） 0000(H) 以外：出错代码（异常结束）	-	系统
(S1)+1	写入数据点数	以字为单位设置写入数据点数。	1 ～ 100	用户

功　能

(1) 多 CPU 系统配置时，将自站 CPU 中指定的软元件 (S2) 后面的数据，以 (S1)+1) 中指定的写入数据点数，存储到其它站 CPU(n) 中指定的软元件 (D1) 后面。



(2) D(P).DDWR 指令的正常 / 异常结束可以通过结束软元件 (D2)+0)、结束时的状态显示软元件 (D2)+1) 进行确认。

(a) 结束软元件 (D2)+0)
在指令结束的扫描的 END 处理时变为 ON。在下一个 END 处理时变为 OFF。

(b) 结束时的状态显示软元件 (D2)+1)
根据指令结束时的状态而 ON/OFF。

- 正常结束时 :OFF
- 异常结束时：在指令结束的扫描的 END 处理时变为 ON。在下一个 END 处理时变为 OFF。
（异常结束时在控制数据 (S1)+0: 结束状态）中存储出错代码。）

(3) 指令中使用的块数取决于写入数据点数。(参阅 11.1 节)

指令中使用的块数

指令中指定的写入点数	D(P).DDWR 指令
1 ～ 4	1
5 ～ 20	2
21 ～ 36	3
37 ～ 52	4
53 ～ 68	5
69 ～ 84	6
85 ～ 100	7

(4) 在多 CPU 高速通信区中无空闲块的情况下，即使执行指令也将异常结束。通过在特殊寄存器 (SD796 ～ SD799) 中设置指令中使用的块数，将特殊继电器 (SM796 ～ SM799) 用作互锁，可以防止异常结束。(参阅 11.1 节)

出 错

在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

(1) 指定的其它站 CPU 有错误时。或者，进行了多 CPU 高速通信专用指令不能使用的设置时。 (出错代码：4350)

- 指定了已进行了预约设置的机号时。
- 指定了未安装的机号时。
- 其它站 CPU 起始 I/O 编号÷ 16n 超出了 3E0H ～ 3E3H 的范围时。
- 在设置为“不使用多 CPU 高速通信功能”的情况下执行了本指令。
- 在 Q02UCPU 中执行了本指令时。
- 指定了自站 CPU 时。
- 指定了不能执行指令的 CPU 时。

(2) 本指令不能在 CPU 中执行时。 (出错代码：4351)

- 其它站 CPU 不支持本指令。

(3) 软元件数有错误时。 (出错代码：4352)

(4) 指定了不能使用的软元件时。 (出错代码：4353)

(5) 以不能处理的字符串指定了软元件时。 (出错代码：4354)

(6) 写入数据点数 (Ⓢ1+1) 超出了 0 ～ 100 的范围时。 (出错代码：4354)

在以下情况下，出错代码将被存储到异常结束的结束状态存储软元件 (Ⓢ1+0) 中指定的软元件中。

(1) 至对象目标 CPU 的指令请求超出了允许值时。(多 CPU 高速通信区中无空闲块) (出错代码：0010H)

(1) Ⓓ1中指定的其它站 CPU 的软元件是在其它站 CPU 中不能使用的软元件，或者超出了软元件范围时。 (出错代码：1001H)

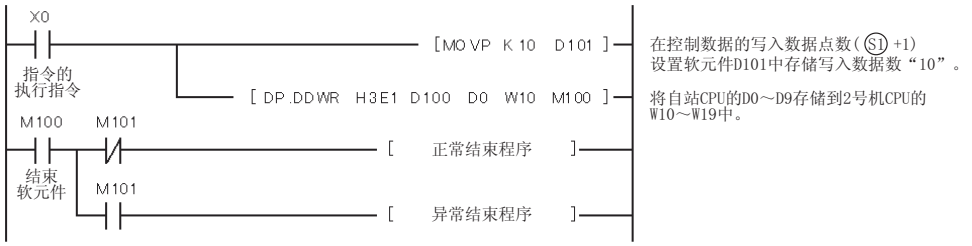
(1) D(P).DDWR 指令中设置的写入数据点数为 0 时。 (出错代码：1080H)

(1) 未从其它站 CPU 模块返回指令响应。(多 CPU 高速通信区中无空闲块) (出错代码：1003H)

程序示例

(1) 以下为 X0 变为 ON 时，将自站 CPU 的 D0 开始的 10 字数据存储到 2 号机 CPU 的 W10 后面的程序。

[梯形图模式]



注意事项

(1) n、(S2)以及(D1)中可以进行位软元件的位数指定。但是，在(S2)以及(D1)中进行位软元件的位数指定时，需要满足下述条件。

- 是 16 位 (4 位数) 的位数指定。
- 开始位软元件为 16(10H) 的倍数。

(2) 应在写入对象的 CPU 处于启动的状态下执行本指令。

如果在写入对象 CPU 处于未启动状态时执行本指令，将变为无处理。

(3) 执行本指令后，在结束软元件变为 ON 之前，如果对设置数据中指定的软元件的范围等进行了变更，系统中存储的数据 (结束状态、结束软元件) 将无法正常存储。

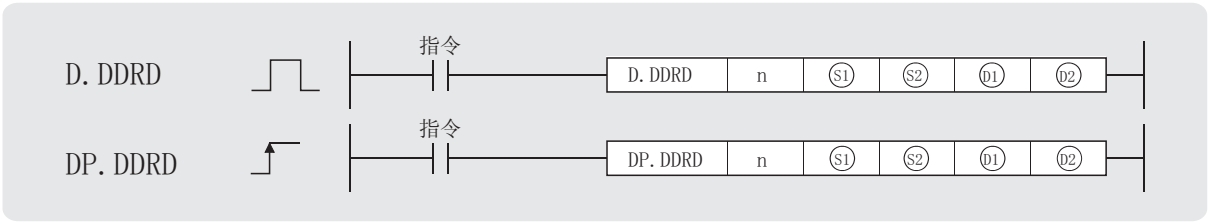
(4) SB、SW、SM、SD 中包含有系统信息区。

通过多 CPU 高速通信专用指令的 D(P).DDWR 指令对上述软元件进行写入时，应注意防止破坏系统信息。

11.3 从其它站读取软元件 (D(P).DDRD)

Q03UDCPU、Q04UDHCPU、Q06UDHCPU：序列号的前 5 位数为“10012”以后

QnUDE(H)CPU



设置数据	内部软元件		R、ZR	J□\□		U□\G□	Zn	常数 K、H	其它
	位	字 *5		位	字				
n *1	–	○	○	–				○	–
(S1) *2	–	△ *3	△ *4	–				–	–
(S2) *2	–	○	○	–				–	–
(D1) *2	–	○	○	–				–	–
(D2) *2	△ *6	–	△ *4	–				–	–

*1：设置数据 n 不能进行变址修饰。

*2：设置数据(S1)～(D2)可以进行变址修饰。

*3：不能使用局部软元件。

*4：不能使用各程序的文件寄存器。

*5：不能使用 FD、@□（间接指定）。

*6：不能使用 FX、FY。

设置数据

设置数据	内容	数据类型
n	其它站 CPU 的起始 I/O 编号÷ 16 1 号机：3E0H、2 号机：3E1H、3 号机：3E2H、4 号机：3E3H	BIN16 位
(S1)	存储控制数据的自站 CPU 的起始软元件	软元件名
(S2)	存储读取数据的其它站 CPU 的起始软元件	
(D1)	存储读取数据的自站 CPU 的起始软元件	软元件名 *7 字符串 *8
(D2)	结束软元件	位

*7： 指定了文件寄存器（R、ZR）的情况下，在自站 CPU 中也可向超出范围的其它站 CPU 的软元件进行读取。

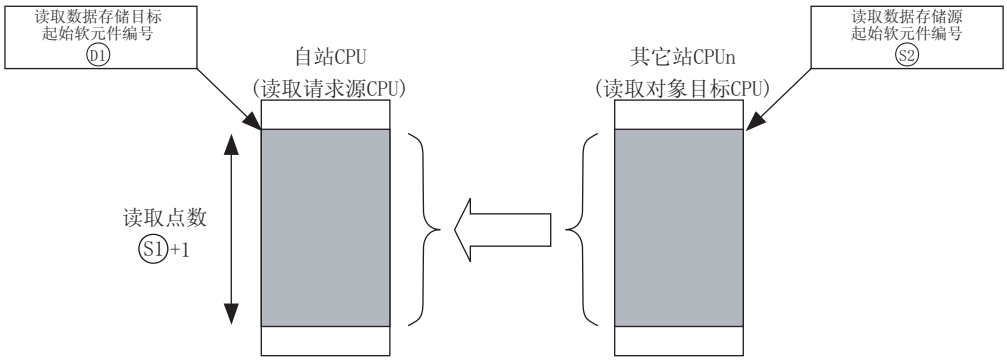
*8： 通过用字符串 “” 指定起始软元件，在执行本指令的自站 CPU 中也可向超出范围的其它站 CPU 的软元件进行读取。

控制数据

软元件	项目	设置数据	设置范围	设置方
ⓢ1+0	结束状态	存储指令结束时的执行结果。 0000(H): 无出错（正常结束） 0000(H) 以外： 出错代码（异常结束）	-	系统
ⓢ1+1	读取数据点数	以字为单位设置读取数据点数。	1 ～ 100	用户

功　能

(1) 多 CPU 系统配置时，将其它站 CPU(n) 中指定的软元件 (ⓢ2) 后面的数据，以 (ⓢ1+1) 中指定的读取数据点数，存储到自站 CPU 中指定的软元件 (ⓓ1) 后面。



(2) D(P).DDRD 指令的正常 / 异常结束可以通过结束软元件 (ⓓ2+0)、结束时的状态显示软元件 (ⓓ2+1) 进行确认。

(a) 结束软元件 (ⓓ2+0)
在指令结束的扫描的 END 处理时变为 ON。在下一个 END 处理时变为 OFF。

(b) 结束时的状态显示软元件 (ⓓ2+1)
根据指令结束时的状态而 ON/OFF。

- 正常结束时 :OFF
- 异常结束时：在指令结束的扫描的 END 处理时变为 ON。在下一个 END 处理时变为 OFF。（异常结束时在控制数据（ⓢ1+0: 结束状态）中存储出错代码。）

(3) 指令中使用的块数取决于读取数据点数。（参阅 12.1 节）

指令中使用的块数

指令中指定的读取点数	D(P).DDRD 指令
1 ～ 100	1

(4) 在多 CPU 高速通信区中无空闲块的情况下，即使执行指令也将异常结束。通过在特殊寄存器 (SD796 ～ SD799) 中设置指令中使用的块数，将特殊继电器 (SM796 ～ SM799) 用作互锁，可以防止异常结束。（参阅 11.1 节）

出 错

在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

(1) 指定的其它站 CPU 有错误时。或者，进行了多 CPU 高速通信专用指令不能使用的设置时。
（出错代码 :4350)

- 指定了已进行了预约设置的机号时。
- 指定了未安装的机号时。
- 其它站 CPU 起始 I/O 编号÷ 16n 超出了 3E0H ～ 3E3H 的范围时。
- 在设置为 “不使用多 CPU 高速通信功能”的情况下执行了本指令。
- 在 Q02UCPU 中执行了本指令时。
- 指定了自站 CPU 时。
- 指定了不能执行指令的 CPU 时。

(2) 本指令不能在 CPU 中执行时。（出错代码： 4351)
- 其它站 CPU 不支持本指令。

(3) 软元件数有错误时。（出错代码： 4352)

(4) 指定了不能使用的软元件时。（出错代码： 4353)

(5) 以不能处理的字符串指定了软元件时。（出错代码： 4354)

(6) 读取数据点数 (Ⓢ1+1) 超出了 0 ～ 100 的范围时。（出错代码： 4355)

在以下情况下，出错代码将被存储到异常结束的结束状态存储软元件 (Ⓢ1+0) 中指定的软元件中。

(1) 至对象目标 CPU 的指令请求超出了允许值时。(多 CPU 高速通信区中无空闲块)
（出错代码： 0010H)

(1) Ⓢ2 中指定的其它站 CPU 的软元件是在其它站 CPU 中不能使用的软元件，或者超出了软元件范围时。（出错代码： 1001H)

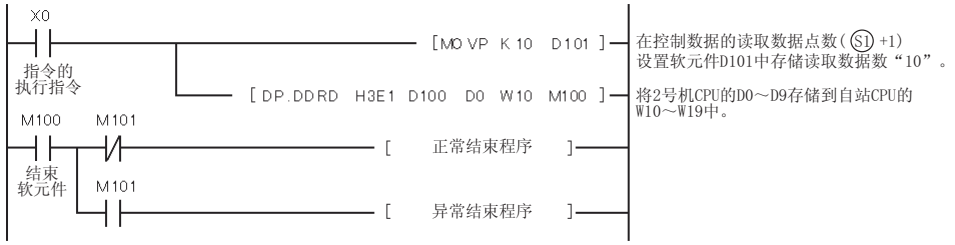
(1) D(P).DDRD 指令中设置的读取数据点数为 0 时。
（出错代码： 1081H)

(1) 未从其它站 CPU 模块返回指令响应。(多 CPU 高速通信区中无空闲块)
（出错代码： 1003H)

程序示例

(1) 以下为 X0 变为 ON 时，将 2 号机 CPU 的 D0 开始的 10 字数据存储到自站 CPU 的 W10 后面的程序。

[梯形图模式]



注意事项

(1) n、Ⓢ2以及Ⓓ1中可以进行位软元件的位数指定。但是，在Ⓢ2以及Ⓓ1中进行位软元件的位数指定时，需要满足下述条件。

- 是 16 位（4 位数）的位数指定。
- 开始位软元件为 16(10H) 的倍数。

(2) 应在读取对象的 CPU 处于启动的状态下执行本指令。
如果在读取对象 CPU 处于未启动状态时执行本指令，将变为无处理。

(3) 执行本指令后，在结束软元件变为 ON 之前，如果对设置数据中指定的软元件的范围等进行了变更，系统中存储的数据（结束状态、结束软元件）将无法正常存储。

12 出错代码

12.1 出错代码一览表

CPU 模块在可编程控制器电源 ON 时，可编程控制器 RUN 时或者 RUN 中发生了异常时，通过自诊断功能进行出错显示 (LED 显示、显示器的信息显示)，并将出错信息存储到特殊继电器 SM、特殊寄存器 SD 中。

此外，从外围设备、智能功能模块以及网络系统向 CPU 模块发出通信请求时如果发生的出错，将向请求源返回出错代码 (4000H ～ 4FFFH)。

以下介绍 CPU 模块中发生的出错内容及出错相应处理方法。

(1) 出错代码一览表的阅读方法

12.1.3 项出错代码一览表 (1000 ～ 1999) ～ 12.1.9 项出错代码一览表 (7000 ～ 10000) 的阅读方法如下所示。

(a) 关于出错代码、公共信息、个别信息

出错代码、公共信息、个别信息的各标题栏的 () 内的字符表示存储各信息的特殊寄存器编号。

(b) 关于对应 CPU

QCPU	: 对应于所有的 Q 系列 CPU 模块。
Q00J/Q00/Q01	: 对应于基本型 QCPU。
Qn(H)	: 对应于高性能型 QCPU。
QnPH	: 对应于过程 CPU。
QnPRH	: 对应于冗余 CPU。
QnU	对应于通用型 QCPU。
各 CPU 模块型号	: 仅对应于记述的 CPU 模块。(例 : Q02U)

12.1.1 出错代码

出错中包含有通过 CPU 模块的自诊断功能检测出的出错及与 CPU 模块通信时检测出的出错。
出错的检测类型、出错检测位置以及出错代码的关系如下表所示。

出错检测类型	出错检测位置	出错代码	出错内容的参阅章节
通过 CPU 模块的自诊断功能检测出	CPU 模块	1000 ～ 10000*1*2	12.1.3 项～ 12.1.9 项
与 CPU 模块通信时检测出	CPU 模块	4000H ～ 4FFFH	• QCPU 用户手册（硬件设计 / 维护点检篇）
	串行通信模块等	7000H ～ 7FFFH	串行通信模块用户手册等
	CC-Link 模块	B000H ～ BFFFH	CC-Link 系统主站 • 本地站模块用户手册
	以太网模块	C000H ～ CFFFH	以太网接口模块用户手册
	CC-Link IE 控制网络模块	E000H ～ EFFFH	CC-Link IE 控制网络参考手册
	MELSECNET/H 网络模块	F000H ～ FFFFH	• Q 系列 MELSECNET/H 网络系统参考手册 • MELSECNET/10 模式 QnA/Q4AR 系列 MELSECNET/10 网络系统参考手册

*1 : CPU 模块的出错代码被分为轻度异常、中度异常、严重异常。
• 轻度异常 : 电池出错等 CPU 模块继续运行型出错。
• 中度异常 : WDT 出错等 CPU 模块停止运行型出错。(出错代码 : 1300 ～ 10000)
• 严重异常 : RAM 异常等 CPU 模块停止运行型出错。(出错代码 : 1000 ～ 1299)
“继续运行型出错”与 “停止运行型出错”可以通过 12.1.3 项～ 12.1.9 项出错代码一览表的 “CPU 动作状态”进行判别。

*2 : 检测出参阅章节的出错代码表中未记述的出错代码时，请向附近的分公司 • 代理处咨询。

12.1.2 出错代码的读取方法

发生了出错时，可以通过 GX Developer 读取出错代码、出错信息等。
有关操作方法的详细内容，请参阅 GX Developer 操作手册。

12.1.3 出错代码一览表 (1000 ～ 1999)

以下介绍出错代码 1000 ～ 1999 的出错信息、异常内容及原因以及处理方法等。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
1000	[MAIN CPU DOWN] CPU 模块死机或故障。 ・噪声等导致误动作。 ・硬件异常。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・常时	・采取防噪声措施。 ・对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是 CPU 模块硬件故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）	RUN：熄灯 ERR.：闪烁 CPU 状态：停止	QCPU
1001	[MAIN CPU DOWN] CPU 模块死机或故障。 ・噪声等导致误动作。 ・硬件异常。 ・在软元件范围检查禁止状态 (SM237=ON) 下访问了超出范围的软元件。（仅在执行 BMOV、FMOV、DFMOV 指令时发生）（仅通用型 QCPU） **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・常时	・采取防噪声措施。 ・对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是 CPU 模块硬件故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。） ・对 BMOV、FMOV、DFMOV 指令中指定的软元件进行重新审核、修正。（仅通用型 QCPU）		
1002 1003 1004 1005 1006 1007 1008	[MAIN CPU DOWN] CPU 模块死机或故障。 ・噪声等导致误动作。 ・硬件异常。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・常时	・采取防噪声措施。 ・对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是 CPU 模块硬件故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		Qn(H) QnPH QnPRH
1009	[MAIN CPU DOWN] ・检测出电源模块、CPU 模块、主基板、扩展基板或者扩展电缆的故障。 ・使用冗余基板时，2 个冗余电源模块均检测出故障。 或者，检测出冗余基板故障。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・常时	・对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是电源模块、CPU 模块、主基板、扩展基板或者扩展电缆故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		Q00J/Q00/Q01 *4 Qn(H)*6 QnPH QnPRH QnU

*4　以功能版本 B 以后为对象。
*6　以序列号的前 5 位数为 “04101” 以后的模块为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
1010	[END NOT EXECUTE] 在未执行 END 指令的状况下执行了全部程序容量的程序。 ・执行 END 指令时，由于噪声等被读取为其它的指令代码。 ・END 指令由于某种原因变为了其它指令代码。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・执行 END 指令时	・采取防噪声措施。 ・对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是 CPU 模块硬件故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		QCPU
1020	[SFCP. END ERROR] SFC 程序未能正常结束。 ・由于噪声等导致 SFC 程序未能正常结束。 ・由于某种原因 SFC 程序未能正常结束。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・执行 SFC 程序时			Q00J/Q00/Q01 *4 QnPH QnU
1035	[MAIN CPU DOWN] CPU 模块死机或故障。 ・噪声等导致误动作。 ・硬件异常。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・常时	・采取防噪声措施。 ・对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是 CPU 模块硬件故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）	RUN：熄灯 ERR.：闪烁 CPU 状态：停止	QnU
1101	[RAM ERROR] CPU 模块内的顺控程序存储用程序存储器异常。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时 / 执行 END 指令时	・采取防噪声措施。 ・对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是 CPU 模块硬件故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		QCPU
1102	[RAM ERROR] ・CPU 模块内的工作区用的 RAM 异常。 ・CPU 模块内的标准 RAM、扩展 RAM 异常。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时 / 执行 END 指令时	・采取防噪声措施。 ・对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是 CPU 模块硬件故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		QCPU

*4　以功能版本 B 以后为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
1103	[RAM ERROR] CPU 模块内的软元件存储器异常。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时	・采取防噪声措施。 ・进行了变址修饰时，确认变址寄存器的值是否超出了软元件的范围。 ・对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是 CPU 模块硬件故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）	RUN：熄灯 ERR.：闪烁 CPU 状态：停止	QCPU
	[RAM ERROR] ・CPU 模块内的软元件存储器异常。 ・通过变址修饰进行了超出软元件范围的访问，对系统用软元件进行了改写。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时 / 执行 END 指令时			Qn(H)*8 QnPH*8 QnPRH*9
1104	[RAM ERROR] CPU 模块内的地址 RAM 异常。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时			QCPU
1105	[RAM ERROR] CPU 模块内的 CPU 存储器异常。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时	・采取防噪声措施。 ・对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是 CPU 模块硬件故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		Q00J/Q00/Q01 QnU
	[RAM ERROR] CPU 模块内的 CPU 共享存储器异常。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时			Qn(H)*4 QnPH QnPRH QnU
1106	[RAM ERROR] 电池用完。 CPU 模块的程序存储器异常。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・STOP → RUN 时 / 执行 END 指令时	・确认电池是否用完，如果用完则更换电池。 ・采取防噪声措施。 ・对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是 CPU 模块硬件故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		Qn(H) QnPH*7 QnPRH

*4　以功能版本 B 以后为对象。
*7　以序列号的前 5 位数为 “07032” 以后的模块为对象。
*8　以序列号的前 5 位数为 “08032” 以后的模块为对象。
*9　以序列号的前 5 位数为 “09012” 以后的模块为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
1107	[RAM ERROR] CPU 模块内的工作区用的 RAM 异常。 ■附加信息 ・公共信息： － ・个别信息： － ■诊断时机 ・电源 ON 时 / 复位时	CPU 模块硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）	RUN：熄灯 ERR.：闪烁 CPU 状态：停止	QnPRH
1108				Qn(H)*8 QnPH*8 QnPRH*9
1109	[RAM ERROR] CPU 模块内的工作区用的 RAM 异常。 ■附加信息 ・公共信息： － ・个别信息： － ■诊断时机 ・常时			
1110	[TRK. CIR. ERROR] 热备用硬件的初始化检查中检测出异常。 ■附加信息 ・公共信息： － ・个别信息： － ■诊断时机 ・电源 ON 时 / 复位时			QnPRH
1111	[TRK. CIR. ERROR] 热备用硬件检测出异常。 ■附加信息 ・公共信息： － ・个别信息： － ■诊断时机 ・电源 ON 时 / 复位时			
1112	[TRK. CIR. ERROR] ・运行过程中检测出热备用硬件异常。 ・在未对待机系统进行电源 OFF 或者复位的状态下插拔了热备电缆。 ・热备电缆未用连接器固定螺栓固定。 ・未遵守冗余系统的启动步骤，启动时变为出错状态。 ■附加信息 ・公共信息： － ・个别信息： － ■诊断时机 ・运行过程中	・应确认热备电缆已安装后再启动。 如果仍然显示相同的出错，则可能是热备电缆或者 CPU 模块的硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。） ・确认冗余系统的启动步骤后进行再启动。详细内容请参阅 QnPRHCPU 用户手册（冗余系统篇）。		
1113				
1115	[TRK. CIR. ERROR] 热备用硬件的初始化检查中检测出异常。 ■附加信息 ・公共信息： － ・个别信息： － ■诊断时机 ・电源 ON 时 / 复位时	CPU 模块硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		

*8　以序列号的前 5 位数为 “08032” 以后的模块为对象。
*9　以序列号的前 5 位数为 “09012” 以后的模块为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
1116	[TRK. CIR. ERROR] ・运行过程中检测出热备用硬件异常。 ・在未对待机系统进行电源 OFF 或者复位的状态下插拔了热备电缆。 ・热备电缆未用连接器固定螺栓固定。 ・未遵守冗余系统的启动步骤，启动时变为出错状态。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・运行过程中	・应确认热备电缆已安装后再启动。 如果仍然显示相同的出错，则可能是热备电缆或者 CPU 模块的硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。） ・确认冗余系统的启动步骤后进行再启动。详细内容请参阅 QnPRHCPU 用户手册（冗余系统篇）。	RUN：熄灯 ERR.：闪烁 CPU 状态：停止	QnPRH
1150	[RAM ERROR] CPU 模块内多 CPU 高速通信区的存储器异常。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时	・采取防噪声措施。 ・对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是 CPU 模块硬件故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		QnU*10
1160	[RAM ERROR] CPU 模块内的程序存储器的数据被改写。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・执行程序时	・采取防噪声措施。 ・对程序存储器进行格式化后，在对全部文件进行可编程控制器写入后，对 CPU 模块进行复位后再次置为 RUN。 仍然显示相同的出错时，可能是 CPU 模块硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		QnU
1161	[RAM ERROR] CPU 模块内的内置软元件存储器的数据被改写。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・常时	・采取防噪声措施。 仍然显示相同的出错时，可能是 CPU 模块硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		
1162	[RAM ERROR] CPU 模块内的由电池保持的数据检测出异常。 （在未设置自动格式化的情况下发生的） **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时	・对 CPU 模块或者 SRAM 卡的电池进行更换。 ・采取防噪声措施。 仍然显示相同的出错时，可能是 CPU 模块硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		QnU
1164	[RAM ERROR] 检测出标准 RAM 中使用的存储器的数据已被破坏。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・执行指令时	・采取防噪声措施。 仍然显示相同的出错时，可能是 CPU 模块硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		QnU*11

*10　以除 Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU 以外的通用型 QCPU 为对象。
*11　以 Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU 为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
1200	[OPE. CIRCUIT ERR.] CPU 模块内的进行变址修饰的运算电路不能正常动作。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时	CPU 模块硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）	RUN：熄灯 ERR.：闪烁 CPU 状态： 停止	QCPU
1201	[OPE. CIRCUIT ERR.] CPU 模块内的硬件（逻辑）不能正常动作。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时			
1202	[OPE. CIRCUIT ERR.] CPU 模块内的进行顺控程序处理的运算电路不能正常动作。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时			
1203	[OPE. CIRCUIT ERR.] CPU 模块内的进行变址修饰的运算电路不能正常动作。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・执行 END 指令时			QnPRH
1204	[OPE. CIRCUIT ERR.] CPU 模块内的硬件（逻辑）不能正常动作。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・执行 END 指令时			
1205	[OPE. CIRCUIT ERR.] CPU 模块内的进行顺控程序处理的运算电路不能正常动作。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・执行 END 指令时			

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
1300	[FUSE BREAK OFF] 存在有保险丝熔断的输出模块。 **■附加信息** ・公共信息： 模块号（插槽号） ［远程 I/O 网时］ 网络号 / 站号 ・个别信息： － **■诊断时机** ・常时	・确认输出模块的 FUSE LED，对亮灯的模块进行更换。 （保险丝熔断的模块也可通过 GX Developer 进行确认。 确认特殊寄存器 SD1300 ～ SD1331 中保险丝熔断的模块所对应的位是否变为“1”。） ・将 GOT 与主基板或者扩展基板通过总线相连接时，确认扩展电缆的连接状态及 GOT 的接地状态。	RUN：熄灯 / 亮灯 ERR.： 闪烁 / 亮灯	Qn(H) QnPH QnPRH QnU
	[FUSE BREAK OFF] 存在有保险丝熔断的输出模块。 **■附加信息** ・公共信息： 模块号（插槽号） ［远程 I/O 网时］ 网络号 / 站号 ・个别信息： － **■诊断时机** ・常时	确认输出模块的ERR LED，对亮灯的模块进行更换。（保险丝熔断的模块也可通过 GX Developer 进行确认。 确认特殊寄存器 SD130 ～ SD137 中保险丝熔断的模块所对应的位是否变为“1”。）	CPU 状态： 停止 / 继续运行 *1	Q00J/Q00/Q01
1310	[I/O INT. ERROR] 未安装中断模块但却发生了中断。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・发生中断时	某个安装的模块的硬件故障。检查安装的模块，对故障模块进行更换。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）	RUN：熄灯 ERR.：闪烁 CPU 状态：停止	QCPU
1311	[I/O INT. ERROR] 检测出从不是中断模块的模块中发出的中断请求。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・发生中断时	采取措施，避免从不是中断模块的模块中发出中断请求。		Q00J/Q00/Q01*4 QnU
	[I/O INT. ERROR] 检测出从未在可编程控制器参数中进行中断指针设置的模块中发出了中断请求。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・发生中断时	・对可编程控制器参数的可编程控制器系统设置的中断指针设置进行修改。 ・采取措施，避免未进行可编程控制器参数的可编程控制器系统设置的中断指针设置的模块中发出中断请求。对网络参数的中断设置进行修改。对智能功能模块的缓冲存储器的中断设置进行修改。对 QD51 的基本程序进行修改。		Q00J/Q00/Q01*5 QnPRH QnU
1320 1321	[LAN CTRL. DOWN] 根据硬件自诊断检测出 LAN 控制器故障。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时	CPU 模块硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		QnU*13

*1　发生出错时的 CPU 的动作状态可以在参数中进行设置。(LED 显示也将联动变化。)
*4　以功能版本 B 以后为对象。
*5　以功能版本 A 为对象。
*13　以以太网板内置的 QCPU 为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
1401	[SP. UNIT DOWN] ・初始化处理时未从智能功能模块 / 特殊功能模块返回信息。 ・智能功能模块 / 特殊功能模块的缓冲存储器的大小异常。 ・安装了不支持的模块。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时 / 智能访问时	安装了不支持的模块时，将该模块卸下。相应模块是可支持的模块时，可能是智能功能模块 / 特殊功能模块、CPU 模块或者基板的硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		QCPU
1402	[SP. UNIT DOWN] 通过程序访问智能功能模块 / 特殊功能模块时未返回信息。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： 程序出错位置 **■诊断时机** ・执行智能访问指令时	智能功能模块 / 特殊功能模块、CPU 模块或者基板的硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）	RUN: 熄灯 / 亮灯 ERR.: 闪烁 / 亮灯	QCPU
1403	[SP. UNIT DOWN] ・安装了不支持的模块。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： － **■诊断时机** ・执行 END 指令时	安装了不支持的模块时，将该模块卸下。相应模块是可支持的模块时，可能是智能功能模块 / 特殊功能模块、CPU 模块或者基板的硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）	CPU 状态： 停止 / 继续运行 *2	QCPU
	[SP. UNIT DOWN] ・执行 END 指令时未从访问智能功能模块 / 特殊功能模块返回信息。 ・检测出智能功能模块 / 特殊功能模块发生了异常。 ・在运行过程中对输入输出模块（包括智能功能模块 / 特殊功能模块）进行了拆装。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： － **■诊断时机** ・常时	CPU 模块、基板或者访问目标智能功能模块 / 特殊功能模块的硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		

*2 对于 QCPU，可以通过参数对各智能功能模块进行出错停止 / 继续运行的选择。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
1411	[CONTROL-BUS. ERR.] 在进行了可编程控制器参数的 I/O 分配设置的情况下，初始化通信时不能对智能功能模块 / 特殊功能模块进行访问。（发生出错时，对象智能功能模块 / 特殊功能模块的起始输入输出编号将被存储到公共信息中。） **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时	对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是智能功能模块 / 特殊功能模块、CPU 模块或者基板的硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）	RUN：熄灯 ERR.：闪烁 CPU 状态：停止	QCPU
1412	[CONTROL-BUS. ERR.] 由于智能功能模块 / 特殊功能模块及控制总线异常，不能执行 FROM/TO 指令。（发生出错时，对象的程序出错位置将被存储到个别信息中。） **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： 程序出错位置 **■诊断时机** ・执行 FROM/TO 指令时			
1413	[CONTROL-BUS. ERR.] 在多 CPU 系统中有，安装了不支持多 CPU 系统的 CPU 模块。 **■附加信息** ・公共信息： - ・个别信息： - **■诊断时机** ・常时	・将不支持多 CPU 系统的 CPU 模块从主基板上卸下。或者将不支持多 CPU 系统的 CPU 模块更换为支持多 CPU 系统的 CPU 模块。 ・智能功能模块、CPU 模块或者基板故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		Q00J/Q00/Q01 *4 Qn(H)*4 QnPH
	[CONTROL-BUS. ERR.] 检测出系统总线上有异常。 ・系统总线自诊断出错 ・CPU 模块的自诊断出错 **■附加信息** ・公共信息： - ・个别信息： - **■诊断时机** ・常时	对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是智能功能模块、CPU 模块或者基板故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		QCPU
1414	[CONTROL-BUS. ERR.] ・检测出安装模块异常。 ・在多 CPU 系统中，安装了不支持多 CPU 系统的 CPU 模块。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： - **■诊断时机** ・常时	・将不支持多 CPU 系统的 CPU 模块从主基板上卸下。或者将不支持多 CPU 系统的 CPU 模块更换为支持多 CPU 系统的 CPU 模块。 ・对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是智能功能模块、CPU 模块或者基板故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。））		Q00J/Q00/Q01*4 Qn(H)*4 QnPH QnU
	[CONTROL-BUS. ERR.] 检测出系统总线上有异常。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： - **■诊断时机** ・常时	对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是智能功能模块、CPU 模块或者基板故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		Q00J/Q00/Q01*4 Qn(H) QnPH QnPRH QnU

*4 以功能版本 B 以后为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
1415	[CONTROL-BUS. ERR.] 检测出主基板以及扩展基板的异常。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： － **■诊断时机** ・执行 END 指令时	对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是智能功能模块、CPU 模块或者基板故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）	RUN：熄灯 ERR.：闪烁 CPU 状态： 停止	Q00J/Q00/Q01 Qn(H)*4 QnPH QnPRH QnU
	[CONTROL-BUS. ERR.] 检测出主基板以及扩展基板的异常。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时 / 执行 END 指令时			Qn(H)*8 QnPH*8
1416	[CONTROL-BUS. ERR.] 电源 ON 或者复位时检测出系统总线的异常。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时			Qn(H)*4 QnPH QnU
	[CONTROL-BUS. ERR.] 在多 CPU 系统中，电源 ON 或者复位时检测出系统总线的异常。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时	对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是智能功能模块、CPU 模块或者基板故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		Q00/Q01*4 QnU
1417	[CONTROL-BUS. ERR.] 检测出系统总线上的复位信号异常。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・常时			QnPRH
1418	[CONTROL-BUS.ERR.] 在冗余系统中，电源 ON/ 复位时，或者系统切换时控制系统向扩展基板的访问权获取失败，无法对扩展基板进行访问。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时 / 执行切换时	对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是 CPU 模块、Q6 □ WRB 或者扩展电缆的硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		QnPRH*9

*4 以功能版本 B 以后为对象。
*8 以序列号的前 5 位数为 “08032” 以后的模块为对象。
*9 以序列号的前 5 位数为 “09012” 以后的模块为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
1430	[MULTI-C.BUS ERR.] 检测出多 CPU 高速通信中自站 CPU 的异常。 ■附加信息 ・公共信息： － ・个别信息： － ■诊断时机 ・电源 ON 时 / 复位时	对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是智能功能模块、CPU 模块或者基板故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）	RUN： 熄灯 ERR.： 闪烁 CPU 状态： 停止	QnU*10
1431	[MULTI-C.BUS ERR.] 检测出多 CPU 高速通信中其它站 CPU 的异常。 ■附加信息 ・公共信息： 模块号（站号） ・个别信息： － ■诊断时机 ・电源 ON 时 / 复位时	・采取防噪声措施。 ・确认 CPU 模块的主基板安装状态。 ・对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是 CPU 模块的硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		
1432	[MULTI-C.BUS ERR.] 检测出与多 CPU 高速通信中其它站 CPU 的通信异常。 ■附加信息 ・公共信息： 模块号（站号） ・个别信息： － ■诊断时机 ・电源 ON 时 / 复位时	对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是 CPU 模块的硬件故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		
1433 1434 1435	[MULTI-C.BUS ERR.] 检测出与多 CPU 高速通信中其它站 CPU 的通信异常。 ■附加信息 ・公共信息： 模块号（站号） ・个别信息： － ■诊断时机 ・常时	・采取防噪声措施。 ・确认 CPU 模块的主基板安装状态。 ・对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是 CPU 模块的硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		
1436	[MULTI-C.BUS ERR.] 检测出多 CPU 高速主基板的异常。（检测出多 CPU 高速通信的异常。）	对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是智能功能模块、CPU 模块或者基板故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		QnU*10
1437	■附加信息 ・公共信息： － ・个别信息： － ■诊断时机 ・电源 ON 时 / 复位时	・采取防噪声措施。 ・确认 CPU 模块的主基板安装状态。 ・对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是 CPU 模块的硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		
1439	[MULTI-C.BUS ERR.] 检测出多 CPU 高速主基板的异常。（检测出多 CPU 高速通信的异常。） ■附加信息 ・公共信息： － ・个别信息： － ■诊断时机 ・电源 ON 时 / 复位时	对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是 CPU 模块或者基板故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		

*10 以除 Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU 以外的通用型 QCPU 为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
1500	[AC/DC DOWN] ・发生了供应电源瞬时掉电。 ・供应电源变为 OFF。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・常时	确认供应电源。	RUN：亮灯 ERR.：熄灯 CPU 状态： 继续运行	QCPU
1510	[SINGLE PS. DOWN] 冗余基板中某一侧的冗余电源模块的供应电压过低。 **■附加信息** ・公共信息： 基板号 / 电源号 ・个别信息： － **■诊断时机** ・常时	确认冗余基板上安装的冗余电源模块的供应电源。	RUN：亮灯 ERR.：亮灯 CPU 状态： 继续运行	Qn(H)*6 QnPH*6 QnPRH QnU*12
1520	[SINGLE PS. ERROR] 检测出冗余基板中某一侧的冗余电源模块的故障。 **■附加信息** ・公共信息： 基板号 / 电源号 ・个别信息： － **■诊断时机** ・常时	冗余电源模块的硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		Qn(H)*6 QnPH*6 QnPRH QnU*12
1600	[BATTERY ERROR*3] ・CPU 模块本体电池电压低于规定值。 ・CPU 模块本体电池的导线连接器未安装。 **■附加信息** ・公共信息： 驱动器名 ・个别信息： － **■诊断时机** ・常时	・对电池进行更换。 ・使用程序存储器、标准 RAM 或者停电保持功能时，安装导线连接器。	RUN： 亮灯 ERR.：熄灯 CPU 状态： 继续运行	QCPU
1601	[BATTERY ERROR*3] 存储卡的电池电压低于规定值。 **■附加信息** ・公共信息： 驱动器名 ・个别信息： － **■诊断时机** ・常时	对电池进行更换。		Qn(H) QnPH QnPRH QnU*14
1610	[FLASH ROM ERROR] 至快闪卡（标准 ROM 以及系统预留区）的写入次数超过了 10 万次。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・ROM 写入时	对 CPU 模块进行更换。	RUN： 亮灯 ERR.：亮灯 CPU 状态： 继续运行	QnU

*3: 发生 BATTERY ERROR 时，BAT.ALM LED 将亮灯 / 闪烁。
*6: 以序列号的前 5 位数为 “04101” 以后的模块为对象。
*12: 以序列号的前 5 位数为 “10042” 以后的模块为对象。
*14: 以除 Q00UJCPU、Q00UCPU、Q01UCPU 以外的通用型 QCPU 为对象。

12.1.4 出错代码一览表 (2000 ～ 2999)

以下介绍出错代码 2000 ～ 2999 的出错信息、异常内容及原因以及处理方法。

12.1 出错代码一览表

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
2000	[UNIT VERIFY ERR.] 在多 CPU 系统中，安装了不支持多 CPU 系统的 CPU 模块。 **■附加信息** ・公共信息：模块号（插槽号）[远程 I/O 网时] 网络号 / 站号 ・个别信息：－ **■诊断时机** ・执行 END 指令时	将不支持多 CPU 系统的 CPU 模块更换为支持多 CPU 系统的 CPU 模块。	RUN: 熄灯 / 亮灯 ERR.: 闪烁 / 亮灯 CPU 状态： 停止 / 继续运行*1	Qn(H)*3 QnPH
	[UNIT VERIFY ERR.] 与接通电源时的输入输出模块信息不相符。 ・在运行过程中对输入输出模块（包括智能功能模块）进行了拆装。 **■附加信息** ・公共信息：模块号（插槽号）[远程 I/O 网时] 网络号 / 站号 ・个别信息：－ **■诊断时机** ・执行 END 指令时	通过 GX Developer 读取出错的公共信息，对与该数值（模块号）对应的模块进行检查、更换。或者通过 GX Developer 监视特殊寄存器 SD150 ～ SD157，对该数据的位为 “1” 的模块进行检查、更换。		Q00J/Q00/Q01
	[UNIT VERIFY ERR.] 与接通电源时的输入输出模块信息不相符。 ・在运行过程中对输入输出模块（包括智能功能模块 / 特殊功能模块）进行了拆装。 **■附加信息** ・公共信息：模块号（插槽号）[远程 I/O 网时] 网络号 / 站号 ・个别信息：－ **■诊断时机** ・执行 END 指令时	・通过外围设备读取出错公共信息，对与该数值（模块号）对应的模块进行检查、更换。 ・通过外围设备监视特殊寄存器 SD1400 ～ SD1431，对该数据的位为 “1” 的模块进行检查、更换。 ・将 GOT 与主基板或者扩展基板通过总线相连接时，确认扩展电缆的连接状态及 GOT 的接地状态。		Qn(H) QnPH QnPRH QnU
2001	[UNIT VERIFY ERR.] 在运行过程中对被进行了 CPU 模块的空闲设置的插槽进行了模块安装。 **■附加信息** ・公共信息：模块号（站号） ・个别信息：－ **■诊断时机** ・执行 END 指令时	不要在运行过程中对被进行了 CPU 模块的空闲设置的插槽进行模块安装。	RUN: 熄灯 / 亮灯 ERR.: 闪烁 / 亮灯 CPU 状态： 停止 / 继续运行*2	Q00J/Q00/Q01*3 QnU

*1 发生出错时的 CPU 的动作状态可以在参数中进行设置。(LED 显示也将联动变化。)
*2 可以通过参数对模块进行出错停止 / 继续运行的选择。
*3 以功能版本 B 以后为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
2010	[BASE LAY ERROR] ・超过了扩展基板的允许使用级数。 ・通过总线连接的 GOT 时，在 GOT 的电源处于 OFF 的状态下对 CPU 模块进行了复位。 **■附加信息** ・公共信息： 基板号 ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时	・将扩展基板的级数设置在允许使用级数以内。 ・重新接通可编程控制器、GOT 的电源。		Q00J/Q00/Q01*3 QnPRH Q00UJ Q00U/Q01U Q02U
2011	[BASE LAY ERROR] 使用了 QA1S6 □ B、QA6 □ B、QA6ADP+A5 □ B/A6 □ B 作为基板。 **■附加信息** ・公共信息： 基板号 ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时	不要将 QA1S6 □ B、QA6 □ B、QA6ADP+A5 □ B/A6 □ B 作为基板使用。		Q00J/Q00/Q01*3 QnPH QnPRH QnU
2012	[BASE LAY ERROR] 在冗余系统的主基板上通过总线连接方式连接了 GOT。 在冗余系统中检测出以下异常。 ・扩展第 1 级中连接了除 Q6 □ WRB 以外的基板。 ・在扩展第 1 级中未安装 Q6 □ WRB 的情况下在扩展第 2 ～ 7 级的某一级中连接了基板。 ・其它系统 CPU 模块不支持扩展基板。 ・连接了 Q5 □ B、QA1S6 □ B、QA6 □ B、QA6ADP+A5 □ B/A6 □ B。 ・两个系统的主基板的插槽数不相同。 ・不能正确地读取 Q6 □ WRB 的信息。 **■附加信息** ・公共信息： 基板号 ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时	・将主基板上连接的 GOT 连接用总线连接电缆卸下。 ・使用 Q6 □ WRB(固定为扩展第 1 级)。 ・将其它系统也安装为支持扩展基板的冗余 CPU 模块。 ・不要将 QA1S6 □ B、QA6 □ B、QA6ADP+A5 □ B/A6 □ B 作为基板使用。 ・使用插槽数相同的主基板。 ・Q6 □ WRB 的硬件故障。(请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。)	RUN：熄灯 ERR.：闪烁 CPU 状态： 停止	
2013	[BASE LAY ERROR] 在冗余系统中，Q6 □ WRB 的级数被识别为除扩展第 1 级以外。 **■附加信息** ・公共信息： 基板号 ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时	Q6 □ WRB 的硬件故障。(请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。)		QnPRH*6
2020	[EXT.CABLE ERR.] 在冗余系统中检测出以下异常。 ・电源 ON/ 复位时，在待机系统中检测出控制系统与 Q6 □ WRB 之间的路径异常。 ・END 处理时，检测出待机系统与 Q6 □ WRB 之间的路径异常。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时 / 执行 END 指令时	确认主基板与 Q6 □ WRB 之间的扩展电缆是否正确连接。未正确连接的情况下，将连接扩展电缆的主基板的电源置为 OFF 后重新进行连接。 如果连接正确，可能是 CPU 模块、Q6 □ WRB 或者扩展电缆的硬件故障。(请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。)		

*3 以功能版本 B 以后为对象。
*6 以序列号的前 5 位数为 “09012” 以后的模块为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
2100	[SP. UNIT LAY ERR.] 在可编程控制器参数的 I/O 分配设置中，将安装了 QI60 的插槽设置为除智能（智能功能模块）或者中断（中断模块）以外。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时	重新设置，使可编程控制器参数的 I/O 分配设置与实际安装状态相符合。	RUN: 熄灯 ERR.: 闪烁 CPU 状态：停止	Qn(H)*3 QnPH QnPRH
	[SP. UNIT LAY ERR.] ・在可编程控制器参数的 I/O 分配设置中，将输入输出模块的位置分配为智能功能模块，或者将智能功能模块的位置分配为输入输出模块。 ・在可编程控制器参数的 I/O 分配设置中，将 CPU 模块的位置分配为其它模块或者设置为空闲。或者将其它模块或设置为空闲的位置分配为 CPU 模块。 ・在可编程控制器参数的 I/O 分配设置中，对无开关设置的模块进行了开关设置。 ・在可编程控制器参数的 I/O 分配设置中，所设置的智能功能模块的分配点数少于安装模块的点数。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时	・重新设置，使可编程控制器参数的 I/O 分配设置与智能功能模块、CPU 模块的实际安装状态相符合。 ・将可编程控制器参数的 I/O 分配设置的开关设置删除。		Qn(H) QnPH QnPRH QnU
	[SP. UNIT LAY ERR.] ・在可编程控制器参数的 I/O 分配设置中，将输入输出模块的位置分配为智能功能模块，或者将智能功能模块的位置分配为输入输出模块。 ・在可编程控制器参数的 I/O 分配设置中，将 CPU 模块的位置分配为其它模块或者设置为空闲。或者将其它模块或设置为空闲的位置分配为 CPU 模块。 ・在可编程控制器参数的 I/O 分配设置中，所设置的智能功能模块的分配点数少于安装模块的点数。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时	重新设置，使可编程控制器参数的 I/O 分配设置与智能功能模块、CPU 模块的实际安装状态相符合。		Q00J/Q00/Q01

*3 以功能版本 B 以后为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
2101	[SP. UNIT LAY ERR.] 安装了 13 个以上可以对 CPU 模块进行中断启动的 A 系列特殊功能模块 (A1SI61 除外)。 **■附加信息** ・公共信息： 模块号 (插槽号) ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时	应安装 12 个以下可以对 CPU 模块进行中断启动的 A 系列特殊功能模块 (A1SI61 除外)。	RUN: 熄灯 ERR.: 闪烁 CPU 状态： 停止	Qn(H)
2102	[SP. UNIT LAY ERR.] 安装了 7 个以上的 A1SD51S。 **■附加信息** ・公共信息： 模块号 (插槽号) ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时	安装了 6 个以下的 A1SD51S。		
2103	[SP. UNIT LAY ERR.] ・在单 CPU 系统中安装了 2 个以上的 QI60/A1SI61。 ・在多 CPU 系统中将 2 个以上的 QI60/A1SI61 设置到同一个管理 CPU 中。 ・在多 CPU 系统中安装了 2 个以上的 A1SI61。 **■附加信息** ・公共信息： 模块号 (插槽号) ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时	・在单 CPU 系统中应安装 1 个 QI60/A1SI61。 ・在多 CPU 系统中应将同一个管理 CPU 下的 QI60/A1SI61 变更为 1 个。 ・在多 CPU 系统中应只安装 1 个 A1SI61。 在多 CPU 系统中，在各个 QCPU 中分别使用了中断模块的情况下，应变更为 QI60(A1SI61: 1 个 + QI60: 最多 3 个)，或者只使用 QI60。		Qn(H)*3 QnPH
	[SP. UNIT LAY ERR.] 安装了 2 个以上的 QI60、A1SI61。 **■附加信息** ・公共信息： 模块号 (插槽号) ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时	应安装 1 个 QI60、A1SI61。		Qn(H) QnPRH
	[SP. UNIT LAY ERR.] 安装了 2 个以上的 QI60。 **■附加信息** ・公共信息： 模块号 (插槽号) ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时	应安装 1 个 QI60。		Q00J/Q00/Q01*5
	[SP. UNIT LAY ERR.] 安装了 2 个以上的未进行中断指针设置的 QI60。 **■附加信息** ・公共信息： 模块号 (插槽号) ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时	・安装 1 个 QI60。 ・在第 2 个及以后的 QI60 中设置中断指针。		Q00J/Q00/Q01*3 QnU

*3 以功能版本 B 以后为对象。
*5 以序列号的前 5 位数为 “04101” 以后的模块为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
2106	[SP. UNIT LAY ERR.] ・安装了 2 个以上的 MELSECNET/H 模块。 ・安装了 2 个以上的 CC-Link IE 控制网络模块。 ・安装了 2 个以上的以太网模块。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时	・只安装 1 个 MELSECNET/H 模块。 ・只安装 1 个 CC-Link IE 控制网络模块。 ・只安装 1 个的以太网模块。	RUN：熄灯 ERR.：闪烁 CPU 状态： 停止	Q00UJ
	[SP. UNIT LAY ERR.] ・在整个系统中，MELSECNET/H 模块及 CC-Link IE 控制网络模块的合计安装个数达到或超过了 5 个。 ・在整个系统中，安装了 2 个或以上的 MELSECNET/H 模块。 ・在整个系统中，安装了 2 个或以上的 CC-Link IE 控制网络模块。 ・在整个系统中，安装了 2 个或以上的以太网模块。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时	・在整个系统中，MELSECNET/H 模块及 CC-Link IE 控制网络模块的合计安装个数应为 4 个或以内。 ・在整个系统中，只应安装 1 个 MELSECNET/H 模块。 ・在整个系统中，只应安装 1 个 CC-Link IE 控制网络模块。 ・在整个系统中，只应安装 1 个以太网模块。		Q00U/Q01U
	[SP. UNIT LAY ERR.] ・在整个系统中，MELSECNET/H 模块及 CC-Link IE 控制网络模块的合计安装个数达到或超过了 3 个。 ・在整个系统中，安装了 3 个或以上的以太网模块。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时	・在整个系统中，MELSECNET/H 模块及 CC-Link IE 控制网络模块的合计安装个数应为 2 个或以内。 ・在整个系统中，只应安装 2 个或以内的以太网模块。		Q02U
	[SP. UNIT LAY ERR.] ・在整个系统中，MELSECNET/H 模块及 CC-Link IE 控制网络模块的合计安装个数达到或超过了 5 个。 ・在整个系统中，安装了 5 个或以上的以太网模块。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时	・在整个系统中，MELSECNET/H 模块及 CC-Link IE 控制网络模块的合计安装个数应为 4 个或以内。 ・在整个系统中，只应安装 4 个或以内的以太网模块。		QnU*7

*7 以除 Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU 以外的通用型 QCPU 为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
2106	[SP. UNIT LAY ERR.] ・在整个系统中，安装了 3 个或以上的 CC-Link IE 控制网络模块。 ・在整个系统中，MELSECNET/H 模块及 CC-Link IE 控制网络模块的合计安装个数达到或超过了 5 个。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时	・在整个系统中，只应安装 2 个或以内的 CC-Link IE 控制网络模块。 ・在整个系统中，MELSECNET/H 模块及 CC-Link IE 控制网络模块的合计安装个数应为 4 个或以内。	RUN：熄灯 ERR.：闪烁 CPU 状态：停止	Qn(H)*6 QnPH*9 QnPRH*9
	[SP. UNIT LAY ERR.] ・在整个系统中，安装了 5 个或以上的 MELSECNET/H 模块。 ・在整个系统中，安装了 5 个或以上的以太网模块。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时	・在整个系统中，只应安装 4 个或以内的 MELSECNET/H 模块。 ・在整个系统中，只应安装 4 个或以内的以太网模块。		Qn(H) QnPH QnPRH
	[SP. UNIT LAY ERR.] ・在整个系统中，安装了 2 个或以上的 MELSECNET/H 模块。 ・在整个系统中，安装了 2 个或以上的以太网模块。 ・在整个系统中，安装了 3 个或以上的 CC-Link 模块。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时	・在整个系统中，只应安装 1 个 MELSECNET/H 模块。 ・在整个系统中，只应安装 1 个以太网模块。 ・在整个系统中，2 个或以内的 CC-Link 模块。		Q00J/Q00/Q01
	[SP. UNIT LAY ERR.] ・在 MELSECNET/H 网络系统中存在有相同的网络号、相同的站号。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时	・对网络号、相同的站号进行确认。		Q00J/Q00/Q01 Qn(H) QnPH QnPRH
2107	[SP. UNIT LAY ERR.] 在可编程控制器参数的 I/O 分配设置中设置的起始 X/Y 与其它模块的的起始 X/Y 重复。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时	重新设置，使可编程控制器参数的 I/O 分配设置与智能功能模块 / 特殊功能模块的实际安装状态相符合。		QCPU

*6 以序列号的前 5 位数为 “09012” 以后的模块为对象。
*9 以序列号的前 5 位数为 “10042” 以后的模块为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
2108	[SP. UNIT LAY ERR.] ・安装了 A2USCPU 用网络模块 A1SJ71LP21、A1SJ71BR11、A1SJ71AP21、A1SJ71AR21、A1SJ71AT21B。 ・安装了 Q2AS 用网络模块 A1SJ71QLP21、A1SJ71QBR11。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时	将 A2USCPU、Q2ASCPU 用网络模块变更为 MELSECNET/H 模块。	RUN：熄灯 ERR.：闪烁 CPU 状态：停止	Qn(H)
2110	[SP. UNIT ERROR] ・通过 FROM/TO 指令指定的位置不是智能功能模块 / 特殊功能模块。 ・通过 FROM/TO 指令指定的模块是不具有缓冲存储器的模块。 ・访问目标智能功能模块 / 特殊功能模块故障。 ・在以 CPU 共享存储器为对象的指令中指定了未安装的站号。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： 程序出错位置 **■诊断时机** ・执行指令时	・通过 GX Developer 读取出错的个别信息，对与该数值（程序出错位置）相对应的 FROM/TO 指令进行检查、修改。 ・访问目标智能功能模块 / 特殊功能模块的硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）	RUN： 熄灯 / 亮灯 ERR.： 闪烁 / 亮灯 CPU 状态： 停止 / 继续运行 *1	Q00J/Q00/Q01 Qn(H)*3 QnPH QnPRH QnU
2111	[SP. UNIT ERROR] ・直接链接软元件（J □ \ □）中指定的位置不是网络模块。 ・在运行过程中对输入输出模块（包括智能功能模块 / 特殊功能模块）进行了拆装。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： 程序出错位置 **■诊断时机** ・执行指令时			QCPU
2112	[SP. UNIT ERROR] ・智能功能模块 / 特殊功能模块专用指令中指定的位置不是智能功能模块 / 特殊功能模块。或者不是相应的智能功能模块 / 特殊功能模块。 ・网络专用指令中指定的网络号不存在。或者中继目标网络不存在。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： 程序出错位置 **■诊断时机** ・执行指令时 /STOP → RUN 时	通过外围设置读取出错的个别信息，对与该数值（程序出错位置）对应的智能功能模块 / 特殊功能模块专用指令（网络用指令）进行检查、修改。		

*1 发生出错时的 CPU 的动作状态可以在参数中进行设置。(LED 显示也将联动变化。)
*3 以功能版本 B 以后为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
2113	[SP. UNIT ERROR] 网络专用指令中指定的位置不是网络模块。 **■附加信息** ・公共信息： $FFFF_H$（固定） ・个别信息： 程序出错位置 **■诊断时机** ・执行指令时 /STOP → RUN 时	通过外围设置读取出错的个别信息，对与该数值（程序出错位置）对应的智能功能模块 / 特殊功能模块专用指令（网络用指令）进行检查、修改。	RUN: 熄灯 / 亮灯 ERR.: 闪烁 / 亮灯 CPU 状态： 停止 / 继续运行 *1	Qn(H) QnPH
2114	[SP. UNIT ERROR] 在指定其它站执行的指令（不能用于自站的指令）中指定了自站。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： 程序出错位置 **■诊断时机** ・执行指令时 /STOP → RUN 时	通过 GX Developer 读取出错的个别信息，对与该数值（程序出错位置）相对应的 FROM/TO 指令进行检查、修改。	RUN: 熄灯 / 亮灯 ERR.: 闪烁 / 亮灯 CPU 状态： 停止 / 继续运行	Q00J/Q00/ Q01*3 Qn(H)*3 QnPH QnU
2115	[SP. UNIT ERROR] 在指定自站执行的指令（不能用于其它站的指令）中指定了其它站的 CPU 模块。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： 程序出错位置 **■诊断时机** ・执行指令时 /STOP → RUN 时			Q00J/Q00/ Q01*3 Qn(H)*3 QnPH
2116	[SP. UNIT ERROR] ・将不能用于其它站管理的模块的指令用于其它站管理的模块。 ・对其它站管理的 A、QnA 用模块执行了指令。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： 程序出错位置 **■诊断时机** ・执行指令时 /STOP → RUN 时			Q00J/Q00/ Q01*3 Qn(H)*3 QnPH QnU
2117	[SP. UNIT ERROR] 在多 CPU 系统专用指令中对不能指定的 CPU 模块进行了指定。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： 程序出错位置 **■诊断时机** ・执行指令时 /STOP → RUN 时			
2118	[SP. UNIT ERROR] 多 CPU 系统时，在可编程控制器参数中将在线模块更换设置设置为允许时，在 FROM 指令 / 智能功能模块软元件（U □ \G □）中指定了其它站管理的智能功能模块。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： 程序出错位置 **■诊断时机** ・执行指令时	・将程序修改为，在多 CPU 系统中进行在线模块更换时，不对其它站管理的智能功能模块进行访问。 ・在多 CPU 系统中访问其它 CPU 管理的智能功能模块时，在参数中将在线模块更换设置设置为禁止。		Qn(H)*3 QnPH QnU*7

*1 发生出错时的 CPU 的动作状态可以在参数中进行设置。(LED 显示也将联动变化。)
*3 以功能版本 B 以后为对象。
*7 以除 Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU 以外的通用型 QCPU 为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
2120	[SP. UNIT LAY ERR.] Q5 □ B、Q6 □ B 及 QA1S6 □ B、QA6 □ B、QA6ADP+A5 □ B/A6 □ B 的配置不正确。 ■附加信息 • 公共信息： － • 个别信息： － ■诊断时机 • 电源 ON 时 / 复位时	重新审核基板的配置。	RUN：熄灯 ERR.：闪烁 CPU 状态： 停止	Q00J/Q00/Q01*4 Qn(H) QnPH
2121	[SP. UNIT LAY ERR.] CPU 模块被安装到 CPU 插槽以外或者 0 ～ 2 插槽以外。 ■附加信息 • 公共信息： － • 个别信息： － ■诊断时机 • 电源 ON 时 / 复位时	确认 CPU 模块的安装位置，安装到正确的插槽中。		Qn(H) QnPH
2122	[SP. UNIT LAY ERR.] 主基板使用了 QA1S6 □ B、QA6 □ B、QA6ADP+A5 □ B/A6 □ B。 ■附加信息 • 公共信息： － • 个别信息： － ■诊断时机 • 电源 ON 时 / 复位时	使用允许使用的基板产品作为主基板。		Qn(H) QnPH QnPRH
2124	[SP. UNIT LAY ERR.] • 在 65 插槽后面安装了模块。 • 在基板分配中设置的插槽数的后面安装了模块。 • 在 I/O 点数 4096 点后面安装了模块。 • 所安装的模块跨越了 I/O 点数 4096 点的界限。 ■附加信息 • 公共信息： － • 个别信息： － ■诊断时机 • 电源 ON 时 / 复位时	• 将 65 插槽后面的模块卸下。 • 将基板分配中设置的插槽数后面安装的模块卸下。 • 将 4096 点后面安装的模块卸下。 • 应将最终模块更换为其占用点数未跨越 4096 点的模块。		Qn(H) QnPH QnPRH QnU*7
	[SP. UNIT LAY ERR.] • 在 25 插槽后面安装了模块。（Q00UJ 时为 17 插槽后面） • 在基板分配中设置的插槽数的后面安装了模块。 • 在 I/O 点数 1024 点后面安装了模块。（Q00UJ 时为 256 点后面） • 所安装的模块跨越了 I/O 点数 1024 点的界限。（Q00UJ 时为 256 点的界限） ■附加信息 • 公共信息： － • 个别信息： － ■诊断时机 • 电源 ON 时 / 复位时	• 将 25 插槽后面的模块卸下。（Q00UJ 时为 17 插槽后面） • 将基板分配中设置的插槽数后面安装的模块卸下。 • 将 1024 点后面安装的模块卸下。（Q00UJ 时为 256 点后面） • 应将最终模块更换为其占用点数未跨越 1024 点的模块。（Q00UJ 时为不跨越 256 点。）		Q00UJ Q00U/Q01U

*4　以功能版本 A 为对象。
*7　以除 Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU 以外的通用型 QCPU 为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
2124	[SP. UNIT LAY ERR.] • 在 37 插槽后面安装了模块。 •在基板分配中设置的插槽数的后面安装了模块。 • 在 I/O 点数 2048 点后面安装了模块。 • 所安装的模块跨越了 I/O 点数 2048 点的界限。 **■附加信息** • 公共信息： － • 个别信息： － **■诊断时机** • 电源 ON 时 / 复位时	• 将 37 插槽后面的模块卸下。 • 将基板分配中设置的插槽数后面安装的模块卸下。 • 将 2048 点后面安装的模块卸下。 • 应将最终模块更换为其占用点数未跨越 2048 点的模块。	RUN：熄灯 ERR.：闪烁 CPU 状态：停止	Q02U
	[SP. UNIT LAY ERR.] • 在 25 插槽后面安装了模块。(Q00J 时为 17 插槽后面） •在基板分配中设置的插槽数的后面安装了模块。 • 在 I/O 点数 1024 点后面安装了模块。(Q00J 时为 256 点后面） • 所安装的模块跨越了 I/O 点数 1024 点的界限。(Q00J 时为 256 点的界限） **■附加信息** • 公共信息： － • 个别信息： － **■诊断时机** • 电源 ON 时 / 复位时	• 将 25 插槽后面的模块卸下。(Q00J 时为 17 插槽后面） • 将基板分配中设置的插槽数后面安装的模块卸下。 • 将 1024 点后面安装的模块卸下。(Q00J 时为 256 点后面） • 应将最终模块更换为其占用点数未跨越 1024 点的模块。(Q00J 时为不跨越 256 点。）		Q00J/Q00/Q01
2125	[SP. UNIT LAY ERR.] • 安装了不能识别的模块。 • 不能从智能功能模块 / 特殊功能模块返回信息。 **■附加信息** • 公共信息： 模块号（插槽号） • 个别信息： － **■诊断时机** • 电源 ON 时 / 复位时	• 安装可以使用的模块。 • 智能功能模块 / 特殊功能模块的硬件故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		QCPU
2126	[SP. UNIT LAY ERR.] 多 CPU 系统中的 CPU 模块配置处于以下状态。 • CPU 模块的左侧有空闲插槽。 • 高性能型 QCPU/ 过程 CPU 的左侧安装了除高性能型 QCPU/ 过程 CPU 以外的模块（包括运动控制 CPU）。 **■附加信息** • 公共信息： 模块号（插槽号） • 个别信息： － **■诊断时机** • 电源 ON 时 / 复位时	• 将模块依次左移填满空闲插槽。（将空闲插槽设置到 CPU 模块的右侧。） • 将安装在高性能型 QCPU/ 过程 CPU 左侧的模块卸下，将高性能型 QCPU/ 过程 CPU 依次左移填满空出的插槽。 将运动控制 CPU 汇总安装到高性能型 QCPU/ 过程 CPU 的右侧。		Qn(H)*3 QnPH

*3　以功能版本 B 以后为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
2128	[SP.UNIT LAY ERR.] 在冗余系统中，在扩展基板上安装了不能使用的模块。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时	・将不能使用的模块从扩展基板上卸下。		QnPRH*6
2150	[SP. UNIT VER. ERR.] 在多 CPU 系统中，将不支持多 CPU 系统的智能功能模块的管理 CPU 设置为除 1 号机以外的 CPU。 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时 / 可编程控制器写入时	・变更为支持多 CPU 系统的智能功能模块（功能版本 B 以后）。 ・将不支持多 CPU 系统的智能功能模块的管理 CPU 变更为 1 号机。		Q00J/Q00/Q01 QnPH QnU*10
2151	[SP. UNIT VER. ERR.] 在冗余系统中安装了不支持冗余系统的下述模块。 ・MELSECNET/H 模块 ・以太网模块 ・CC-Link IE 控制网络模块 **■附加信息** ・公共信息： 模块号（插槽号） ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时 / 可编程控制器写入时	使用支持冗余系统的下述模块。 ・MELSECNET/H 模块 ・以太网模块 ・CC-Link IE 控制网络模块		QnPRH
2200	[MISSING PARA.] 通过插杆开关的参数有效驱动器指定的驱动器中没有参数文件。 **■附加信息** ・公共信息： 驱动器名 ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	・对插杆开关的参数有效驱动器的设置进行检查、修改。 ・在通过插杆开关的参数有效驱动器指定的驱动器中设置参数文件。	RUN：熄灯 ERR.：闪烁 CPU 状态： 停止	Qn(H) QnPH QnPRH
	[MISSING PARA.] 程序存储器中没有参数文件。 **■附加信息** ・公共信息： 驱动器名 ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	在程序存储器中设置参数文件。		Q00J/Q00/Q01
	[MISSING PARA.] 在参数有效的全部驱动器中没有参数文件。 **■附加信息** ・公共信息： 驱动器名 ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	在有效的驱动器中设置参数文件。		QnU
2210	[BOOT ERROR] ・引导文件的内容不正确。 **■附加信息** ・公共信息： 驱动器名 ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时	重新审核引导设置。		Q00J/Q00/Q01*3 Qn(H) QnPH QnPRH QnU

*3 以功能版本 B 以后为对象。
*6: 以序列号的前 5 位数为 “09012” 以后的模块为对象。
*10: 以除 Q00UJCPU 以外的通用型 QCPU 为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
2211	[BOOT ERROR] 引导时的格式化处理失败。 **■附加信息** ・公共信息： 驱动器名 ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时	・再次进行引导。 ・CPU 模块的硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		Qn(H) QnPRH QnU
2220	[RESTORE ERROR] ・通过软元件数据备份功能备份的软元件信息（点数）与可编程控制器参数的软元件点数不相同。 发生本出错后，使备份时的软元件点数与可编程控制器参数的软元件点数一致，或者在删除备份数据之前通过电源 ON/ 复位进行恢复。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时	・将可编程控制器参数的软元件点数设置与备份时的软元件点数设置为一致后，进行电源 OFF → ON 或者复位→复位解除。 ・删除备份数据后，进行电源 OFF → ON 或者 CPU 复位→复位解除。		
2221	[RESTORE ERROR] ・通过软元件数据备份功能备份的软元件信息处于不完整状态。（有可能在进行备份的过程中进行了电源 OFF 或者复位。） 发生本出错时，进行数据恢复。或者发生本出错时将处于不完整状态的软元件信息删除。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时	对 CPU 模块进行复位后再次置为 RUN。	RUN：熄灯 ERR.：闪烁 CPU 状态： 停止	
2225	[RESTORE ERROR] 恢复目标 CPU 模块与备份源 CPU 模块的型号不相同。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时	以与备份源 CPU 模块相同型号的 CPU 模块进行恢复。		QnU
2226	[RESTORE ERROR] ・备份数据文件的校验不一致。 ・从存储卡的备份数据读取未能正常完成。 ・SRAM 卡的写保护开关处于有效（禁止写入）状态，“仅初次恢复”设置无法生效。 **■附加信息** ・公共信息： － ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时	・备份数据可能已被损坏，使用其它的备份数据进行还原。 ・将SRAM卡的写保护开关设置为无效(允许写入)。		
2227	[RESTORE ERROR] 恢复目标驱动器的备份数据写入未能正常完成。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时	CPU 模块有可能故障，对其它的 CPU 模块再次执行恢复。		

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
2300	[ICM. OPE. ERROR] ・在未将存储卡插拔开关置为 OFF 的状况下拔下了存储卡。 ・在未安装存储卡的状况下将存储卡插拔开关置为 ON。 **■附加信息** ・公共信息： 驱动器名 ・个别信息： － **■诊断时机** ・拆装存储卡时	・应将存储卡插拔开关置为 OFF 之后，再拔下存储卡。 ・安装了存储卡之后将插拔开关置为 ON。	RUN: 熄灯 / 亮灯 ERR.: 闪烁 / 亮灯 CPU 状态： 停止 / 继续运行 *1	Qn(H) QnPH QnPRH QnU*11
2301	[ICM. OPE. ERROR] ・存储卡未进行格式化。 ・存储卡的格式化状态不正确。 **■附加信息** ・公共信息： 驱动器名 ・个别信息： － **■诊断时机**	・对存储卡进行格式化。 ・对存储卡重新进行格式化。 在 Flash 卡的情况下，以下述某种方式将数据写入到 Flash 卡中。 1) 程序存储器的 ROM 化 2) 可编程控制器写入（快闪 ROM） 3) 备份到存储卡中 4) 通过外围设备（存储卡写卡器等）写入图像数据		
	[ICM. OPE. ERROR] ・在 Flash 卡没有 QCPU 的文件。 **■附加信息** ・公共信息： 驱动器名 ・个别信息： －	・将 QCPU 的文件写入到 Flash 卡中。		
	[ICM. OPE. ERROR] ・检测出 SRAM 卡的异常。 （设置为不进行自动格式化时发生） ・对文件寄存器设置中的快闪卡进行了写入。 **■附加信息** ・公共信息： 驱动器名 ・个别信息： － **■诊断时机** ・拆装存储卡时 / 写入存储卡时	・更换 SRAM 卡的电池后，对 SRAM 卡进行格式化。 ・在参数中将文件寄存器设置为 “不使用” 后写入到 CPU 中，然后执行操作。		QnU*11
2302	[ICM. OPE. ERROR] 安装了 CPU 模块中不能使用的存储卡。 **■附加信息** ・公共信息： 驱动器名 ・个别信息： － **■诊断时机** ・拆装存储卡时	・对存储卡进行格式化。 ・对存储卡重新进行格式化。 ・对存储卡进行检查。		Qn(H) QnPH QnPRH QnU*11

*1 发生出错时的 CPU 的动作状态可以在参数中进行设置。(LED 显示也将联动变化。)
*11 以除 Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU 以外的通用型 QCPU 为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
2400	[FILE SET ERROR] 在不支持标准 ROM 自动写入的 CPU 模块中，执行了至标准 ROM 的自动写入。 （安装了选定为通过引导文件进行至标准 ROM 的自动写入的存储卡后，将拨杆开关的参数有效驱动器设置为存储卡。） **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时 / 可编程控制器写入时	・在支持至标准 ROM 的自动写入的 CPU 模块中，执行至标准 ROM 的自动写入。 ・使用 GX Developer，对标准 ROM 进行参数、程序写入。 ・将存储卡更换为未进行至标准 ROM 的自动写入设置的存储卡后，进行从存储卡的引导运行。	RUN：熄灯 ERR.：闪烁 CPU 状态： 停止	Qn(H)*3 QnPH QnPRH
	[FILE SET ERROR] 参数中指定的文件不存在。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时 / 可编程控制器写入时	・通过外围设置读取出错的个别信息，对与该数值（参数号）对应的参数项目的驱动器名、文件名进行检查、修改。 ・创建指定的文件后，写入到 CPU 模块中。		QCPU
2401	[FILE SET ERROR] 由于引导操作以及至标准 ROM 的自动写入操作，超出了程序存储器的容量。 **■附加信息** ・公共信息： 模块号 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 / 可编程控制器写入时	・对参数（引导设置）进行检查、修改。 ・将程序存储器内不需要的文件删除。 ・在参数中选择引导时 “清除程序存储器”，在清除程序存储器之后进行引导。		Qn(H)*3 QnPH QnPRH
	[FILE SET ERROR] 由于引导操作，超出了程序存储器的容量。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 / 可编程控制器写入时			QnU
	[FILE SET ERROR] 无法创建参数中指定的文件。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 / 可编程控制器写入时	・通过外围设置读取出错的个别信息，对与该数值（参数号）对应的参数项目的驱动器名、文件名进行检查、修改。 ・对存储卡的存储器容量的剩余量进行检查。		QCPU
	[FILE SET ERROR] ・在可编程控制器文件设置中进行了使用软元件数据存储用文件的设置，但标准 ROM 中没有创建软元件数据存储用文件所必需可用空间。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 / 可编程控制器写入时	在标准 ROM 中预留出可用空间。		QnU

*3 以功能版本 B 以后为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
2410	[FILE OPE. ERROR] ・指定的程序在程序存储器中不存在。 ・在执行 ECALL、EFCALL、PSTOP、PSCAN、POFF、PLOW 指令时有可能发生。 ・指定的文件不存在。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 程序出错位置 **■诊断时机** ・执行指令时	・通过外围设置读取出错的个别信息，对与该数值（程序出错位置）对应的程序进行检查、修改。创建指定的文件后，写入到 CPU 模块中。 ・指定的文件不存在时，将文件写入到对象存储器中，或者重新审核通过指令进行的文件指定。	RUN: 熄灯 / 亮灯 ERR.: 闪烁 / 亮灯 CPU 状态： 停止 / 继续运行 *1	Qn(H) QnPH QnPRH QnU
2411	[FILE OPE. ERROR] ・是顺控程序中不能指定的文件（注释文件等）。 ・虽然指定的程序存在于程序存储器内，但未登录到参数的程序设置中。在执行 ECALL、EFCALL、PSTOP、PSCAN、POFF、PLOW 指令时有可能发生此出错。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 程序出错位置 **■诊断时机** ・执行指令时	通过外围设置读取出错的个别信息，对与该数值（程序出错位置）对应的程序进行检查、修改。		
2412	[FILE OPE. ERROR] 是顺控程序中不能指定的 SFC 程序文件。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 程序出错位置 **■诊断时机** ・执行指令时	通过外围设置读取出错的个别信息，对与该数值（程序出错位置）对应的程序进行检查、修改。		Qn(H) QnPH QnPRH QnU
2413	[FILE OPE. ERROR] 无法对顺控程序中指定的文件进行数据写入。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 程序出错位置 **■诊断时机** ・执行指令时	通过外围设置读取出错的个别信息，对与该数值（程序出错位置）对应的程序进行检查、修改。 确认指定的文件是否处于禁止写入状态。		Qn(H) QnPH QnPRH
2500	[CAN'T EXE. PRG.] ・存在有使用了超出可编程控制器参数的软元件设置中设置的软元件分配范围的软元件的程序文件。 ・对可编程控制器参数的软元件设置进行变更后，仅对参数进行了可编程写入。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	・通过外围设备读取出错的公共信息，对与该数值（文件名）对应的程序文件的软元件分配及可编程控制器参数的软元件设置的软元件分配进行检查、修改。 ・对可编程控制器参数的软元件设置进行了变更时，对参数及程序文件批量地进行可编程控制器写入。	RUN：熄灯 ERR.：闪烁 CPU 状态： 停止	QCPU
	[CAN'T EXE. PRG.] ・对可编程控制器参数的变址修饰设置进行了变更后，仅对参数进行了可编程控制器写入。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	・对可编程控制器参数的变址修饰设置进行了变更时，对参数及程序文件批量地进行可编程控制器写入。		QnU

*1 发生出错时的 CPU 的动作状态可以在参数中进行设置。(LED 显示也将联动变化。)

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
2501	[CAN'T EXE. PRG.] 虽然可编程控制器参数的程序设置被设置为“无”，但却存在有多个程序文件。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	将可编程控制器参数的程序设置修改为“有”。 或者将不需要的程序删除。		Qn(H) QnPH QnPRH QnU
	[CAN'T EXE. PRG.] ・程序文件有 3 个以上。 ・程序名与程序内容不相符。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	・将不需要的程序删除。 ・使程序名与程序内容相符。		Q00J/Q00/Q01
2502	[CAN'T EXE. PRG.] 程序文件不正确。 或者文件内容不是顺控程序。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	检查程序文件的类型是否为 ***.QPG 后，检查文件内容是否为顺控程序。		QCPU
	[CAN'T EXE. PRG.] 所创建的程序文件不是用于冗余 CPU 的程序文件。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	通过 GX Developer 或者 PX Developer 创建可编程控制器类型被设置为用于冗余 CPU(Q12PRH/Q25PRH)的程序后，将其进行可编程控制器写入到 CPU 模块中。	RUN：熄灯 ERR.：闪烁 CPU 状态：停止	QnPRH
2503	[CAN'T EXE. PRG.] 程序文件 1 个也没有。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	确认程序构成。 确认参数及程序构成。		QCPU
2504	[CAN'T EXE. PRG.] 执行了 2 个以上的 SFC 程序的普通程序以及管理程序。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时			Qn(H) QnPH QnPRH QnU
	[CAN'T EXE. PRG.] 存在有 2 个以上的 SFC 程序。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： － **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	将 SFC 程序设置为 1 个。		Q00J/Q00/Q01*3

*3 以功能版本 B 以后为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
2700	[REMOTE PASS.FAIL] 远程口令不一致的次数达到了上限。 ■附加信息 ・公共信息： － ・个别信息： － ■诊断时机 ・常时	确认是否为非法访问。如果是非法访问，进行相应连接的通信禁止等的处理。如果不是非法访问，则清除出错后执行下述项目。 （通过出错清除远程口令累计次数也将被清除。） ・确认发送的远程口令是否正确。 ・确认是否进行了远程口令的锁定处理。 ・确认是否从多个设备通过 UDP 对 1 个连接同时进行了访问。 ・确认远程口令的不一致上限值是否过小。	RUN：亮灯 ERR.：亮灯 CPU 状态： 继续运行	QnU*8
2710	[SNTP OPE.ERROR] 可编程控制器电源 ON/ 复位时的时间设置失败。 ■附加信息 ・公共信息： － ・个别信息： － ■诊断时机 ・执行时间设置功能时	・确认时间设置功能的设置是否正确。 ・确认指定的 SNTP 服务器是否正常动作，至指定的 SNTP 服务器用个人计算机的网络是否发生了故障。	RUN： 熄灯 / 亮灯 ERR.： 闪烁 / 亮灯 CPU 状态： 停止 / 继续运行*1	

*1　发生出错时的 CPU 的动作状态可以在参数中进行设置。（LED 显示也将联动变化。）
*8　以以太网板内置的 QCPU 为对象。

12.1.5 出错代码一览表 (3000 ～ 3999)

以下介绍出错代码 3000 ～ 3999 的出错信息、异常内容及原因以及处理方法。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
3000	[PARAMETER ERROR] 在多 CPU 系统中，在可编程控制器参数的中断指针设置中指定了其它站管理的智能功能模块。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时	・指定自站管理的智能功能模块的起始 I/O 编号。 ・删除可编程控制器参数的中断指针设置。	RUN：熄灯 ERR.：闪烁 CPU 状态： 停止	Qn(H)*1 QnPH QnU*10
	[PARAMETER ERROR] 可编程控制器参数的定时器时限设置、RUN-PAUSE 触点、公共指针号、一般数据处理、空闲插槽点数、系统中断设置、波特率设置、服务处理设置的各个设置超出了 CPU 的允许使用范围。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时	・通过外围设备读取出错的个别信息，对与该数值（参数号）对应的参数项目进行检查、修改。 ・将修改后的参数重新写入到 CPU 模块中，进行可编程控制器的电源再启动或者 CPU 模块的复位。 ・仍然发生相同的出错时，可能是硬件故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		QCPU
	[PARAMETER ERROR] 在程序存储器检查中设置的检查容量设置超出了 CPU 模块的允许使用范围。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时			QnPH QnPRH*5
	[PARAMETER ERROR] 出错个别信息（特殊寄存器 SD16）显示的参数内容不正确。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时			QCPU
	[PARAMETER ERROR] 在可编程控制器文件设置中，将文件寄存器的指定驱动器设置为 “存储卡 (ROM)”，并设置为 “下述文件指定”或者 “与程序相同”（二者之一）时，实际的存储卡插槽中却安装了 ATA 卡。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时			QnU*11

*1 以功能版本 B 以后为对象。
*5: 以序列号的前 5 位数为 “07032” 以后的模块为对象。
*10: 以除 Q00UJCPU 以外的通用型 QCPU 为对象。
*11: 以除 Q00UJCPU、Q00UCPU、Q01UCPU 以外的通用型 QCPU 为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
3001	[PARAMETER ERROR] 参数的内容已被损坏。 ■附加信息 ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 ■诊断时机 ・电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时	・通过外围设备读取出错的个别信息，对与该数值（参数号）对应的参数项目进行检查、修改。 ・将修改后的参数重新写入到 CPU 模块中，进行可编程控制器的电源再启动或者 CPU 模块的复位。 ・仍然发生相同的出错时，可能是硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）	RUN：熄灯 ERR.：闪烁 CPU 状态： 停止	QCPU
3002	[PARAMETER ERROR] 通过可编程控制器参数的可编程控制器文件设置在文件寄存器中选择了 “使用下述文件”时，虽然对文件寄存器的容量进行了设置，但指定的文件不存在。 ■附加信息 ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 ■诊断时机 ・电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时			Qn(H) QnPH QnPRH
	[PARAMETER ERROR] 通过可编程控制器参数的可编程控制器文件设置在文件寄存器中设置了 “使用下述文件”，且未对文件寄存器的容量进行设置时，指定的对象存储器中文件寄存器文件不存在。 ■附加信息 ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 ■诊断时机 ・电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时			QnU*10
	[PARAMETER ERROR] 通过可编程控制器参数的可编程控制器文件将软元件数据存储用文件设置为 “使用下述文件”，且未对容量进行设置时，对象存储器中软元件数据存储用文件不存在。 ■附加信息 ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 ■诊断时机 ・电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时			QnU

*10　以除 Q00UJCPU 以外的通用型 QCPU 为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
3003	[PARAMETER ERROR] 多 CPU 系统的自动刷新范围超出了文件寄存器的容量。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・执行 END 指令时	变更为可进行全部范围刷新的文件寄存器。	RUN：熄灯 ERR.：闪烁 CPU 状态： 停止	Qn(H)*1 QnPH QnU*10
	[PARAMETER ERROR] 在可编程控制器参数的软元件设置中设置的软元件点数超出了 CPU 模块的允许使用范围。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时	・通过外围设备读取出错的个别信息，对与该数值（参数号）对应的参数项目进行检查、修改。 ・修改了参数后仍然发生相同的出错时，可能是 CPU 模块的程序存储器、存储卡的存储器故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		QCPU
3004	[PARAMETER ERROR] 参数文件不正确。 或者文件内容不是参数。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时	检查参数文件的类型是否为 ***.QPA，检查文件的内容是否为参数。		
3005	[PARAMETER ERROR] 参数的内容已损坏。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	・通过外围设备读取出错的个别信息，对与该数值（参数号）对应的参数项目进行检查、修改。 ・将修改后的参数重新写入到 CPU 模块中，进行可编程控制器的电源再启动或者 CPU 模块的复位。 ・仍然发生相同的出错时，可能是硬件故障。请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。		Qn(H)*7 QnPH*9 QnPRH*9
3006	[PARAMETER ERROR] ・对 Q2CPU 进行了高速中断设置。 ・在多 CPU 系统中进行了高速中断设置。 ・使用 QA1S □ B/QA6 □ B 时进行了高速中断设置。 ・在高速中断设置中设置的 I/O 地址中未安装模块。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时	・将 Q2CPU 的高速中断设置删除。 使用高速中断时，将 CPU 模块变更为 Q02H/Q06H/Q12H/Q25HCPU。 ・在多 CPU 系统中使用时，将高速中断设置删除。 使用高速中断时，应设置为单 CPU 系统。 ・使用 QA1S □ B/QA6 □ B 时，将高速中断设置删除。 ・使用高速中断时，不使用 QA1S □ B/QA6 □ B。 ・对高速中断设置中设置的 I/O 地址进行重新审核。		Qn(H)*4

*1　以功能版本 B 以后为对象。
*4　以序列号的前 5 位数为 “04012” 以后的模块为对象。
*7:　以序列号的前 5 位数为 “09012” 以后的模块为对象。
*9:　以序列号的前 5 位数为 “10042” 以后的模块为对象。
*10:　以除 Q00UJCPU 以外的通用型 QCPU 为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
3007	[PARAMETER ERROR] 插杆开关的参数有效驱动器上的参数文件不是 CPU 模块中可以使用的参数文件。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时	使用 GX Developer 创建参数，将参数写入到插杆开关的参数有效驱动器中指定的驱动器中。	RUN: 熄灯 ERR.: 闪烁 CPU 状态： 停止	QnPRH
3009	[PARAMETER ERROR] 在多 CPU 系统中，将 AnS 用、A 用、Q2AS 用、QnA 用的模块设置到多个管理 CPU 中。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时	对可编程控制器参数的 I/O 分配设置进行变更，设置为通过 1 个 CPU 模块进行管理。（对多 CPU 系统的所有站的可编程控制器参数进行变更。）		Qn(H)*1
3010	[PARAMETER ERROR] 在多 CPU 系统中，可编程控制器参数的 CPU 模块个数与实际安装数不相符。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时	使（多 CPU 设置的 CPU 个数）-(I/O 分配的 CPU（空闲）设置）与 CPU 实际安装个数相符。		Qn(H)*1 QnPH
3012	[PARAMETER ERROR] 在多 CPU 系统中，多 CPU 设置、管理 CPU 的设置与基准 CPU 号的设置不匹配。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时	使可编程控制器参数的多 CPU 设置、管理 CPU 的设置与基准 CPU 号（1 号机）相匹配。		Q00/Q01*1 Qn(H)*1 QnU

*1 以功能版本 B 以后为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
3013	[PARAMETER ERROR] 在多 CPU 系统中，多 CPU 自动刷新设置处于下述状态。 ・将位软元件指定为刷新软元件时，未将刷新起始软元件的编号指定为 16 的倍数。 ・指定的软元件不是允许指定的软元件。 ・发送点数为奇数。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时	在多 CPU 设置的刷新设置中进行下述检查、修改。 ・指定位软元件时，将刷新起始软元件的编号指定为 16 的倍数。 ・指定允许指定的刷新软元件。 ・将发送点数设置为偶数。	RUN：熄灯 ERR.：闪烁 CPU 状态： 停止	Qn(H)*1 QnPH
	[PARAMETER ERROR] 在多 CPU 系统中，多 CPU 自动刷新设置处于下述状态。 ・发送点数的合计值超出了最大刷新点数。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时	在多 CPU 设置的刷新设置中进行下述检查、修改。 ・将发送点数的合计值设置在最大刷新点数的范围内。		Q00/Q01*1
	[PARAMETER ERROR] 在多 CPU 系统中，多 CPU 自动刷新设置处于下述状态。 ・指定的软元件不是允许指定的软元件。 ・发送点数为奇数。 ・发送点数的合计值超过了最大刷新点数。 ・刷新范围设置跨越了内部用户软元件及扩展数据寄存器 (D)、扩展链接寄存器 (W) 的界限。 ・未对自站的发送范围进行软元件设置。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时	在多 CPU 设置的刷新设置中进行下述检查、修改。 ・指定允许指定的刷新软元件。 ・将发送点数设置为偶数。 ・在设置时应使发送点数的合计值不超过最大刷新点数范围。 ・在进行刷新范围设置时应避免跨越内部用户软元件及扩展数据寄存器 (D)、扩展链接寄存器 (W) 的界限。 ・必须在自站的发送范围中进行刷新目标软元件指定。不需要发送范围的情况下，将相应发送范围删除。		QnU*10
3014	[PARAMETER ERROR] ・在多 CPU 系统中，在线模块更换参数（多 CPU 系统参数）与基准机号 CPU 的设置内容不相同。 ・在多 CPU 系统中，安装了不支持在线模块更换参数的 CPU 模块，却将在线模块更换设置设置为允许。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时	・使在线模块更换参数与基准机号 CPU 的相符。 ・安装了不支持在线模块更换的 CPU 模块时，更换为支持在线模块更换的 CPU 模块。		Qn(H) QnPH QnU*8

*1　以功能版本 B 以后为对象。
*8　以除 Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU 以外的通用型 QCPU 为对象。
*10　以除 Q00UJCPU 以外的通用型 QCPU 为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
3015	[PARAMETER ERROR] 多 CPU 系统配置时，参数设置与进行了校验的机号的 CPU 不相同。 ■附加信息 ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 /CPU 号 ■诊断时机 ・电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时	通过外围设置读取出错的个别信息，对与该数值（参数号 /CPU 号）对应的参数项目、对象 CPU 号的参数进行检查、修改。		QnU*8
3016	[PARAMETER ERROR] 多 CPU 同步启动设置中，将不支持多 CPU 同步启动的 CPU 模块设置为同步启动的对象。 ■附加信息 ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 /CPU 号 ■诊断时机 ・电源 ON 时 / 复位时 / 可编程控制器写入时	将不支持多 CPU 同步启动的 CPU 模块从同步启动对象中删除后重新进行设置。		
3040	[PARAMETER ERROR] 参数文件已损坏。 ■附加信息 ・公共信息： - ・个别信息： - ■诊断时机 ・电源 ON 时 / 复位时	通过 GX Developer 将 “可编程控制器参数 / 网络参数 / 远程口令”写入到参数有效驱动器中后，执行系统电源的再接通操作或者 CPU 模块的复位操作。如果仍然发生相同的出错，则可能是硬件故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）	RUN：熄灯 ERR.：闪烁 CPU 状态： 停止	
3041	[PARAMETER ERROR] 智能功能模块参数文件已损坏。 ■附加信息 ・公共信息： - ・个别信息： - ■诊断时机 ・电源 ON 时 / 复位时	通过 GX Developer 将 “智能功能模块参数”写入到参数有效驱动器中后，执行系统电源的再接通操作或者 CPU 模块的复位操作。如果仍然发生相同的出错，则可能是硬件故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		
3042	[PARAMETER ERROR] 存储远程口令设置内容的系统文件已损坏。 ■附加信息 ・公共信息： - ・个别信息： - ■诊断时机 ・电源 ON 时 / 复位时	・通过 GX Developer 将 “可编程控制器参数 / 网络参数 / 远程口令”写入到参数有效驱动器中后，执行系统电源的再接通操作或者 CPU 模块的复位操作。如果仍然发生相同的出错，则可能是硬件故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。） ・将参数有效驱动器设置为除 “程序存储器”以外时，应设置为通过引导文件设置将参数文件 (PARAM) 传送到程序存储器中。 通过 GX Developer 将 “可编程控制器参数 / 网络参数 / 远程口令”写入到参数有效驱动器中后，执行系统电源的再接通操作或者 CPU 模块的复位操作。如果仍然发生相同的出错，则可能是硬件故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		Qn(H)*5 QnPH*5 QnPRH*5

*5　以序列号的前 5 位数为 “07032”以后的模块为对象。
*8　以除 Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU 以外的通用型 QCPU 为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
3100	[LINK PARA. ERROR] 在多 CPU 系统中，在 CC-Link IE 控制网络模块的网络参数的起始 I/O 编号中，指定了其它站管理的 CC-Link IE 控制网络模块。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	・将其它站管理的 CC-Link IE 控制网络模块的网络参数删除。 ・变更为自站管理的 CC-Link IE 控制网络模块的起始 I/O 编号。	RUN：熄灯 ERR.：闪烁 CPU 状态： 停止	Qn(H)*7 QnPRH*9 QnU
	[LINK PARA. ERROR] 将以普通站动作中的 CC-Link IE 控制网络模块的网络参数改写为管理站。或者将以管理站动作中的 CC-Link IE 控制网络模块的网络参数改写为普通站。（通过复位可将网络参数反映到模块侧。） **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	对 CPU 模块进行复位。		
	[LINK PARA. ERROR] ・CC-Link IE 控制网络模块的网络参数的模块个数与实际安装个数不相符。 ・CC-Link IE 控制网络模块的网络参数的起始 I/O 编号与实际安装的 I/O 编号不相符。 ・参数中存在有不能处理的数据。 ・在电源处于 ON 的状态下对 CC-Link IE 控制网络模块的网络类型进行了改写。（网络类型变更时需要进行 RESET → RUN 操作。） **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	・确认网络参数与实际安装状态，不相符时使网络参数与实际安装状态一致。对网络参数进行了修改时，将其写入到 CPU 模块中。 ・确认扩展基板的扩展级数设置。 ・对扩展基板以及扩展电缆的连接状态进行确认。将 GOT 通过总线连接到主基板或者扩展基板时，对其连接状态也进行确认。 进行了上述确认后仍然发生出错时，可能是硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		Qn(H)*7 QnPH*9 QnPRH*9 QnU
	[LINK PARA. ERROR] ・在 MELSECNET/H 的网络参数的起始 I/O 编号中，指定了 CC-Link IE 控制网络模块。 ・在 CC-Link IE 控制网络模块的网络参数的起始 I/O 编号中，指定了 MELSECNET/H 模块。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时			

*7 以序列号的前 5 位数为 “09012” 以后的模块为对象。
*9 以序列号的前 5 位数为 “10042” 以后的模块为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
3100	[LINK PARA. ERROR] • 虽然安装了 CC-Link IE 控制网络模块，但未设置 CC-Link IE 控制网络模块的网络参数。 • 虽然安装了 CC-Link IE 控制网络模块及 MELSECNET/H 模块，但未设置 MELSECNET/H 的网络参数。 ■附加信息 • 公共信息： 文件名 / 驱动器名 • 个别信息： 参数号 ■诊断时机 • 电源 ON 时 / 复位时 /STOP → RUN 时	• 确认网络参数与实际安装状态，不相符时使网络参数与实际安装状态一致。对网络参数进行了修改时，将其写入到 CPU 模块中。 • 确认扩展基板的扩展级数设置。 • 对扩展基板以及扩展电缆的连接状态进行确认。将 GOT 通过总线连接到主基板或者扩展基板时，对其连接状态也进行确认。 进行了上述确认后仍然发生出错时，可能是硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）	RUN：熄灯 ERR.：闪烁 CPU 状态： 停止	Qn(H)*7 QnPH*9 QnPRH*9 QnU
	[LINK PARA. ERROR] 在多 CPU 系统中，在 MELSECNET/H 的网络参数的起始 I/O 编号中，指定了其它站管理的 MELSECNET/H 模块。 ■附加信息 • 公共信息： 文件名 / 驱动器名 • 个别信息： 参数号 ■诊断时机 • 电源 ON 时 / 复位时 /STOP → RUN 时	• 将其它站管理的 MELSECNET/H 模块的网络参数删除。 • 变更为自站管理的 MELSECNET/H 模块的起始 I/O 编号。		Q00/Q01*1 Qn(H)*1 QnPH QnU*10
	[LINK PARA. ERROR] 将以普通站动作中的 MELSECNET/H 模块的网络参数改写为管理站。或者将以管理站动作中的 MELSECNET/H 模块的网络参数改写为普通站。（通过复位可将网络参数反映到模块侧。） ■附加信息 • 公共信息： 文件名 / 驱动器名 • 个别信息： 参数号 ■诊断时机 • 电源 ON 时 / 复位时 /STOP → RUN 时	对 CPU 模块进行复位。		Qn(H)*1 QnPH QnPRH QnU
	[LINK PARA. ERROR] • MELSECNET/H 模块的网络参数的模块个数与实际安装个数不相符。 • MELSECNET/H 模块的网络参数的起始 I/O 编号与实际安装的 I/O 编号不相符。 • 参数中存在有不能处理的数据。 • 在电源处于 ON 的状态下对 MELSECNET/H 模块的网络类型进行了改写。（网络类型变更时需要进行 RESET → RUN 操作。） • MELSECNET/H 模块 *5 的模式开关超出了范围。 ■附加信息 • 公共信息： 文件名 / 驱动器名 • 个别信息： 参数号 ■诊断时机 • 电源 ON 时 / 复位时 /STOP → RUN 时	• 确认网络参数与实际安装状态，不相符时使网络参数与实际安装状态一致。 对网络参数进行了修改时，将其写入到 CPU 模块中。 • 确认扩展基板的扩展级数设置。 • 对扩展基板以及扩展电缆的连接状态进行确认。将 GOT 通过总线连接到主基板或者扩展基板时，对其连接状态也进行确认。 进行了上述确认后仍然发生出错时，可能是硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。） • 将 MELSECNET/H 模块 *5 的模式开关设置在范围内。		QCPU

*1 以功能版本 B 以后为对象。
*5: 以序列号的前 5 位数为 “07032” 以后的模块为对象。
*7: 以序列号的前 5 位数为 “09012” 以后的模块为对象。
*9: 以序列号的前 5 位数为 “10042” 以后的模块为对象。
*10: 以除 Q00UJCPU 以外的通用型 QCPU 为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
3101	[LINK PARA. ERROR] 链接刷新范围超出了文件寄存器的容量。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・执行 END 指令时	变更为可进行全部范围刷新的文件寄存器。	RUN：熄灯 ERR.：闪烁 CPU 状态： 停止	Qn(H)*1 QnPH QnPRH QnU*10
	[LINK PARA. ERROR] ・MELSECNET/H 的站号为 0 时，设置了可编程控制器网络的参数。 ・MELSECNET/H 的站号为 0 以外时，设置了远程主站的参数。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	对网络参数的 MELSECNET/H 模块的类型或者站号进行修改，使其与所使用的系统相符。		Qn(H)*1 QnPH QnPRH
	[LINK PARA. ERROR] CC-Link IE 控制网络的刷新参数超出了范围。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	・确认网络参数与实际安装状态，不相符时使网络参数与实际安装状态一致。 对网络参数进行了修改时，将其写入到 CPU 模块中。 ・确认扩展基板的扩展级数设置。 ・对扩展基板以及扩展电缆的连接状态进行确认。 将 GOT 通过总线连接到主基板或者扩展基板时，对其连接状态也进行确认。 进行了上述确认后仍然发生出错时，可能是硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		Qn(H)*7 QnPH*9 QnPRH*9 QnU
	[LINK PARA. ERROR] ・网络参数中指定的网络号与实际安装不相符。 ・网络参数中指定的起始 I/O 编号与实际安装不相符。 ・网络参数中指定的网络类型与实际安装不相符。 ・MELSECNET/H、MELSECNET/10 的刷新参数超出了范围。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时			QCPU
	[LINK PARA. ERROR] 在不支持 MELSECNET/H 多重远程 I/O 网络的模块中实施了多重远程 I/O 网络功能。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	安装支持 MELSECNET/H 多重远程 I/O 网络的模块。		QnPH

*1 以功能版本 B 以后为对象。
*7: 以序列号的前 5 位数为 “09012” 以后的模块为对象。
*9: 以序列号的前 5 位数为 “10042” 以后的模块为对象。
*10: 以除 Q00UJCPU 以外的通用型 QCPU 为对象。

<table>
<tr><th>出错代码</th><th>异常内容及原因</th><th>处理方法</th><th>LED 状态
CPU 状态</th><th>对应 CPU</th></tr>
<tr><td rowspan="4">3101</td><td>[LINK PARA. ERROR]
・CPU 模块中 A 系统侧的 MELSECNET/H 远程主站的站号为 0 以外。
・CPU 模块中 B 系统侧的 MELSECNET/H 远程主站的站号为 0 以外。
■附加信息
・公共信息： 文件名 / 驱动器名
・个别信息： 参数号
■诊断时机
・电源 ON 时 / 复位时 /STOP → RUN 时</td><td>・将 CPU 模块中 A 系统侧的 MELSECNET/H 远程主站的站号设置为 0。
・将 CPU 模块中 B 系统侧的 MELSECNET/H 远程主站的站号设置为 1 ～ 64。</td><td rowspan="4">RUN: 熄灯
ERR.：闪烁

CPU 状态： 停止</td><td>QnPRH</td></tr>
<tr><td>[LINK PARA. ERROR]
在未设置 MELSECNET/H 的参数时，由于可编程控制器参数的软元件设置的 B/W 的软元件点数少于下表的 B/W 刷新软元件点数，因此不能对 MELSECNET/H 进行刷新。
<table><tr><th colspan="2">刷新软元件</th><th>B软元件的
刷新软元件点数</th><th>W软元件的
刷新软元件点数</th></tr><tr><td rowspan="4">网络模块
安装个数</td><td>1个</td><td>8192点
(8192点×1个模块)</td><td>8192点
(8192点×1个模块)</td></tr><tr><td>2个</td><td>8192点
(4096点×2个模块)</td><td>8192点
(4096点×2个模块)</td></tr><tr><td>3个</td><td>6144点
(2048点×3个模块)</td><td>6144点
(2048点×3个模块)</td></tr><tr><td>4个</td><td>8192点
(2048点×4个模块)</td><td>8192点
(2048点×4个模块)</td></tr></table>■附加信息
・公共信息： 文件名 / 驱动器名
・个别信息： 参数号
■诊断时机
・电源 ON 时 / 复位时 /STOP → RUN 时</td><td>设置 MELSECNET/H 的刷新参数时，应与可编程控制器参数的软元件设置的 B/W 的软元件点数相符。</td><td>Qn(H)*6
QnPH*6
QnPRH*6
QnU</td></tr>
<tr><td>[LINK PARA. ERROR]
刷新范围设置跨越了内部用户软元件及扩展数据寄存器 (D)、扩展链接寄存器 (W) 的界限。
■附加信息
・公共信息： 文件名 / 驱动器名
・个别信息： 参数号
■诊断时机
・电源 ON 时 / 复位时 /STOP → RUN 时</td><td>在进行刷新范围设置时应避免跨越内部用户软元件及扩展数据寄存器 (D)、扩展链接寄存器 (W) 的界限。</td><td>QnU</td></tr>
<tr><td rowspan="2">3102</td><td>[LINK PARA. ERROR]
检测出 CC-Link IE 控制网络模块的网络参数异常。
■附加信息
・公共信息： 文件名 / 驱动器名
・个别信息： 参数号
■诊断时机
・电源 ON 时 / 复位时 /STOP → RUN 时</td><td rowspan="2">・修改了网络参数后，进行写入。
・修改后仍然发生出错时，可能是硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）</td><td>Qn(H)*7
QnPH*9
QnPRH*9
QnU</td></tr>
<tr><td>[LINK PARA. ERROR]
・检测出网络模块网络参数异常。
・检测出 MELSECNET/H 的网络参数异常。
■附加信息
・公共信息： 文件名 / 驱动器名
・个别信息： 参数号
■诊断时机
・电源 ON 时 / 复位时 /STOP → RUN 时</td><td>QCPU</td></tr>
</table>

*6 以序列号的前 5 位数为 “08102” 以后的 MELSECNET/H 模块为对象。
*7 以序列号的前 5 位数为 “09012” 以后的模块为对象。
*9 以序列号的前 5 位数为 “10042” 以后的模块为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
3102	[LINK PARA. ERROR] 进行了双工设置的站号不正确。 ・站号不是连号。 ・对普通站的 CPU 模块未进行双工设置。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	参阅网络模块的故障排除，双工设置导致出错时对网络参数的双工设置重新进行审核。	RUN: 熄灯 ERR.: 闪烁 CPU 状态： 停止	QnPRH
	[LINK PARA. ERROR] 安装了序列号的前 5 位数为 “09041” 以前的 CC-Link IE 控制网络模块。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	安装序列号的前 5 位数为 “09042” 以后的 CC-Link IE 控制网络模块。		QnU
	[LINK PARA. ERROR] 对不支持组循环功能的 CC-Link IE 控制网络进行了组循环功能的设置。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	对功能版本 D 以后的 CC-Link IE 控制网络进行组循环功能的设置。		QnU*9
	[LINK PARA. ERROR] 对除冗余系统以外的 CPU 中安装的 CC-Link IE 控制网络模块进行了双工设置。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	对管理站的网络参数的双工设置重新进行审核。		Q00J/Q00/Q01 Qn(H)*9 QnPH*9 QnU*9
	[LINK PARA. ERROR] ・将 LB/LW 的自站发送范围设置到 LB/LW4000 以后。 ・进行了 LB/LW 设置 (2) 的设置。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	应对管理站的网络参数的网络范围分配重新进行审核。		Q00J/Q00/Q01

*9 以序列号的前 5 位数为 “10042” 以后的模块为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
3103	[LINK PARA. ERROR] 在多 CPU 系统中，以太网的网络参数的起始 I/O 编号中，指定了其它站管理的以太网模块。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	・将其它站管理的以太网模块的网络参数删除。 ・变更为自站管理的以太网模块的起始 I/O 编号。	RUN: 熄灯 ERR.：闪烁 CPU 状态： 停止	Q00/Q01*1 Qn(H)*1 QnPH QnU*10
	[LINK PARA. ERROR] ・虽然将以太网的模块个数设置参数的模块个数设置为 1 个以上，但实际安装数为 0。 ・以太网的网络参数的起始 I/O 编号与实际安装的起始 I/O 编号不相符。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	・修改网络参数后，进行写入。 ・修改后仍然发生出错时，可能是硬件故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		QCPU
	[LINK PARA. ERROR] ・在冗余系统中，网络类型设置为 “以太网（主基板）” 的以太网模块被安装到扩展基板上。 ・在冗余系统中，网络类型设置为 “以太网（扩展基板）” 的以太网模块被安装到主基板上。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时			QnPRH*7
3104	[LINK PARA. ERROR] ・以太网、MELSECNET/H、MELSECNET/10 中使用了相同的网络号。 ・网络参数设置的网络号、站号、组号超出了范围。 ・I/O 编号指定超出了所使用的 CPU 模块的范围。 ・以太网固有的参数内容不正常。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时			QCPU

*1 以功能版本 B 以后为对象。
*7 以序列号的前 5 位数为 “09012” 以后的模块为对象。
*10 以除 Q00UJCPU 以外的通用型 QCPU 为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
3105	[LINK PARA. ERROR] 在多 CPU 系统中，在 CC-Link IE 控制网络参数的起始 I/O 编号中，指定了其它站管理的 CC-Link 模块。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	・将其它站管理的 CC-Link 模块的网络参数删除。 ・变更为自站管理的 CC-Link 模块的起始 I/O 编号。	RUN: 熄灯 ERR.: 闪烁 CPU 状态： 停止	Q00/Q01*1 Qn(H)*1 QnPH QnU*10
	[LINK PARA. ERROR] ・虽然将 CC-Link 模块个数设置参数的模块个数设置为 1 个以上，但实际安装数为 0。 ・公共参数的起始 I/O 编号与实际安装的 I/O 编号不相符。 ・CC-Link 模块个数设置参数的站类型不一致。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	・修改网络参数后，进行写入。 ・修改后仍然发生出错时，可能是硬件故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		QCPU
	[LINK PARA. ERROR] ・在冗余系统中，将站类型设置为 “主站（支持冗余功能）” 的 CC-Link 模块安装到扩展基板上。 ・在冗余系统中，将站类型设置为 “主站（扩展基板）” 的 CC-Link 模块安装到主基板上。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时			QnPRH*7
3106	[LINK PARA. ERROR] CC-Link 链接刷新范围超出了文件寄存器的容量。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・执行 END 指令时	变更为可进行全部范围刷新的文件寄存器。		Qn(H)*1 QnPH QnPRH QnU
	[LINK PARA. ERROR] CC-Link 的网络刷新参数超出了范围。 **■附加信息** ・公共信息： 文件名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	对参数设置重新进行审核。		QCPU
	[LINK PARA. ERROR] 网络刷新范围设置跨越了内部用户软元件及扩展数据寄存器 (D)、扩展链接寄存器 (W) 的界限。 **■附加信息** ・公共信息： 文件名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	在进行网络刷新范围设置时应避免跨越内部用户软元件及扩展数据寄存器 (D)、扩展链接寄存器 (W) 的界限。		QnU

*1　以功能版本 B 以后为对象。
*7　以序列号的前 5 位数为 “09012” 以后的模块为对象。
*10　以除 Q00UJCPU 以外的通用型 QCPU 为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
3107	[LINK PARA. ERROR] • CC-Link 的参数内容不正常。 • 实际安装的 CC-Link 模块的版本不支持所设置的模式。 ■附加信息 • 公共信息： 文件名 • 个别信息： 参数号 ■诊断时机 • 电源 ON 时 / 复位时 /STOP → RUN 时	对参数设置进行重新审核。		QCPU
3200	[SFC PARA. ERROR] 参数内容非法。 • 在可编程控制器参数的 SFC 设置中将块 0 设置为“自动启动”，但块 0 并不存在。 ■附加信息 • 公共信息： 文件名 • 个别信息： 参数号 ■诊断时机 • STOP → RUN 时	通过外围设备读取出错的公共信息，对与该数值（程序出错位置）对应的出错步进行检查、修改。	RUN: 熄灯 ERR.: 闪烁 CPU 状态： 停止	Q00J/Q00/Q01*1 QnPH QnPRH QnU
3201	[SFC PARA. ERROR] 块参数内容非法。 ■附加信息 • 公共信息： 文件名 • 个别信息： 参数号 ■诊断时机 • STOP → RUN 时			Qn(H) QnPH QnPRH
3202	[SFC PARA. ERROR] 可编程控制器参数的软元件设置中设置的步进继电器的个数少于程序中使用的个数。 ■附加信息 • 公共信息： 文件名 • 个别信息： 参数号 ■诊断时机 • STOP → RUN 时			
3203	[SFC PARA. ERROR] 在可编程控制器参数的程序设置中设置的 SFC 程序的执行类型为除扫描执行型以外。 ■附加信息 • 公共信息： 文件名 • 个别信息： 参数号 ■诊断时机 • 电源 ON 时 / 复位时 /STOP → RUN 时 *3			Qn(H) QnPH QnPRH QnU
3300	[SP. PARA ERROR] GX Configurator 中设置的智能功能模块的参数的起始 I/O 编号与实际安装的 I/O 编号不相符。 ■附加信息 • 公共信息： 文件名 • 个别信息： 参数号 *2 ■诊断时机 • 电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时	对参数设置进行重新审核。		QCPU

*1 以功能版本 B 以后为对象。
*2 参数号为 GX Configurator 中设置的智能功能模块的参数的起始 I/O 编号÷ 10H 后所得的值。
*3 除通用型 QCPU 以外的 CPU 模块的诊断时机仅为 STOP → RUN 时。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
3301	[SP. PARA ERROR] ・智能功能模块的刷新范围超出了文件寄存器的容量。 ・GX Configurator 中设置的智能功能模块与实际安装的模块不相符。 **■附加信息** ・公共信息： 文件名 ・个别信息： 参数号 *2 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时	・变更为可进行全部范围刷新的文件寄存器。 ・对参数设置进行重新审核。 ・对自动刷新的设置进行重新审核。	RUN: 熄灯 ERR.：闪烁 CPU 状态： 停止	Q00J/Q00/Q01 Qn(H)*1 QnPH QnPRH QnU
	[SP. PARA ERROR] 智能功能模块的刷新参数超出了范围。 **■附加信息** ・公共信息： 文件名 ・个别信息： 参数号 *2 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时	・对参数设置进行重新审核。 ・对自动刷新的设置进行重新审核。		QCPU
	[SP. PARA ERROR] 刷新参数的范围设置跨越了内部用户软元件及扩展数据寄存器 (D)、扩展链接寄存器 (W) 的界限。 **■附加信息** ・公共信息： 文件名 ・个别信息： 参数号 *2 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时	在进行刷新范围设置时应避免跨越内部用户软元件及扩展数据寄存器 (D)、扩展链接寄存器 (W) 的界限。		QnU
3302	[SP. PARA ERROR] 智能功能模块的参数不正常。 **■附加信息** ・公共信息： 文件名 ・个别信息： 参数号 *2 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时	对参数设置进行重新审核。		QCPU
3303	[SP. PARA ERROR] 在多 CPU 系统中，对其它站管理的智能功能模块进行了自动刷新设置等参数设置。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时	・将其它站管理的智能功能模块的自动刷新设置等参数设置删除。 ・变更为自站管理的智能功能模块的自动刷新设置等参数设置。		Q00/Q01*1 Qn(H)*1 QnPH QnU*10

*1 以功能版本 B 以后为对象。
*2 参数号为 GX Configurator 中设置的智能功能模块的参数的起始 I/O 编号÷ 10H 后所得的值。
*10 以除 Q00UJCPU 以外的通用型 QCPU 为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
3400	[REMOTE PASS. ERR.] 远程口令的对象模块的起始 I/O 编号设置超出了 0H ～ 0FF0H 的范围。 **■附加信息** ・公共信息：－ ・个别信息：－ **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	将对象模块的起始 I/O 编号设置在 0H ～ 0FF0H 的范围内。	RUN：熄灯 ERR.：闪烁 CPU 状态：停止	Qn(H)*1 QnPH QnPRH QnU*7
	[REMOTE PASS. ERR.] 远程口令的对象模块的起始 I/O 编号设置超出了 0H ～ 07E0H 的范围。 **■附加信息** ・公共信息：－ ・个别信息：－ **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	将对象模块的起始 I/O 编号设置在 0H ～ 07E0H 的范围内。		Q02U
	[REMOTE PASS. ERR.] 远程口令的对象模块的起始 I/O 编号设置超出了下述范围。 Q00JCPU：0H ～ 1E0H Q00CPU/Q01CPU：0H ～ 3E0H **■附加信息** ・公共信息：－ ・个别信息：－ **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	将对象模块的起始 I/O 编号设置变更为以下范围内。 ・Q00JCPU：0H ～ 1E0H ・Q00CPU/Q01CPU：0H ～ 3E0H		Q00J/Q00/Q01*1

*1　以功能版本 B 以后为对象。
*7　以序列号的前 5 位数为“09012”以后的模块为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
3401	[REMOTE PASS. ERR.] 远程口令的起始 I/O 编号中指定的位置异常。 ・未安装模块。 ・安装了除智能功能模块以外（I/O 模块）的模块。 ・安装了除串行通信模块、调制解调器接口模块、以太网模块以外的智能功能模块。 ・安装了功能版本 A 的串行通信模块、以太网模块。 安装了不支持远程口令的智能功能模块。 **■附加信息** ・公共信息：－ ・个别信息：－ **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	在远程口令的起始 I/O 编号中指定的位置上安装功能版本 B 以后的串行通信模块、调制解调器接口模块、以太网模块。	RUN：熄灯 ERR.：闪烁 CPU 状态：停止	Qn(H)*1 QnPH QnPRH QnU
	[REMOTE PASS. ERR.] 在远程口令的起始 I/O 编号中指定的插槽中未安装下述模块。 ・功能版本 B 以后的串行通信模块 ・功能版本 B 以后的以太网模块 ・功能版本 B 以后的调制解调器接口模块 **■附加信息** ・公共信息：－ ・个别信息：－ **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	在远程口令的起始 I/O 编号中指定的插槽中安装下述模块。 ・功能版本 B 以后的串行通信模块 ・功能版本 B 以后的以太网模块 ・功能版本 B 以后的调制解调器接口模块		Q00J/Q00/Q01*1
	[REMOTE PASS. ERR.] 在多 CPU 系统中，指定了其它站管理的功能版本 B 以后的串行通信模块、调制解调器接口模块、以太网模块。 **■附加信息** ・公共信息：－ ・个别信息：－ **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时	・指定自站管理的功能版本 B 以后的以太网模块。 ・删除远程口令的设置。		Qn(H)*1 QnPH QnU*10

*1　以功能版本 B 以后为对象。
*10　以除 Q00UJCPU 以外的通用型 QCPU 为对象。

12.1.6 出错代码一览表 (4000 ～ 4999)

以下介绍出错代码 4000 ～ 4999 的出错信息、异常内容及原因以及处理方法。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
4000	[INSTRCT. CODE ERR] ・在程序中包含有 CPU 模块无法解读的指令代码。 ・程序中包含有无法使用的指令。 **■附加信息** ・公共信息： 程序出错位置 ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时 / 执行指令时	通过外围设备读取出错的公共信息，对与该数值（程序出错位置）相对应的出错步进行检查、修改。	RUN: 熄灯 ERR.：闪烁 CPU 状态： 停止	QCPU
4001	[INSTRCT. CODE ERR] 在不是 SFC 程序的程序中包含有 SFC 专用指令。 **■附加信息** ・公共信息： 程序出错位置 ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时 / 执行指令时			Q00J/Q00/Q01*2 Qn(H) QnPH QnPRH QnU
4002	[INSTRCT. CODE ERR] ・程序中指定的专用指令的指令名称有误。 ・程序中指定的专用指令在指定的模块中不能执行。 **■附加信息** ・公共信息： 程序出错位置 ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时 / 执行指令时			QCPU
4003	[INSTRCT. CODE ERR] 程序中指定的专用指令的软元件数有误。 **■附加信息** ・公共信息： 程序出错位置 ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时 / 执行指令时			
4004	[INSTRCT. CODE ERR] 指定了程序中指定的专用指令不能使用的软元件。 **■附加信息** ・公共信息： 程序出错位置 ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时 / 执行指令时			

*2 以功能版本 B 以后为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
4010	[MISSING END INS.] 程序内没有 END(FEND) 指令。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・电源 ON 时 / 复位时 /STOP → RUN 时	通过外围设备读取出错的公共信息，对与该数值（程序出错位置）相对应的出错步进行检查、修改。	RUN：熄灯 ERR.：闪烁 CPU 状态： 停止	QCPU
4020	[CAN'T SET(P)] 程序中使用的文件内指针的总数超出了参数中设置的文件内指针的个数。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・电源 ON 时 / 复位时 /STOP → RUN 时			Qn(H) QnPH QnPRH QnU
4021	[CAN'T SET(P)] ・各文件中使用的公共指针的指针号重复。 ・各文件内使用的本地指针的指针号重复。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・电源 ON 时 / 复位时 /STOP → RUN 时			QCPU
4030	[CAN'T SET(I)] 各文件中使用的中断指针的指针号重复。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・电源 ON 时 / 复位时 /STOP → RUN 时			
4100	[OPERATION ERROR] 指令中包含有无法处理的数据。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时		RUN: 熄灯 / 亮灯 ERR.： 闪烁 / 亮灯 CPU 状态： 停止 / 继续运行 *1	
	[OPERATION ERROR] 通过 SP.FREAD/SP.FWRITE 指令访问 ATA 卡发生异常。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时	・采取防噪声措施。 ・对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是 ATA 卡的硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		Qn(H) QnPH QnPRH QnU*11
	[OPERATION ERROR] 通过 SP.FWRITE 指令访问了其它功能正在访问的文件。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时	・不要通过 SP.FWRITE 指令访问其它功能正在访问的文件。 ・不要将其它功能的访问与 SP.FWRITE 指令的执行同时进行。		QnU*11

*1 发生出错时的 CPU 的动作状态可以在参数中进行设置。(LED 显示也将联动变化。)
*11 以除 Q00UJCPU、Q00UCPU、Q01UCPU 以外的通用型 QCPU 为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
4101	[OPERATION ERROR] ・指令中处理数据的设置使用数超出了允许使用范围。 ・指令中指定的软元件的存储数据、常数超出了允许使用范围。 ・在至自站 CPU 共享存储器的写入中，在写入目标地址中指定了写入指定禁止区。 ・指令中指定的软元件的存储数据范围重复。 ・指令中指定的软元件超出了软元件点数范围。 ・指令中指定的中断指针号超出了允许使用范围。 ・通过 BMOV 指令在 (S)、(D) 二者中均指定了链接直接软元件、智能功能模块软元件、多 CPU 共享软元件。 **■附加信息** ・公共信息：程序出错位置 ・个别信息：－ **■诊断时机** ・执行指令时	通过外围设备读取出错公共信息，对与该数值（程序出错位置）对应的出错步进行检查、更换。	RUN: 熄灯 / 亮灯 ERR.: 闪烁 / 亮灯 CPU 状态： 停止 / 继续运行 *1	QCPU
	[OPERATION ERROR] ・指令中指定的文件寄存器的存储数据超出了允许使用范围。或者未进行文件寄存器设置。 **■附加信息** ・公共信息：程序出错位置 ・个别信息：－ **■诊断时机** ・执行指令时			QnU*10
	[OPERATION ERROR] ・指定的块数据跨越了内部用户软元件及扩展数据寄存器 (D)、扩展链接寄存器 (W) 的界限。（也包括 BIN32 位、实数（单精度、双精度）、间接地址、控制数据等。） **■附加信息** ・公共信息：程序出错位置 ・个别信息：－ **■诊断时机** ・执行指令时			QnU

*1　发生出错时的 CPU 的动作状态可以在参数中进行设置。(LED 显示也将联动变化。)
*10　以除 Q00UJCPU 以外的通用型 QCPU 为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
4102	[OPERATION ERROR] 在多 CPU 系统中，对其它站管理的网络模块指定了链接直接软元件（J □ \ □ ）。 **■附加信息** ・公共信息： 程序出错位置 ・个别信息： - **■诊断时机** ・执行指令时	・将指定了其它站管理的网络模块的链接直接软元件从程序中删除。 ・在链接直接软元件中指定自站管理的网络模块。	RUN: 熄灯 / 亮灯 ERR.: 闪烁 / 亮灯 CPU 状态： 停止 / 继续运行 *1	Q00/Q01*2 Qn(H)*2 QnPH QnU*10
	[OPERATION ERROR] ・专用指令中指定的网络号、站号有误。 ・链接直接软元件（J □ \ □ ）的设置不正确。 ・专用指令中指定的模块号 / 网络号 / 字符串个数超出了允许指定范围。 **■附加信息** ・公共信息： 程序出错位置 ・个别信息： - **■诊断时机** ・执行指令时	通过外围设备读取出错的公共信息，对与该数值（程序出错位置）相对应的出错步进行检查、修改。		QCPU
	[OPERATION ERROR] 专用指令中指定的字符串的指定（ “ ”）不是可用的字符串。 **■附加信息** ・公共信息： 程序出错位置 ・个别信息： - **■诊断时机** ・执行指令时			QnU
4103	[OPERATION ERROR] PID 专用指令的构成不正确。 **■附加信息** ・公共信息： 程序出错位置 ・个别信息： - **■诊断时机** ・执行指令时			Q00J/Q00/ Q01*2 Qn(H) QnPRH QnU
4105	[OPERATION ERROR] 设置了程序存储器检查时，执行了 PLOADP/PUNLOADP/PSWAPP 指令。 **■附加信息** ・公共信息： 程序出错位置 ・个别信息： - **■诊断时机** ・执行指令时	・删除程序存储器检查设置。 ・使用程序存储器检查的情况下，删除 PLOADP/PUNLOADP/PSWAPP 指令。	RUN: 熄灯 / 亮灯 ERR.: 闪烁 / 亮灯 CPU 状态： 停止 / 继续运行	QnPH*5
4107	[OPERATION ERROR] 从 1 个 CPU 模块执行了 33 条以上的多 CPU 专用指令。 **■附加信息** ・公共信息： 程序出错位置 ・个别信息： - **■诊断时机** ・执行指令时	通过多 CPU 专用指令的结束位采取互锁，防止从 1 个 CPU 模块执行 33 条以上的多 CPU 专用指令。	RUN: 熄灯 / 亮灯 ERR.: 闪烁 / 亮灯 CPU 状态： 停止 / 继续运行 *1	Q00/Q01*2 Qn(H)*2 QnPH Q00U/Q01U Q02U

*1　发生出错时的 CPU 的动作状态可以在参数中进行设置。(LED 显示也将联动变化。)
*2　以功能版本 B 以后为对象。
*5　以序列号的前 5 位数为 “07032” 以后的模块为对象。
*10　以除 Q00UJCPU 以外的通用型 QCPU 为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
4109	[OPERATION ERROR] 设置了高速中断时执行了 PR、PRC、UDCNT1、UDCNT2、PLSY、PWM 指令 。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时	删除高速中断设置。 使用高速中断时，删除 PR、PRC、UDCNT1、UDCNT2、PLSY、PWM 指令。	RUN: 熄灯 / 亮灯 ERR.: 闪烁 / 亮灯 CPU 状态： 停止 / 继续运行*1	Qn(H)*3
4111	[OPERATION ERROR] 试图通过指令对自站 CPU 模块的 CPU 共享存储器写入 / 读取禁止区进行写入 / 读取。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时	通过 GX Developer 读取出错的公共信息，对与该数值（程序出错位置）相对应的出错步进行检查、修改。		Q00/Q01*2 QnU
4112	[OPERATION ERROR] 通过多 CPU 专用指令指定的不能指定的 CPU 模块。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时			Q00/Q01*2 QnU*10
4113	[OPERATION ERROR] ・执行 SP.DEVST 指令时，当天至标准 ROM 的写入次数超过了 SD695 中指定的值。 ・SD695 中设置了超出范围的值。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时	・确认 SP.DEVST 指令的执行次数是否合适。 ・从翌日或以后再次执行 SP.DEVST 指令，或者对 SD695 的值进行调整。 ・将 SD695 的值修改为允许范围内的值。	RUN: 熄灯 / 亮灯 ERR.: 闪烁 / 亮灯 CPU 状态： 停止 / 继续运行	QnU
4120	[OPERATION ERROR] 由于手动切换允许标志（特殊寄存器 SM1592）处于 OFF 状态，因此不能通过控制系统切换指令 (SP.CONTSW) 进行手动切换。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时	通过 SP.CONTSW 指令进行控制系统切换时，将手动切换允许标志（特殊寄存器 SM1592）置为 ON 后再执行操作。	RUN: 熄灯 / 亮灯 ERR.: 闪烁 / 亮灯 CPU 状态： 停止 / 继续运行*1	QnPRH
4121	[OPERATION ERROR] ・分开模式时，在待机系统 CPU 模块中执行了控制系统切换指令 (SP.CONTSW)。 ・调试模式时执行了控制系统切换指令 (SP.CONTSW)。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时	・重新审核 SP.CONTSW 指令的互锁信号，确认 SP.CONTSW 指令只能在控制系统中执行。（由于在待机系统中不能执行 SP.CONTSW 指令，因此建议通过运行模式信号等采取互锁。 ・由于在调试模式下不能执行 SP.CONTSW 指令，因此建议对运行模式相关的互锁信号进行重新审核。		

*1 发生出错时的 CPU 的动作状态可以在参数中进行设置。(LED 显示也将联动变化。)
*2 以功能版本 B 以后为对象。
*3 以序列号的前 5 位数为 “04012” 以后的模块为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
4122	[OPERATION ERROR] ・在冗余系统中，试图对安装在扩展基板上的模块执行专用指令。 ・在分开模式时从待机系统向安装在扩展基板上的智能功能模块执行了访问指令。 **■附加信息** ・公共信息： 程序出错位置 ・个别信息： - **■诊断时机** ・执行指令时	・将以安装在扩展基板上的模块为对象的专用指令删除。 ・将从待机系统向安装在扩展基板上的智能功能模块执行访问的指令删除。		QnPRH*6
4130	[OPERATION ERROR] 以 ATA 卡内的注释文件为对象，执行了 SFC 步注释读取指令 (S(P).SFCSCOMR)、SFC 移位条件注释读取指令 (S(P).SFCTCOMR)。 **■附加信息** ・公共信息： 程序出错位置 ・个别信息： - **■诊断时机** ・执行指令时 / 执行 END 指令时	将对象注释文件设置为除 ATA 卡内的注释文件以外。	RUN: 熄灯 / 亮灯 ERR.: 闪烁 / 亮灯 CPU 状态： 停止 / 继续运行	Qn(H)*4 QnPH*5 QnPRH
4131	[OPERATION ERROR] 在 SFC 程序的动作过程中，通过指令启动了另一个 SFC 程序。 **■附加信息** ・公共信息： 程序出错位置 ・个别信息： - **■诊断时机** ・执行指令时	确认相应指令中指定的程序，或者确认 SFC 程序的执行状态。		
4140	[OPERATION ERROR] 输入数据以特殊数（“-0”、非正规数、非数、±∞）进行了运算。 **■附加信息** ・公共信息： 程序出错位置 ・个别信息： - **■诊断时机** ・执行指令时	通过外围设备读取出错的公共信息，对与该数值（程序出错位置）对应的出错步进行检查、修改。		QnU
4141	[OPERATION ERROR] 运算时发生了溢出。 **■附加信息** ・公共信息： 程序出错位置 ・个别信息： - **■诊断时机** ・执行指令时			
4200	[FOR-NEXT ERROR] 在未执行 FOR 指令的情况下执行了 NEXT 指令。 或者，NEXT 指令的个数少于 FOR 指令的个数。 **■附加信息** ・公共信息： 程序出错位置 ・个别信息： - **■诊断时机** ・执行指令时		RUN: 熄灯 ERR.: 闪烁 CPU 状态： 停止	QCPU

*4 以序列号的前 5 位数为“07012”以后的模块为对象。
*5 以序列号的前 5 位数为“07032”以后的模块为对象。
*6 以序列号的前 5 位数为“09012”以后的模块为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
4201	[FOR-NEXT ERROR] 在未执行 FOR 指令的情况下执行了 NEXT 指令。 或者，NEXT 指令的个数多于 FOR 指令的个数。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时	通过外围设备读取出错的公共信息，对与该数值（程序出错位置）对应的出错步进行检查、修改。	RUN: 熄灯 ERR.：闪烁 CPU 状态： 停止	QCPU
4202	[FOR-NEXT ERROR] 嵌套超过了 16 级。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时	将嵌套置为 16 级以内。		
4203	[FOR-NEXT ERROR] 在未执行 FOR 指令的情况下执行了 BREAK 指令。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时	通过外围设备读取出错的公共信息，对与该数值（程序出错位置）对应的出错步进行检查、修改。		
4210	[CAN'T EXECUTE(P)] 执行了 CALL 指令但跳转目标指针不存在。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时			
4211	[CAN'T EXECUTE(P)] 执行后的子程序中 RET 指令不存在。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时			
4212	[CAN'T EXECUTE(P)] 在主程序的 FEND 指令的前面存在有 RET 指令。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时			
4213	[CAN'T EXECUTE(P)] 嵌套超过了 16 级。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时	将嵌套置为 16 级以内。		

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
4220	[CAN'T EXECUTE(I)] 发生了中断输入，但对应的中断指针不存在。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时	通过外围设备读取出错的公共信息，对与该数值（程序出错位置）对应的出错步进行检查、修改。	RUN：熄灯 ERR.：闪烁 CPU 状态： 停止	QCPU
4221	[CAN'T EXECUTE(I)] 执行的中断程序中 IRET 指令不存在。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时			
4223	[CAN'T EXECUTE(I)] 主程序的 FEND 指令的前面存在有 IRET 指令。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时			
	[CAN'T EXECUTE(I)] ・在恒定周期执行型程序中执行了 IRET 指令。 ・在恒定周期执行型程序中执行了 STOP 指令。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时			QnU
4225	[CAN'T EXECUTE(P)] 在冗余系统中，以安装在扩展基板上的模块为对象设置了中断指针。 ■附加信息 ・公共信息： - ・个别信息： - ■诊断时机 ・电源 ON 时 / 复位时	由于以安装在扩展基板上的模块为对象的中断指针不能使用，因此将该设置删除。		QnPRH*6
4230	[INST. FORMAT ERR.] CHK 指令与 CHKEND 指令的个数未呈 1 对 1 状态。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时	通过外围设备读取出错的公共信息，对与该数值（程序出错位置）对应的出错步进行检查、修改。		Qn(H) QnPH

*6 以序列号的前 5 位数为 “09012” 以后的模块为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
4231	[INST. FORMAT ERR.] IX 指令与 IXEND 指令的个数未呈 1 对 1 状态。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时	通过外围设备读取出错的公共信息，对与该数值（程序出错位置）对应的出错步进行检查、修改。	RUN：熄灯 ERR.：闪烁 CPU 状态： 停止	QCPU
4235	[INST. FORMAT ERR.] CHK 指令的检查条件的构成不正确。 或者，在低速执行型程序内执行了 CHK 指令。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时			Qn(H) QnPH
4350	[MULTI-COM. ERROR] ・程序中指定的多 CPU 高速通信专用指令中指定的对象目标 CPU 模块有误。或者不是多 CPU 高速通信专用指令可使用的设置。 ・指定了已进行了预约设置的 CPU 号。 ・指定了未安装的 CPU 号。 ・对象目标 CPU 模块的起始 I/O 编号÷ 16(n1) 超出了 3E0H ～ 3E3H 的范围。 ・指定了不能执行指令的 CPU 模块。 ・在单 CPU 系统中执行了指令。 ・指定了自站 CPU。 ・在设置为“不使用多 CPU 高速通信功能”的情况下执行了多 CPU 高速通信专用指令。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时			QnU*7
4351	[MULTI-COM. ERROR] ・程序中指定的多 CPU 高速通信专用指令在指定的对象目标 CPU 模块中不能执行。 ・指令名有错误。 ・指定了在对象目标 CPU 模块中不支持的指令。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时			
4352	[MULTI-COM. ERROR] 程序中指定的多 CPU 高速通信专用指令的软元件数有误。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时			

*7 以除 Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU 以外的通用型 QCPU 为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
4353	[MULTI-COM. ERROR] 指定了程序中指定的多 CPU 高速通信专用指令不能使用的软元件。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时	通过外围设备读取出错的公共信息，对与该数值（程序出错位置）对应的出错步进行检查、修改。		QnU*7
4354	[MULTI-COM. ERROR] 指定了程序中指定的多 CPU 高速通信专用指令不能处理的字符串。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时			
4355	[MULTI-COM. ERROR] 程序中指定的多 CPU 高速通信专用指令的读取 / 写入数据数（请求 / 接收数据数）不是有效值。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时			
4400	[SFCP. CODE ERROR] SFC 程序中没有 SFCP 指令及 SFCPEND 指令。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・STOP → RUN 时	通过 GX Developer 重新将程序写入到 CPU 模块中。	RUN：熄灯 ERR.：闪烁 CPU 状态： 停止	Qn(H) QnPH QnPRH
4410	[CAN'T SET(BL)] SFC 程序中指定的块号超过最大设置值。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・STOP → RUN 时			Q00J/Q00/ Q01 *2 Qn(H) QnPH QnPRH QnU
4411	[CAN'T SET(BL)] SFC 程序中指定的块号重复。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・STOP → RUN 时			
4420	[CAN'T SET(S)] SFC 程序中指定的步号超过最大设置值。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・STOP → RUN 时			

*2 以功能版本 B 以后为对象。
*7 以除 Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU 以外的通用型 QCPU 为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
4421	[CAN'T SET(S)] SFC 程序的总步数的总数超过了最大值。 ■附加信息 • 公共信息： 程序出错位置 • 个别信息： - ■诊断时机 • STOP → RUN 时	通过 GX Developer 重新将程序写入到 CPU 模块中。		Q00J/Q00/ Q01 *2 Qn(H) QnPH QnPRH QnU
4422	[CAN'T SET(S)] SFC 程序中指定的步号重复。 ■附加信息 • 公共信息： 程序出错位置 • 个别信息： - ■诊断时机 • STOP → RUN 时			
4423	[CAN'T SET(S)] 各程序的（最大步号 +1）的合计超过了步进继电器的总数。 ■附加信息 • 公共信息： 程序出错位置 • 个别信息： - ■诊断时机 • STOP → RUN 时	进行修改，使各程序的（最大步号 +1）的合计不超过步进继电器的总数。	RUN：熄灯 ERR.：闪烁 CPU 状态： 停止	Q00J/Q00/ Q01*2 QnU
4430	[SFC EXE. ERROR] 无法执行 SFC 程序。 • 块信息设置的内容非法。 • 块信息设置的 SFC 信息软元件超出了可编程控制器参数中设置的软元件设置范围。 ■附加信息 • 公共信息： 文件名 / 驱动器名 • 个别信息： - ■诊断时机 • STOP → RUN 时	• 通过 GX Developer 重新将程序写入到 CPU 模块中。 • 对 SFC 信息软元件的设置进行修改后，写入到 CPU 模块中。 • 对可编程控制器参数中设置的软元件设置范围进行修改后，写入到 CPU 模块中。		
4431	[SFC EXE. ERROR] 无法执行 SFC 程序。 • 块参数的设置异常。 ■附加信息 • 公共信息： 文件名 / 驱动器名 • 个别信息： - ■诊断时机 • STOP → RUN 时	通过 GX Developer 重新将程序写入到 CPU 模块中。		
4432	[SFC EXE. ERROR] 无法执行 SFC 程序。 • SFC 程序的构成非法。 ■附加信息 • 公共信息： 文件名 / 驱动器名 • 个别信息： - ■诊断时机 • STOP → RUN 时			

*2　以功能版本 B 以后为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
4500	[SFCP. FORMAT ERR.] SFC 程序中 BLOCK 指令及 BEND 指令的个数未呈 1 对 1 状态。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・STOP → RUN 时	通过外围设备重新将程序写入到 CPU 模块中。	RUN：熄灯 ERR.：闪烁 CPU 状态： 停止	Qn(H) QnPH QnPRH
4501	[SFCP. FORMAT ERR.] SFC 程序中 STEP* ～ TRAN* ～ TSET ～ SEND 指令的构成不正确。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・STOP → RUN 时			
4502	[SFCP. FORMAT ERR.] SFC 程序的构成非法。 ・SFC 程序的块内没有 STEPI* 指令。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・STOP → RUN 时			Q00J/Q00/Q01*2 Qn(H) QnPH QnPRH QnU
4503	[SFCP. FORMAT ERR.] SFC 程序的构成非法。 ・TSET 指令中指定地方步不存在。 ・在跳转移位中，对指定目标步号指定了自身步号。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・STOP → RUN 时	・通过外围设备重新将程序写入到 CPU 模块中。 ・通过外围设备读取出错的公共信息，对与该数值（程序出错位置）对应的出错步进行检查、修改。		
4504	[SFCP. FORMAT ERR.] SFC 程序的构成非法。 ・TAND 指令中指定的步不存在。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・STOP → RUN 时	通过外围设备重新将程序写入到 CPU 模块中。		
4505	[SFCP. FORMAT ERR.] SFC 程序的构成非法。 ・在步的动作输出中，对自身步指定了 SET Sn/BLmSn、RST Sn/BLmSn 指令。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・STOP → RUN 时	通过 GX Developer 读取出错的公共信息，对与该数值（程序出错位置）对应的出错步进行检查、修改。		Q00J/Q00/Q01*2 QnU

*2 以功能版本 B 以后为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
4506	[SFCP. FORMAT ERR.] SFC 程序的构成非法。 ・在复位步中，在指定目标步中指定了自身步号。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・STOP → RUN 时	通过 GX Developer 读取出错的公共信息，对与该数值（程序出错位置）对应的出错步进行检查、修改。	RUN：熄灯 ERR.：闪烁 CPU 状态： 停止	Q00J/Q00/Q01*2 QnU
4600	[SFCP. OPE. ERROR] SFC 程序中包含有无法处理的数据。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时	通过外围设备读取出错的公共信息，对与该数值（程序出错位置）对应的出错步进行检查、修改。	RUN： 熄灯 / 亮灯 ERR.： 闪烁 / 亮灯 CPU 状态： 停止 / 继续运行*1	Qn(H) QnPH QnPRH
4601	[SFCP. OPE. ERROR] 超出了 SFC 程序可指定的范围。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时			
4602	[SFCP. OPE. ERROR] 在 SFC 程序的块控制中在开始指令的前面执行了结束指令。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时			
4610	[SFCP. EXE. ERROR] 通过 SFC 程序进行继续运行启动时的激活步信息不正确。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・STOP → RUN 时	通过外围设备读取出错的公共信息，对与该数值（程序出错位置）对应的出错步进行检查、修改。 程序将自动进行初始化启动。	RUN：亮灯 ERR.：亮灯 CPU 状态： 继续运行	
4611	[SFCP. EXE. ERROR] 通过 SFC 程序指定继续运行启动时，在 RUN 过程中被复位。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・STOP → RUN 时			
4620	[BLOCK EXE. ERROR] 通过 SFC 程序对已经启动的块执行了启动操作。 ■附加信息 ・公共信息： 程序出错位置 ・个别信息： - ■诊断时机 ・执行指令时	通过外围设备读取出错的公共信息，对与该数值（程序出错位置）对应的出错步进行检查、修改。	RUN：熄灯 ERR.：闪烁 CPU 状态： 停止	

*1　发生出错时的 CPU 的动作状态可以在参数中进行设置。(LED 显示也将联动变化。)
*2　以功能版本 B 以后为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
4621	[BLOCK EXE. ERROR] SFC 程序中对不存在的块执行了启动。 **■附加信息** ・公共信息： 程序出错位置 ・个别信息： - **■诊断时机** ・执行指令时	・通过外围设备读取出错的公共信息，对与该数值（程序出错位置）对应的出错步进行检查、修改。 ・如果特殊继电器 SM321 为 OFF 状态则将其置为 ON。		Q00J/Q00/Q01*2 Qn(H) QnPH QnPRH QnU
4630	[STEP EXE. ERROR] 通过 SFC 程序对已启动的步执行了启动。 **■附加信息** ・公共信息： 程序出错位置 ・个别信息： - **■诊断时机** ・执行指令时	通过外围设备读取出错的公共信息，对与该数值（程序出错位置）对应的出错步进行检查、修改。		Qn(H) QnPH QnPRH
4631	[STEP EXE. ERROR] ・在 SFC 程序中对不存在的步执行了启动。 或者在 SFC 程序中对不存在的步进行了结束指定。 ・在 SFC 程序中对不存在的转移条件进行了强制转移。 或者在 SFC 程序中对不存在的转移条件的强制转移进行了解除。 **■附加信息** ・公共信息： 程序出错位置 ・个别信息： - **■诊断时机** ・执行指令时	・通过外围设备读取出错的公共信息，对与该数值（程序出错位置）对应的出错步进行检查、修改。 ・如果特殊继电器 SM321 为 OFF 状态则将其置为 ON。	RUN：熄灯 ERR.：闪烁 CPU 状态： 停止	Q00J/Q00/Q01*2 Qn(H) QnPH QnPRH QnU
4632	[STEP EXE. ERROR] 在 SFC 程序中可指定的块的同时激活步数溢出。 **■附加信息** ・公共信息： 程序出错位置 ・个别信息： - **■诊断时机** ・执行指令时	通过外围设备读取出错的公共信息，对与该数值（程序出错位置）对应的出错步进行检查、修改。		Qn(H) QnPH QnPRH QnU
4633	[STEP EXE. ERROR] 可指定的所有同时激活步数溢出。 **■附加信息** ・公共信息： 程序出错位置 ・个别信息： - **■诊断时机** ・执行指令时			

*2　以功能版本 B 以后为对象。

12.1.7 出错代码一览表 (5000 ～ 5999)

以下介绍出错代码 5000 ～ 5999 的出错信息、异常内容及原因以及处理方法。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
5000	[WDT ERROR] ・初始执行型程序的扫描时间超出了在可编程控制器参数的可编程控制器 RAS 设置中设置的初始执行监视时间。 **■附加信息** ・公共信息：时间（设置值） ・个别信息：时间（实测值） **■诊断时机** ・常时	・通过外围设备读取出错的个别信息，对该数值（时间）进行检查，缩短扫描时间。 ・在可编程控制器参数的可编程控制器 RAS 设置中对初始执行监视时间或者 WDT 设置值进行变更。 ・通过跳转移位解除无限循环。	RUN：熄灯 ERR.：闪烁 CPU 状态：停止	Qn(H) QnPH QnPRH QnU
	[WDT ERROR] ・发生了待机系统的电源 OFF。 ・在未进行待机系统的电源 OFF 或者复位的状态下，对热备电缆进行了插拔。 ・热备电缆未用连接器固定螺栓固定。 **■附加信息** ・公共信息：时间（设置值） ・个别信息：时间（实测值） **■诊断时机** ・常时	・如果发生了待机系统的电源 OFF，控制系统的扫描时间将被延迟，因此应在考虑控制系统的扫描时间延迟的基础上对 WDT 设置值重新进行设置。 ・在运行过程中热备电缆脱落时，将热备电缆切实安装后执行再启动。仍然显示相同的出错时，可能是热备电缆或 CPU 模块的硬件故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		QnPRH
5001	[WDT ERROR] ・程序的扫描时间超过了可编程控制器参数的可编程控制器 RAS 设置中设置的 WDT 设置值。 **■附加信息** ・公共信息：时间（设置值） ・个别信息：时间（实测值） **■诊断时机** ・常时	・通过外围设备读取出错的个别信息，对该数值（时间）进行检查，缩短扫描时间。 ・在可编程控制器参数的可编程控制器 RAS 设置中对初始执行监视时间或者 WDT 设置值进行变更。 ・通过跳转移位解除无限循环。		QCPU
	[WDT ERROR] ・发生了待机系统的电源 OFF。 ・在未进行待机系统的电源 OFF 或者复位的状态下，对热备电缆进行了插拔。 ・热备电缆未用连接器固定螺栓固定。 **■附加信息** ・公共信息：时间（设置值） ・个别信息：时间（实测值） **■诊断时机** ・常时	・如果发生了待机系统的电源 OFF，控制系统的扫描时间将被延迟，因此应在考虑控制系统的扫描时间延迟的基础上对 WDT 设置值重新进行设置。 ・在运行过程中热备电缆脱落时，将热备电缆切实安装后执行再启动。仍然显示相同的出错时，可能是热备电缆或 CPU 模块的硬件故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		QnPRH

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
5010	[PRG. TIME OVER] ・程序扫描时间超出了可编程控制器参数的可编程控制器 RAS 设置中设置的恒定扫描设置时间。 **■附加信息** ・公共信息：时间（设置值） ・个别信息：时间（实测值） **■诊断时机** ・常时	・对恒定扫描设置时间进行重新审核。 ・为了充分确保恒定扫描的剩余时间，应对可编程控制器参数的恒定扫描时间、低速程序执行时间进行重新审核。	RUN：亮灯 ERR.：亮灯 CPU 状态： 继续运行	Qn(H) QnPH QnPRH QnU
	[PRG. TIME OVER] ・可编程控制器参数的可编程控制器 RAS 设置中设置的低速程序执行时间超出了恒定扫描的剩余时间。 **■附加信息** ・公共信息：时间（设置值） ・个别信息：时间（实测值） **■诊断时机** ・常时			Qn(H) QnPH QnPRH
	[PRG. TIME OVER] 程序的扫描时间超出了可编程控制器参数的可编程控制器 RAS 设置中设置的恒定扫描设置时间。 **■附加信息** ・公共信息：时间（设置值） ・个别信息：时间（实测值） **■诊断时机** ・常时	・为了充分确保恒定扫描的剩余时间，应对可编程控制器参数的恒定扫描时间进行重新审核。		Q00J/Q00/Q01
5011	[PRG. TIME OVER] 低速执行型程序的扫描时间超出了可编程控制器参数的可编程控制器 RAS 设置中设置的低速执行监视时间。 **■附加信息** ・公共信息：时间（设置值） ・个别信息：时间（实测值） **■诊断时机** ・常时	通过外围设备读取出错的个别信息，对该数值（时间）进行检查，缩短扫描时间。 在可编程控制器参数的可编程控制器 RAS 设置中对低速执行监视时间进行变更。		Qn(H) QnPH

12.1.8 出错代码一览表 (6000 ～ 6999)

以下介绍出错代码 6000 ～ 6999 的出错信息、异常内容及原因以及处理方法。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
6000	[FILE DIFF.] 在冗余系统中，控制系统与待机系统的程序、参数不相同。 对于在两个系统中检测出的不相同的文件的类型，可以通过出错公共信息的文件名进行确认。 ・程序不相同。 （文件名 = ********.QPG） ・可编程控制器参数 / 网络参数 / 冗余参数不相同。 （文件名 = PARAM.QPA） ・远程口令不相同。 （文件名 = PARAM.QPA） ・智能功能模块参数不相同。 （文件名 = IPARAM.QPA） ・软元件初始值不相同。 （文件名 = ********.QDI） ・多个块 RUN 中写入允许区的容量不相同。 （文件名 = MBOC.QMB） （只有在冗余系统的待机系统中可以检测出） **■附加信息** ・公共信息：文件名 ・个别信息：- **■诊断时机** ・电源 ON 时 / 复位时 / 连接热备电缆时 / 变更为备份模式时 /RUN 中写入结束时 / 系统切换时 / 切换为两个系统 RUN 时	・使控制系统与待机系统的程序和参数相同。 ・按照下述 1）或 2）的步骤执行可编程控制器校验，以确认两个系统的文件之间的差异，然后改正错误的文件，并对改正后的文件重新进行可编程控制器写入。 1）使用 GX Developer 或者 PX Developer 读取 A 系统的程序・参数后，与 B 系统的程序・参数进行校验。 2）将离线环境下保存的 GX Developer 或者 PX Developer 的程序・参数与写入到两个系统的 CPU 模块中的程序・参数进行校验。 ・在两个系统的可用于多个块 RUN 中写入的区域容量不相同时，采取下述 1）或 2）的步骤进行处理。 1）使用从控制系统至待机系统的存储器拷贝功能，将控制系统的程序存储器的内容拷贝到待机系统中。 2）对两个系统的 CPU 模块的程序存储器进行格式化。（将两个系统的可用于多个块 RUN 中写入的区域容量设置为相同的值。）	RUN：熄灯 ERR.：闪烁 CPU 状态：停止	QnPRH
6001	[FILE DIFF] 在冗余系统中，插杆开关的参数有效驱动器的设置（SW2、SW3）不相同。 **■附加信息** ・公共信息：- ・个别信息：- **■诊断时机** ・电源 ON 时 / 复位时 / 连接热备电缆时 / 变更为备份模式时	将控制系统与待机系统的插杆开关的参数有效驱动器的设置（SW2、SW3）设置为相同。		
6010	[OPE. MODE DIFF.] 在冗余系统中，控制系统与待机系统的运行状态不相同。 （只有在冗余系统的待机系统中才可以检测出此出错。） **■附加信息** ・公共信息：- ・个别信息：- **■诊断时机** ・常时	将控制系统与待机系统的运行状态置为相同。	RUN：亮灯 ERR.：亮灯 CPU 状态： 继续运行	

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
6020	[OPE. MODE DIFF.] 电源 ON 时 / 复位时，冗余系统中控制系统与待机系统的 RUN/STOP 开关的状态不相同。 （在冗余系统的控制系统或待机系统中可以检测出此出错。） **■附加信息** ・公共信息：－ ・个别信息：－ **■诊断时机** ・电源 ON 时 / 复位时	将控制系统与待机系统的 RUN/STOP 开关的状态置为相同。	RUN：熄灯 ERR.：闪烁 CPU 状态：停止	QnPRH
6030	[UNIT LAY. DIFF.] ・在冗余系统中，控制系统与待机系统的模块安装配置不相同。 ・两个系统的网络模块的模式设置不相同。 （在冗余系统的控制系统或待机系统中可以检测出此出错。） **■附加信息** ・公共信息：模块号 ・个别信息：－ **■诊断时机** ・电源 ON 时 / 复位时 / 连接热备电缆时 / 变更运行模式时	・使控制系统与待机系统的模块安装状态一致。 ・在网络参数的冗余设置中，使 B 系统的模式设置与 A 系统的设置相同。		
6035	[[UNIT LAY. DIFF.] 在冗余系统中，控制系统与待机系统的 CPU 模块的型号不相同。 （只有在冗余系统的待机系统中可以检测出） **■附加信息** ・公共信息：－ ・个别信息：－ **■诊断时机** ・电源 ON 时 / 复位时 / 连接热备电缆时 / 变更运行模式时	使控制系统与待机系统的 CPU 模块型号一致。		
6036	[UNIT LAY. DIFF.] 在冗余系统的控制系统与待机系统之间，检测出 MELSECNET/H 多重远程 I/O 网络的远程 I/O 配置不相同。 （在冗余系统的控制系统或待机系统中可以检测出此出错。） **■附加信息** ・公共信息：模块号 ・个别信息：－ **■诊断时机** ・常时	确认 MELSECNET/H 多重远程 I/O 网络的网络电缆是否发生了断开。		
6040	[CARD TYPE DIFF.] 在冗余系统中，控制系统与待机系统的存储卡的安装有无状态不相同。 **■附加信息** ・公共信息：－ ・个别信息：－ **■诊断时机** ・电源 ON 时 / 复位时	使控制系统与待机系统的存储卡的安装有无状态相同。		

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
6041	[CARD TYPE DIFF.] 在冗余系统中控制系统与待机系统的存储卡类型不相同。 **■附加信息** ・公共信息：－ ・个别信息：－ **■诊断时机** ・电源 ON 时 / 复位时	使控制系统与待机系统的存储卡类型相同。	RUN：熄灯 ERR.：闪烁 CPU 状态：停止	QnPRH
6050	[CAN'T EXE. MODE] 执行了在调试模式及运行模式（备份 / 分开）下不能执行的功能。 （在冗余系统的控制系统或待机系统中可以检测出此出错。） **■附加信息** ・公共信息：－ ・个别信息：－ **■诊断时机** ・常时	执行在调试模式及运行模式（备份 / 分开）下可以执行的功能。	RUN：亮灯 ERR.：亮灯 CPU 状态： 继续运行	
6060	[CPU MODE DIFF.] 在冗余系统中，控制系统与待机系统的运行模式（备份 / 分开）不相同。（只有在冗余系统的待机系统中可以检测出此出错。） **■附加信息** ・公共信息：－ ・个别信息：－ **■诊断时机** ・电源 ON 时 / 复位时 / 连接热备电缆时	将控制系统与待机系统的运行模式置为相同。	RUN：熄灯 ERR.：闪烁 CPU 状态：停止	
6061	[CPU MODE DIFF.] 在冗余系统中，控制系统与待机系统的运行模式（备份 / 分开）不相同。（只有在冗余系统的待机系统中可以检测出此出错。） **■附加信息** ・公共信息：－ ・个别信息：－ **■诊断时机** ・执行 END 指令时			
6062	[CPU MODE DIFF.] A 系统与 B 系统的系统状态相同（A 系统与 B 系统均为控制系统）。 （在冗余系统的 B 系统中可以检测出此出错。） **■附加信息** ・公共信息：－ ・个别信息：－ **■诊断时机** ・电源 ON 时 / 复位时 / 连接热备电缆时	对处于停止出错状态的 CPU 模块（B 系统）进行电源 OFF → ON 操作。		
6100	[TRK. TRANS. ERR.] ・热备发送出错（重试次数溢出等）。 （热备电缆的脱落、其它系统电源断电（复位也包括在内）也可能导致发生此出错。） ・由于未遵守冗余系统的启动步骤，因此启动时变为出错状态。 **■附加信息** ・公共信息：热备通信中的通信数据类型 ・个别信息：－ **■诊断时机** ・常时	・对 CPU 模块或者热备电缆进行检查。如果仍然显示相同的出错，则可能是 CPU 模块或者热备电缆的硬件故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。） ・确认冗余系统启动步骤后，执行再启动。	RUN：亮灯 ERR.：亮灯 CPU 状态： 继续运行	

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
6101	[TRK. TRANS. ERR.] ・热备发送中发生超时出错。 （热备电缆脱落、其它系统电源断电（复位也包括在内）也可能导致发生此出错。） ・由于未遵守冗余系统的启动步骤，因此启动时变为出错状态。 （在冗余系统的控制系统或待机系统中可以检测出此出错。） **■附加信息** ・公共信息： 热备通信中的通信数据类型 ・个别信息： - **■诊断时机** ・常时	・对 CPU 模块或者热备电缆进行检查。如果仍然显示相同的出错，则可能是 CPU 模块或者热备电缆的硬件故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。） ・确认冗余系统启动步骤后，执行再启动。	RUN：亮灯 ERR.：亮灯 CPU 状态： 继续运行	QnPRH
6102	[TRK. TRANS. ERR.] 热备接收中发生了数据的和数值出错。 （在冗余系统的控制系统或待机系统中可以检测出此出错。） **■附加信息** ・公共信息： - ・个别信息： - **■诊断时机** ・常时			
6103	[TRK. TRANS. ERR.] ・热备接收中发生了数据出错（和数值以外）。 （热备电缆的脱落、其它系统电源断电（复位也包括在内）也可能导致发生此出错。） ・由于未遵守冗余系统的启动步骤，因此启动时变为出错状态。 （在冗余系统的控制系统或待机系统中可以检测出此出错。） **■附加信息** ・公共信息： - ・个别信息： - **■诊断时机** ・常时			
6105	[TRK. TRANS. ERR.] ・热备发送出错（重试次数溢出等）。 （热备电缆的脱落、其它系统电源断电（复位也包括在内）也可能导致发生此出错。） ・由于未遵守冗余系统的启动步骤，因此启动时变为出错状态。 （在冗余系统的控制系统或待机系统中可以检测出此出错。） **■附加信息** ・公共信息： 热备通信中的通信数据类型 ・个别信息： - **■诊断时机** ・常时			

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
6106	[TRK. TRANS. ERR.] ・热备发送中发生超时出错。 （热备电缆脱落、其它系统电源断电（复位也包括在内）也可能导致发生此出错。） ・由于未遵守冗余系统的启动步骤，因此启动时变为出错状态。 （在冗余系统的控制系统或待机系统中可以检测出此出错。） **■附加信息** ・公共信息：热备通信中的通信数据类型 ・个别信息：－ **■诊断时机** ・常时	・对 CPU 模块或者热备电缆进行检查。如果仍然显示相同的出错，则可能是 CPU 模块或者热备电缆的硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。） ・确认冗余系统启动步骤后，执行再启动。	RUN：亮灯 ERR.：亮灯 CPU 状态： 继续运行	QnPRH
6107	[TRK. TRANS. ERR.] 热备接收中发生了数据的和数值出错。 （在冗余系统的控制系统或待机系统中可以检测出此出错。） **■附加信息** ・公共信息：－ ・个别信息：－ **■诊断时机** ・常时			
6108	[TRK. TRANS. ERR.] ・热备接收中发生了数据出错（和数值以外）。 （热备电缆的脱落、其它系统电源断电（复位也包括在内）也可能导致发生此出错。） ・由于未遵守冗余系统的启动步骤，因此启动时变为出错状态。 （在冗余系统的控制系统或待机系统中可以检测出此出错。） **■附加信息** ・公共信息：－ ・个别信息：－ **■诊断时机** ・常时			
6110	[TRK. SIZE ERROR] 热备容量超出了允许范围。 （在冗余系统的控制系统或待机系统中可以检测出此出错。） **■附加信息** ・公共信息：热备容量超过的出错原因 ・个别信息：－ **■诊断时机** ・执行 END 指令时	对热备容量进行重新审核。		
6111	[TRK. SIZE ERROR] 控制系统的文件寄存器容量少于热备设置的文件寄存器的点数。 （在冗余系统的控制系统或待机系统中可以检测出此出错。） **■附加信息** ・公共信息：－ ・个别信息：－ **■诊断时机** ・执行 END 指令时	切换为热备设置的文件寄存器容量以上的文件寄存器。		

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
6112	[TRK. SIZE ERROR] 从控制系统热备发送了超过待机系统的文件寄存器容量的文件。 （只有在冗余系统的待机系统中可以检测出此出错。） **■附加信息** ・公共信息：－ ・个别信息：－ **■诊断时机** ・执行 END 指令时	切换为热备设置的文件寄存器容量以上的文件寄存器。	RUN：亮灯 ERR.：亮灯 CPU 状态： 继续运行	QnPRH
6120	[TRK. CABLE ERR.] ・在未安装热备电缆的状态下进行了启动。 ・在热备电缆故障的状态下进行了启动。 ・由于 CPU 模块侧的热备通信硬件故障，因此无法经由热备电缆与其它系统通信。 （在冗余系统的控制系统或待机系统中可以检测出此出错。） **■附加信息** ・公共信息：－ ・个别信息：－ **■诊断时机** ・电源 ON 时 / 复位时	安装了热备电缆后再进行启动。如果仍然显示相同的出错，则可能是热备电缆或者 CPU 模块侧的热备通信硬件故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）	RUN：熄灯 ERR.：闪烁 CPU 状态：停止	
6130	[TRK. DISCONNECT] ・热备电缆脱落。 ・运行过程中热备电缆故障。 ・ CPU 模块侧的热备通信硬件发生了故障。 （在冗余系统的控制系统或待机系统中可以检测出此出错。） **■附加信息** ・公共信息：－ ・个别信息：－ **■诊断时机** ・常时	・热备电缆脱落时，将热备电缆安装到两个系统 CPU 模块的连接器上。 ・如果将热备电缆安装到两个系统 CPU 模块的连接器上并进行出错解除后，仍然无法解除出错时，则可能是热备电缆或者 CPU 模块侧的热备通信硬件发生故障。 （请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）	RUN：亮灯 ERR.：亮灯 CPU 状态： 继续运行	
6140	[TRK.INIT. ERROR] ・在电源 ON 时 / 复位时与其它系统进行初始化通信的过程中，从其它系统未返回响应。 ・由于未遵守冗余系统的启动步骤，因此启动时变为出错状态。 （在冗余系统的控制系统或待机系统中可以检测出此出错。） **■附加信息** ・公共信息：－ ・个别信息：－ **■诊断时机** ・电源 ON 时 / 复位时	・对检测出出错的 CPU 模块再次进行电源 OFF → ON 或者复位→复位解除。 如果仍然显示相同的出错，则可能是 CPU 模块故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。） ・确认冗余系统启动步骤后，执行再启动。	RUN：熄灯 ERR.：闪烁 CPU 状态：停止	

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
6200	[CONTROL EXE.] 在冗余系统中，待机系统被切换为控制系统。（检测出待机系统→控制系统的 CPU 模块） 由于本出错代码不是 CPU 模块的异常信息，而是表示状态的内容，因此出错代码及存储信息不被存储到 SD0 ～ 26 中，而是在每次发生系统切换时被存储到出错履历中。 （应通过 GX Developer 读取出错履历中的出错信息，并加以确认。） **■附加信息** ・公共信息： 系统切换原因 ・个别信息： - **■诊断时机** ・常时	——	RUN：亮灯 ERR.：熄灯 CPU 状态： 无出错	QnPRH
6210	[STANDBY] 在冗余系统中，控制系统被切换为待机系统。（检测出控制系统→待机系统的 CPU 模块） 由于本出错代码不是 CPU 模块的异常信息，而是表示状态的内容，因此出错代码及存储信息不被存储到 SD0 ～ 26 中，而是在每次发生系统切换时被存储到出错履历中。 （应通过 GX Developer 读取出错履历中的出错信息，并加以确认。） **■附加信息** ・公共信息： 系统切换原因 ・个别信息： - **■诊断时机** ・常时			
6220	[CAN'T SWITCH] 由于待机系统异常・热备电缆异常・正在分开模式下进行在线模块更换导致不能执行系统切换。在控制系统中系统切换原因如下所示。 ・由于 SP.CONTSW 导致的系统切换 ・来自于网络模块的系统切换请求 **■附加信息** ・公共信息： 系统切换原因 ・个别信息： 系统切换失败原因 **■诊断时机** ・执行切换时	・确认待机系统的状态后，对出错进行解除。 ・结束在线模块更换。	RUN：亮灯 ERR.：亮灯 CPU 状态： 继续运行	
6300	[STANDBY SYS. DOWN] 备份模式时检测出以下异常。 ・冗余系统中待机系统未启动。 ・冗余系统中待机系统处于停止出错状态。 ・在运行状态下的控制系统中，连接了调试模式的 CPU 模块。 （只有在冗余系统的控制系统中可以检测出此出错。） **■附加信息** ・公共信息： - ・个别信息： - **■诊断时机** ・常时	・确认待机系统的电源是否接通，如果未接通，则接通电源。 ・确认待机系统是否处于复位状态，，如果处于复位状态，则进行复位解除。 ・确认待机系统是否处于停止出错状态，如果处于出错状态，则消除出错原因后，执行再启动。 ・在备份模式下处于运行状态的控制系统中连接调试模式的 CPU 模块时，应在确保控制系统与待机系统的组合正确的状况下进行连接。		

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
6310	[CONTROL SYS. DOWN] 在备份模式时检测出以下异常。 ・冗余系统中控制系统未启动。 ・冗余系统中控制系统处于停止出错状态。 ・在运行状态下的待机系统中，连接了调试模式的 CPU 模块。 ・由于未遵守冗余系统的启动步骤，启动时变为出错状态。 （只有在冗余系统的待机系统中可以检测出此出错。） **■附加信息** ・公共信息：－ ・个别信息：－ **■诊断时机** ・常时	・待机系统存在，但是控制系统不存在。 ・确认除待机系统以外的其它系统电源是否接通，如果未接通，则接通电源。 ・确认除待机系统以外的其它系统是否处于复位状态，，如果处于复位状态，则进行复位解除。 ・确认除待机系统以外的其它系统是否处于停止出错状态，如果处于出错状态，则消除出错原因，使控制系统与待机系统的动作状态一致后，执行再启动。 ・在备份模式下处于运行状态的待机系统中连接调试模式的 CPU 模块时，应在确保控制系统与待机系统的组合正确的状况下进行连接。 ・确认冗余系统的启动步骤后执行再启动。		QnPRH
6311 6312	[CONTROL SYS. DOWN] ・在冗余系统中，由于未从控制系统发送一致性检查数据，因此无法作为待机系统启动。 ・由于未遵守冗余系统的启动步骤，启动时变为出错状态。 （只有在冗余系统的待机系统中可以检测出此出错。） **■附加信息** ・公共信息：－ ・个别信息：－ **■诊断时机** ・电源 ON 时 / 复位时	・更换热备电缆。如果仍然显示相同的出错，则可能是 CPU 模块的硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。） ・确认冗余系统的启动步骤后执行再启动。	RUN：熄灯 ERR.：闪烁 CPU 状态：停止	
6313	[CONTROL SYS. DOWN] 在冗余系统中，控制系统检测出系统配置异常，且通知了待机系统（自系统）。 **■附加信息** ・公共信息：－ ・个别信息：－ **■诊断时机** ・电源 ON 时 / 复位时	确认基板之间的连接以及系统配置（模块类型、个数、参数）正常后，执行再启动。		QnPRH*1
6400	[PRG. MEM. CLEAR] 被执行了从控制系统至待机系统的存储器拷贝功能，程序存储器被清除。 **■附加信息** ・公共信息：－ ・个别信息：－ **■诊断时机** ・执行从控制系统至待机系统的存储器拷贝功能时	从控制系统至待机系统的存储器拷贝功能结束后进行电源的 OFF → ON，或者复位操作。		QnPRH
6410	[MEM. COPY EXE.] 被执行了从控制系统至待机系统的存储器拷贝功能。 （只有在冗余系统的控制系统中可以检测出此出错。） **■附加信息** ・公共信息：－ ・个别信息：－ **■诊断时机** ・执行从控制系统至待机系统的存储器拷贝功能时	——	RUN：亮灯 ERR.：亮灯 CPU 状态： 继续运行	

*1　以序列号的前 5 位数为“09012”以后的模块为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
6500	[TRK. PARA. ERROR] 可编程控制器参数的热备设置中指定的文件寄存器文件不存在。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时	通过 GX Developer 读取出错的个别信息，对驱动器名、文件名进行检查、修改。创建指定的文件。	RUN: 熄灯 ERR.：闪烁 CPU 状态： 停止	QnPRH
6501	[TRK. PARA. ERROR] 可编程控制器参数的热备设置的软元件详细设置中指定的文件寄存器的范围超出了指定的文件寄存器容量。 **■附加信息** ・公共信息： 文件名 / 驱动器名 ・个别信息： 参数号 **■诊断时机** ・电源 ON 时 / 复位时	通过 GX Developer 读取出错的个别信息，增加文件寄存器的容量。		

12.1.9 出错代码一览表 (7000 ～ 7999)

以下介绍出错代码 7000 ～ 7999 的出错信息、异常内容及原因以及处理方法。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
7000	[MULTI CPU DOWN] ・在多 CPU 系统中，在动作模式为“选择了系统停止”的 CPU 模块中发生了异常。 ・在多 CPU 系统中，安装了不支持多 CPU 系统的 CPU 模块。 ・在运行过程中将除 1 号机以外的 CPU 模块从基板上卸下。或者对除 1 号机以外的 CPU 模块进行了复位。 **■附加信息** ・公共信息： 模块号 (CPU 号) ・个别信息： - **■诊断时机** ・常时	・通过 GX Developer 读取出错的个别信息，对异常的 CPU 模块的出错进行确认后，消除出错。 ・将不支持多 CPU 系统的 CPU 模块从主基板上卸下。 ・对除 1 号机以外的 CPU 模块的安装状态、复位有无进行确认。		Q00/Q01*1 Qn(H)*1 QnPH QnU*6
	[MULTI CPU DOWN] 在多 CPU 系统中，电源 ON 时由于 1 号机发生停止出错导致其它机号的 CPU 无法启动。(2 ～ 4 号机中发生) **■附加信息** ・公共信息： 模块号 (CPU 号) ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时	通过 GX Developer 读取出错的个别信息，对异常的 CPU 模块的出错进行确认后，消除出错。		
7002	[MULTI CPU DOWN] ・多 CPU 系统的初始化通信时，初始化通信对象的 CPU 模块未返回响应信息。 ・在多 CPU 系统中，安装了不支持多 CPU 系统的 CPU 模块。 **■附加信息** ・公共信息： 模块号 (CPU 号) ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时	・对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是某个 CPU 模块的硬件故障。(请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。) ・将不支持多 CPU 系统的 CPU 模块从主基板上卸下。或者，将不支持多 CPU 系统的 CPU 模块更换为支持多 CPU 系统的 CPU 模块。	RUN: 熄灯 ERR.: 闪烁 CPU 状态： 停止	Q00/Q01*1 Qn(H)*1 QnPH
	[MULTI CPU DOWN] 多 CPU 系统的初始化通信时，初始化通信对象的 CPU 模块未返回响应信息。 **■附加信息** ・公共信息： 模块号 (CPU 号) ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时	对 CPU 模块进行复位后再次置为 RUN。仍然显示相同的出错时，可能是某个 CPU 模块的硬件故障。(请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。)		QnU*6
7003	[MULTI CPU DOWN] 多 CPU 系统的初始化通信时，初始化通信对象的 CPU 模块未返回响应信息。 **■附加信息** ・公共信息： 模块号 (CPU 号) ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时			Q00/Q01*1 Qn(H)*1 QnPH

*1 以功能版本 B 以后为对象。
*6 以除 Q00UJCPU 以外的通用型 QCPU 为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
7004	[MULTI CPU DOWN] 在多 CPU 系统中，CPU 模块之间的通信中发生了数据异常。 ■附加信息 ・公共信息： 模块号 (CPU 号) ・个别信息： - ■诊断时机 ・常时	・确认系统配置中是否存在有超过 I/O 点数安装的模块。 ・如果系统配置方面不存在问题，则可能是 CPU 模块的硬件故障。(请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。)		Q00/Q01*1 QnU*6
7010	[MULTI EXE. ERROR] ・在多 CPU 系统中，安装了有故障的 CPU 模块。 ・在多 CPU 系统中，安装了不支持多 CPU 系统的 CPU 模块。 (在支持多 CPU 系统的 CPU 模块中检测出此出错。) ・在多 CPU 系统中，在电源为 ON 的状态下对 2 ～ 4 号机进行了复位。(在进行了复位解除的 CPU 模块中检测出此出错。) ■附加信息 ・公共信息： 模块号 (CPU 号) ・个别信息： - ■诊断时机 ・电源 ON 时 / 复位时	・通过 GX Developer 读取出错的个别信息，对故障的 CPU 模块进行更换。 ・更换为支持多 CPU 系统的模块。 ・不对 2 ～ 4 号机的 CPU 模块进行复位。 ・对 1 号机的 CPU 模块进行复位后，进行多 CPU 系统的再启动。		Q00/Q01*1 Qn(H)*1 QnPH QnU*6
	[MULTI EXE. ERROR] 在多 CPU 系统中，个人计算机 CPU 模块使用了版本为 1.06 或以前的对应软件 (PPC-DRV-01)*5。 ■附加信息 ・公共信息： 模块号 (CPU 号) ・个别信息： - ■诊断时机 ・电源 ON 时 / 复位时	个人计算机 CPU 模块应使用版本为 1.07 或以后的对应软件 (PPC-DRV-01)*5。	RUN: 熄灯 ERR.: 闪烁 CPU 状态： 停止	Q00/Q01*1
	[MULTI EXE. ERROR] 多 CPU 高速主基板 (Q3□DB) 中，安装了 Q172(H)CPU(N)、Q173(H)CPU(N)。(有可能导致模块故障。) ■附加信息 ・公共信息： 模块号 (CPU 号) ・个别信息： - ■诊断时机 ・电源 ON 时 / 复位时	将 Q172(H)CPU(N)、Q173(H)CPU(N) 更换为支持多 CPU 高速主基板的运动控制 CPU。		Qn(H)*4 QnPH*4
	[MULTI EXE. ERROR] 通用型 QCPU(除 Q02UCPU 以外) 与 Q172(H)CPU(N) 被安装在同一个基板上。(有可能导致模块故障。) ■附加信息 ・公共信息： 模块号 (CPU 号) ・个别信息： - ■诊断时机 ・电源 ON 时 / 复位时	确认可进行多 CPU 系统配置的 QCPU 及运动控制 CPU，变更为组合正确的系统配置。		

*1 以功能版本 B 以后为对象。
*4 以序列号的前 5 位数为 “09082” 以后的模块为对象。
*5 正式名称为 MELSEC-Q 系列个人计算机 CPU 模块用总线接口驱动软件包。
*6 以除 Q00UJCPU 以外的通用型 QCPU 为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
	[MULTI EXE. ERROR] 在多 CPU 系统中进行了下述设置之一。 ・对不能执行多 CPU 自动刷新的 CPU 模块进行了多 CPU 自动刷新设置。 ・对不能进行组外获取的 CPU 模块进行组外的输入输出设置。 **■附加信息** ・公共信息： 模块号 (CPU 号) ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时	・对多 CPU 自动刷新设置进行修改。 ・对组外的输入输出设置进行修改。		Q00/Q01*1 QnU*6
7011	[MULTI EXE. ERROR] 系统配置不能满足多 CPU 高速通信功能的使用。 ・1 号机中未使用 QnUCPU。 ・未使用多 CPU 高速主基板 (Q3 □ DB)。 ・对于不支持多 CPU 高速通信功能的 CPU 模块，将其发送范围设置为除 0 点以外。 ・对于不支持多 CPU 共享缓冲区的 CPU 模块，将其发送范围设置为除 0 点以外。 **■附加信息** ・公共信息： 模块号 (CPU 号) ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时	・将系统变更为可满足多 CPU 高速通信功能的使用系统配置。 ・通过多 CPU 兼容区进行自动刷新时，将不支持多 CPU 共享缓冲区的 CPU 的发送范围设置为 0 点。	RUN: 熄灯 ERR.：闪烁 CPU 状态： 停止	QnU*3
7013	[MULTI EXE. ERROR] CPU 插槽或插槽 0 ～ 2 中安装了 Q172(H)CPU(N) 或 Q173(H)CPU(N)。(有可以导致模块故障。) **■附加信息** ・公共信息： 模块号 (CPU 号) ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时	・确认为可构成多 CPU 系统的 QCPU 及运动控制 CPU，变更为可组合的系统配置。 ・将不兼容的运动控制 CPU 卸下。		QnU
7020	[MULTI CPU ERROR] 在多 CPU 系统中，在动作模式为 “不选择系统停止” 的 CPU 模块中发生了异常。 (在未发生异常的 CPU 模块中可检测出此出错。) **■附加信息** ・公共信息： 模块号 (CPU 号) ・个别信息： - **■诊断时机** ・常时	通过 GX Developer 读取出错的个别信息，对异常的 CPU 模块的出错进行确认后，消除出错。	RUN: 亮灯 ERR.：亮灯 CPU 状态： 继续运行	Q00/Q01*1 Qn(H)*1 QnPH QnU*6
7030	[CPU LAY ERROR] 在超出了可编程控制器参数的多 CPU 设置中设置的 CPU 模块个数的可安装 CPU 的插槽 (CPU 插槽、I/O 插槽 0、1) 中发生了分配异常。 **■附加信息** ・公共信息： 模块号 (CPU 号) ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时	・使可编程控制器参数的多 CPU 设置中设置的 CPU 模块个数与安装的 CPU 模块个数 (包括 CPU(空闲)) 一致。 ・使可编程控制器参数的 I/O 分配设置中设置的类型与 CPU 模块的安装状态相符。	RUN: 熄灯 ERR.：闪烁 CPU 状态： 停止	Q00J/Q00/ Q01 *1 QnU

*1 以功能版本 B 以后为对象。
*3 以除 Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU 以外的通用型 QCPU 为对象。
*6 以除 Q00UJCPU 以外的通用型 QCPU 为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
7031	[CPU LAY ERROR] 在可编程控制器参数的多 CPU 设置中设置的 CPU 模块个数的范围内发生了分配异常。 **■附加信息** ・公共信息：模块号 (CPU 号) ・个别信息：－ **■诊断时机** ・电源 ON 时 / 复位时	・使可编程控制器参数的多 CPU 设置中设置的 CPU 模块个数与安装的 CPU 模块个数（包括 CPU(空闲)）一致。 ・使可编程控制器参数的 I/O 分配设置中设置的类型与 CPU 模块的安装状态相符。		Q00J/Q00/ Q01 *1 QnU
7032	[CPU LAY ERROR] 构成多 CPU 系统的各 CPU 模块的安装个数有误。 **■附加信息** ・公共信息：模块号 (CPU 号) ・个别信息：－ **■诊断时机** ・电源 ON 时 / 复位时	进行系统配置时，不要使各 CPU 的安装个数超过规格中规定的最多个数。		Q00J/Q00/ Q01 *1 QnU*6
7035	[CPU LAY ERROR] 在不能安装 CPU 模块的插槽中安装了 CPU 模块。 **■附加信息** ・公共信息：模块号（插槽号） ・个别信息：－ **■诊断时机** ・电源 ON 时 / 复位时	将 CPU 模块安装到可安装 CPU 模块的插槽中。	RUN：熄灯 ERR.：闪烁 CPU 状态：停止	Q00J/Q00/Q01*1 QnPRH QnU
7036	[CPU LAY ERROR] 多 CPU 设置中设置的本机编号与根据 CPU 模块的安装位置确定的本机编号不一致。 **■附加信息** ・公共信息：模块号 (CPU 号) ・个别信息：－ **■诊断时机** ・电源 ON 时 / 复位时	・将 CPU 模块正确地安装到相应插槽中。 ・将 CPU 设置中设置的本机 CPU 号修改为根据 CPU 模块的安装位置确定的 CPU 号。		QnU*3
8031	[INCORRECT FILE] 检测出存储的文件（有效的参数文件）中有异常。 **■附加信息** ・公共信息：－ ・个别信息：文件诊断信息 **■诊断时机** ・电源 ON 时 / 复位时 /STOP → RUN 时 / 可编程控制器写入时	将个别信息的 SD17 ～ SD22 中显示的文件写入到个别信息的 SD16(L) 中显示的驱动器中后，CPU 模块的电源 OFF → ON 或者复位→复位解除。 如果仍然显示相同的出错，则可能是 CPU 模块的硬件故障。（请向附近的系统服务中心、代理商或分公司说明故障症状，进行协商。）		QnU
9000	[F****] 报警器 (F) 变为 ON。 （出错信息的 **** 表示检测出的报警器编号。） **■附加信息** ・公共信息：程序出错位置 ・个别信息：报警器号 **■诊断时机** ・执行指令时	通过外围设备读取出错个别信息，对该数值（报警器号）的程序进行检查。	RUN：亮灯 ERR.： 亮灯 / 熄灯 *2 CPU 状态： 继续运行 RUN、ERR.： USER LED 亮灯 CPU 状态： 继续运行	QCPU

*1 以功能版本 B 以后为对象。
*2 在基本型 QCPU 中，根据 LED 显示优先顺序设置用特殊寄存器 (SD207 ～ SD209)，可以使其熄灯。（在高性能型 QCPU、过程 CPU、冗余 CPU、通用型 QCPU 中，仅为熄灯。）
*3 以除 Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU 以外的通用型 QCPU 为对象。
*6 以除 Q00UJCPU 以外的通用型 QCPU 为对象。

出错代码	异常内容及原因	处理方法	LED 状态 CPU 状态	对应 CPU
9010	[<CHK>ERR ***-***] 通过 CHK 指令检测出出错。 （出错信息的 *** 为检测出的触点及线圈的编号。） **■附加信息** ・公共信息： 程序出错位置 ・个别信息： 故障号 **■诊断时机** ・执行指令时	通过外围设置读取出错的个别信息，对该数值（故障号）的程序进行检查。	RUN：亮灯 ERR.：熄灯 CPU 状态： 继续运行 RUN、ERR.： USER LED 亮灯 CPU 状态： 继续运行	Qn(H) QnPH QnPRH
9020	[BOOT OK] 通过标准 ROM 的自动写入操作正常地完成了 ROM 化。 (BOOT LED 也闪烁） **■附加信息** ・公共信息： - ・个别信息： - **■诊断时机** ・电源 ON 时 / 复位时	将插杆开关的参数有效驱动器设置为标准 ROM 后，进行电源的再启动，执行从标准 ROM 的引导运行。	RUN：熄灯 ERR.：闪烁 CPU 状态： 停止	Qn(H)*1 QnPH QnPRH
10000	[CONT. UNIT ERROR] 在多 CPU 系统中，除过程 CPU/ 高性能型 QCPU 以外的 CPU 模块中发生了出错。 **■附加信息** ・公共信息： - ・个别信息： - **■诊断时机** ・常时	发生的出错详细内容可以通过 GX Developer 连接到相应的 CPU 模块中进行确认。	RUN：熄灯 ERR.：闪烁 CPU 状态： 继续运行	Qn(H)*1 QnPH

*1 以功能版本 B 以后为对象。

12.2 出错的解除

对于 CPU 模块，只有当 CPU 模块的动作为继续运行出错时，才可以进行出错解除操作。

出错解除应按以下步骤进行。

1) 消除出错原因。
2) 将要解除的出错代码存储到特殊寄存器 SD50 中。
3) 对特殊继电器 SM50 进行 OFF → ON 操作。
4) 解除要解除的出错。

通过出错解除使 CPU 模块恢复后，与出错相关的特殊继电器、特殊寄存器以及 LED/LED 显示器将返回到出错发生之前的状态。

如果在出错解除后再次发生相同的出错，将被重新登录到故障履历中。

当对检测出的多个报警器 (F) 进行解除时，只有最先检测出的报警号的报警器被解除。

关于出错解除的详细信息，请参阅以下手册。

• 所使用的 CPU 的用户手册（功能解说 / 程序基础篇）

要 点

1. 将要解除的出错代码存储到 SD50 中并进行了出错解除的情况下，下 1 位数的出错代码编号将被忽略。

   例

   如果发生了代码 2100、2101 的出错，则出错代码 2100 被解除时，出错代码 2101 也将被解除。
   如果发生了代码 2100、2111 的出错，则出错代码 2100 被解除时，出错代码 2111 不会被解除。

2. 对于 CPU 模块以外原因引起的出错，即使通过特殊继电器 (SM50) 和特殊寄存器 (SD50) 进行了出错解除，也不能消除出错原因。

   例

   由于 “SP.UNIT DOWN” 是发生在基板（包括扩展电缆）、智能功能模块等部件上的出错，因此即使通过特殊继电器 (SM50) 和特殊寄存器 (SD50) 进行了出错解除，也不能消除出错原因。
   应参照出错代码一览表，消除出错原因。

附　录

附录 1　运算处理时间

附录 1.1　运算处理时间的思路

(1) QCPU 的处理时间是下述处理时间的合计。

- 各指令的处理时间的合计
- END 处理时间（包括 I/O 刷新时间）
- 扫描时间延迟功能的处理时间

(2) 各指令的处理时间

是附录 1.2、附录 1.3、附录 1.4 中记述的各指令的处理时间的合计时间。

(3) END 处理时间、I/O 刷新时间、扫描时间延迟功能的处理时间

关于 END 处理时间、I/O 刷新时间、扫描时间延迟功能的处理时间的内容，请参阅以下手册。

- QnUCPU 用户手册（功能解说 / 程序基础篇）
- Qn(H)/QnPH/QnPRHCPU 用户手册（功能解说 / 程序基础篇）

附录 1.2　基本型 QCPU 的运算处理时间

各指令的运算处理时间如本页以后的表中所示。

由于运算处理时间根据源、目标的内容而有所不同，因此表中的值只应作为处理时间的大致参考。

(1) 顺控程序指令

指令	条件（软元件）	处理时间（μs）		
		Q00JCPU	Q00CPU	Q01CPU
LD LDI AND ANI OR ORI	X0	0.20	0.16	0.10
	D0.0	0.30	0.24	0.15
LDP LDF ANDP ANDF ORP ORF	X0	0.30	0.24	0.15
	D0.0			
ANB ORB MPS MRD MPP	-	0.20	0.16	0.10
INV	未执行时	0.20	0.16	0.10
	执行时			
MEP MEF	未执行时	0.30	0.24	0.15
	执行时			
EGP	未执行时 (OFF→OFF) (ON→ON)	0.20	0.16	0.10
	执行时 (OFF→ON) (ON→OFF)			
EGF	未执行时 (OFF→OFF) (ON→ON)	17	9.5	9.4
	执行时 (OFF→ON) (ON→OFF)	18	14	14

指令		条件（软元件）			处理时间（μs）		
					Q00JCPU	Q00CPU	Q01CPU
OUT	Y	无变化时 (OFF→OFF) (ON→ON)			0.20	0.16	0.10
		变化时 (OFF→ON) (ON→OFF)			0.20	0.16	0.10
	D0.0	无变化时 (OFF→OFF) (ON→ON)			0.40	0.32	0.20
		变化时 (OFF→ON) (ON→OFF)			0.40	0.32	0.20
	F	OFF 时			24	20	19
		ON 时	显示时		260	210	200
			显示结束		205	165	155
	T	未执行时			1.1	0.88	0.55
		执行时	时间到后		1.1	0.88	0.55
			加法运算时	K	1.1	0.88	0.55
				D	1.2	0.96	0.60
	C	未执行时			1.1	0.88	0.55
		执行时	时间到后		1.1	0.88	0.55
			加法运算时	K	1.1	0.88	0.55
				D	1.2	0.96	0.60
OUTH	T	未执行时			1.1	0.88	0.55
		执行时	时间到后		1.1	0.88	0.55
			加法运算时	K	1.1	0.88	0.55
				D	1.2	0.96	0.60
SET	Y	未执行时			0.20	0.16	0.10
		执行时	无变化时（ON→ON）		0.20	0.16	0.10
			变化时（OFF→ON）		0.20	0.16	0.10
	D0.0	未执行时			0.40	0.32	0.20
		执行时	无变化时（ON→ON）		0.40	0.32	0.20
			变化时（OFF→ON）		0.40	0.32	0.20
	F	未执行时			0.50	0.44	0.25
		执行时	显示时		255	205	195
			显示结束		195	160	150
RST	Y	未执行时			0.20	0.16	0.10
		执行时	无变化时（OFF→OFF）		0.20	0.16	0.10
			变化时（ON→OFF）		0.20	0.16	0.10
	D0.0	未执行时			0.40	0.32	0.20
		执行时	无变化时（ON→ON）		0.40	0.32	0.20
			变化时（OFF→ON）		0.40	0.32	0.20
	SM	未执行时			0.20	0.16	0.10
		执行时			0.20	0.16	0.10
	F	未执行时			0.48	0.44	0.25
		执行时	显示时		75	69	65
			显示结束		43	35	33
	T,C	未执行时			0.80	0.64	0.40
		执行时			1.0	0.80	0.50
	D	未执行时			0.40	0.32	0.20
		执行时			0.60	0.48	0.30
	Z	未执行时			0.50	0.40	0.25
		执行时			9.4	7.9	7.4
	R	未执行时			-	0.32	0.20
		执行时			-	0.48	0.30

指令		条件（软元件）	处理时间（μs）		
			Q00JCPU	Q00CPU	Q01CPU
PLS			12	9.5	9.2
PLF			11	9.5	8.9
FF	Y	未执行时	0.68	0.40	0.25
		执行时	7.5	6.2	5.7
DELTA	DY0	未执行时	0.50	0.40	0.25
		执行时	26	21	21
DELTAP	DY0	未执行时	0.48	0.40	0.25
		执行时	58	45	43
SFT SFTP		未执行时	0.50	0.34	0.25
		执行时	12	8.7	8.3
MC		M0	0.40	0.32	0.20
		D0.0	3.3	2.9	2.8
MCR		-	0.20	0.16	0.10
FEND END		进行出错检查	660	600	520
		不进行出错检查 （・电池检查） （・保险丝熔断检查） （・I/O 模块校验）	660	600	520
NOP		-	0.20	0.16	0.10
NOPLF PAGE		-	0.20	0.16	0.10

(2) 基本指令

未执行时的处理时间如下所示。

Q00JCPU ..... 0.20 ×（各指令的步数 +1）μs
Q00CPU ..... 0.16 ×（各指令的步数 +1）μs
Q01CPU ..... 0.10 ×（各指令的步数 +1）μs

指令	条件（软元件）		处理时间（μs）		
			Q00JCPU	Q00CPU	Q01CPU
LD =	导通时		0.80	0.64	0.40
	非导通时		0.80	0.64	0.40
AND =	未执行时		0.70	0.56	0.35
	执行时	导通时	0.80	0.64	0.40
		非导通时	0.80	0.64	0.40
OR =	未执行时		0.70	0.56	0.35
	执行时	导通时	0.80	0.64	0.40
		非导通时	0.80	0.64	0.40
LD < >	导通时		0.80	0.64	0.40
	非导通时		0.80	0.64	0.40
AND < >	未执行时		0.70	0.56	0.35
	执行时	导通时	0.80	0.64	0.40
		非导通时	0.80	0.64	0.40

指令	条件（软元件）		处理时间（μs）		
			Q00JCPU	Q00CPU	Q01CPU
OR < >	未执行时		0.70	0.56	0.35
	执行时	导通时	0.80	0.64	0.40
		非导通时	0.80	0.64	0.40
LD >	导通时		0.80	0.64	0.40
	非导通时		0.80	0.64	0.40
AND >	未执行时		0.70	0.56	0.35
	执行时	导通时	0.80	0.64	0.40
		非导通时	0.80	0.64	0.40
OR >	未执行时		0.70	0.56	0.35
	执行时	导通时	0.80	0.64	0.40
		非导通时	0.80	0.64	0.40
LD < =	导通时		0.80	0.64	0.40
	非导通时		0.80	0.64	0.40
AND < =	未执行时		0.70	0.56	0.35
	执行时	导通时	0.80	0.64	0.40
		非导通时	0.80	0.64	0.40
OR < =	未执行时		0.70	0.56	0.35
	执行时	导通时	0.80	0.64	0.40
		非导通时	0.80	0.64	0.40
LD<	导通时		0.80	0.64	0.40
	非导通时		0.80	0.64	0.40
AND<	未执行时		0.70	0.56	0.35
	执行时	导通时	0.80	0.64	0.40
		非导通时	0.80	0.64	0.40
OR<	未执行时		0.70	0.56	0.35
	执行时	导通时	0.80	0.64	0.40
		非导通时	0.80	0.64	0.40
LD > =	导通时		0.80	0.64	0.40
	非导通时		0.80	0.64	0.40
AND > =	未执行时		0.70	0.56	0.35
	执行时	导通时	0.80	0.64	0.40
		非导通时	0.80	0.64	0.40
OR > =	未执行时		0.70	0.56	0.35
	执行时	导通时	0.80	0.64	0.40
		非导通时	0.80	0.64	0.40
LDD =	导通时		1.0	0.80	0.50
	非导通时		1.0	0.80	0.50
ANDD =	未执行时		0.80	0.64	0.40
	执行时	导通时	1.0	0.80	0.50
		非导通时	1.0	0.80	0.50
ORD =	未执行时		0.80	0.64	0.40
	执行时	导通时	1.0	0.80	0.50
		非导通时	1.0	0.80	0.50
LDD < >	导通时		1.0	0.80	0.50
	非导通时		1.0	0.80	0.50
ANDD < >	未执行时		0.80	0.64	0.40
	执行时	导通时	1.0	0.80	0.50
		非导通时	1.0	0.80	0.50

指令	条件（软元件）		处理时间（μs）		
			Q00JCPU	Q00CPU	Q01CPU
ORD < >	未执行时		0.80	0.64	0.40
	执行时	导通时	1.0	0.80	0.50
		非导通时	1.0	0.80	0.50
LDD >	导通时		1.0	0.80	0.50
	非导通时		1.0	0.80	0.50
ANDD >	未执行时		0.80	0.64	0.40
	执行时	导通时	1.0	0.80	0.50
		非导通时	1.0	0.80	0.50
ORD >	未执行时		0.80	0.64	0.40
	执行时	导通时	1.0	0.80	0.50
		非导通时	1.0	0.80	0.50
LDD < =	导通时		1.0	0.80	0.50
	非导通时		1.0	0.80	0.50
ANDD < =	未执行时		0.80	0.64	0.40
	执行时	导通时	1.0	0.80	0.50
		非导通时	1.0	0.80	0.50
ORD < =	未执行时		0.80	0.64	0.40
	执行时	导通时	1.0	0.80	0.50
		非导通时	1.0	0.80	0.50
LDD<	导通时		1.0	0.80	0.50
	非导通时		1.0	0.80	0.50
ANDD<	未执行时		0.80	0.64	0.40
	执行时	导通时	1.0	0.80	0.50
		非导通时	1.0	0.80	0.50
ORD<	未执行时		0.80	0.64	0.40
	执行时	导通时	1.0	0.80	0.50
		非导通时	1.0	0.80	0.50
LDD > =	导通时		1.0	0.80	0.50
	非导通时		1.0	0.80	0.50
ANDD > =	未执行时		0.80	0.64	0.40
	执行时	导通时	1.0	0.80	0.50
		非导通时	1.0	0.80	0.50
ORD > =	未执行时		0.80	0.64	0.40
	执行时	导通时	1.0	0.80	0.50
		非导通时	1.0	0.80	0.50
BKCMP = (S1) (S2) (D) n	n = 1		130	105	97
BKCMP = P (S1) (S2) (D) n	n = 96		205	175	165
BKCMP < > (S1) (S2) (D) n	n = 1		130	105	98
BKCMP < > P (S1) (S2) (D) n	n = 96		210	180	165
BKCMP > (S1) (S2) (D) n	n = 1		130	105	97
BKCMP > P (S1) (S2) (D) n	n = 96		210	180	165
BKCMP > = (S1) (S2) (D) n	n = 1		130	105	98
BKCMP > = P (S1) (S2) (D) n	n = 96		205	175	165

指令	条件（软元件）	处理时间（μs）		
		Q00JCPU	Q00CPU	Q01CPU
BKCMP< (S1) (S2) (D) n	n = 1	130	105	98
BKCMP< P (S1) (S2) (D) n	n = 96	210	180	165
BKCMP <= (S1) (S2) (D) n	n = 1	130	105	97
BKCMP <= P (S1) (S2) (D) n	n = 96	205	175	165
+ (S) (D) + P (S) (D)	执行时	1.0	0.80	0.50
+ (S1) (S2) (D) + P (S1) (S2) (D)	执行时	1.2	0.96	0.60
- (S) (D) - P (S) (D)	执行时	1.0	0.80	0.50
- (S1) (S2) (D) - P (S1) (S2) (D)	执行时	1.2	0.96	0.60
D + (S) (D) D + P (S) (D)	执行时	1.3	1.04	0.65
D + (S1) (S2) (D) D + P (S1) (S2) (D)	执行时	1.5	1.2	0.75
D - (S) (D) D - P (S) (D)	执行时	1.3	1.04	0.65
D - (S1) (S2) (D) D - P (S1) (S2) (D)	执行时	1.5	1.2	0.75
* (S1) (S2) (D) * P (S1) (S2) (D)	执行时	1.1	0.88	0.55
/ (S1) (S2) (D) / P (S1) (S2) (D)	-	19	16	15
D * (S1) (S2) (D) D * P (S1) (S2) (D)	-	41	34	31
D / (S1) (S2) (D) D / P (S1) (S2) (D)	-	28	23	21
B + (S) (D) B + P (S) (D)	-	34	28	26
B + (S1) (S2) (D) B + P (S1) (S2) (D)	-	47	39	37
B - (S) (D) B - P (S) (D)	-	34	28	26
B - (S1) (S2) (D) B - P (S1) (S2) (D)	-	48	40	38
DB + (S) (D) DB + P (S) (D)	-	58	48	44
DB + (S1) (S2) (D) DB + P (S1) (S2) (D)	-	60	49	46
DB - (S) (D) DB - P (S) (D)	-	59	48	45
DB - (S1) (S2) (D) DB - P (S1) (S2) (D)	-	60	51	45

指令	条件（软元件）	处理时间（μs）		
		Q00JCPU	Q00CPU	Q01CPU
B * (S1) (S2) (D) B * P (S1) (S2) (D)	-	42	35	33
B / (S1) (S2) (D) B / P (S1) (S2) (D)	-	48	40	37
DB * (S1) (S2) (D) DB * P (S1) (S2) (D)	-	140	120	110
DB / (S1) (S2) (D) DB / P (S1) (S2) (D)	-	83	69	65
BK + (S1) (S2) (D) n	n = 1	105	86	80
BK + P (S1) (S2) (D) n	n = 96	185	155	140
BK - (S1) (S2) (D) n	n = 1	105	86	80
BK - P (S1) (S2) (D) n	n = 96	185	155	140
INC INCP	-	0.70	0.56	0.35
DINC DINCP	-	0.90	0.72	0.45
DEC DECP	-	0.70	0.56	0.35
DDEC DDECP	-	0.90	0.72	0.45
BCD BCDP	-	20	16	15
DBCD DBCDP	-	26	21	20
BIN BINP	-	19	16	15
DBIN DBINP	-	22	18	17
DBL DBLP	-	19	16	15
WORD WORDP	-	23	19	17
GRY GRYP	-	19	16	15
DGRY DGRYP	-	23	19	17
GBIN GBINP	-	52	42	40
DGBIN DGBINP	-	110	88	84
NEG NEGP	-	16	13	12
DNEG DNEGP	-	19	17	15

指令		条件（软元件）	处理时间（μs）		
			Q00JCPU	Q00CPU	Q01CPU
BKBCD (S) (D) n		n = 1	78	63	57
BKBCDP (S) (D) n		n = 96	315	275	250
BKBIN (S) (D) n		n = 1	74	61	57
BKBINP (S) (D) n		n = 96	285	255	230
MOV		(S) = D0, (D) = D1	0.70	0.56	0.35
MOVP		(S) = D0, (D) = J1\W1	155	130	120
DMOV		(S) = D0, (D) = D1	0.90	0.72	0.45
DMOVP		(S) = D0, (D) = J1\W1	165	135	120
$MOV		0 字符	46	38	35
$MOVP		32 字符	98	80	73
CML CMLP		-	0.70	0.56	0.35
DCML DCMLP		-	0.90	0.72	0.45
BMOV (S) (D) n		n = 1	27	21	20
BMOVP (S) (D) n		n = 96	72	62	53
FMOV (S) (D) n		n = 1	23	19	17
FMOVP (S) (D) n		n = 96	48	41	36
XCH XCHP		-	7.6	6.3	5.7
DXCH DXCHP		-	9.5	8.0	7.1
BXCH (D1) (D2) n		n = 1	62	51	48
BXCHP (D1) (D2) n		n = 96	165	140	125
SWAP SWAPP		-	17	14	13
CJ		-	10	8.5	8.1
SCJ		-	10	8.5	8.1
JMP		-	11	8.5	8.1
GOEND		-	3.3	2.9	2.8
DI		-	13	12	11
EI		-	14	11	11
IMASK		-	41	34	35
IRET		-	205	170	155
RFS RFSP	X	n = 1	55	46	43
		n = 96	79	64	59
	Y	n = 1	54	45	41
		n = 96	73	61	56

(3) 应用指令

未执行时的处理时间如下所示。

Q00JCPU ..... 0.20 × (各指令的步数 +1) μ s
Q00CPU ..... 0.16 × (各指令的步数 +1) μ s
Q01CPU ..... 0.10 × (各指令的步数 +1) μ s

指令	条件 (软元件)	处理时间 (μ s)		
		Q00JCPU	Q00CPU	Q01CPU
WAND (S) (D) WANDP (S) (D)	执行时	1.0	0.80	0.50
WAND (S1) (S2) (D) WANDP (S1) (S2) (D)	执行时	1.2	0.96	0.60
DAND (S) (D) DANDP (S) (D)	执行时	1.3	1.04	0.65
DAND (S1) (S2) (D) DANDP (S1) (S2) (D)	执行时	1.5	1.2	0.75
BKAND (S1) (S2) (D) n	n = 1	110	87	79
BKANDP (S1) (S2) (D) n	n = 96	185	155	140
WOR (S) (D) WORP (S) (D)	执行时	1.0	0.80	0.50
WOR (S1) (S2) (D) WORP (S1) (S2) (D)	执行时	1.2	0.96	0.60
DOR (S) (D) DORP (S) (D)	执行时	1.3	1.04	0.65
DOR (S1) (S2) (D) DORP (S1) (S2) (D)	执行时	1.5	1.2	0.75
BKOR (S1) (S2) (D) n	n = 1	110	87	81
BKORP (S1) (S2) (D) n	n = 96	185	155	140
WXOR (S) (D) WXORP (S) (D)	执行时	1.0	0.80	0.50
WXOR (S1) (S2) (D) WXORP (S1) (S2) (D)	执行时	1.2	0.96	0.60
DXOR (S) (D) DXORP (S) (D)	执行时	1.3	1.04	0.65
DXOR (S1) (S2) (D) DXORP (S1) (S2) (D)	执行时	1.5	1.2	0.75
BKXOR (S1) (S2) (D) n	n = 1	110	87	81
BKXORP (S1) (S2) (D) n	n = 96	185	155	140

指令	条件（软元件）	处理时间（μs）		
		Q00JCPU	Q00CPU	Q01CPU
WXNR (S) (D) WXNRP (S) (D)	执行时	1.0	0.80	0.50
WXNR (S1) (S2) (D) WXNRP (S1) (S2) (D)	执行时	1.2	0.96	0.60
DXNR (S) (D) DXNRP (S) (D)	执行时	1.3	1.04	0.65
DXNR (S1) (S2) (D) DXNRP (S1) (S2) (D)	执行时	1.5	1.2	0.75
BKXNR (S1) (S2) (D) n	n = 1	110	87	82
BKXNRP (S1) (S2) (D) n	n = 96	185	155	140
ROR (D) n	n = 1	13	11	9.7
RORP (D) n	n = 15	13	11	9.7
RCR (D) n	n = 1	15	12	12
RCRP (D) n	n = 15	15	13	12
ROL (D) n	n = 1	13	11	10
ROLP (D) n	n = 15	13	11	10
RCL (D) n	n = 1	15	13	12
RCLP (D) n	n = 15	16	13	12
DROR (D) n	n = 1	15	12	12
DRORP (D) n	n = 31	15	13	12
DRCR (D) n	n = 1	17	14	14
DRCRP (D) n	n = 31	18	16	15
DROL (D) n	n = 1	14	13	12
DROLP (D) n	n = 31	14	13	12
DRCL (D) n	n = 1	18	15	14
DRCLP (D) n	n = 31	20	17	16
SFR (D) n	n = 1	13	10	9.7
SFRP (D) n	n = 15	13	11	9.5
SFL (D) n	n = 1	12	10	9.5
SFLP (D) n	n = 15	12	9.8	9.5
BSFR (D) n	n = 1	42	35	33
BSFRP (D) n	n = 96	69	58	54
BSFL (D) n	n = 1	41	34	32
BSFLP (D) n	n = 96	63	53	50
DSFR (D) n	n = 1	19	16	15
DSFRP (D) n	n = 96	71	61	53
DSFL (D) n	n = 1	19	16	15
DSFLP (D) n	n = 96	70	60	52

指令	条件（软元件）		处理时间（μs）		
			Q00JCPU	Q00CPU	Q01CPU
BSET (D) n	n = 1		27	22	20
BSETP (D) n	n = 15		27	22	20
BRST (D) n	n = 1		27	22	21
BRSTP (D) n	n = 15		27	22	21
TEST (S1) (S2) (D) TESTP (S1) (S2) (D)	-		35	30	27
DTEST (S1) (S2) (D) DTESTP (S1) (S2) (D)	-		37	31	28
BKRST (D) n	n = 1		49	41	38
BKRSTP (D) n	n = 96		64	54	50
SER (S1) (S2) (D) n	n = 1	全部一致	56	54	42
		全部不一致	56	54	42
SERP (S1) (S2) (D) n	n = 96	全部一致	280	240	220
		全部不一致	280	240	220
DSER (S1) (S2) (D) n	n = 1	全部一致	71	67	53
		全部不一致	71	67	54
DSERP (S1) (S2) (D) n	n = 96	全部一致	495	415	375
		全部不一致	500	415	375
SUM SUMP	(S) = 0		32	26	25
	(S) = $FFFF_H$		27	22	21
DSUM DSUMP	(S) = 0		54	44	42
	(S) = $FFFFFFFF_H$		54	44	42
DECO (S) (D) n	n = 2		60	50	46
DECOP (S) (D) n	n = 8		80	65	61
ENCO (S) (D) n	n = 2	M1 = ON	66	55	51
		M4 = ON	66	54	51
ENCOP (S) (D) n	n = 8	M1 = ON	90	76	71
		M256 = ON	76	74	71
SEG SEGP	-		8.0	6.8	6.1
DIS (S) (D) n	n = 1		47	39	36
DISP (S) (D) n	n = 4		53	43	40
UNI (S) (D) n	n = 1		54	44	41
UNIP (S) (D) n	n = 4		60	49	46
NDIS (S1) (D) (S2) NDISP (S1) (D) (S2)	-		92	76	38
NUNI (S1) (D) (S2) NUNIP (S1) (D) (S2)	-		47	39	36

指令	条件（软元件）	处理时间（μs）		
		Q00JCPU	Q00CPU	Q01CPU
WTOB (S) (D) n	n = 1	56	46	42
WTOBP (S) (D) n	n = 96	190	155	145
BTOW (S) (D) n	n = 1	56	46	42
BTOWP (S) (D) n	n = 96	190	155	145
MAX (S) (D) n	n = 1	48	40	36
MAXP (S) (D) n	n = 96	300	240	235
MIN (S) (D) n	n = 1	48	40	36
MINP (S) (D) n	n = 96	300	240	235
DMAX (S) (D) n	n = 1	52	43	39
DMAXP (S) (D) n	n = 96	600	490	460
DMIN (S) (D) n	n = 1	52	43	39
DMINP (S) (D) n	n = 96	585	475	445
SORT (S1) n (S2) (D1) (D2)	n = 1	66	55	50
	n = 96	105	86	80
DSORT (S1) n (S2) (D1) (D2)	n = 1	98	57	52
	n = 96	115	96	88
WSUM (S) (D) n	n = 1	52	43	40
WSUMP (S) (D) n	n = 96	175	140	135
DWSUM (S) (D) n	n = 1	61	51	46
DWSUMP (S) (D) n	n = 96	515	420	395
FOR n	n = 0	11	8.9	8.1
NEXT	-	8.8	7.3	6.8
BREAK	-	37	30	28
BREAKP				
CALL Pn CALLP Pn	-	17	14	13
CALL Pn (S1) ～ (S5)	-	245	200	190
CALLP Pn (S1) ～ (S5)				
RET	返回至自身程序	16	13	12
FCALL Pn FCALLP Pn	-	29	24	22
FCALL Pn (S1) ～ (S5) FCALLP Pn (S1) ～ (S5)	-	250	205	190

指令	条件（软元件）	处理时间（μs）		
		Q00JCPU	Q00CPU	Q01CPU
COM	－	110	77	72
IX	－	65	54	51
IXEND	－	30	26	25
IXDEV + IXSET	触点数 1	145	120	110
	触点数 14	770	630	585
FIFW FIFWP	数据数 0	36	32	28
	数据数 96	36	32	28
FIFR FIFRP	数据数 1	45	41	36
	数据数 96	93	82	70
FPOP FPOPP	数据数 1	40	37	32
	数据数 96	40	37	32
FINS FINSP	数据数 0	53	44	38
	数据数 96	100	89	76
FDEL FDELP	数据数 1	60	50	43
	数据数 96	110	95	82
FROM n1 n2 (D) n3	n3 = 1	125	105	93
FROMP n1 n2 (D) n3 *1	n3 = 1000	740	695	685
DFRO n1 n2 (D) n3	n3 = 1	130	110	100
DFROP n1 n2 (D) n3 *1	n3 = 500	745	695	675
TO n1 n2 (S) n3	n3 = 1	120	105	92
TOP n1 n2 (S) n3 *1	n3 = 1000	735	680	645
DTO n1 n2 (S) n3	n3 = 1	130	110	99
DTOP n1 n2 (S) n3 *1	n3 = 500	740	680	640

*1: FROM/TO 指令的处理时间根据插槽数及安装模块而有所不同。
（根据扩展基板的类型其处理时间也有所不同。）

指令	条件（软元件）	处理时间（μs）		
		Q00JCPU	Q00CPU	Q01CPU
LIMIT LIMITP	－	34	28	26
DLIMIT DLIMITP	－	41	34	30
BAND BANDP	－	33	28	25
DBAND DBANDP	－	40	34	30
ZONE ZONEP	－	31	25	24
DZONE DZONEP	－	37	29	28
RSET RSETP	－	－	18	16
DATERD DATERDP	－	30	25	23
DATEWR DATEWRP	－	69	57	54
DATE + DATE + P	无进位	47	39	36
	有进位	50	42	38
DATE － DATE － P	无进位	47	40	36
	有进位	50	42	38
SECOND SECONDP	－	28	24	22
HOUR HOURP	－	38	32	29
WDT WDTP	－	18	15	14
DUTY	－	41	36	32
ZRRDB ZRRDBP	－	－	24	22
ZRWRB ZRWRBP	－	－	27	24
ADRSET ADRSETP	－	23	19	18
ZPUSH ZPUSHP	－	38	33	30
ZPOP ZPOPP	－	37	31	29
ZCOM	－	105	82	80

(4) QCPU 用指令处理时间（QCPU 专用）

指令	条件（软元件）	处理时间（μs）		
		Q00JCPU	Q00CPU	Q01CPU
UNIRD UNIRDP	n = 1	96	80	74
	n = 16	440	370	340

(5) 序列号的前 5 位数为 “04122” 以后中可使用的指令

指令	条件（软元件）			处理时间（μs）		
				Q00JCPU	Q00CPU	Q01CPU
LDE =	单精度	导通时		43.0	35.5	33.0
		非导通时		46.0	38.0	35.5
ANDE =	单精度	未执行时		1.5	1.2	1.0
		执行时	导通时	35.5	29.5	26.5
			非导通时	42.0	35.0	32.5
ORE =	单精度	未执行时		1.5	1.2	1.0
		执行时	导通时	42.0	35.0	32.5
			非导通时	37.0	31.0	28.5
LDE < >	单精度	导通时		46.0	38.0	35.5
		非导通时		43.5	36.0	33.0
ANDE < >	单精度	未执行时		1.5	1.2	1.0
		执行时	导通时	38.5	31.5	29.0
			非导通时	39.5	33.0	30.5
ORE < >	单精度	未执行时		1.5	1.2	1.0
		执行时	导通时	45.0	37.5	35.0
			非导通时	34.5	29.0	26.5
LDE >	单精度	导通时		46.0	37.5	35.5
		非导通时		46.0	38.5	35.0
ANDE >	单精度	未执行时		1.5	1.2	1.0
		执行时	导通时	38.5	32.0	29.0
			非导通时	42.0	35.0	32.5
ORE >	单精度	未执行时		1.5	1.2	1.0
		执行时	导通时	45.0	37.5	34.5
			非导通时	37.0	31.0	29.0
LDE < =	单精度	导通时		45.5	37.5	35.0
		非导通时		46.5	38.5	35.5
ANDE < =	单精度	未执行时		1.5	1.2	1.0
		执行时	导通时	38.5	31.5	29.0
			非导通时	42.5	35.5	32.5
ORE < =	单精度	未执行时		1.5	1.2	1.0
		执行时	导通时	45.0	37.5	34.5
			非导通时	37.5	31.5	28.5
LDE<	单精度	导通时		45.5	37.5	35.0
		非导通时		46.5	38.5	35.5
ANDE<	单精度	未执行时		1.5	1.2	1.0
		执行时	导通时	38.0	31.5	29.0
			非导通时	42.5	35.5	32.5
ORE<	单精度	未执行时		1.5	1.2	1.0
		执行时	导通时	45.0	37.5	34.5
			非导通时	37.5	31.5	29.0
LDE > =	单精度	导通时		45.5	38.0	35.5
		非导通时		46.5	38.0	35.0
ANDE > =	单精度	未执行时		1.5	1.2	1.0
		执行时	导通时	38.5	32.0	29.0
			非导通时	42.5	35.5	32.5

指令	条件（软元件）			处理时间（μs）		
				Q00JCPU	Q00CPU	Q01CPU
ORE > =	单精度	未执行时		1.5	1.2	1.0
		执行时	导通时	45.0	38.5	34.5
			非导通时	37.5	31.0	28.5
E+ (S) (D) E+ P (S) (D)	单精度	(S) = 0, (D) = 0		29.5	25.0	23.0
		(S) = 2^{127}, (D) = 2^{127}		65.5	60.5	49.5
E+ (S1) (S2) (D) E+ P (S1) (S2) (D)	单精度	(S1) = 0, (S2) = 0		31.0	27.0	24.0
		(S1) = 2^{127}, (S2) = 2^{127}		66.5	56.0	51.0
E- (S) (D) E- P (S) (D)	单精度	(S) = 0, (D) = 0		29.5	25.0	23.0
		(S) = 2^{127}, (D) = 2^{127}		48.5	41.0	37.5
E- (S1) (S2) (D) E- P (S1) (S2) (D)	单精度	(S1) = 0, (S2) = 0		31.0	27.0	24.0
		(S1) = 2^{127}, (S2) = 2^{127}		50.5	42.5	38.5
E* (S1) (S2) (D) E* P (S1) (S2) (D)	单精度	(S1) = 0, (S2) = 0		30.0	25.5	23.0
		(S1) = 2^{127}, (S2) = 2^{127}		65.5	55.0	49.5
E/ (S1) (S2) (D) E/ P (S1) (S2) (D)	单精度	(S1) = 0, (S2) = 1		30.0	26.0	23.0
		(S1) = 2^{127}, (S2) = -2^{126}		69.5	57.5	53.0
INT INTP	单精度	(S) = 0		21.5	18.5	16.0
		(S) = 32766.5		38.0	32.0	29.5
DINT DINTP	单精度	(S) = 0		23.0	19.5	17.5
		(S) = 1234567890.3		42.0	35.5	32.0
FLT FLTP	单精度	(S) = 0		22.5	19.5	17.0
		(S) = $7FFF_H$		26.5	23.0	20.0
DFLT DFLTP	单精度	(S) = 0		23.0	20.0	17.5
		(S) = $7FFFFFFF_H$		26.0	23.5	19.5
ENEG ENEGP	(S) = 0			20.5	17.0	15.5
	(S) = E - 1.0			31.5	26.0	24.0
EMOV EMOVP	-			1.5	1.2	1.0
ESTR ESTRP	-			604.0	686.0	831.0
EVAL EVALP	小数点形式全 2 位数指定			138.0	148.0	196.0
	指数形式全 6 位数指定			164.0	177.0	214.0

指令	条件（软元件）		处理时间（μs）		
			Q00JCPU	Q00CPU	Q01CPU
SIN SINP	单精度		204.0	173.0	157.0
COS COSP	单精度		187.0	158.0	144.0
TAN TANP	单精度		224.0	190.0	173.0
RAD RADP	单精度		51.0	43.0	39.0
DEG DEGP	单精度		51.0	43.0	39.0
SQR SQRP	单精度		60.0	51.0	46.5
EXP EXPP	单精度	Ⓢ = -10	306.0	259.0	235.0
		Ⓢ = 1	306.0	259.0	235.0
LOG LOGP	单精度	Ⓢ = 1	73.0	61.5	56.0
		Ⓢ = 10	301.0	255.0	232.0
RND RNDP	-		12.5	11.0	10.0
SRND SRNDP	-		13.5	12.0	11.0

指令名	条件・处理点数		处理时间（μs）		
			Q00JCPU	Q00CPU	Q01CPU
COM *2	有 CPU 共享存储器的自动刷新	刷新范围：2k 字（各 CPU 0.5k 字的均等分配）	-	920	880
	无 CPU 共享存储器的自动刷新	-	-	150	135
FROM	本机 CPU 共享存储器读取	n3 = 1	-	100	90
		n3 = 320	-	440	420
	其它机号 CPU 共享存储器读取	n3 = 1	-	110	105
		n3 = 320	-	305	290
TO	至本机 CPU 共享存储器的写入	n3 = 1	-	100	95
		n3 = 320	-	440	425
S.TO	至本机 CPU 共享存储器的写入	n4 = 1	-	205	195
		n4 = 320	-	545	525

*2:　在多 CPU 系统中，与其它机号 CPU 的处理重叠时，将最多延迟下述时间。

仅主基板系统时

（指令的延迟时间）= 4 × 0.54 ×（处理点数）×（其它机号 CPU 个数）(μs)

包括扩展基板时

（指令的延迟时间）= 4 × 1.30 ×（处理点数）×（其它机号 CPU 个数）(μs)

附录 1.3 高性能型 QCPU/ 过程 CPU/ 冗余 CPU 的运算处理时间

各指令的运算处理时间如本页以后的表中所示。

由于运算处理时间根据源、目标的内容而有所不同，因此表中的值只应作为处理时间的大致参考。

(1) 顺控程序指令

指令	条件（软元件）	处理时间（μs）			
		Qn	QnH	QnPH	QnPRH
LD LDI AND ANI OR ORI	-	0.079	0.034	0.034	0.034
LDP LDF ANDP ANDF ORP ORF	-	0.158	0.068	0.068	0.068
ANB ORB MPS MRD MPP	-	0.079	0.034	0.034	0.034
INV	未执行时	0.079	0.034	0.034	0.034
	执行时				
MEP MEF	未执行时	0.173	0.073	0.073	0.073
	执行时				
EGP EGF	未执行时 (OFF→OFF) (ON→ON)	0.158	0.068	0.068	0.068
	执行时 (OFF→ON) (ON→OFF)				

指令	条件（软元件）				处理时间（μs）			
					Qn	QnH	QnPH	QnPRH
OUT	无变化时 (OFF→OFF) (ON→ON)				0.158	0.068	0.068	0.068
	变化时 (OFF→ON) (ON→OFF)				0.158	0.068	0.068	0.068
	F	OFF 时			2.8	1.2	1.2	1.2
		ON 时	显示时		162	69.7	69.7	69.7
			显示结束		126	54	54	54
	T	未执行时			0.63	0.27	0.27	0.27
		执行时	时间到后		0.63	0.27	0.27	0.27
			加法运算时	K	0.63	0.27	0.27	0.27
				D	0.63	0.27	0.27	0.27
	C	未执行时			0.63	0.27	0.27	0.27
		执行时	时间到后		0.63	0.27	0.27	0.27
			加法运算时	K	0.63	0.27	0.27	0.27
				D	0.63	0.27	0.27	0.27
OUTH	T	未执行时			0.63	0.27	0.27	0.27
		执行时	时间到后		0.63	0.27	0.27	0.27
			加法运算时	K	0.63	0.27	0.27	0.27
				D	0.63	0.27	0.27	0.27
SET	未执行时				0.158	0.068	0.068	0.068
	执行时	无变化时 (ON→ON)			0.158	0.068	0.068	0.068
		变化时 (OFF→ON)			0.158	0.068	0.068	0.068
	F	未执行时			0.47	0.20	0.20	0.20
		执行时	显示时		161	69	69	69
			显示结束		0.47	0.20	0.20	0.20
RST	未执行时				0.158	0.068	0.068	0.068
	执行时	无变化时 (OFF→OFF)			0.158	0.068	0.068	0.068
		变化时 (ON→OFF)			0.158	0.068	0.068	0.068
	SM	未执行时			0.158	0.068	0.068	0.068
		执行时			0.158	0.068	0.068	0.068
	F	未执行时			0.47	0.20	0.20	0.20
		执行时	显示时		90	38	38	38
			显示结束		0.47	0.20	0.20	0.20
	T,C	未执行时			0.63	0.27	0.27	0.27
		执行时			0.63	0.27	0.27	0.27
	D	未执行时			0.24	0.10	0.10	0.10
		执行时			0.24	0.10	0.10	0.10
	Z	未执行时			0.47	0.20	0.20	0.20
		执行时			4.3	1.9	1.9	1.9
	R	未执行时			0.40	0.17	0.17	0.17
		执行时			0.40	0.17	0.17	0.17
PLS PLF	-				1.0	0.44	0.44	0.44
FF	Y	未执行时			0.47	0.20	0.20	0.20
		执行时			0.47	0.20	0.20	0.20
DELTA DELTAP	DY0	未执行时			0.47	0.20	0.20	0.20
		执行时			5.9	2.6	2.6	2.6

指令	条件（软元件）	处理时间（μs）			
		Qn	QnH	QnPH	QnPRH
SFT SFTP	未执行时	0.47	0.20	0.20	0.20
	执行时	1.66	0.71	0.71	0.71
MC	-	0.24	0.10	0.10	0.10
MCR	-	0.079	0.034	0.034	0.034
FEND END	进行出错检查	380	150	150	500
	不进行出错检查 （・电池检查） （・保险丝熔断检查） （・I/O 模块校验）	380	150	150	500
NOP	-	0.079	0.034	0.034	0.034
NOPLF PAGE	-	0.079	0.034	0.034	0.034

(2) 基本指令

未执行时的处理时间如下所示。

Q02CPU.....0.079 ×（ 各指令的步数 +1）μ s

Q02HCPU、Q06HCPU、Q12HCPU、Q25HCPU、Q02PHCPU、Q06PHCPU、
Q12PHCPU、Q25PHCPU、Q12PRHCPU、Q25PRHCPU

.....0.034 ×（ 各指令的步数 +1）μ s

指令	条件（软元件）		处理时间（μ s）			
			Qn	QnH	QnPH	QnPRH
LD =	导通时		0.24	0.10	0.10	0.10
	非导通时		0.24	0.10	0.10	0.10
AND =	未执行时		0.24	0.10	0.10	0.10
	执行时	导通时	0.24	0.10	0.10	0.10
		非导通时	0.24	0.10	0.10	0.10
OR =	未执行时		0.24	0.10	0.10	0.10
	执行时	导通时	0.24	0.10	0.10	0.10
		非导通时	0.24	0.10	0.10	0.10
LD < >	导通时		0.24	0.10	0.10	0.10
	非导通时		0.24	0.10	0.10	0.10
AND < >	未执行时		0.24	0.10	0.10	0.10
	执行时	导通时	0.24	0.10	0.10	0.10
		非导通时	0.24	0.10	0.10	0.10
OR < >	未执行时		0.24	0.10	0.10	0.10
	执行时	导通时	0.24	0.10	0.10	0.10
		非导通时	0.24	0.10	0.10	0.10
LD >	导通时		0.24	0.10	0.10	0.10
	非导通时		0.24	0.10	0.10	0.10
AND >	未执行时		0.24	0.10	0.10	0.10
	执行时	导通时	0.24	0.10	0.10	0.10
		非导通时	0.24	0.10	0.10	0.10
OR >	未执行时		0.24	0.10	0.10	0.10
	执行时	导通时	0.24	0.10	0.10	0.10
		非导通时	0.24	0.10	0.10	0.10
LD < =	导通时		0.24	0.10	0.10	0.10
	非导通时		0.24	0.10	0.10	0.10
AND < =	未执行时		0.24	0.10	0.10	0.10
	执行时	导通时	0.24	0.10	0.10	0.10
		非导通时	0.24	0.10	0.10	0.10

指令	条件（软元件）		处理时间（μs）			
			Qn	QnH	QnPH	QnPRH
OR < =	未执行时		0.24	0.10	0.10	0.10
	执行时	导通时	0.24	0.10	0.10	0.10
		非导通时	0.24	0.10	0.10	0.10
LD<	导通时		0.24	0.10	0.10	0.10
	非导通时		0.24	0.10	0.10	0.10
AND<	未执行时		0.24	0.10	0.10	0.10
	执行时	导通时	0.24	0.10	0.10	0.10
		非导通时	0.24	0.10	0.10	0.10
OR<	未执行时		0.24	0.10	0.10	0.10
	执行时	导通时	0.24	0.10	0.10	0.10
		非导通时	0.24	0.10	0.10	0.10
LD > =	导通时		0.24	0.10	0.10	0.10
	非导通时		0.24	0.10	0.10	0.10
AND > =	未执行时		0.24	0.10	0.10	0.10
	执行时	导通时	0.24	0.10	0.10	0.10
		非导通时	0.24	0.10	0.10	0.10
OR > =	未执行时		0.24	0.10	0.10	0.10
	执行时	导通时	0.24	0.10	0.10	0.10
		非导通时	0.24	0.10	0.10	0.10
LDD =	导通时		0.55	0.24	0.24	0.24
	非导通时		0.39	0.17	0.17	0.17
ANDD =	未执行时		0.39	0.17	0.17	0.17
	执行时	导通时	0.55	0.24	0.24	0.24
		非导通时	0.39	0.17	0.17	0.17
ORD =	未执行时		0.39	0.17	0.17	0.17
	执行时	导通时	0.55	0.24	0.24	0.24
		非导通时	0.55	0.24	0.24	0.24
LDD < >	导通时		0.55	0.24	0.24	0.24
	非导通时		0.55	0.24	0.24	0.24
ANDD < >	未执行时		0.39	0.17	0.17	0.17
	执行时	导通时	0.55	0.24	0.24	0.24
		非导通时	0.55	0.24	0.24	0.24
ORD < >	未执行时		0.39	0.17	0.17	0.17
	执行时	导通时	0.55	0.24	0.24	0.24
		非导通时	0.55	0.24	0.24	0.24
LDD >	导通时		0.55	0.24	0.24	0.24
	非导通时		0.55	0.24	0.24	0.24
ANDD >	未执行时		0.39	0.17	0.17	0.17
	执行时	导通时	0.55	0.24	0.24	0.24
		非导通时	0.55	0.24	0.24	0.24
ORD >	未执行时		0.39	0.17	0.17	0.17
	执行时	导通时	0.55	0.24	0.24	0.24
		非导通时	0.55	0.24	0.24	0.24
LDD < =	导通时		0.55	0.24	0.24	0.24
	非导通时		0.55	0.24	0.24	0.24
ANDD < =	未执行时		0.39	0.17	0.17	0.17
	执行时	导通时	0.55	0.24	0.24	0.24
		非导通时	0.55	0.24	0.24	0.24
ORD < =	未执行时		0.39	0.17	0.17	0.17
	执行时	导通时	0.55	0.24	0.24	0.24
		非导通时	0.55	0.24	0.24	0.24

指令	条件（软元件）			处理时间（μs）			
				Qn	QnH	QnPH	QnPRH
LDD<	导通时			0.55	0.24	0.24	0.24
	非导通时			0.55	0.24	0.24	0.24
ANDD<	未执行时			0.39	0.17	0.17	0.17
	执行时		导通时	0.55	0.24	0.24	0.24
			非导通时	0.55	0.24	0.24	0.24
ORD<	未执行时			0.39	0.17	0.17	0.17
	执行时		导通时	0.55	0.24	0.24	0.24
			非导通时	0.55	0.24	0.24	0.24
LDD > =	导通时			0.55	0.24	0.24	0.24
	非导通时			0.55	0.24	0.24	0.24
ANDD > =	未执行时			0.39	0.17	0.17	0.17
	执行时		导通时	0.55	0.24	0.24	0.24
			非导通时	0.55	0.24	0.24	0.24
ORD > =	未执行时			0.39	0.17	0.17	0.17
	执行时		导通时	0.55	0.24	0.24	0.24
			非导通时	0.55	0.24	0.24	0.24
LDE = *1	单精度	导通时		93	40	6.4	6.4
				14.9	6.4		
		非导通时		92	40	6.4	6.4
				14.9	6.4		
	双精度	导通时		93	40	-	-
				14.9	6.4		
		非导通时		92	40	-	-
				14.9	6.4		
ANDE = *1	单精度	未执行时		0.55	0.24	0.24	0.24
		执行时	导通时	93	40	6.4	6.4
				14.9	6.4		
			非导通时	92	40	6.4	6.4
				14.9	6.4		
	双精度	未执行时		-	-	-	-
		执行时	导通时	93	40	-	-
				14.9	6.4		
			非导通时	92	40	-	-
				14.9	6.4		
ORE = *1	单精度	未执行时		0.55	0.24	0.24	0.24
		执行时	导通时	93	40	6.4	6.4
				14.9	6.4		
			非导通时	92	40	6.4	6.4
				14.9	6.4		
	双精度	未执行时		0.55	0.24	-	-
		执行时	导通时	93	40	-	-
				14.9	6.4		
			非导通时	92	40	-	-
				14.9	6.4		

*1: Qn/QnH 的处理时间根据 CPU 模块的序列号的不同而有所不同。
上栏：序列号的前 5 位数为“05031”以前
下栏：序列号的前 5 位数为“05032”以后
但是，未执行时的条件不存在上下栏的区别。

指令	条件（软元件）			处理时间（μs）			
				Qn	QnH	QnPH	QnPRH
LDE＜＞ *1	单精度	导通时		92	40	6.4	6.4
				14.9	6.4		
		非导通时		92	40	6.4	6.4
				14.9	6.4		
	双精度	导通时		92	40	－	－
				14.9	6.4		
		非导通时		92	40	－	－
				14.9	6.4		
ANDE＜＞ *1	单精度	未执行时		0.55	0.24	0.24	0.24
		执行时	导通时	92	40	6.4	6.4
				14.9	6.4		
			非导通时	93	40	6.4	6.4
				14.9	6.4		
	双精度	未执行时		0.55	0.24	－	－
		执行时	导通时	92	40	－	－
				14.9	6.4		
			非导通时	92	40	－	－
				14.9	6.4		
ORE＜＞ *1	单精度	未执行时		0.55	0.24	0.24	0.24
		执行时	导通时	93	40	6.4	6.4
				14.9	6.4		
			非导通时	92	40	6.4	6.4
				14.9	6.4		
	双精度	未执行时		0.55	0.24	－	－
		执行时	导通时	93	40	－	－
				14.9	6.4		
			非导通时	92	40	－	－
				14.9	6.4		
LDE＞ *1	单精度	未执行时		92	40	6.4	6.4
		导通时		14.9	6.4		
				92	40	6.4	6.4
				14.9	6.4		
	双精度	非导通时		92	40	－	－
				14.9	6.4		
				92	40	－	－
				14.9	6.4		
ANDE＞ *1	单精度	未执行时		0.55	0.24	0.24	0.24
		执行时	导通时	92	40	6.4	6.4
				14.9	6.4		
			非导通时	93	40	6.4	6.4
				14.9	6.4		
	双精度	未执行时		0.55	0.24	－	－
		执行时	导通时	92	40	－	－
				14.9	6.4		
			非导通时	92	40	－	－
				14.9	6.4		

*1: Qn/QnH 的处理时间根据 CPU 模块的序列号的不同而有所不同。
上栏：序列号的前 5 位数为“05031”以前
下栏：序列号的前 5 位数为“05032”以后
但是，未执行时的条件不存在上下栏的区别。

指令	条件（软元件）			处理时间（μs）			
				Qn	QnH	QnPH	QnPRH
ORE > *1	单精度	未执行时		0.55	0.24	0.24	0.24
		执行时	导通时	93	40	6.4	6.4
				14.9	6.4		
			非导通时	92	40	6.4	6.4
				14.9	6.4		
	双精度	未执行时		0.55	0.24	-	-
		执行时	导通时	93	40	-	-
				14.9	6.4		
			非导通时	92	40	-	-
				14.9	6.4		
LDE < = *1	单精度	导通时		93	40	6.4	6.4
				14.9	6.4		
		非导通时		92	40	6.4	6.4
				14.9	6.4		
	双精度	导通时		93	40	-	-
				14.9	6.4		
		非导通时		92	40	-	-
				14.9	6.4		
ANDE < = *1	单精度	未执行时		0.55	0.24	0.24	0.24
		执行时	导通时	92	40	6.4	6.4
				14.9	6.4		
			非导通时	92	40	6.4	6.4
				14.9	6.4		
	双精度	未执行时		0.55	0.24	-	-
		执行时	导通时	92	40	-	-
				14.9	6.4		
			非导通时	92	40	-	-
				14.9	6.4		
ORE < = *1	单精度	未执行时		0.55	0.24	0.24	0.24
		执行时	导通时	92	40	6.4	6.4
				14.9	6.4		
			非导通时	92	40	6.4	6.4
				14.9	6.4		
	双精度	未执行时		0.55	0.24	-	-
		执行时	导通时	92	40	-	-
				14.9	6.4		
			非导通时	92	40	-	-
				14.9	6.4		
LDE< *1	单精度	导通时		92	40	6.4	6.4
				14.9	6.4		
		非导通时		92	40	6.4	6.4
				14.9	6.4		
	双精度	导通时		92	40	-	-
				14.9	6.4		
		非导通时		92	40	-	-
				14.9	6.4		

*1: Qn/QnH 的处理时间根据 CPU 模块的序列号的不同而有所不同。
上栏：序列号的前 5 位数为“05031”以前
下栏：序列号的前 5 位数为“05032”以后
但是，未执行时的条件不存在上下栏的区别。

指令	条件（软元件）			处理时间（μs）			
				Qn	QnH	QnPH	QnPRH
ANDE< *1	单精度	未执行时		0.55	0.24	0.24	0.24
		执行时	导通时	92	40	6.4	6.4
				14.9	6.4		
			非导通时	92	40	6.4	6.4
				14.9	6.4		
	双精度	未执行时		0.55	0.24	-	-
		执行时	导通时	92	40	-	-
				14.9	6.4		
			非导通时	92	40	-	-
				14.9	6.4		
ORE< *1	单精度	未执行时		0.55	0.24	0.24	0.24
		执行时	导通时	93	40	6.4	6.4
				14.9	6.4		
			非导通时	92	40	6.4	6.4
				14.9	6.4		
	双精度	未执行时		0.55	0.24	-	-
		执行时	导通时	93	40	-	-
				14.9	6.4		
			非导通时	92	40	-	-
				14.9	6.4		
LDE > = *1	单精度	导通时		93	40	6.4	6.4
				14.9	6.4		
		非导通时		92	40	6.4	6.4
				14.9	6.4		
	双精度	导通时		93	40	-	-
				14.9	6.4		
		非导通时		92	40	-	-
				14.9	6.4		
ANDE > = *1	单精度	未执行时		0.55	0.24	0.24	0.24
		执行时	导通时	92	40	6.4	6.4
				14.9	6.4		
			非导通时	92	40	6.4	6.4
				14.9	6.4		
	双精度	未执行时		0.55	0.24	-	-
		执行时	导通时	92	40	-	-
				14.9	6.4		
			非导通时	92	40	-	-
				14.9	6.4		

*1: Qn/QnH 的处理时间根据 CPU 模块的序列号的不同而有所不同。
上栏：序列号的前 5 位数为“05031”以前
下栏：序列号的前 5 位数为“05032”以后
但是，未执行时的条件不存在上下栏的区别。

指令	条件（软元件）			处理时间（μs）			
				Qn	QnH	QnPH	QnPRH
ORE > = *1	单精度	未执行时		0.55	0.24	0.24	0.24
		执行时	导通时	92	40	6.4	6.4
				14.9	6.4		
			非导通时	92	40	6.4	6.4
				14.9	6.4		
	双精度	未执行时		0.55	0.24	-	-
		执行时	导通时	92	40	-	-
				14.9	6.4		
			非导通时	92	40	-	-
				14.9	6.4		
LD$ =	导通时			38	16	16	16
	非导通时			34	15	15	15
AND$ =	未执行时			0.56	0.23	0.23	0.23
	执行时		导通时	39	17	17	17
			非导通时	32	14	14	14
OR$ =	未执行时			0.56	0.24	0.24	0.24
	执行时		导通时	40	17	17	17
			非导通时	33	14	14	14
LD$ < >	导通时			32	14	14	14
	非导通时			40	17	17	17
AND$ < >	未执行时			0.56	0.23	0.23	0.23
	执行时		导通时	33	14	14	14
			非导通时	39	17	17	17
OR$ < >	未执行时			0.56	0.24	0.24	0.24
	执行时		导通时	32	14	14	14
			非导通时	39	17	17	17
LD$ >	导通时			32	14	14	14
	非导通时			40	17	17	17

*1: Qn/QnH 的处理时间根据 CPU 模块的序列号的不同而有所不同。
上栏：序列号的前 5 位数为“05031”以前
下栏：序列号的前 5 位数为“05032”以后
但是，未执行时的条件不存在上下栏的区别。

指令	条件（软元件）		处理时间（μs）			
			Qn	QnH	QnPH	QnPRH
AND$ >	未执行时		0.56	0.23	0.23	0.23
	执行时	导通时	33	14	14	14
		非导通时	39	17	17	17
OR$ >	未执行时		0.56	0.24	0.24	0.24
	执行时	导通时	32	14	14	14
		非导通时	39	17	17	17
LD$ < =	导通时		40	17	17	17
	非导通时		32	14	14	14
AND$ < =	未执行时		0.56	0.23	0.23	0.23
	执行时	导通时	39	17	17	17
		非导通时	32	14	14	14
OR$ < =	未执行时		0.56	0.24	0.24	0.24
	执行时	导通时	40	17	17	17
		非导通时	33	14	14	14
LD$<	导通时		32	14	14	14
	非导通时		40	17	17	17
AND$<	未执行时		0.56	0.23	0.23	0.23
	执行时	导通时	32	14	14	14
		非导通时	39	16	16	16
OR$<	未执行时		0.56	0.24	0.24	0.24
	执行时	导通时	32	14	14	14
		非导通时	39	16	16	16
LD$ > =	导通时		40	17	17	17
	非导通时		32	14	14	14
AND$ > =	未执行时		0.56	0.23	0.23	0.23
	执行时	导通时	39	16	16	16
		非导通时	32	14	14	14
OR$ > =	未执行时		0.56	0.24	0.24	0.24
	执行时	导通时	39	17	17	17
		非导通时	32	14	14	14
BKCMP = (S1) (S2) (D) n	n = 1		48	21	21	21
BKCMP = P (S1) (S2) (D) n	n = 96		142	61	61	61
BKCMP < > (S1) (S2) (D) n	n = 1		48	21	21	21
BKCMP < >P (S1) (S2) (D) n	n = 96		150	65	65	65
BKCMP > (S1) (S2) (D) n	n = 1		48	21	21	21
BKCMP >P (S1) (S2) (D) n	n = 96		142	61	61	61
BKCMP > = (S1) (S2) (D) n	n = 1		48	21	21	21
BKCMP > = P (S1) (S2) (D) n	n = 96		150	65	65	65
BKCMP< (S1) (S2) (D) n	n = 1		48	21	21	21
BKCMP< P (S1) (S2) (D) n	n = 96		158	68	68	68
BKCMP < = (S1) (S2) (D) n	n = 1		48	21	21	21
BKCMP < = P (S1) (S2) (D) n	n = 96		150	65	65	65

指令	条件（软元件）	处理时间（μs）			
		Qn	QnH	QnPH	QnPRH
+ (S) (D) + P (S) (D)	执行时	0.39	0.17	0.17	0.17
+ (S1) (S2) (D) + P (S1) (S2) (D)	执行时	0.47	0.20	0.20	0.20
- (S) (D) - P (S) (D)	执行时	0.39	0.17	0.17	0.17
- (S1) (S2) (D) - P (S1) (S2) (D)	执行时	0.47	0.20	0.20	0.20
D + (S) (D) D + P (S) (D)	执行时	0.71	0.31	0.31	0.31
D + (S1) (S2) (D) D + P (S1) (S2) (D)	执行时	0.79	0.34	0.34	0.34
D - (S) (D) D - P (S) (D)	执行时	0.71	0.30	0.30	0.30
D - (S1) (S2) (D) D - P (S1) (S2) (D)	执行时	0.79	0.34	0.34	0.34
* (S1) (S2) (D) * P (S1) (S2) (D)	执行时	0.47	0.20	0.20	0.20
/ (S1) (S2) (D) / P (S1) (S2) (D)	-	2.7	1.2	1.2	1.2
D * (S1) (S2) (D) D * P (S1) (S2) (D)	-	7.9	3.4	3.4	3.4
D / (S1) (S2) (D) D / P (S1) (S2) (D)	-	14	6.1	6.1	6.1
B + (S) (D) B + P (S) (D)	-	2.2	1.0	1.0	1.0
B + (S1) (S2) (D) B + P (S1) (S2) (D)	-	5.0	2.2	2.2	2.2
B - (S) (D) B - P (S) (D)	-	2.0	0.9	0.9	0.9
B - (S1) (S2) (D) B - P (S1) (S2) (D)	-	4.9	2.1	2.1	2.1
DB + (S) (D) DB + P (S) (D)	-	12	5.0	5.0	5.0
DB + (S1) (S2) (D) DB + P (S1) (S2) (D)	-	12	5.3	5.3	5.3
DB - (S) (D) DB - P (S) (D)	-	11	4.8	4.8	4.8
DB - (S1) (S2) (D) DB - P (S1) (S2) (D)	-	12	5.2	5.2	5.2
B * (S1) (S2) (D) B * P (S1) (S2) (D)	-	3.7	1.6	1.6	1.6

指令	条件（软元件）		处理时间（μs）			
			Qn	QnH	QnPH	QnPRH
B / (S1) (S2) (D) B / P (S1) (S2) (D)	－		3.8	1.6	1.6	1.6
DB * (S1) (S2) (D) DB * P (S1) (S2) (D)	－		24	10	10	10
DB / (S1) (S2) (D) DB / P (S1) (S2) (D)	－		27	12	12	12
E+ (S) (D) E+ P (S) (D)	单精度	(S) = 0, (D) = 0	1.8	0.78	0.78	0.78
		(S) = 2^{127}, (D) = 2^{127}	1.8	0.78	0.78	0.78
	双精度	(S) = 0, (D) = 0	203	87	－	－
		(S) = 2^{127}, (D) = 2^{127}	203	87	－	－
E+ (S1) (S2) (D) E+ P (S1) (S2) (D)	单精度	(S1) = 0, (S2) = 0	2.4	1.1	1.1	1.1
		(S1) = 2^{127}, (S2) = 2^{127}	2.4	1.1	1.1	1.1
	双精度	(S1) = 0, (S2) = 0	209	90	－	－
		(S1) = 2^{127}, (S2) = 2^{127}	209	90	－	－
E- (S) (D) E- P (S) (D)	单精度	(S) = 0, (D) = 0	1.8	0.78	0.78	0.78
		(S) = 2^{127}, (D) = 2^{127}	1.8	0.78	0.78	0.78
	双精度	(S) = 0, (D) = 0	202	87	－	－
		(S) = 2^{127}, (D) = 2^{127}	202	87	－	－
E- (S1) (S2) (D) E- P (S1) (S2) (D)	单精度	(S1) = 0, (S2) = 0	2.4	1.1	1.1	1.1
		(S1) = 2^{127}, (S2) = 2^{127}	2.4	1.1	1.1	1.1
	双精度	(S1) = 0, (S2) = 0	210	90	－	－
		(S1) = 2^{127}, (S2) = 2^{127}	210	90	－	－
E* (S1) (S2) (D) E* P (S1) (S2) (D)	单精度	(S1) = 0, (S2) = 0	2.4	1.1	1.1	1.1
		(S1) = 2^{126}, (S2) = 2^{127}	2.4	1.1	1.1	1.1
	双精度	(S1) = 0, (S2) = 0	222	96	－	－
		(S1) = 2^{126}, (S2) = 2^{127}	222	96	－	－
E/ (S1) (S2) (D) E/ P (S1) (S2) (D)	单精度	(S1) = 0, (S2) = 1	12	5.2	5.2	5.2
		(S1) = 2^{127}, (S2) = -2^{126}	12	5.2	5.2	5.2
	双精度	(S1) = 0, (S2) = 1	369	159	－	－
		(S1) = 2^{127}, (S2) = -2^{126}	369	159	－	－

指令	条件（软元件）		处理时间（μs）			
			Qn	QnH	QnPH	QnPRH
$+ (S) (D) $+ P (S) (D)	－		68	29	29	29
$+ (S1) (S2) (D) $+ P (S1) (S2) (D)	－		81	35	35	35
INC INCP	－		0.32	0.14	0.14	0.14
DINC DINCP	－		0.47	0.20	0.20	0.20
DEC DECP	－		0.32	0.14	0.14	0.14
DDEC DDECP	－		0.47	0.20	0.20	0.20
BCD BCDP	－		1.1	0.48	0.48	0.48
DBCD DBCDP	－		3.2	1.4	1.4	1.4
BIN BINP	－		1.0	0.44	0.44	0.44
DBIN DBINP	－		1.9	0.82	0.82	0.82
INT INTP	单精度	(S) = 0	3.2	1.4	1.4	1.4
		(S) = 32766.5	3.2	1.4	1.4	1.4
	双精度	(S) = 0	22	9.3	－	－
		(S) = 32766.5	22	9.3	－	－
DINT DINTP	单精度	(S) = 0	2.5	1.1	1.1	1.1
		(S) = 1234567890.3	2.5	1.1	1.1	1.1
	双精度	(S) = 0	24	10	－	－
		(S) = 1234567890.3	24	10	－	－
FLT FLTP	单精度	(S) = 0	2.1	0.92	0.92	0.92
		(S) = $7FFF_H$	2.1	0.92	0.92	0.92
	双精度	(S) = 0	22	9.6	－	－
		(S) = $7FFF_H$	22	9.6	－	－
DFLT DFLTP	单精度	(S) = 0	2.1	0.88	0.88	0.88
		(S) = $7FFFFFFF_H$	2.1	0.88	0.88	0.88
	双精度	(S) = 0	26	11	－	－
		(S) = $7FFFFFFF_H$	26	11	－	－

指令	条件（软元件）	处理时间（μs）			
		Qn	QnH	QnPH	QnPRH
DBL DBLP	-	4.5	1.9	1.9	1.9
WORD WORDP	-	4.7	2.0	2.0	2.0
GRY GRYP	-	4.7	2.0	2.0	2.0
DGRY DGRYP	-	5.3	2.3	2.3	2.3
GBIN GBINP	-	18	7.7	7.7	7.7
DGBIN DGBINP	-	32	14	14	14
NEG NEGP	-	3.6	1.6	1.6	1.6
DNEG DNEGP	-	4.3	1.8	1.8	1.8
ENEG ENEGP	-	3.9	1.7	1.7	1.7
BKBCD (S) (D) n	n = 1	38	17	17	17
BKBCDP (S) (D) n	n = 96	99	43	43	43
BKBIN (S) (D) n	n = 1	38	17	17	17
BKBINP (S) (D) n	n = 96	99	43	43	43
MOV MOVP	(S) = D0, (D) = D1	0.24	0.10	0.10	0.10
	(S) = D0, (D) = J1\W1	-	-	-	-
		-	-	-	-
		140*1	60*1	60*1	60*1
DMOV DMOVP	(S) = D0, (D) = D1	0.47	0.20	0.20	0.20
	(S) = D0, (D) = J1\W1	-	-	-	-
		-	-	-	-
		147*1	64*1	64*1	64*1
EMOV EMOVP	-	0.63	0.27	0.27	0.27
$MOV $MOVP	-	40	17	17	17
CML CMLP	-	0.40	0.17	0.17	0.17
DCML DCMLP	-	0.55	0.24	0.24	0.24

*1: 上栏表示使用 A38B/A1S38B 以及扩展基板时的处理时间。
中栏表示使用 A38HB/A1S38HB 时的处理时间。
下栏表示使用 Q312B 时的处理时间。

指令	条件（软元件）	处理时间（μs）			
		Qn	QnH	QnPH	QnPRH
BMOV (S) (D) n	n = 1	17	7.1	7.1	7.1
BMOVP (S) (D) n	n = 96	32	14	14	14
FMOV (S) (D) n	n = 1	6.7	2.9	2.9	2.9
FMOVP (S) (D) n	n = 96	14	6.1	6.1	6.1
XCH XCHP DXCH DXCHP	-	1.3	0.54	0.54	0.54
BXCH (D1) (D2) n	n = 1	31	13	13	13
BXCHP (D1) (D2) n	n = 96	84	36	36	36
SWAP SWAPP	-	3.7	1.6	1.6	1.6
CJ	-	3.2	1.4	1.4	1.4
SCJ	-	3.2	1.4	1.4	1.4
JMP	-	3.2	1.4	1.4	1.4
GOEND	-	0.39	0.34	0.34	0.34
DI	-	0.95	0.41	0.41	0.41
EI	-	1.3	0.54	0.54	0.54
IMASK	-	11	4.6	4.6	4.6
IRET	-	1.6	0.68	0.68	0.68
RFS	n = 1	6.7	4.7	4.7	4.7
RFSP	n = 96	19	13	13	13
UDCNT1	-	15	6.5	6.5	-
UDCNT2	-	16	6.8	6.8	-
TTMR	-	10	4.4	4.4	-
STMR	-	20	7.1	7.1	-
ROTC	-	26	11	11	-
RAMP	-	18	7.7	7.7	-
SPD	-	19	8.3	8.3	-
PLSY	-	10	4.5	4.5	-
PWM	-	9.1	3.9	3.9	-
MTR	-	11	4.9	4.9	-

(3) 应用指令

未执行时的处理时间如下所示。

Q02CPU .....0.079 ×（ 各指令的步数 +1）μ s

Q02HCPU、Q06HCPU、Q12HCPU、Q25HCPU、Q02PHCPU、Q06PHCPU、
Q12PHCPU、Q25PHCPU、Q12PRHCPU、Q25PRHCPU

.....0.034 ×（ 各指令的步数 +1）μ s

指令	条件（软元件）	处理时间（μ s）			
		Qn	QnH	QnPH	QnPRH
WAND ⓢ ⓓ WANDP ⓢ ⓓ	执行时	0.39	0.17	0.17	0.17
WAND ⓢ1 ⓢ2 ⓓ WANDP ⓢ1 ⓢ2 ⓓ	执行时	0.47	0.20	0.20	0.20
DAND ⓢ ⓓ DANDP ⓢ ⓓ	执行时	0.71	0.31	0.31	0.31
DAND ⓢ1 ⓢ2 ⓓ DANDP ⓢ1 ⓢ2 ⓓ	执行时	0.79	0.34	0.34	0.34
BKAND ⓢ1 ⓢ2 ⓓ n	n = 1	36	16	16	16
BKANDP ⓢ1 ⓢ2 ⓓ n	n = 96	74	32	32	32
WOR ⓢ ⓓ WORP ⓢ ⓓ	执行时	0.40	0.17	0.17	0.17
WOR ⓢ1 ⓢ2 ⓓ WORP ⓢ1 ⓢ2 ⓓ	执行时	0.47	0.20	0.20	0.20
DOR ⓢ ⓓ DORP ⓢ ⓓ	执行时	0.71	0.31	0.31	0.31
DOR ⓢ1 ⓢ2 ⓓ DORP ⓢ1 ⓢ2 ⓓ	执行时	0.79	0.34	0.34	0.34
BKOR ⓢ1 ⓢ2 ⓓ n	n = 1	36	16	16	16
BKORP ⓢ1 ⓢ2 ⓓ n	n = 96	74	32	32	32
WXOR ⓢ ⓓ WXORP ⓢ ⓓ	执行时	0.39	0.17	0.17	0.17
WXOR ⓢ1 ⓢ2 ⓓ WXORP ⓢ1 ⓢ2 ⓓ	执行时	0.47	0.20	0.20	0.20
DXOR ⓢ ⓓ DXORP ⓢ ⓓ	执行时	0.71	0.31	0.31	0.31
DXOR ⓢ1 ⓢ2 ⓓ DXORP ⓢ1 ⓢ2 ⓓ	执行时	0.79	0.34	0.34	0.34
BKXOR ⓢ1 ⓢ2 ⓓ n	n = 1	36	16	16	16
BKXORP ⓢ1 ⓢ2 ⓓ n	n = 96	74	32	32	32

指令	条件（软元件）	处理时间（μs）			
		Qn	QnH	QnPH	QnPRH
WXNR (S) (D) WXNRP (S) (D)	执行时	0.40	0.17	0.17	0.17
WXNR (S1) (S2) (D) WXNRP (S1) (S2) (D)	执行时	0.47	0.20	0.20	0.20
DXNR (S) (D) DXNRP (S) (D)	执行时	0.71	0.31	0.31	0.31
DXNR (S1) (S2) (D) DXNRP (S1) (S2) (D)	执行时	0.79	0.34	0.34	0.34
BKXNR (S1) (S2) (D) n	n = 1	36	16	16	16
BKXNRP (S1) (S2) (D) n	n = 96	74	32	32	32
ROR (D) n	n = 1	2.0	0.85	0.85	0.85
RORP (D) n	n = 15	2.0	0.85	0.85	0.85
RCR (D) n	n = 1	1.6	0.68	0.68	0.68
RCRP (D) n	n = 15	1.6	0.68	0.68	0.68
ROL (D) n	n = 1	2.0	0.85	0.85	0.85
ROLP (D) n	n = 15	2.0	0.85	0.85	0.85
RCL (D) n	n = 1	1.6	0.68	0.68	0.68
RCLP (D) n	n = 15	1.6	0.68	0.68	0.68
DROR (D) n	n = 1	3.9	1.7	1.7	1.7
DRORP (D) n	n = 31	4.0	1.7	1.7	1.7
DRCR (D) n	n = 1	4.3	1.8	1.8	1.8
DRCRP (D) n	n = 31	4.3	1.9	1.9	1.9
DROL (D) n	n = 1	3.9	1.7	1.7	1.7
DROLP (D) n	n = 31	4.0	1.7	1.7	1.7
DRCL (D) n	n = 1	4.3	1.8	1.8	1.8
DRCLP (D) n	n = 31	4.3	1.9	1.9	1.9
SFR (D) n	n = 1	1.7	0.75	0.75	0.75
SFRP (D) n	n = 15	2.0	0.85	0.85	0.85
SFL (D) n	n = 1	1.7	0.75	0.75	0.75
SFLP (D) n	n = 15	2.0	0.85	0.85	0.85
BSFR (D) n	n = 1	20	8.6	8.6	8.6
BSFRP (D) n	n = 96	24	10	10	10
BSFL (D) n	n = 1	20	8.5	8.5	8.5
BSFLP (D) n	n = 96	23	10	10	10
DSFR (D) n	n = 1	1.3	0.58	0.58	0.58
DSFRP (D) n	n = 96	25	11	11	11
DSFL (D) n	n = 1	1.3	0.58	0.58	0.58
DSFLP (D) n	n = 96	26	11	11	11

指令	条件（软元件）		处理时间（μs）			
			Qn	QnH	QnPH	QnPRH
BSET (D) n	n = 1		7.6	3.3	3.3	3.3
BSETP (D) n	n = 15		7.6	3.3	3.3	3.3
BRST (D) n	n = 1		7.6	3.3	3.3	3.3
BRSTP (D) n	n = 15		7.6	3.3	3.3	3.3
TEST (S1) (S2) (D) TESTP (S1) (S2) (D)	-		8.2	3.5	3.5	3.5
DTEST (S1) (S2) (D) DTESTP (S1) (S2) (D)	-		9.2	3.9	3.9	3.9
BKRST (D) n	n = 1		18	7.8	7.8	7.8
BKRSTP (D) n	n = 96		19	8.2	8.2	8.2
SER (S1) (S2) (D) n SERP (S1) (S2) (D) n	n = 1	全部一致	22	9.6	9.6	9.6
		全部不一致	21	8.9	8.9	8.9
	n = 96	全部一致	115	49	49	49
		全部不一致	133	57	57	57
DSER (S1) (S2) (D) n DSERP (S1) (S2) (D) n	n = 1	全部一致	23	9.9	9.9	9.9
		全部不一致	23	9.7	9.7	9.7
	n = 96	全部一致	142	61	61	61
		全部不一致	132	57	57	57
SUM SUMP	(S) = 0 (S) = FFFF		3.9	1.7	1.7	1.7
DSUM DSUMP	(S) = 0		4.7	2.0	2.0	2.0
	(S) = FFFFFFFF$_H$		12	5.0	5.0	5.0
DECO (S) (D) n	n = 2		20	8.6	8.6	8.6
DECOP (S) (D) n	n = 8		27	12	12	12
ENCO (S) (D) n ENCOP (S) (D) n	n = 2	M1 = ON	21	9.1	9.1	9.1
		M4 = ON	21	9.1	9.1	9.1
	n = 8	M1 = ON	28	12	12	12
		M256 = ON	26	11	11	11
SEG SEGP	-		1.3	0.54	0.54	0.54
DIS (S) (D) n	n = 1		18	7.7	7.7	7.7
DISP (S) (D) n	n = 4		19	8.3	8.3	8.3
UNI (S) (D) n	n = 1		21	8.9	8.9	8.9
UNIP (S) (D) n	n = 4		23	9.7	9.7	9.7
NDIS (S1) (D) (S2) NDISP (S1) (D) (S2)	-		41	18	18	18
NUNI (S1) (D) (S2) NUNIP (S1) (D) (S2)	-		42	18	18	18
WTOB (S) (D) n	n = 1		47	20	20	20
WTOBP (S) (D) n	n = 96		99	43	43	43
BTOW (S) (D) n	n = 1		45	19	19	19
BTOWP (S) (D) n	n = 96		89	38	38	38

指令	条件（软元件）	处理时间（μs）			
		Qn	QnH	QnPH	QnPRH
MAX (S) (D) n	n = 1	17	7.1	7.1	7.1
MAXP (S) (D) n	n = 96	136	59	59	59
MIN (S) (D) n	n = 1	17	7.1	7.1	7.1
MINP (S) (D) n	n = 96	159	69	69	69
DMAX (S) (D) n	n = 1	27	12	12	12
DMAXP (S) (D) n	n = 96	181	78	78	78
DMIN (S) (D) n	n = 1	27	12	12	12
DMINP (S) (D) n	n = 96	112	48	48	48
SORT (S1) n (S2) (D1) (D2)	n = 1	16	7.1	7.1	7.1
	n = 96	14	6.2	6.2	6.2
DSORT (S1) n (S2) (D1) (D2)	n = 1	17	7.1	7.1	7.1
	n = 96	16	6.8	6.8	6.8
WSUM (S) (D) n	n = 1	16.4	7.1	7.1	7.1
WSUMP (S) (D) n	n = 96	68.4	29.5	29.5	29.5
DWSUM (S) (D) n	n = 1	18.9	8.2	8.2	8.2
DWSUMP (S) (D) n	n = 96	130.4	56.1	56.1	56.1
FOR n	n = 0	2.3	1.0	1.0	1.0
NEXT	-	3.3	1.4	1.4	1.4
BREAK	-	11	4.6	4.6	4.6
BREAKP					
CALL Pn	文件内指针	2.1	0.88	0.88	0.88
CALLP Pn	公共指针	33	14	14	14
CALL Pn (S1) ～ (S5)	-	135	58	58	58
CALLP Pn (S1) ～ (S5)					
RET	返回至本程序	2.9	1.3	1.3	1.3
	返回至其它程序	20	8.5	8.5	8.5
FCALL Pn	文件内指针	3.6	1.6	1.6	1.6
FCALLP Pn	公共指针	20	8.7	8.7	8.7
FCALL Pn (S1) ～ (S5) FCALLP Pn (S1) ～ (S5)	-	134	57	57	57
ECALL * Pn ECALLP * Pn *: 程序名	-	77	33	33	33
ECALL * Pn (S1) ～ (S5) ECALLP * Pn (S1) ～ (S5) *: 程序名	-	162	70	70	70
EFCALL * Pn EFCALLP * Pn *: 程序名	-	78	34	34	34
EFCALL * Pn (S1) ～ (S5) EFCALLP * Pn (S1) ～ (S5) *: 程序名	-	200	86	86	86

指令	条件（软元件）		处理时间（μs）			
			Qn	QnH	QnPH	QnPRH
COM	－		55	16	16	16
IX	－		12	5.2	5.2	5.2
IXEND	－		4.7	2.0	2.0	2.0
IXDEV+IXSET	触点数 1		48	21	21	21
	触点数 14		93	40	40	40
FIFW	数据数 0		11	4.5	4.5	4.5
FIFWP	数据数 96		11	4.5	4.5	4.5
FIFR	数据数 1		13	5.6	5.6	5.6
FIFRP	数据数 96		32	14	14	14
FPOP	数据数 1		16	7.0	7.0	7.0
FPOPP	数据数 96		16	7.0	7.0	7.0
FINS	数据数 0		20	8.4	8.4	8.4
FINSP	数据数 96		36	15	15	15
FDEL	数据数 1		19	7.5	7.5	7.5
FDELP	数据数 96		39	15	15	15
FROM n1 n2 (D) n3 FROMP n1 n2 (D) n3 *1	n3 = 1		－	－	－	－
			－	－	－	－
			47	22	22	22
	n3 = 1000		－	－	－	－
			－	－	－	－
			476	437	437	437
DFRO n1 n2 (D) n3 DFROP n1 n2 (D) n3 *1	n3 = 1		－	－	－	－
			－	－	－	－
			51	24	24	24
	n3 = 500		－	－	－	－
			－	－	－	－
			478	437	437	437
TO n1 n2 (S) n3 TOP n1 n2 (S) n3 *1	n3 = 1		－	－	－	－
			－	－	－	－
			48	20	20	20
	n3 = 1000		－	－	－	－
			－	－	－	－
			479	412	412	412
DTO n1 n2 (S) n3 DTOP n1 n2 (S) n3 *1	n3 = 1		－	－	－	－
			－	－	－	－
			50	23	23	23
	n3 = 500		－	－	－	－
			－	－	－	－
			457	416	416	416
PR	SM7010N	可变 1 字符	33	11	11	－
		可变 32 字符	48	18	18	－
	SM7010FF		21	7.8	7.8	－
PRC	－		181	16	16	－
LED	显示时		－	－	－	－
	显示结束		－	－	－	－

*1: 上栏表示使用 A38B/A1S38B 以及扩展基板时的处理时间。
中栏表示使用 A38HB/A1S38HB 时的处理时间。
下栏表示使用 Q312B 时，对第 0 号插槽的 QJ71C24 执行了指令时的处理时间。
FROM/TO 指令的处理时间根据插槽数及安装模块而有所不同。
（在 QnCPU/QnHCPU 中根据扩展基板的类型其处理时间也有所不同。）

指令	条件（软元件）	处理时间（μs）			
		Qn	QnH	QnPH	QnPRH
LEDC	显示时	-	-	-	-
	显示结束	-	-	-	-
LEDR	无显示→无显示	0.40	0.17	0.17	0.17
	LED 指令执行→无显示	103	44	44	44
CHKST	-	5.8	2.5	2.5	2.5
CHK	无 1 触点出错	24	10	10	10
	无 150 触点出错	1676	721	721	721
	有 1 触点出错	88	38	38	38
CHKCIR	10 步	5.8	2.5	2.5	2.5
SLT	全内置软元件	-	-	-	-
	文件寄存器 8k 点	-	-	-	-
	SLT 执行结束	-	-	-	-
SLTR	-	-	-	-	-
STRA	开始	-	-	-	-
	STRA 执行结束	-	-	-	-
STRAR	-	-	-	-	-
PTRA	-	-	-	-	-
PTRAR	-	-	-	-	-
PTRAEXE PTRAEXEP	运行中	-	-	-	-
	跟踪中	-	-	-	-
BINDA BINDAP	(S) = 1	15	6.7	6.7	6.7
	(S) = -32768	24	10	10	10
DBINDA DBINDAP	(S) = 1	43	18	18	18
	(S) = -2147483648	86	37	37	37
BINHA BINHAP	(S) = 1	18	7.7	7.7	7.7
	(S) = $FFFF_H$	19	8.2	8.2	8.2
DBINHA DBINHAP	(S) = 1	23	10	10	10
	(S) = $FFFFFFFF_H$	24	10	10	10
BCDDA BCDDAP	(S) = 1	23	9.8	9.8	9.8
	(S) = 9999	21	8.9	8.9	8.9
DBCDDA DBCDDAP	(S) = 1	22	9.5	9.5	9.5
	(S) = 99999999	29	13	13	13
DABIN DABINP	(S) = 1	57	25	25	25
	(S) = -32768	58	25	25	25
DDABIN DDABINP	(S) = 1	92	40	40	40
	(S) = -2147483648	106	46	46	46
HABIN HABINP	(S) = 1	13	5.8	5.8	5.8
	(S) = $FFFF_H$	15	6.4	6.4	6.4
DHABIN DHABINP	(S) = 1	22	9.5	9.5	9.5
	(S) = $FFFFFFFF_H$	25	11	11	11

指令	条件（软元件）		处理时间（μs）			
			Qn	QnH	QnPH	QnPRH
DABCD DABCDP	(S) = 1		16	6.9	6.9	6.9
	(S) = 9999		17	7.2	7.2	7.2
DDABCD DDABCDP	(S) = 1		25	11	11	11
	(S) = 99999999		29	13	13	13
COMRD COMRDP	-		40	17	17	17
LEN LENP	1 字符		18	8.0	8.0	8.0
	96 字符		86	37	37	37
STR STRP	-		53	23	23	23
DSTR DSTRP	-		123	53	53	53
VAL VALP	-		95	41	41	41
DVAL DVALP	-		166	72	72	72
ESTR ESTRP	-		564	243	243	243
EVAL EVALP	小数点形式全 2 位数指定		100	43	43	43
	指数形式全 6 位数指定		127	55	55	55
ASC (S) (D) n	n = 1		64	28	28	28
ASCP (S) (D) n	n = 96		289	125	125	125
HEX (S) (D) n	n = 1		60	26	26	26
HEXP (S) (D) n	n = 96		343	148	148	148
RIGHT (S) (D) n	n = 1		49	21	21	21
RIGHTP (S) (D) n	n = 96		131	56	56	56
LEFT (S) (D) n	n = 1		50	21	21	21
LEFTP (S) (D) n	n = 96		131	56	56	56
MIDR MIDRP	-		53	23	23	23
MIDW MIDWP	-		128	55	55	55
INSTR INSTRP	无一致		58	25	25	25
	有一致	起始	55	24	24	24
		最终	58	25	25	25

指令	条件（软元件）		处理时间（μs）			
			Qn	QnH	QnPH	QnPRH
EMOD EMODP	-		527	227	227	227
EREXP EREXPP	-		1656	713	713	713
SIN SINP	单精度		115	50	50	50
	双精度		1945	837	-	-
COS COSP	单精度		122	53	53	53
	双精度		2618	1127	-	-
TAN TANP	单精度		123	53	53	53
	双精度		2618	1127	-	-
ASIN ASINP	单精度		111	48	48	48
	双精度		2491	1072	-	-
ACOS ACOSP	单精度		115	49	49	49
	双精度		2367	1019	-	-
ATAN ATANP	单精度		157	68	68	68
	双精度		3140	1352	-	-
RAD RADP	单精度		17	7.2	7.2	7.2
	双精度		24	10	-	-
DEG DEGP	单精度		17	7.2	7.2	7.2
	双精度		23	9.9	-	-
SQR SQRP	单精度		28	12	12	12
	双精度		1812	780	-	-
EXP EXPP	单精度	Ⓢ = -10	129	56	56	56
		Ⓢ = 1				
	双精度	Ⓢ = -10	2386	1026	-	-
		Ⓢ = 1				
LOG LOGP	单精度	Ⓢ = 1	113	49	49	49
		Ⓢ = 10				
	双精度	Ⓢ = 1	2146	924	-	-
		Ⓢ = 10				
RND RNDP	-		3.9	1.7	1.7	1.7
SRND SRNDP	-		3.5	1.5	1.5	1.5

指令	条件（软元件）	处理时间（μs）			
		Qn	QnH	QnPH	QnPRH
BSQR BSQRP	(S) = 0	6.2	2.7	2.7	2.7
	(S) = 9999	38	16	16	16
BDSQR BDSQRP	(S) = 0	6.2	2.7	2.7	2.7
	(S) = 99999999	38	16	16	16
BSIN BSINP	-	12	5.1	5.1	5.1
BCOS BCOSP	-	12	5.2	5.2	5.2
BTAN BTANP	-	12	5.2	5.2	5.2
BASIN BASINP	-	20	8.7	8.7	8.7
BACOS BACOSP	-	21	9.0	9.0	9.0
BATAN BATANP	-	22	9.6	9.6	9.6
LIMIT LIMITP	-	10	4.3	4.3	4.3
DLIMIT DLIMITP	-	11	4.7	4.7	4.7
BAND BANDP	-	9.8	4.2	4.2	4.2
DBAND DBANDP	-	11	4.9	4.9	4.9
ZONE ZONEP	-	9.1	3.9	3.9	3.9
DZONE DZONEP	-	11	4.6	4.6	4.6
RSET RSETP	-	6.8	2.9	2.9	2.9
QDRSET QDRSETP	-	205	88	88	88
QCDSET QCDSETP	-	147	63	63	63
DATERD DATERDP	-	13	5.5	5.5	5.5
DATEWR DATEWRP	-	15	6.4	6.4	6.4

指令	条件（软元件）	处理时间（μs）			
		Qn	QnH	QnPH	QnPRH
DATE+ DATE+ P	无进位	13	5.4	5.4	5.4
	有进位	13	5.4	5.4	5.4
DATE- DATE- P	无进位	12	5.2	5.2	5.2
	有进位	12	5.2	5.2	5.2
SECOND SECONDP	-	10	4.5	4.5	4.5
HOUR HOURP	-	12	5.2	5.2	5.2
MSG	1 字符	3.0	1.3	1.3	1.3
	32 字符	3.0	1.3	1.3	1.3
PKEY	初次	20	8.6	8.6	8.6
	无受理	19	8.2	8.2	8.2
PSTOP PSTOPP	-	79	34	34	34
POFF POFFP	-	79	34	34	34
PSCAN PSCNAP	-	75	32	32	32
PLOW PLOWP	-	80	34	34	-
WDT WDTP	-	5.9	2.6	2.6	2.6
DUTY	-	9.3	4.0	4.0	4.0
ZRRDB ZRRDBP	-	7.9	3.4	3.4	3.4
ZRWRB ZRWRBP	-	9.4	4.0	4.0	4.0
ADRSET ADRSETP	-	4.9	2.1	2.1	2.1
KEY	-	17	7.3	7.3	-
ZPUSH ZPUSHP	-	11	4.7	4.7	4.7
ZPOP ZPOPP	-	5.1	2.2	2.2	2.2
EROMWR EROMWRP	-	-	-	-	-

指令	条件（软元件）	处理时间（μs）			
		Qn	QnH	QnPH	QnPRH
ZCOM	－	691	289	289	289
READ	－	－	－	－	－
SREAD	－	－	－	－	－
WRITE	－	－	－	－	－
SWRITE	－	－	－	－	－
SEND	－	－	－	－	－
RECV	－	－	－	－	－
REQ	－	－	－	－	－
ZNFR	－	－	－	－	－
ZNTO	－	－	－	－	－
ZNRD	MELSECNET/10	－	－	－	－
	MELSECNET(II)	－	－	－	－
ZNWR	MELSECNET/10	－	－	－	－
	MELSECNET(II)	－	－	－	－
RFRP	－	－	－	－	－
RTOP	－	－	－	－	－

(4) QCPU 用指令处理时间（QCPU 专用）

(a) 功能版本 A 中可使用的指令

指令	条件（软元件）		处理时间（μs）			
			Qn	QnH	QnPH	QnPRH
UNIRD	－		79	34	34	34
TRACE	开始		176	76	76	76
	STRA 执行结束		6.3	2.7	2.7	2.7
TRACER	－		19	8.2	8.2	8.2
SP.FWRITE	－		84	36	36	36
SP.FREAD	－		82	35	35	35
PLOADP	－		58	25	25	－
PUNLOADP	－		272	117	117	－
PSWAPP	－		308	133	133	－
RBMOV	SRAM 使用文件寄存器→文件寄存器的传送时间	1 点	69	29	29	29
		1000 点	580	308	308	308

(b) 功能版本 B 中可使用的指令

指令	条件・处理点数			处理时间（μs）			
				Qn	QnH	QnPH	QnPRH
COM *1	有 CPU 共享存储器的自动刷新	刷新范围：2k 字（各 CPU 0.5k 字的均等分配）		720	660	660	-
		刷新范围：4k 字（各 CPU 1k 字的均等分配）		860	730	730	-
	无 CPU 共享存储器的自动刷新	-		43	20	20	20
FROM *1	其它机号 CPU 共享存储器读取	n3 = 1		59	29	29	-
		n3 = 1000		530	500	500	-
	智能功能模块的缓冲存储器读取 *2	n3 = 1	主基板	51	24	24	-
			扩展基板	54	27	27	-
		n3 = 1000	主基板	540	480	480	-
			扩展基板	1100	1050	1050	-
S.TO	至本机 CPU 共享存储器的写入	n2 = 1		74	33	33	-
		n2 = 256		126	54	54	-
S (P). DATERD *3	扩展时钟数据的读取	-		25	11	11	11
S (P). DATE+ *3	扩展时钟数据的加法运算	-		38	17	17	17
S (P). DATE- *3	扩展时钟数据的减法运算	-		38	17	17	17

*1: 在多 CPU 系统中，与其它机号 CPU 的处理重叠时，将最多延迟下述时间。

仅主基板的系统时

（指令的延迟时间）= 0.54 ×（处理点数）×（其它机号 CPU 个数）(μs)

包括扩展基板时

（指令的延迟时间）= 1.30 ×（处理点数）×（其它机号 CPU 个数）(μs)

*2: 在多 CPU 系统中，本机 CPU 管理的智能功能模块与及其它机号管理的智能功能模块的指令处理时间相同。

*3: 以序列号的前 5 位数为 “07032” 以后的模块为对象。

(5) 冗余系统用指令（冗余 CPU 专用）

指令	条件（软元件）	处理时间（μs）			
		Qn	QnH	QnPH	QnPRH
SP.CONTSW	-	-	-	-	9.6

附录 1.4　通用型 QCPU 的运算处理时间

各指令的运算处理时间如本页以后的表中所示。

由于运算处理时间根据源、目标的内容而有所不同，因此表中的值只应作为处理时间的大致参考。

附录 1.4.1　子集指令处理时间

子集处理指令的处理时间如下所示。

要 点

1. 指令中使用的软元件满足子集处理的软元件条件之一时的子集指令的指令处理时间一览表如下述 (1) 所示。
   (关于子集处理的软元件条件，请参阅 3.5.1 项。)
2. 使用文件寄存器、模块访问软元件 (U3En\G10000 ～) 时，请参阅 (2) 的加法运算时间，对各指令的处理时间进行加法运算。
3. 在 OUT/SET/RST 指令中使用 F、T(ST)、C 软元件时，请参阅 (3) 中列出的加法运算时间，对各指令的处理时间进行加法运算。
4. 在通用型 QCPU 中，由于高速缓冲存储器功能的影响，各指令的处理时间不恒定，因此对最小值和最大值进行了记载。

(1) 子集指令的指令处理时间一览表

(a) 使用 Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU 时

分类	指令	条件 (软元件)		处理时间 (μ s)							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
顺控程序指令	LD LDI AND ANI OR ORI LDP LDF ANDP ANDF ORP ORF	执行时		0.120		0.080		0.060		0.040	
	LDPI LDFI	执行时		0.360		0.240		0.180		0.120	
	ANDPI ANDFI ORPI ORFI	执行时		0.480		0.320		0.240		0.160	
	OUT	无变化时		0.120		0.080		0.060		0.040	
		变化时									
	SET RST	未执行时		0.120		0.080		0.060		0.040	
		执行时	无变化时								
			变化时								

分类	指令	条件（软元件）		处理时间（μs）							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
基本指令	LD =	导通时		0.360		0.240		0.180		0.120	
		非导通时									
	AND =	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时								
			非导通时								
	OR =	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时								
			非导通时								
	LD< >	导通时		0.360		0.240		0.180		0.120	
		非导通时									
	AND< >	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时								
			非导通时								
	OR< >	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时								
			非导通时								
	LD >	导通时		0.360		0.240		0.180		0.120	
		非导通时									
	AND >	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时								
			非导通时								
	OR >	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时								
			非导通时								
	LD< =	导通时		0.360		0.240		0.180		0.120	
		非导通时									
	AND< =	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时								
			非导通时								
	OR< =	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时								
			非导通时								
	LD<	导通时		0.360		0.240		0.180		0.120	
		非导通时									
	AND<	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时								
			非导通时								
	OR<	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时								
			非导通时								
	LD > =	导通时		0.360		0.240		0.180		0.120	
		非导通时									
	AND > =	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时								
			非导通时								
	OR > =	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时								
			非导通时								

分类	指令	条件（软元件）		处理时间（μs）							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
基本指令	LDD =	导通时		0.360		0.240		0.180		0.120	
		非导通时									
	ANDD =	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时								
			非导通时								
	ORD =	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时								
			非导通时								
	LDD< >	导通时		0.360		0.240		0.180		0.120	
		非导通时									
	ANDD< >	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时								
			非导通时								
	ORD< >	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时								
			非导通时								
	LDD >	导通时		0.360		0.240		0.180		0.120	
		非导通时									
	ANDD >	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时								
			非导通时								
	ORD >	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时								
			非导通时								
	LDD< =	导通时		0.360		0.240		0.180		0.120	
		非导通时									
	ANDD< =	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时								
			非导通时								
	ORD< =	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时								
			非导通时								
	LDD<	导通时		0.360		0.240		0.180		0.120	
		非导通时									
	ANDD<	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时								
			非导通时								
	ORD<	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时								
			非导通时								
	LDD > =	导通时		0.360		0.240		0.180		0.120	
		非导通时									
	ANDD > =	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时								
			非导通时								
	ORD > =	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时								
			非导通时								

分类	指令	条件（软元件）		处理时间（μs）							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
基本指令	+ (S) (D)	执行时		0.360		0.240		0.180		0.120	
	+ (S1) (S2) (D)	执行时		0.480		0.320		0.240		0.160	
	- (S) (D)	执行时		0.360		0.240		0.180		0.120	
	- (S1) (S2) (D)	执行时		0.480		0.320		0.240		0.160	
	D + (S) (D)	执行时		0.360		0.240		0.180		0.120	
	D + (S1) (S2) (D)	执行时		0.480		0.320		0.240		0.160	
	D - (S) (D)	执行时		0.360		0.240		0.180		0.120	
	D - (S1) (S2) (D)	执行时		0.480		0.320		0.240		0.160	
	* (S1) (S2) (D)	执行时		0.420		0.300		0.240		0.180	
	/ (S1) (S2) (D)	执行时		0.520		0.400		0.340		0.280	
	D * (S1) (S2) (D)	执行时		0.500		0.380		0.320		0.260	
	D / (S1) (S2) (D)	执行时		0.640		0.520		0.460		0.400	
	B + (S) (D)	执行时		3.100	12.300	3.100	12.300	3.100	12.300	3.300	8.300
	B + (S1) (S2) (D)	执行时		5.900	13.500	5.900	13.500	5.900	13.500	4.600	6.200
	B - (S) (D)	执行时		3.150	12.300	3.150	12.300	3.150	12.300	3.300	9.000
	B - (S1) (S2) (D)	执行时		5.950	13.600	5.950	13.600	5.950	13.600	4.600	8.200
	B * (S1) (S2) (D)	执行时		3.700	12.100	3.700	12.100	3.700	12.100	4.000	8.200
	B / (S1) (S2) (D)	执行时		4.000	14.000	4.000	14.000	4.000	14.000	4.200	12.400
	E+ (S) (D)	单精度	(S) = 0, (D) = 0	0.420		0.300		0.240		0.180	
			(S) = 2^{127}, (D) = 2^{127}	0.420		0.300		0.240		0.180	
	E+ (S1) (S2) (D)	单精度	(S1) = 0, (S2) = 0	0.540		0.380		0.300		0.220	
			(S1) = 2^{127}, (S2) = 2^{127}	0.540		0.380		0.300		0.220	
	E- (S) (D)	单精度	(S) = 0, (D) = 0	0.420		0.300		0.240		0.180	
			(S) = 2^{127}, (D) = 2^{127}	0.420		0.300		0.240		0.180	
	E- (S1) (S2) (D)	单精度	(S1) = 0, (S2) = 0	0.540		0.380		0.300		0.220	
			(S1) = 2^{127}, (S2) = 2^{127}	0.540		0.380		0.300		0.220	
	E* (S1) (S2) (D)	单精度	(S1) = 0, (S2) = 0	0.420		0.300		0.240		0.180	
			(S1) = 2^{127}, (S2) = 2^{127}	0.420		0.300		0.240		0.180	
	E/ (S1) (S2) (D)	单精度	(S1) = 2^{127}, (S2) = 2^{127}	4.900	18.900	4.900	18.900	4.900	18.900	5.100	14.100

分类	指令	条件（软元件）		处理时间（μs）							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
基本指令	INC	执行时		0.240		0.160		0.120		0.080	
	DINC	执行时		0.240		0.160		0.120		0.080	
	DEC	执行时		0.240		0.160		0.120		0.080	
	DDEC	执行时		0.240		0.160		0.120		0.080	
	BCD	执行时		0.320		0.240		0.200		0.160	
	DBCD	执行时		0.400		0.320		0.280		0.240	
	BIN	执行时		0.260		0.180		0.140		0.100	
	DBIN	执行时		0.260		0.180		0.140		0.100	
	FLT	单精度	Ⓢ = 0	0.300		0.220		0.180		0.140	
			Ⓢ = 7FFFH	0.300		0.220		0.180		0.140	
	DFLT	单精度	Ⓢ = 0	0.300		0.220		0.180		0.140	
			Ⓢ = 7FFFFFFFH	0.300		0.220		0.180		0.140	
	INT	单精度	Ⓢ = 0	0.300		0.220		0.180		0.140	
			Ⓢ = 32766.5	0.300		0.220		0.180		0.140	
	DINT	单精度	Ⓢ = 0	0.300		0.220		0.180		0.140	
			Ⓢ = 1234567890.3	0.300		0.220		0.180		0.140	
	MOV	－		0.240		0.160		0.120		0.080	
	DMOV	－		0.240		0.160		0.120		0.080	
	EMOV	－		0.240		0.160		0.120		0.080	
	CML	－		0.240		0.160		0.120		0.080	
	DCML	－		0.240		0.160		0.120		0.080	
	BMOV	SM237 = ON	n = 1	4.200	4.600	4.200	4.600	4.200	4.600	4.100	4.500
			n = 96	4.850	5.150	4.850	5.150	4.850	5.150	4.700	5.100
		SM237 = OFF	n = 1	6.800	11.300	6.800	11.300	6.800	11.300	6.300	8.900
			n = 96	7.450	11.900	7.450	11.900	7.450	11.900	5.900	9.500
	FMOV	SM237 = ON	n = 1	4.100	4.600	4.100	4.600	4.100	4.600	4.100	4.600
			n = 96	4.800	5.200	4.800	5.200	4.800	5.200	4.800	5.200
		SM237 = OFF	n = 1	4.600	8.250	4.600	8.250	4.600	8.250	4.600	7.900
			n = 96	6.150	10.600	6.150	10.600	6.150	10.600	5.300	8.500
	XCH	－		2.250	8.100	2.250	8.100	2.250	8.100	2.500	6.000
	DXCH	－		2.400	8.200	2.400	8.200	2.400	8.200	2.800	7.900
	DFMOV	SM237 = ON	n = 1	2.700	2.800	2.700	2.800	2.700	2.800	2.350	2.450
			n = 96	6.500	6.800	6.500	6.800	6.500	6.800	5.950	6.000
		SM237 = OFF	n = 1	4.000	8.150	4.000	8.150	4.000	8.150	3.000	6.950
			n = 96	8.000	12.200	8.000	12.200	8.000	12.200	6.600	10.600
	CJ	－		3.500	10.100	3.500	10.100	3.500	10.100	1.900	10.100
	SCJ	－		3.500	10.100	3.500	10.100	3.500	10.100	1.900	10.100
	JMP	－		3.500	10.100	3.500	10.100	3.500	10.100	1.900	10.100

分类	指令	条件（软元件）	处理时间（μs）							
			Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
			最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
应用指令	WAND (S) (D)	执行时	0.360		0.240		0.180		0.120	
	WAND (S1) (S2) (D)	执行时	0.480		0.320		0.240		0.160	
	DAND (S) (D)	执行时	0.360		0.240		0.180		0.120	
	DAND (S1) (S2) (D)	执行时	0.480		0.320		0.240		0.160	
	WOR (S) (D)	执行时	0.360		0.240		0.180		0.120	
	WOR (S1) (S2) (D)	执行时	0.480		0.320		0.240		0.160	
	DOR (S) (D)	执行时	0.360		0.240		0.180		0.120	
	DOR (S1) (S2) (D)	执行时	0.480		0.320		0.240		0.160	
	WXOR (S) (D)	执行时	0.360		0.240		0.180		0.120	
	WXOR (S1) (S2) (D)	执行时	0.480		0.320		0.240		0.160	
	DXOR (S) (D)	执行时	0.360		0.240		0.180		0.120	
	DXOR (S1) (S2) (D)	执行时	0.480		0.320		0.240		0.160	
	WXNR (S) (D)	执行时	0.360		0.240		0.180		0.120	
	WXNR (S1) (S2) (D)	执行时	0.480		0.320		0.240		0.160	
	DXNR (S) (D)	执行时	0.360		0.240		0.180		0.120	
	DXNR (S1) (S2) (D)	执行时	0.480		0.320		0.240		0.160	

分类	指令	条件（软元件）	处理时间（μs）							
			Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
			最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
应用指令	ROR (D) n	n = 1	2.250	10.800	2.250	10.800	2.250	10.800	2.300	7.800
		n = 15	2.250	10.800	2.350	10.800	2.350	10.800	2.400	7.800
	RCR (D) n	n = 1	2.250	10.800	2.250	10.800	2.250	10.800	2.300	3.900
		n = 15	2.250	10.800	2.250	10.800	2.250	10.800	2.400	4.100
	ROL (D) n	n = 1	2.250	10.800	2.350	10.800	2.350	10.800	2.500	4.600
		n = 15	2.250	10.800	2.350	10.800	2.350	10.800	2.400	4.600
	RCL (D) n	n = 1	2.250	11.500	2.300	11.500	2.300	11.500	2.400	7.500
		n = 15	2.250	11.500	2.300	11.500	2.300	11.500	2.500	7.500
	DROR (D) n	n = 1	2.350	11.500	2.350	11.500	2.350	11.500	2.400	10.300
		n = 31	2.350	11.500	2.350	11.500	2.350	11.500	2.500	10.300
	DRCR (D) n	n = 1	2.350	13.300	2.350	13.300	2.350	13.300	2.500	12.700
		n = 31	2.350	14.900	2.350	14.900	2.350	14.900	2.500	12.700
	DROL (D) n	n = 1	2.350	10.800	2.350	10.800	2.350	10.800	2.500	11.800
		n = 31	2.350	10.800	2.350	10.800	2.350	10.800	2.500	11.800
	DRCL (D) n	n = 1	2.350	13.300	2.350	13.300	2.350	13.300	2.500	5.100
		n = 31	2.350	13.300	2.350	13.300	2.350	13.300	2.500	5.100
	SFR (D) n	n = 1	2.350	9.900	2.350	9.900	2.350	9.900	2.400	6.100
		n = 15	2.350	9.900	2.350	9.900	2.350	9.900	2.300	5.700
	SFL (D) n	n = 1	2.350	9.850	2.350	9.850	2.350	9.850	2.400	4.300
		n = 15	2.350	9.850	2.350	9.850	2.350	9.850	2.400	4.300
	DSFR (D) n	n = 1	3.250	15.500	3.250	15.500	3.250	15.500	3.300	12.000
		n = 96	32.600	45.000	32.600	45.000	32.600	45.000	32.600	42.200
	DSFL (D) n	n = 1	3.200	15.500	3.200	15.500	3.200	15.500	3.300	8.200
		n = 96	32.600	45.100	32.600	45.100	32.600	45.100	32.600	37.700
	SUM	(S) = 0	3.100	8.950	3.100	8.950	3.100	8.950	3.400	6.700
		(S) = $FFFF_H$	3.000	8.850	3.000	8.850	3.000	8.850	3.500	6.700
	SEG	执行时	2.100	7.700	2.100	7.700	2.100	7.700	2.100	5.900
	FOR	-	1.500	7.500	1.500	7.500	1.500	7.500	1.200	6.300
	CALL Pn	文件内指针	4.800	5.400	4.800	5.400	4.800	5.400	2.700	4.800
		公共指针	7.100	30.500	7.100	30.500	7.100	30.500	4.400	5.700
	CALL Pn (S1) ～ (S5)	-	50.200	62.000	50.200	62.000	50.200	62.000	28.700	42.600

备注

对于表中未记述上升沿执行指令（□ P）的指令，其处理时间与 ON 时执行的指令的处理时间相同。

例 MOVP 指令、WANDP 指令等。

(b) 使用 Q03UD(E)CPU、Q04UD(E)HCPU、Q06UD(E)HCPU、Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU 时

分类	指令	条件（软元件）		处理时间（μs）					
				Q03UD(E)CPU		Q04UD(E)HCPU、Q06UD(E)HCPU		Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU	
				最小值	最大值	最小值	最大值	最小值	最大值
顺控程序指令	LD LDI AND ANI OR ORI LDP LDF ANDP ANDF ORP ORF	执行时		0.020		0.0095		0.0095	
	LDPI LDFI	执行时		0.060		0.0285		0.0285	
	ANDPI ANDFI ORPI ORFI	执行时		0.080		0.038		0.038	
	OUT	无变化时		0.0020		0.0095		0.0095	
		变化时							
	SET RST	未执行时		0.0020		0.0095		0.0095	
		执行时	无变化时						
			变化时						

分类	指令	条件（软元件）		处理时间（μs）					
				Q03UD(E)CPU		Q04UD(E)HCPU、Q06UD(E)HCPU		Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU	
				最小值	最大值	最小值	最大值	最小值	最大值
基本指令	LD =	导通时		0.060		0.0285		0.0285	
		非导通时							
	AND =	未执行时		0.060		0.0285		0.0285	
		执行时	导通时						
			非导通时						
	OR =	未执行时		0.060		0.0285		0.0285	
		执行时	导通时						
			非导通时						
	LD< >	导通时		0.060		0.0285		0.0285	
		非导通时							
	AND< >	未执行时		0.060		0.0285		0.0285	
		执行时	导通时						
			非导通时						
	OR< >	未执行时		0.060		0.0285		0.0285	
		执行时	导通时						
			非导通时						
	LD >	导通时		0.060		0.0285		0.0285	
		非导通时							
	AND >	未执行时		0.060		0.0285		0.0285	
		执行时	导通时						
			非导通时						
	OR >	未执行时		0.060		0.0285		0.0285	
		执行时	导通时						
			非导通时						
	LD< =	导通时		0.060		0.0285		0.0285	
		非导通时							
	AND< =	未执行时		0.060		0.0285		0.0285	
		执行时	导通时						
			非导通时						
	OR< =	未执行时		0.060		0.0285		0.0285	
		执行时	导通时						
			非导通时						
	LD<	导通时		0.060		0.0285		0.0285	
		非导通时							
	AND<	未执行时		0.060		0.0285		0.0285	
		执行时	导通时						
			非导通时						
	OR<	未执行时		0.060		0.0285		0.0285	
		执行时	导通时						
			非导通时						
	LD > =	导通时		0.060		0.0285		0.0285	
		非导通时							
	AND > =	未执行时		0.060		0.0285		0.0285	
		执行时	导通时						
			非导通时						
	OR > =	未执行时		0.060		0.0285		0.0285	
		执行时	导通时						
			非导通时						

分类	指令	条件（软元件）		处理时间（μs）					
				Q03UD(E)CPU		Q04UD(E)HCPU、Q06UD(E)HCPU		Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU	
				最小值	最大值	最小值	最大值	最小值	最大值
基本指令	LDD =	导通时		0.060		0.0285		0.0285	
		非导通时							
	ANDD =	未执行时		0.060		0.0285		0.0285	
		执行时	导通时						
			非导通时						
	ORD =	未执行时		0.060		0.0285		0.0285	
		执行时	导通时						
			非导通时						
	LDD< >	导通时		0.060		0.0285		0.0285	
		非导通时							
	ANDD< >	未执行时		0.060		0.0285		0.0285	
		执行时	导通时						
			非导通时						
	ORD< >	未执行时		0.060		0.0285		0.0285	
		执行时	导通时						
			非导通时						
	LDD >	导通时		0.060		0.0285		0.0285	
		非导通时							
	ANDD >	未执行时		0.060		0.0285		0.0285	
		执行时	导通时						
			非导通时						
	ORD >	未执行时		0.060		0.0285		0.0285	
		执行时	导通时						
			非导通时						
	LDD< =	导通时		0.060		0.0285		0.0285	
		非导通时							
	ANDD< =	未执行时		0.060		0.0285		0.0285	
		执行时	导通时						
			非导通时						
	ORD< =	未执行时		0.060		0.0285		0.0285	
		执行时	导通时						
			非导通时						
	LDD<	导通时		0.060		0.0285		0.0285	
		非导通时							
	ANDD<	未执行时		0.060		0.0285		0.0285	
		执行时	导通时						
			非导通时						
	ORD<	未执行时		0.060		0.0285		0.0285	
		执行时	导通时						
			非导通时						
	LDD > =	导通时		0.060		0.0285		0.0285	
		非导通时							
	ANDD > =	未执行时		0.060		0.0285		0.0285	
		执行时	导通时						
			非导通时						
	ORD > =	未执行时		0.060		0.0285		0.0285	
		执行时	导通时						
			非导通时						

分类	指令	条件（软元件）		处理时间（μs）					
				Q03UD(E)CPU		Q04UD(E)HCPU、Q06UD(E)HCPU		Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UDEHCPU	
				最小值	最大值	最小值	最大值	最小值	最大值
基本指令	+ (S) (D)	执行时		0.060		0.0285		0.0285	
	+ (S1) (S2) (D)	执行时		0.080		0.038		0.038	
	- (S) (D)	执行时		0.060		0.0285		0.0285	
	- (S1) (S2) (D)	执行时		0.080		0.038		0.038	
	D + (S) (D)	执行时		0.060		0.0285		0.0285	
	D + (S1) (S2) (D)	执行时		0.080		0.038		0.038	
	D - (S) (D)	执行时		0.060		0.0285		0.0285	
	D - (S1) (S2) (D)	执行时		0.080		0.038		0.038	
	* (S1) (S2) (D)	执行时		0.120		0.057		0.057	
	/ (S1) (S2) (D)	执行时		0.220		0.110		0.110	
	D * (S1) (S2) (D)	执行时		0.200		0.095		0.095	
	D / (S1) (S2) (D)	执行时		0.340		0.170		0.170	
	B + (S) (D)	执行时		3.300	5.500	3.000	4.100	3.000	4.100
	B + (S1) (S2) (D)	执行时		4.600	6.200	4.200	5.900	4.200	5.900
	B - (S) (D)	执行时		3.300	4.400	2.900	3.800	2.900	3.800
	B - (S1) (S2) (D)	执行时		4.600	6.300	4.200	4.600	4.200	4.600
	B * (S1) (S2) (D)	执行时		4.000	4.800	3.400	4.800	3.400	4.800
	B / (S1) (S2) (D)	执行时		4.200	5.700	3.700	5.200	3.700	5.200
	E+ (S) (D)	单精度	(S) = 0, (D) = 0	0.120		0.057		0.057	
			(S) = 2^{127}, (D) = 2^{127}	0.120		0.057		0.057	
	E+ (S1) (S2) (D)	单精度	(S1) = 0, (S2) = 0	0.140		0.0665		0.0665	
			(S1) = 2^{127}, (S2) = 2^{127}	0.140		0.0665		0.0665	
	E- (S) (D)	单精度	(S) = 0, (D) = 0	0.120		0.057		0.057	
			(S) = 2^{127}, (D) = 2^{127}	0.120		0.057		0.057	
	E- (S1) (S2) (D)	单精度	(S1) = 0, (S2) = 0	0.140		0.0665		0.0665	
			(S1) = 2^{127}, (S2) = 2^{127}	0.140		0.0665		0.0665	
	E* (S1) (S2) (D)	单精度	(S1) = 0, (S2) = 0	0.120		0.057		0.057	
			(S1) = 2^{127}, (S2) = 2^{127}	0.120		0.057		0.057	
	E/ (S1) (S2) (D)	单精度	(S1) = 2^{127}, (S2) = 2^{127}	4.500	5.600	3.900	4.900	0.285	

分类	指令	条件（软元件）		处理时间（μs）					
				Q03UD(E)CPU		Q04UD(E)HCPU、Q06UD(E)HCPU		Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU	
				最小值	最大值	最小值	最大值	最小值	最大值
基本指令	INC	执行时		0.040		0.019		0.019	
	DINC	执行时		0.040		0.019		0.019	
	DEC	执行时		0.040		0.019		0.019	
	DDEC	执行时		0.040		0.019		0.019	
	BCD	执行时		0.120		0.057		0.057	
	DBCD	执行时		0.200		0.095		0.095	
	BIN	执行时		0.060		0.0285		0.0285	
	DBIN	执行时		0.060		0.0285		0.0285	
	FLT	单精度	Ⓢ = 0	0.100		0.0475		0.0475	
			Ⓢ = 7FFFH	0.100		0.0475		0.0475	
	DFLT	单精度	Ⓢ = 0	0.100		0.0475		0.0475	
			Ⓢ = 7FFFFFFFH	0.100		0.0475		0.0475	
	INT	单精度	Ⓢ = 0	0.100		0.0475		0.0475	
			Ⓢ = 32766.5	0.100		0.0475		0.0475	
	DINT	单精度	Ⓢ = 0	0.100		0.0475		0.0475	
			Ⓢ = 1234567890.3	0.100		0.0475		0.0475	
	MOV	－		0.040		0.019		0.019	
	DMOV	－		0.040		0.019		0.019	
	EMOV	－		0.040		0.019		0.019	
	CML	－		0.040		0.019		0.019	
	DCML	－		0.040		0.019		0.019	
	BMOV	n = 1		6.300	8.200	5.400	7.000	5.400	7.000
			SM237 = OFF*1	8.200	10.600	3.900	5.100	3.900	5.100
			SM237 = ON*1	6.000	7.800	2.900	3.700	2.900	3.700
		n = 96		7.100	8.800	5.900	7.600	5.900	7.600
			SM237 = OFF*1	9.300	11.900	4.400	5.700	4.400	5.700
			SM237 = ON*1	7.100	9.100	3.400	4.300	3.400	4.300
	FMOV	n = 1		5.300	5.900	4.200	4.800	4.200	4.800
			SM237 = OFF*1	7.000	8.000	3.400	3.800	3.400	3.800
			SM237 = ON*1	5.900	6.800	2.800	3.200	2.800	3.200
		n = 96		5.300	7.600	4.400	6.800	4.400	6.800
			SM237 = OFF*1	7.400	12.200	3.600	5.800	3.600	5.800
			SM237 = ON*1	6.300	11.000	3.000	5.200	3.000	5.200
	XCH	－		2.500	2.900	1.800	2.300	1.800	2.300
	DXCH	－		2.800	3.700	2.100	2.900	2.100	2.900
	DFMOV*2	n = 1	SM237 = OFF	2.600	3.750	2.250	3.150	2.250	3.150
			SM237 = ON	2.050	2.250	1.750	1.750	1.750	1.750
		n = 96	SM237 = OFF	5.850	7.350	4.200	5.500	4.200	5.500
			SM237 = ON	5.300	6.000	3.650	4.150	3.650	4.150
	CJ	－		1.800	2.800	1.400	2.400	1.400	2.400
	SCJ	－		1.800	2.800	1.400	2.400	1.400	2.400
	JMP	－		1.800	2.800	1.100	2.400	1.100	2.400

*1: 在序列号的前 5 位数为 “10012” 以后的 Q03UDCPU、Q04UDHCPU、Q06UDHCPU 中可以使用。
*2: 在序列号的前 5 位数为 “10102” 的 Q03UD(E)CPU、Q04UD(E)HCPU、Q06UD(E)HCPU、Q13UD(E)HCPU、Q26UD(E)HCPU 中可以使用。

分类	指令	条件（软元件）	处理时间（μs）					
			Q03UD(E)CPU		Q04UD(E)HCPU、Q06UD(E)HCPU		Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU	
			最小值	最大值	最小值	最大值	最小值	最大值
应用指令	WAND (S) (D)	执行时	0.060		0.0285		0.0285	
	WAND (S1) (S2) (D)	执行时	0.080		0.038		0.038	
	DAND (S) (D)	执行时	0.060		0.0285		0.0285	
	DAND (S1) (S2) (D)	执行时	0.080		0.038		0.038	
	WOR (S) (D)	执行时	0.060		0.0285		0.0285	
	WOR (S1) (S2) (D)	执行时	0.080		0.038		0.038	
	DOR (S) (D)	执行时	0.060		0.0285		0.0285	
	DOR (S1) (S2) (D)	执行时	0.080		0.038		0.038	
	WXOR (S) (D)	执行时	0.060		0.0285		0.0285	
	WXOR (S1) (S2) (D)	执行时	0.080		0.038		0.038	
	DXOR (S) (D)	执行时	0.060		0.0285		0.0285	
	DXOR (S1) (S2) (D)	执行时	0.080		0.038		0.038	
	WXNR (S) (D)	执行时	0.060		0.0285		0.0285	
	WXNR (S1) (S2) (D)	执行时	0.080		0.038		0.038	
	DXNR (S) (D)	执行时	0.060		0.0285		0.0285	
	DXNR (S1) (S2) (D)	执行时	0.080		0.038		0.038	

分类	指令	条件（软元件）	处理时间（μs）					
			Q03UD(E)CPU		Q04UD(E)HCPU、Q06UD(E)HCPU		Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU	
			最小值	最大值	最小值	最大值	最小值	最大值
应用指令	ROR (D) n	n = 1	2.300	3.100	1.700	2.500	1.700	2.500
		n = 15	2.400	3.100	1.800	2.500	1.800	2.500
	RCR (D) n	n = 1	2.300	3.900	1.700	3.200	1.700	3.200
		n = 15	2.400	4.100	1.700	3.200	1.700	3.200
	ROL (D) n	n = 1	2.400	3.300	1.800	3.200	1.800	3.200
		n = 15	2.400	3.300	1.800	3.200	1.800	3.200
	RCL (D) n	n = 1	2.400	2.700	1.800	2.100	1.800	2.100
		n = 15	2.400	2.800	1.800	2.200	1.800	2.200
	DROR (D) n	n = 1	2.400	3.400	1.900	2.700	1.900	2.700
		n = 31	2.500	3.400	1.900	2.700	1.900	2.700
	DRCR (D) n	n = 1	2.500	4.800	1.900	4.200	1.900	4.200
		n = 31	2.500	4.900	1.900	4.200	1.900	4.200
	DROL (D) n	n = 1	2.500	3.900	1.800	3.200	1.800	3.200
		n = 31	2.500	3.900	1.800	3.300	1.800	3.300
	DRCL (D) n	n = 1	2.500	4.800	1.900	3.800	1.900	3.800
		n = 31	2.500	4.600	1.900	3.800	1.900	3.800
	SFR (D) n	n = 1	2.400	3.900	1.700	2.600	1.700	2.600
		n = 15	2.300	3.900	1.800	2.600	1.800	2.600
	SFL (D) n	n = 1	2.400	4.300	1.800	2.700	1.800	2.700
		n = 15	2.400	4.300	1.800	2.700	1.800	2.700
	DSFR (D) n	n = 1	2.700	4.800	2.200	4.300	2.200	4.300
		n = 96	32.600	35.900	23.900	26.100	23.900	26.100
	DSFL (D) n	n = 1	2.700	4.600	2.100	4.000	2.100	4.000
		n = 96	32.600	35.300	23.700	25.800	23.700	25.800
	SUM	(S) = 0	3.400	4.300	2.900	3.600	2.900	3.600
		(S) = $FFFF_H$	3.500	4.200	2.900	3.600	2.900	3.600
	SEG	执行时	2.100	2.800	1.500	2.100	1.500	2.100
	FOR	-	1.200	2.400	0.870	2.100	0.870	2.100
	CALL Pn	文件内指针	2.600	4.000	2.300	3.600	2.300	3.600
		公共指针	4.000	5.300	3.200	4.900	3.200	4.900
	CALL Pn (S1) ～ (S5)	-	28.700	33.400	26.100	29.300	26.100	29.300

备注

对于表中未记述上升沿执行指令（□P）的指令，其处理时间与ON时执行的指令的处理时间相同。

例 MOVP指令、WANDP指令等。

(2) 使用了文件寄存器、模块访问软元件(U3En\G10000 ～)时的加法运算时间一览表

(a) 使用Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU时

软元件名		数据	软元件指定位置	加法运算时间(μs)			
				Q00UJCPU	Q00UCPU	Q01UCPU	Q02UCPU
文件寄存器(R)	文件寄存器(R)	位	源	0.100	0.100	0.100	0.100
			目标	0.220	0.220	0.220	0.220
		字	源	0.100	0.100	0.100	0.100
			目标	0.100	0.100	0.100	0.100
		双字	源	0.200	0.200	0.200	0.200
			目标	0.200	0.200	0.200	0.200
	使用SRAM卡(Q2MEM-1MBS、Q2MEM-2MBS)时	位	源	-	-	-	0.220
			目标	-	-	-	0.420
		字	源	-	-	-	0.220
			目标	-	-	-	0.180
		双字	源	-	-	-	0.440
			目标	-	-	-	0.380
	使用SRAM卡(Q3MEM-4MBS、Q3MEM-8MBS)时	位	源	-	-	-	0.160
			目标	-	-	-	0.320
		字	源	-	-	-	0.160
			目标	-	-	-	0.140
		双字	源	-	-	-	0.320
			目标	-	-	-	0.300
文件寄存器(ZR)、扩展数据寄存器(D)、扩展链接寄存器(W)	使用标准RAM时	位	源	0.220	0.180	0.160	0.140
			目标	0.360	0.320	0.300	0.280
		字	源	0.220	0.180	0.160	0.140
			目标	0.220	0.180	0.160	0.140
		双字	源	0.320	0.280	0.260	0.240
			目标	0.320	0.280	0.260	0.240
	使用SRAM卡(Q2MEM-1MBS、Q2MEM-2MBS)时	位	源	-	-	-	0.260
			目标	-	-	-	0.480
		字	源	-	-	-	0.260
			目标	-	-	-	0.220
		双字	源	-	-	-	0.480
			目标	-	-	-	0.420
	使用SRAM卡(Q3MEM-4MBS、Q3MEM-8MBS)时	位	源	-	-	-	0.200
			目标	-	-	-	0.380
		字	源	-	-	-	0.200
			目标	-	-	-	0.180
		双字	源	-	-	-	0.360
			目标	-	-	-	0.340
模块访问软元件(多CPU高速通信区)(U3En\G10000 ～)		位	源	-	-	-	-
			目标	-	-	-	-
		字	源	-	-	-	-
			目标	-	-	-	-
		双字	源	-	-	-	-
			目标	-	-	-	-

使用了文件寄存器、模块访问软元件（U3En\G10000 ～）时的加法运算时间一览表

(b) 使用 Q03UD(E)CPU、Q04UD(E)HCPU、Q06UD(E)HCPU、Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU 时

软元件名		数据	软元件指定位置	加法运算时间（μs）		
				Q03UD(E)CPU	Q04UD(E)HCPU、Q06UD(E)HCPU	Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU
文件寄存器(R)	文件寄存器(R)	位	源	0.100	0.048	0.048
			目标	0.100	0.038	0.038
		字	源	0.100	0.048	0.048
			目标	0.100	0.038	0.038
		双字	源	0.200	0.095	0.095
			目标	0.200	0.086	0.086
	使用 SRAM 卡(Q2MEM-1MBS、Q2MEM-2MBS)时	位	源	0.220	0.200	0.200
			目标	0.180	0.162	0.162
		字	源	0.220	0.200	0.200
			目标	0.180	0.162	0.162
		双字	源	0.440	0.399	0.399
			目标	0.380	0.361	0.361
	使用 SRAM 卡(Q3MEM-4MBS、Q3MEM-8MBS)时	位	源	0.160	0.152	0.152
			目标	0.140	0.133	0.133
		字	源	0.160	0.152	0.152
			目标	0.140	0.133	0.133
		双字	源	0.320	0.304	0.304
			目标	0.300	0.295	0.295
文件寄存器(ZR)、扩展数据寄存器(D)、扩展链接寄存器(W)	使用标准 RAM 时	位	源	0.120	0.057	0.057
			目标	0.120	0.048	0.048
		字	源	0.120	0.057	0.057
			目标	0.120	0.048	0.048
		双字	源	0.220	0.105	0.105
			目标	0.220	0.095	0.095
	使用 SRAM 卡(Q2MEM-1MBS、Q2MEM-2MBS)时	位	源	0.240	0.209	0.209
			目标	0.200	0.171	0.171
		字	源	0.240	0.209	0.209
			目标	0.200	0.171	0.171
		双字	源	0.460	0.409	0.409
			目标	0.400	0.371	0.371
	使用 SRAM 卡(Q3MEM-4MBS、Q3MEM-8MBS)时	位	源	0.180	0.162	0.162
			目标	0.160	0.143	0.143
		字	源	0.180	0.162	0.162
			目标	0.160	0.143	0.143
		双字	源	0.340	0.314	0.314
			目标	0.320	0.304	0.304
模块访问软元件（多 CPU 高速通信区）(U3En\G10000 ～)		位	源	0.220	0.181	0.181
			目标	0.140	0.105	0.105
		字	源	0.220	0.181	0.181
			目标	0.140	0.105	0.105
		双字	源	0.500	0.437	0.437
			目标	0.340	0.285	0.285

(3) 在 OUT/SET/RST 指令的软元件中，使用了 F、T(ST)、C 软元件时的加法运算时间一览表

(a) 使用 Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU 时

指令名	软元件名	条件		加法运算时间（μs）			
				Q00UJCPU	Q00UCPU	Q01UCPU	Q02UCPU
OUT	F	未执行时		2.900	2.900	2.900	2.100
		执行时	显示时	116.000	116.000	116.000	68.800
			显示结束	116.000	116.000	116.000	61.600
	T(ST)，C	未执行时		0.360	0.240	0.180	0.120
		执行时	时间到后	0.360	0.240	0.180	0.120
			计数时	0.360	0.240	0.180	0.120
SET	F	未执行时		0.120	0.080	0.006	0.004
		执行时	显示时	116.000	116.000	116.000	68.600
			显示结束	116.000	116.000	116.000	65.700
RST	F	未执行时		0.120	0.080	0.006	0.004
		执行时	显示时	55.800	55.800	55.800	26.500
			显示结束	29.200	29.200	29.200	21.600
	T(ST)，C	未执行时		0.360	0.240	0.180	0.120
		执行时		0.360	0.240	0.180	0.120

(b) 使用 Q03UD(E)CPU、Q04UD(E)HCPU、Q06UD(E)HCPU、Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU 时

指令名	软元件名	条件		加法运算时间（μs）		
				Q03UD(E)CPU	Q04UD(E)HCPU、Q06UD(E)HCPU	Q10UD(E)HCPU、Q13UD(E)HCPU，Q20UD(E)HCPU、Q26UD(E)HCPU
OUT	F	未执行时		1.940	1.570	1.570
		执行时	显示时	39.930	38.090	38.090
			显示结束	39.750	37.980	37.980
	T(ST)，C	未执行时		0.060	0.030	0.030
		执行时	时间到后	0.060	0.030	0.030
SET	F	未执行时		0.000	0.000	0.000
		执行时	显示时	42.900	40.600	40.600
			显示结束	39.270	37.900	37.900
RST	F	未执行时		0.000	0.000	0.000
		执行时	显示时	45.260	36.600	36.600
			显示结束	19.020	16.190	16.190
	T(ST)，C	未执行时		0.060	0.030	0.030
		执行时		0.060	0.030	0.030

附录 1.4.2 子集指令以外的指令处理时间

子集处理指令以外的处理时间如下所示。

(1) 子集指令以外的指令处理时间一览表

(a) 使用 Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU 时

分类	指令	条件（软元件）	处理时间（μs）							
			Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
			最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
顺控程序指令	ANB ORB MPS MRD MPP	-	0.120		0.080		0.060		0.040	
	INV	未执行时	0.120		0.080		0.060		0.040	
		执行时								
	MEP MEF	未执行时	0.120		0.080		0.060		0.040	
		执行时								
	EGP EGF	未执行时	0.120		0.080		0.060		0.040	
		执行时								
	PLS	-	1.800	1.900	1.800	1.900	1.800	1.900	1.300	1.600
	PLF	-	1.800	1.900	1.800	1.900	1.800	1.900	1.600	1.700
	FF	未执行时	0.240		0.160		0.120		0.080	
		执行时	1.700	1.800	1.700	1.800	1.700	1.800	1.200	1.500
	DELTA	未执行时	0.240		0.160		0.120		0.080	
		执行时	4.000	14.700	4.000	14.700	4.000	14.700	2.800	3.600
	SFT	未执行时	0.240		0.160		0.120		0.800	
		执行时	1.800	12.600	1.800	12.600	1.800	12.600	1.600	6.600
	MC	-	0.240		0.160		0.120		0.080	
	MCR	-	0.120		0.080		0.060		0.040	
	FEND	进行出错检查	250.000	250.000	250.000	250.000	250.000	250.000	175.000	252.000
	END	不进行出错检查	250.000	250.000	250.000	250.000	250.000	250.000	175.000	221.000
	STOP	-	-		-		-		-	

分类	指令	条件（软元件）			处理时间（μs）							
					Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
					最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
顺控程序指令	NOP NOPLF PAGE	-			0.120		0.080		0.060		0.040	
基本指令	LDE =	单精度	导通时		4.400	20.900	4.400	20.900	4.400	20.900	4.700	10.100
			非导通时		4.400	20.900	4.400	20.900	4.400	20.900	4.700	10.100
	ANDE =	单精度	未执行时		0.360		0.240		0.180		0.120	
			执行时	导通时	4.200	19.600	4.200	19.600	4.200	19.600	4.200	12.500
				非导通时	4.200	19.600	4.200	19.600	4.200	19.600	4.400	11.900
	ORE =	单精度	未执行时		0.360		0.240		0.180		0.120	
			执行时	导通时	4.200	17.400	4.200	17.400	4.200	17.400	4.600	10.800
				非导通时	4.200	17.400	4.200	17.400	4.200	17.400	4.500	9.800
	LDE < >	单精度	导通时		4.400	20.900	4.400	20.900	4.400	20.900	4.700	7.700
			非导通时		4.400	20.900	4.400	20.900	4.400	20.900	4.600	8.200
	ANDE < >	单精度	未执行时		0.360		0.240		0.180		0.120	
			执行时	导通时	4.200	19.600	4.200	19.600	4.200	19.600	4.300	14.200
				非导通时	4.200	19.600	4.200	19.600	4.200	19.600	4.400	14.200
	ORE < >	单精度	未执行时		0.360		0.240		0.180		0.120	
			执行时	导通时	4.200	17.400	4.200	17.400	4.200	17.400	4.600	6.700
				非导通时	4.200	17.400	4.200	17.400	4.200	17.400	4.400	6.600
	LDE >	单精度	导通时		4.400	20.900	4.400	20.900	4.400	20.900	4.700	13.700
			非导通时		4.400	20.900	4.400	20.900	4.400	20.900	4.600	13.700
	ANDE >	单精度	未执行时		0.360		0.240		0.180		0.120	
			执行时	导通时	4.200	19.600	4.200	19.600	4.200	19.600	4.300	8.100
				非导通时	4.200	19.600	4.200	19.600	4.200	19.600	4.200	8.100
	ORE >	单精度	未执行时		0.360		0.240		0.180		0.120	
			执行时	导通时	4.200	17.400	4.200	17.400	4.200	17.400	4.600	8.500
				非导通时	4.200	17.400	4.200	17.400	4.200	17.400	4.400	8.100
	LDE< =	单精度	导通时		4.400	20.900	4.400	20.900	4.400	20.900	4.700	11.100
			非导通时		4.400	20.900	4.400	20.900	4.400	20.900	4.700	9.600
	ANDE< =	单精度	未执行时		0.360		0.240		0.180		0.120	
			执行时	导通时	4.200	19.600	4.200	19.600	4.200	19.600	4.100	7.800
				非导通时	4.200	19.600	4.200	19.600	4.200	19.600	4.400	8.200
	ORE< =	单精度	未执行时		0.360		0.240		0.180		0.120	
			执行时	导通时	4.200	17.400	4.200	17.400	4.200	17.400	4.500	10.300
				非导通时	4.200	17.400	4.200	17.400	4.200	17.400	4.400	9.800
	LDE<	单精度	导通时		4.400	20.900	4.400	20.900	4.400	20.900	4.700	11.500
			非导通时		4.400	20.900	4.400	20.900	4.400	20.900	4.700	10.900
	ANDE<	单精度	未执行时		0.360		0.240		0.180		0.120	
			执行时	导通时	4.200	19.600	4.200	19.600	4.200	19.600	4.300	9.200
				非导通时	4.200	19.600	4.200	19.600	4.200	19.600	4.400	9.400

分类	指令	条件（软元件）			处理时间（μs）							
					Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
					最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
基本指令	ORE<	单精度	未执行时		0.360		0.240		0.180		0.120	
			执行时	导通时	4.200	17.400	4.200	17.400	4.200	17.400	4.600	10.400
				非导通时	4.200	17.400	4.200	17.400	4.200	17.400	4.400	9.800
	LDE > =	单精度	导通时		4.400	20.900	4.400	20.900	4.400	20.900	4.700	12.200
			非导通时		4.400	20.900	4.400	20.900	4.400	20.900	4.700	11.800
	ANDE > =	单精度	未执行时		0.360		0.240		0.180		0.120	
			执行时	导通时	4.200	19.600	4.200	19.600	4.200	19.600	4.100	6.700
				非导通时	4.200	19.600	4.200	19.600	4.200	19.600	4.400	7.000
	ORE > =	单精度	未执行时		0.360		0.240		0.180		0.120	
			执行时	导通时	4.200	17.400	4.200	17.400	4.200	17.400	4.600	14.000
				非导通时	4.200	17.400	4.200	17.400	4.200	17.400	4.500	14.300
	LDED =	双精度	导通时		4.700	37.400	4.700	37.400	4.700	37.400	4.200	21.000
			非导通时		4.700	37.400	4.700	37.400	4.700	37.400	5.100	21.900
	ANDED =	双精度	未执行时		0.360		0.240		0.180		0.120	
			执行时	导通时	4.500	34.700	4.500	34.700	4.500	34.700	3.800	17.800
				非导通时	4.500	34.700	4.500	34.700	4.500	34.700	4.100	18.100
	ORED =	双精度	未执行时		0.360		0.240		0.180		0.120	
			执行时	导通时	4.700	33.200	4.700	33.200	4.700	33.200	4.100	23.800
				非导通时	4.700	33.200	4.700	33.200	4.700	33.200	4.900	25.500
	LDED < >	双精度	导通时		4.700	37.400	4.700	37.400	4.700	37.400	5.100	23.500
			非导通时		4.700	37.400	4.700	37.400	4.700	37.400	4.200	22.600
	ANDED < >	双精度	未执行时		0.360		0.240		0.180		0.120	
			执行时	导通时	4.500	34.700	4.500	34.700	4.500	34.700	4.000	18.800
				非导通时	4.500	34.700	4.500	34.700	4.500	34.700	4.000	18.700
	ORED < >	双精度	未执行时		0.360		0.240		0.180		0.120	
			执行时	导通时	4.700	33.200	4.700	33.200	4.700	33.200	5.000	25.200
				非导通时	4.700	33.200	4.700	33.200	4.700	33.200	4.100	23.400
	LDED >	双精度	导通时		4.700	37.400	4.700	37.400	4.700	37.400	5.100	25.100
			非导通时		4.700	37.400	4.700	37.400	4.700	37.400	4.200	23.400
	ANDED >	双精度	未执行时		0.360		0.240		0.180		0.120	
			执行时	导通时	4.500	34.700	4.500	34.700	4.500	34.700	4.000	19.500
				非导通时	4.500	34.700	4.500	34.700	4.500	34.700	4.100	19.700
	ORED >	双精度	未执行时		0.360		0.240		0.180		0.120	
			执行时	导通时	4.700	33.200	4.700	33.200	4.700	33.200	5.000	24.200
				非导通时	4.700	33.200	4.700	33.200	4.700	33.200	4.900	25.800
	LDED< =	双精度	导通时		4.700	37.400	4.700	37.400	4.700	37.400	4.200	22.500
			非导通时		4.700	37.400	4.700	37.400	4.700	37.400	4.200	13.500
	ANDED< =	双精度	未执行时		0.360		0.240		0.180		0.120	
			执行时	导通时	4.500	34.700	4.500	34.700	4.500	34.700	4.000	19.600
				非导通时	4.500	34.700	4.500	34.700	4.500	34.700	4.100	19.700
	ORED< =	双精度	未执行时		0.360		0.240		0.180		0.120	
			执行时	导通时	4.700	33.200	4.700	33.200	4.700	33.200	5.000	26.300
				非导通时	4.700	33.200	4.700	33.200	4.700	33.200	5.000	25.200

分类	指令	条件（软元件）			处理时间（μs）							
					Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
					最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
基本指令	LDED<	双精度	导通时		4.700	37.400	4.700	37.400	4.700	37.400	5.100	25.000
			非导通时		4.700	37.400	4.700	37.400	4.700	37.400	4.200	24.100
	ANDED<	双精度	未执行时		0.360		0.240		0.180		0.120	
			执行时	导通时	4.500	34.700	4.500	34.700	4.500	34.700	4.000	19.400
				非导通时	4.500	34.700	4.500	34.700	4.500	34.700	4.100	19.700
	ORED<	双精度	未执行时		0.360		0.240		0.180		0.120	
			执行时	导通时	4.700	33.200	4.700	33.200	4.700	33.200	5.000	25.100
				非导通时	4.700	33.200	4.700	33.200	4.700	33.200	5.000	25.100
	LDED > =	双精度	导通时		4.700	37.400	4.700	37.400	4.700	37.400	4.200	13.100
			非导通时		4.700	37.400	4.700	37.400	4.700	37.400	4.300	13.100
	ANDED > =	双精度	未执行时		0.360		0.240		0.180		0.120	
			执行时	导通时	4.500	34.700	4.500	34.700	4.500	34.700	3.900	19.500
				非导通时	4.500	34.700	4.500	34.700	4.500	34.700	4.100	19.800
	ORED > =	双精度	未执行时		0.360		0.240		0.180		0.120	
			执行时	导通时	4.700	33.200	4.700	33.200	4.700	33.200	5.000	25.100
				非导通时	4.700	33.200	4.700	33.200	4.700	33.200	4.200	18.500
	LD$ =	导通时			8.300	38.500	8.300	38.500	8.300	38.500	5.500	14.900
		非导通时			8.300	38.500	8.300	38.500	8.300	38.500	5.500	15.600
	AND$ =	未执行时			0.360		0.240		0.180		0.120	
		执行时	导通时		7.200	37.300	7.200	37.300	7.200	37.300	5.200	13.800
			非导通时		7.200	37.300	7.200	37.300	7.200	37.300	5.300	14.500
	OR$ =	未执行时			0.360		0.240		0.180		0.120	
		执行时	导通时		7.500	36.600	7.500	36.600	7.500	36.600	5.500	14.900
			非导通时		7.500	36.600	7.500	36.600	7.500	36.600	5.300	14.600
	LD$ < >	导通时			8.300	39.300	8.300	39.300	8.300	39.300	5.600	15.200
		非导通时			8.300	39.300	8.300	39.300	8.300	39.300	5.600	15.400
	AND$ < >	未执行时			0.360		0.240		0.180		0.120	
		执行时	导通时		8.000	38.200	8.000	38.200	8.000	38.200	4.300	21.500
			非导通时		8.000	38.200	8.000	38.200	8.000	38.200	4.500	23.400
	OR$ < >	未执行时			0.360		0.240		0.180		0.120	
		执行时	导通时		8.300	37.300	8.300	37.300	8.300	37.300	5.400	17.700
			非导通时		8.300	37.300	8.300	37.300	8.300	37.300	5.300	19.400
	LD$ >	导通时			8.300	41.600	8.300	41.600	8.300	41.600	6.400	19.200
		非导通时			8.300	41.600	8.300	41.600	8.300	41.600	5.600	20.100
	AND$ >	未执行时			0.360		0.240		0.180		0.120	
		执行时	导通时		8.000	38.100	8.000	38.100	8.000	38.100	4.500	15.400
			非导通时		8.000	38.100	8.000	38.100	8.000	38.100	4.600	15.300
	OR$ >	未执行时			0.360		0.240		0.180		0.120	
		执行时	导通时		8.200	35.700	8.200	35.700	8.200	35.700	5.400	20.000
			非导通时		8.200	35.700	8.200	35.700	8.200	35.700	5.400	22.100
	LD$< =	导通时			8.300	39.200	8.300	39.200	8.300	39.200	5.800	12.800
		非导通时			8.300	39.200	8.300	39.200	8.300	39.200	6.300	13.900
	AND$< =	未执行时			0.360		0.240		0.180		0.120	
		执行时	导通时		7.100	36.500	7.100	36.500	7.100	36.500	6.000	16.000
			非导通时		7.100	36.500	7.100	36.500	7.100	36.500	6.100	16.200

分类	指令	条件（软元件）		处理时间（μs）							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
基本指令	OR$< =	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时	7.400	35.600	7.400	35.600	7.400	35.600	4.700	14.600
			非导通时	7.400	35.600	7.400	35.600	7.400	35.600	4.600	14.400
	LD$<	导通时		7.400	40.000	7.400	40.000	7.400	40.000	4.800	17.000
		非导通时		7.400	40.000	7.400	40.000	7.400	40.000	5.500	18.000
	AND$<	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时	8.000	37.300	8.000	37.300	8.000	37.300	5.900	13.400
			非导通时	8.000	37.300	8.000	37.300	8.000	37.300	6.200	14.500
	OR$<	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时	8.300	35.600	8.300	35.600	8.300	35.600	6.200	18.700
			非导通时	8.300	35.600	8.300	35.600	8.300	35.600	5.400	19.700
	LD$ > =	导通时		7.400	38.300	7.400	38.300	7.400	38.300	4.800	10.000
		非导通时		7.400	38.300	7.400	38.300	7.400	38.300	5.500	11.200
	AND$ > =	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时	7.200	37.300	7.200	37.300	7.200	37.300	4.400	21.600
			非导通时	7.200	37.300	7.200	37.300	7.200	37.300	4.500	21.800
	OR$ > =	未执行时		0.360		0.240		0.180		0.120	
		执行时	导通时	8.200	36.400	8.200	36.400	8.200	36.400	5.400	15.400
			非导通时	8.200	36.400	8.200	36.400	8.200	36.400	5.300	15.300
	BKCMP = (S1) (S2) (D) n	n = 1		15.300	36.100	15.300	36.100	15.300	36.100	8.200	22.600
		n = 96		64.500	85.500	64.500	85.500	64.500	85.500	57.400	72.500
	BKCMP < > (S1) (S2) (D) n	n = 1		15.300	36.100	15.300	36.100	15.300	36.100	8.200	22.500
		n = 96		66.600	87.500	66.600	87.500	66.600	87.500	59.500	74.500
	BKCMP > (S1) (S2) (D) n	n = 1		15.300	36.100	15.300	36.100	15.300	36.100	8.200	23.100
		n = 96		66.600	87.500	66.600	87.500	66.600	87.500	59.500	74.400
	BKCMP< = (S1) (S2) (D) n	n = 1		15.300	36.100	15.300	36.100	15.300	36.100	8.200	22.500
		n = 96		64.500	85.500	64.500	85.500	64.500	85.500	57.400	72.400
	BKCMP< (S1) (S2) (D) n	n = 1		15.300	36.100	15.300	36.100	15.300	36.100	8.300	23.000
		n = 96		66.600	87.500	66.600	87.500	66.600	87.500	59.500	74.500
	BKCMP > = (S1) (S2) (D) n	n = 1		15.300	36.100	15.300	36.100	15.300	36.100	8.200	22.500
		n = 96		64.500	85.500	64.500	85.500	64.500	85.500	57.400	72.400
	DBKCMP = (S1) (S2) (D) n	n = 1		15.800	36.300	15.800	36.300	15.800	36.300	9.350	29.000
		n = 96		64.900	85.700	64.900	85.700	64.900	85.700	60.700	78.400
	DBKCMP < > (S1) (S2) (D) n	n = 1		15.700	36.300	15.700	36.300	15.700	36.300	9.350	28.900
		n = 96		67.000	87.700	67.000	87.700	67.000	87.700	62.500	80.300
	DBKCMP > (S1) (S2) (D) n	n = 1		15.800	36.300	15.800	36.300	15.800	36.300	9.350	29.000
		n = 96		67.000	87.700	67.000	87.700	67.000	87.700	62.600	80.300
	DBKCMP< = (S1) (S2) (D) n	n = 1		15.700	36.300	15.700	36.300	15.700	36.300	9.350	29.000
		n = 96		64.800	85.700	64.800	85.700	64.800	85.700	60.800	78.400
	DBKCMP< (S1) (S2) (D) n	n = 1		15.800	36.300	15.800	36.300	15.800	36.300	9.350	29.000
		n = 96		67.000	87.700	67.000	87.700	67.000	87.700	62.700	80.400
	DBKCMP > = (S1) (S2) (D) n	n = 1		15.700	36.300	15.700	36.300	15.700	36.300	9.300	29.000
		n = 96		64.800	85.700	64.800	85.700	64.800	85.700	60.700	78.400
	DB + (S) (D)	执行时		5.750	13.300	5.750	13.300	5.750	13.300	4.900	7.500
	DB + (S1) (S2) (D)	执行时		5.650	13.200	5.650	13.200	5.650	13.200	5.200	11.000
	DB - (S) (D)	执行时		5.750	12.700	5.750	12.700	5.750	12.700	4.900	10.200
	DB - (S1) (S2) (D)	执行时		5.650	12.600	5.650	12.600	5.650	12.600	5.200	8.600
	DB * (S1) (S2) (D)	执行时		8.750	40.200	8.750	40.200	8.750	40.200	8.300	22.200
	DB / (S1) (S2) (D)	执行时		5.750	21.500	5.750	21.500	5.750	21.500	6.100	19.200

分类	指令	条件（软元件）		处理时间（μs）							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
基本指令	ED + (S) (D)	双精度	(S) = 0, (D) = 0	4.500	26.700	4.500	26.700	4.500	26.700	4.800	16.800
			(S) = 2^{1023}, (D) = 2^{1023}	5.800	32.900	5.800	32.900	5.800	32.900	4.800	16.800
	ED + (S1) (S2) (D)	双精度	(S1) = 0, (S2) = 0	5.450	35.400	5.450	35.400	5.450	35.400	7.100	20.100
			(S1) = 2^{1023}, (S2) = 2^{1023}	6.750	41.400	6.750	41.400	6.750	41.400	7.100	20.100
	ED - (S) (D)	双精度	(S) = 0, (D) = 0	5.200	25.900	5.200	25.900	5.200	25.900	5.000	17.300
			(S) = 2^{1023}, (D) = 2^{1023}	6.000	27.700	6.000	27.700	6.000	27.700	5.000	17.300
	ED - (S1) (S2) (D)	双精度	(S1) = 0, (S2) = 0	5.550	32.900	5.550	32.900	5.550	32.900	6.000	16.300
			(S1) = 2^{1023}, (S2) = 2^{1023}	5.750	33.900	5.750	33.900	5.750	33.900	6.000	16.300
	ED * (S1) (S2) (D)	双精度	(S1) = 0, (S2) = 0	5.550	34.400	5.550	34.400	5.550	34.400	10.500	22.300
			(S1) = 2^{1023}, (S2) = 2^{1023}	5.950	39.100	5.950	39.100	5.950	39.100	10.500	22.300
	ED / (S1) (S2) (D)	双精度	(S1) = 2^{1023}, (S2) = 2^{1023}	8.050	44.200	8.050	44.200	8.050	44.200	7.500	27.200
	BK + (S1) (S2) (D) n	n = 1		13.500	28.500	13.500	28.500	13.500	28.500	12.100	19.700
		n = 96		63.100	78.200	63.100	78.200	63.100	78.200	61.700	69.300
	BK - (S1) (S2) (D) n	n = 1		13.500	28.500	13.500	28.500	13.500	28.500	12.100	20.600
		n = 96		63.100	78.200	63.100	78.200	63.100	78.200	61.700	70.200
	DBK + (S1) (S2) (D) n	n = 1		10.100	24.200	10.100	24.200	10.100	24.200	7.050	19.200
		n = 96		59.800	73.900	59.800	73.900	59.800	73.900	59.400	68.900
	DBK - (S1) (S2) (D) n	n = 1		10.100	24.200	10.100	24.200	10.100	24.200	7.050	19.900
		n = 96		59.800	73.900	59.800	73.900	59.800	73.900	59.400	69.600
	$+ (S) (D)	-		15.400	64.300	15.400	64.300	15.400	64.300	14.400	34.000
	$+ (S1) (S2) (D)	-		19.700	71.000	19.700	71.000	19.700	71.000	9.200	22.900
	FLTD	双精度	(S) = 0	3.100	19.600	3.100	19.600	3.100	19.600	4.000	8.900
			(S) = 7FFFH	3.350	19.900	3.350	19.900	3.350	19.900	3.400	9.000
	DFLTD	双精度	(S) = 0	3.200	20.400	3.200	20.400	3.200	20.400	4.100	10.800
			(S) = 7FFFFFFFH	3.450	20.500	3.450	20.500	3.450	20.500	3.600	10.800
	INTD	双精度	(S) = 0	3.200	22.900	3.200	22.900	3.200	22.900	3.500	9.300
			(S) = 32766.5	4.100	34.300	4.100	34.300	4.100	34.300	5.100	19.500
	DINTD	双精度	(S) = 0	3.200	23.000	3.200	23.000	3.200	23.000	2.600	6.800
			(S) = 1234567890.3	4.050	33.500	4.050	33.500	4.050	33.500	3.400	11.700

分类	指令	条件（软元件）	处理时间（μs）							
			Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
			最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
基本指令	DBL	执行时	3.300	5.900	3.300	5.900	3.300	5.900	2.700	3.800
	WORD	执行时	3.000	7.250	3.000	7.250	3.000	7.250	2.900	7.000
	GRY	执行时	3.350	7.500	3.350	7.500	3.350	7.500	2.700	6.100
	DGRY	执行时	3.000	7.200	3.000	7.200	3.000	7.200	2.900	4.600
	GBIN	执行时	4.600	9.700	4.600	9.700	4.600	9.700	4.000	8.200
	DGBIN	执行时	5.550	10.700	5.550	10.700	5.550	10.700	5.500	8.000
	NEG	执行时	3.300	6.850	3.300	6.850	3.300	6.850	2.400	4.100
	DNEG	执行时	3.050	5.700	3.050	5.700	3.050	5.700	2.500	4.300
	ENEG	浮点 = 0	3.100	7.350	3.100	7.350	3.100	7.350	2.500	3.400
		浮点 = -1.0	3.350	11.700	3.350	11.700	3.350	11.700	2.700	4.500
	EDNEG	浮点 = 0	3.000	21.200	3.000	21.200	3.000	21.200	2.200	3.500
		浮点 = -1.0	3.100	22.900	3.100	22.900	3.100	22.900	2.400	3.500
	BKBCD (S) (D) n	n = 1	8.700	27.600	8.700	27.600	8.700	27.600	9.700	22.000
		n = 96	84.200	104.000	84.200	104.000	84.200	104.000	74.200	86.500
	BKBIN (S) (D) n	n = 1	8.450	28.100	8.450	28.100	8.450	28.100	8.900	16.300
		n = 96	56.100	75.800	56.100	75.800	56.100	75.800	58.500	65.100
	ECON	-	3.100	21.300	3.100	21.300	3.100	21.300	4.300	6.800
	EDCON	-	5.050	24.000	5.050	24.000	5.050	24.000	2.800	5.400
	EDMOV	-	2.900	22.900	2.900	22.900	2.900	22.900	3.200	7.800
	$MOV	传送字符串 = 0	6.250	30.100	6.250	30.100	6.250	30.100	4.500	13.900
		传送字符串 = 32	15.500	39.300	15.500	39.300	15.500	39.300	15.400	17.500
	BXCH (D1) (D2) n	n = 1	8.400	20.900	8.400	20.900	8.400	20.900	8.700	15.200
		n = 96	67.100	79.900	67.100	79.900	67.100	79.900	67.200	74.000
	SWAP	-	3.300	3.550	3.300	3.550	3.300	3.550	2.400	2.700
	GOEND	-	0.550		0.550		0.550		0.500	
	DI	-	2.800	8.400	2.800	8.400	2.800	8.400	1.800	2.200
	EI	-	4.300	12.300	4.300	12.300	4.300	12.300	3.100	3.800
	IMASK	-	12.900	40.600	12.900	40.600	12.900	40.600	9.800	25.000
	IRET	-	1.000		1.000		1.000		1.000	
	RFS X n	n = 1	7.500	26.500	7.500	26.500	7.500	26.500	4.300	16.100
		n = 96	11.400	30.400	11.400	30.400	11.400	30.400	11.400	23.700
	RFS Y n	n = 1	7.300	26.300	7.300	26.300	7.300	26.300	3.800	10.000
		n = 96	10.900	29.900	10.900	29.900	10.900	29.900	8.500	15.200
	UDCNT1	-	1.500	7.100	1.500	7.100	1.500	7.100	1.000	2.000
	UDCNT2	-	1.500	6.300	1.500	6.300	1.500	6.300	1.000	4.000
	TTMR	-	5.300	20.900	5.300	20.900	5.300	20.900	3.900	6.100
	STMR	-	8.900	49.800	8.900	49.800	8.900	49.800	7.200	30.000
	ROTC	-	52.300	52.600	52.300	52.600	52.300	52.600	15.200	16.100
	RAMP	-	7.400	30.900	7.400	30.900	7.400	30.900	5.900	18.300
	SPD	-	1.500	6.300	1.500	6.300	1.500	6.300	1.000	2.800
	PLSY	-	6.400	7.100	6.400	7.100	6.400	7.100	3.500	4.700
	PWM	-	3.900	4.600	3.900	4.600	3.900	4.600	3.400	3.400
	MTR	-	10.100	61.400	10.100	61.400	10.100	61.400	20.500	28.400

分类	指令	条件（软元件）		处理时间（μs）							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
应用指令	BKAND (S1) (S2) (D) n	n = 1		13.600	28.500	13.600	28.500	13.600	28.500	12.100	20.100
		n = 96		63.200	78.200	63.200	78.200	63.200	78.200	57.400	63.200
	BKOR (S1) (S2) (D) n	n = 1		13.500	28.500	13.500	28.500	13.500	28.500	7.700	13.200
		n = 96		63.100	78.200	63.100	78.200	63.100	78.200	57.400	62.800
	BKXOR (S1) (S2) (D) n	n = 1		13.600	28.300	13.600	28.300	13.600	28.300	7.800	13.200
		n = 96		63.100	78.000	63.100	78.000	63.100	78.000	57.300	62.800
	BKXNR (S1) (S2) (D) n	n = 1		13.500	28.300	13.500	28.300	13.500	28.300	7.800	14.100
		n = 96		63.100	78.000	63.100	78.000	63.100	78.000	57.400	62.900
	BSFR (D) n	n = 1		5.050	21.100	5.050	21.100	5.050	21.100	3.700	6.300
		n = 96		9.000	34.800	9.000	34.800	9.000	34.800	10.200	12.800
	BSFL (D) n	n = 1		4.800	19.100	4.800	19.100	4.800	19.100	4.500	8.900
		n = 96		8.550	34.300	8.550	34.300	8.550	34.300	10.100	14.300
	SFTBR (D) n1 n2	进行了移位的位的数=16/移位数 = 1		10.300	46.500	10.300	46.500	10.300	46.500	8.800	43.400
		进行了移位的位的数=16/移位数 = 15		10.300	46.400	10.300	46.400	10.300	46.400	8.750	43.400
	SFTBL (D) n1 n2	进行了移位的位的数=16/移位数 = 1		10.500	49.800	10.500	49.800	10.500	49.800	8.050	45.100
		进行了移位的位的数=16/移位数 = 15		10.500	49.800	10.500	49.800	10.500	49.800	8.050	45.100
	SFTWR (D) n1 n2	进行了移位的位的数=16/移位数 = 1		7.950	24.000	7.950	24.000	7.950	24.000	6.500	22.800
		进行了移位的位的数=16/移位数 = 15		7.950	24.100	7.950	24.100	7.950	24.100	6.500	22.800
	SFTWL (D) n1 n2	进行了移位的位的数=16/移位数 = 1		8.700	23.600	8.700	23.600	8.700	23.600	7.350	23.600
		进行了移位的位的数=16/移位数 = 15		8.650	23.700	8.650	23.700	8.650	23.700	7.300	23.700
	BSET (D) n	n = 1		4.550	4.750	4.550	4.750	4.550	4.750	3.000	3.400
		n = 15		4.550	4.750	4.550	4.750	4.550	4.750	3.000	3.500
	BRST (D) n	n = 1		4.600	4.750	4.600	4.750	4.600	4.750	3.000	3.400
		n = 15		4.600	4.750	4.600	4.750	4.600	4.750	3.000	3.400
	TEST	执行时		7.250	13.200	7.250	13.200	7.250	13.200	4.400	6.900
	DTEST	执行时		6.950	12.900	6.950	12.900	6.950	12.900	4.500	7.000
	BKRST (S) n	n = 1		7.350	11.600	7.350	11.600	7.350	11.600	4.300	5.200
		n = 96		10.100	22.600	10.100	22.600	10.100	22.600	6.500	13.200
	SER (S1) (S2) (D) n	n = 1	全部一致	6.650	6.800	6.650	6.800	6.650	6.800	5.000	5.300
			全部不一致	6.650	6.800	6.650	6.800	6.650	6.800	5.000	5.300
		n = 96	全部一致	34.000	42.300	34.000	42.300	34.000	42.300	32.300	35.900
			全部不一致	34.000	42.300	34.000	42.300	34.000	42.300	32.400	35.900
	DSER (S1) (S2) (D) n	n = 1	全部一致	8.000	16.300	8.000	16.300	8.000	16.300	6.800	10.200
			全部不一致	8.000	16.300	8.000	16.300	8.000	16.300	6.800	10.200
		n = 96	全部一致	54.100	62.600	54.100	62.600	54.100	62.600	52.800	56.300
			全部不一致	54.100	62.600	54.100	62.600	54.100	62.600	52.800	56.300
	DSUM (S) (D)	(S) = 0		4.100	4.200	4.100	4.200	4.100	4.200	3.700	4.100
		(S) = FFFFFFFFH		4.100	4.200	4.100	4.200	4.100	4.200	3.800	4.100
	DECO (S) (D) n	n = 2		8.850	23.000	8.850	23.000	8.850	23.000	6.000	16.400
		n = 8		13.600	36.600	13.600	36.600	13.600	36.600	8.100	15.200
	ENCO (S) (D) n	n = 2	M1 = ON	7.650	11.900	7.650	11.900	7.650	11.900	5.300	6.300
			M4 = ON	7.500	11.700	7.500	11.700	7.500	11.700	5.200	6.200
		n = 8	M1 = ON	14.600	27.800	14.600	27.800	14.600	27.800	10.400	17.900
			M256 = ON	10.600	23.700	10.600	23.700	10.600	23.700	5.700	13.300

分类	指令	条件（软元件）	处理时间（μs）							
			Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
			最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
应用指令	DIS (S) (D) n	n = 1	6.500	14.800	6.500	14.800	6.500	14.800	5.000	10.900
		n = 4	6.900	15.200	6.900	15.200	6.900	15.200	5.400	11.300
	UNI (S) (D) n	n = 1	6.800	15.100	6.800	15.100	6.800	15.100	5.500	8.900
		n = 4	7.500	15.900	7.500	15.900	7.500	15.900	6.200	9.600
	NDIS	执行时	4.750	18.700	4.750	18.700	4.750	18.700	11.000	16.300
	NUNI	执行时	4.750	18.700	4.750	18.700	4.750	18.700	10.600	16.000
	WTOB (S) (D) n	n = 1	6.600	14.900	6.600	14.900	6.600	14.900	5.000	6.500
		n = 96	37.700	46.100	37.700	46.100	37.700	46.100	36.000	38.400
	BTOW (S) (D) n	n = 1	7.350	15.600	7.350	15.600	7.350	15.600	5.100	6.100
		n = 96	32.100	40.500	32.100	40.500	32.100	40.500	29.900	32.000
	MAX (S) (D) n	n = 1	8.250	24.900	8.250	24.900	8.250	24.900	4.300	6.900
		n = 96	34.200	51.600	34.200	51.600	34.200	51.600	32.000	34.300
	MIN (S) (D) n	n = 1	8.250	24.800	8.250	24.800	8.250	24.800	4.400	6.800
		n = 96	34.200	51.600	34.200	51.600	34.200	51.600	30.300	34.800
	DMAX (S) (D) n	n = 1	6.800	34.900	6.800	34.900	6.800	34.900	4.800	14.200
		n = 96	60.300	89.200	60.300	89.200	60.300	89.200	56.400	68.000
	DMIN (S) (D) n	n = 1	7.600	35.700	7.600	35.700	7.600	35.700	4.800	9.300
		n = 96	59.400	90.000	59.400	90.000	59.400	90.000	55.400	62.800
	SORT (S1) n (S2) (D1) (D2)	n = 1	10.100	28.900	10.100	28.900	10.100	28.900	6.200	12.200
		n = 96	52.100	92.400	52.100	92.400	52.100	92.400	6.200	13.100
	DSORT (S1) n (S2) (D1) (D2)	n = 1	9.300	29.000	9.300	29.000	9.300	29.000	6.200	10.500
		n = 96	43.600	89.600	43.600	89.600	43.600	89.600	6.100	10.500
	WSUM (S) (D) n	n = 1	6.700	15.000	6.700	15.000	6.700	15.000	4.800	6.200
		n = 96	28.900	37.100	28.900	37.100	28.900	37.100	26.900	28.700
	DWSUM (S) (D) n	n = 1	8.600	26.800	8.600	26.800	8.600	26.800	5.500	7.000
		n = 96	56.200	74.700	56.200	74.700	56.200	74.700	53.000	56.300
	MEAN (S) (D) n	n = 1	5.850	19.800	5.850	19.800	5.850	19.800	4.300	17.300
		n = 96	17.300	38.200	17.300	38.200	17.300	38.200	16.000	35.500
	DMEAN (S) (D) n	n = 1	6.900	23.300	6.900	23.300	6.900	23.300	5.750	21.900
		n = 96	29.400	49.900	29.400	49.900	29.400	49.900	29.200	48.600
	NEXT	-	1.000	1.100	1.000	1.100	1.000	1.100	0.980	1.400
	BREAK	-	4.700	25.000	4.700	25.000	4.700	25.000	21.300	17.900
	RET	返回至本程序	4.100	19.500	4.100	19.500	4.100	19.500	2.000	3.000
		返回至其它程序	4.700	16.700	4.700	16.700	4.700	16.700	2.300	4.900
	FCALL Pn	文件内指针	5.400	5.400	5.400	5.400	5.400	5.400	3.300	5.300
		公共指针	7.600	30.500	7.600	30.500	7.600	30.500	4.900	6.600
	FCALL Pn (S1) ～ (S5)	-	50.400	62.700	50.400	62.700	50.400	62.700	19.800	23.700
	ECALL * Pn *: 程序名	-	105.000	214.000	105.000	214.000	105.000	214.000	75.700	134.000
	ECALL * Pn (S1) ～ (S5) *: 程序名	-	164.000	271.000	164.000	271.000	164.000	271.000	109.000	173.000
	EFCALL * Pn *: 程序名	-	105.000	214.000	105.000	214.000	105.000	214.000	76.200	134.000
	EFCALL * Pn (S1) ～ (S5) *: 程序名	-	164.000	271.000	164.000	271.000	164.000	271.000	90.500	170.000
	XCALL	-	5.100	6.700	5.100	6.700	5.100	6.700	3.800	6.400

分类	指令	条件（软元件）	处理时间（μs）							
			Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
			最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
应用指令	CCOM COM	仅选择 I/O 刷新时	18.100	89.100	18.100	89.100	18.100	89.100	12.800	79.000
		仅选择 CC-Link 刷新时（主站侧）	33.300	132.000	33.300	132.000	33.300	132.000	24.900	119.000
		仅选择 CC-Link 刷新时（本地站侧）	33.300	132.000	33.300	132.000	33.300	132.000	24.900	119.000
		仅选择 MELSECNET/H 刷新时（管理站侧）	78.600	231.000	78.600	231.000	78.600	231.000	54.000	212.000
		仅选择 MELSECNET/H 刷新时（普通站侧）	78.600	231.000	78.600	231.000	78.600	231.000	54.000	212.000
		仅选择智能自动刷新时	18.100	89.000	18.100	89.000	18.100	89.000	12.800	79.000
		仅选择组外输入输出时（仅输入）	15.700	71.600	15.700	71.600	15.700	71.600	8.600	76.500
		仅选择组外输入输出时（仅输出）	40.200	152.000	40.200	152.000	40.200	152.000	26.300	135.000
		仅选择组外输入输出时（输入及输出均选择）	45.800	153.000	45.800	153.000	45.800	153.000	26.100	135.000
		进行选择多 CPU 高速通信区刷新时	－	－	－	－	－	－	－	－
		仅选择与外围设备通信时	18.200	89.000	18.200	89.000	18.200	89.000	7.250	54.300

分类	指令	条件（软元件）	处理时间（μs）							
			Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
			最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
应用指令	FIFW	数据数 = 0	6.100	14.200	6.100	14.200	6.100	14.200	3.700	10.100
		数据数 = 96	6.100	14.200	6.100	14.200	6.100	14.200	3.800	5.200
	FIFR	数据数 = 1	7.500	15.600	7.500	15.600	7.500	15.600	4.400	5.800
		数据数 = 96	37.000	45.000	37.000	45.000	37.000	45.000	33.500	35.200
	FPOP	数据数 = 1	7.600	15.600	7.600	15.600	7.600	15.600	4.400	10.800
		数据数 = 96	7.600	15.600	7.600	15.600	7.600	15.600	4.400	10.800
	FINS	数据数 = 0	6.900	15.000	6.900	15.000	6.900	15.000	5.000	10.700
		数据数 = 96	36.600	44.700	36.600	44.700	36.600	44.700	4.400	10.900
	FDEL	数据数 = 1	8.000	16.100	8.000	16.100	8.000	16.100	4.900	11.300
		数据数 = 96	37.300	45.500	37.300	45.500	37.300	45.500	34.200	35.900
	FROM n1 n2 (D) n3	n3 = 1	17.400	74.600	17.400	74.600	17.400	74.600	12.200	36.300
		n3 = 1000	399.000	499.000	399.000	499.000	399.000	499.000	465.000	539.000
	DFRO n1 n2 (D) n3	n3 = 1	18.200	86.000	18.200	86.000	18.200	86.000	11.300	42.200
		n3 = 500	399.000	498.000	399.000	498.000	399.000	498.000	457.000	535.000
	TO n1 n2 (S) n3	n3 = 1	16.400	68.800	16.400	68.800	16.400	68.800	8.400	22.600
		n3 = 1000	381.000	472.000	381.000	472.000	381.000	472.000	483.000	539.000
	DTO n1 n2 (S) n3	n3 = 1	18.400	82.700	18.400	82.700	18.400	82.700	8.800	29.600
		n3 = 500	380.000	470.000	380.000	470.000	380.000	470.000	476.000	537.000

分类	指令	条件（软元件）	处理时间（μs）							
			Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
			最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
应用指令	LEDR	无显示→无显示	1.500	7.100	1.500	7.100	1.500	7.100	5.100	5.100
		LED 指令执行→无显示	38.900	109.000	38.900	109.000	38.900	109.000	35.700	89.200
	BINDA (S) (D)	(S) = 1	5.600	13.900	5.600	13.900	5.600	13.900	4.900	6.500
		(S) = -32768	7.800	16.200	7.800	16.200	7.800	16.200	7.200	8.700
	DBINDA (S) (D)	(S) = 1	6.200	14.500	6.200	14.500	6.200	14.500	5.700	7.100
		(S) = -2147483648	11.000	19.200	11.000	19.200	11.000	19.200	10.400	12.200
	BINHA (S) (D)	(S) = 1	5.050	13.400	5.050	13.400	5.050	13.400	4.400	5.900
		(S) = FFFFH	5.050	13.400	5.050	13.400	5.050	13.400	4.400	5.800
	DBINHA (S) (D)	(S) = 1	5.600	13.900	5.600	13.900	5.600	13.900	5.200	6.700
		(S) = FFFFFFFFH	5.600	13.900	5.600	13.900	5.600	13.900	5.100	6.500
	BCDDA (S) (D)	(S) = 1	4.850	13.200	4.850	13.200	4.850	13.200	4.300	5.800
		(S) = 9999	5.300	13.600	5.300	13.600	5.300	13.600	4.700	6.100
	DBCDDA (S) (D)	(S) = 1	5.300	13.600	5.300	13.600	5.300	13.600	4.800	6.300
		(S) = 99999999	6.200	14.500	6.200	14.500	6.200	14.500	5.600	7.100
	DABIN (S) (D)	(S) = 1	7.000	18.500	7.000	18.500	7.000	18.500	6.500	9.000
		(S) = -32768	6.950	18.500	6.950	18.500	6.950	18.500	6.300	8.900
	DDABIN (S) (D)	(S) = 1	9.450	21.000	9.450	21.000	9.450	21.000	9.400	12.000
		(S) = -2147483648	9.450	21.000	9.450	21.000	9.450	21.000	9.100	11.600
	HABIN (S) (D)	(S) = 1	5.650	17.100	5.650	17.100	5.650	17.100	4.900	7.500
		(S) = FFFFH	5.750	17.300	5.750	17.300	5.750	17.300	5.100	8.100
	DHABIN (S) (D)	(S) = 1	6.800	18.200	6.800	18.200	6.800	18.200	6.000	8.500
		(S) = FFFFFFFFH	7.100	18.600	7.100	18.600	7.100	18.600	6.300	8.900
	DABCD (S) (D)	(S) = 1	5.650	17.200	5.650	17.200	5.650	17.200	5.000	7.500
		(S) = 9999	5.700	17.200	5.700	17.200	5.700	17.200	5.000	7.500
	DDABCD (S) (D)	(S) = 1	6.850	18.300	6.850	18.300	6.850	18.300	6.200	8.800
		(S) = 99999999	6.850	18.300	6.850	18.300	6.850	18.300	6.200	8.800
	COMRD	-	185.000	188.000	185.000	188.000	185.000	188.000	97.300	97.400
	LEN	1 字符	4.700	16.200	4.700	16.200	4.700	16.200	4.100	6.600
		96 字符	20.600	32.900	20.600	32.900	20.600	32.900	19.800	22.400
	STR	-	9.800	36.500	9.800	36.500	9.800	36.500	6.900	14.400
	DSTR	-	12.100	40.400	12.100	40.400	12.100	40.400	10.200	20.800
	VAL	-	12.200	40.900	12.200	40.900	12.200	40.900	9.800	23.900
	DVAL	-	19.400	45.600	19.400	45.600	19.400	45.600	14.000	33.100
	ESTR	-	29.700	87.800	29.700	87.800	29.700	87.800	22.100	52.400

分类	指令	条件（软元件）		处理时间（μs）							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
应用指令	EVAL	小数点形式全2位数指定		23.900	70.400	23.900	70.400	23.900	70.400	23.300	36.500
		指数形式全6位数指定		23.700	70.300	23.700	70.300	23.700	70.300	23.300	36.400
	ASC (S) (D) n	n = 1		10.200	41.800	10.200	41.800	10.200	41.800	5.600	19.700
		n = 96		31.900	66.600	31.900	66.600	31.900	66.600	30.200	44.700
	HEX (S) (D) n	n = 1		8.600	43.400	8.600	43.400	8.600	43.400	7.500	23.100
		n = 96		77.100	115.000	77.100	115.000	77.100	115.000	37.500	53.300
	RIGHT (S) (D) n	n = 1		10.900	29.600	10.900	29.600	10.900	29.600	7.600	11.400
		n = 96		41.400	60.300	41.400	60.300	41.400	60.300	36.300	46.000
	LEFT (S) (D) n	n = 1		10.600	29.300	10.600	29.300	10.600	29.300	6.500	16.100
		n = 96		41.300	60.200	41.300	60.200	41.300	60.200	36.200	46.200
	MIDR	-		11.700	30.600	11.700	30.600	11.700	30.600	9.500	19.100
	MIDW	-		12.400	24.000	12.400	24.000	12.400	24.000	10.300	18.200
	INSTR	无一致		22.000	38.200	22.000	38.200	22.000	38.200	19.300	29.000
		有一致	起始	13.300	29.600	13.300	29.600	13.300	29.600	10.300	20.000
			最终	21.900	38.100	21.900	38.100	21.900	38.100	51.100	60.800
	EMOD	-		11.600	24.000	11.600	24.000	11.600	24.000	10.300	15.300
	EREXP	-		19.700	28.000	19.700	28.000	19.700	28.000	19.300	22.300
	STRINS (S) (D) n	S = 128/D = 40/n = 1		47.000	102.000	47.000	102.000	47.000	102.000	44.300	96.700
		S = 128/D = 40/n = 48		70.100	134.000	70.100	134.000	70.100	134.000	58.800	112.000
	STRDEL (S) (D) n	S = 128/D = 40/n = 1		46.400	93.600	46.400	93.600	46.400	93.600	39.000	78.100
		S = 128/D = 40/n = 48		44.500	70.600	44.500	70.600	44.500	70.600	36.000	69.200
	SIN	单精度		6.400	13.900	6.400	13.900	6.400	13.900	4.500	9.900
	COS	单精度		6.100	13.500	6.100	13.500	6.100	13.500	4.300	8.200
	TAN	单精度		8.300	15.000	8.300	15.000	8.300	15.000	5.100	7.200
	ASIN	单精度		7.300	15.600	7.300	15.600	7.300	15.600	6.100	13.700
	ACOS	单精度		8.100	16.500	8.100	16.500	8.100	16.500	6.800	11.100
	ATAN	单精度		5.350	12.000	5.350	12.000	5.350	12.000	4.000	6.900
	SIND	双精度		13.400	51.300	13.400	51.300	13.400	51.300	9.600	26.000
	COSD	双精度		14.700	51.700	14.700	51.700	14.700	51.700	10.000	26.900
	TAND	双精度		17.400	54.400	17.400	54.400	17.400	54.400	11.400	25.300
	ASIND	双精度		22.600	60.300	22.600	60.300	22.600	60.300	12.100	30.800
	ACOSD	双精度		19.700	60.000	19.700	60.000	19.700	60.000	11.700	28.000
	ATAND	双精度		15.000	51.800	15.000	51.800	15.000	51.800	9.700	22.000
	RAD	单精度		3.200	10.300	3.200	10.300	3.200	10.300	2.500	4.800
	RADD	双精度		5.200	43.100	5.200	43.100	5.200	43.100	4.100	16.400
	DEG	单精度		3.200	11.500	3.200	11.500	3.200	11.500	2.500	4.700
	DEGD	双精度		5.150	43.800	5.150	43.800	5.150	43.800	5.000	18.100
	SQR	单精度		3.900	12.300	3.900	12.300	3.900	12.300	3.500	9.300
	SQRD	双精度		7.000	45.700	7.000	45.700	7.000	45.700	5.700	25.400
	EXP (S) (D)	单精度	(S) = -10	6.350	13.800	6.350	13.800	6.350	13.800	4.000	13.000
			(S) = 1	6.350	13.800	6.350	13.800	6.350	13.800	4.000	13.000
	EXPD (S) (D)	双精度	(S) = -10	15.800	52.700	15.800	52.700	15.800	52.700	8.800	27.600
			(S) = 1	15.400	52.500	15.400	52.500	15.400	52.500	8.500	27.300
	LOG (S) (D)	单精度	(S) = 1	5.800	14.900	5.800	14.900	5.800	14.900	4.100	8.100
			(S) = 10	7.450	16.500	7.450	16.500	7.450	16.500	6.200	10.300
	LOGD (S) (D)	双精度	(S) = 1	11.000	48.900	11.000	48.900	11.000	48.900	9.500	28.300
			(S) = 10	12.600	51.300	12.600	51.300	12.600	51.300	11.100	29.900

分类	指令	条件（软元件）		处理时间（μs）							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
应用指令	RND	-		1.950	5.450	1.950	5.450	1.950	5.450	1.200	2.300
	SRND	-		2.750	4.550	2.750	4.550	2.750	4.550	1.400	2.400
	BSQR (S) (D)	(S) = 0		2.500	6.800	2.500	6.800	2.500	6.800	1.800	3.300
		(S) = 9999		6.400	15.500	6.400	15.500	6.400	15.500	5.100	8.800
	BDSQR (S) (D)	(S) = 0		2.600	6.050	2.600	6.050	2.600	6.050	1.900	3.700
		(S) = 99999999		8.450	17.600	8.450	17.600	8.450	17.600	7.500	10.900
	BSIN	-		11.500	32.800	11.500	32.800	11.500	32.800	8.700	20.200
	BCOS	-		10.400	32.500	10.400	32.500	10.400	32.500	7.800	14.400
	BTAN	-		12.100	33.700	12.100	33.700	12.100	33.700	9.000	17.000
	BASIN	-		13.300	32.800	13.300	32.800	13.300	32.800	12.200	15.100
	BACOS	-		13.400	33.700	13.400	33.700	13.400	33.700	13.100	14.900
	BATAN	-		12.600	31.400	12.600	31.400	12.600	31.400	11.400	15.700
	POW (S1) (S2) (D)	单精度	S1 = 12.3E+5 S2 = 3.45E+0	12.200	22.100	12.200	22.100	12.200	22.100	8.950	19.500
	POWD (S1) (S2) (D)	双精度	S1 = 12.3E+5 S2 = 3.45E+0	27.300	61.000	27.300	61.000	27.300	61.000	19.400	55.200
	LOG10	单精度		8.200	16.500	8.200	16.500	8.200	16.500	5.950	14.800
	LOG10D	双精度		15.100	48.000	15.100	48.000	15.100	48.000	12.400	46.500
	LIMIT	-		5.350	5.500	5.350	5.500	5.350	5.500	5.200	5.400
	DLIMIT	-		6.000	6.150	6.000	6.150	6.000	6.150	5.700	5.900
	BAND	-		5.450	12.400	5.450	12.400	5.450	12.400	5.400	6.300
	DBAND	-		6.050	11.900	6.050	11.900	6.050	11.900	5.800	6.900
	ZONE	-		6.250	10.700	6.250	10.700	6.250	10.700	5.200	11.100
	DZONE	-		6.000	11.900	6.000	11.900	6.000	11.900	5.700	10.800

分类	指令	条件（软元件）		处理时间（μs）							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
应用指令	SCL (S1) (S2) (D)	SM750 = ON	点 No.1 <(S1)< 点 No.2	14.900	50.100	14.900	50.100	14.900	50.100	14.700	48.000
			点 No.9 <(S1)< 点 No.10	15.800	50.900	15.800	50.900	15.800	50.900	19.600	50.400
		SM750 = OFF	点 No.1 <(S1)< 点 No.2	13.900	53.100	13.900	53.100	13.900	53.100	13.700	51.000
			点 No.9 <(S1)< 点 No.10	16.600	56.600	16.600	56.600	16.600	56.600	20.400	56.200
	DSCL (S1) (S2) (D)	SM750 = ON	点 No.1 <(S1)< 点 No.2	13.400	52.400	13.400	52.400	13.400	52.400	12.800	50.300
			点 No.9 <(S1)< 点 No.10	14.200	54.100	14.200	54.100	14.200	54.100	17.300	53.500
		SM750 = OFF	点 No.1 <(S1)< 点 No.2	12.300	53.200	12.300	53.200	12.300	53.200	11.500	51.100
			点 No.9 <(S1)< 点 No.10	15.000	57.600	15.000	57.600	15.000	57.600	18.100	57.100
	SCL2 (S1) (S2) (D)	SM750 = ON	点 No.1 <(S1)< 点 No.2	14.200	53.300	14.200	53.300	14.200	53.300	13.200	51.200
			点 No.9 <(S1)< 点 No.10	14.900	55.000	14.900	55.000	14.900	55.000	18.000	54.500
		SM750 = OFF	点 No.1 <(S1)< 点 No.2	15.000	53.500	15.000	53.500	15.000	53.500	14.000	51.300
			点 No.9 <(S1)< 点 No.10	16.300	56.400	16.300	56.400	16.300	56.400	19.300	55.800
	DSCL2 (S1) (S2) (D)	SM750 = ON	点 No.1 <(S1)< 点 No.2	13.400	52.700	13.400	52.700	13.400	52.700	13.100	50.500
			点 No.9 <(S1)< 点 No.10	14.200	54.300	14.200	54.300	14.200	54.300	18.100	53.700
		SM750 = OFF	点 No.1 <(S1)< 点 No.2	12.300	53.200	12.300	53.200	12.300	53.200	12.100	51.000
			点 No.9 <(S1)< 点 No.10	15.000	57.600	15.000	57.600	15.000	57.600	18.900	57.100

分类	指令	条件（软元件）	处理时间（μs）							
			Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
			最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
应用指令	RSET	标准 RAM	6.800	26.900	6.800	26.900	6.800	26.900	3.000	16.400
		SRAM	-	-	-	-	-	-	3.000	16.400
	QDRSET	SRAM →标准 RAM	-	-	-	-	-	-	230.000	327.000
		标准 RAM → SRAM	-	-	-	-	-	-	997.000	1066.000
	QCDSET	SRAM →标准 ROM	-	-	-	-	-	-	525.000	690.000
		标准 ROM → SRAM	-	-	-	-	-	-	490.000	655.000
	DATERD	-	5.600	27.800	5.600	27.800	5.600	27.800	5.100	14.700
	DATEWR	-	7.800	42.100	7.800	42.100	7.800	42.100	7.100	23.000
	DATE +	无进位	14.200	41.200	14.200	41.200	14.200	41.200	6.500	13.100
		有进位	14.200	41.200	14.200	41.200	14.200	41.200	5.700	21.200
	DATE-	无进位	15.100	41.200	15.100	41.200	15.100	41.200	6.500	11.500
		有进位	15.100	41.200	15.100	41.200	15.100	41.200	5.700	17.200
	SECOND	-	5.800	20.500	5.800	20.500	5.800	20.500	2.600	5.900
	HOUR	-	6.200	22.500	6.200	22.500	6.200	22.500	3.000	5.300

分类	指令	条件（软元件）		处理时间（μs）							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
应用指令	LDDT =	与指定的日期比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	8.200	25.500
		与当前的日期比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
	ANDDT =	未执行时		0.480		0.320		0.240		0.160	
		与指定的日期比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
		与当前的日期比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
	ORDT =	未执行时		0.480		0.320		0.240		0.160	
		与指定的日期比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
		与当前的日期比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
	LDDT< >	与指定的日期比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
		与当前的日期比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
	ANDDT< >	未执行时		0.480		0.320		0.240		0.160	
		与指定的日期比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
		与当前的日期比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
	ORDT< >	未执行时		0.480		0.320		0.240		0.160	
		与指定的日期比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
		与当前的日期比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
	LDDT >	与指定的日期比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
		与当前的日期比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
	ANDDT >	未执行时		0.480		0.320		0.240		0.160	
		与指定的日期比较	导通时	8.200	25.500	8.200	25.500	8.200	25.500	7.200	23.400
			非导通时	8.200	25.500	8.200	25.500	8.200	25.500	7.200	23.400
		与当前的日期比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
	ORDT >	未执行时		0.480		0.320		0.240		0.160	
		与指定的日期比较	导通时	8.200	25.500	8.200	25.500	8.200	25.500	7.400	23.300
			非导通时	8.200	25.500	8.200	25.500	8.200	25.500	7.400	23.300
		与当前的日期比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000

分类	指令	条件（软元件）		处理时间（μs）							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
应用指令	LDDT< =	与指定的日期比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
		与当前的日期比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
	ANDDT< =	未执行时		0.480		0.320		0.240		0.160	
		与指定的日期比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
		与当前的日期比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
	ORDT< =	未执行时		0.480		0.320		0.240		0.160	
		与指定的日期比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
		与当前的日期比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
	LDDT<	与指定的日期比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
		与当前的日期比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
	ANDDT<	未执行时		0.480		0.320		0.240		0.160	
		与指定的日期比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
		与当前的日期比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
	ORDT<	未执行时		0.480		0.320		0.240		0.160	
		与指定的日期比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
		与当前的日期比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
	LDDT > =	与指定的日期比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
		与当前的日期比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
	ANDDT > =	未执行时		0.480		0.320		0.240		0.160	
		与指定的日期比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
		与当前的日期比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
	ORDT > =	未执行时		0.480		0.320		0.240		0.160	
		与指定的日期比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
		与当前的日期比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
	LDTM =	与指定的日期比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
		与当前的日期比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100

分类	指令	条件（软元件）		处理时间（μs）							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
应用指令	ANDTM =	未执行时		0.480		0.320		0.240		0.160	
		与指定的时间比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
		与当前的时间比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
	ORTM =	未执行时		0.480		0.320		0.240		0.160	
		与指定的时间比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
		与当前的时间比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
	LDTM< >	与指定的时间比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
		与当前的时间比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
	ANDTM< >	未执行时		0.480		0.320		0.240		0.160	
		与指定的时间比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
		与当前的时间比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
	ORTM< >	未执行时		0.480		0.320		0.240		0.160	
		与指定的时间比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
		与当前的时间比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
	LDTM >	与指定的时间比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
		与当前的时间比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
	ANDTM >	未执行时		0.480		0.320		0.240		0.160	
		与指定的时间比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
		与当前的时间比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
	ORTM >	未执行时		0.480		0.320		0.240		0.160	
		与指定的时间比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
		与当前的时间比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000

分类	指令	条件（软元件）		处理时间（μs）							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
应用指令	LDTM< =	与指定的时间比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
		与当前的时间比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
	ANDTM< =	未执行时		0.480		0.320		0.240		0.160	
		与指定的时间比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
		与当前的时间比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
	ORTM< =	未执行时		0.480		0.320		0.240		0.160	
		与指定的时间比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
		与当前的时间比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
	LDTM<	与指定的时间比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
		与当前的时间比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
	ANDTM<	未执行时		0.480		0.320		0.240		0.160	
		与指定的时间比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
		与当前的时间比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
	ORTM<	未执行时		0.480		0.320		0.240		0.160	
		与指定的时间比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
		与当前的时间比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
	LDTM > =	与指定的时间比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
		与当前的时间比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
	ANDTM > =	未执行时		0.480		0.320		0.240		0.160	
		与指定的时间比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
		与当前的时间比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
	ORTM > =	未执行时		0.480		0.320		0.240		0.160	
		与指定的时间比较	导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
			非导通时	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
		与当前的时间比较	导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
			非导通时	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100

分类	指令	条件（软元件）		处理时间（μs）							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
应用指令	S.DATERD	-		9.250	51.000	9.250	51.000	9.250	51.000	7.500	23.400
	S.DATE+	无进位		16.800	75.400	16.800	75.400	16.800	75.400	9.100	23.400
		有进位		16.800	75.400	16.800	75.400	16.800	75.400	8.900	22.200
	S.DATE-	无进位		17.600	75.300	17.600	75.300	17.600	75.300	9.000	22.200
		有进位		16.900	75.300	16.900	75.300	16.900	75.300	9.800	22.100
	PSTOP	-		82.200	199.000	82.200	199.000	82.200	199.000	61.400	84.500
	POFF	-		82.600	198.000	82.600	198.000	82.600	198.000	121.000	246.000
	PSCAN	-		83.600	200.000	83.600	200.000	83.600	200.000	126.000	232.000
	WDT	-		2.900	12.000	2.900	12.000	2.900	12.000	1.300	3.000
	DUTY	-		7.700	27.500	7.700	27.500	7.700	27.500	4.900	24.300
	TIMCHK	-		5.350	24.500	5.350	24.500	5.350	24.500	7.400	23.300
	ZRRDB	标准 RAM 的文件寄存器		4.100	4.200	4.100	4.200	4.100	4.200	2.400	2.600
		SRAM 的文件寄存器		-	-	-	-	-	-	2.500	2.800
	ZRWRB	标准 RAM 的文件寄存器		5.400	5.500	5.400	5.500	5.400	5.500	3.100	3.300
		SRAM 的文件寄存器		-	-	-	-	-	-	3.300	3.600
	ADRSET	-		2.400	6.650	2.400	6.650	2.400	6.650	4.200	4.900
	ZPUSH	-		9.200	20.500	9.200	20.500	9.200	20.500	6.900	14.000
	ZPOP	-		9.000	15.500	9.000	15.500	9.000	15.500	7.500	12.500
	S.ZCOM	安装 CC-Link 模块时（主站侧）		29.400	91.700	29.400	91.700	29.400	91.700	20.600	55.000
		安装 CC-Link 模块时（本地站侧）		29.500	91.600	29.500	91.600	29.500	91.600	20.600	66.100
		安装 MELSECNET/H、CC-Link IE 控制网络模块时（管理站侧）		79.900	214.000	79.900	214.000	79.900	214.000	102.000	180.000
		安装 MELSECNET/H、CC-Link IE 控制网络模块时（普通站侧）		79.900	214.000	79.900	214.000	79.900	214.000	29.800	102.000
	S.RTREAD	-		9.200	57.700	9.200	57.700	9.200	57.700	6.700	33.500
	S.RTWRITE	-		10.900	67.100	10.900	67.100	10.900	67.100	8.300	26.000
	UNIRD n1 Ⓓ n2	n2 = 1		6.000	33.100	6.000	33.100	6.000	33.100	4.000	29.100
		n2 = 16		16.500	43.600	16.500	43.600	16.500	43.600	12.500	37.600
	TRACE	开始		174.000	174.000	174.000	174.000	174.000	174.000	96.600	103.000
	TRACER	-		5.100	15.500	5.100	15.500	5.100	15.500	3.800	13.600
	RBMOV Ⓢ Ⓓ n	使用标准 RAM	1 点	-	-	12.600	34.800	12.600	34.800	10.100	32.200
			1000 点	-	-	233.000	255.000	233.000	255.000	231.000	253.000
		使用 SRAM	1 点	-	-	-	-	-	-	9.200	12.100
			1000 点	-	-	-	-	-	-	489.000	544.000
	SP.FWRITE	-		-	-	-	-	-	-	87.000	144.000
	SP.FREAD	-		-	-	-	-	-	-	127.000	140.000
	SP.DEVST	-		125.000	125.000	125.000	125.000	125.000	125.000	8.100	98.000
	S.DEVLD	-		18.300	36.700	18.300	36.700	18.300	36.700	13.000	43.000

分类	指令	条件（软元件）		处理时间（μs）							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
多CPU专用指令	S.TO n1 n2 n3 n4 (D)	至本机CPU共享存储器的写入	n4 = 1	64.600	78.100	64.600	78.100	64.600	78.100	64.600	78.100
			n4 = 320	115.000	126.000	115.000	126.000	115.000	126.000	154.000	126.000
	TO n1 n2 (S) n3	至本机CPU共享存储器的写入	n3 = 1	13.900	62.700	13.900	62.700	13.900	62.700	6.800	18.600
			n3 = 320	63.500	113.000	63.500	113.000	63.500	113.000	38.400	53.000
	DTO n1 n2 (S) n3	至本机CPU共享存储器的写入	n3 = 1	13.600	61.400	13.600	61.400	13.600	61.400	7.000	20.600
			n3 = 320	113.000	161.000	113.000	161.000	113.000	161.000	65.200	81.000
	FROM n1 n2 (D) n3	本机CPU共享存储器读取	n3 = 1	13.000	58.500	13.000	58.500	13.000	58.500	7.300	23.400
			n3 = 320	56.100	102.000	56.100	102.000	56.100	102.000	51.300	71.700
		其它机号CPU共享存储器读取	n3 = 1	24.400	82.900	24.400	82.900	24.400	82.900	16.600	37.000
			n3 = 320	152.000	243.000	152.000	243.000	152.000	243.000	153.000	185.000
			n3 = 1000	418.000	518.000	418.000	518.000	418.000	518.000	432.000	485.000
	DFRO n1 n2 (D) n3	本机CPU共享存储器读取	n3 = 1	10.700	57.200	10.700	57.200	10.700	57.200	7.400	24.800
			n3 = 320	97.300	144.000	97.300	144.000	97.300	144.000	98.700	111.000
		其它机号CPU共享存储器读取	n3 = 1	24.800	94.200	24.800	94.200	24.800	94.200	16.600	47.300
			n3 = 320	276.000	367.000	276.000	367.000	276.000	367.000	278.000	339.000
			n3 = 1000	799.000	892.000	799.000	892.000	799.000	892.000	841.000	892.000

备注

对于表中未记述上升沿执行指令（□P）的指令，其处理时间与ON时执行的指令的处理时间相同。

例 WORDP指令、TOP指令等。

(1) 子集指令以外的指令处理时间一览表

分类	指令	条件（软元件）	处理时间（μs）					
			Q03UD(E)CPU		Q04UD(E)HCPU、Q06UD(E)HCPU		Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU	
			最小值	最大值	最小值	最大值	最小值	最大值
顺控程序指令	ANB ORB MPS MRD MPP	-	0.020		0.0095		0.0095	
	INV	未执行时	0.020		0.0095		0.0095	
		执行时						
	MEP MEF	未执行时	0.020		0.0095		0.0095	
		执行时						
	EGP EGF	未执行时	0.020		0.0095		0.0095	
		执行时						
	PLS	-	1.300	1.600	0.890	1.100	0.890	1.100
	PLF	-	1.500	1.600	0.940	1.200	0.940	1.200
	FF	未执行时	0.040		0.0185		0.0185	
		执行时	1.200	1.500	0.790	0.910	0.790	0.910
	DELTA	未执行时	0.040		0.0185		0.0185	
		执行时	2.800	3.600	2.400	3.200	2.400	3.200
	SFT	未执行时	0.040		0.0185		0.0185	
		执行时	1.600	3.300	1.100	2.700	1.100	2.700
	MC	-	0.040		0.0185		0.0185	
	MCR	-	0.040		0.0185		0.0185	
	FEND END	进行出错检查	108.000	130.000	75.800	89.300	75.800	89.300
		不进行出错检查	107.000	124.000	75.800	89.800	75.800	89.800

分类	指令	条件（软元件）			处理时间（μs）					
					Q03UD(E)CPU		Q04UD(E)HCPU、Q06UD(E)HCPU		Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU	
					最小值	最大值	最小值	最大值	最小值	最大值
顺控程序指令	NOP NOPLF PAGE	-			0.020		0.0095		0.0095	
基本指令	LDE =	单精度	导通时		3.700	4.700	3.300	4.300	3.300	4.300
			非导通时		3.800	5.000	3.400	4.500	3.400	4.500
	ANDE =	单精度	未执行时		0.060		0.0285		0.0285	
			执行时	导通时	3.300	5.800	3.000	5.100	3.000	5.100
				非导通时	3.500	5.600	3.000	5.200	3.000	5.200
	ORE =	单精度	未执行时		0.060		0.0285		0.0285	
			执行时	导通时	3.600	4.500	3.200	4.200		
				非导通时	3.500	4.800	3.200	4.300		
	LDE < >	单精度	导通时		4.000	4.700	3.600	4.200	0.0285	
			非导通时		3.900	4.500	3.500	4.000		
	ANDE < >	单精度	未执行时		0.060		0.0285		0.0285	
			执行时	导通时	3.300	5.100	3.000	4.800		
				非导通时	3.500	5.000	3.100	4.600		
	ORE < >	单精度	未执行时		0.060		0.0285		0.0285	
			执行时	导通时	3.600	6.000	3.300	5.500		
				非导通时	3.500	5.800	3.100	5.300		
	LDE >	单精度	导通时		3.800	5.000	3.300	4.600	0.0285	
			非导通时		3.700	4.900	3.300	4.400		
	ANDE >	单精度	未执行时		0.060		0.0285		0.0285	
			执行时	导通时	3.500	4.700	3.100	4.200		
				非导通时	3.600	4.500	3.100	4.000		
	ORE >	单精度	未执行时		0.060		0.0285		0.0285	
			执行时	导通时	3.600	5.100	3.300	4.600		
				非导通时	3.500	4.800	3.200	4.500		
	LDE< =	单精度	导通时		3.800	5.600	3.400	5.200	0.0285	
			非导通时		3.800	5.600	3.400	5.100		
	ANDE< =	单精度	未执行时		0.060		0.0285		0.0285	
			执行时	导通时	3.200	4.600	2.800	4.200		
				非导通时	3.500	5.000	3.100	4.500		
	ORE< =	单精度	未执行时		0.060		0.0285		0.0285	
			执行时	导通时	3.700	5.800	3.400	5.400		
				非导通时	3.800	5.700	3.300	5.300		
	LDE<	单精度	导通时		4.000	5.400	3.500	4.900	0.0285	
			非导通时		4.000	5.200	3.500	4.900		
	ANDE<	单精度	未执行时		0.060		0.0285		0.0285	
			执行时	导通时	3.400	4.600	3.000	4.200		
				非导通时	3.500	4.900	3.100	4.400		

分类	指令	条件（软元件）			处理时间（μs）					
					Q03UD(E)CPU		Q04UD(E)HCPU、Q06UD(E)HCPU		Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU	
					最小值	最大值	最小值	最大值	最小值	最大值
基本指令	ORE<	单精度	未执行时		0.060		0.0285		0.0285	
			执行时	导通时	3.600	5.200	3.300	4.900		
				非导通时	3.400	4.900	3.200	4.500		
	LDE > =	单精度	导通时		3.800	6.000	3.300	5.500	0.0285	
			非导通时		3.800	5.900	3.400	5.400		
	ANDE > =	单精度	未执行时		0.060		0.0285		0.0285	
			执行时	导通时	3.200	4.800	2.900	4.600		
				非导通时	3.500	5.400	3.100	5.100		
	ORE > =	单精度	未执行时		0.060		0.0285		0.0285	
			执行时	导通时	3.600	5.200	3.300	4.700		
				非导通时	3.500	5.200	3.200	4.700		
	LDED =	双精度	导通时		4.100	7.700	3.500	7.200	3.500	7.200
			非导通时		4.300	8.100	3.800	7.400	3.800	7.400
	ANDED =	双精度	未执行时		0.060		0.0285		0.0285	
			执行时	导通时	3.600	7.600	3.200	7.000	3.200	7.000
				非导通时	3.900	7.700	3.400	7.400	3.400	7.400
	ORED =	双精度	未执行时		0.060		0.0285		0.0285	
			执行时	导通时	3.800	8.800	3.400	8.300	3.400	8.300
				非导通时	4.000	9.300	3.700	8.800	3.700	8.800
	LDED < >	双精度	导通时		4.400	8.200	3.900	7.700	3.900	7.700
			非导通时		4.100	7.900	3.500	7.500	3.500	7.500
	ANDED < >	双精度	未执行时		0.060		0.0285		0.0285	
			执行时	导通时	3.800	7.600	3.300	7.200	3.300	7.200
				非导通时	3.800	7.700	3.400	7.300	3.400	7.300
	ORED < >	双精度	未执行时		0.060		0.0285		0.0285	
			执行时	导通时	4.100	9.300	3.700	8.900	3.700	8.900
				非导通时	3.800	8.900	3.400	8.400	3.400	8.400
	LDED >	双精度	导通时		4.300	8.100	3.800	7.500	3.800	7.500
			非导通时		4.100	7.800	3.500	7.200	3.500	7.200
	ANDED >	双精度	未执行时		0.060		0.0285		0.0285	
			执行时	导通时	3.800	7.700	3.300	7.300	3.300	7.300
				非导通时	4.000	7.900	3.500	7.500	3.500	7.500
	ORED >	双精度	未执行时		0.060		0.0285		0.0285	
			执行时	导通时	4.100	9.300	3.700	8.800	3.700	8.800
				非导通时	4.100	9.300	3.700	8.800	3.700	8.800
	LDED< =	双精度	导通时		4.000	8.000	3.500	7.400	3.500	7.400
			非导通时		4.100	9.400	3.600	8.800	3.600	8.800
	ANDED< =	双精度	未执行时		0.060		0.0285		0.0285	
			执行时	导通时	3.800	7.700	3.300	7.200	3.300	7.200
				非导通时	3.900	7.700	3.500	7.400	3.500	7.400
	ORED< =	双精度	未执行时		0.060		0.0285		0.0285	
			执行时	导通时	4.100	9.600	3.700	9.200	3.700	9.200
				非导通时	4.100	9.600	3.700	9.200	3.700	9.200

分类	指令	条件（软元件）			处理时间（μs）					
					Q03UD(E)CPU		Q04UD(E)HCPU、Q06UD(E)HCPU		Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU	
					最小值	最大值	最小值	最大值	最小值	最大值
基本指令	LDED<	双精度	导通时		4.300	8.300	3.800	7.600	3.800	7.600
			非导通时		3.700	7.900	3.500	7.400	3.500	7.400
	ANDED<	双精度	未执行时		0.060		0.0285		0.0285	
			执行时	导通时	3.800	7.800	3.300	7.300	3.300	7.300
				非导通时	3.900	7.900	3.400	3.900	3.400	3.900
	ORED<	双精度	未执行时		0.060		0.0285		0.0285	
			执行时	导通时	4.100	9.600	3.700	9.200	3.700	9.200
				非导通时	4.000	9.600	3.700	9.200	3.700	9.200
	LDED > =	双精度	导通时		4.100	9.600	3.600	9.000	3.600	9.000
			非导通时		4.100	9.600	3.600	8.900	3.600	8.900
	ANDED > =	双精度	未执行时		0.060		0.0285		0.0285	
			执行时	导通时	3.800	7.900	3.400	7.400	3.400	7.400
				非导通时	3.900	8.100	3.400	7.500	3.400	7.500
	ORED > =	双精度	未执行时		0.060		0.0285		0.0285	
			执行时	导通时	4.100	9.600	3.700	9.200	3.700	9.200
				非导通时	4.000	7.200	3.600	6.600	3.600	6.600
	LD$ =	导通时			5.300	8.900	4.700	8.100	4.700	8.100
		非导通时			4.700	9.000	4.200	8.200	4.200	8.200
	AND$ =	未执行时			0.060		0.0285		0.0285	
		执行时	导通时		4.400	6.800	3.900	6.400	3.900	6.400
			非导通时		4.500	6.700	4.000	6.300	4.000	6.300
	OR$ =	未执行时			0.060		0.0285		0.0285	
		执行时	导通时		5.100	8.200	4.200	7.600	4.200	7.600
			非导通时		5.000	8.100	4.000	7.200	4.000	7.200
	LD$ < >	导通时			4.800	8.100	4.300	7.500	4.300	7.500
		非导通时			4.700	8.400	4.200	7.800	4.200	7.800
	AND$ < >	未执行时			0.060		0.0285		0.0285	
		执行时	导通时		4.300	5.500	4.100	5.100	4.100	5.100
			非导通时		4.500	5.900	4.400	5.400	4.400	5.400
	OR$ < >	未执行时			0.060		0.0285		0.0285	
		执行时	导通时		5.200	7.300	4.100	6.700	4.100	6.700
			非导通时		5.100	7.200	4.100	6.700	4.100	6.700
	LD$ >	导通时			4.800	7.200	4.300	6.700	4.300	6.700
		非导通时			4.800	7.700	4.200	7.100	4.200	7.100
	AND$ >	未执行时			0.060		0.0285		0.0285	
		执行时	导通时		4.500	7.100	4.000	6.700	4.000	6.700
			非导通时		4.600	7.600	4.300	7.000	4.300	7.000
	OR$ >	未执行时			0.060		0.0285		0.0285	
		执行时	导通时		5.100	6.800	4.300	6.200	4.300	6.200
			非导通时		5.200	7.200	4.300	6.600	4.300	6.600
	LD$< =	导通时			5.000	6.300	4.400	5.700	4.400	5.700
		非导通时			4.800	6.400	4.200	5.800	4.200	5.800
	AND$< =	未执行时			0.060		0.0285		0.0285	
		执行时	导通时		4.600	7.600	4.100	7.200	4.100	7.200
			非导通时		4.700	7.700	4.200	7.300	4.200	7.300

分类	指令	条件（软元件）		处理时间（μs）					
				Q03UD(E)CPU		Q04UD(E)HCPU、Q06UD(E)HCPU		Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU	
				最小值	最大值	最小值	最大值	最小值	最大值
基本指令	OR$< =	未执行时		0.060		0.0285		0.0285	
		执行时	导通时	4.700	7.700	4.400	7.200	4.400	7.200
			非导通时	4.600	7.600	4.400	7.100	4.400	7.100
	LD$<	导通时		4.800	8.100	4.500	7.500	4.500	7.500
		非导通时		5.000	8.300	4.500	7.900	4.500	7.900
	AND$<	未执行时		0.060		0.0285		0.0285	
		执行时	导通时	4.500	7.100	4.000	6.600	4.000	6.600
			非导通时	4.900	7.500	4.400	7.100	4.400	7.100
	OR$<	未执行时		0.060		0.0285		0.0285	
		执行时	导通时	5.100	7.800	4.100	7.200	4.100	7.200
			非导通时	5.000	8.100	4.100	7.600	4.100	7.600
	LD$ > =	导通时		4.800	6.700	4.500	6.200	4.500	6.200
		非导通时		5.000	6.700	4.400	6.300	4.400	6.300
	AND$ > =	未执行时		0.060		0.0285		0.0285	
		执行时	导通时	4.400	6.800	4.100	6.300	4.100	6.300
			非导通时	4.500	7.000	4.200	6.600	4.200	6.600
	OR$ > =	未执行时		0.060		0.0285		0.0285	
		执行时	导通时	5.400	6.600	4.100	5.800	4.100	5.800
			非导通时	5.300	6.300	4.100	5.700	4.100	5.700
	BKCMP = (S1) (S2) (D) n	n = 1		8.200	10.700	7.500	10.000	7.500	10.000
		n = 96		57.400	61.800	46.400	48.700	46.400	48.700
	BKCMP < > (S1) (S2) (D) n	n = 1		8.200	10.700	7.500	10.000	7.500	10.000
		n = 96		59.500	63.300	45.600	50.400	45.600	50.400
	BKCMP > (S1) (S2) (D) n	n = 1		8.200	10.800	7.500	10.100	7.500	10.100
		n = 96		59.500	63.400	47.700	50.500	47.700	50.500
	BKCMP< = (S1) (S2) (D) n	n = 1		8.200	10.600	7.500	10.000	7.500	10.000
		n = 96		57.400	61.700	46.400	49.000	46.400	49.000
	BKCMP< (S1) (S2) (D) n	n = 1		8.300	10.600	7.500	10.000	7.500	10.000
		n = 96		59.500	63.600	47.600	50.500	47.600	50.500
	BKCMP > = (S1) (S2) (D) n	n = 1		8.200	10.900	7.500	10.000	7.500	10.000
		n = 96		57.400	62.000	46.400	48.900	46.400	48.900
	DBKCMP = (S1) (S2) (D) n	n = 1		9.250	14.000	8.600	13.000	8.600	13.000
		n = 96		60.700	67.500	47.900	52.800	47.900	52.800
	DBKCMP < > (S1) (S2) (D) n	n = 1		9.250	14.000	8.600	13.000	8.600	13.000
		n = 96		60.700	67.500	47.900	52.800	47.900	52.800
	DBKCMP > (S1) (S2) (D) n	n = 1		9.250	14.000	8.600	13.000	8.600	13.000
		n = 96		60.700	67.500	47.900	52.800	47.900	52.800
	DBKCMP< = (S1) (S2) (D) n	n = 1		9.250	14.000	8.600	13.000	8.600	13.000
		n = 96		60.700	67.500	47.900	52.800	47.900	52.800
	DBKCMP< (S1) (S2) (D) n	n = 1		9.250	14.000	8.600	13.000	8.600	13.000
		n = 96		60.700	67.500	47.900	52.800	47.900	52.800
	DBKCMP > = (S1) (S2) (D) n	n = 1		9.250	14.000	8.600	13.000	8.600	13.000
		n = 96		60.700	67.500	47.900	52.800	47.900	52.800
	DB + (S) (D)	执行时		4.900	7.000	4.600	6.400	4.600	6.400
	DB + (S1) (S2) (D)	执行时		5.200	7.300	4.800	6.700	4.800	6.700
	DB - (S) (D)	执行时		4.900	6.600	4.700	6.000	4.700	6.000
	DB - (S1) (S2) (D)	执行时		5.200	7.500	4.800	6.600	4.800	6.600
	DB * (S1) (S2) (D)	执行时		8.300	12.100	8.100	11.600	8.100	11.600
	DB / (S1) (S2) (D)	执行时		6.100	9.100	5.800	8.800	5.800	8.800

分类	指令	条件（软元件）		处理时间（μs）					
				Q03UD(E)CPU		Q04UD(E)HCPU、Q06UD(E)HCPU		Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU	
				最小值	最大值	最小值	最大值	最小值	最大值
基本指令	ED + (S) (D)	双精度	(S) = 0, (D) = 0	4.800	8.000	4.300	7.200	4.300	7.200
			(S) = 2^{1023}, (D) = 2^{1023}	4.800	8.000	4.300	7.200	4.300	7.200
	ED + (S1) (S2) (D)	双精度	(S1) = 0, (S2) = 0	5.500	9.800	4.800	9.200	4.800	9.200
			(S1) = 2^{1023}, (S2) = 2^{1023}	5.500	9.800	4.800	9.200	4.800	9.200
	ED - (S) (D)	双精度	(S) = 0, (D) = 0	5.000	8.200	4.400	7.500	4.400	7.500
			(S) = 2^{1023}, (D) = 2^{1023}	5.000	8.200	4.400	7.500	4.400	7.500
	ED - (S1) (S2) (D)	双精度	(S1) = 0, (S2) = 0	4.400	8.100	3.800	7.500	3.800	7.500
			(S1) = 2^{1023}, (S2) = 2^{1023}	4.400	8.100	3.800	7.500	3.800	7.500
	ED * (S1) (S2) (D)	双精度	(S1) = 0, (S2) = 0	5.800	9.500	5.100	8.800	5.100	8.800
			(S1) = 2^{1023}, (S2) = 2^{1023}	5.800	9.500	5.100	8.800	5.100	8.800
	ED / (S1) (S2) (D)	双精度	(S1) = 2^{1023}, (S2) = 2^{1023}	6.600	10.600	5.900	10.000	5.900	10.000
	BK + (S1) (S2) (D) n	n = 1		9.100	11.200	8.500	10.600	8.500	10.600
		n = 96		60.700	62.900	44.600	47.000	44.600	47.000
	BK - (S1) (S2) (D) n	n = 1		9.700	12.000	8.900	11.300	8.900	11.300
		n = 96		61.300	63.600	45.600	47.900	45.600	47.900
	DBK + (S1) (S2) (D) n	n = 1		7.000	10.700	6.450	9.950	6.450	9.950
		n = 96		59.400	63.100	43.700	47.500	43.700	47.500
	DBK - (S1) (S2) (D) n	n = 1		7.000	10.700	6.450	9.950	6.450	9.950
		n = 96		59.400	63.100	43.700	47.500	43.700	47.500
	$+ (S) (D)	-		8.800	14.600	8.100	13.900	8.100	13.900
	$+ (S1) (S2) (D)	-		7.300	11.100	6.500	10.300	6.500	10.300
	FLTD	双精度	(S) = 0	2.300	5.000	1.800	4.700	1.800	4.700
			(S) = 7FFFH	2.500	5.200	2.200	4.800	2.200	4.800
	DFLTD	双精度	(S) = 0	2.400	5.200	2.000	4.900	2.000	4.900
			(S) = 7FFFFFFFH	2.700	5.400	2.300	5.100	2.300	5.100
	INTD	双精度	(S) = 0	2.700	4.100	2.200	4.100	2.200	4.100
			(S) = 32766.5	3.700	5.900	3.200	5.600	3.200	5.600
	DINTD	双精度	(S) = 0	2.600	3.900	2.200	3.400	2.200	3.400
			(S) = 1234567890.3	3.400	5.600	3.000	5.100	3.000	5.100

分类	指令	条件（软元件）	处理时间（μs）					
			Q03UD(E)CPU		Q04UD(E)HCPU、Q06UD(E)HCPU		Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU	
			最小值	最大值	最小值	最大值	最小值	最大值
基本指令	DBL	执行时	2.700	3.400	2.300	2.700	2.300	2.700
	WORD	执行时	2.900	4.300	2.600	3.600	2.600	3.600
	GRY	执行时	2.700	3.900	2.300	3.400	2.300	3.400
	DGRY	执行时	2.900	3.500	2.500	3.000	2.500	3.000
	GBIN	执行时	4.000	4.800	3.800	4.300	3.800	4.300
	DGBIN	执行时	5.500	6.100	5.000	5.900	5.000	5.900
	NEG	执行时	2.400	3.900	2.000	3.300	2.000	3.300
	DNEG	执行时	2.500	3.700	2.500	3.300	2.500	3.300
	ENEG	浮点 = 0	2.500	3.300	2.300	2.800	2.300	2.800
		浮点 = -1.0	2.700	4.500	2.500	3.900	2.500	3.900
	EDNEG	浮点 = 0	2.200	3.500	1.800	3.100	1.800	3.100
		浮点 = °1.0	2.400	3.500	1.900	3.000	1.900	3.000
	BKBCD (S) (D) n	n = 1	6.600	8.900	5.900	8.200	5.900	8.200
		n = 96	71.300	74.100	61.000	63.400	61.000	63.400
	BKBIN (S) (D) n	n = 1	6.500	9.800	5.600	9.300	5.600	9.300
		n = 96	56.300	59.500	49.200	52.500	49.200	52.500
	ECON	-	2.600	5.400	2.100	4.500	2.100	4.500
	EDCON	-	2.800	5.400	2.500	5.400	2.500	5.400
	EDMOV	-	2.300	5.500	1.700	5.000	1.700	5.000
	$MOV	传送字符串 = 0	4.000	6.300	3.400	5.600	3.400	5.600
		传送字符串 = 32	14.600	16.500	11.400	13.300	11.400	13.300
	BXCH (D1) (D2) n	n = 1	6.200	7.900	5.500	7.300	5.500	7.300
		n = 96	67.000	68.800	47.300	49.300	47.300	49.300
	SWAP	-	2.400	2.700	1.900	2.200	1.900	2.200
	GOEND	-	0.500		0.500		0.500	
	DI	-	1.800	2.200	1.500	1.800	1.500	1.800
	EI	-	3.100	3.800	3.000	3.300	3.000	3.300
	IMASK	-	9.800	13.300	7.200	10.500	7.200	10.500
	IRET	-	1.000		1.000		1.000	
	RSF X n	n = 1	4.200	5.900	3.700	5.600	3.700	5.600
		n = 96	11.400	13.800	10.700	12.400	10.700	12.400
	RSF Y n	n = 1	3.800	4.800	3.400	4.800	3.400	4.800
		n = 96	8.500	9.500	8.100	8.900	8.100	8.900
	UDCNT1	-	0.900	1.500	0.500	0.983	0.500	0.983
	UDCNT2	-	0.900	1.700	0.600	1.300	0.600	1.300
	TTMR	-	3.900	6.100	3.400	5.400	3.400	5.400
	STMR	-	6.800	13.500	5.800	12.500	5.800	12.500
	ROTC	-	9.000	10.500	8.000	9.400	8.000	9.400
	RAMP	-	5.900	8.800	5.200	8.400	5.200	8.400
	SPD	-	0.900	1.900	0.500	1.400	0.500	1.400
	PLSY	-	1.900	2.200	1.500	1.800	1.500	1.800
	PWM	-	1.200	1.600	0.900	1.200	0.900	1.200
	MTR	-	10.400	19.800	9.400	10.000	9.400	10.000

分类	指令	条件（软元件）		处理时间（μs）					
				Q03UD(E)CPU		Q04UD(E)HCPU、Q06UD(E)HCPU		Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU	
				最小值	最大值	最小值	最大值	最小值	最大值
应用指令	BKAND (S1) (S2) (D) n	n = 1		9.000	11.700	8.300	11.000	8.300	11.000
		n = 96		57.400	63.100	43.800	47.300	43.800	47.300
	BKOR (S1) (S2) (D) n	n = 1		7.700	10.000	7.700	9.500	7.700	9.500
		n = 96		57.400	61.900	44.300	45.800	44.300	45.800
	BKXOR (S1) (S2) (D) n	n = 1		7.800	10.100	7.300	9.200	7.300	9.200
		n = 96		57.300	61.500	43.800	45.800	43.800	45.800
	BKXNR (S1) (S2) (D) n	n = 1		7.800	9.600	7.600	8.900	7.600	8.900
		n = 96		57.400	61.400	43.900	45.300	43.900	45.300
	BSFR (D) n	n = 1		3.700	5.400	3.200	4.800	3.200	4.800
		n = 96		6.900	9.000	5.800	7.700	5.800	7.700
	BSFL (D) n	n = 1		4.100	5.900	3.400	5.100	3.400	5.100
		n = 96		7.100	9.100	6.000	7.900	6.000	7.900
	SFTBR (D) n1 n2	进行了移位的位的数=16/移位数 = 1		7.950	17.500	7.600	16.900	7.600	16.900
		进行了移位的位的数=16/移位数 = 15		7.950	17.500	7.550	16.900	7.550	16.900
	SFTBL (D) n1 n2	进行了移位的位的数=16/移位数 = 1		7.950	17.900	7.500	17.400	7.500	17.400
		进行了移位的位的数=16/移位数 = 15		7.900	17.800	7.500	17.300	7.500	17.300
	SFTWR (D) n1 n2	进行了移位的位的数=16/移位数 = 1		5.950	10.600	4.600	8.700	4.600	8.700
		进行了移位的位的数=16/移位数 = 15		5.900	10.600	4.600	8.700	4.600	8.700
	SFTWL (D) n1 n2	进行了移位的位的数=16/移位数 = 1		5.950	10.700	4.550	8.700	4.550	8.700
		进行了移位的位的数=16/移位数 = 15		5.950	10.700	4.600	8.800	4.600	8.800
	BSET (D) n	n = 1		3.000	3.400	2.500	2.800	2.500	2.800
		n = 15		3.000	3.500	2.500	2.800	2.500	2.800
	BRST (D) n	n = 1		3.000	3.400	2.600	2.800	2.600	2.800
		n = 15		3.000	3.400	2.500	2.800	2.500	2.800
	TEST	执行时		4.400	5.300	3.700	4.700	3.700	4.700
	DTEST	执行时		4.500	5.400	3.900	4.800	3.900	4.800
	BKRST (S) n	n = 1		4.300	4.600	3.700	4.100	3.700	4.100
		n = 96		6.000	6.800	5.100	6.000	5.100	6.000
	SER (S1) (S2) (D) n	n = 1	全部一致	4.900	5.300	4.200	4.600	4.200	4.600
			全部不一致	5.000	5.300	4.200	4.600	4.200	4.600
		n = 96	全部一致	32.300	32.900	25.900	26.300	25.900	26.300
			全部不一致	32.400	32.900	25.900	26.300	25.900	26.300
	DSER (S1) (S2) (D) n	n = 1	全部一致	6.100	6.500	5.400	5.700	5.400	5.700
			全部不一致	6.200	6.600	5.500	5.900	5.500	5.900
		n = 96	全部一致	52.800	54.200	41.200	41.800	41.200	41.800
			全部不一致	52.800	54.200	41.200	41.800	41.200	41.800
	DSUM (S) (D)	(S) = 0		3.700	4.100	3.300	3.600	3.300	3.600
		(S) = FFFFFFFFH		3.800	4.100	3.200	3.700	3.200	3.700
	DECO (S) (D) n	n = 2		6.000	7.500	5.300	6.900	5.300	6.900
		n = 8		8.100	9.300	6.800	7.800	6.800	7.800
	ENCO (S) (D) n	n = 2	M1 = ON	5.300	5.700	4.700	5.100	4.700	5.100
			M4 = ON	5.200	5.700	4.600	5.000	4.600	5.000
		n = 8	M1 = ON	10.400	11.400	9.000	10.000	9.000	10.000
			M256 = ON	5.700	6.800	5.100	6.100	5.100	6.100

分类	指令	条件（软元件）	处理时间（μs）					
			Q03UD(E)CPU		Q04UD(E)HCPU、Q06UD(E)HCPU		Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU	
			最小值	最大值	最小值	最大值	最小值	最大值
应用指令	DIS (S) (D) n	n = 1	4.400	5.300	3.800	4.600	3.800	4.600
		n = 4	4.800	5.700	4.000	5.000	4.000	5.000
	UNI (S) (D) n	n = 1	5.000	5.300	3.500	4.800	3.500	4.800
		n = 4	5.600	6.000	4.000	5.100	4.000	5.100
	NDIS	执行时	11.000	13.100	11.000	13.200	11.000	13.200
	NUNI	执行时	10.600	12.700	7.300	13.200	7.300	13.200
	WTOB (S) (D) n	n = 1	5.000	6.500	4.400	5.800	4.400	5.800
		n = 96	36.000	38.400	28.200	29.300	28.200	29.300
	BTOW (S) (D) n	n = 1	5.100	6.100	4.600	5.500	4.600	5.500
		n = 96	29.900	32.000	22.800	23.800	22.800	23.800
	MAX (S) (D) n	n = 1	4.300	6.900	4.000	6.100	4.000	6.100
		n = 96	31.200	33.500	24.700	27.000	24.700	27.000
	MIN (S) (D) n	n = 1	4.400	6.800	4.000	6.000	4.000	6.000
		n = 96	30.300	34.800	26.500	28.300	26.500	28.300
	DMAX (S) (D) n	n = 1	4.800	9.100	4.800	8.100	4.800	8.100
		n = 96	56.400	62.200	47.100	49.600	47.100	49.600
	DMIN (S) (D) n	n = 1	4.800	6.800	4.300	5.900	4.300	5.900
		n = 96	55.400	60.200	45.400	47.400	45.400	47.400
	SORT (S1) n (S2) (D1) (D2)	n = 1	6.200	9.300	5.600	8.800	5.600	8.800
		n = 96	6.200	9.400	5.600	8.600	5.600	8.600
	DSORT (S1) n (S2) (D1) (D2)	n = 1	6.200	9.300	5.600	8.200	5.600	8.200
		n = 96	6.100	9.100	5.600	8.400	5.600	8.400
	WSUM (S) (D) n	n = 1	4.800	6.200	4.200	5.500	4.200	5.500
		n = 96	26.900	28.700	21.300	22.300	21.300	22.300
	DWSUM (S) (D) n	n = 1	5.500	7.000	4.800	6.100	4.800	6.100
		n = 96	53.000	56.300	42.700	44.000	42.700	44.000
	MEAN (S) (D) n	n = 1	4.300	8.650	3.900	7.800	3.900	7.800
		n = 96	16.000	21.400	12.900	18.000	12.900	18.000
	DMEAN (S) (D) n	n = 1	5.700	10.600	5.300	9.950	5.300	9.950
		n = 96	29.200	35.200	23.000	28.800	23.000	28.800
	NEXT	-	0.940	1.400	0.770	1.200	0.770	1.200
	BREAK	-	10.400	5.500	9.100	5.000	9.100	5.000
	RET	返回至本程序	2.000	3.000	1.600	2.600	1.600	2.600
		返回至其它程序	2.300	3.700	2.000	3.100	2.000	3.100
	FCALL Pn	文件内指针	3.100	4.400	2.700	3.600	2.700	3.600
		公共指针	4.000	5.700	3.600	5.100	3.600	5.100
	FCALL Pn (S1) ～ (S5)	-	19.300	21.500	16.500	18.600	16.500	18.600
	ECALL * Pn *: 程序名	-	70.300	82.300	65.900	77.600	65.900	77.600
	ECALL * Pn (S1) ～ (S5) *: 程序名	-	101.000	114.000	91.800	105.000	91.800	105.000
	EFCALL * Pn *: 程序名	-	70.700	82.800	66.200	78.100	66.200	78.100
	EFCALL * Pn (S1) ～ (S5) *: 程序名	-	86.500	107.000	78.800	91.600	78.800	91.600
	XCALL	-	3.800	5.700	3.700	5.200	3.700	5.200

分类	指令	条件（软元件）	处理时间（μs）					
			Q03UD(E)CPU		Q04UD(E)HCPU，Q06UD(E)HCPU		Q10UD(E)HCPU，Q13UD(E)HCPU，Q20UD(E)HCPU，Q26UD(E)HCPU	
			最小值	最大值	最小值	最大值	最小值	最大值
应用指令	COM CCOM	仅选择 I/O 刷新时	12.800	29.100	12.400	28.600	12.400	28.600
		仅选择 CC-Link 刷新时（主站侧）	16.000	39.500	15.500	39.100	15.500	39.100
		仅选择 CC-Link 刷新时（本地站侧）	16.100	39.500	15.500	39.100	15.500	39.100
		仅选择 MELSECNET/H 时（管理站侧）	34.700	70.400	34.400	69.800	34.400	69.800
		仅选择 MELSECNET/H 时（普通站侧）	34.700	70.400	34.400	69.800	34.400	69.800
		仅选择智能自动刷新时	12.800	33.200	12.800	33.200	12.800	33.200
		仅选择组外输入输出时（仅输入）	7.900	21.100	7.700	20.700	7.700	20.700
		仅选择组外输入输出时（仅输出）	16.900	44.800	16.500	44.200	16.500	44.200
		仅选择组外输入输出时（输入及输出）	22.600	52.600	22.400	52.600	22.400	52.600
		仅选择多 CPU 高速通信区刷新时	13.000	33.800	12.700	33.200	12.700	33.200
		仅选择与外围设备的通信时	7.250	18.800	7.100	18.500	7.100	18.500
	FIFW	数据数 = 0	3.700	5.300	3.200	4.600	3.200	4.600
		数据数 = 96	3.800	4.400	3.300	3.800	3.300	3.800
	FIFR	数据数 = 0	4.300	5.000	3.800	4.400	3.800	4.400
		数据数 = 96	33.500	35.500	24.800	25.700	24.800	25.700
	FPOP	数据数 = 0	4.300	5.900	3.800	5.300	3.800	5.300
		数据数 = 96	4.300	5.900	3.700	5.400	3.700	5.400
	FINS	数据数 = 0	4.800	5.900	3.700	5.300	3.700	5.300
		数据数 = 96	4.300	5.900	3.700	5.300	3.700	5.300
	FDEL	数据数 = 0	4.900	6.500	4.200	5.800	4.200	5.800
		数据数 = 96	34.200	35.900	25.400	25.900	25.400	25.900
	FROM n1 n2 (D) n3	n3 = 1	10.500	16.900	10.000	16.400	10.000	16.400
		n3 = 1000	423.000	455.000	421.000	447.000	421.000	447.000
	DFRO n1 n2 (D) n3	n3 = 1	11.300	18.500	10.800	18.000	10.800	18.000
		n3 = 500	422.000	451.000	421.000	450.000	421.000	450.000
	TO n1 n2 (S) n3	n3 = 1	8.400	16.800	8.100	16.800	8.100	16.800
		n3 = 1000	466.000	498.000	466.000	494.000	466.000	494.000

分类	指令	条件（软元件）	处理时间（μs）					
			Q03UD(E)CPU		Q04UD(E)HCPU、Q06UD(E)HCPU		Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU	
			最小值	最大值	最小值	最大值	最小值	最大值
应用指令	DTO n1 n2 Ⓢ n3	n3 = 1	8.800	17.400	8.300	16.900	8.300	16.900
		n3 = 500	465.000	497.000	465.000	497.000	465.000	497.000
	LEDR	无显示→无显示	2.400	2.600	1.900	2.000	1.900	2.000
		LED 指令执行→无显示	28.100	39.400	24.400	35.800	24.400	35.800
	BINDA Ⓢ Ⓓ	Ⓢ = 1	4.900	6.500	4.300	5.600	4.300	5.600
		Ⓢ = -32768	7.200	8.700	6.500	8.000	6.500	8.000
	DBINDA Ⓢ Ⓓ	Ⓢ = 1	5.700	7.100	4.900	6.300	4.900	6.300
		Ⓢ = -2147483648	10.400	12.000	9.600	11.000	9.600	11.000
	BINHA Ⓢ Ⓓ	Ⓢ = 1	4.400	5.900	3.800	5.200	3.800	5.200
		Ⓢ = FFFFH	4.400	5.800	3.700	5.200	3.700	5.200
	DBINHA Ⓢ Ⓓ	Ⓢ = 1	5.200	6.700	4.600	6.000	4.600	6.000
		Ⓢ = FFFFFFFFH	5.100	6.500	4.600	6.000	4.600	6.000
	BCDDA Ⓢ Ⓓ	Ⓢ = 1	4.300	5.800	3.600	5.000	3.600	5.000
		Ⓢ = 9999	4.700	6.100	4.100	5.400	4.100	5.400
	DBCDDA Ⓢ Ⓓ	Ⓢ = 1	4.800	6.300	4.000	5.500	4.000	5.500
		Ⓢ = 99999999	5.600	7.100	4.900	6.300	4.900	6.300
	DABIN Ⓢ Ⓓ	Ⓢ = 1	6.500	8.500	5.800	7.800	5.800	7.800
		Ⓢ = -32768	6.300	8.300	5.600	7.700	5.600	7.700
	DDABIN Ⓢ Ⓓ	Ⓢ = 1	9.400	11.500	8.500	10.500	8.500	10.500
		Ⓢ = -2147483648	9.100	11.200	8.100	10.200	8.100	10.200
	HABIN Ⓢ Ⓓ	Ⓢ = 1	4.900	7.100	4.400	6.400	4.400	6.400
		Ⓢ = FFFFH	5.100	7.300	4.600	6.500	4.600	6.500
	DHABIN Ⓢ Ⓓ	Ⓢ = 1	6.000	8.100	5.300	7.300	5.300	7.300
		Ⓢ = FFFFFFFFH	6.300	8.500	5.600	7.700	5.600	7.700
	DABCD Ⓢ Ⓓ	Ⓢ = 1	5.000	7.100	4.400	6.300	4.400	6.300
		Ⓢ = 9999	5.000	7.100	4.300	6.300	4.300	6.300
	DDABCD Ⓢ Ⓓ	Ⓢ = 1	6.200	8.300	5.500	7.400	5.500	7.400
		Ⓢ = 99999999	6.200	8.300	5.500	7.500	5.500	7.500
	COMRD	-	51.600	52.400	50.900	51.200	50.900	51.200
	LEN	1 字符	4.100	6.200	3.600	5.500	3.600	5.500
		96 字符	19.800	22.200	16.800	18.700	16.800	18.700
	STR	-	6.900	11.100	6.600	10.400	6.600	10.400
	DSTR	-	10.200	12.500	9.600	11.500	9.600	11.500
	VAL	-	9.800	14.200	8.900	13.000	8.900	13.000
	DVAL	-	14.000	18.700	12.700	16.800	12.700	16.800
	ESTR	-	18.700	24.100	17.900	23.100	17.900	23.100

分类	指令	条件（软元件）		处理时间（μs）					
				Q03UD(E)CPU		Q04UD(E)HCPU、Q06UD(E)HCPU		Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU	
				最小值	最大值	最小值	最大值	最小值	最大值
应用指令	LDDT =	与指定的日期比较	导通时	7.400	11.400	6.800	10.900	6.800	10.900
			非导通时	7.400	11.600	6.800	10.900	6.800	10.900
		与当前的日期比较	导通时	5.900	10.000	5.500	9.700	5.500	9.700
			非导通时	5.900	10.100	5.500	9.700	5.500	9.700
	ANDDT =	未执行时		0.008		0.038		0.038	
		与指定的日期比较	导通时	7.200	11.400	6.500	10.700	6.500	10.700
			非导通时	7.200	11.400	6.500	10.700	6.500	10.700
		与当前的日期比较	导通时	5.700	9.900	5.300	9.300	5.300	9.300
			非导通时	5.700	9.900	5.300	9.300	5.300	9.300
	ORDT =	未执行时		0.008		0.038		0.038	
		与指定的日期比较	导通时	7.400	11.500	6.700	10.800	6.700	10.800
			非导通时	7.400	11.500	6.700	10.800	6.700	10.800
		与当前的日期比较	导通时	5.900	10.000	5.400	9.600	5.400	9.600
			非导通时	5.900	10.000	5.400	9.600	5.400	9.600
	LDDT< >	与指定的日期比较	导通时	7.400	11.400	6.800	10.900	6.800	10.900
			非导通时	7.400	11.600	6.800	10.900	6.800	10.900
		与当前的日期比较	导通时	5.900	10.000	5.500	9.700	5.500	9.700
			非导通时	5.900	10.100	5.500	9.700	5.500	9.700
	ANDDT< >	未执行时		0.008		0.038		0.038	
		与指定的日期比较	导通时	7.200	11.400	6.500	10.700	6.500	10.700
			非导通时	7.200	11.400	6.500	10.700	6.500	10.700
		与当前的日期比较	导通时	5.700	9.900	5.300	9.300	5.300	9.300
			非导通时	5.700	9.900	5.300	9.300	5.300	9.300
	ORDT< >	未执行时		0.008		0.038		0.038	
		与指定的日期比较	导通时	7.400	11.500	6.700	10.800	6.700	10.800
			非导通时	7.400	11.500	6.700	10.800	6.700	10.800
		与当前的日期比较	导通时	5.900	10.000	5.400	9.600	5.400	9.600
			非导通时	5.900	10.000	5.400	9.600	5.400	9.600
	LDDT >	与指定的日期比较	导通时	7.400	11.400	6.800	10.900	6.800	10.900
			非导通时	7.400	11.600	6.800	10.900	6.800	10.900
		与当前的日期比较	导通时	5.900	10.000	5.500	9.700	5.500	9.700
			非导通时	5.900	10.100	5.500	9.700	5.500	9.700
	ANDDT >	未执行时		0.008		0.038		0.038	
		与指定的日期比较	导通时	7.200	11.400	6.500	10.700	6.500	10.700
			非导通时	7.200	11.400	6.500	10.700	6.500	10.700
		与当前的日期比较	导通时	5.700	9.900	5.300	9.300	5.300	9.300
			非导通时	5.700	9.900	5.300	9.300	5.300	9.300
	ORDT >	未执行时		0.008		0.038		0.038	
		与指定的日期比较	导通时	7.400	11.500	6.700	10.800	6.700	10.800
			非导通时	7.400	11.500	6.700	10.800	6.700	10.800
		与当前的日期比较	导通时	5.900	10.000	5.400	9.600	5.400	9.600
			非导通时	5.900	10.000	5.400	9.600	5.400	9.600

分类	指令	条件（软元件）		处理时间（μs）					
				Q03UD(E)CPU		Q04UD(E)HCPU、Q06UD(E)HCPU		Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU	
				最小值	最大值	最小值	最大值	最小值	最大值
应用指令	LDDT< =	与指定的日期比较	导通时	7.400	11.400	6.800	10.900	6.800	10.900
			非导通时	7.400	11.600	6.800	10.900	6.800	10.900
		与当前的日期比较	导通时	5.900	10.000	5.500	9.700	5.500	9.700
			非导通时	5.900	10.100	5.500	9.700	5.500	9.700
	ANDDT< =	未执行时		0.008		0.038		0.038	
		与指定的日期比较	导通时	7.200	11.400	6.500	10.700	6.500	10.700
			非导通时	7.200	11.400	6.500	10.700	6.500	10.700
		与当前的日期比较	导通时	5.700	9.900	5.300	9.300	5.300	9.300
			非导通时	5.700	9.900	5.300	9.300	5.300	9.300
	ORDT< =	未执行时		0.008		0.038		0.038	
		与指定的日期比较	导通时	7.400	11.500	6.700	10.800	6.700	10.800
			非导通时	7.400	11.500	6.700	10.800	6.700	10.800
		与当前的日期比较	导通时	5.900	10.000	5.400	9.600	5.400	9.600
			非导通时	5.900	10.000	5.400	9.600	5.400	9.600
	LDDT<	与指定的日期比较	导通时	7.400	11.400	6.800	10.900	6.800	10.900
			非导通时	7.400	11.600	6.800	10.900	6.800	10.900
		与当前的日期比较	导通时	5.900	10.000	5.500	9.700	5.500	9.700
			非导通时	5.900	10.100	5.500	9.700	5.500	9.700
	ANDDT<	未执行时		0.008		0.038		0.038	
		与指定的日期比较	导通时	7.200	11.400	6.500	10.700	6.500	10.700
			非导通时	7.200	11.400	6.500	10.700	6.500	10.700
		与当前的日期比较	导通时	5.700	9.900	5.300	9.300	5.300	9.300
			非导通时	5.700	9.900	5.300	9.300	5.300	9.300
	ORDT<	未执行时		0.008		0.038		0.038	
		与指定的日期比较	导通时	7.400	11.500	6.700	10.800	6.700	10.800
			非导通时	7.400	11.500	6.700	10.800	6.700	10.800
		与当前的日期比较	导通时	5.900	10.000	5.400	9.600	5.400	9.600
			非导通时	5.900	10.000	5.400	9.600	5.400	9.600
	LDDT > =	与指定的日期比较	导通时	7.400	11.400	6.800	10.900	6.800	10.900
			非导通时	7.400	11.600	6.800	10.900	6.800	10.900
		与当前的日期比较	导通时	5.900	10.000	5.500	9.700	5.500	9.700
			非导通时	5.900	10.100	5.500	9.700	5.500	9.700
	ANDDT > =	未执行时		0.008		0.038		0.038	
		与指定的日期比较	导通时	7.200	11.400	6.500	10.700	6.500	10.700
			非导通时	7.200	11.400	6.500	10.700	6.500	10.700
		与当前的日期比较	导通时	5.700	9.900	5.300	9.300	5.300	9.300
			非导通时	5.700	9.900	5.300	9.300	5.300	9.300
	ORDT > =	未执行时		0.008		0.038		0.038	
		与指定的日期比较	导通时	7.400	11.500	6.700	10.800	6.700	10.800
			非导通时	7.400	11.500	6.700	10.800	6.700	10.800
		与当前的日期比较	导通时	5.900	10.000	5.400	9.600	5.400	9.600
			非导通时	5.900	10.000	5.400	9.600	5.400	9.600

分类	指令	条件（软元件）		处理时间（μs）					
				Q03UD(E)CPU		Q04UD(E)HCPU、Q06UD(E)HCPU		Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU	
				最小值	最大值	最小值	最大值	最小值	最大值
应用指令	LDTM =	与指定的时间比较	导通时	7.300	11.500	6.700	10.800	6.700	10.800
			非导通时	7.300	11.500	6.700	10.800	6.700	10.800
		与当前的时间比较	导通时	5.800	9.900	5.400	9.500	5.400	9.500
			非导通时	5.800	9.900	5.400	9.500	5.400	9.500
	ANDTM =	未执行时		0.008		0.038		0.038	
		与指定的时间比较	导通时	7.000	11.500	6.300	10.800	6.300	10.800
			非导通时	7.000	11.500	6.300	10.800	6.300	10.800
		与当前的时间比较	导通时	5.500	9.900	5.100	9.500	5.100	9.500
			非导通时	5.500	9.900	5.100	9.500	5.100	9.500
	ORTM =	未执行时		0.008		0.038		0.038	
		与指定的时间比较	导通时	7.300	11.500	6.600	10.800	6.600	10.800
			非导通时	7.300	11.500	6.600	10.800	6.600	10.800
		与当前的时间比较	导通时	5.900	9.900	5.300	9.500	5.300	9.500
			非导通时	5.900	9.900	5.300	9.500	5.300	9.500
	LDTM< >	与指定的时间比较	导通时	7.300	11.500	6.700	10.800	6.700	10.800
			非导通时	7.300	11.500	6.700	10.800	6.700	10.800
		与当前的时间比较	导通时	5.800	9.900	5.400	9.500	5.400	9.500
			非导通时	5.800	9.900	5.400	9.500	5.400	9.500
	ANDTM< >	未执行时		0.008		0.038		0.038	
		与指定的时间比较	导通时	7.000	11.500	6.300	10.800	6.300	10.800
			非导通时	7.000	11.500	6.300	10.800	6.300	10.800
		与当前的时间比较	导通时	5.500	9.900	5.100	9.500	5.100	9.500
			非导通时	5.500	9.900	5.100	9.500	5.100	9.500
	ORTM< >	未执行时		0.008		0.038		0.038	
		与指定的时间比较	导通时	7.300	11.500	6.600	10.800	6.600	10.800
			非导通时	7.300	11.500	6.600	10.800	6.600	10.800
		与当前的时间比较	导通时	5.900	9.900	5.300	9.500	5.300	9.500
			非导通时	5.900	9.900	5.300	9.500	5.300	9.500
	LDTM >	与指定的时间比较	导通时	7.300	11.500	6.700	10.800	6.700	10.800
			非导通时	7.300	11.500	6.700	10.800	6.700	10.800
		与当前的时间比较	导通时	5.800	9.900	5.400	9.500	5.400	9.500
			非导通时	5.800	9.900	5.400	9.500	5.400	9.500
	ANDTM >	未执行时		0.008		0.038		0.038	
		与指定的时间比较	导通时	7.000	11.500	6.300	10.800	6.300	10.800
			非导通时	7.000	11.500	6.300	10.800	6.300	10.800
		与当前的时间比较	导通时	5.500	9.900	5.100	9.500	5.100	9.500
			非导通时	5.500	9.900	5.100	9.500	5.100	9.500
	ORTM >	未执行时		0.008		0.038		0.038	
		与指定的时间比较	导通时	7.300	11.500	6.600	10.800	6.600	10.800
			非导通时	7.300	11.500	6.600	10.800	6.600	10.800
		与当前的时间比较	导通时	5.900	9.900	5.300	9.500	5.300	9.500
			非导通时	5.900	9.900	5.300	9.500	5.300	9.500

分类	指令	条件（软元件）		处理时间（μs）					
				Q03UD(E)CPU		Q04UD(E)HCPU、Q06UD(E)HCPU		Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU	
				最小值	最大值	最小值	最大值	最小值	最大值
应用指令	LDTM< =	与指定的时间比较	导通时	7.300	11.500	6.700	10.800	6.700	10.800
			非导通时	7.300	11.500	6.700	10.800	6.700	10.800
		与当前的时间比较	导通时	5.800	9.900	5.400	9.500	5.400	9.500
			非导通时	5.800	9.900	5.400	9.500	5.400	9.500
	ANDTM< =	未执行时		0.008		0.038		0.038	
		与指定的时间比较	导通时	7.000	11.500	6.300	10.800	6.300	10.800
			非导通时	7.000	11.500	6.300	10.800	6.300	10.800
		与当前的时间比较	导通时	5.500	9.900	5.100	9.500	5.100	9.500
			非导通时	5.500	9.900	5.100	9.500	5.100	9.500
	ORTM< =	未执行时		0.008		0.038		0.038	
		与指定的时间比较	导通时	7.300	11.500	6.600	10.800	6.600	10.800
			非导通时	7.300	11.500	6.600	10.800	6.600	10.800
		与当前的时间比较	导通时	5.900	9.900	5.300	9.500	5.300	9.500
			非导通时	5.900	9.900	5.300	9.500	5.300	9.500
	LDTM<	与指定的时间比较	导通时	7.300	11.500	6.700	10.800	6.700	10.800
			非导通时	7.300	11.500	6.700	10.800	6.700	10.800
		与当前的时间比较	导通时	5.800	9.900	5.400	9.500	5.400	9.500
			非导通时	5.800	9.900	5.400	9.500	5.400	9.500
	ANDTM<	未执行时		0.008		0.038		0.038	
		与指定的时间比较	导通时	7.000	11.500	6.300	10.800	6.300	10.800
			非导通时	7.000	11.500	6.300	10.800	6.300	10.800
		与当前的时间比较	导通时	5.500	9.900	5.100	9.500	5.100	9.500
			非导通时	5.500	9.900	5.100	9.500	5.100	9.500
	ORTM<	未执行时		0.008		0.038		0.038	
		与指定的时间比较	导通时	7.300	11.500	6.600	10.800	6.600	10.800
			非导通时	7.300	11.500	6.600	10.800	6.600	10.800
		与当前的时间比较	导通时	5.900	9.900	5.300	9.500	5.300	9.500
			非导通时	5.900	9.900	5.300	9.500	5.300	9.500
	LDTM > =	与指定的时间比较	导通时	7.300	11.500	6.700	10.800	6.700	10.800
			非导通时	7.300	11.500	6.700	10.800	6.700	10.800
		与当前的时间比较	导通时	5.800	9.900	5.400	9.500	5.400	9.500
			非导通时	5.800	9.900	5.400	9.500	5.400	9.500
	ANDTM > =	未执行时		0.008		0.038		0.038	
		与指定的时间比较	导通时	7.000	11.500	6.300	10.800	6.300	10.800
			非导通时	7.000	11.500	6.300	10.800	6.300	10.800
		与当前的时间比较	导通时	5.500	9.900	5.100	9.500	5.100	9.500
			非导通时	5.500	9.900	5.100	9.500	5.100	9.500
	ORTM > =	未执行时		0.008		0.038		0.038	
		与指定的时间比较	导通时	7.300	11.500	6.600	10.800	6.600	10.800
			非导通时	7.300	11.500	6.600	10.800	6.600	10.800
		与当前的时间比较	导通时	5.900	9.900	5.300	9.500	5.300	9.500
			非导通时	5.900	9.900	5.300	9.500	5.300	9.500

分类	指令	条件（软元件）		处理时间（μs）					
				Q03UD(E)CPU		Q04UD(E)HCPU、Q06UD(E)HCPU		Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU	
				最小值	最大值	最小值	最大值	最小值	最大值
应用指令	EVAL	小数点形式全2位数指定		23.300	30.400	22.800	29.000	22.800	29.000
		指数形式全6位数指定		23.300	30.500	22.500	29.000	22.500	29.000
	ASC Ⓢ Ⓓ n	n = 1		5.600	9.000	5.400	8.300	5.400	8.300
		n = 96		28.700	32.100	25.200	28.400	25.200	28.400
	HEX Ⓢ Ⓓ n	n = 1		6.000	9.700	5.400	9.000	5.400	9.000
		n = 96		35.600	39.800	31.300	35.000	31.300	35.000
	RIGHT Ⓢ Ⓓ n	n = 1		7.600	9.400	7.300	6.600	7.300	6.600
		n = 96		36.300	40.000	29.200	31.600	29.200	31.600
	LEFT Ⓢ Ⓓ n	n = 1		6.500	8.900	5.900	8.200	5.900	8.200
		n = 96		36.200	39.700	29.200	31.500	29.200	31.500
	MIDR	-		9.500	12.100	8.100	10.300	8.100	10.300
	MIDW	-		10.300	12.000	8.800	10.200	8.800	10.200
	INSTR	无一致		19.300	21.800	16.600	18.400	16.600	18.400
		有一致	起始	10.300	12.800	9.100	10.900	9.100	10.900
			最终	51.100	54.200	42.700	44.900	42.700	44.900
	EMOD	-		10.300	11.800	9.600	11.000	9.600	11.000
	EREXP	-		19.300	21.000	18.800	20.100	18.800	20.100
	STRINS Ⓢ Ⓓ n	S = 128/D = 40/n = 1		41.100	54.200	35.300	47.600	35.300	47.600
		S = 128/D = 40/n = 48		56.700	81.400	48.600	61.700	48.600	61.700
	STRDEL Ⓢ Ⓓ n	S = 128/D = 40/n = 1		39.000	49.500	34.800	44.600	34.800	44.600
		S = 128/D = 40/n = 48		36.000	45.200	29.200	38.100	29.200	38.100
	SIN	单精度		4.500	6.200	4.100	5.700	4.100	5.700
	COS	单精度		4.300	6.000	4.000	5.600	4.000	5.600
	TAN	单精度		5.100	7.200	5.100	6.700	5.100	6.700
	ASIN	单精度		6.100	8.900	5.900	8.500	5.900	8.500
	ACOS	单精度		6.800	9.300	6.700	8.900	6.700	8.900
	ATAN	单精度		4.000	6.500	3.900	6.000	3.900	6.000
	SIND	双精度		8.800	14.300	8.500	13.800	8.500	13.800
	COSD	双精度		9.300	15.100	8.800	14.600	8.800	14.600
	TAND	双精度		11.200	16.900	10.800	16.500	10.800	16.500
	ASIND	双精度		12.000	17.100	11.600	16.600	11.600	16.600
	ACOSD	双精度		11.700	16.500	11.200	16.200	11.200	16.200
	ATAND	双精度		9.500	14.200	9.100	13.800	9.100	13.800
	RAD	单精度		2.500	4.800	2.100	4.300	2.100	4.300
	RADD	双精度		4.000	9.600	3.600	9.200	3.600	9.200
	DEG	单精度		2.500	4.700	2.200	4.400	2.200	4.400
	DEGD	双精度		4.300	9.000	3.800	9.000	3.800	9.000
	SQR	单精度		3.000	4.600	2.600	4.300	2.600	4.300
	SQRD	双精度		5.600	11.500	5.200	11.000	5.200	11.000
	EXP Ⓢ Ⓓ	单精度	Ⓢ = -10	4.000	6.100	3.800	5.500	3.800	5.500
			Ⓢ = 1	4.000	6.100	3.800	5.600	3.800	5.600
	EXPD Ⓢ Ⓓ	双精度	Ⓢ = -10	8.700	13.900	8.200	13.500	8.200	13.500
			Ⓢ = 1	8.400	13.600	8.000	13.200	8.000	13.200
	LOG Ⓢ Ⓓ	单精度	Ⓢ = 1	4.100	6.900	3.800	6.400	3.800	6.400
			Ⓢ = 10	5.600	8.200	5.200	7.700	5.200	7.700
	LOGD Ⓢ Ⓓ	双精度	Ⓢ = 1	8.100	13.000	7.700	12.500	7.700	12.500
			Ⓢ = 10	9.700	14.800	9.200	14.300	9.200	14.300
	RND	-		1.200	2.300	0.800	1.800	0.800	1.800

分类	指令	条件（软元件）		处理时间（μs）					
				Q03UD(E)CPU		Q04UD(E)HCPU、Q06UD(E)HCPU		Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU	
				最小值	最大值	最小值	最大值	最小值	最大值
应用指令	SRND	-		1.400	2.400	1.100	2.000	1.100	2.000
	BSQR (S) (D)	(S) = 0		1.800	3.300	1.600	2.800	1.600	2.800
		(S) = 9999		5.100	8.800	5.100	8.000	5.100	8.000
	BDSQR (S) (D)	(S) = 0		1.900	3.400	1.500	3.000	1.500	3.000
		(S) = 99999999		7.500	10.200	7.500	9.900	7.500	9.900
	BSIN	-		8.600	15.100	8.100	14.500	8.100	14.500
	BCOS	-		7.800	14.400	7.800	13.700	7.800	13.700
	BTAN	-		9.000	13.800	9.000	13.300	9.000	13.300
	BASIN	-		10.600	13.400	10.100	12.800	10.100	12.800
	BACOS	-		11.600	14.400	11.100	14.100	11.100	14.100
	BATAN	-		9.800	11.700	9.100	10.900	9.100	10.900
	POW (S1) (S2) (D)	单精度	S1 = 12.3E+5 S2 = 3.45E+0	8.750	11.400	8.400	10.900	8.400	10.900
	POWD (S1) (S2) (D)	双精度	S1 = 12.3E+5 S2 = 3.45E+0	18.600	27.200	18.200	26.500	18.200	26.500
	LOG10	-		5.900	8.550	5.700	8.050	5.700	8.050
	LOG10D	-		11.500	19.400	11.100	18.600	11.100	18.600
	LIMIT	-		2.800	3.100	2.400	2.700	2.400	2.700
	DLIMIT	-		3.200	3.500	2.800	3.000	2.800	3.000
	BAND	-		3.000	4.300	2.700	3.800	2.700	3.800
	DBAND	-		3.600	5.100	3.300	4.600	3.300	4.600
	ZONE	-		3.000	4.700	2.600	4.300	2.600	4.300
	DZONE	-		3.400	5.000	3.000	4.600	3.000	4.600

分类	指令	条件（软元件）		处理时间（μs）					
				Q03UD(E)CPU		Q04UD(E)HCPU、Q06UD(E)HCPU		Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU	
				最小值	最大值	最小值	最大值	最小值	最大值
应用指令	SCL (S1) (S2) (D)	SM750 = ON	点 No.1<(S1)<点 No.2	13.200	23.600	12.300	22.500	12.300	22.500
			点 No.9<(S1)<点 No.10	13.300	23.600	12.600	22.700	12.600	22.700
		SM750 = OFF	点 No.1<(S1)<点 No.2	12.000	23.100	11.400	22.200	11.400	22.200
			点 No.9<(S1)<点 No.10	14.100	25.300	12.800	23.900	12.800	23.900
	DSCL (S1) (S2) (D)	SM750 = ON	点 No.1<(S1)<点 No.2	12.800	23.800	11.900	23.000	11.900	23.000
			点 No.9<(S1)<点 No.10	12.900	23.900	12.100	23.000	12.100	23.000
		SM750 = OFF	点 No.1<(S1)<点 No.2	11.500	22.400	10.900	21.500	10.900	21.500
			点 No.9<(S1)<点 No.10	13.800	24.900	12.700	23.600	12.700	23.600
	SCL2 (S1) (S2) (D)	SM750 = ON	点 No.1<(S1)<点 No.2	12.700	24.200	11.900	23.300	11.900	23.300
			点 No.9<(S1)<点 No.10	12.900	24.600	12.100	23.300	12.100	23.300
		SM750 = OFF	点 No.1<(S1)<点 No.2	12.300	23.400	11.500	22.600	11.500	22.600
			点 No.9<(S1)<点 No.10	13.700	25.000	12.600	23.900	12.600	23.900
	DSCL2 (S1) (S2) (D)	SM750 = ON	点 No.1<(S1)<点 No.2	12.600	23.800	11.800	22.900	11.800	22.900
			点 No.9<(S1)<点 No.10	13.000	23.900	12.200	22.800	12.200	22.800
		SM750 = OFF	点 No.1<(S1)<点 No.2	11.500	22.400	11.000	21.400	11.000	21.400
			点 No.9<(S1)<点 No.10	13.900	24.900	12.800	23.600	12.800	23.600

分类	指令	条件（软元件）		处理时间（μs）					
				Q03UD(E)CPU		Q04UD(E)HCPU、Q06UD(E)HCPU		Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU	
				最小值	最大值	最小值	最大值	最小值	最大值
应用指令	RSET	标准 RAM		3.000	6.300	2.700	5.900	2.700	5.900
		SRAM		3.000	6.400	2.600	5.800	2.600	5.800
	DATE-	无进位		5.800	8.500	4.600	7.000	4.600	7.000
		有进位		5.700	7.400	4.600	6.500	4.600	6.500
	SECOND	-		2.600	3.900	2.200	3.400	2.200	3.400
	HOUR	-		2.900	4.800	2.400	4.300	2.400	4.300
	QDRSET	SRAM →标准 RAM		120.000	134.000	115.000	134.000	115.000	134.000
		标准 RAM → SRAM		533.000	560.000	520.000	553.000	520.000	553.000
	QCDSET	SRAM →标准 ROM		306.000	346.000	305.000	346.000	305.000	346.000
		标准 ROM → SRAM		311.000	342.000	300.000	334.000	300.000	334.000
	DATERD	-		3.200	5.000	2.500	4.200	2.500	4.200
	DATEWR	-		4.900	9.700	4.100	8.900	4.100	8.900
	DATE +	无进位		5.100	8.000	4.700	6.600	4.700	6.600
		有进位		5.700	8.000	4.600	6.500	4.600	6.500
	S.DATERD	-		5.600	8.000	4.800	7.100	4.800	7.100
	S.DATE+	无进位		9.100	11.700	7.400	10.000	7.400	11.000
		有进位		8.900	11.800	7.400	10.000	7.400	11.000
	S.DATE-	无进位		9.000	12.000	7.400	10.300	7.400	10.300
		有进位		9.000	12.200	7.500	10.200	7.500	10.200
	PSTOP	-		61.400	84.500	56.600	79.800	56.600	79.800
	POFF	-		61.800	84.500	57.200	79.800	57.200	79.800
	PSCAN	-		64.900	84.700	60.100	79.900	60.100	79.900
	WDT	-		1.300	2.700	1.100	2.400	1.100	2.400
	DUTY	-		4.900	10.100	4.800	9.600	4.800	9.600
	TIMCHK	-		3.800	5.300	3.500	4.700	3.500	4.700
	ZRRDB	标准 RAM 的文件寄存器		2.400	2.600	1.800	2.100	1.800	2.100
		SRAM 的文件寄存器		2.500	2.800	2.000	2.300	2.000	2.300
	ZRWRB	标准 RAM 的文件寄存器		3.000	3.300	2.400	2.700	2.400	2.700
		SRAM 的文件寄存器		3.200	3.600	2.600	3.000	2.600	3.000
	ADRSET	-		2.600	3.100	2.100	2.600	2.100	2.600
	ZPUSH	-		6.900	9.200	5.800	7.500	5.800	7.500
	ZPOP	-		7.500	8.800	5.800	6.400	5.800	6.400
	S.ZCOM	安装 CC-Link 模块时（主站侧）		19.600	26.500	19.300	26.000	19.300	26.000
		安装 CC-Link 模块时（本地站侧）		19.600	26.500	19.100	26.200	19.100	26.200
		MELSECNET/H、CC-Link IE 控制网络模块 装着時（管理站侧）		53.500	73.500	53.000	72.700	53.000	72.700
		安装 MELSECNET/H、CC-Link IE 控制网络模块时（普通站侧）		29.800	41.200	29.800	40.600	29.800	40.600
	S.RTREAD	-		5.900	11.000	5.400	10.500	5.400	10.500
	S.RTWRITE	-		6.700	11.100	6.000	10.400	6.000	10.400
	UNIRD n1 Ⓓ n2	n2 = 1		4.000	8.400	3.700	8.000	3.700	8.000
		n2 = 16		12.500	17.000	12.200	16.600	12.200	16.600
	TRACE	开始		46.600	48.300	43.800	44.700	43.800	44.700
	TRACER	-		3.300	6.800	2.600	6.000	2.600	6.000
	RBMOV Ⓢ Ⓓ n	使用标准 RAM	1 点	10.100	16.500	9.470	16.000	9.470	16.000
			1000 点	231.000	237.000	119.000	125.000	119.000	125.000
		使用 SRAM	1 点	9.200	12.100	9.100	11.500	9.100	11.500
			1000 点	489.000	509.000	464.000	487.000	464.000	487.000

分类	指令	条件（软元件）		处理时间（μs）					
				Q03UD(E)CPU		Q04UD(E)HCPU、Q06UD(E)HCPU		Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU	
				最小值	最大值	最小值	最大值	最小值	最大值
应用指令	SP.FWRITE	-		4.100	55.600	3.700	53.700	3.700	53.700
	SP.FREAD	-		4.600	54.600	4.000	53.000	4.000	53.000
	SP.DEVST	-		4.500	36.500	4.000	34.500	4.000	34.500
	S.DEVLD	-		11.000	17.800	10.000	17.000	10.000	17.000
多CPU专用指令	S.TO n1 n2 n3 n4 (D)	至本机CPU共享存储器的写入	n4 = 1	34.700	34.900	33.500	34.400	33.500	34.400
			n4 = 320	85.900	87.600	75.200	75.500	75.200	75.500
	TO n1 n2 (S) n3	至本机CPU共享存储器的写入	n3 = 1	5.600	10.200	3.300	9.900	3.300	9.900
			n3 = 320	36.700	42.400	34.300	39.200	34.300	39.200
	DTO n1 n2 (S) n3	至本机CPU共享存储器的写入	n3 = 1	5.000	12.100	3.100	10.500	3.100	10.500
			n3 = 320	59.100	66.800	55.300	65.100	55.300	65.100
	FROM n1 n2 (D) n3	本机CPU共享存储器读取	n3 = 1	3.300	12.700	2.400	9.600	2.400	9.600
			n3 = 320	50.900	64.400	45.200	48.200	45.200	48.200
		其它机号CPU共享存储器读取	n3 = 1	11.600	17.700	10.600	13.900	10.600	13.900
			n3 = 320	142.000	160.000	142.000	149.000	142.000	149.000
			n3 = 1000	431.000	463.000	422.000	448.000	422.000	448.000
	DFRO n1 n2 (D) n3	本机CPU共享存储器读取	n3 = 1	6.700	12.600	2.800	9.900	2.800	9.900
			n3 = 320	96.500	105.000	84.300	96.700	84.300	96.700
		其它机号CPU共享存储器读取	n3 = 1	12.900	20.800	12.200	17.100	12.200	17.100
			n3 = 320	277.000	299.000	274.000	291.000	274.000	291.000
			n3 = 1000	838.000	860.000	835.000	857.000	835.000	857.000
多CPU高速通信专用指令	D.DDWR n (S1) (S2) (D1) (D2)	至其它机号CPU的软元件写入	值 = 1	82.000	133.000	80.000	130.000	80.000	130.000
			值 = 16	91.000	142.000	89.000	139.000	89.000	139.000
			n = 96*1	136.000	189.000	132.000	182.000	132.000	182.000
	DP.DDWR n (S1) (S2) (D1) (D2)		n = 1	82.000	133.000	80.000	130.000	80.000	130.000
			n = 16	91.000	142.000	89.000	139.000	89.000	139.000
			n = 96*1	136.000	189.000	132.000	182.000	132.000	182.000
	D.DDRD n (S1) (S2) (D1) (D2)	至其它机号CPU的软元件读取	n = 1	82.000	133.000	80.000	130.000	80.000	130.000
			n = 16	82.000	133.000	80.000	130.000	80.000	130.000
			n = 96*1	82.000	133.000	80.000	130.000	80.000	130.000
	DP.DDRD n (S1) (S2) (D1) (D2)		n = 1	82.000	133.000	80.000	130.000	80.000	130.000
			n = 16	82.000	133.000	80.000	130.000	80.000	130.000
			n = 96*1	82.000	133.000	80.000	130.000	80.000	130.000

*1：序列号的前5位数为“10012”以后的Q02UCPU、Q03UDCPU、Q04UDHCPU、Q06UDHCPU、Q13UDHCPU、Q26UDHCPU可以使用。

备注

对于表中未记述上升沿执行指令（□P）的指令，其处理时间与ON时执行的指令的处理时间相同。

例 WORDP指令、TOP指令等。

(2) 使用了文件寄存器、模块访问软元件、链接直接软元件时的加法运算时间一览表

软元件名		数据	软元件指定位置	加法运算时间（μs）			
				Q00UJCPU	Q00UCPU	Q01UCPU	Q02UCPU
文件寄存器(R)	使用标准 RAM 时	位	源	0.100	0.100	0.100	0.100
			目标	0.100	0.100	0.100	0.100
		字	源	0.100	0.100	0.100	0.100
			目标	0.100	0.100	0.100	0.100
		双字	源	0.100	0.100	0.100	0.200
			目标	0.100	0.100	0.100	0.200
	使用 SRAM 卡(Q2MEM-1MBS、Q2MEM-2MBS)时	位	源	-	-	-	0.220
			目标	-	-	-	0.180
		字	源	-	-	-	0.220
			目标	-	-	-	0.180
		双字	源	-	-	-	0.440
			目标	-	-	-	0.380
	使用 SRAM 卡(Q3MEM-4MBS、Q3MEM-8MBS)时	位	源	-	-	-	0.160
			目标	-	-	-	0.140
		字	源	-	-	-	0.160
			目标	-	-	-	0.140
		双字	源	-	-	-	0.320
			目标	-	-	-	0.300
文件寄存器(ZR)、扩展数据寄存器(D)、扩展链接寄存器(W)	使用标准 RAM 时	位	源	0.120	0.120	0.120	0.120
			目标	0.120	0.120	0.120	0.120
		字	源	0.120	0.120	0.120	0.120
			目标	0.120	0.120	0.120	0.120
		双字	源	0.120	0.120	0.120	0.220
			目标	0.120	0.120	0.120	0.220
	使用 SRAM 卡(Q2MEM-1MBS、Q2MEM-2MBS)时	位	源	-	-	-	0.240
			目标	-	-	-	0.200
		字	源	-	-	-	0.240
			目标	-	-	-	0.200
		双字	源	-	-	-	0.460
			目标	-	-	-	0.400
	使用 SRAM 卡(Q3MEM-4MBS、Q3MEM-8MBS)时	位	源	-	-	-	0.180
			目标	-	-	-	0.160
		字	源	-	-	-	0.180
			目标	-	-	-	0.160
		双字	源	-	-	-	0.340
			目标	-	-	-	0.320
模块访问软元件(Un\G□、U3En\G0 ～ G4095)		位	源	-	-	-	12.000
			目标	-	-	-	17.300
		字	源	-	-	-	9.700
			目标	-	-	-	33.000
		双字	源	-	-	-	24.200
			目标	-	-	-	34.800
链接直接软元件(Jn\□)		位	源	-	-	-	32.900
			目标	-	-	-	67.300
		字	源	-	-	-	37.200
			目标	-	-	-	37.000
		双字	源	-	-	-	39.500
			目标	-	-	-	41.900

软元件名		数据	软元件指定位置	加法运算时间（μs）		
				Q03UD(E)CPU	Q04UD(E)HCPU、Q06UD(E)HCPU	Q10UD(E)HCPU、Q13UD(E)HCPU、Q20UD(E)HCPU、Q26UD(E)HCPU
文件寄存器(R)	使用标准 RAM 时	位	源	0.100	0.048	0.048
			目标	0.100	0.038	0.038
		字	源	0.100	0.048	0.048
			目标	0.100	0.038	0.038
		双字	源	0.200	0.095	0.095
			目标	0.200	0.086	0.086
	使用 SRAM 卡(Q2MEM-1MBS、Q2MEM-2MBS)时	位	源	0.220	0.200	0.200
			目标	0.180	0.162	0.162
		字	源	0.220	0.200	0.200
			目标	0.180	0.162	0.162
		双字	源	0.440	0.399	0.399
			目标	0.380	0.361	0.361
	使用 SRAM 卡(Q3MEM-4MBS、Q3MEM-8MBS)时	位	源	0.160	0.152	0.152
			目标	0.140	0.133	0.133
		字	源	0.160	0.152	0.152
			目标	0.140	0.133	0.133
		双字	源	0.320	0.304	0.304
			目标	0.300	0.295	0.295
文件寄存器(ZR)、扩展数据寄存器(D)、扩展链接寄存器(W)	使用标准 RAM 时	位	源	0.120	0.057	0.057
			目标	0.120	0.048	0.048
		字	源	0.120	0.057	0.057
			目标	0.120	0.048	0.048
		双字	源	0.220	0.105	0.105
			目标	0.220	0.095	0.095
	使用 SRAM 卡(Q2MEM-1MBS、Q2MEM-2MBS)时	位	源	0.240	0.209	0.209
			目标	0.200	0.171	0.171
		字	源	0.240	0.209	0.209
			目标	0.200	0.171	0.171
		双字	源	0.460	0.409	0.409
			目标	0.400	0.371	0.371
	使用 SRAM 卡(Q3MEM-4MBS、Q3MEM-8MBS)时	位	源	0.180	0.162	0.162
			目标	0.160	0.143	0.143
		字	源	0.180	0.162	0.162
			目标	0.160	0.143	0.143
		双字	源	0.340	0.314	0.314
			目标	0.320	0.304	0.304
模块访问软元件(Un\G□、U3En\G0～G4095)		位	源	11.700	11.200	11.200
			目标	15.400	15.300	15.300
		字	源	9.460	9.410	9.410
			目标	19.000	19.000	19.000
		双字	源	11.000	10.900	10.900
			目标	18.800	18.700	18.700
链接直接软元件(Jn\□)		位	源	32.700	31.300	31.300
			目标	52.300	29.900	29.900
		字	源	28.500	17.300	17.300
			目标	27.500	14.700	14.700
		双字	源	30.300	18.100	18.100
			目标	30.600	15.700	15.700

附录 2　CPU 的性能比较

附录 2.1　QCPU 与 AnNCPU/AnACPU/AnUCPU 的比较

附录 2.1.1　可用的软元件

附表 2.1 软元件的比较表

软元件名		QCPU				AnUCPU	AnACPU	AnNCPU
输入输出点数 *9		Q00J：256 点 Q00：1024 点 Q01：1024 点	Q00UJ：256 点 Q00U：1024 点 Q01U：1024 点 Q02、Q02H、 Q06H、Q12H、 Q25H、Q02PH、 Q06PH、Q12PH、 Q25PH、Q12PRH、 Q03UD、Q04UDH、 Q06UDH、Q10UDH、 Q13UDH、Q20UDH、 Q26UDH、Q03UDE、 Q04UDEH、Q06UDEH、 Q10UDH、Q13UDEH、 Q26UDEH Q02U：2048 点	4096 点		A2U：512 点 A2U-S1：1024 点 A3U：2048 点 A4U：4096 点	A2A：512 点 A2A-S1：1024 点 A3A：2048 点	A1N：256 点 A2N：512 点 A2N-S1：1024 点 A3N：2048 点
输入输出软元件点数 *8		2048 点 *1	8192 点 *1			8192 点	与各 CPU 的输入输出相同	
内部继电器		8192 点 *1				合计 8192 点		合计 2048 点
锁存继电器		2048 点 *1	8192 点 *1					
步进继电器	顺控程序用	--						--
	SFC 用	2048 点 *6	8192 点			--		
报警器		1024 点 *1	2048 点 *1			2048 点		256 点
变址继电器		1024 点 *1	2048 点 *1			--		
链接继电器		2048 点 *1	8192 点 *1			8192 点	4096 点	1024 点
链接用特殊继电器		1024 点	2048 点			56 点		
定时器		512 点 *1	2048 点 *1			合计 2048 点		合计 256 点
累计定时器		0 点 *1						
计数器		512 点 *1	1024 点 *1			1024 点		256 点
数据寄存器		11136 点 *1	12288 点 *1			8192 点	6144 点	1024 点
链接寄存器		2048 点 *1	8192 点 *1			8192 点	4096 点	1024 点
链接用特殊寄存器		1024 点	2048 点			56 点		
功能输入		16 点 (FX0 ～ FXF) *7				--		
功能输出		16 点 (FY0 ～ FYF) *7						
特殊继电器		1000 点	2048 点			256 点		
功能寄存器		5 点 (FD0 ～ FD4)				--		
特殊寄存器		1000 点	2048 点			256 点		
链接直接软元件		通过 J □ \ □指定				--		
智能功能模块软元件		通过 U □ \G □指定						
变址寄存器	Z	10 点 (Z0 ～ Z9)	16 点 (Z0 ～ Z15)			7 点 (Z, Z1 ～ Z6)		1 点 (Z)
	V *2	--				7 点 (V, V1 ～ V6)		1 点 (V)
文件寄存器		32768 点 / 块 *5 (R0 ～ R32767)	32768 点 / 块 (R0 ～ R32767)			8192 点 / 块 (R0 ～ R8191)		
累加器 *3		--				2 点		
嵌套		15 点				8 点		
指针		300 点	4096 点			256 点		
中断指针		128 点	256 点		48 点	32 点		
SFC 块		126 *6	320 点			--		
SFC 移位软元件		--	512 点					
10 进制常数		K-2147483648 ～ K2147483647						
16 进制常数		H0 ～ HFFFFFFFF						
实数常数 *6		E ± 1.17550-38 ～ E ± 3.40282 + 38				--		
字符串		“QnACPU”, “ABCD” *4						

*1: 可以对参数中使用的点数进行变更。
*2: 在 QCPU 中，将 V 作为变址继电器使用。
*3: 在 AnNCPU/AnACPU/AnUCPU 中使用了累加器的指令，在 QCPU 中指令的格式将改变。
*4: 在 Q00JCPU、Q00CPU、Q01CPU 中只能通过 $MOV 指令使用。
*5: Q00JCPU 中没有文件寄存器。
*6: 序列号的前 5 位数为 “04122” 以后的 Q00JCPU、Q00CPU、Q01CPU 可以使用。
*7: 在程序中，只能使用 FX0 ～ FX4、FY0 ～ FY4 的各 5 点。
*8: 可用于程序的点数。
*9: 实际输入输出模块的可访问点数。

附录 2.1.2 I/O 控制方式

附表 2.2 I/O 控制方式

I/O 控制方式			QCPU	AnUCPU	AnACPU	AnNCPU
刷新方式			○	○	○	○ *2
	直接输入输出方式	部分刷新指令	○	○	○	○
		专用指令 *1	－	○	○	－
		直接访问输入	○	－	－	－
		直接访问输出	○	－	－	－
直接方式			－	－	－	○ *2

表中的符号…○： 可以使用； －： 不能使用

*1: 直接输出专用指令中，有 DOUT、DSET、SRST 指令。
没有直接输入专用指令。
*2: 刷新方式与直接方式的切换是通过 AnNCPU 的插杆开关进行。

附录 2.1.3 可用于指令的数据

附表 2.3 可用于指令的数据

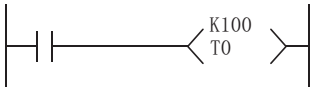
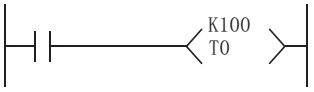
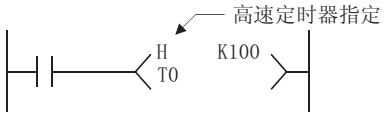
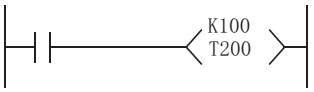
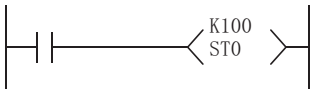
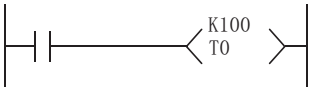
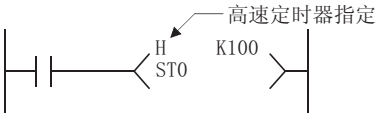
设置数据		QCPU	AnUCPU	AnACPU	AnNCPU
位数据	位软元件	○	○	○	○
	字软元件	○（需要位指定）	－	－	－
字数据	位软元件	○（需要位数指定）	○（需要位数指定）	○（需要位数指定）	○（需要位数指定）
	字软元件	○	○	○	○
双字数据	位软元件	○（需要位数指定）	○（需要位数指定）	○（需要位数指定）	○（需要位数指定）
	字软元件	○	○	○	○
实数数据		○ *2	○	○	△ *1
字符串数据		○ *3	－	－	－

表中的符号…○： 可以使用； △： 有条件使用； －： 不能使用

*1: SW0SRXV-FUN2 型软元件包的浮点实数类型用微机软件包登录时可以使用。
*2: 序列号的前 5 位数为 “04122” 以后的 Q00JCPU、Q00CPU、Q01CPU 可以使用。
*3: 在 Q00JCPU、Q00CPU、Q01CPU 中只能通过 $MOV 指令使用。

附录 2.1.4 定时器的比较

附表 2.4 定时器的比较

功能		QCPU	AnUCPU	AnACPU	AnNCPU
低速定时器	计量单位	100ms(默认值) 在参数中可对计量单位进行变更。 QCPU : 1 ～ 1000ms(1ms 单位) QnACPU: 10 ～ 1000ms(10ms 单位)	固定为 100ms		
	指定方法				
高速定时器	计量单位	10ms(默认值) 通过参数可对计量单位进行变更。 QnUCPU : 0.01 ～ 100ms(0.01ms 单位) QCPU(除 QnUCPU 以外) : 0.1 ～ 100ms(0.1ms 单位) QnACPU : 1 ～ 1000ms(1ms 单位)	固定为 10ms		
	指定方法	 ※高速定时器设置 : 通过顺控程序进行。	 ※高速定时器设置 : 通过参数进行。		
累计定时器	计量单位	与低速定时器的计量单位相同。	固定为 100ms		
	指定方法				
高速累计定时器	计量单位	与高速定时器的计量单位相同。	无		
	指定方法	 ※高速定时器设置 : 通过顺控程序进行。			
设置值的设置范围		1 ～ 32767	1 ～ 32767		
设置值 0 的处理		瞬时 ON	无限大(无时间到)		
变址修饰	触点	可以(仅 Z0、Z1 可以使用)	可以		不能
	线圈	可以(仅 Z0、Z1 可以使用)	不能		不能
	设置值	可以(Z0 ～ Z15 可以使用)*1	不能		不能
	当前值	可以(Z0 ～ Z15 可以使用)*1	可以		可以
当前值的更新处理		执行 OUT Tn 指令时	END 处理时		
触点的 ON/OFF 处理					

*1: Q00JCPU、Q00CPU、Q01CPU 可以使用 Z0 ～ Z9。
通用型 QCPU 可以使用 Z0 ～ Z19。

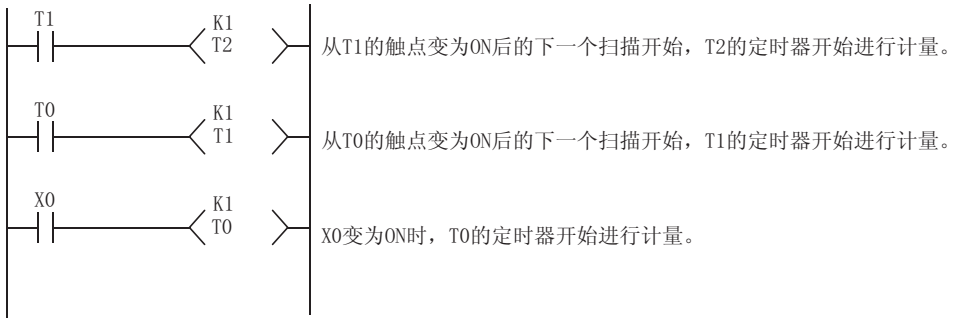
(1) 使用定时器时的注意事项

在 QCPU 中，执行 OUT T □指令时进行当前值的更新及触点的 ON/OFF 操作。
因此定时器的线圈变为 ON 时，如果(当前值)≥(设置值)，则该定时器的触点将 ON。
创建通过定时器的触点进行其它定时器的计量的程序时，应按从后计量的定时器开始的顺序进行编程。
在下述情况下，如果按计量顺序进行编程，所有的定时器将在同一个扫描中变为 ON。

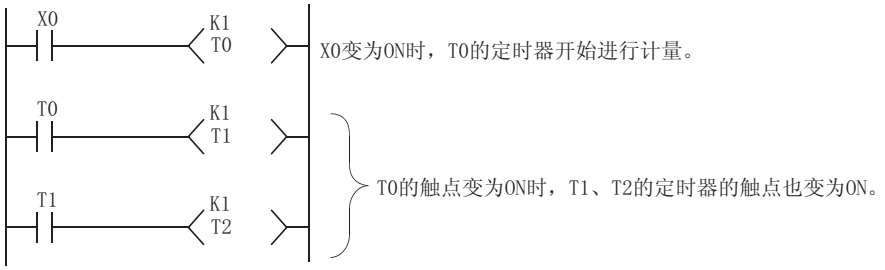
- 高速定时器中设置值短于扫描时间时
- 低速定时器中设置值为“1”时

例

• 将 T0 ～ T2 的定时器按照从后计量的定时器开始的顺序进行编程时

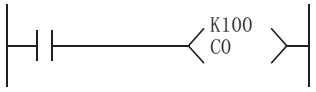
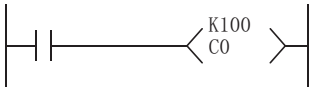


• 将 T0 ～ T2 的定时器按照计量顺序进行编程时



附录 2.1.5 计数器的比较

附表 2.5 计数器的比较

功能		QCPU	AnUCPU	AnACPU	AnNCPU
指定方法					
变址修饰	触点	• 可以（仅 Z0、Z1 可以使用）	• 可以		• 不能
	线圈	• 可以（仅 Z0、Z1 可以使用）	• 不能		• 不能
	设置值	• 不能	• 不能		• 不能
	当前值	• 可以（Z0 ～ Z15 可以使用）*1	• 可以		• 可以
当前值的更新处理 触点的 ON/OFF 处理		• 执行 OUT Cn 指令时	• END 处理时		

*1: Q00JCPU、Q00CPU、Q01CPU 可以使用 Z0 ～ Z9。
通用型 QCPU 可以使用 Z0 ～ Z19。

附录 2.1.6 显示指令的比较

附表 2.6 显示指令的比较

指令	QCPU	AnUCPU	AnACPU	AnNCPU
PR*1	• SM701 变为 OFF 时：输出直到 00H 为止 • SM701 变为 ON 时 ： 输出 16 字符	• M9049 变为 OFF 时：输出直到 00H 为止 • M9049 变为 ON 时 ： 输出 16 字符		
PRC*1	• SM701 变为 OFF 时：输出 32 字符的注释 • SM701 变为 ON 时 ： 输出高 16 位字符	输出 16 字符的注释		

*1: 在 Q00JCPU、Q00CPU、Q01CPU 中不能使用。

附录 2.1.7 指定格式被变更的指令 (AnACPU/AnUCPU 的专用指令除外)

由于在 QCPU 中没有累加器 (A0、A1)，因此对于 AnUCPU、AnACPU、AnNCPU 中使用了累加器的指令，其格式将被变更。

附表 2.7 表示方式被变更的指令

功能	QCPU		AnUCPU/AnACPU/AnNCPU	
	指令的格式	备注	指令的格式	备注
16 位右旋转	ROR D n	• D: 旋转数据	ROR n	• 旋转数据被设置到 A0 中。
	RCR D n	• D: 旋转数据 • 进位标志使用 SM700。	RCR n	• 旋转数据被设置到 A0 中。 • 进位标志使用 M9012。
16 位左旋转	ROL D n	• D: 旋转数据	ROL n	• 旋转数据被设置到 A0 中。
	RCL D n	• D: 旋转数据 • 进位标志使用 SM700。	RCL n	• 旋转数据被设置到 A0 中。 • 进位标志使用 M9012。
32 位右旋转	DROR D n	• D: 旋转数据	DROR n	• 旋转数据被设置到 A0、A1 中。
	DRCR D n	• D: 旋转数据 • 进位标志使用 SM700。	DRCR n	• 旋转数据被设置到 A0、A1 中。 • 进位标志使用 M9012。
32 位左旋转	DROL D n	• D: 旋转数据	DROL n	• 旋转数据被设置到 A0、A1 中。
	DRCL D n	• D: 旋转数据 • 进位标志使用 SM700。	DRCL n	• 旋转数据被设置到 A0、A1 中。 • 进位标志使用 M9012。
16 位数据搜索	SER S1 S2 D n	• 搜索结果被存储到 D、D+1 的软元件中。	SER S1 S2 n	• 搜索结果被存储到 A0、A1 中。
32 位数据搜索	DSER S1 S2 D n	• 搜索结果被存储到 D、D+1 的软元件中。	DSER S1 S2 n	• 搜索结果被存储到 A0、A1 中。
16 位数据位检查	SUM S D	• 检查结果被存储到 D 的软元件中。	SUM S	• 检查结果被存储到 A0 中。
32 位数据位检查	DSUM S D	• 检查结果被存储到 D 的软元件中。	DSUM S	• 检查结果被存储到 A0 中。
部分刷新	RFS D n	• 添加专用指令。	SEG D n	• 仅在 M9052 为 ON 时。
8 字符的 ASCII 码转换	$MOV (字符串) D	-	ASC (字符串) D	-
进位标志的设置	SET SM700	• 无专用指令。	STC	-
进位标志的复位	RST SM700	• 无专用指令。	CLC	-
至 END 指令的跳转	GOEND	• 添加专用指令。	CJ P255	• P255: END 指令指定
CHK 指令 *1	CHKST CHK	• 添加 CHKST 指令	CJ Pn P254 CHK	-

*1: 在 Q00J/Q00/Q01CPU 中不能使用。

附录 2.1.8 AnACPU/AnUCPU 的专用指令

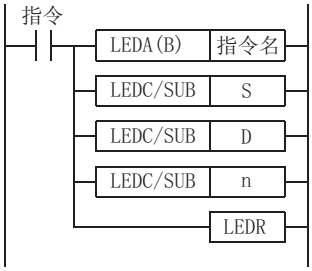
(1) 专用指令的表示方法

对于 QCPU，将 AnACPU/AnUCPU 中 LEDA、LEDB、LEDC、SUB、LEDR 指令等专用指令变更为与基本指令 / 应用指令相同的格式。

对于 QCPU/QnACPU 中由于无相应指令而无法转换的指令，将被转换为 OUT SM1255/OUT SM999(Q00J/Q00/Q01CPU 时)。

应将被转换为 OUT SM1255/OUT SM999 的指令用其它指令替换或者将其删除。

附表 2.8 专用指令的表示方法

QCPU	AnUCPU/AnACPU
	 ※：S、D、n 表示指令中使用的数据。

(2) 指令名被变更的专用指令

对于在 AnACPU/AnUCPU 的专用指令的指令名中，与 QCPU 的基本指令 / 应用指令具有相同名称的指令，在 QCPU 中其名称将被变更。

附表 2.9 指令名被变更的专用指令

功能	QCPU	AnUCPU/AnACPU
浮点数加法运算	E+	ADD
浮点数减法运算	E－	SUB
浮点数乘法运算	E＊	MUL
浮点数除法运算	E/	DIV
数据的分离	NDIS	DIS
数据的合并	NUNI	UNI
检查模式的变更	CHKCIR,CHKEND	CHK,CHKEND

附录 3　特殊继电器一览表

特殊继电器 SM 是可编程控制器内部的有固定规格的内部继电器。因此，不能象普通的内部继电器那样被用于顺控程序中。但是，根据需要，可以对其进行 ON/OFF 操作以控制 CPU 模块。

一览表的各项目的阅读方法如附表 3.1 所示。

附表 3.1 特殊继电器一览表的阅读方法

项目	项目说明
编号	• 表示特殊继电器的编号。
名称	• 表示特殊继电器的名称。
内容	• 表示特殊继电器的有关内容。
详细内容	• 说明特殊继电器的详细内容。
设置方 （设置时机）	• 说明设置方以及由系统设置时的设置时机。 〈设置方〉 S　　：由系统设置。 U　　：由用户（通过顺控程序或者外围设备的测试操作）进行设置。 S/U　：系统 / 用户均进行设置。 〈设置时机〉 每次 END　　：在每次的 END 处理时进行设置。 初始化　　：仅在初始化（电源 ON、STOP → RUN 等）时进行设置。 状态变化　　：仅在状态发生了变化时进行设置。 发生出错　　：仅在发生出错时进行设置。 执行指令　　：仅在执行指令时进行设置。 请求时　　：仅在有用户请求时（通过 SM 等）进行设置。 系统切换时　：在执行了系统切换时进行设置。
对应 ACPU M9□□□	• 表示对应于 ACPU 的特殊继电器 (M9□□□)。 （内容有变更时，通过 M9□□□的状态变化来表示。不对应于 Q00J/Q00/Q01、QnPRH。） • 标为新增时，表示在 Q 系列 CPU 模块中新增的内容。
对应 CPU	表示对应的 CPU 模块。 QCPU　　：对应于所有的 Q 系列 CPU 模块。 Q00J/Q00/Q01　：对应于基本型 QCPU。 Qn(H)　　：对应于高性能型 QCPU。 QnPH　　：对应于过程 CPU。 QnPRH　　：对应于冗余 CPU。 QnU　　：对应于通用型 QCPU。 各 CPU 模块型号：仅对应于所记述的 CPU 模块。（例：Q02U）

下述项目的详细内容请参阅以下手册。

• 网络相关　→ 各网络模块的手册

• SFC 相关　→ QCPU(Q 模式)/QnACPU 编程手册 (SFC 篇)

要 点

对于由系统设置的特殊继电器，不要通过用户程序或软元件测试等操作进行变更。否则有可能导致系统宕机或者无法通信。

(1) 诊断信息

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU M9□□□	对应 CPU
SM0	诊断出错	OFF ：无出错 ON ：有出错	• 诊断的结果。产生了出错时将变为 ON。（报警器的 ON、通过 CHK 指令检测出出错时也包括在内。） • 此后即使变为正常时仍将保持为 ON 状态不变。	S（发生出错）	新增	Qn(H) QnPH QnPRH
			• 诊断的结果。产生了出错时将变为 ON。（报警器 ON 时也包括在内。） • 此后即使变为正常时仍将保持为 ON 状态不变。	S（发生出错）	新增	Q00J/Q00/Q01 QnU
SM1	自诊断出错	OFF ：无自诊断出错 ON ：有自诊断出错	• 自诊断的结果。产生了出错时将变为 ON。（报警器的 ON、通过 CHK 指令检测出出错时不包括在内。） • 此后即使变为正常时仍将保持为 ON 状态不变。	S（发生出错）	M9008	Qn(H) QnPH QnPRH
			• 自诊断的结果。产生了出错时将变为 ON。（报警器 ON 时不包括在内。） • 此后即使变为正常时仍将保持为 ON 状态不变。	S（发生出错）	新增	Q00J/Q00/Q01 QnU
SM5	出错公共信息	OFF ：无出错公共信息 ON ：有出错公共信息	• SM0 变为 ON 时，如果有出错公共信息则置为 ON。	S（发生出错）	新增	QCPU
SM16	出错个别信息	OFF ：无出错个别信息 ON ：有出错个别信息	• SM0 变为 ON 时，如果有出错个别信息则置为 ON。	S（发生出错）	新增	
SM50	出错解除	OFF → ON：出错解除	• 执行出错解除动作。	U	新增	
SM51	电池过低锁存	OFF ：正常 ON ：电池电压过低	• CPU 模块、存储卡的电池电压低于规定值时置为 ON。 • 此后即使电池电压变为正常时 SM51 也仍将保持为 ON 状态不变。 • 与 BAT.LED 同步。	S（发生出错）	M9007	Qn(H) QnPH QnPRH QnU
			• CPU 模块的电池电压低于规定值时置为 ON。 • 此后即使电池电压变为正常时 SM51 也仍将保持为 ON 状态不变。 • 与ERR.LED 同步。	S（发生出错）	新增	Q00J/Q00/Q01
SM52	电池过低	OFF ：正常 ON ：电池电压过低	• 与 SM51 相同，但此后如果电池电压变为正常时 SM52 将变为 OFF。	S（发生出错）	M9006	QCPU
SM53	检测出 AC/DC DOWN	OFF ：无 AC/DC DOWN ON ：有 AC/DC DOWN	• 使用 AC 电源模块的情况下，如果有 20ms 以内的瞬停时将置为 ON。 通过电源的 OFF → ON 被复位。	S（发生出错）	M9005	
			• 使用DC 电源模块的情况下，如果有 10ms 以内的瞬停时将置为 ON。 通过电源的 OFF → ON 被复位。			
SM56	运算出错	OFF ：正常 ON ：有运算出错	• 发生了运算出错时 SM56 将置为 ON。 • 此后即使变为正常时 SM56 也仍将保持为 ON 状态不变。	S（发生出错）	M9011	
SM60	检测出保险丝熔断	OFF ：正常 ON ：有保险丝熔断的模块	• 如果有 1 个变为保险丝熔断状态的输出模块则 SM60 将置为 ON。 • 此后即使电池电压变为正常时 SM60 也仍将保持为 ON 状态不变。 • 对远程 I/O 站的输出模块也将进行保险丝熔断状态的检查。	S（发生出错）	M9000	
SM61	输入输出模块校验出错	OFF ：正常 ON ：有出错	• 在电源接通时如果输入输出模块与登录状态不相同的情况下 SM61 将置为 ON。 • 此后即使变为正常时 SM61 也仍将保持为 ON 状态不变。 • 对远程 I/O 站的模块也进行输入输出模块校验。	S（发生出错）	M9002	
SM62	检测出报警器	OFF ：未检测出 ON ：检测出	• 只要有 1 个报警器 (F) 报警则 SM62 将置为 ON。	S（指令执行）	M9009	
SM80	CHK 检测	OFF ：未检测出 ON ：检测出	• 通过 CHK 指令检测出异常时 SM80 将置为 ON。 • 此后即使变为正常时 SM80 也仍将保持为 ON 状态不变。	S（指令执行）	新增	Qn(H) QnPH QnPRH

编号	名称	内容	详细内容		设置方（设置时机）	对应 ACPU M9□□□	对应 CPU
SM90	步移位监视定时器启动（仅在有 SFC 程序时有效）	OFF ：未启动状态（监视定时器复位） ON ：启动状态（监视定时器启动）	对应于 SD90	• 开始步移位监视定时器的计量时置为 ON。 • OFF 时进行步移位监视定时器的复位。	U	M9108	Qn(H) QnPH QnPRH
SM91			对应于 SD91			M9109	
SM92			对应于 SD92			M9110	
SM93			对应于 SD93			M9111	
SM94			对应于 SD94			M9112	
SM95			对应于 SD95			M9113	
SM96			对应于 SD96			M9114	
SM97			对应于 SD97			新增	
SM98			对应于 SD98			新增	
SM99			对应于 SD99			新增	
SM100	串行通信功能使用标志	OFF ：未使用串行通信功能 ON ：使用串行通信功能	• 存储串行通信设置参数的串行通信功能的使用 / 不使用。		S（电源 ON、复位时）	新增	Q00/Q01 Q00UJ Q00U Q01U Q02U*7
SM101	通讯协议状态标志	OFF ：GX Developer ON ：MC 协议通讯软设备	• 存储通过 RS-232 接口通信的设备是 GX Developer 还是 MC 协议通讯软设备。		S（RS-232 通信时）		
SM110	协议异常	OFF ：正常 ON ：异常	• 在串行通信功能中通过异常的协议进行通信时，SM110 将 ON。 • 此后，即使变为正常也将保持为 ON 状态不变。		S（发生出错）		
SM111	通信状态	OFF ：正常 ON ：异常	• 在串行通信功能中通过与设置不相同的模式进行通信时，SM111 将 ON。 • 此后，即使变为正常也将保持为 ON 状态不变。		S（发生出错）		
SM112	出错信息清除	通过 ON 进行清除	• 对 SM110、SM111 以及 SD110、SD111 中存储的出错代码进行清除时置为 ON。 （通过 OFF → ON 执行动作）		U		
SM113	溢出出错	OFF ：正常 ON ：异常	• 在串行通讯出错中发生溢出出错时，变为 ON。		S（发生出错）		
SM114	奇偶性出错	OFF ：正常 ON ：异常	• 在串行通讯出错中发生奇偶性出错时，变为 ON。		S（发生出错）		
SM115	结构出错	OFF ：正常 ON ：异常	• 在串行通讯出错中发生结构出错时，变为 ON。		S（发生出错）		
SM165	程序存储器批量传送执行状态	OFF ：结束 ON ：未执行或者未结束	• 程序高速缓冲存储器写入时变为 ON。 • 程序批量传送结束时变为 OFF。 • 写入到程序存储器中后，不进行程序批量传送时，保持 ON 状态。		S（状态变化）	新增	QnU*6

*6: 以下述模块为对象。
- 序列号的前 5 位数为 “10012” 以后的通用型 QCPU
- Q13UDHCPU、Q26UDHCPU

*7: 以序列号的前 5 位数为 “10012” 以后的模块为对象。

(2) 系统信息

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU M9□□□	对应 CPU
SM202	LED 熄灯指令	OFF → ON：LED 熄灯	• 本继电器从 OFF 变为 ON 时，SD202 的各位中对应的 LED 将熄灯。	U	新增	Qn(H) QnPH QnPRH QnU
SM203	STOP 触点	STOP 状态	• STOP 状态时，变为 ON。	S（状态变化）	M9042	QCPU
SM204	PAUSE 触点	PAUSE 状态	• PAUSE 状态时，变为 ON。	S（状态变化）	M9041	
SM206	PAUSE 许可线圈	OFF ：禁止 PAUSE ON ：允许 PAUSE	• PAUSE 触点变为 ON 时，本继电器如果变为 ON 则进入 PAUSE 状态。	U	M9040	
SM210	时钟数据设置请求	OFF ：无处理 ON ：有设置请求	• 本继电器由 OFF 变为 ON 的扫描的 END 指令执行后，将 SDSD210 ～ SD213 中存储的时钟数据写入到 CPU 模块中。	U	M9025	
SM211	时钟数据出错	OFF ：无出错 ON ：有出错	• 时钟数据（SD210 ～ SD213）的值发生了出错时变为 ON，无出错时变为 OFF。	S（请求时）	M9026	
SM213	时钟数据读取请求	OFF ：无处理 ON ：读取请求	• 本继电器变为 ON 时将时钟数据以 BCD 值读取到 SD210 ～ SD213 中。	U	M9028	
SM220	1 号机准备完毕	OFF ：1 号机准备未完毕 ON ：1 号机准备完毕	• 接通电源时或者复位操作时，可以从其它机号 CPU 模块访问 1 号机 CPU 模块时变为 ON。在多 CPU 同步设置中设置为非同步的情况下，本 SM 被作为访问 1 号 CPU 模块的互锁使用。	S（状态变化时）	新增	QnU
SM221	2 号机准备完毕	OFF ：2 号机准备未完毕 ON ：2 号机准备完毕	• 接通电源时或者复位操作时，可以从其它机号 CPU 模块访问 2 号机 CPU 模块时变为 ON。在多 CPU 同步设置中设置为非同步的情况下，本 SM 被作为访问 2 号 CPU 模块的互锁使用。			QnU*8
SM222	3 号机准备完毕	OFF ：3 号机准备未完毕 ON ：3 号机准备完毕	• 接通电源时或者复位操作时，可以从其它机号 CPU 模块访问 3 号机 CPU 模块时变为 ON。在多 CPU 同步设置中设置为非同步的情况下，本 SM 被作为访问 3 号 CPU 模块的互锁使用。			
SM223	4 号机准备完毕	OFF ：4 号机准备未完毕 ON ：4 号机准备完毕	• 接通电源时或者复位操作时，可以从其它机号 CPU 模块访问 4 号机 CPU 模块时变为 ON。在多 CPU 同步设置中设置为非同步的情况下，本 SM 被作为访问 4 号 CPU 模块的互锁使用。			QnU*5

*5: 以除 Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU 以外的通用型 QCPU 为对象。
*8: 以除 Q00UJCPU 以外的通用型 QCPU 为对象。

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU M9□□□	对应 CPU
SM235	在线模块更换标志	OFF ：未进行在线模块更换 ON ：正在进行在线模块更换	• 正在进行在线模块更换时该标志将变为 ON。（用于本机）	S（实施在线模块更换时）	新增	QnPH
SM236	在线模块更换后仅 1 个扫描 ON 标志	OFF ：在线模块更换未结束 ON ：在线模块更换结束	• 在线模块更换结束后，仅 1 个扫描周期 ON。 • 本触点只能在扫描程序中使用。（用于本机）	S （模块更换结束时）	新增	
SM237	软元件范围检查禁止标志	OFF ：进行软元件范围检查。 ON ：不进行软元件范围检查。	• 选择在 BMOV、FMOV、DFMOV 指令（仅在子集条件成立时）中是否进行软元件范围检查。	U	新增	QnU*6
SM240	1 号机复位标志	OFF ：1 号机复位解除 ON ：1 号机复位中	• 1 号机 CPU 模块处于复位解除状态时该标志变为 OFF。 • 1 号机 CPU 模块处于复位（包括将 CPU 模块从主基板上卸下时。）时该标志变为 ON。 其它机号 CPU 也变为复位状态。	S（状态变化时）	新增	Q00/Q01*1 Qn(H)*1 QnPH QnU*8
SM241	2 号机复位标志	OFF ：2 号机复位解除 ON ：2 号机复位中	• 2 号机 CPU 模块处于复位解除状态时该标志变为 OFF。 • 2 号机 CPU 模块处于复位（包括将 CPU 模块从主基板上卸下时。）时该标志变为 ON。 其它机号 CPU 变为“MULTI CPU DOWN”（出错代码：7000）状态。			
SM242	3 号机复位标志	OFF ：3 号机复位解除 ON ：3 号机复位中	• 3 号机 CPU 模块处于复位解除状态时该标志变为 OFF。 • 3 号机 CPU 模块处于复位（包括将 CPU 模块从主基板上卸下时。）时该标志变为 ON。 其它机号 CPU 变为“MULTI CPU DOWN”（出错代码：7000）状态。			
SM243	4 号机复位标志	OFF ：4 号机复位解除 ON ：4 号机复位中	• 4 号机 CPU 模块处于复位解除状态时该标志变为 OFF。 • 4 号机 CPU 模块处于复位（包括将 CPU 模块从主基板上卸下时。）时该标志变为 ON。 其它机号 CPU 变为“MULTI CPU DOWN”（出错代码：7000）状态。			Qn(H)*1 QnPH QnU*5

*1: 以功能版本 B 以后为对象。
*5: 以除 Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU 以外的通用型 QCPU 为对象。
*6: 以下述模块为对象。
- 序列号的前 5 位数为“10012”以后的通用型 QCPU
- Q13UDHCPU、Q26UDHCPU

*8: 以除 Q00UJCPU 以外的通用型 QCPU 为对象。

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU M9□□□	对应 CPU
SM244	1 号机出错标志	OFF ：1 号机正常 ON ：1 号机停止出错中	• 1 号机 CPU 模块正常（也包括继续运行出错。）时该标志变为 OFF。 • 1 号机 CPU 模块处于停止出错中状态时该标志将变为 ON。	S（状态变化时）	新增	Q00/Q01*1 Qn(H)*1 QnPH QnU*8
SM245	2 号机出错标志	OFF ：2 号机正常 ON ：2 号机停止出错中	• 2 号机 CPU 模块正常（也包括继续运行出错。）时该标志变为 OFF。 • 2 号机 CPU 模块处于停止出错中状态时该标志将变为 ON。			
SM246	3 号机出错标志	OFF ：3 号机正常 ON ：3 号机停止出错中	• 3 号机 CPU 模块正常（也包括继续运行出错。）时该标志变为 OFF。 • 3 号机 CPU 模块处于停止出错中状态时该标志将变为 ON。			
SM247	4 号机出错标志	OFF ：4 号机正常 ON ：4 号机停止出错中	• 4 号机 CPU 模块正常（也包括继续运行出错。）时该标志变为 OFF。 • 4 号机 CPU 模块处于停止出错中状态时该标志将变为 ON。			Qn(H)*1 QnPH QnU*5
SM250	实际安装最大 I/O 读取	OFF ：无处理 ON ：读取	• 本继电器由 OFF 变为 ON 时将实际安装最大输入输出编号读取到 SD250 中。	U	新增	Qn(H) QnPH QnPRH
SM254	所有站刷新指令	OFF ：到达站刷新 ON ：所有站刷新	• 批量刷新时有效。（低速循环中也有效） • 在 MELSECNET/H 中，指定是仅到达站接收，还是所有站接收。	U	新增	Qn(H) QnPH QnPRH
			• 在 CC-Link IE 控制网络中，指定是仅到达站接收，还是所有站接收。			Qn(H)*2 QnPH*6 QnPRH*6
			• 批量刷新时有效。（低速循环中也有效） • 在 MELSECNET/H 中或者 CC-Link IE 控制网络中，指定是仅到达站接收，还是所有站接收。			QnU

*1: 以功能版本 B 以后为对象。
*2: 以序列号的前 5 位数为“09012”以后的模块为对象。
*5: 以除 Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU 以外的通用型 QCPU 为对象。
*6: 以序列号的前 5 位数为“10042”以后的模块为对象。
*8: 以除 Q00UJCPU 以外的通用型 QCPU 为对象。

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU M9□□□	对应 CPU
SM255	第 1 个 MELSECNET/10、MELSECNET/H 信息	OFF ：运行网络 ON ：待机网络	• 待机网络时变为 ON。 （未进行运行、待机的指定时变为运行。）	S（初始化）	新增	Qn(H) QnPH QnPRH
SM256		OFF ：进行读取 ON ：不进行读取	• 用于链接→ CPU 模块方向的刷新（B、W 等），指定是否从链接模块读取。	U	新增	
SM257		OFF ：进行写入 ON ：不进行写入	• 用于 CPU 模块→链接方向的刷新（B、W 等），指定是否对链接模块进行写入。	U	新增	
SM260	第 2 个 MELSECNET/10、MELSECNET/H 信息	OFF ：运行网络 ON ：待机网络	• 待机网络时变为 ON。 （未进行运行、待机的指定时变为运行。）	S（初始化）	新增	
SM261		OFF ：进行读取 ON ：不进行读取	• 用于链接→ CPU 模块方向的刷新（B、W 等），指定是否从链接模块读取。	U	新增	
SM262		OFF ：进行写入 ON ：不进行写入	• 用于 CPU 模块→链接方向的刷新（B、W 等），指定是否对链接模块进行写入。	U	新增	
SM265	第 3 个 MELSECNET/10、MELSECNET/H 信息	OFF ：运行网络 ON ：待机网络	• 待机网络时变为 ON。 （未进行运行、待机的指定时变为运行。）	S（初始化）	新增	
SM266		OFF ：进行读取 ON ：不进行读取	• 用于链接→ CPU 模块方向的刷新（B、W 等），指定是否从链接模块读取。	U	新增	
SM267		OFF ：进行写入 ON ：不进行写入	• 用于 CPU 模块→链接方向的刷新（B、W 等），指定是否对链接模块进行写入。	U	新增	
SM270	第 4 个 MELSECNET/10、MELSECNET/H 信息	OFF ：运行网络 ON ：待机网络	• 待机网络时变为 ON。 （未进行运行、待机的指定时变为运行。）	S（初始化）	新增	
SM271		OFF ：进行读取 ON ：不进行读取	• 用于链接→ CPU 模块方向的刷新（B、W 等），指定是否从链接模块读取。	U	新增	
SM272		OFF ：进行写入 ON ：不进行写入	• 用于 CPU 模块→链接方向的刷新（B、W 等），指定是否对链接模块进行写入。	U	新增	

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU M9□□□	对应 CPU
SM280	CC-Link 出错	OFF ：正常 ON ：异常	• 在安装的 CC-Link 模块中，检测出某个 CC-Link 模块异常时将变为 ON。此后，恢复正常时将变为 OFF。	S（状态变化）	新增	Qn(H) QnPH QnPRH
SM315	通讯预留时间的时间等待有效 / 无效标志	OFF ：不进行时间等待 ON ：进行时间等待	• 当通讯处理预留时间被设定到 SD315 中时，此标志被激活。 • 希望在 END 处理中进行时间等待（在 SD315 中为执行通讯处理而设置的时间）时将该标志置为 ON。（扫描时间将延迟相当于在 SD315 中设置的时间。） • 当无通讯处理时，不希望在 END 处理中进行时间等待（在 SD315 中为执行通讯处理而设置的时间）时将该标志置为 OFF。（默认为 OFF）	U	新增	Q00J/Q00/Q01
SM320	SFC 程序的有无	OFF ：无 SFC 程序 ON ：有 SFC 程序	• 登录了 SFC 程序时变为 ON。 • 未登录 SFC 程序时变为 OFF。	S（初始化）	M9100	Q00J/Q00/Q01*1 Qn(H) QnPH QnPRH QnU
SM321	SFC 程序的启动 / 停止	OFF ：不执行 SFC 程序（停止） ON ：执行 SFC 程序（启动）	• 初始值被设置为与 SM320 相同的值。（如果存在有 SFC 程序则自动变为 ON。） • 通过将本继电器从 ON 变为 OFF 以停止 SFC 程序的执行。 • 通过将本继电器从 OFF 变为 ON 以恢复 SFC 程序的执行。	S（初始化）/U	M9101 状态变化	
SM322	SFC 程序的启动状态	OFF ：初始化启动 ON ：热启动	• 初始值被设置为可编程控制器参数的 SFC 设置的 SFC 程序启动模式。 初始化启动时 ：ON 热启动时 ：OFF	S（初始化）/U	M9102 状态变化	
SM323	所有块连续移位的有无	OFF ：无连续移位 ON ：有连续移位	对于未设置 SFC 信息软元件的“连续移位位”的块，设置是否进行连续移位。	U	M9103	
SM324	连续移位阻止标志	OFF ：执行移位时 ON ：未执行移位时	• 在有连续移位模式下动作或者正在进行连续移位时变为 OFF，在未进行连续移位时变为 ON。 • 在无连续移位模式下动作的情况下将为常时 ON。	S（指令执行）	M9104	
				S（状态变化）	新增	

*1： 以功能版本 B 以后为对象。

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU M9□□□	对应 CPU
SM325	块停止时的输出模式	OFF ：OFF ON ：保持	选择块停止时是否保持激活步的线圈输出。 ・参数的块停止时的输出模式的初始值为线圈输出 OFF 时为 OFF，线圈输出保持时为 ON。 ・本继电器为 OFF 时线圈输出全部为 OFF。 ・本继电器为 ON 时保持线圈输出。	S（初始化）/U	M9196	Q00J/Q00/Q01*1 Qn(H) QnPH QnPRH QnU
SM326	SFC 的软元件清除模式	OFF ：软元件清除 ON ：软元件保持	对 STOP →程序写入→ RUN 时的软元件的状态进行选择。（除步进继电器以外的所有软元件）	U	新增	Q00J/Q00/Q01*1 Qn(H) QnPH QnPRH QnU
SM327	执行 END 步时的输出	OFF ：保持步的输出 OFF ON ：保持步的输出保持	本继电器为 OFF 时，对于由于移位成立而处于保持中的步（SC、SE、ST），在到达 END 步时将线圈输出置为 OFF。	S（初始化）/U	新增	Qn(H) QnPH QnPRH QnU
				U		Q00J/Q00/Q01*1
SM328	到达 END 步时清除处理模式	OFF ：进行清除处理 ON ：不进行清除处理	到达 END 步时，在块内存在有除保持中以外的激活步的情况下，选择是否进行清除处理。 ・本继电器为 OFF 时，强制结束所有激活步，结束块的执行。 ・本继电器为 ON 时，在保持原样不变的状态下继续块的执行。 ・到达 END 步时不存在有除保持中以外的激活步的情况下，将所有保持中的步结束，结束块的执行。	U	新增	Q00J/Q00/Q01*1 QnU
SM330	低速执行型程序的动作方式	OFF ：非同步方式 ON ：同步方式	选择对低速执行型程序是以非同步方式执行还是以同步方式执行。 ・非同步方式（将本继电器置为 OFF。） 是剩余时间内继续执行低速执行型程序的运算的方式。 ・同步方式（将本继电器置为 ON。） 即使存在有剩余时间也不继续执行低速执行型程序的运算，从下一个扫描开始执行运算的方式。	U	新增	Qn(H) QnPH
SM331	普通 SFC 程序执行状态	OFF ：未执行 ON ：执行中	・表示普通 SFC 程序是否处于执行状态。 ・作为 SFC 控制指令的执行互锁使用。	S（状态变化）	新增	Qn(H)*3 QnPH*4 QnPRH
SM332	程序执行管理用 SFC 程序执行状态	OFF ：未执行 ON ：执行中	・表示程序执行管理用 SFC 程序是否处于执行状态。 ・作为 SFC 控制指令的执行互锁使用。			
SM390	访问执行标志	通过将该标志置为 ON 结束智能功能模块的访问	・存储之前执行的智能功能模块访问指令的状态。 （如果再次执行智能智能功能模块访问指令则该信息将被覆盖。） ・用户在程序中将其置为结束位使用。	S（状态变化）	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。 ・在指令执行前为 OFF。 ・指令结束后执行为 ON。	S（指令执行）	新增	QnU

*1: 以功能版本 B 以后为对象。
*3: 以序列号的前 5 位数为“04122”以后的模块为对象。
*4: 以序列号的前 5 位数为“07032”以后的模块为对象。

(3) 系统时钟 / 计数器

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU M9□□□	对应 CPU
SM400	常时 ON	ON OFF	• 置为常时 ON。	S（每次 END）	M9036	QCPU
SM401	常时 OFF	ON OFF	• 置为常时 OFF。	S（每次 END）	M9037	
SM402	RUN 后仅 1 个扫描 ON	ON OFF 1个扫描	• RUN 后仅 1 个扫描 ON。 • 本触点只能被用于扫描执行型程序。 • 使用初始化执行型程序时，在 RUN 后第 1 个扫描的扫描执行型程序的 END 处理中将 SM402 置为 OFF。 ON OFF 初始化执行型程序 扫描执行型程序的1个扫描	S（每次 END）	M9038	Qn(H) QnPH QnPRH QnU
			• RUN 后仅 1 个扫描 ON。	S（每次 END）	新增	Q00J/Q00/Q01
SM403	RUN 后仅 1 个扫描 OFF。	ON OFF 1个扫描	• RUN 后仅 1 个扫描 OFF。 • 本触点只能被用于扫描执行型程序。 • 使用初始化执行型程序时，在 RUN 后第 1 个扫描的扫描执行型程序的 END 处理中将 SM403 置为 ON。 ON OFF 初始化执行型程序 扫描执行型程序的1个扫描	S（每次 END）	M9039	Qn(H) QnPH QnPRH QnU
			• RUN 后仅 1 个扫描 OFF。	S（每次 END）	新增	Q00J/Q00/Q01
SM404	低速执行型程序 RUN 后仅 1 个扫描 ON	ON OFF 1个扫描	• RUN 后仅 1 个扫描 ON。 • 本触点只能被用于低速执行型程序。	S（每次 END）	新增	Qn(H) QnPH
SM405	低速执行型程序 RUN 后仅 1 个扫描 OFF	ON OFF 1个扫描	• RUN 后仅 1 个扫描 OFF。 • 本触点只能被用于低速执行型程序。	S（每次 END）	新增	
SM409	0.01 秒时钟	0.005秒 0.005秒	• 每隔 5ms 重复进行 ON/OFF。 • 进行了可编程控制器的电源 ON 或者 CPU 模块的复位时将该时钟从 OFF 状态置为启动状态。 （即使是在程序的执行过程中到达了指定时间，ON-OFF 状态也将发生变化，应加以注意。）	S（状态变化）	新增	Qn(H) QnPH QnPRH QnU
SM410	0.1 秒时钟	0.05秒 0.05秒	• 是每隔一定时间重复进行 ON/OFF 的继电器。 • 进行了可编程控制器的电源 ON 或者 CPU 模块的复位时将该时钟从 OFF 状态置为启动状态。 （即使是在程序的执行过程中到达了指定时间，ON-OFF 状态也将发生变化，应加以注意。）	S（状态变化）	M9030	QCPU
SM411	0.2 秒时钟	0.1秒 0.1秒			M9031	
SM412	1 秒时钟	0.5秒 0.5秒			M9032	
SM413	2 秒时钟	1秒 1秒			M9033	
SM414	2n 秒时钟	n秒 n秒	• 是每隔 SD414 中指定的时间（单位：秒）重复进行 ON/OFF 的继电器。 • 进行了可编程控制器的电源 ON 或者 CPU 模块的复位时将该时钟从 OFF 状态置为启动状态。 （即使是在程序的执行过程中到达了指定时间，ON-OFF 状态也将发生变化，应加以注意。）	S（状态变化）	M9034 状态变化	

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU M9□□□	对应 CPU
SM415	2n(ms)时钟	n(ms) n(ms)	• 是每隔 SD415 中指定的时间（单位：ms）重复进行 ON/OFF 的继电器。 • 进行了可编程控制器的电源 ON 或者 CPU 模块的复位时将该时钟从 OFF 状态置为启动状态。 （即使是在程序的执行过程中到达了指定时间，ON-OFF 状态也将发生变化，应加以注意。）	S（状态变化）	新增	Qn(H) QnPH QnPRH QnU
SM420	0 号用户定时时钟	n2扫描 n2扫描 n1扫描	• 是以指定扫描间隔重复进行 ON/OFF 的继电器。 • 进行了可编程控制器的电源 ON 或者 CPU 模块的复位时将该时钟从 OFF 状态置为启动状态。 （但是，冗余 CPU 的情况下在系统切换后变为常时 OFF。） • 设置根据 DUTY 指令而 ON/OFF 的扫描间隔。 DUTY | n1 | n2 | SM420 n1：ON 的扫描间隔 n2：OFF 的扫描间隔	S（每次 END）	M9020	QCPU
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				M9024	
SM430	5 号用户定时时钟		• 用于 SM420 ～ SM424 的低速执行型程序。	S（每次 END）	新增	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) 扫描信息

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU M9□□□	对应 CPU
SM510	低速执行型程序执行标志	OFF ：结束或者未执行 ON ：执行中	• 在低速执行型程序的执行过程中变为 ON。	S（每次 END）	新增	Qn(H) QnPH
SM551	模块服务间隔读取	OFF ：无处理 ON ：读取	• 本继电器由 OFF 变为 ON 时将 SD550 中指定的模块的服务间隔读取到 SD551 ～ 552 中。	U	新增	Qn(H) QnPH QnPRH

(5) I/O 刷新

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU M9□□□	对应 CPU
SM580	程序间 I/O 刷新	OFF ：不进行刷新 ON ：进行刷新	• 如果将本继电器置为 ON，则在执行第一个程序后进行 I/O 刷新，然后执行下一个程序。执行顺控程序及 SFC 程序时，执行顺控程序后，进行 I/O 刷新，然后执行 SFC 程序。	U	新增	Q00J/Q00/Q01*1

*1：以功能版本 B 以后为对象。

(6) 驱动器信息

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU M9□□□	对应 CPU
SM600	存储卡使用允许标志	OFF ：不能使用 ON ：可以使用	• 处于用户可以使用存储卡的状态时，该标志变为 ON。	S（状态变化）	新增	Qn(H) QnPH QnPRH QnU*1
SM601	存储卡保护标志	OFF ：无保护 ON ：有保护	• 存储卡的保护开关为 ON 时，该标志变为 ON。	S（状态变化）	新增	
SM602	驱动器 1 标志	OFF ：无驱动器 1 ON ：有驱动器 1	• 安装的存储卡为 RAM 时，该标志变为 ON。	S（状态变化）	新增	
SM603	驱动器 2 标志	OFF ：无驱动器 2 ON ：有驱动器 2	• 安装的存储卡为 ROM 时，该标志变为 ON。	S（状态变化）	新增	
SM604	存储卡使用中标志	OFF ：未使用 ON ：使用中	• 存储卡处于使用中时，该标志变为 ON。	S（状态变化）	新增	
SM605	存储卡拆卸禁止标志	OFF ：允许拆卸 ON ：禁止拆卸	• 处于禁止拆卸存储卡状态时，该标志变为 ON。	U	新增	
SM609	存储卡拆卸允许标志	OFF ：禁止拆卸 ON ：允许拆卸	• 置为允许拆卸存储卡状态时，由用户将该标志置为 ON。 • 拆下存储卡后由系统将其置为 OFF。 • 本触点仅在 SM604、SM605 处于 OFF 状态时可以使用。	S/U	新增	Qn(H) QnPH QnPRH QnU*1
SM620	驱动器 3/4 使用允许标志	OFF ：不能使用 ON ：可以使用	• 常时 ON。	S（初始化）	新增	QCPU
SM622	驱动器 3 标志	OFF ：无驱动器 3 ON ：有驱动器 3	• 常时 ON。	S（初始化）	新增	Q00J/Q00/Q01 Qn(H) QnPH QnPRH QnU*2
SM623	驱动器 4 标志	OFF ：无驱动器 4 ON ：有驱动器 4	• 常时 ON。	S（初始化）	新增	QCPU
SM624	驱动器 3、4 使用中标志	OFF ：未使用 ON ：使用中	• 使用驱动器 3（标准 RAM）、驱动器 4（标准 ROM）中存在的文件的情况下该标志变为 ON。	S（状态变化）	新增	Qn(H) QnPH QnPRH QnU
SM640	文件寄存器使用	OFF ：未使用文件寄存器 ON ：文件寄存器使用中	• 在使用文件寄存器时该标志变为 ON。	S（状态变化）	新增	Q00J/Q00/Q01 Qn(H) QnPH QnPRH QnU*2
SM650	注释使用	OFF ：未使用注释 ON ：注释使用中	• 在使用注释文件时该标志变为 ON。	S（状态变化）	新增	Qn(H) QnPH QnPRH QnU
SM660	引导运行	OFF ：内置存储器执行 ON ：引导运行中	• 在引导运行中时该标志变为 ON。 • 将引导指定开关置为 OFF 时，该继电器变为 OFF。	S（状态变化）	新增	Qn(H) QnPH QnPRH
		OFF ：程序存储器执行 ON ：引导运行中	• 引导运行中该继电器变为 ON。	S（状态变化）	新增	Q00J/Q00/Q01 QnU*1

*1: 以除 Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU 以外的通用型 QCPU 为对象。

*2: 以除 Q00UJCPU 以外的通用型 QCPU 为对象。

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU M9□□□	对应 CPU
SM671	至标准 ROM 的锁存数据备份结束标志	OFF ：未结束 ON ：结束	• 至标准 ROM 的锁存数据备份结束时该标志变为 ON。 • 在进行了至标准 ROM 的锁存数据备份时存储到 SD672 后面。	S（状态变化）	新增	QnU
SM672	存储卡文件寄存器访问范围标志	OFF ：访问范围内 ON ：超出访问范围	• 在进行了超出存储卡的文件寄存器的范围的访问时该标志将变为 ON。（在 END 处理时设置。） • 在用户程序中进行复位。	S/U	新增	Qn(H) QnPH QnPRH
SM675	至标准 ROM 的锁存数据备份异常结束	OFF ：无出错 ON ：有出错	• 执行至标准 ROM 的锁存数据备份时，备份未能正常结束的情况下将该标志置为 ON。 • 至标准 ROM 的锁存数据备份正常结束时将该标志置为 OFF。	S	新增	QnU
SM676	恢复重复执行指定	OFF ：无指定 ON ：有指定	• 在本 SM 为 ON 的状态下执行了锁存数据备份时，则从下一次的电源 OFF → ON 开始每次电源 OFF → ON 时执行恢复。 • 在删除锁存数据的备份数据，或者再次执行锁存数据备份操作之前，在每次电源 OFF → ON 时执行恢复。	U	新增	
SM680	程序存储器写入异常	ON ：写入异常 OFF ：写入未执行 / 正常	对程序存储器（刷新 ROM）进行写入时，检测出写入出错时该继电器将变为 ON。 在有写入指示时将该继电器置为 OFF。	S（写入时）	新增	
SM681	程序存储器写入中标志	ON ：正在执行写入 OFF ：未执行写入	在对程序存储器（刷新 ROM）进行写入处理的执行过程中将该继电器置为 ON，写入结束时将该继电器置为 OFF。	S（写入时）	新增	
SM682	程序存储器改写次数异常标志	ON ：改写次数达到 10 万次 OFF ：改写次数不到 10 万次	程序存储器（刷新 ROM）的改写次数达到 10 万次时该继电器将变为 ON。 （需要更换 CPU 模块。）	S（写入时）	新增	
SM685	标准 ROM 写入异常	ON ：写入异常 OFF ：写入未执行 / 正常	对标准 ROM（刷新 ROM）进行写入时，检测出写入出错时该继电器将变为 ON。 在有写入指示时将该继电器置为 OFF。	S（写入时）	新增	
SM686	标准 ROM 写入中标志	ON ：正在执行写入 OFF ：未执行写入	在对标准 ROM（刷新 ROM）进行写入处理的过程中将该继电器置为 ON，写入结束后置为 OFF。	S（写入时）	新增	
SM687	标准 ROM（刷新 ROM）改写次数异常标志	ON ：改写次数达到 10 万次 OFF ：改写次数不到 10 万次	标准 ROM（刷新 ROM）的改写次数达到 10 万次时该继电器将变为 ON。 （需要更换 CPU 模块。）	S（写入时）	新增	
SM691	备份开始准备状态标志	OFF ：备份开始准备未结束 ON ：备份开始准备结束	备份开始准备结束时将该继电器置为 ON。	S（状态变化）	新增	QnU*1
SM692	恢复结束标志	OFF ：恢复未结束 ON ：恢复结束	存储卡内的备份数据的恢复结束时该继电器将变为 ON。	S（状态变化）	新增	

*1：以序列号的前 5 位数为“10102”以后的模块为对象。（但是，Q00UJCPU、Q00UCPU、Q01UCPU 除外。）

(7) 指令相关

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU M9□□□	对应 CPU
SM700	进位标志	OFF ：进位 OFF ON ：进位 ON	• 是可用于应用指令中的进位标志。	S（执行指令时）	M9012	QCPU
SM701	输出字符数切换	输出的字符数、输出模式切换	• 用于 PR、PRC、BINDA、DBINDA、BINHA、DBINHA、BCDDA、DBCDDA、COMRD 指令。	U	M9049	Qn(H) QnPH QnPRH QnU
SM702	搜索方法	OFF ：逐次搜索 ON ：2 分搜索	• 指定搜索指令的搜索方法。 • 2 分搜索时，需要进行数据排列。	U	新增	QCPU
SM703	排序顺序	OFF ：升序 ON ：降序	• 指定排序指令中数据的排列方法是按升序还是降序进行。	U	新增	
SM704	块比较	OFF ：有不一致 ON ：全部一致	• BKCMP 指令中全部数据条件成立时该继电器变为 ON。	S（执行指令时）	新增	QCPU
			• DBKCMP 指令中全部数据条件成立时该继电器变为 ON。	S（执行指令时）	新增	QnU*2
SM710	CHK 指令优先顺序标志	OFF ：条件优先 ON ：模式优先	• OFF 时保持原样不变。 • ON 时对 CHK 的优先顺序进行变更。	S（执行指令时）	新增	Qn(H) QnPH QnPRH
SM709	DT、TM 指令非法数据检测标志	OFF ：无非法数据 ON ：有非法数据	• 在 DT、TM 指令中比较对象数据不能被识别为日期数据或者时钟数据时，或者比较对象软元件（3 字）超出了指定软元件范围时该标志将变为 ON。	S（执行指令时）/U	新增	QnU*2
SM715	EI 标志	OFF ：DI 中 ON ：EI 中	• 执行 EI 指令时该标志变为 ON。	S（执行指令时）	新增	QCPU
SM716	块比较（除中断程序以外）	OFF ：有不一致 ON ：无不一致	DBKCMP 指令中全部数据条件一致时该继电器变为 ON。（初始执行型程序 / 扫描执行型程序 / 通过初始执行型程序或者扫描执行型程序执行的待机型程序）	S（执行指令时）	新增	QnU*2
SM717	块比较（中断程序）	OFF ：有不一致 ON ：无不一致	DBKCMP 指令中全部数据条件一致时该继电器变为 ON。（中断程序 / 恒定周期执行型程序 / 通过中断程序或者恒定周期执行型程序执行的待机型程序）	S（执行指令时）	新增	
SM718	块比较（中断程序（I45））	OFF ：有不一致 ON ：无不一致	DBKCMP 指令中全部数据条件一致时该继电器变为 ON。（中断程序（I45）/ 通过中断程序（I45）执行的待机型程序）	S（执行指令时）	新增	QnU*3
SM720	注释读取结束标志	OFF ：注释读取未结束 ON ：注释读取结束	• COMRD、PRC 指令的处理结束时该标志仅 1 个扫描 ON。	S（状态变化）	新增	Qn(H) QnPH
			• COMRD 指令的处理结束时该标志仅 1 个扫描 ON。			QnPRH QnU

*1：以序列号的前 5 位数为“09042”以后的模块为对象。
*2：以下述模块为对象。
- 序列号的前 5 位数为“10102”以后的通用型 QCPU
- Q00UJCPU、Q00UCPU、Q01UCPU

*3：以下述模块为对象。
- 序列号的前 5 位数为“10102”以后的通用型 QCPU
- Q00UCPU、Q01UCPU

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU M9□□□	对应 CPU
SM721	文件访问中	OFF ：除文件访问中以外 ON ：文件访问中	• 在 SP.FWRITE、SP.FREAD、COMRD、PRC、LEDC 指令中，在文件的访问过程中该继电器将变为 ON。	S（状态变化）	新增	Qn(H) QnPH
			• 在 SP.FWRITE、SP.FREAD、COMRD、LEDC 指令中，在文件的访问过程中该继电器将变为 ON。			Qn(H) QnPH QnPRH
			• 在 SP.FWRITE、SP.FREAD、COMRD、SP.DEVST 指令中，在文件的访问过程中该继电器将变为 ON。			QnU
			• 在访问 ATA 卡及标准 ROM 时该继电器将变为 ON。			QnU*1
SM722	BIN、DBIN 指令出错禁止标志	OFF ：进行出错检测 ON ：不进行出错检测	• 不希望在 BIN、DBIN 指令中出现 “OPERATION ERROR” 时将该标志置为 ON。	U	新增	QCPU
SM734	XCALL 指令执行条件指定	OFF ：在执行条件的上升沿时不执行 ON ：在执行条件的上升沿时执行	• 该继电器为 OFF 时在执行条件的上升沿时不执行 XCALL 指令。 • 该继电器为 ON 时在执行条件的上升盐时执行 XCALL 指令。	U	新增	Qn(H)*1
SM735	SFC 注释读取指令执行中标志	OFF ：不执行 SFC 注释读取指令 ON ：正在执行 SFC 注释读取指令	• 在 SFC 步注释读取指令 (S(P).SFCSCOMR)、SFC 移位条件注释读取指令 (S(P).SFCTCOMR) 的执行过程中该标志将变为 ON。	S（状态变化）	新增	Qn(H)*2 QnPH*3 QnPRH*3
SM738	MSG 指令受理标志	OFF ：未执行指令 ON ：执行指令	• 执行了 MSG 指令时该标志将变为 ON。	S（执行指令时）	新增	Qn(H) QnPRH
SM750	标度指令搜索方法设置	OFF ：逐次搜索 ON ：二分搜索	确定执行标度指令时的搜索方法。	U	新增	QnU*4

*1 ：以序列号的前 5 位数为 “06082” 以后的模块为对象。
*2 ：以序列号的前 5 位数为 “07012” 以后的模块为对象。
*3 ：以序列号的前 5 位数为 “07032” 以后的模块为对象。
*4 ：以下述模块为对象。
- 序列号的前 5 位数为 “10102” 以后的通用型 QCPU
- Q00UJCPU、Q00UCPU、Q01UCPU

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU M9□□□	对应 CPU
SM774	PID 无冲击处理（用于完全微分）	OFF ：使其一致 ON ：不使其一致	• 手动模式时，指定是否使设置值（SV）与测定值（PV）一致。	U	新增	Q00J/Q00/Q01*4 Qn(H) QnPRH QnU
SM775	执行 COM/CCOM 指令时刷新处理选择	OFF ：进行链接刷新 ON ：不进行链接刷新	• 执行 COM 指令时仅进行与 CPU 模块的通信的情况下，选择是否进行链接刷新处理。	U	新增	Q00J/Q00/Q01 Qn(H) QnPH
		OFF ：进行除 I/O 刷新以外的刷新处理 ON ：进行 SD778 中设置的刷新	• 选择执行 COM、CCOM 指令时是进行除 I/O 刷新以外的刷新，还是进行 SD778 中设置的刷新。	U	新增	Q00J/Q00/Q01*4 Qn(H)*5 QnPH*3 QnPRH QnU
SM776	CALL 时局部软元件的允许/禁止设置	OFF ：局部软元件禁止 ON ：局部软元件允许	• 设置执行 CALL 指令时被调用的子程序的局部软元件的有效/无效。	U	新增	Qn(H) QnPH QnPRH QnU*8
SM777	中断程序中局部软元件的允许/禁止设置	OFF ：局部软元件禁止 ON ：局部软元件允许	• 设置执行中断程序时局部软元件的有效/无效。	U	新增	
SM794	PID 无冲击处理（用于不完全微分）	OFF ：使其一致 ON ：不使其一致	• 手动模式时，指定是否使设置值（SV）与测定值（PV）一致。	U	新增	Q00J/Q00/Q01*4 Qn(H)*6 QnPRH QnU
SM796	多 CPU 高速通信专用指令使用块信息（1 号机用）	OFF ：块预留 ON ：未能预留出 SD796 中设置的块数	• 多 CPU 高速通信专用指令（对象 CPU=1 号机）中使用的专用指令传送区的剩余块数低于 SD796 中指定的块数时该继电器将变为 ON。 执行指令时该继电器变为 ON。END 处理时区域中有空余容量时该继电器变为 OFF。	S（执行指令时/END 处理时）	新增	QnU*7
SM797	多 CPU 高速通信专用指令使用块信息（2 号机用）	OFF ：块预留 ON ：未能预留出 SD797 中设置的块数	• 多 CPU 高速通信专用指令（对象 CPU=2 号机）中使用的专用指令传送区的剩余块数低于 SD797 中指定的块数时该继电器将变为 ON。 执行指令时该继电器变为 ON。END 处理时区域中有空余容量时该继电器变为 OFF。	S（执行指令时/END 处理时）	新增	
SM798	多 CPU 高速通信专用指令使用块信息（3 号机用）	OFF ：块预留 ON ：未能预留出 SD798 中设置的块数	• 多 CPU 高速通信专用指令（对象 CPU=3 号机）中使用的专用指令传送区的剩余块数低于 SD798 中指定的块数时该继电器将变为 ON。 执行指令时该继电器变为 ON。END 处理时区域中有空余容量时该继电器变为 OFF。	S（执行指令时/END 处理时）	新增	
SM799	多 CPU 高速通信专用指令使用块信息（4 号机用）	OFF ：块预留 ON ：未能预留出 SD799 中设置的块数	• 多 CPU 高速通信专用指令（对象 CPU=4 号机）中使用的专用指令传送区的剩余块数低于 SD799 中指定的块数时该继电器将变为 ON。 执行指令时该继电器变为 ON。END 处理时区域中有空余容量时该继电器变为 OFF。	S（执行指令时/END 处理时）	新增	

*3：以序列号的前 5 位数为“07032”以后的模块为对象。
*4：以功能版本 B 以后为对象。
*5：以序列号的前 5 位数为“04012”以后的模块为对象。
*6：以序列号的前 5 位数为“05032”以后的模块为对象。
*7：以除 Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU 以外的通用型 QCPU 为对象。
*8：以除 Q00UJCPU 以外的通用型 QCPU 为对象。

(8) 调试

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU M9□□□	对应 CPU
SM800	跟踪准备	OFF ：未准备 ON ：准备结束	• 跟踪的准备结束时该继电器变为 ON。	S（状态变化）	新增	Qn(H) QnPH QnPRH QnU*1
SM801	跟踪开始	OFF ：中止 ON ：开始	• 该继电器为 ON 时，开始跟踪。 • 该继电器为 OFF 时中止跟踪。（相关特殊继电器 OFF）	U	M9047	
SM802	跟踪执行中	OFF ：中止 ON ：开始	• 在跟踪执行中状态时该继电器变为 ON。	S（状态变化）	M9046	
SM803	跟踪触发	OFF → ON：开始	• OFF → ON 时跟踪的触发变为 ON。 （与 TRACE 指令执行状态相同。）	U	M9044	
SM804	跟踪触发后	OFF ：不处于触发后状态 ON ：处于触发后状态	• 跟踪触发后该继电器变为 ON。	S（状态变化）	新增	
ΣM804	跟踪结束	OFF ：未结束 ON ：结束	• 跟踪结束时该继电器变为 ON。	S（状态变化）	M9043	
SM826	跟踪出错	OFF ：正常 ON ：出错	• 跟踪执行过程中发生了出错时该继电器变为 ON。	S（状态变化）	新增	
SM829	跟踪设置的强制登录指定	ON ：强制登录有效 OFF ：强制登录无效	• 通过将本 SM 置为 ON，在 GX Developer 中登录采样跟踪设置，即使在触发条件成立的状态下，也可将采样跟踪设置登录到 CPU 模块中。	U	新增	QnU*1

*1：以除 Q00UJCPU 以外的通用型 QCPU 为对象。

(9) A→Q 转换对应

ACPU 的特殊继电器 M9000 ～ M9255 通过 A → Q 转换进行了转换时的对应特殊继电器为 SM1000 ～ SM1255。
（但是，基本型 QCPU、冗余 CPU 不支持 A → Q 转换。）
这些特殊继电器均由系统方进行设置。用户不能通过程序进行 ON/OFF。
用户希望对其进行 ON/OFF 时，应将程序中的继电器修改为 QCPU 用的特殊继电器。
但是，对于 SM1084、SM1200 ～ SM1255，只有在转换前的 M9084、M9200 ～ M9255 中可以由用户进行 ON/OFF 的特殊继电器，才可以在转换后的 SM1084、SM1200 ～ SM1255 中由用户进行 ON/OFF。
关于 ACPU 的特殊继电器的详细内容，请参阅各 CPU 模块的用户手册以及 MELSECNET、MELSECNET/B 数据链接系统参考手册。

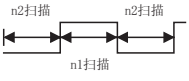
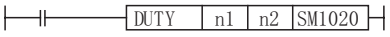
要 点

在高性能型 QCPU、过程 CPU 以及通用型 QCPU 中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，对 “A 系列 CPU 兼容设置”的复选框进行勾选。
但是，使用转换后的特殊继电器时需要耗费处理时间，因此不使用的情况下应将 “A 系列 CPU 兼容设置”的勾选取消。

备注

以下为对修改用的特殊继电器一栏的补充说明。

1. 对于修改用特殊继电器一栏中记述的软元件编号，应修改为记述的 QCPU 用的特殊继电器。
2. 记述了□的软元件编号可以使用转换后的特殊继电器。
3. 记述了☒的软元件编号在 QCPU 中不能使用。

ACPU 的特殊继电器	转换后的特殊继电器	修改用的特殊继电器	名称	内容	详细内容	对应 CPU
M9000	SM1000	--	保险丝熔断	OFF ：正常 ON ：有保险丝熔断模块	• 如果存在有某个处于保险丝熔断状态的输出模块，则该继电器将变为 ON。 • 此后即使恢复正常后也保持为 ON 状态不变。 • 对远程 I/O 站的输出模块也进行保险丝熔断状态检查。	Qn(H) QnPH QnU*1
M9002	SM1002	--	I/O 模块校验出错	OFF ：正常 ON ：有出错	• I/O 模块的电源接通时与登录的状态不相同时该继电器将变为 ON。 • 此后即使恢复正常后也保持为 ON 状态不变。 • 对远程 I/O 站的模块也进行 I/O 模块校验。 • 仅在对特殊寄存器 SD1116 ～ SD1123 进行了复位时该继电器才被复位。	
M9005	SM1005	--	检测出 AC DOWN	OFF ：无 AC DOWN ON ：有 AC DOWN	• 使用 AC 电源模块时发生了 20ms 以内的瞬间掉电时该继电器变为 ON。 • 通过电源的 OFF → ON 该继电器被复位。	
					• 使用 DC 电源模块时发生了 10ms 以内的瞬间掉电时该继电器变为 ON。 • 通过电源的 OFF → ON 该继电器被复位。	
M9006	SM1006	--	电池电压过低	OFF ：正常 ON ：电池电压过低	• 电池电压低于规定值时该继电器将变为 ON。 • 此后当电池电压正常时该继电器将变为 OFF。	
M9007	SM1007	--	电池电压过低锁存	OFF ：正常 ON ：电池电压过低	• 电池电压低于规定值时该继电器将变为 ON。 • 此后即使电池电压正常时该继电器仍将保持为 ON 状态不变。	
M9008	SM1008	SM1	自诊断出错	OFF ：无出错 ON ：有出错	• 自诊断的结果为有出错时该继电器将变为 ON。	
M9009	SM1009	SM62	报警器检测	OFF ：未检测出 ON ：检测出	• 执行了 OUT F、SET F 的指令时该继电器将变为 ON。 • SD1124 的内容变为 0 时该继电器将变为 OFF。	
M9011	SM1011	SM56	运算出错标志	OFF ：无出错 ON ：有出错	• 在应用指令的执行过程中发生了运算出错时该继电器将变为 ON。 •此后即使变为正常时该继电器仍将保持为ON状态不变。	
M9012	SM1012	SM700	进位标志	OFF ：进位 OFF ON ：进位 ON	• 应用指令中使用的进位标志。	Qn(H) QnPH
M9016	SM1016	×	数据存储器清除标志	OFF ：无处理 ON ：输出清除	• SM1016 为 ON 时，通过计算机等进行远程 RUN 时对包含锁存范围在内的所有数据存储器（特殊继电器・特殊寄存器除外）进行全部清除。	
M9017	SM1017	×	数据存储器清除标志	OFF ：无处理 ON ：输出清除	• SM1017 为 ON 时，通过计算机等进行远程 RUN 时对未锁存的所有数据存储器（特殊继电器・特殊寄存器除外）进行全部清除。	
M9020	SM1020	--	0 号用户定时时钟		• 是以指定扫描间隔重复进行 ON/OFF 的继电器。 • 可编程控制器的电源 ON 时或者 CPU 模块的复位时将时钟从 OFF 变为启动状态。 设置通过 DUTY 指令进行 ON/OFF 的间隔。  n1：ON 的扫描间隔 n2：OFF 的扫描间隔 ※：在除通用型 QCPU 以外的程序中，对于将 SM1020 ～ SM1024 指定为用户用定时时钟的 DUTY 指令，将可编程控制器类型变更为通用型 QCPU 时，将被替换为 SM420 ～ SM424。 （在通用型 QCPU 中，不能指定 SM1020 ～ SM1024。）	Qn(H) QnPH QnU*1
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 ：无处理 ON ：有设置请求	• 在 SM1025 由 OFF 变为 ON 时的扫描的 END 指令执行后，将 SD1025 ～ SD1028 中存储的时钟数据写入到 CPU 模块中。	
M9026	SM1026	--	时钟数据出错	OFF ：无出错 ON ：有出错	• 时钟数据（SD1025 ～ SD1028）的值发生了出错时该继电器将变为 ON，无出错时将变为 OFF。	
M9028	SM1028	--	时钟数据读取请求	OFF ：无处理 ON ：读取请求	• SM1028 为 ON 时将时钟数据以 BCD 值读取到 SD1025 ～ SD1028 中。	

*1：以下述模块为对象。

• 序列号的前 5 位数为 “10102” 以后的通用型 QCPU

• Q00UJCPU、Q00UCPU、Q01UCPU

ACPU 的特殊继电器	转换后的特殊继电器	修改用的特殊继电器	名称	内容	详细内容	对应 CPU
M9029	SM1029	×	数据通信请求批量处理	OFF ：不进行批量处理 ON ：进行批量处理	• 通过使用顺控程序将 SM1029 置为 ON，将 1 个扫描期间受理的数据通信请求在该扫描的 END 处理时进行全部处理。 • 对于数据通信请求批量处理，可以在 RUN 中进行 ON/OFF 变更。 • 默认为 OFF（按照数据通信请求的受理顺序，每个 END 处理时处理 1 个请求。）	Qn(H) QnPH
M9030	SM1030	--	0.1 秒时钟	0.05秒 0.05秒	• 生成 0.1 秒、0.2 秒、1 秒、2 秒的各个时钟。 • 不是在每个扫描中进行 ON/OFF，即使在扫描过程中如果经过了相应时间也将进行 ON/OFF。 • 在可编程控制器的电源 ON 或者 CPU 模块的复位时将该时钟从 OFF 置为启动。	Qn(H) QnPH QnU*1
M9031	SM1031	--	0.2 秒时钟	0.1秒 0.1秒		
M9032	SM1032	--	1 秒时钟	0.5秒 0.5秒		
M9033	SM1033	--	2 秒时钟	1秒 1秒		
M9034	SM1034	--	2n 秒时钟 *2 （1 分时钟）	n秒 n秒	• 是按照 SD414 中指定的秒数重复进行 ON/OFF 的继电器。（默认值： n=30） • 不是在每个扫描中进行 ON/OFF，即使在扫描过程中如果经过了相应时间也将进行 ON/OFF。 • 在可编程控制器的电源 ON 或者 CPU 模块的复位时将该时钟从 OFF 置为启动。	
M9036	SM1036	--	常时 ON	ON OFF	• 该继电器是在顺控程序中作为初始化及应用指令的虚拟触点使用的继电器。 • SM1036、SM1037 的 ON 及 OFF 与 CPU 模块前面的键开关的状态无关，SM1038、SM1039 的 ON 及 OFF 根据 CPU 模块前面的键开关的状态而变化。键开关处于 STOP 时，变为 OFF 状态。 键开关处于 STOP 以外的状态时，SM1038 仅 1 个扫描 ON，SM1039 仅 1 个扫描 OFF。	
M9037	SM1037	--	常时 OFF	ON OFF		
M9038	SM1038	--	RUN 后仅 1 个扫描 ON	ON 1个扫描 OFF		
M9039	SM1039	--	RUN 标志（RUN 后仅 1 个扫描 OFF）	ON 1个扫描 OFF		
M9040	SM1040	SM206	PAUSE 允许线圈	OFF ：禁止 PAUSE ON ：允许 PAUSE	• RUN 键开关为 PAUSE 或者 PAUSE 触点为 ON 时，如果将 SM1040 置为 ON 则变为 PAUSE 状态，SM1041 将变为 ON。	Qn(H) QnPH
M9041	SM1041	SM204	PAUSE 状态触点	OFF ：不处于 PAUSE 状态 ON ：处于 PAUSE 状态		Qn(H) QnPH QnU*1
M9042	SM1042	SM203	STOP 状态触点	OFF ：不处于 STOP 状态 ON ：处于 STOP 状态	• RUN 键开关或者 RUN/STOP 开关处于 STOP 时该继电器将变为 ON。	
M9043	SM1043	SM805	采样跟踪结束	OFF ：采样跟踪中 ON ：采样跟踪结束	• 执行 STRA 指令后，参数中设置的次数的采样跟踪结束时该继电器将变为 ON。 此后通过执行 STRAR 指令而被复位。	
M9044	SM1044	SM803	采样跟踪	OFF → ON：与执行 STRA 指令时相同 ON → OFF：与执行 STRAR 指令时相同	• 通过 SM1044 的 ON/OFF 可以模拟执行 STRA/STRAR 指令。（通过外围设备对 SM1044 进行强制 ON/OFF。） SM1044 由 OFF → ON 时执行 STRA 指令 SM1044 由 ON → OFF 时执行 STRAR 指令 此时，采样跟踪的条件使用 SD1044 中存储的值。 扫描时、定时时→时间（10ms 单位）	Qn(H) QnPH
M9045	SM1045	×	看门狗定时器（WDT）的复位	OFF ：不进行 WDT 复位 ON ：进行 WDT 复位	• 通过将 SM1045 置为 ON，在执行 ZCOM 指令及数据通信请求批量处理时进行 WDT 的复位。 （用于扫描时间超过了 200ms 的情况下）	

*1：以下述模块为对象。
- 序列号的前 5 位数为 “10102” 以后的通用型 QCPU
- Q00UJCPU、Q00UCPU、Q01UCPU

*2：1 分时钟表示 ACPU 的特殊继电器（M9034）的名称。

ACPU 的特殊继电器	转换后的特殊继电器	修改用的特殊继电器	名称	内容	详细内容	对应 CPU
M9046	SM1046	SM802	采样跟踪	OFF ：跟踪中以外 ON ：跟踪中	• 在采样跟踪执行过程中变为 ON。	Qn(H) QnPH QnU*1
M9047	SM1047	SM801	采样跟踪准备	OFF ：中止采样跟踪 ON ：开始采样跟踪	• 在执行采样跟踪时如果未将 SM1047 置为 ON 则无法执行。 通过将 SM1047 置为 OFF 可以中止采样跟踪。	Qn(H) QnPH
M9049	SM1049	SM701	输出字符数切换	OFF ：输出至 NULL 码为止 ON ：输出 16 字符	• SM1049 为 OFF 状态时，输出至 NULL(00H) 码为止。 • SM1049 为 ON 状态时，输出 16 字符的 ASCII 码。	
M9051	SM1051	×	禁止执行 CHG 指令	OFF ：允许 ON ：禁止	• 禁止执行 CHG 指令时将该继电器置为 ON。 • 程序传送请求时将其置为 ON，传送结束时自动地变为 OFF。	
M9052	SM1052	×	SEG 指令切换	OFF ：显示 7SEG ON ：I/O 部分刷新	• SM1052 为 ON 时，作为 I/O 部分刷新指令执行。 SM1052 为 OFF 时，作为 7SEG 显示指令执行。	
M9056	SM1056	×	主程序侧 P、I 设置请求	OFF ：P、I 设置请求中以外 ON ：P、I 设置请求中	• RUN 中其它程序（例如主程序为 RUN 中状态时则表示子程序）的传送结束时将 P、I 设置请求置为 ON。P、I 设置结束时自动地置为 OFF。	
M9057	SM1057	×	子程序 (1) 侧 P、I 设置请求	OFF ：P、I 设置请求中以外 ON ：P、I 设置请求中		
M9058	SM1058	×	主程序侧 P、I 设置请求	P、I 设置结束时瞬时 ON	• P、I 设置结束时瞬时 ON 后，立即 OFF。	
M9059	SM1059	×	子程序侧 P、I 设置请求	P、I 设置结束时瞬时 ON		
M9060	SM1060	×	子程序 2 侧 P、I 设置请求	OFF ：P、I 设置请求中以外 ON ：P、I 设置请求中	• RUN 中其它程序（例如主程序为 RUN 中状态时则表示子程序）的传送结束时将 P、I 设置请求置为 ON。P、I 设置结束时自动地置为 OFF。	
M9061	SM1061	×	子程序 3 侧 P、I 设置请求	OFF ：P、I 设置请求中以外 ON ：P、I 设置请求中		
M9070	SM1070	×	A8UPU/A8PUJ*3 的搜索所需时间	OFF ：无读取时间的缩短 ON ：有读取时间的缩短	• 通过将该继电器置为 ON，可以缩短 A8UPU/A8PUJ 的搜索所需时间。 （在这种情况下扫描时间将延迟 10%）	
M9084	SM1084	×	出错检查	OFF ：有出错检查 ON ：无出错检查	设置 END 指令处理时是否执行下述出错检查。（用于 END 指令处理时间设置） • 保险丝熔断检查 • 电池检查 • I/O 模块校验检查	
M9091	SM1091	×	运算出错详细标志	OFF ：无出错 ON ：有出错	• 运算出错的详细原因被存储到 SD1091 中时该标志将变为 ON。 • 此后即使恢复正常后也仍将保持为 ON 状态不变。	

*1：以下述模块为对象。

• 序列号的前 5 位数为“10102”以后的通用型 QCPU

• Q00UJCPU、Q00UCPU、Q01UCPU

*3：在 QCPU 中不能使用 A8UPU/A8PUJ。

ACPU 的特殊继电器	转换后的特殊继电器	修改用的特殊继电器	名称	内容	详细内容	对应 CPU
M9100	SM1100	SM320	SFC 程序的有无	OFF ：无 SFC 程序 ON ：有 SFC 程序	• SFC 程序已登录时该继电器将变为 ON。 • SFC 程序未登录时该继电器将变为 OFF。	Qn(H) QnPH
M9101	SM1101	SM321	SFC 程序的启动 / 停止	OFF ：停止 SFC 程序 ON ：启动 SFC 程序	• 初始值被设置为与 SM1100 相同的值。 （有 SFC 程序时自动地变为 ON。） • 通过将本继电器由 ON 变为 OFF，可以停止 SFC 程序的执行。 • 通过将本继电器由 OFF 变为 ON，可以重新执行 SFC 程序。	
M9102	SM1102	SM322	SFC 程序的启动状态	OFF ：初始化启动 ON ：热启动	• 初始值被设置为可编程控制器参数的 SFC 设置的 SFC 程序启动模式。 初始化启动时 ：OFF 热启动时 ：ON	
M9103	SM1103	SM323	连续移位的有无	OFF ：无连续移位 ON ：有连续移位	• 对于未设置 SFC 信息软元件的“连续移位位”的程序，设置连续移位的有无。	
M9104	SM1104	SM324	连续移位阻止标志	OFF ：执行移位时 ON ：未移位时	• 在有连续移位模式下运行的过程中或者在连续移位过程中该标志为 OFF 状态，在未进行连续移位时该标志为 ON 状态。 • 在无连续移位模式下运行的过程中该标志为常时 ON。	
M9108	SM1108	SM90	步移位监视定时器启动（对应于 SD90）	OFF ：监视定时器复位 ON ：监视定时器复位启动	• 开始步移位监视定时器的计量时将该继电器置为 ON。 如果将其置为 OFF 则对步移位监视定时器进行复位。	
M9109	SM1109	SM91	步移位监视定时器启动（对应于 SD91）			
M9110	SM1110	SM92	步移位监视定时器启动（对应于 SD92）			
M9111	SM1111	SM93	步移位监视定时器启动（对应于 SD93）			
M9112	SM1112	SM94	步移位监视定时器启动（对应于 SD94）			
M9113	SM1113	SM95	步移位监视定时器启动（对应于 SD95）			

ACPU 的特殊继电器	转换后的特殊继电器	修改用的特殊继电器	名称	内容	详细内容	对应 CPU
M9114	SM1114	SM96	步移位监视定时器启动（对应于 SD96）	OFF ：监视定时器复位 ON ：监视定时器复位启动	• 开始步移位监视定时器的计量时将该继电器置为 ON。 如果将其置为 OFF 则对步移位监视定时器进行复位。	Qn(H) QnPH
M9196	SM1196	SM325	块停止时的动作输出	OFF ：线圈输出 OFF ON ：线圈输出 ON	• 选择执行块停止时的动作输出。 ON 时 ：对块停止时执行的步的动作输出中使用的线圈的 ON/OFF 状态进行保持。 OFF 时 ：将线圈输出全部置为 OFF。 （通过 SET 指令进行的动作输出与 SM1196 的 ON/OFF 状态无关，将被保持。）	Qn(H) QnPH
M9197	SM1197	×	保险丝熔断、I/O 校验出错显示切换	SM1197 / SM1198 / 显示对象 I/O 编号 OFF / OFF / X/Y0 ～ 7F0 ON / OFF / X/Y800 ～ FF0 OFF / ON / X/Y1000 ～ 17F0 ON / ON / X/Y1800 ～ 1FF0	通过 SM1197、SM1198 的 ON/OFF 的组合，对保险丝熔断模块存储寄存器（SD1100 ～ SD1107）、I/O 模块校验出错存储寄存器（SD1116 ～ SD1123）的 I/O 编号进行切换。	
M9198	SM1198	×				
M9199	SM1199	×	在线采样跟踪状态锁存的数据恢复	OFF ：不进行数据恢复 ON ：进行数据恢复	• 执行了采样跟踪 / 状态锁存时，对 CPU 模块中存储的设置数据进行恢复使处于可重新开始状态。 • 重新执行时应将 SM1199 置为 ON。（不需要通过外围设备重新进行数据写入）	

(10)以太网端口内置 QCPU 对应

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU M9□□□	对应 CPU
SM1270	执行时间设置功能（SNTP 客户）	OFF：未执行时间设置功能（SNTP 客户） ON ：执行时间设置功能（SNTP 客户）	执行时间设置功能（SNTP 客户）的情况下，将该继电器置为 ON。（只有在时间设置参数中时间设置功能被指定为“使用”时）	U	新增	QnU*1
SM1273	远程口令不一致的累计次数清除	OFF：未执行清除 ON ：执行清除	对不一致的远程口令的累计次数（SD979 ～ 999）进行清除时，将该继电器置为 ON。	U	新增	

*1：以以太网端口内置 QCPU 为对象。

(11)过程控制指令

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU M9□□□	对应 CPU
SM1500	保持模式	OFF ：不保持 ON ：有保持	• 指定在 S.IN 指令的范围检查中发生了范围溢出时，是否保持输出值。	U	新增	QnPH QnPRH
SM1501	保持模式	OFF ：不保持 ON ：有保持	• 指定在 S.OUT 指令的范围检查中发生了范围溢出时，是否保持输出值。	U	新增	

(12)冗余对应（自站系统 CPU 信息 *1）

SM1510 ～ SM1599 仅在冗余系统时有效。

在单系统时，全部为 OFF。

<table>
<tr><th>编号</th><th>名称</th><th>内容</th><th>详细内容</th><th>设置方
（设置时机）</th><th>对应 ACPU
M9□□□</th><th>对应 CPU</th></tr>
<tr><td>SM1510</td><td>运行模式</td><td>OFF ：冗余系统备份模式、单系统
ON ：冗余系统分开模式</td><td>• 运行模式为冗余系统的分开模式时该继电器变为 ON。</td><td>S（每次 END）</td><td>新增</td><td rowspan="5">QnPRH</td></tr>
<tr><td>SM1511</td><td>A 系统判别标志</td><td rowspan="2">• 显示冗余系统的 A 系统 /B 系统。
• 即使热备电缆中途脱落时也不变化。
<table><tr><th></th><th>A系统</th><th>B系统</th><th>发生TRK.CABLE ERR.(出错代码：6120)时
(系统未确定)</th></tr><tr><td>SM1511</td><td>ON</td><td>OFF</td><td>OFF</td></tr><tr><td>SM1512</td><td>OFF</td><td>ON</td><td>OFF</td></tr></table></td><td rowspan="2">S（初始化）</td><td rowspan="2">新增</td></tr>
<tr><td>SM1512</td><td>B 系统判别标志</td></tr>
<tr><td>SM1513</td><td>调试模式运行中</td><td>OFF ：不处于调试模式运行状态
ON ：处于调试模式运行状态</td><td>• CPU 模块处于调试模式运行状态时该继电器变为 ON。</td><td>S（初始化）</td><td>新增</td></tr>
<tr><td>SM1515</td><td>控制系统判别标志</td><td rowspan="2">• 显示 CPU 模块的运行状态。
• 即使热备电缆中途脱落时也不变化。
<table><tr><th></th><th>控制系统</th><th>待机系统</th><th>发生TRK.CABLE ERR.(出错代码：6120)时
(系统未确定)</th></tr><tr><td>SM1515</td><td>ON</td><td>OFF</td><td>OFF</td></tr><tr><td>SM1516</td><td>OFF</td><td>ON</td><td>OFF</td></tr></table></td><td rowspan="2">S（状态变化）</td><td rowspan="2">新增</td></tr>
<tr><td>SM1516</td><td>待机系统判别标志</td></tr>
</table>

*1：存储自身系统 CPU 模块的信息。

编号	名称	内容	详细内容			设置方（设置时机）	对应 ACPU M9□□□	对应 CPU
SM1517	CPU 模块启动状态	OFF ：电源 ON 启动 ON ：运行系统切换启动	• CPU 模块通过运行系统切换（从待机系统切换为控制系统）启动时该继电器变为 ON。通过电源 ON 启动变为控制系统时保持 OFF 状态不变。			S（状态变化）	新增	QnPRH
SM1518	从待机系统切换为控制系统后仅 1 个扫描 ON	ON 1个扫描 OFF	• 从待机系统切换为控制系统后，仅 1 个扫描 ON。 • 本触点只能在扫描执行型程序中使用。			S（每次 END）	新增	
SM1519	上次控制系统判别标志	ON 1个扫描 OFF	• 上次控制系统为 B 系统的情况下，A 系统 /B 系统同时电源 ON/ 复位解除时仅在 A 系统侧 RUN 后 1 个扫描 ON。			S（每次 END）	新增	
SM1520	数据热备传送触发指定	OFF ：无触发 ON ：有触发	SM1520	块 1	• 通过冗余参数的热备设置进行数据传送时，对对象块进行触发指定。 • 在热备设置中选择了“热备块号 1 自动传送”的情况下，SM1520 在电源 ON/STOP → RUN 时由系统置为 ON。除此以外的情况下，SM1520 ～ SM1583 由用户置为 ON。	S（初始化）/U	新增	
SM1521			SM1521	块 2				
SM1522			SM1522	块 3				
SM1523			SM1523	块 4				
SM1524			SM1524	块 5				
SM1525			SM1525	块 6				
SM1526			SM1526	块 7				
SM1527			SM1527	块 8				
SM1528			SM1528	块 9				
SM1529			SM1529	块 10				
SM1530			SM1530	块 11				
SM1531			SM1531	块 12				
SM1532			SM1532	块 13				
SM1533			SM1533	块 14				
SM1534			SM1534	块 15				
SM1535			SM1535	块 16				
SM1536			SM1536	块 17				
SM1537			SM1537	块 18				

编号	名称	内容	详细内容			设置方（设置时机）	对应 ACPU M9□□□	对应 CPU
SM1538	数据热备传送触发指定	OFF ：无触发 ON ：有触发	SM1538	块 19	• 通过冗余参数的热备设置进行数据传送时，对对象块进行触发指定。 • 在热备设置中选择了“热备块号 1 自动传送”的情况下，SM1520 在电源 ON/STOP → RUN 时由系统置为 ON。除此以外的情况下，SM1520 ～ SM1583 由用户置为 ON。	S（初始化）/U	新增	QnPRH
SM1539			SM1539	块 20				
SM1540			SM1540	块 21				
SM1541			SM1541	块 22				
SM1542			SM1542	块 23				
SM1543			SM1543	块 24				
SM1544			SM1544	块 25				
SM1545			SM1545	块 26				
SM1546			SM1546	块 27				
SM1547			SM1547	块 28				
SM1548			SM1548	块 29				
SM1549			SM1549	块 30				
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				

编号	名称	内容	详细内容			设置方（设置时机）	对应 ACPU M9□□□	对应 CPU
SM1562	数据热备传送触发指定	OFF ：无触发 ON ：有触发	SM1562	块 43	• 通过冗余参数的热备设置进行数据传送时，对对象块进行触发指定。 • 在热备设置中选择了“热备块号 1 自动传送”的情况下，SM1520 在电源 ON/STOP → RUN 时由系统置为 ON。除此以外的情况下，SM1520 ～ SM1583 由用户置为 ON。	S（初始化）/U	新增	QnPRH
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				

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU M9□□□	对应 CPU
SM1590	来自于网络模块的系统切换有无标志	OFF ：无系统切换请求发出模块 ON ：有系统切换请求发出模块	• 通过网络模块发出了系统切换请求的情况下该标志将变为 ON。通过 SD1590 可以确认发出了系统切换请求的模块号。 • SD1590 的各个位全部为 OFF 时将该标志置为 OFF。	S（每次 END）	新增	QnPRHSM
SM1591	系统切换时的待机系统出错检测无效标志	ON ：系统切换时新待机系统侧不进行出错检测 OFF ：系统切换时新待机系统侧进行出错检测	由于下述原因导致系统切换时，指定系统切换后的新待机系统中是否进行出错 “STANDBY”（出错代码：6210）检测。 〈对象的系统切换原因〉 • 通过 GX Developer 进行的系统切换 • 根据系统切换指令进行的系统切换 • 根据来自于网络模块的系统切换请求进行的系统切换	U	新增	
SM1592	手动切换允许标志	OFF ：禁止手动切换 ON ：允许手动切换	• 通过 GX Developer 或者系统切换指令（SP.CONTSW）指定手动切换动作。	U	新增	
SM1593	至待机系统 CPU 的扩展基板的访问设置	OFF ：出错 ON ：无处理	设置分开模式时从待机系统 CPU 向扩展基板上安装的智能功能模块的缓冲存储器进行了访问时的动作。 OFF：从待机系统 CPU 向扩展基板的智能功能模块的缓冲存储器进行了访问时发生 “OPERATION ERROR”（出错代码：4112）。 ON：从待机系统 CPU 向扩展基板的智能功能模块的缓冲存储器进行了访问时执行无处理。	U		QnPRH*2
SM1595	从控制系统至待机系统的存储器拷贝开始标志	OFF ：拷贝开始请求 ON ：拷贝未实施	• 将 SM1595 从 OFF 变为 ON 时，从控制系统至待机系统的存储器拷贝开始。此外，将 SM1595 从 OFF 变为 ON 时，如果 SD1595 中未存储拷贝目标的 I/O 地址号（待机系统 CPU 模块：3D1H）则不开始拷贝。	U	新增	QnPRH
SM1596	从控制系统至待机系统的存储器拷贝执行中标志	OFF ：拷贝未实施 ON ：拷贝实施中	• 在从控制系统至待机系统的存储器拷贝执行过程中该标志将变为 ON。 • 拷贝结束时该标志将变为 OFF。	S（拷贝开始/结束时）		
SM1597	从控制系统至待机系统的存储器拷贝结束标志	OFF ：拷贝未结束 ON ：拷贝结束	• 从控制系统至待机系统的存储器拷贝结束时该标志将变为 ON。	S（拷贝结束时）/U		
SM1598	从控制系统至待机系统的存储器拷贝标准 ROM 拷贝标志	OFF ：进行拷贝 ON ：不进行拷贝	• 从控制系统至待机系统的存储器拷贝时，不进行标准 ROM 拷贝的情况下将该标志置为 ON。	U		

*2 ：以序列号的前 5 位数为 “09012”以后的模块为对象。

(13)冗余对应（其它系统 CPU 信息 *1）

SM1600 ～ SM1650 仅在冗余系统的备份模式时有效，在分开模式时不能被刷新。
SM1651 ～ SM1699 在备份模式、分开模式时均有效。
在单系统时，SM1600 ～ SM1699 均处于 OFF 状态。

编号	名称	内容	详细内容	设置方（设置时机）	对应本机系统 SM□□ *2	对应 CPU
SM1600	其它系统异常标志	OFF ：无异常 ON ：有异常	• 冗余系统出错检查中检测出了出错时该标志将变为 ON。(SD1600 的某个位为 ON 时该标志将变为 ON。） • 此后，如果异常消除则变为 OFF。	S（每次 END）	－	QnPRH
SM1610	其它系统诊断出错有无	OFF ：无出错 ON ：有出错	• 其它系统中发生了诊断出错时该标志将变为 ON。（包括报警器的 ON、通过 CHK 指令检测出的出错。） • 反映其它系统 CPU 模块的 SM0 的信息。	S（每次 END）	SM0	
SM1611	其它系统自诊断出错有无	OFF ：无自诊断出错 ON ：有自诊断出错	• 其它系统中发生了自诊断出错时该标志将变为 ON。（不包括报警器的 ON、通过 CHK 指令检测出的出错。） • 反映其它系统 CPU 模块的 SM1 的状态。	S（每次 END）	SM1	
SM1615	其它系统出错公共信息有无	OFF ：无公共信息 ON ：有公共信息	• 在其它系统中有出错的公共信息时，该标志将变为 ON。 • 反映其它系统 CPU 模块的 SM5 的状态。	S（每次 END）	SM5	
SM1626	其它系统出错个别信息	OFF ：无个别信息 ON ：有个别信息	• 在其它系统中有出错的个别信息时，该标志将变为 ON。 • 反映其它系统 CPU 模块的 SM16 的状态。	S（每次 END）	SM16	
SM1649	待机系统出错解除指令	OFF → ON: 对待机系统中发生的出错进行解除	通过将本继电器由 OFF 变为 ON，对待机系统 CPU 模块中发生的继续运行出错进行解除。 通过 SD1649 指定要解除的出错的出错代码。	U	－	

*1：存储其它系统 CPU 模块的诊断信息、系统信息。
*2：表示本站系统 CPU 模块中对应的特殊继电器（SM□□）。

(14)冗余对应（热备）

SM1700 ～ SM1799 在备份模式、分开模式中均有效。
在单系统时，全部为 OFF。

编号	名称	内容	详细内容			设置方（设置时机）	对应 ACPU M9□□□	对应 CPU
SM1700	传送触发结束标志	OFF ：传送未结束 ON ：传送结束	• 块 1 ～块 64 的某个块的传送结束时，该标志仅 1 个扫描 ON。			S（状态变化）	新增	QnPRH
SM1709	RUN 中写入冗余追踪执行中的用户切换禁止 / 允许设置	ON ：允许用户切换（禁止被解除的状态） OFF ：禁止用户切换	(1) 通过将本继电器由 OFF 变为 ON，可以将 RUN 中写入冗余追踪处理中的用户切换置为允许。对用户切换禁止状态进行了解除后，系统将自动地将 SM1709 置为 OFF。 (2) 由于下述原因导致系统切换时，与本继电器的状态无关，即使是在 RUN 中写入冗余追踪过程中也将进行系统切换。 • 电源 OFF、复位、H/W 故障、CPU 停止出错 (3) 在下述状态下，也可通过本继电器对系统切换禁止状态进行解除。 • 多个块 RUN 中写入冗余追踪执行中状态 • 文件批量 RUN 中写入冗余追踪执行中状态			S（请求时）/U		
SM1710	RUN 中写入冗余追踪执行中的软元件存储器热备传送有无	OFF ：不进行软元件存储器热备传送 ON ：进行软元件存储器热备传送	(1) 设置在 RUN 中写入冗余追踪执行中是否进行下述控制数据的热备传送。 • 软元件存储器 （包括自动进行热备传送的 SM/SD） • PIDINIT 信息 • S.PIDINIT 信息 • SFC 信息 (2) 通过 SM1710 也可设置在多个块 RUN 中写入冗余追踪、文件批量 RUN 中写入冗余追踪执行过程中是否进行热备传送。 (3) 通过热备传送将 SM1710 状态从控制系统 CPU 模块热备到待机系统 CPU 模块中。			U		
SM1712	传送触发结束标志	OFF ：传送未结束 ON ：传送结束	SM1712	块 1	相应块的传送结束时，相应标志仅 1 个扫描 ON。	S（状态变化）	新增	
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				

编号	名称	内容	详细内容			设置方（设置时机）	对应 ACPU M9□□□	对应 CPU
SM1723	传送触发结束标志	OFF ：传送未结束 ON ：传送结束	SM1723	块 12	相应块的传送结束时，相应标志仅 1 个扫描 ON。	S（状态变化）	新增	QnPRH
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				

编号	名称	内容	详细内容			设置方（设置时机）	对应 ACPU M9□□□	对应 CPU
SM1747	传送触发结束标志	OFF ：传送未结束 ON ：传送结束	SM1747	块 36	相应块的传送结束时，相应标志仅 1 个扫描 ON。	S（状态变化）	新增	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	传送触发结束标志	OFF ：传送未结束 ON ：传送结束	SM1772	块 61	相应块的传送结束时，相应标志仅 1 个扫描 ON。	S（状态变化）	新增	
SM1773			SM1773	块 62				
SM1774			SM1774	块 63				
SM1775			SM1775	块 64				

(15)冗余电源模块信息

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU M9□□□	对应 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	• 检测出存在有输入电源为 OFF 状态的冗余电源模块时该标志将变为 ON。 • SD1780 的某个位为 ON 时本继电器将变为 ON。 • SD1780 的所有位为 OFF 时本继电器也变为 OFF。 • 主基板不是冗余主基板（Q38RB）时，本继电器将变为 OFF。 • 多 CPU 系统配置时，标志仅被存储到 1 号机的 CPU 模块中。	S（每次 END）	新增	Qn(H)*2 QnPH*2 QnPRH QnU*3
SM1781	电源故障检测标志	OFF ：无故障的冗余电源模块 ON ：有故障的冗余电源模块	• 检测出冗余电源模块故障时本继电器将变为 ON。 • SD1781 的某个位为 ON 时本继电器将变为 ON。 • SD1781 的所有位为 OFF 时本继电器也变为 OFF。 • 主基板不是冗余主基板（Q38RB）时，本继电器将变为 OFF。 • 多 CPU 系统配置时，标志仅被存储到 1 号机的 CPU 模块中。	S（每次 END）		
SM1782	电源 1*1 用瞬间掉电检测标志	OFF ： 未检测出瞬间掉电 ON ： 检测出瞬间掉电	• 检测出至电源 1、2 的输入电源瞬间掉电时本继电器将变为 ON。变为 ON 后即使瞬间掉电已消除也仍将维持 ON 状态不变。 • CPU 模块启动时将电源 1、电源 2 的标志（SM1782、1783）置为 OFF。 • 至某一方的冗余电源模块的输入电源变为 OFF 状态时，输入电源 OFF 的冗余电源模块所对应的标志将变为 OFF。 • 主基板不是冗余主基板（Q38RB）时，本继电器将变为 OFF。 • 多 CPU 系统配置时，标志仅被存储到 1 号机的 CPU 模块中。	S（每次 END）		
SM1783	电源 2*1 用瞬间掉电检测标志					

*1: “电源 1”表示冗余基板（Q38RB/Q68RB/Q65WRB）的 POWER1 插槽中安装的冗余电源模块。
“电源 2”表示冗余基板（Q38RB/Q68RB/Q65WRB）的 POWER2 插槽中安装的冗余电源模块。

*2: 以序列号的前 5 位数为 “07032”以后的模块为对象。
但是，多 CPU 系统配置时，所有的 CPU 模块均以序列号的前 5 位数为 “07032”以后的模块为对象。

*3: 以序列号的前 5 位数为 “10042”以后的模块为对象。

附录 4　特殊寄存器一览表

特殊寄存器 SD 是可编程控制器内部的有固定规格的内部寄存器。因此，不能象普通的内部寄存器那样被用于顺控程序中。但是，根据需要，可以对其进行数据写入操作以控制 CPU 模块。

特殊寄存器中存储的数据在未特别指出的情况下将以 BIN 值进行存储。

一览表的各项目的阅读方法如附表 4.1 所示。

附表 4.1 特殊寄存器一览表的阅读方法

项目	项目说明
编号	• 表示特殊寄存器的编号。
名称	• 表示特殊寄存器的名称。
内容	• 表示特殊寄存器的有关内容。
详细内容	• 说明特殊寄存器的详细内容。
设置方 （设置时机）	• 说明设置方以及由系统设置时的设置时机。 〈设置方〉 S　：由系统设置。 U　：由用户（通过顺控程序或者外围设备的测试操作）进行设置。 S/U　：系统 / 用户均进行设置。 〈设置时机〉 仅在由系统设置时，表示设置时机。 每次 END　：在每次的 END 处理时进行设置。 初始化　：仅在初始化（电源 ON、STOP → RUN 等）时进行设置。 状态变化　：仅在状态发生了变化时进行设置。 发生出错　：仅在发生出错时进行设置。 执行指令　：仅在执行指令时进行设置。 请求时　：仅在有用户请求时（通过 SM 等）进行设置。 系统切换时　：在执行了系统切换时进行设置。
对应 ACPU D9□□□	• 表示对应于 ACPU 的特殊寄存器 (D9□□□)。 （内容有变更时，通过 D9□□□的状态变化来表示。不对应于 Q00J/Q00/Q01、QnPRH。） • 标为新增时，表示在 Q 系列 CPU 模块中新增的内容。
对应 CPU	表示对应的 CPU 模块。 QCPU　：对应于所有的 Q 系列 CPU 模块。 Q00J/Q00/Q01　：对应于基本型 QCPU。 Qn(H)　：对应于高性能型 QCPU。 QnPH　：对应于过程 CPU。 QnPRH　：对应于冗余 CPU。 QnU　：对应于通用型 QCPU。 各 CPU 模块型号：仅对应于所记述的 CPU 模块。（例：Q02U）

下述项目的详细内容请参阅以下手册。

- 网络相关　→　各网络模块的手册
- SFC 相关　→　QCPU(Q 模式)/QnACPU 编程手册 (SFC 篇)

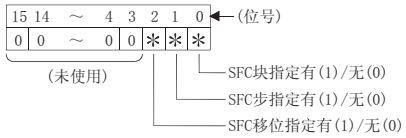
要 点

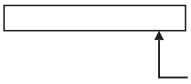
对于由系统设置的特殊寄存器，不要通过用户程序或软元件测试等操作进行变更。否则有可能导致系统宕机或者无法通信。

(1) 诊断信息

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU D9□□□	对应 CPU
SD0	诊断出错	诊断出错代码	• 通过诊断检测出出错时的出错代码以 BIN 码存储。 • 与故障履历的最新信息具有相同内容。	S（发生出错）	D9008 状态变化	
SD1	诊断出错发生时间	诊断出错发生时间	• SD0 的数据被更新时的年（公历，低 2 位）、月以 BCD 码 2 位数存储。 b15 ～ b8 b7 ～ b0 年(0～99) | 月(1～12) （例）95年10月 9510H	S（发生出错）	新增	
SD2			• SD0 的数据被更新时的日、时以 BCD 码 2 位数存储。 b15 ～ b8 b7 ～ b0 日(1～31) | 时(0～23) （例）25日10时 1510H			
SD3			• SD0 的数据被更新时的分、秒以 BCD 码 2 位数存储。 b15 ～ b8 b7 ～ b0 分(0～59) | 秒(0～59) （例）35分48秒 3548H			
SD4	出错信息分类	出错信息分类代码	分别存储到公共信息（SD5 ～ SD15）、个别信息（SD16 ～ SD26）中。 b15 ～ b8 b7 ～ b0 个别信息分类代码 | 公共信息分类代码 • 公共信息分类代码中存储下述代码。 0：无 1：模块号（插槽号 /CPU 机号 / 基板号）*1 2：文件名 / 驱动器名 3：时间（设置值） 4：程序出错位置 5：系统切换原因（冗余 CPU 专用） 6：热备容量超限出错原因（冗余 CPU 专用） 7：基板号 / 电源号（以序列号的前 5 位数为 “10042” 以后的通用型 QCPU 为对象。） 8：热备通信中的通信数据类型（冗余 CPU 专用） *1：对于包含基本型 QCPU、高性能型 QCPU、过程 CPU 和通用型 QCPU 的多 CPU 系统，根据发生的出错存储模块号或者 CPU 号。（究竟存储了哪个号请参阅各出错代码。） 1 号机：1； 2 号机：2； 3 号机：3； 4 号机：4 • 个别信息分类代码中存储下述代码。 0：无 1：（空闲） 2：文件名 / 驱动器名 3：时间（实测值） 4：程序出错位置 5：参数号 6：报警器 (F) 号 7：CHK 指令故障号（基本型 QCPU、通用型 QCPU 不支持） 8：系统切换失败原因（冗余 CPU 专用） 12：文件诊断信息（通用型 QCPU 专用） 13：参数号 /CPU 机号（通用型 QCPU 专用）	S（发生出错）	新增	QCPU

<table>
<tr><th>编号</th><th>名称</th><th>内容</th><th>详细内容</th><th>设置方
（设置时机）</th><th>对应
ACPU
D9□□□</th><th>对应 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>
・存储出错代码（SD0）对应的公共信息。
・存储的信息中有以下 8 种类型。
・根据 SD4 的 “公共信息分类代码”，可以判定出错公共信息的类型。
（SD4 中存储的 “公共信息分类代码”的值对应于下述 1）～ 8）。）
1)　模块号
<table>
<tr><th>编号</th><th>内容</th></tr>
<tr><td>SD5</td><td>插槽号/CPU机号/基板号*1、*2、*3、*4</td></tr>
<tr><td>SD6</td><td>I/O地址号*5</td></tr>
<tr><td>SD7
SD8
SD9
SD10
SD11
SD12
SD13
SD14
SD15</td><td>（空闲）</td></tr>
</table>
*1:　对于包含基本型 QCPU、高性能型 QCPU、过程 CPU 和通用型 QCPU 的多 CPU 系统，依据发生的出错存储插槽号或者 CPU 号。
（究竟存储了哪个号请参阅各出错代码。）
1 号机： 1； 2 号机： 2； 3 号机： 3； 4 号机： 4
*2:　如果在 MELSECNET/H 的远程 I/O 站中安装的模块上发生了保险丝熔断或者 I/O 校验出错，网络号将被存储到高 8 位，站号将被存储到低 8 位中。
通过 I/O 地址号可以确认发生保险丝熔断或者 I/O 校验出错的模块。
*3:　在基本型 QCPU 中，SD5 中存储了 255 时，表示对可实际安装的最终插槽后面的模块执行了指令。
*4:　基板号和插槽号的定义
〈基板号〉
此值用于识别安装了 CPU 模块的基板。基板号的定义如下所示。
<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>1～7</td><td>表示扩展基板。
通过扩展基板中的级数设置连接器进行的级数设置成为基板号。
级数设置扩展第1级 ： 基板号=1
级数设置扩展第7级 ： 基板号=7</td></tr>
</table>
〈插槽号〉
该值用于识别各基板的插槽及插槽中安装的模块。
・将主基板的 I/O 插槽 0(CPU 插槽右侧的插槽）定义为 “插槽号=0” 的插槽。
・对于插槽号，按照主基板、扩展基板第 1 级～扩展基板第 7 级的顺序，以连号方式分配给各基板的各插槽。
・在可编程控制器参数的 I/O 分配设置中进行了基板的插槽数设置时，仅分配相当于插槽数设置数的插槽号。
*5:　SD6(I/O 地址号）中存储了 FFFFH 时，表示由于可编程控制器参数的 I/O 分配设置中 I/O 地址号的重复等导致输入输出编号无法确定，因此通过 SD5 确定异常位置。
2)　文件名 / 驱动器名
<table>
<tr><th>编号</th><th colspan="2">内容</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
(ASCII码: 8字符)</td></tr>
<tr><td>SD10</td><td>扩展名 *6</td><td>2EH(.)</td></tr>
<tr><td>SD11</td><td colspan="2">(ASCII码: 3字符)</td></tr>
<tr><td>SD12
SD13
SD14
SD15</td><td colspan="2">(空闲)</td></tr>
</table>
(例)文件名=ABCDEFGH.IJK
<table>
<tr><th>b15～b8</th><th>b7～b0</th></tr>
<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
（发生出错）</td><td>新增</td><td>QCPU</td></tr>
</table>

<table>
<tr><th>编号</th><th>名称</th><th>内容</th><th>详细内容</th><th>设置方
（设置时机）</th><th>对应
ACPU
D9□□□</th><th>对应 CPU</th></tr>
<tr><td>SD5</td><td rowspan="11">出错公共信息</td><td rowspan="11">出错公共信息</td><td rowspan="11">3)　时间（设置值）
<table><tr><th>编号</th><th>内容</th></tr><tr><td>SD5</td><td>时间 ： 1μs单位(0～999μs)</td></tr><tr><td>SD6</td><td>时间 ： 1ms单位(0～65535ms)</td></tr><tr><td>SD7</td><td rowspan="9">(空闲)</td></tr><tr><td>SD8</td></tr><tr><td>SD9</td></tr><tr><td>SD10</td></tr><tr><td>SD11</td></tr><tr><td>SD12</td></tr><tr><td>SD13</td></tr><tr><td>SD14</td></tr><tr><td>SD15</td></tr></table>
4)　程序出错位置
<table><tr><th>编号</th><th>内容</th></tr><tr><td>SD5</td><td rowspan="4">文件名
(ASCII码：8字符)</td></tr><tr><td>SD6</td></tr><tr><td>SD7</td></tr><tr><td>SD8</td></tr><tr><td>SD9</td><td>扩展名*6　2EH(.)</td></tr><tr><td>SD10</td><td>(ASCII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/移位条件</td></tr><tr><td>SD14</td><td>顺控程序步号(L)</td></tr><tr><td>SD15</td><td>顺控程序步号(H)</td></tr></table>
*7：模式数据的内容
 
</td><td rowspan="11">S（发生出错）</td><td rowspan="11">新增</td><td rowspan="11">QCPU</td></tr>
<tr><td>SD6</td></tr>
<tr><td>SD7</td></tr>
<tr><td>SD8</td></tr>
<tr><td>SD9</td></tr>
<tr><td>SD10</td></tr>
<tr><td>SD11</td></tr>
<tr><td>SD12</td></tr>
<tr><td>SD13</td></tr>
<tr><td>SD14</td></tr>
<tr><td>SD15</td></tr>
</table>

<table>
<tr><th>编号</th><th>名称</th><th>内容</th><th>详细内容</th><th>设置方
（设置时机）</th><th>对应
ACPU
D9□□□</th><th>对应 CPU</th></tr>
<tr><td>SD5
SD6
SD7
SD8
SD9
SD10
SD11</td><td rowspan="2">出错公共信息</td><td rowspan="2">出错公共信息</td><td>5) 系统切换原因
<table><tr><th>编号</th><th>内容</th></tr><tr><td>SD5</td><td>系统切换原因 *13</td></tr><tr><td>SD6</td><td>控制系统切换指令变量</td></tr><tr><td>SD7
SD8
SD9
SD10
SD11
SD12
SD13
SD14
SD15</td><td>（空闲）</td></tr></table>*13：系统切换原因的内容
 
0 ： 无系统切换原因(默认)
1 ： 电源OFF、复位、硬件故障、看门狗定时器出错
2 ： 停止出错
（除看门狗定时器出错以外）
3 ： 来自于网络模块的系统切换请求
16 ： 控制系统切换指令
17 ： 来自于GX Developer的控制系统切换请求
6) 热备容量超限出错原因
超过了热备允许数据量（100k）时的块号通过相应的特殊继电器的位模式进行表示。
<table><tr><th></th><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>SD5</td><td>1
(SM1535)
(块16)</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1
(SM1528)
(块9)</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1
(SM1520)
(块1)</td></tr><tr><td>SD6</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>SD7</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>SD8</td><td>1
(SM1583)
(块64)</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1
(SM1568)
(块49)</td></tr><tr><td>SD9</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>SD15</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr></table></td><td>S（发生出错）</td><td>新增</td><td>QnPRH</td></tr>
<tr><td>SD12
SD13
SD14
SD15</td><td>7) 电源号
<table><tr><th>编号</th><th>内容</th></tr><tr><td>SD5</td><td>基板号</td></tr><tr><td>SD6</td><td>电源号</td></tr><tr><td>SD7
SD8
SD9
SD10
SD11
SD12
SD13
SD14
SD15</td><td>（空闲）</td></tr></table>1：电源1异常
2：电源2异常
“电源1”：冗余基板(Q38RB、Q68RB、Q65WRB)的POWER1插槽中安装的冗余电源模块
“电源2”：冗余基板(Q38RB、Q68RB、Q65WRB)的POWER2插槽中安装的冗余电源模块</td><td>S（发生出错）</td><td>新增</td><td>Qn(H)*1
QnPH*1
QnPRH
QnU*2</td></tr>
</table>

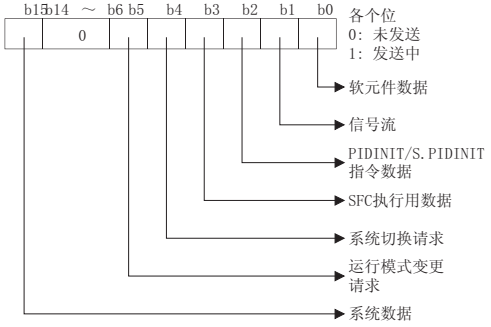
*1：以序列号的前 5 位数为“07032”以后的模块为对象。
*2：以序列号的前 5 位数为“10042”以后的模块为对象。

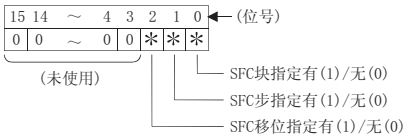
编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU D9□□□	对应 CPU
SD5						
SD6						
SD7						
SD8						
SD9						
SD10	出错公共信息	出错公共信息		S（发生出错）	新增	QnPRH
SD11						
SD12						
SD13						
SD14						
SD15						

8) 热备通信中的通信数据类型
存储热备通信中的通信数据类型

编号	内容
SD5	数据类型 *15
SD6	(空闲)
SD7	
SD8	
SD9	
SD10	
SD11	
SD12	
SD13	
SD14	
SD15	

*15: 数据类型的内容



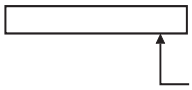
<table>
<tr><th>编号</th><th>名称</th><th>内容</th><th>详细内容</th><th>设置方
（设置时机）</th><th>对应
ACPU
D9□□□</th><th>对应 CPU</th></tr>
<tr><td>SD16
SD17
SD18
SD19
SD20
SD21
SD22
SD23
SD24
SD25
SD26</td><td>出错个别信息</td><td>出错个别信息</td><td>
・存储出错代码（SD0）对应的个别信息。
・存储的信息中有下述 10 种类型。
・根据 SD4 的 “个别信息分类代码”，可以判定出错个别信息的类型。
（SD4 中存储的 “个别信息分类代码”的值对应于下述 1）～ 8）、12）、13）。）
1） （空闲）
2） 文件名 / 驱动器名
<table>
<tr><th>编号</th><th>内容</th></tr>
<tr><td>SD16</td><td>驱动器</td></tr>
<tr><td>SD17
SD18
SD19
SD20</td><td>文件名
（ASCII码：8字符）</td></tr>
<tr><td>SD21</td><td>扩展名 *6　2EH(.)</td></tr>
<tr><td>SD22</td><td>（ASCII码：3字符）</td></tr>
<tr><td>SD23
SD24
SD25
SD26</td><td>（空闲）</td></tr>
</table>
（例）文件名=ABCDEFGH.IJK
<table>
<tr><th>b15～b8</th><th>b7～b0</th></tr>
<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>
3） 时间（实测值）
<table>
<tr><th>编号</th><th>内容</th></tr>
<tr><td>SD16</td><td>时间：1μs单位(0～999μs)</td></tr>
<tr><td>SD17</td><td>时间：1ms单位(0～65535ms)</td></tr>
<tr><td>SD18
SD19
SD20
SD21
SD22
SD23
SD24
SD25
SD26</td><td>（空闲）</td></tr>
</table>
4） 程序出错位置
<table>
<tr><th>编号</th><th>内容</th></tr>
<tr><td>SD16
SD17
SD18
SD19</td><td>文件名
（ASCII码：8字符）</td></tr>
<tr><td>SD20</td><td>扩展名*6　2EH(.)</td></tr>
<tr><td>SD21</td><td>（ASCII码：3字符）</td></tr>
<tr><td>SD22</td><td>模式*7</td></tr>
<tr><td>SD23</td><td>块号</td></tr>
<tr><td>SD24</td><td>步号/移位条件号</td></tr>
<tr><td>SD25</td><td>顺控程序步号(L)</td></tr>
<tr><td>SD26</td><td>顺控程序步号(H)</td></tr>
</table>
*7： 模式数据的内容


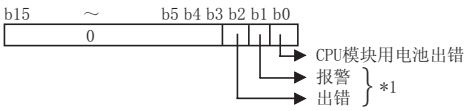
5）参数号　　6）报警器号
7）CHK指令故障号
<table>
<tr><th>编号</th><th>内容</th></tr>
<tr><td>SD16</td><td>参数号*16</td></tr>
<tr><td>SD17
SD18
SD19
SD20
SD21
SD22
SD23
SD24
SD25
SD26</td><td>（空闲）</td></tr>
</table>
<table>
<tr><th>编号</th><th>内容</th></tr>
<tr><td>SD16</td><td>No.</td></tr>
<tr><td>SD17
SD18
SD19
SD20
SD21
SD22
SD23
SD24
SD25
SD26</td><td>（空闲）</td></tr>
</table>
*16： 关于参数号的详细内容，请参阅所使用的 CPU 模块的用户手册（功能解说 / 程序基础篇）。
</td><td>S（发生出错）</td><td>新增</td><td>QCPU</td></tr>
</table>

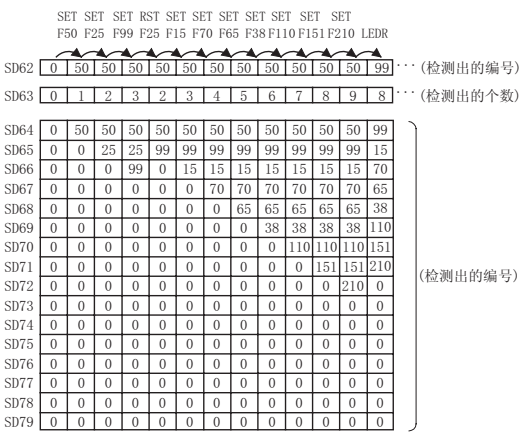
*6: 扩展名的名称如附表 4.2 所示。

附表 4.2 扩展名的名称

SDn	SDn+1		扩展名	文件的类型
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	软元件初始值
51H	44H	52H	QDR	文件寄存器
51H	44H	4CH	QDL	局部软元件 （基本型 QCPU 除外）
51H	54H	44H	QTD	采样跟踪数据 （基本型 QCPU 除外）
51H	46H	44H	QFD	故障履历数据 （基本型 QCPU、通用型 QCPU 除外）
51H	53H	54H	QST	SP.DEVST / S.DEVLD 指令用文件 （仅通用型 QCPU）

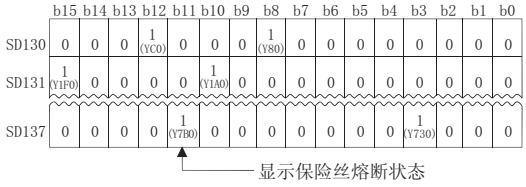
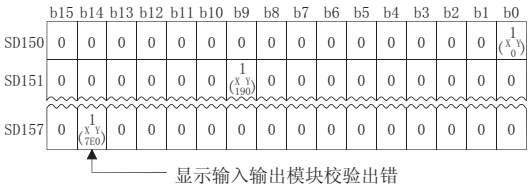
<table>
<tr><th>编号</th><th>名称</th><th>内容</th><th>详细内容</th><th>设置方
（设置时机）</th><th>对应
ACPU
D9□□□</th><th>对应 CPU</th></tr>
<tr><td rowspan="2">SD26</td><td rowspan="2">出错个别信息</td><td rowspan="2">出错个别信息</td><td>8)　系统切换失败原因
<table><tr><th>编号</th><th>内容</th></tr><tr><td>SD16</td><td>系统切换失败原因 *14</td></tr><tr><td>SD17</td><td rowspan="10">（空闲）</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>*14:　系统切换失败原因的内容
 
0 ： 正常切换结束(默认)
1 ： 热备电缆异常(电缆脱落、电缆异常、内部电路异常、硬件异常)
2 ： 待机系统硬件故障、电源OFF中、复位中、看门狗定时器出错发生中。
3 ： 控制系统中硬件故障、电源OFF中、复位中、看门狗定时器出错发生中。
4 ： 热备通信准备中
5 ： 超时
6 ： 待机系统停止出错(除看门狗定时器出错以外)
7 ： 两个系统的动作不相同(仅备份模式时)
8 ： 正在进行从控制系统至待机系统的存储器拷贝
9 ： 正在进行RUN中写入
10 ： 由待机系统的网络模块检测出异常
11 ： 正在执行系统切换
12 ： 正在进行在线模块更换</td><td rowspan="2">S（发生出错）</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1(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字符)</td></tr><tr><td>SD21
SD22</td><td>扩展名*6</td><td>2EH(.)</td></tr><tr><td colspan="3">(ASCII码：3字符)</td></tr><tr><td>SD23
SD24</td><td colspan="2">故障信息2
(读取的CRC值)</td></tr><tr><td>SD25
SD26</td><td colspan="2">故障信息3
(计算的CRC值)</td></tr></table>13)　参数号 /CPU 机号
<table><tr><th>编号</th><th>内容</th></tr><tr><td>SD16</td><td>参数号 *16</td></tr><tr><td>SD17</td><td>CPU机号(1～4)</td></tr><tr><td>SD18</td><td rowspan="10">（空闲）</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></td><td>QnU</td></tr>
<tr><td>SD50</td><td>出错解除</td><td>进行出错解除的出错代码</td><td>・存储进行出错解除的出错代码。</td><td>U</td><td>新增</td><td>QCPU</td></tr>
</table>

<table>
<tr><th>编号</th><th>名称</th><th>内容</th><th>详细内容</th><th>设置方
（设置时机）</th><th>对应
ACPU
D9□□□</th><th>对应 CPU</th></tr>
<tr><td>SD51</td><td>电池电压过低锁存</td><td>发生电池电压过低的对象的位模式</td><td>・发生了电池电压过低时，对应的位将变为 1(ON)。
・此后，即使电池电压恢复正常也仍将保持为 ON 状态不变。
 

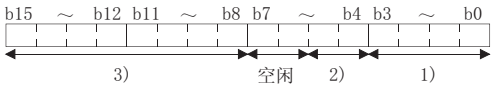
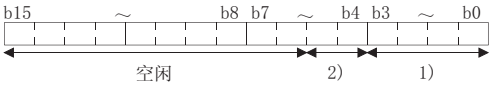
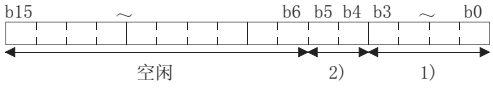
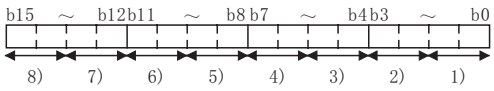
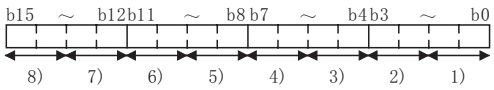
*1：在基本型QCPU中不支持。
・在报警中，电池电压过低时在规定的时间内数据可被保持。
・出错时表示电池被完全放电。</td><td>S（发生出错）</td><td>新增</td><td></td></tr>
<tr><td>SD52</td><td>电池电压过低</td><td>发生电池电压过低的对象的位模式</td><td>・与上述 SD51 的构成相同。
・检测出报警 (ON) 后，在检测出出错 (ON) 时报警将被 OFF。（仅通用型 QCPU）
・此后，如果电池电压恢复正常则该继电器将变为 0(OFF)。</td><td>S（发生出错）</td><td>新增</td><td></td></tr>
<tr><td>SD53</td><td>AC/DC DOWN 检测</td><td>检测出 AC/DC DOWN 的次数</td><td>・在 CPU 模块的运算过程中，输入电压低于额定值的 85%(AC 电源）/65%(DC 电源）时将被 +1，该值将以 BIN 码被存储。
・按 0 → 32767 → -32768 → 0 反复进行计数。</td><td>S（发生出错）</td><td>D9005</td><td></td></tr>
<tr><td>SD60</td><td>保险丝熔断模块号</td><td>保险丝熔断模块号</td><td>・存储保险丝熔断的模块的最小号的 I/O 地址号。</td><td>S（发生出错）</td><td>D9000</td><td></td></tr>
<tr><td>SD61</td><td>输入输出模块校验出错号</td><td>输入输出模块校验出错模块号</td><td>・存储发生了输入输出模块校验出错的模块的最小号的 I/O 地址号。</td><td>S（发生出错）</td><td>D9002</td><td></td></tr>
<tr><td>SD62</td><td>报警器号</td><td>报警器号</td><td>・存储最先检测出的报警器的编号（F 编号）。</td><td>S（指令执行）</td><td>D9009</td><td></td></tr>
<tr><td>SD63</td><td>报警器个数</td><td>报警器个数</td><td>・存储检测出的报警器个数。</td><td>S（指令执行）</td><td>D9124</td><td></td></tr>
<tr><td>SD64</td><td rowspan="16">检测出的报警器编号表</td><td rowspan="16">检测出的报警器编号</td><td rowspan="16">通过 OUT F、SET F 指令使 F 变为 ON 时，变为 ON 的 F 编号将依次被登录到 SD64 ～ SD79 中。
通过 RST F 指令变为 OFF 的 F 编号将从 SD64 ～ SD79 中被删除，存储在被删除的 F 编号后面的 F 编号将依次向前填充对齐。
通过 LEDR 指令的执行将 SD64 ～ SD79 的内容向前移动一位。检测出的报警器个数达到 16 个时，即使检测出第 17 个也不能被存储到 SD64 ～ SD79 中。
 
</td><td rowspan="16">S（指令执行）</td><td>D9125</td><td rowspan="16">QCPU</td></tr>
<tr><td>SD65</td><td>D9126</td></tr>
<tr><td>SD66</td><td>D9127</td></tr>
<tr><td>SD67</td><td>D9128</td></tr>
<tr><td>SD68</td><td>D9129</td></tr>
<tr><td>SD69</td><td>D9130</td></tr>
<tr><td>SD70</td><td>D9131</td></tr>
<tr><td>SD71</td><td>D9132</td></tr>
<tr><td>SD72</td><td>新增</td></tr>
<tr><td>SD73</td><td>新增</td></tr>
<tr><td>SD74</td><td>新增</td></tr>
<tr><td>SD75</td><td>新增</td></tr>
<tr><td>SD76</td><td>新增</td></tr>
<tr><td>SD77</td><td>新增</td></tr>
<tr><td>SD78</td><td>新增</td></tr>
<tr><td>SD79</td><td>新增</td></tr>
<tr><td>SD80</td><td>CHK 编号</td><td>CHK 编号</td><td>・通过 CHK 指令检测出的出错编号以 BCD 码被存储。</td><td>S（指令执行）</td><td>新增</td><td>Qn(H)
QnPH
QnPRH</td></tr>
</table>

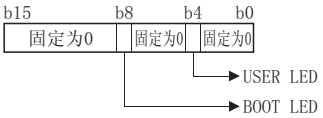
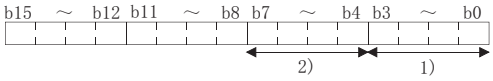
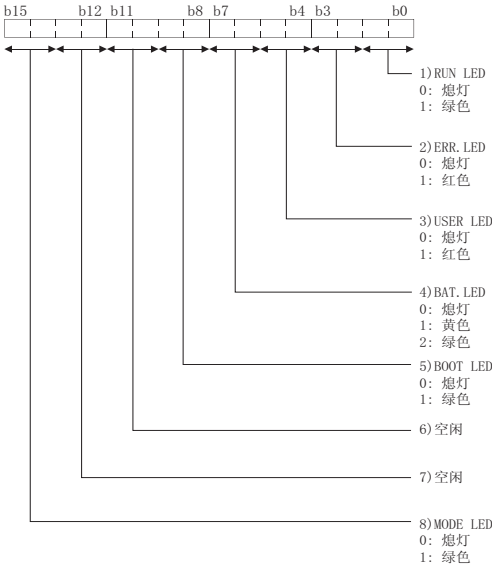
编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU D9□□□	对应 CPU
SD90	步移位监视定时器设置值（仅在有 SFC 程序时有效）	定时器设置值以及超时时的 F 编号	对应于 SM90 对应于 SM91 对应于 SM92 对应于 SM93 对应于 SM94 对应于 SM95 对应于 SM96 对应于 SM97 对应于 SM98 对应于 SM99	• 对步移位监视定时器的设置值以及监视定时器超时时变为 ON 的报警器的编号（F 编号）进行设置。 b15 ～ b8 b7 ～ b0 F编号的设置（0～255） 定时器时限的设置（1～255s：（1s单位）） • 通过将 SM90 ～ SM99 在激活步中变为 ON 以启动定时器，在定时器时限内相应步的下一个移位条件未成立时，设置的报警器（F）将变为 ON。	U	D9108 D9109 D9110 D9111 D9112 D9113 D9114 新增 新增 新增	Qn(H) QnPH QnPRH
SD100	传送速度存储区	存储串行通信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 ON、复位解除时）	新增	
SD101	通信设置存储区	存储串行通信设置中指定的通信设置	b15 ～ b6 b5 b4 b3 ～ b0 *1 *1 RUN中写入设置 0：禁止 1：允许 总数检查有无 0：无 1：有 *1：由于是系统所用，因此数据为不定值。		S（电源 ON、复位解除时）	新增	Q00/Q01 Q00UJ Q00U Q01U Q02U*4
SD102	传送等待时间存储区	存储串行通信设置中指定的传送等待时间	0 ：无等待时间 10 ～ 150：等待时间（单位：ms） 默认值为 0		S（电源 ON、复位解除时）	新增	
SD105	CH1 传送速度设置（RS-232）	存储使用 GX Developer 时的设置传送速度	96 ：9600bps、 192 ：19.2kbps、 384 ：38.4kbps、 576 ：57.6kbps、 1152 ：115.2kbps *：在除 RS-232 连接以外的情况下，将保持为 RS-232 连接时的数据。 （未连接时，默认值 =1152。）		S	新增	Qn(H) QnPH QnPRH QnU*3
SD110	数据传送结果存储区	存储使用串行通信功能时的数据发送结果	存储数据发送时的出错代码。		S（发生出错）	新增	Q00/Q01 Q00UJ Q00U Q01U Q02U*4
SD111	数据接收结果存储区	存储使用串行通信功能时的数据接收结果	存储数据接收时的出错代码。		S（发生出错）		
SD118	电池使用度	电池使用度	显示当前的电池使用度。 值的范围：1 ～ 2（Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU、Q03UD(E)CPU、Q04UD(E)HCPU） 1 ～ 3（Q06UD(E)HCPU） 1 ～ 4（Q10UD(E)HCPU、Q20UD(E)HCPU、Q13UD(E)HCPU、Q26UD(E)HCPU）		S（状态变化）	新增	QnU
SD119	电池长寿化原因	电池长寿化原因	存储电池长寿化功能有效的原因。 本 SD 为 0 以外时，处于电池长寿化功能有效状态。 0：无原因 1：有原因 b15 ～ b2 b1 b0 固定为0 b0：CPU的开关设置 b1：正在通过至标准ROM的锁存数据备份功能进行备份		S（状态变化）	新增	QnU

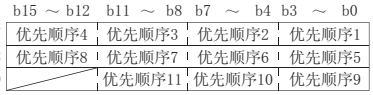
*3：以除以太网端口内置的 QCPU 以外的通用型 QCPU 为对象。
*4：以序列号的前 5 位数为“10102”以后的模块为对象。

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU D9□□□	对应 CPU
SD130 SD131 SD132 SD133 SD134 SD135 SD136 SD137	保险丝熔断模块	保险丝熔断模块的16点单位的位模式 0：无保险丝熔断 1：有保险丝熔断	• 变为保险丝熔断状态的输出模块编号（16点单位）以位模式输入。 （在参数中进行了设置时为所设置的编号）  • 即使变为正常也不会被清除。 通过出错解除进行清除。	S（发生出错）	新增	Q00J/Q00/Q01
SD150 SD151 SD152 SD153 SD154 SD155 SD156 SD157	输入输出模块校验出错	校验出错模块的16点单位的位模式 0：无输入输出校验出错 1：有输入输出校验出错	• 检测出与电源ON时登录的输入输出模块信息不相同的输入输出模块时，输入该输入输出模块编号（16点单位）。 （在参数中进行了设置时为所设置的输入输出模块编号）  • 即使变为正常也不会被清除。 通过出错解除进行清除。			

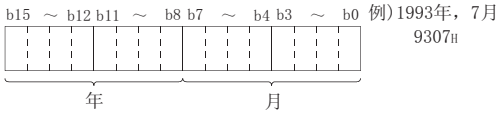
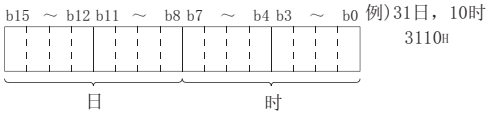
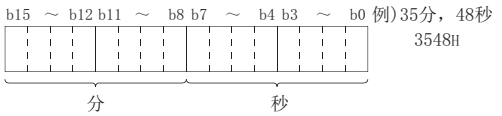
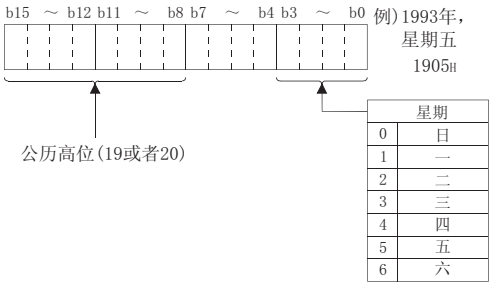
(2) 系统信息

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU D9□□□	对应 CPU
SD200	开关状态	CPU 开关状态	• 按以下格式存储 CPU 模块的开关状态。  1)：CPU开关状态 — 0：RUN 1：STOP 2：L.CLR 2)：存储卡开关 — 常时OFF 3)：插杆开关 — b8～b12与系统设置开关1的SW1～SW5相对应。0时为OFF，1时为ON。b13～b15为空闲。	S（每次 END）	新增	Qn(H) QnPH QnPRH
			• 按以下格式存储 CPU 模块的开关状态。  1)：CPU开关状态 — 0：RUN 1：STOP 2)：存储卡开关 — 常时OFF	S（每次 END）	新增	Q00J/Q00/Q01
			• 按以下格式存储 CPU 模块的开关状态。  1)：CPU开关状态 — 0：RUN 1：STOP 2)：存储卡开关 — 常时OFF	S (RUN/STOP/RESET 开关变化时)	新增	QnU
SD201	LED 状态	CPU-LED 状态	• 按下述位模式存储如下所示的 CPU 模块的 LED 状态。 • 为 0 时表示熄灯，为 1 时表示亮灯，为 2 时表示闪烁。  1)：RUN 5)：BOOT 2)：ERR. 6)：空闲 MODE 的位模式 3)：USER 7)：空闲 0：熄灯 1：绿色 4)：BAT. 8)：MODE 2：橙色 （在基本型 QCPU 中，无 3)～8)。）	S（状态变化）	新增	Q00J/Q00/Q01 Qn(H) QnPH QnPRH
			• 按下述位模式存储如下所示的 CPU 模块的 LED 状态。 • 为 0 时表示熄灯，为 1 时表示亮灯，为 2 时表示闪烁。  1)：RUN 5)：BOOT 2)：ERR. 6)：空闲 3)：USER 7)：空闲 4)：BAT. 8)：MODE （在 Q00UJCPU、Q00UCPU、Q01UCPU 中，没有 5)。）	S（状态变化）	新增	QnU

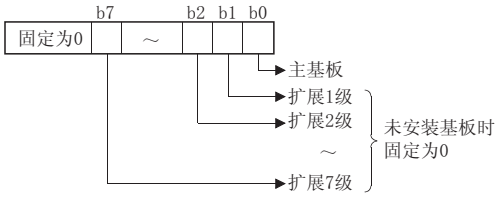
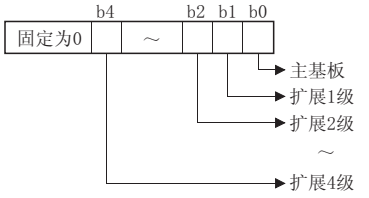
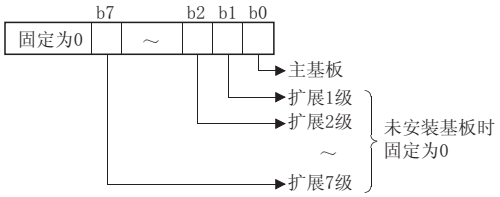
编号	名称	内容	详细内容	设置方（设置时机）	对应ACPU D9□□□	对应 CPU
SD202	LED 熄灯指令	熄灯的 LED 的位模式	• 指定通过本寄存器进行熄灯的 LED，通过将 SM202 由 OFF 变为 ON 使指定的 LED 熄灯。 可以将 USER、BOOT 指定为要熄灯的 LED。 • 通过下图的位模式指定要熄灯的 LED。 （为 1 时熄灯，为 0 时不熄灯。）  （在 Q00UJCPU、Q00UCPU、Q01UCPU 中，不能指定 BOOT LED。）	U	新增	Qn(H) QnPH QnPRH QnU
SD203	CPU 动作状态	CPU 动作状态	• 按下图所示存储 CPU 模块的动作状态。  1)：CPU动作状态 0：RUN 1：STEP-RUN(仅QnACPU) 2：STOP 3：PAUSE 2)：STOP/PAUSE原因 0：RUN/STOP开关(在基本型QCPU、通用型QCPU中为“RUN/STOP/RESET开关”) 注）先出现的优先存储（但是，在通用型QCPU中，存储使动作状态变更的最新的原因） 1：远程触点 2：通过GX Developer/串行通信等进行的远程操作 3：程序内的指令 4：出错	S（每次 END）	D9015 状态变化	QCPU
SD204	LED 显示色	CPU-LED 显示色	• 存储 SD201 1) ～ 8) 中所示的 LED 状态的 LED 显示色。  （在 Q00UJCPU、Q00UCPU、Q01UCPU 中没有 5)。）	S（状态变化）	新增	QnU

<table>
<tr><th>编号</th><th>名称</th><th>内容</th><th>详细内容</th><th>设置方
（设置时机）</th><th>对应
ACPU
D9□□□</th><th>对应 CPU</th></tr>
<tr><td>SD207</td><td rowspan="3">LED 显示优先顺序</td><td>优先顺序 1 ～ 4</td><td rowspan="3">• 通过出错项目号设置发生异常时，LED 显示的优先顺序。（在基本型 QCPU 中，只支持报警器（出错项目号 7）。）
• 在通用型 QCPU 中，设置是否执行发生异常时的各优先顺序相应的出错的 LED 显示。
• 优先顺序的设置区的情况如下所示。
 

（优先顺序11在使用冗余CPU时有效。）

默认值
SD207=4321H（在基本型 QCPU 中为 0000H）
SD208=8765H（在基本型 QCPU 中为 0700H）
（在冗余 CPU 中为 0765H）
SD209=00A9H（在基本型 QCPU 中为 0000H）
（在冗余 CPU 中为 0B09H）
• 设置为 “0” 时不显示。
• 在基本型 QCPU 中，如果在优先顺序 1 ～ 11 的某一个上设置了 “7”，则报警器 ON 时 ERR.LED 将亮灯。
• 在基本型 QCPU 中，如果在优先顺序 1 ～ 11 的任何一个位置上均未设置为 “7”，则及时报警器 ON 时 ERR.LED 也不亮灯。
但是，即使设置为 “0” 时，也将无条件地对 CPU 模块的运算停止（包括参数设置）出错进行 LED 显示。</td><td rowspan="3">U</td><td>D9038</td><td rowspan="3">Q00J/ Q00/ Q01 *9
Qn(H)
QnPH
QnPRH
QnU</td></tr>
<tr><td>SD208</td><td>优先顺序 5 ～ 8</td><td>D9039
状态变化</td></tr>
<tr><td>SD209</td><td>优先顺序 9 ～ 11</td><td>新增</td></tr>
<tr><td>SD210</td><td>时钟数据</td><td>时钟数据
（公历，月）</td><td>• 如下图所示以 BCD 码存储年（公历，低 2 位）、月。
 
</td><td rowspan="4">S（请求时）/U</td><td>D9025</td><td rowspan="4">QCPU</td></tr>
<tr><td>SD211</td><td>时钟数据</td><td>时钟数据
（日，时）</td><td>• 如下图所示以 BCD 码存储日、时。
 
</td><td>D9026</td></tr>
<tr><td>SD212</td><td>时钟数据</td><td>时钟数据
（分，秒）</td><td>• 如下图所示以 BCD 码存储分、秒。
 
</td><td>D9027</td></tr>
<tr><td>SD213</td><td>时钟数据</td><td>时钟数据
（公历高位，星期）</td><td>• 如下图所示以 BCD 码存储年（公历，高 2 位）及星期。
 
</td><td>S（请求时）/U</td><td>D9028</td></tr>
</table>

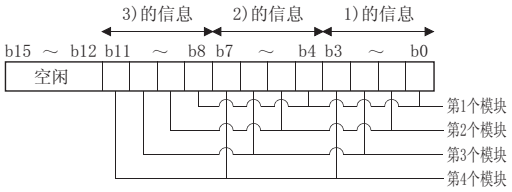
*9：以功能版本 B 以后为对象。

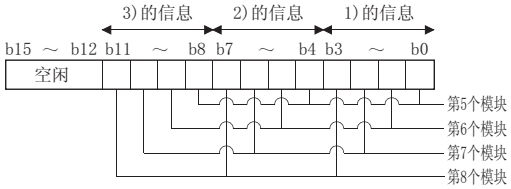
<table>
<tr><th>编号</th><th>名称</th><th>内容</th><th>详细内容</th><th>设置方
（设置时机）</th><th>对应
ACPU
D9□□□</th><th>对应 CPU</th></tr>
<tr><td>SD220
SD221
SD222
SD223
SD224
SD225
SD226
SD227</td><td>显示器数据</td><td>显示器数据</td><td>• 存储显示器的 ASCII 数据（16 字符）。
（在基本型 QCPU 中，存储发生出错（包括报警器 ON 时）时的信息（ASCII 数据 16 字符）。
b15 ～ b8 b7 ～ b0
SD220 右起第15个字符 右起第16个字符
SD221 右起第13个字符 右起第14个字符
SD222 右起第11个字符 右起第12个字符
SD223 右起第9个字符 右起第10个字符
SD224 右起第7个字符 右起第8个字符
SD225 右起第5个字符 右起第6个字符
SD226 右起第3个字符 右起第4个字符
SD227 右起第1个字符 右起第2个字符
• 在基本型 QCPU、通用型 QCPU 中，不存储 CHK 时的显示器数据。</td><td>S（变化时）</td><td>新增</td><td>QCPU</td></tr>
<tr><td>SD235</td><td>在线模块更换中模块</td><td>在线模块更换中模块的起始 I/O 地址号÷10H</td><td>• 存储在线模块更换中模块的起始 I/O 地址号÷10H。</td><td>S（实施在线模块更换时）</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</td></tr>
<tr><td>SD241</td><td>扩展级数</td><td>0：仅主基板
1～7：扩展级数</td><td>• 存储实际安装的扩展基板的最多级数。</td><td>S（初始化）</td><td>新增</td></tr>
<tr><td rowspan="3">SD242</td><td>A/Q 基板判别</td><td>基板类型的判别
0：安装了 QA**B（A 模式）
1：安装了 Q**B（Q 模式）</td><td> 
</td><td>S（初始化）</td><td>新增</td><td>Qn(H)
QnPH
QnPRH</td></tr>
<tr><td>Q 基板安装有无</td><td>基本类型判别
0：未安装基板
1：安装了 Q**B</td><td> 
</td><td>S（初始化）</td><td>新增</td><td>Q00J/Q00/Q01</td></tr>
<tr><td>Q 基板安装有无</td><td>基本类型判别
0：未安装基板
1：安装了 Q**B</td><td> 

• 在 Q00UJCPU 中，扩展 3 ～ 7 级固定为 0。
• 在 Q00UCPU、Q01UCPU、Q02UCPU 中，扩展 5 ～ 7 级固定为 0。</td><td>S（初始化）</td><td>新增</td><td>QnU</td></tr>
<tr><td>SD243
SD244</td><td>基板插槽个数</td><td>基板插槽个数</td><td>b15～b12 b11～b8 b7～b4 b3～b0
SD243 扩展3 扩展2 扩展1 主基板
SD244 扩展7 扩展6 扩展5 扩展4
• 在上述各区域中，存储实际安装基板的插槽个数。
• 在 Q00UJCPU 中，扩展 3 ～ 7 级固定为 0。
• 在 Q00UCPU、Q01UCPU、Q02UCPU 中，扩展 5 ～ 7 级固定为 0。</td><td>S（初始化）</td><td>新增</td><td>Qn(H)
QnPH
QnPRH
QnU</td></tr>
</table>

编号	名称	内容		详细内容	设置方（设置时机）	对应 ACPU D9□□□	对应 CPU
SD243 SD244	基板个数（动作状态）	基板插槽个数		b15～b12 b11～b8 b7～b4 b3～b0 SD243 扩展3 扩展2 扩展1 主基板 SD244 固定为0 固定为0 固定为0 扩展4 ・存储上述各区域中实际安装基板的插槽个数。（进行参数设置时设置的插槽个数）	S（初始化）	新增	Q00J/Q00/Q01
SD245 SD246	基板插槽个数（安装状态）	基板插槽个数		b15～b12 b11～b8 b7～b4 b3～b0 SD245 扩展3 扩展2 扩展1 主基板 SD246 固定为0 固定为0 固定为0 扩展4 ・存储上述各区域中基板安装状态的插槽个数（安装基板的实际插槽个数）。	S（初始化）		Q00J/Q00/Q01*9
SD250	实际安装最大 I/O 号	实际安装最大 I/O 地址号		・将 SM250 由 OFF 变为 ON 时，以 BIN 值存储实际安装模块的最终 I/O 地址号 +1 的高 2 位数。	S（请求 END）	新增	Qn(H) QnPH QnPRH
				・以 BIN 值存储实际安装模块的最终 I/O 地址号 +1 的高 2 位数。	S（初始化）	新增	Q00J/Q00/Q01 QnU
SD254	MELSECNET/10、MELSECNET/H 信息	安装个数		・显示所安装的 MELSECNET/10 模块或者 MELSECNET/H 模块的个数。	S（初始化）	新增	QCPU
SD255		第 1 个模块的信息	I/O 地址号	・显示所安装的 MELSECNET/10 模块或者 MELSECNET/H 模块的 I/O 地址号。			
SD256			网络号	・显示所安装的 MELSECNET/10 模块或者 MELSECNET/H 模块的网络号。			
SD257			组号	・显示所安装的 MELSECNET/10 模块或者 MELSECNET/H 模块的组号。			
SD258			站号	・显示所安装的 MELSECNET/10 模块或者 MELSECNET/H 模块的站号。			
SD259			待机信息	・待机站的情况下，存储是第几个模块的待机站。（1 ～ 4）			Qn(H) QnPH QnPRH QnU*10
SD260 ～ SD264		第 2 个模块信息		・其构成与第 1 个模块的相同。			
SD265 ～ SD269		第 3 个模块信息		・其构成与第 1 个模块的相同。			Qn(H) QnPH QnPRH QnU*11
SD270 ～ SD274		第 4 个模块信息		・其构成与第 1 个模块的相同。			

*9：以功能版本 B 以后为对象。
*10：以除 Q00UJCPU、Q00UCPU、Q01UCPU 以外的通用型 QCPU 为对象。
*11：以除 Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU 以外的通用型 QCPU 为对象。

<table>
<tr><th>编号</th><th>名称</th><th>内容</th><th>详细内容</th><th>设置方
（设置时机）</th><th>对应
ACPU
D9□□□</th><th>对应 CPU</th></tr>
<tr><td>SD280</td><td>CC-Link 出错</td><td>出错检测状态</td><td>1)： 安装的 CC-Link 模块的 Xn0 变为 ON 时，相应站的位将变为 1(ON)。
2)： 如果将所安装的 CC-Link 模块的 Xn1/XnF 中之一变为 OFF，则相应站的位将变为 1(ON)。
3)： 安装的 CC-Link 模块无法与 CPU 模块进行通信时，相应站的位将变为 1(ON)。
 

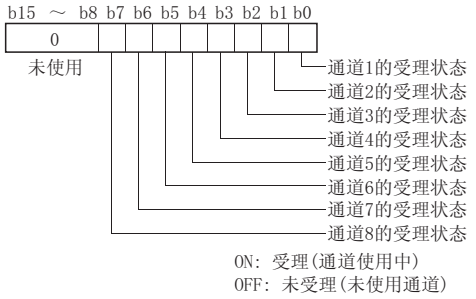
上述第 n 个模块是按照起始 I/O 地址号顺序进行编号的。
（但是，未进行参数设置的模块不计算在内。）</td><td rowspan="2">S（发生出错）</td><td rowspan="2">新增</td><td>Qn(H)
QnPH
QnPRH</td></tr>
<tr><td>SD281</td><td>CC-Link 出错</td><td>出错检测状态</td><td>1)： 安装的 CC-Link 模块的 Xn0 变为 ON 时，相应站的位将变为 1(ON)。
2)： 如果将所安装的 CC-Link 模块的 Xn1/XnF 中之一变为 OFF，则相应站的位将变为 1(ON)。
3)： 安装的 CC-Link 模块无法与 CPU 模块进行通信时，相应站的位将变为 1(ON)。
 

上述第 n 个模块是按照起始 I/O 地址号顺序进行编号的。
（但是，未进行参数设置的模块不计算在内。）</td><td>Qn(H)*14
QnPH*14
QnPRH*15</td></tr>
<tr><td>SD286</td><td rowspan="4">软元件分配</td><td rowspan="2">M 分配点数
（扩展用）</td><td rowspan="2">• 以 32 位存储 M 的分配点数。
• 即使 M 的分配点数为 32k 点以下，也进行点数分配。</td><td rowspan="4">S（初始化）</td><td rowspan="4">新增</td><td rowspan="4">QnU*16</td></tr>
<tr><td>SD287</td></tr>
<tr><td>SD288</td><td rowspan="2">B 分配点数
（扩展用）</td><td rowspan="2">• 以 32 位存储 B 的分配点数。
• 即使 B 的分配点数为 32k 点以下，也进行点数分配。</td></tr>
<tr><td>SD289</td></tr>
</table>

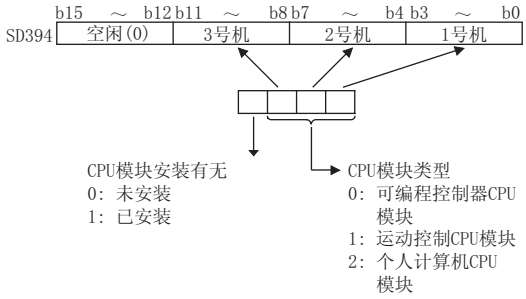
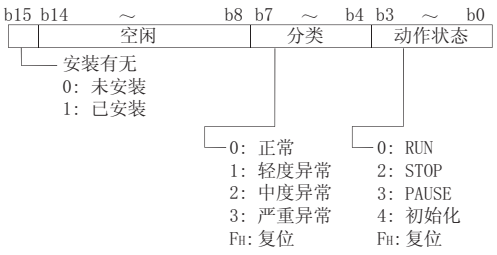
*14：以序列号的前 5 位数为 “08032”以后的模块为对象。
*15：以序列号的前 5 位数为 “09012”以后的模块为对象。
*16：以序列号的前 5 位数为 “10042”以后的模块为对象。

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU D9□□□	对应 CPU
SD290	软元件分配（与参数的内容相同）	X 分配点数	• 存储当前设置的软元件 X 的点数。	S（初始化）	新增	QCPU
SD291		Y 分配点数	• 存储当前设置的软元件 Y 的点数。			
SD292		M 分配点数	• 存储当前设置的软元件 M 的点数。			
SD293		L 分配点数	• 存储当前设置的软元件 L 的点数。			
SD294		B 分配点数	• 存储当前设置的软元件 B 的点数。			
SD295		F 分配点数	• 存储当前设置的软元件 F 的点数。			
SD296		SB 分配点数	• 存储当前设置的软元件 SB 的点数。			
SD297		V 分配点数	• 存储当前设置的软元件 V 的点数。			
SD298		S 分配点数	• 存储当前设置的软元件 S 的点数。			
SD299		T 分配点数	• 存储当前设置的软元件 T 的点数。			
SD300		ST 分配点数	• 存储当前设置的软元件 ST 的点数。			
SD301		C 分配点数	• 存储当前设置的软元件 C 的点数。			
SD302		D 分配点数	• 存储当前设置的软元件 D 的点数。			
SD303		W 分配点数	• 存储当前设置的软元件 W 的点数。			
SD304		SW 分配点数	• 存储当前设置的软元件 SW 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 位的范围进行修饰的编制寄存器 (Z) 的点数。（根据参数的 ZR 软元件的变址修饰设置）	S（初始化）	新增	QnU
SD306	软元件分配（与参数的内容相同）	ZR 分配点数（扩展）	• 存储 ZR 软元件的点数。 （扩展数据寄存器 (D)、扩展链接寄存器 (W) 的点数除外。） 只有在将扩展数据寄存器 (D)、扩展链接寄存器 (W) 设置为 1k 点以上时，才将 ZR 的分配点数存储到本 SD 中。	S（初始化）	新增	QnU*17
SD307						
SD308	软元件分配（包含扩展数据寄存器 (D)、扩展链接寄存器 (W) 设置在内的分配）	D 分配点数（内部 + 扩展）	• 存储内部软元件存储区的数据寄存器与扩展数据寄存器 (D) 的合计点数。（以 32 位的 BIN 值存储）			
SD309						
SD310		W 分配点数（内部 + 扩展）	• 存储内部软元件存储区的链接寄存器与扩展链接寄存器 (W) 的合计点数。（以 32 位的 BIN 值存储）			
SD311						
SD315	通信处理预留时间	通信处理预留时间	• 为与 GX Developer 等的通信处理时间预留出指定的时间。 • 指定的值越大，与其它设备 (GX Developer、串行通信模块等）通信的响应时间越快。 但是，扫描时间将延长相当于指定时间的量。 • 设置范围：1 ～ 100ms • 设置值超出了上述范围时，将作为无设置处理。	U	新增	Q00J/Q00/Q01 Qn(H) QnPH QnPRH

*17：以除 Q00UJCPU 以外的通用型 QCPU 为对象。

编号	名称	内容		详细内容	设置方（设置时机）	对应 ACPU D9□□□	对应 CPU
SD340	以太网信息	安装个数		• 显示安装的以太网模块的个数。	S（初始化）	新增	QCPU
SD341		第1个模块的信息	I/O 地址号	• 显示所安装的以太网模块的 I/O 地址号。			
SD342			网络号	• 显示所安装的以太网模块的网络号。			
SD343			组号	• 显示所安装的以太网模块的组号。			
SD344			站号	• 显示所安装的以太网模块的站号。			
SD345 ～ SD346			空闲	• 空闲 （在 QCPU 中将安装的第 1 个以太网模块的 IP 地址存储到缓冲存储器中。）			Qn(H) QnPH QnPRH QnU*10
SD347			空闲	• 空闲 （在 QCPU 中通过 ERRRD 指令读取安装的第 1 个以太网模块的出错代码。）			
SD348 ～ SD354	以太网信息	第 2 个模块信息		• 其构成与第 1 个模块的相同。	S（初始化）	新增	
SD355 ～ SD361		第 3 个模块信息		• 其构成与第 1 个模块的相同。			Qn(H) QnPH QnPRH QnU*11
SD362 ～ SD368		第 4 个模块信息		• 其构成与第 1 个模块的相同。			
SD380	以太网指令受理状态	第 1 个模块指令受理状态			S（执行指令）	新增	QnPRH
SD381		第 2 个模块指令受理状态		• 其构成与第 1 个模块的相同。			
SD382		第 3 个模块指令受理状态		• 其构成与第 1 个模块的相同。			
SD383		第 4 个模块指令受理状态		• 其构成与第 1 个模块的相同。			

*10：以除 Q00UJCPU、Q00UCPU、Q01UCPU 以外的通用型 QCPU 为对象。
*11：以除 Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU 以外的通用型 QCPU 为对象。

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU D9□□□	对应 CPU
SD393	多 CPU 系统信息	多 CPU 个数	• 存储构成多 CPU 系统的 CPU 模块的个数。（1 ～ 3，空闲也包含在内。）	S（初始化）	新增	Q00/Q01*9 QnU
SD394		CPU 安装信息	• 存储 1 ～ 3 号机的 CPU 模块类型及 CPU 模块安装有无。 	S（初始化）	新增	Q00/Q01*9
SD395		多 CPU 机号	• 存储多 CPU 系统配置时的自站 CPU 的机号。 1 号机：1； 2 号机：2； 3 号机：3； 4 号机：4	S（初始化）	新增	Q00/Q01*9 Qn(H)*9 QnPH QnU
SD396		1 号机动作状态	存储各机号的动作信息。 （存储相当于 SD393 中显示的多 CPU 个数的信息。） 	S （END 处理时、发生出错时）	新增	Q00/Q01*9 QnU
SD397		2 号机动作状态				Q00/Q01*9 QnU*17
SD398		3 号机动作状态				
SD399		4 号机动作状态				QnU*11

*9：以功能版本 B 以后为对象。
*11：以除 Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU 以外的通用型 QCPU 为对象。
*17：以除 Q00UJCPU 以外的通用型 QCPU 为对象。

(3) 系统时钟 / 计数器

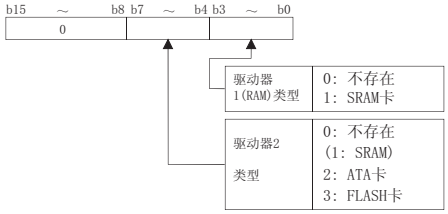
编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU D9□□□	对应 CPU
SD412	1 秒计数器	1 秒单位的计数数	• CPU 模块 RUN 后，每 1 秒进行 +1。 • 按照 0 → 32767 → -32768 → 0 的循环反复进行计数。	S（状态变化）	D9022	QCPU
SD414	2n 秒时钟设置	2n 秒时钟的单位	• 存储 2n 秒时钟的 n。（默认： 30） • 可设置范围为 1 ～ 32767。	U	新增	
SD415	2nms时钟设置	2nms时钟的单位	• 存储 2nms时钟的 n。（默认： 30） • 可设置范围为 1 ～ 32767。	U	新增	Qn(H) QnPH QnPRH QnU
SD420	扫描计数器	每个扫描的计数数	• CPU 模块 RUN 后，在扫描执行型程序的每 1 个扫描进行 +1。 （在初始执行型程序的扫描中不进行计数。） • 按照 0 → 32767 → -32768 → 0 的循环反复进行计数。	S（每次 END）	新增	Qn(H) QnPH QnPRH QnU
			• CPU 模块 RUN 后，在每 1 个扫描进行 +1。 • 按照 0 → 32767 → -32768 → 0 的循环反复进行计数。	S（每次 END）	新增	Q00J/Q00/Q01
SD430	低速扫描计数器	每个扫描的计数数	• CPU 模块 RUN 后，在低速执行型程序的每 1 个扫描进行 +1。 • 按照 0 → 32767 → -32768 → 0 的循环反复进行计数。 • 低速执行型程序专用	S（每次 END）	新增	Qn(H) QnPH

(4) 扫描信息

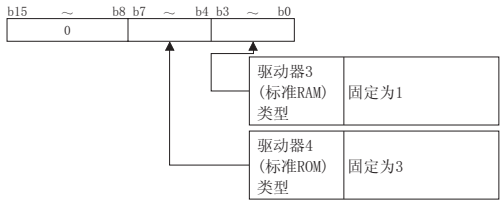
编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU D9□□□	对应 CPU
SD500	执行程序号	执行中的程序号	• 以 BIN 值存储当前正在执行的程序号。	S（状态变化）	新增	Qn(H) QnPH QnPRH QnU
SD510	低速执行型程序号	执行中的低速执行型程序号	• 以 BIN 值存储当前正在执行的低速执行型程序号。 • 仅在 SM510 为 ON 时有效。	S（每次 END）	新增	Qn(H) QnPH
SD520	当前扫描时间	当前扫描时间（ms 单位）	• 将当前的扫描时间存储到 SD520、SD521 中。 （计量是以 100μs 为单位进行。（在通用型 QCPU 中，是以 1μs 为单位进行。）） SD520：存储 ms 的位（存储范围：0 ～ 65535） SD521：存储μs 的位（存储范围：0 ～ 900（在通用型 QCPU 中存储范围为 0 ～ 999） （例）当前的扫描时间为 23.6 时，存储情况如下所示。 SD520=23 SD521=600	S（每次 END）	D9018 状态变化	QCPU
SD521		当前扫描时间（μs 单位）		S（每次 END）	新增	
SD522	初始扫描时间	初始扫描时间（ms 单位）	• 将初始执行型程序的扫描时间存储到 SD522、SD523 中。（计量是以 100μs 为单位进行。（在通用型 QCPU 中，是以 1μs 为单位进行。）） SD522：存储 ms 的位（存储范围：0 ～ 65535） SD523：存储μs 的位（存储范围：0 ～ 900（在通用型 QCPU 中存储范围为 0 ～ 999）	S(初次 END)	新增	Qn(H) QnPH QnPRH QnU
SD523		初始扫描时间（μs 单位）				
SD524	最小扫描时间	最小扫描时间（ms 单位）	• 除初始执行型程序的扫描时间以外，将扫描时间的最小值存储到 SD524、SD525 中。（计量是以 100μs 为单位进行。（在通用型 QCPU 中，是以 1μs 为单位进行。）） SD524：存储 ms 的位（存储范围：0 ～ 65535） SD525：存储μs 的位（存储范围：0 ～ 900（在通用型 QCPU 中存储范围为 0 ～ 999）	S（每次 END）	D9017 状态变化	
SD525		最小扫描时间（μs 单位）		S（每次 END）	新增	
SD526	最大扫描时间	最大扫描时间（ms 单位）	• 除初始执行型程序的扫描时间以外，将扫描时间的最大值存储到 SD526、SD527 中。（计量是以 100μs 为单位进行。（在通用型 QCPU 中，是以 1μs 为单位进行。）） SD526：存储 ms 的位（存储范围：0 ～ 65535） SD527：存储μs 的位（存储范围：0 ～ 900（在通用型 QCPU 中存储范围为 0 ～ 999））	S（每次 END）	D9019 状态变化	
SD527		最大扫描时间（μs 单位）			新增	
SD528	低速执行型程序用当前扫描时间	当前扫描时间（ms 单位）	• 将低速执行型程序的当前的扫描时间存储到 SD528、SD529 中。 （计量是以 100μs 为单位进行。） SD528：存储 ms 的位（存储范围：0 ～ 65535） SD529：存储μs 的位（存储范围：0 ～ 900）	S（每次 END）	新增	Qn(H) QnPH
SD529		当前扫描时间（μs 单位）				
SD532	低速执行型程序用最小扫描时间	最小扫描时间（ms 单位）	• 将低速执行型程序的扫描时间的最小值存储到 SD532、SD533 中。 （计量是以 100μs 为单位进行） SD532：存储 ms 的位（存储范围：0 ～ 65535） SD533：存储μs 的位（存储范围：0 ～ 900）	S（每次 END）	新增	
SD533		最小扫描时间（μs 单位）				
SD534	低速执行型程序用最大扫描时间	最大扫描时间（ms 单位）	• 除低速执行型程序的第 1 个扫描以外，将扫描时间的最大值存储到 SD534、SD535 中。 （计量是以 100μs 为单位进行） SD534：存储 ms 的位（存储范围：0 ～ 65535） SD535：存储μs 的位（存储范围：0 ～ 900）	S（每次 END）	新增	
SD535		最大扫描时间（μs 单位）				
SD540	END 处理时间	END 处理时间（ms 单位）	• 扫描执行型程序结束后，将至下一个扫描开始为止的时间存储到 SD540、SD541 中。 （计量是以 100μs 为单位进行。（在通用型 QCPU 中，是以 1μs 为单位进行。）） SD540：存储 ms 的位（存储范围：0 ～ 65535） SD541：存储μs 的位（存储范围：0 ～ 900（在通用型 QCPU 中存储范围为 0 ～ 999））	S（每次 END）	新增	Qn(H) QnPH QnPRH QnU
SD541		END 处理时间（μs 单位）				

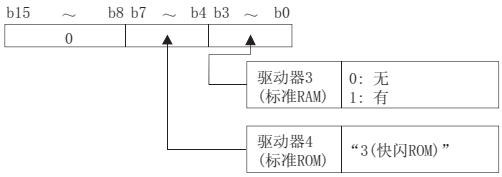
编号	名称	内容	详细内容	设置方（设置时机）	对应ACPU D9□□□	对应 CPU
SD524	最小扫描时间	最小扫描时间（ms 单位）	• 将扫描时间的最小值存储到 SD524、SD525 中。 （计量是以 100μs 为单位进行） SD524：存储 ms 的位（存储范围：0 ～ 65535） SD525：存储μs 的位（存储范围：0 ～ 900）	S（每次 END）	新增	Q00J/Q00/Q01
SD525		最小扫描时间（μs 单位）				
SD526	最大扫描时间	最大扫描时间（ms 单位）	• 将扫描时间的最大值存储到 SD526、SD527 中。 （计量是以 100μs 为单位进行） SD526：存储 ms 的位（存储范围：0 ～ 65535） SD527：存储μs 的位（存储范围：0 ～ 900）	S（每次 END）		
SD527		最大扫描时间（μs 单位）				
SD540	END 处理时间	END 处理时间（ms 单位）	• 扫描执行型程序结束后，将至下一个扫描开始为止的时间存储到 SD540、SD541 中。 （计量是以 100μs 为单位进行） SD540：存储 ms 的位（存储范围：0 ～ 65535） SD541：存储μs 的位（存储范围：0 ～ 900）	S（每次 END）		
SD541		END 处理时间（μs 单位）				
SD542	恒定扫描等待时间	恒定扫描等待时间（ms 单位）	• 将恒定扫描设置时的等待时间存储到 SD542、SD543 中。 （计量是以 100μs 为单位进行（在通用型 QCPU 中，是以 1μs 为单位进行）） SD542：存储 ms 的位（存储范围：0 ～ 65535） SD543：存储μs 的位（存储范围：0 ～ 900（在通用型 QCPU 中存储范围为 0 ～ 999））	S（每次 END）	新增	QCPU
SD543		恒定扫描等待时间（μs 单位）				
SD544	低速执行型程序累计执行时间	低速执行型程序累计执行时间（ms 单位）	• 将低速执行型程序的累计时间存储到 SD544、SD545 中。 （计量是以 100μs 为单位进行） SD544：存储 ms 的位（存储范围：0 ～ 65535） SD545：存储μs 的位（存储范围：0 ～ 900） • 低速 1 个扫描结束后将被清 0。	S（每次 END）	新增	Qn(H) QnPH
SD545		低速执行型程序累计执行时间（μs 单位）				
SD546	低速执行型程序执行时间	低速执行型程序执行时间（ms 单位）	• 将 1 个扫描的低速执行型程序的的执行时间存储到 SD546、SD547 中。 （计量是以 100μs 为单位进行） SD546：存储 ms 的位（存储范围：0 ～ 65535） SD547：存储μs 的位（存储范围：0 ～ 900） • 每个扫描被存储。	S（每次 END）	新增	
SD547		低速执行型程序执行时间（μs 单位）				
SD548	扫描执行型程序执行时间	扫描执行型程序执行时间（ms 单位）	• 将 1 个扫描中的扫描执行型程序的执行时间存储到 SD548、SD549 中。 （计量是以 100μs 为单位进行） SD548：存储 ms 的位（存储范围：0 ～ 65535） SD549：存储μs 的位（存储范围：0 ～ 900） • 每个扫描被存储。	S（每次 END）	新增	Qn(H) QnPH QnPRH
SD549		扫描执行型程序执行时间（μs 单位）				
SD548	扫描程序执行时间	扫描程序执行时间（ms 单位）	• 将 1 个扫描中的扫描程序执行时间存储到 SD548、SD549 中。 （计量是以 100μs 为单位进行（在通用型 QCPU 中，是以 1μs 为单位进行）） SD548：存储 ms 的位（存储范围：0 ～ 65535） SD549：存储μs 的位（存储范围：0 ～ 900（在通用型 QCPU 中存储范围为 0 ～ 999）） • 每个扫描被存储。	S（每次 END）	新增	Q00J/Q00/Q01 QnU
SD549		扫描程序执行时间（μs 单位）				
SD550	服务间隔测定模块	模块号	• 设置对服务间隔进行测定的模块的 I/O 地址号。	U	新增	Qn(H) QnPH QnPRH
SD551	服务间隔时间	模块服务间隔（ms 单位）	• 将 SM551 置为 ON 时，将 SD550 中指定模块的服务间隔存储到 SD551、SD552 中。 （计量是以 100μs 为单位进行） SD551：存储 ms 的位（存储范围：0 ～ 65535） SD552：存储μs 的位（存储范围：0 ～ 900）	S（请求时）	新增	
SD552		模块服务间隔（μs 单位）				

(5) 驱动器信息

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU D9□□□	对应 CPU
SD600	存储卡类型	存储卡类型	• 显示所安装的存储卡的类型。  （在 Q00UJCPU、Q00UCPU、Q01UCPU 中，驱动器 1(RAM) 类型、驱动器 2(ROM) 类型固定为 0。）	S（初始化以及卡脱落时）	新增	Qn(H) QnPH QnPRH QnU
SD602	驱动器 1（存储卡 RAM）容量	驱动器 1 的容量	• 以 1k 字节单位存储驱动器 1 的容量。（存储格式化后的可用空间。）	S（初始化以及卡脱落时）	新增	Qn(H) QnPH QnPRH QnU*2
SD603	驱动器 2（存储卡 ROM）容量	驱动器 2 的容量	• 以 1k 字节单位存储驱动器 2 的容量。*1	S（初始化以及卡脱落时）	新增	
SD604	存储卡使用状况	存储卡使用状况	• 以位模式存储存储卡的使用状况。(ON 时表示使用中。) • 各位模式的含义如下所示。 b0：引导运行(QBT) b1：参数(QPA) b2：软元件注释(QCD) b3：软元件初始值(QDI) b4：文件寄存器(QDR) b5：采样跟踪(QTD) b6：未使用 b7：未使用 b8：未使用 b9：CPU故障履历(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（状态变化）	新增	Qn(H) QnPH QnPRH
SD604	存储卡使用状况	存储卡使用状况	• 以位模式存储存储卡的使用状况。(ON 时表示使用中。) • 各位模式的含义如下所示。 b0：引导运行(QBT)*1 b1：参数(QPA) b2：软元件注释(QCD) b3：软元件初始值(QDI)*2 b4：文件寄存器(QDR) b5：采样跟踪(QTD) b6：未使用 b7：备份数据(QBP)*3 b8：未使用 b9：未使用 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 *1：引导开始时变为 ON，结束时变为 OFF。 *2：软元件初始值反映开始时变为 ON，结束时变为 OFF。 *3：以序列号的前 5 位数为 “10102” 以后的模块为对象。	S（状态变化）	新增	QnU*2

*1： 使用了 Q2MEM-8MBA 时，根据 ATA 卡的生产管理编号，特殊寄存器 SD603 中存储的值有所不同。
详细内容请参阅 QCPU 用户手册（硬件设计 / 维护点检篇）。
*2： 以除 Q00UJCPU、Q00UCPU、Q01UCPU 以外的通用型 QCPU 为对象。

<table>
<tr><th>编号</th><th>名称</th><th>内容</th><th>详细内容</th><th>设置方
（设置时机）</th><th>对应
ACPU
D9□□□</th><th>对应 CPU</th></tr>
<tr><td rowspan="2">SD620</td><td rowspan="2">驱动器 3/4 类型</td><td rowspan="2">驱动器 3/4 类型</td><td>・显示驱动器 3/4 类型。
 

（在 Q00UJCPU 中，驱动器 3（标准 RAM）类型固定为 0。）</td><td>S（初始化）</td><td>新增</td><td>Qn(H)
QnPH
QnPRH
QnU</td></tr>
<tr><td>・显示驱动器 3/4 的类型。
 
</td><td>S（初始化）</td><td>新增</td><td>Q00J/Q00/Q01</td></tr>
<tr><td>SD622</td><td>驱动器 3
（标准 RAM）容量</td><td>驱动器 3 的容量</td><td>・以 1k 字节单位存储驱动器 3 的容量。
（存储格式化后的可用空间。）</td><td>S（初始化）</td><td>新增</td><td>Qn(H)
QnPH
QnPRH
QnU</td></tr>
<tr><td>SD623</td><td>驱动器 4
（标准 ROM）容量</td><td>驱动器 4 的容量</td><td>・以 1k 字节单位存储驱动器 4 的容量。
（存储格式化后的可用空间。）</td><td>S（初始化）</td><td>新增</td><td>Qn(H)
QnPH
QnPRH
QnU</td></tr>
<tr><td rowspan="2">SD624</td><td>驱动器 3/4 使用状况</td><td>驱动器 3/4 使用状况</td><td>・以位模式存储驱动器 3/4 的使用状况。
(ON 时表示使用中）
・各位模式的含义如下所示。
<table><tr><td>b0：引导运行(QBT)</td><td>b8：未使用</td></tr><tr><td>b1：参数(QPA)</td><td>b9：CPU故障履历(QFD)</td></tr><tr><td>b2：软元件注释(QCD)</td><td>b10：SFC跟踪(QTS)</td></tr><tr><td>b3：软元件初始值(QDI)</td><td>b11：局部软元件(QDL)</td></tr><tr><td>b4：文件寄存器(QDR)</td><td>b12：未使用</td></tr><tr><td>b5：采样跟踪(QTD)</td><td>b13：未使用</td></tr><tr><td>b6：未使用</td><td>b14：未使用</td></tr><tr><td>b7：未使用</td><td>b15：未使用</td></tr></table></td><td>S（状态变化）</td><td>新增</td><td>Qn(H)
QnPH
QnPRH</td></tr>
<tr><td>驱动器 3/4 使用状况</td><td>驱动器 3/4 使用状况</td><td>・以位模式存储驱动器 3/4 的使用状况。
(ON 时表示使用中）
・各位模式的含义如下所示。
<table><tr><td>b0：未使用</td><td>b8：未使用</td></tr><tr><td>b1：参数(QPA)</td><td>b9：未使用</td></tr><tr><td>b2：软元件注释(QCD)</td><td>b10：未使用</td></tr><tr><td>b3：软元件初始值(QDI)*1</td><td>b11：局部软元件(QDL)</td></tr><tr><td>b4：文件寄存器(QDR)</td><td>b12：未使用</td></tr><tr><td>b5：采样跟踪(QTD)</td><td>b13：未使用</td></tr><tr><td>b6：未使用</td><td>b14：未使用</td></tr><tr><td>b7：未使用</td><td>b15：未使用</td></tr></table>*1：反映软元件初始值时变为 ON，结束时变为 OFF。</td><td>S（状态变化）</td><td>新增</td><td>QnU</td></tr>
</table>

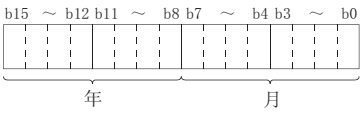
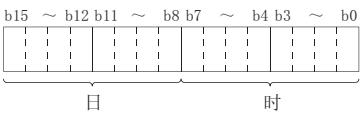
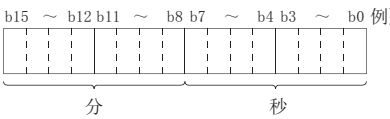
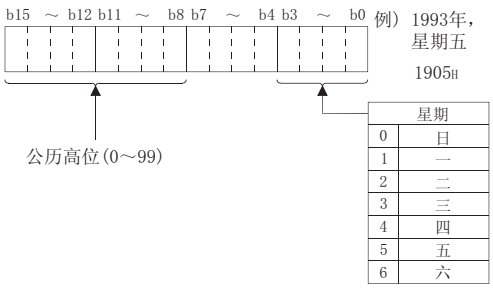
编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU D9□□□	对应 CPU
SD622	驱动器 3（标准 RAM）容量	驱动器 3 的容量	• 以 1k 字节单位存储驱动器 3 的容量。	S（初始化）	新增	Q00J/Q00/Q01
SD623	驱动器 4（标准 ROM）容量	驱动器 4 的容量	• 以 1k 字节单位存储驱动器 4 的容量。	S（初始化）		
SD624	驱动器 3/4 使用状况	驱动器 3/4 使用状况	• 以位模式存储驱动器 3/4 的使用状况。 b15 ～ b5 b4 ～ b0 0 | 0 | 0 | 0 | 0 | 0 引导运行(QBT) 0：未使用 1：使用中 文件寄存器(QDR) 0：未使用 1：使用中	S（状态变化）		
SD640	文件寄存器驱动器	驱动器编号	• 存储文件寄存器中使用的驱动器编号。	S（状态变化）*10	新增	Q00J/Q00/Q01 Qn(H) QnPH QnPRH QnU*3
SD641 SD642 SD643	文件寄存器文件名	文件寄存器文件名	• 以 ASCII 码存储参数或者 QDRSET 指令中选择的文件寄存器的文件名（带扩展名）。 SD641：b15～b8 第2个字符，b7～b0 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：扩展名第1个字符，2EH(.) SD646：扩展名第3个字符，扩展名第2个字符	S（状态变化）	新增	Qn(H) QnPH QnPRH QnU*3
SD644 SD645 SD646			• 以 ASCII 码存储文件寄存器的文件名 (MAIN.QDR)。 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：扩展名第1个字符(Q)，2EH(.) SD646：扩展名第3个字符(R)，扩展名第2个字符(D)	S（初始化）	新增	Q00J/Q00/Q01
SD647	文件寄存器容量	文件寄存器容量	• 以 1k 字节单位存储当前选择的文件寄存器的数据容量。	S（状态变化）	新增	Qn(H) QnPH QnPRH QnU*3
				S（初始化）		Q00J/Q00/Q01
SD648	文件寄存器块号	文件寄存器块号	• 存储当前选择的文件寄存器的块号。	S（状态变化）*10	D9035	Q00J/Q00/Q01 Qn(H) QnPH QnPRH QnU*3

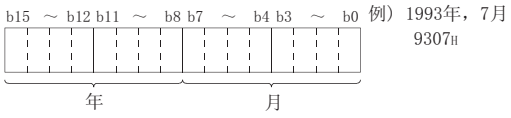
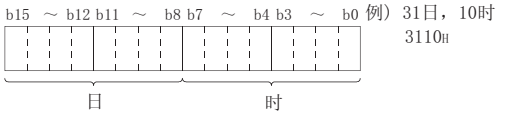
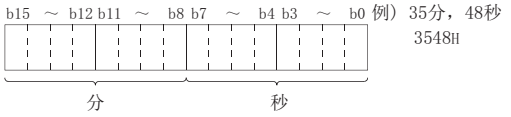
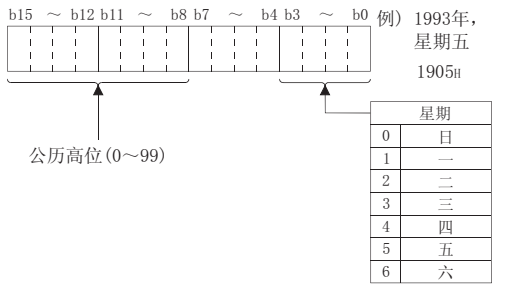
*3: 以除 Q00UJCPU 以外的通用型 QCPU 为对象。

*10: 在基本型 QCPU 中，在参数执行后的 STOP → RUN 或者 RSET 指令执行时数据将被设置。

<table>
<tr><th>编号</th><th>名称</th><th>内容</th><th>详细内容</th><th>设置方
（设置时机）</th><th>对应
ACPU
D9□□□</th><th>对应 CPU</th></tr>
<tr><td>SD650</td><td>注释驱动器</td><td>注释驱动器编号</td><td>• 存储通过参数或者 QCDSET 指令选择的注释的驱动器编号。</td><td>S（状态变化）</td><td>新增</td><td rowspan="7">Qn(H)
QnPH
QnPRH
QnU</td></tr>
<tr><td>SD651</td><td rowspan="6">注释文件名</td><td rowspan="6">注释文件名</td><td rowspan="6">• 以 ASCII 码（带扩展名）存储通过参数或者 QCDSET 指令选择的注释的文件名。
<table><tr><th></th><th>b15 ～ b8</th><th>b7 ～ b0</th></tr><tr><td>SD651</td><td>第2个字符</td><td>第1个字符</td></tr><tr><td>SD652</td><td>第4个字符</td><td>第3个字符</td></tr><tr><td>SD653</td><td>第6个字符</td><td>第5个字符</td></tr><tr><td>SD654</td><td>第8个字符</td><td>第7个字符</td></tr><tr><td>SD655</td><td>扩展名第1个字符</td><td>2EH(.)</td></tr><tr><td>SD656</td><td>扩展名第3个字符</td><td>扩展名第2个字符</td></tr></table></td><td rowspan="6">S（状态变化）</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>
<tr><td>SD660</td><td rowspan="7">引导运行指定文件</td><td>引导指定文件驱动器编号</td><td>• 存储储存了引导指定文件（*.QBT）的驱动器编号。</td><td>S（初始化）</td><td>新增</td><td rowspan="7">Qn(H)
QnPH
QnPRH
QnU*4</td></tr>
<tr><td>SD661</td><td rowspan="6">引导指定文件文件名</td><td rowspan="6">• 存储引导指定文件（*.QBT）的文件名。
<table><tr><th></th><th>b15 ～ b8</th><th>b7 ～ b0</th></tr><tr><td>SD661</td><td>第2个字符</td><td>第1个字符</td></tr><tr><td>SD662</td><td>第4个字符</td><td>第3个字符</td></tr><tr><td>SD663</td><td>第6个字符</td><td>第5个字符</td></tr><tr><td>SD664</td><td>第8个字符</td><td>第7个字符</td></tr><tr><td>SD665</td><td>扩展名第1个字符</td><td>2EH(.)</td></tr><tr><td>SD666</td><td>扩展名第3个字符</td><td>扩展名第2个字符</td></tr></table></td><td rowspan="6">S（初始化）</td><td rowspan="6">新增</td></tr>
<tr><td>SD662</td></tr>
<tr><td>SD663</td></tr>
<tr><td>SD664</td></tr>
<tr><td>SD665</td></tr>
<tr><td>SD666</td></tr>
</table>

*4：以除 Q00UJCPU、Q00UCPU、Q01UCPU 以外的通用型 QCPU 为对象。

<table>
<tr><th>编号</th><th>名称</th><th>内容</th><th>详细内容</th><th>设置方
（设置时机）</th><th>对应
ACPU
D9□□□</th><th>对应 CPU</th></tr>
<tr><td>SD670</td><td>参数有效驱动器信息</td><td>参数有效驱动器号</td><td>・存储有效的参数存储目标驱动器的信息。
0：驱动器 0（程序存储器）
1：驱动器 1(SRAM 卡）
2：驱动器 2(Flash 卡 /ATA 卡）
4：驱动器 4（标准 ROM）
（在 Q00UJCPU、Q00UCPU、Q01UCPU，仅驱动器 0、驱动器 4 有效。）</td><td>S（初始化）</td><td>新增</td><td rowspan="5">QnU</td></tr>
<tr><td>SD671</td><td>锁存数据备份功能状态</td><td>状态显示</td><td>显示数据备份功能的状态。
<table>
<tr><th colspan="2">状态</th><th>备份数据有无</th><th>此后电源OFF→ON时的恢复动作</th></tr>
<tr><td>0</td><td>无备份数据</td><td>无</td><td>不进行恢复</td></tr>
<tr><td>1</td><td>恢复准备结束</td><td rowspan="4">有</td><td>仅在下一次电源OFF→ON时进行恢复。</td></tr>
<tr><td>2</td><td>恢复执行结束</td><td>不进行恢复</td></tr>
<tr><td>3</td><td>备份执行等待</td><td>不进行恢复</td></tr>
<tr><td>4</td><td>恢复重复执行准备结束</td><td>在每次电源OFF→ON时进行恢复。</td></tr>
</table>
・“2：恢复执行结束”是恢复执行之后的状态。
“3：备份执行等待”是“2：恢复执行结束”时电源 OFF → ON 后的状态。</td><td>S（状态变化）</td><td rowspan="4">新增</td></tr>
<tr><td>SD672</td><td rowspan="4">备份信息</td><td>备份时间
（公历，月）</td><td>・以 BCD 码 2 位数存储备份的年（公历，低 2 位）、月。
 
</td><td rowspan="4">S（写入时）</td></tr>
<tr><td>SD673</td><td>备份时间
（日、时）</td><td>・以 BCD 码 2 位数存储备份的日、时。
 
</td></tr>
<tr><td>SD674</td><td>备份时间
（分、秒）</td><td>・以 BCD 码 2 位数存储备份的分、秒。
 
</td></tr>
<tr><td>SD675</td><td>备份时间
（公历高位、星期）</td><td>・以 BCD 码存储备份的年（公历，高 2 位）、星期。
 
</td></tr>
</table>

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU D9□□□	对应 CPU
SD676	备份数据恢复信息	恢复时间（公历，月）	• 以 BCD 码 2 位数存储恢复的年（公历，低 2 位）、月。  	S（初始化）	新增	QnU
SD677		恢复时间（日、时）	• 以 BCD 码 2 位数存储恢复的日、时。  			
SD678		恢复时间（分、秒）	• 以 BCD 码 2 位数存储恢复的分、秒。  			
SD679		恢复时间（公历高位、星期）	• 以 BCD 码存储恢复的年（公历，高 2 位）、星期。  			
SD681	程序存储器写入（传送）状况	写入（传送）状况显示（百分比）	• 以百分比显示至程序存储器（快闪 ROM）的写入（传送）状况（0 ～ 100%）。有写入指示时设置 “0”。	S（写入时）		
SD682 SD683	程序存储器写入次数指标	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存储器（快闪 ROM）写入操作次数的指标值。 指标值超过了 10 万次时将变为 “FLASH ROM ERROR”（出错代码：1610）。（指标值超过了 10 万次时仍然进行计数。） 注）写入次数不等于指标值。（通过事先由系统实施快闪 ROM 的写入寿命延长，约 2 次的写入操作后增加 1。）			
SD686	标准 ROM 写入（传送）状况	写入（传送）状况显示（百分比）	• 以百分比显示至标准 ROM（快闪 ROM）的写入（传送）状况（0 ～ 100%）。有写入指示时设置 “0”。			
SD687 SD688	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（快闪 ROM）写入操作次数的指标值。 指标值超过了 10 万次时将变为 “FLASH ROM ERROR”（出错代码：1610）。（指标值超过了 10 万次时仍然进行计数。） 注）写入次数不等于指标值。（通过事先由系统实施快闪 ROM 的写入寿命延长，上次计数到之后开始的合计写入容量达到了约 1M 字节时增加 1。）			

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU D9□□□	对应 CPU
SD689	备份原因	备份出错原因	存储备份时发生的出错原因。 0H ：无出错 100H：未安装存储卡 200H：备份对象数据尺寸超过 300H：存储卡写入禁止设置 400H：存储卡写入异常 500H：备份对象数据读取异常（程序存储器） 503H：备份对象数据读取异常（标准 RAM） 504H：备份对象数据读取异常（标准 ROM） 510H：备份对象数据读取异常（系统数据）	S（发生备份出错）	新增	QnU*1
SD690	备份状态	备份状态	存储当前的备份状态。 0：　备份开始前 1：　备份开始准备 2：　备份开始准备结束 3：　备份执行中 4：　备份结束 FF：　备份出错	S（状态变化）		
SD691	备份执行状况	备份执行状况显示（百分比）	・以百分比显示至存储卡的备份执行状况（0 ～ 100%）。 ・备份执行开始时设置“0”。	S（状态变化）		
SD692	恢复出错原因	恢复时发生的出错原因	存储恢复时发生的出错原因。 各出错原因如下所示。 800H：CPU 模块的型号不一致。 801H：备份数据文件的校验不一致。 　　或者从存储卡的备份数据的读取未正常结束。 810H：恢复目标驱动器的备份数据写入未正常结束。	S（发生出错）		
SD693	恢复状态	当前的恢复状态	存储当前的恢复执行状态。 各执行状态如下所示。 0：　恢复开始前 1：　恢复执行中 2：　恢复结束 FF：恢复出错 但是在进行自动恢复时，在恢复结束时设置“0：恢复开始前”。	S（状态变化）		
SD694	恢复执行状况	恢复执行状况显示（百分比）	・以百分比显示 CPU 模块的恢复执行状况（0 ～ 100%）。 ・恢复执行开始时设置“0”。 但是在进行自动恢复时，在恢复结束时设置“0：恢复开始前”。	S（状态变化）		
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	・指定每天的标准 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出了本 SD 中设置的次数时，将发生“OPERATION ERROR”（出错代码：4113）。 ・本 SD 的设置范围为 1 ～ 32767。设置了 0 或者超出范围的值的情况下，执行标准 ROM 写入指令时将发生“OPERATION ERROR”（出错代码：4113）。	U		QnU
SD696 SD697	存储卡可用空间	存储卡可用空间	存储存储卡的可用空间。（以 32 位的 BIN 值存储）	S（备份操作时）		QnU*1
SD698 SD699	备份数据容量	备份数据容量	存储备份的数据容量。（以 32 位的 BIN 值存储）			

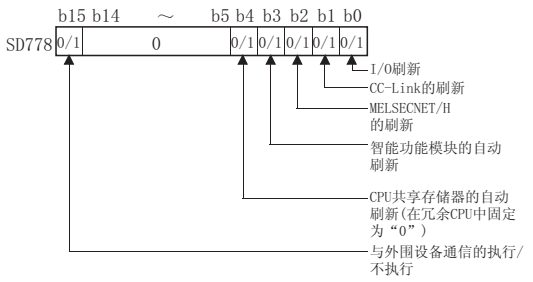
*1：以序列号的前 5 位数为“10102”以后的模块为对象。（但是，Q00UJCPU、Q00UCPU、Q01UCPU 除外。）

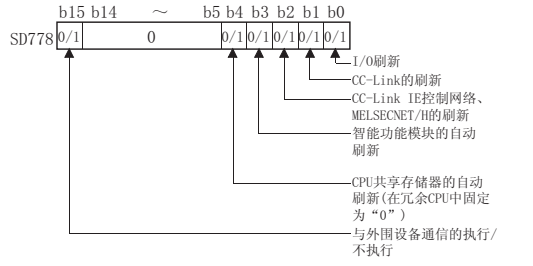
(6) 指令相关

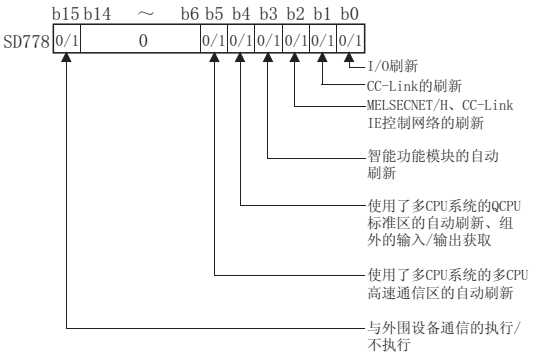
<table>
<tr><th>编号</th><th>名称</th><th>内容</th><th>详细内容</th><th>设置方
（设置时机）</th><th>对应
ACPU
D9□□□</th><th>对应 CPU</th></tr>
<tr><td>SD705
SD706</td><td>屏蔽模式</td><td>屏蔽模式</td><td>・块运算时，通过将 SM705 置为 ON，可以通过 SD705（双字时为 SD705、SD706）中存储的屏蔽模式将块中的所有数据均作为屏蔽值进行运算。</td><td>U</td><td>新增</td><td>Q00J/Q00/Q01
Qn(H)
QnPH
QnPRH</td></tr>
<tr><td>SD715
SD716
SD717</td><td>IMASK 指令屏蔽模式</td><td>屏蔽模式</td><td>・将通过 IMASK 指令进行了屏蔽的屏蔽模式按下述方式存储。
<table><tr><th></th><th>b15</th><th></th><th>b1</th><th>b0</th></tr><tr><td>SD715</td><td>115</td><td>～</td><td>11</td><td>10</td></tr><tr><td>SD716</td><td>131</td><td>～</td><td>117</td><td>116</td></tr><tr><td>SD717</td><td>147</td><td>～</td><td>133</td><td>132</td></tr></table></td><td>S（执行时）</td><td>新增</td><td rowspan="2">QCPU</td></tr>
<tr><td>SD718
SD719</td><td>累加器</td><td>累加器</td><td>・作为 A 系列程序中的累加器转换使用。</td><td>S/U</td><td>新增</td></tr>
<tr><td>SD720</td><td>PLOADP 指令程序号指定</td><td>PLOADP 指令程序号指定</td><td>希望通过 PLOADP 指令指定装载程序的程序号时进行存储。
指定范围：1～124</td><td>U</td><td>新增</td><td>Qn(H)
QnPH</td></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>信息存储</td><td>信息存储</td><td>・通过 MSG 指令存储指定的信息。
<table><tr><th></th><th>b15 ～ b8</th><th>b7 ～ 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（执行时）</td><td>新增</td><td>QCPU</td></tr>
<tr><td>SD774～
SD775</td><td>PID 极限限制设置
（完全微分用）</td><td>0：有极限限制
1：无极限限制</td><td>・按以下方式指定 PID 各环路的极限限制。
<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>～</td><td>环路2</td><td>环路1</td></tr><tr><td>SD775</td><td>环路32</td><td>～</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 极限限制设置
（完全微分用）</td><td>0：有极限限制
1：无极限限制</td><td>・按以下方式指定 PID 各环路的极限限制。
<table><tr><th></th><th>b15 ～ b8</th><th>b7 ～ b1</th><th>b0</th></tr><tr><td>SD774</td><td></td><td>环路8 ～ 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>
</table>

*9：以功能版本 B 以后为对象。

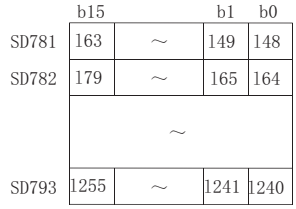
<table>
<tr><th>编号</th><th>名称</th><th>内容</th><th>详细内容</th><th>设置方
（设置时机）</th><th>对应
ACPU
D9□□□</th><th>对应 CPU</th></tr>
<tr><td rowspan="3">SD778</td><td rowspan="3">执行 COM/CCOM
指令时刷新处理
选择</td><td rowspan="3">b0 ～ b14
（默认：0）
0：不执行刷新
1：执行刷新

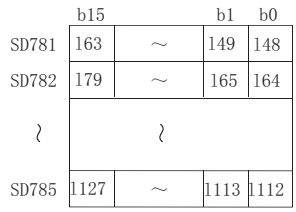
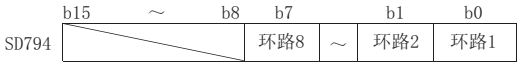
b15
0：执行与 CPU 模
块的通信
1：不执行与 CPU
模块的通信</td><td>• 执行 COM 指令时，选择是否执行各刷新处理。
• SD778 的指定只有在 SM775 处于 ON 的状态时有效。
 

• 通过 COM 指令进行的多 CPU 之间的刷新是在下述情况下执行。
来自于其它机号 CPU 的接收动作：SD778 的 b4(CPU 共享存储器的自动刷新）为 1 时
从自站 CPU 的发送动作：SD778 的 b15（与外围设备通信的执行 / 不执行）为 0 时</td><td rowspan="3">U</td><td rowspan="3">新增</td><td>Q00J/Q00
/Q01*9
Qn(H)*11</td></tr>
<tr><td>• 执行 COM 指令时，选择是否执行各刷新处理。
• SD778 的指定只有在 SM775 处于 ON 的状态时有效。
 

• 通过 COM 指令进行的多 CPU 之间的刷新是在下述情况下执行。
来自于其它机号 CPU 的接收动作：SD778 的 b4(CPU 共享存储器的自动刷新）为 1 时
从自站 CPU 的发送动作：SD778 的 b15（与外围设备通信的执行 / 不执行）为 0 时
• SD778 的 b2(CC-Link IE 控制网络、MELSECNET/H 的刷新）为 1 时，CC-Link IE 控制网络、MELSECNET/H 均进行刷新。因此，刷新点数较多时，COM 指令的处理时间将延长。</td><td>Qn(H)*13
QnPH*12
QnPRH</td></tr>
<tr><td>• 执行 COM、CCOM 指令时，选择是否执行各刷新处理。
• SD778 的指定只有在 SM775 处于 ON 的状态时有效。
 
</td><td>QnU</td></tr>
</table>

*9: 以功能版本 B 以后为对象。
*11: 以序列号的前 5 位数为 “04012” 以后的模块为对象。
*12: 以序列号的前 5 位数为 “07032” 以后的模块为对象。
*13: 以序列号的前 5 位数为 “09012” 以后的模块为对象。

<table>
<tr><th>编号</th><th>名称</th><th>内容</th><th>详细内容</th><th>设置方
（设置时机）</th><th>对应
ACPU
D9□□□</th><th>对应 CPU</th></tr>
<tr><td>SD781～
SD793</td><td>IMASK 指令屏蔽模式</td><td>屏蔽模式</td><td>・将通过 IMASK 指令进行了屏蔽的屏蔽模式按下述方式存储。
 

（在 Q00UJCPU、Q00UCPU、Q01UCPU 中，不能使用 SD786～793。）</td><td>S（执行时）</td><td>新增</td><td>Qn(H)
QnPH
QnPRH
QnU</td></tr>
<tr><td>SD781
～SD785</td><td>IMASK 指令屏蔽模式</td><td>屏蔽模式</td><td>・将通过 IMASK 指令进行了屏蔽的屏蔽模式按下述方式存储。
 
</td><td>S（执行时）</td><td>新增</td><td>Q00J/Q00/Q01</td></tr>
<tr><td>SD794～
SD795</td><td>PID 极限限制设置（不完全微分用）</td><td>0：有极限限制
1：无极限限制</td><td>・按以下方式指定 PID 各环路的极限限制。
 
</td><td>U</td><td>新增</td><td>Qn(H)*13
QnPRH
QnU</td></tr>
<tr><td>SD794</td><td>PID 极限限制设置（不完全微分用）</td><td>0：有极限限制
1：无极限限制</td><td>・按以下方式指定 PID 各环路的极限限制。
 
</td><td>U</td><td>新增</td><td>Q00J/Q00
/Q01*9</td></tr>
<tr><td>SD796</td><td>多 CPU 高速通信专用指令最多使用块数设置
(1 号机用）</td><td rowspan="4">专用指令最多使用块数范围 1～7
（默认：2）
※设置了超出范围的值的情况下，按 7 执行动作。</td><td>・指定多 CPU 高速通信专用指令（对象机号 =1 号机）的最多使用块数。对 1 号机执行了多 CPU 高速通信专用指令时，专用指令传送区的空闲块数低于本寄存器的设置值的情况下，将 SM796 置为 ON，作为多 CPU 高速通信专用指令的连续执行互锁信号使用。</td><td>U(RUN 后
1 个扫描时）</td><td>新增</td><td rowspan="4">QnU*14*15</td></tr>
<tr><td>SD797</td><td>多 CPU 高速通信专用指令最多使用块数设置
(2 号机用）</td><td>・指定多 CPU 高速通信专用指令（对象机号 =2 号机）的最多使用块数。对 2 号机执行了多 CPU 高速通信专用指令时，专用指令传送区的空闲块数低于本寄存器的设置值的情况下，将 SM797 置为 ON，作为多 CPU 高速通信专用指令的连续执行互锁信号使用。</td><td>U(RUN 后
1 个扫描时）</td><td>新增</td></tr>
<tr><td>SD798</td><td>多 CPU 高速通信专用指令最多使用块数设置
(3 号机用）</td><td>・指定多 CPU 高速通信专用指令（对象机号 =3 号机）的最多使用块数。对 3 号机执行了多 CPU 高速通信专用指令时，专用指令传送区的空闲块数低于本寄存器的设置值的情况下，将 SM798 置为 ON，作为多 CPU 高速通信专用指令的连续执行互锁信号使用。</td><td>U(RUN 后
1 个扫描时）</td><td>新增</td></tr>
<tr><td>SD799</td><td>多 CPU 高速通信专用指令最多使用块数设置
(4 号机用）</td><td>・指定多 CPU 高速通信专用指令（对象机号 =4 号机）的最多使用块数。对 4 号机执行了多 CPU 高速通信专用指令时，专用指令传送区的空闲块数低于本寄存器的设置值的情况下，将 SM799 置为 ON，作为多 CPU 高速通信专用指令的连续执行互锁信号使用。</td><td>U(RUN 后
1 个扫描时）</td><td>新增</td></tr>
</table>

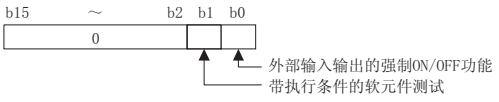
*9: 以功能版本 B 以后为对象。

*13: 以序列号的前 5 位数为“05032”以后的模块为对象。

*14: 以除 Q00UJCPU、Q00UCPU、Q01UCPU、Q02UCPU 以外的通用型 QCPU 为对象。

*15: 对于序列号的前 5 位数为“10012”以前的 Q03UDCPU、Q04UDHCPU、Q06UDHCPU，设置范围为 1～9（默认值：2）。此外，在设置值超出了 1～9 的范围的情况下，将按 9 执行动作。

(7) 调试

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU D9□□□	对应 CPU
SD840	调试功能使用状况	调试功能使用状况	存储下述调试功能的使用状况。 范围 0：外部输入输出的强制 ON/OFF 功能 1：带执行条件的软元件测试 2 ～ 15：空闲（固定为 0）  (0：不使用 1：正在使用）	S（状态变化）	新增	QnU*1

*1：以序列号的前 5 位数为 “10042”以后的模块为对象。

(8) 冗余 CPU 信息（自站 CPU 信息 *1）

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU D9□□□	对应 CPU
SD952	从控制系统至待机系统的存储器拷贝执行履历	以前的控制系统至待机系统的存储器拷贝状态	存储以前执行的从控制系统至待机系统的存储器拷贝的结束状态。 1) 在从控制系统至待机系统的存储器拷贝正常结束・异常结束的时点存储与 SD1596 中存储的值相同的值。 2) 由于执行了停电保持，因此以前执行的从控制系统至待机系统的存储器拷贝状态将被保持。 3) 通过锁存清除操作进行清 0。	S（状态变化）	新增	QnPRH

*1：存储自站系统 CPU 模块的信息。

(9) 远程口令累计次数

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU D9□□□	对应 CPU
SD979	MELSOFT 直接连接	解锁处理异常结束的累计次数	存储解锁处理异常的次数。 范围：0 ～ 0FFFEH（超出了范围时将变为 0FFFFH。）	S（状态变化）	新增	QnU*1
SD980 ～ SD995	连接 1 ～ 16					
SD997	MELSOFT 连接 UDP 端口					
SD998	MELSOFT 连接 TCP 端口					
SD999	FTP 通信端口					

*1：以以太网端口内置 QCPU 为对象。

(10)A → Q 转换对应

通过 A → Q 转换进行了转换时，ACPU 的特殊寄存器 D9000 ～ D9255 所对应的特殊寄存器为 SD1000 ～ SD1255。

（但是，基本型 QCPU 及冗余 CPU 不支持 A → Q 转换。）

该特殊寄存器全部由系统方进行设置。用户不能通过程序进行设置。

如果用户希望对其数据进行设置，应将程序修改为 Q 系列用的特殊寄存器。

但是，对于 SD1200 ～ SD1255，只有在转换前的 D9200 ～ D9255 中可以由用户方进行数据设置的特殊寄存器，才可以在转换后的SD1200～SD1255中也能由用户方进行数据设置。

关于 ACPU 的特殊寄存器的详细内容，请参阅各 CPU 模块的用户手册以及 MELSECNET、MELSECNET/B 数据链接系统参考手册。

要 点

在高性能型 QCPU、过程 CPU 以及通用型 QCPU 中使用转换后的特殊寄存器时，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对 “A 系列 CPU 兼容设置”进行勾选。

但是，由于使用转换后的特殊寄存器时需要耗费处理时间，因此在不使用的情况下应取消 “A 系列 CPU 兼容设置”的勾选。

备 注

以下为对修改用的特殊寄存器栏的补充说明。

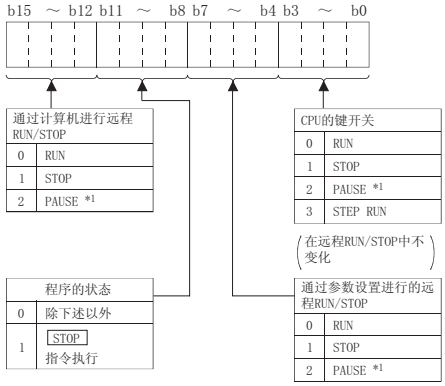
1. 对于修改用特殊寄存器一栏中记述的软元件编号，应修改为所记述的 QCPU 用的特殊寄存器。
2. 记述了□的软元件编号可以使用转换后的特殊寄存器。
3. 记述了☒的软元件编号在 QCPU 中不能使用。

<table>
<tr><th>ACPU 的特殊寄存器</th><th>转换后的特殊寄存器</th><th>修改用的特殊寄存器</th><th>名称</th><th>内容</th><th>详细内容</th><th>对应 CPU</th></tr>
<tr><td>D9000</td><td>SD1000</td><td>－</td><td>保险丝熔断</td><td>保险丝熔断模块号</td><td>・检测出保险丝熔断的模块时，以 16 进制数存储检测出的模块的最小号的 I/O 地址号的起始。（例：Y50 ～ 6F 的输出模块保险丝熔断…以 16 进制数存储为“50”）
通过外围设备进行监视的情况下，进行 16 进制数显示的监视操作。
（SD1100 ～ SD1107 的内容全部变为 0 时，将被清除。）
・对远程 I/O 站的输出模块也进行保险丝熔断状态的检查。</td><td>Qn(H)
QnPH
QnU*1</td></tr>
<tr><td>D9001</td><td>SD1001</td><td>－</td><td>保险丝熔断</td><td>保险丝熔断模块号</td><td>・有保险丝熔断时，存储设置开关号或者基板的插槽号所对应的号。
<table>
<tr><th colspan="2">AOJ2用输入输出模块时</th><th colspan="2">扩展基板时</th></tr>
<tr><th>设置开关</th><th>存储数据</th><th>基板的插槽号</th><th>存储数据</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>
・远程 I/O 站的情况下，存储（模块的 I/O 地址号 /10H)+1 的值。</td><td>Qn(H)
QnPH</td></tr>
<tr><td>D9002</td><td>SD1002</td><td>－</td><td>输入输出模块校验出错</td><td>输入输出模块校验出错号</td><td>・接通电源时检测出与所登录的信息不相同的输入输出模块时，以 16 进制数存储检测出的模块的最小号的 I/O 地址号的起始。（存储方法与 SD1000 相同）
通过外围设备进行监视的情况下，进行 16 进制数显示的监视操作。
（SD1116 ～ SD1123 的内容全部变为 0 时，将被清除。）
・对远程 I/O 站的模块也进行输入输出模块校验的检查。</td><td rowspan="4">Qn(H)
QnPH
QnU*1</td></tr>
<tr><td>D9005</td><td>SD1005</td><td>－</td><td>AC DOWN 计数器</td><td>AC DOWN 次数</td><td>・使用 AC 电源模块时发生了 20ms 以内的瞬间掉电的情况下本寄存器将变为 ON。
通过将电源由 OFF 变为 ON 可进行复位。
・使用 DC 电源模块时发生了 10ms 以内的瞬间掉电的情况下本寄存器将变为 ON。
通过将电源由 OFF 变为 ON 可进行复位。</td></tr>
<tr><td>D9008</td><td>SD1008</td><td>SD0</td><td>自诊断出错</td><td>自诊断出错代码</td><td>・以 BIN 码存储发生了自诊断出错时的出错代码。</td></tr>
<tr><td>D9009</td><td>SD1009</td><td>SD62</td><td>报警器的检测</td><td>发生了外部故障的 F 编号</td><td>・通过 OUT F、SET F 指令使 F0 ～ 2047 中的某一个变为 ON 时，以 BIN 码存储变为 ON 的 F 编号中最先检测出的 F 编号。
・SD1009 的清除是通过 RST F、LEDR 指令进行。
检测出其它的 F 编号时，如果将 SD1009 清除，则下一个编号将被存储到 SD1009 中。</td></tr>
<tr><td>D9010</td><td>SD1010</td><td>×</td><td>出错步</td><td>发生了运算出错的步号</td><td>・在应用指令的执行过程中发生了运算出错时，以 BIN 码存储发生了该出错的步号。此后每当发生了运算出错时 SD1010 的内容将被更新。</td><td rowspan="3">Qn(H)
QnPH</td></tr>
<tr><td>D9011</td><td>SD1011</td><td>×</td><td>出错步</td><td>发生了运算出错的步号</td><td>・在应用指令的执行过程中发生了运算出错时，以 BIN 码存储发生了该出错的步号。由于至 SD1011 的存储是在 SM1011 由 OFF 变为 ON 时进行，因此如果没有通过用户程序将 SM1011 清除，则 SD1011 的内容将不能被更新。</td></tr>
<tr><td>D9014</td><td>SD1014</td><td>×</td><td>输入输出控制方式</td><td>输入输出控制方式编号</td><td>・设置的输入输出控制方式通过下述编号返回。
0：输入输出均直接
1：输入刷新、输出直接
3：输入输出均刷新</td></tr>
</table>

*1：以下述模块为对象。

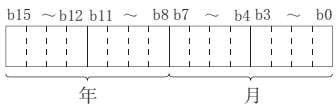
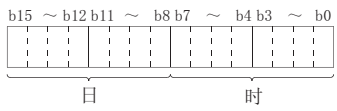
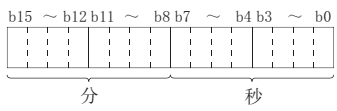
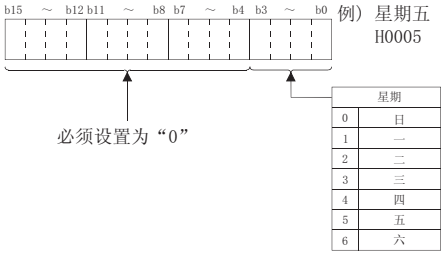
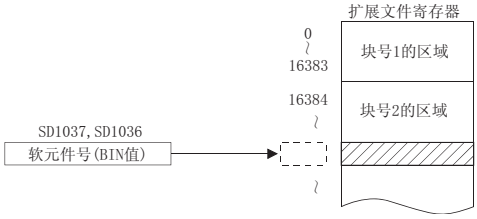
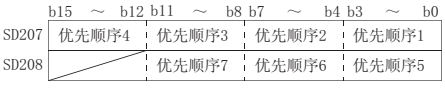
・序列号的前 5 位数为“10102”以后的通用型 QCPU

・Q00UJCPU、Q00UCPU、Q01UCPU

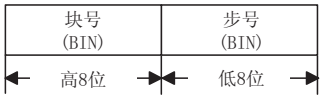
ACPU 的特殊寄存器	转换后的特殊寄存器	修改用的特殊寄存器	名称	内容	详细内容	对应 CPU
D9015	SD1015	SD203	CPU 动作状态	CPU 动作状态	• SD1015 中按下图方式存储 CPU 的动作状态。  *1 ：在 CPU 模块的 RUN 过程中 SM1040 变为 OFF 时，即使执行了 PAUSE 也仍将保持为 RUN 状态不变。	Qn(H) QnPH QnU*1
D9016	SD1016	×	程序编号	0：主程序 (ROM) 1：主程序 (RAM) 2：子程序 1(RAM) 3：子程序 2(RAM) 4：子程序 3(RAM) 5：子程序 1(ROM) 6：子程序 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)	• 显示当前正在执行哪个顺控程序，以 BIN 存储 0 ～ B 的某个值。	Qn(H) QnPH
D9017	SD1017	SD524	扫描时间	最短扫描时间 (10ms 单位)	• 每次 END 时扫描时间短于 SD1017 的内容的情况下将重新存储该值。即 SD1017 中以 BIN 码存储扫描时间的最小值。	Qn(H) QnPH QnU*1
D9018	SD1018	SD520	扫描时间	扫描时间 (10ms 单位)	• 每次 END 时以 BIN 码存储扫描时间，常时进行改写。	
D9019	SD1019	SD526	扫描时间	最长扫描时间 (10ms 单位)	• 每次 END 时扫描时间长于 SD1019 的内容的情况下将重新存储该值。即 SD1019 中以 BIN 码存储扫描时间的最大值。	
D9020	SD1020	×	恒定扫描	恒定扫描时间 (由用户以 10ms 为单位进行设置)	• 以一定间隔执行用户程序时，以 10ms 为单位设置执行间隔。 0 ：无恒定扫描功能 1 ～ 200 ：有恒定扫描功能 (以设置值×10ms 的间隔执行)	Qn(H) QnPH
D9021	SD1021	－	扫描时间	扫描时间 (1ms 单位)	• 每次 END 时以 BIN 码存储扫描时间，常时进行改写。	Qn(H) QnPH QnU*1
D9022	SD1022	SD412	1 秒计数器	1s 单位的计数数	• 可编程控制器 CPU RUN 后，每 1s+1。 • 计数按 0 → 32767 → -32768 → 0 的循环反复进行。	

*1: 以下述模块为对象。

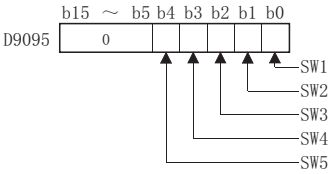
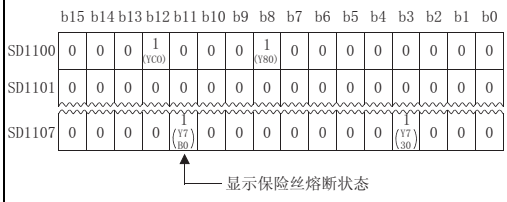
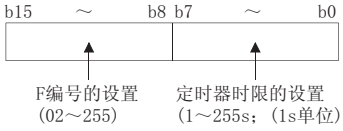
- 序列号的前 5 位数为 “10102”以后的通用型 QCPU
- Q00UJCPU、Q00UCPU、Q01UCPU

ACPU 的特殊寄存器	转换后的特殊寄存器	修改用的特殊寄存器	名称	内容	详细内容	对应 CPU
D9025	SD1025	-	时钟数据	时钟数据（公历，月）	• 如下图所示以 BCD 码存储年（公历，低 2 位）、月。  例）87年，7月 H8707	Qn(H) QnPH QnU*1
D9026	SD1026	-	时钟数据	时钟数据（日，时）	• 如下图所示以 BCD 码存储日、时。  例）31日，10时 H3110	
D9027	SD1027	-	时钟数据	时钟数据（分，秒）	• 如下图所示以 BCD 码存储分、秒。  例）35分，48秒 H3548	
D9028	SD1028	-	时钟数据	时钟数据（星期）	• 如下图所示以 BCD 码存储星期。  例）星期五 H0005	
D9035	SD1035	SD648	扩展文件寄存器	使用块号	• 以 BIN 码存储当前正在使用的扩展文件寄存器的块号。	
D9036	SD1036	×	用于扩展文件寄存器软元件编号指定	对扩展文件寄存器的各软元件进行直接访问时的软元件编号	• 将进行直接读取、写入的扩展文件寄存器的软元件编号以 BIN 值指定到 SD1036、SD1037 的 2 字中。 可以在无需理会块号的情况下，从块号 1 的 R0 开始对软元件编号进行连续编号指定。 	Qn(H) QnPH
D9037	SD1037	×				
D9038	SD1038	SD207	LED 显示优先顺序	优先顺序 1 ～ 4	• 通过出错项目号设置发生异常时，“ERROR”LED 亮灯（闪烁）的优先顺序。 • 优先顺序的设置区的情况如下所示。  • 有关详细内容请参阅所使用的 CPU 模块的用户手册或者 ACPU 编程手册（基础篇）SH-3435（版本 I 以后）。	
D9039	SD1039	SD208		优先顺序 5 ～ 7		
D9044	SD1044	×	采样跟踪用	采样跟踪时的步或者时间	• 通过外围设备对 SM803 进行 ON/OFF，使采样跟踪的 STRA、STRAR 指令动作时，将 SD1044 中存储的值作为采样跟踪的条件使用。 扫描时 --------0 定时时 -------- 时间（10ms 单位） 在 SD1044 中以 BIN 码进行值的存储。	

*1：以下述模块为对象。
- 序列号的前 5 位数为“10102”以后的通用型 QCPU
- Q00UJCPU、Q00UCPU、Q01UCPU

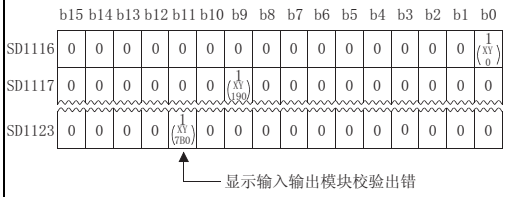
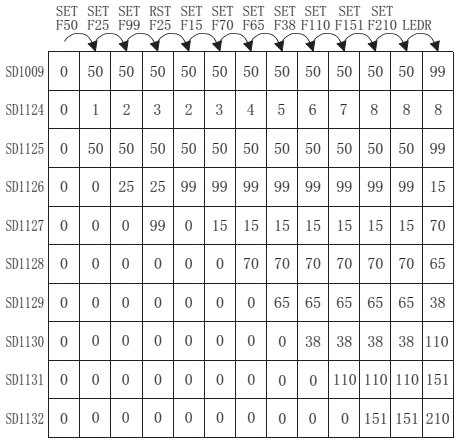
ACPU 的特殊寄存器	转换后的特殊寄存器	修改用的特殊寄存器	名称	内容	详细内容	对应 CPU
D9049	SD1049	×	SFC 程序执行用工作区	作为 SFC 程序执行用工作区使用的扩展文件寄存器块号	• 以 BIN 值存储作为 SFC 程序执行用工作区使用的扩展文件寄存器块号。 • 使用不能成为扩展文件寄存器块号 1 的 16k 字节以下的空闲区域时，或者 SM320 为 OFF 时，存储 “0”。	
D9050	SD1050	×	SFC 程序出错编号	SFC 程序中发生的出错编号	• 以 BIN 值存储 SFC 程序中发生的出错编号。 0: 无出错 80: SFC 程序用参数出错 81: SFC 代码出错 82: 同时执行步数超限 83: 块启动出错 84: SFC 程序运算出错	
D9051	SD1051	×	出错块	发生了出错的块号	• 以 BIN 值存储 SFC 程序中发生了出错的块号。 但是，出错 83 的情况下，存储启动源的块号。	
D9052	SD1052	×	出错步	发生了出错的步号	• 以 BIN 值存储 SFC 程序中发生了出错代码 84 的步号。 • 发生了出错代码 80、81、82 时，存储 “0”。 • 发生了出错代码 83 时，存储块启动步号。	
D9053	SD1053	×	出错转移	发生了出错的转移条件号	• 以 BIN 值存储 SFC 程序中发生了出错代码 84 的转移条件号。 发生了出错代码 80、81、82 、83时，存储 “0”。	
D9054	SD1054	×	出错顺控程序步	发生了出错的顺控程序步号	• 在 SFC 程序中发生了出错代码 84 的转移条件、步中，以 BIN 值存储在转移条件、动作输出的第几个顺控程序步中发生了出错。	Qn(H) QnPH
D9055	SD1055	SD812	状态锁存执行步号	状态锁存执行步号	• 存储执行了状态锁存时的步号。 • 在主顺控程序中执行状态锁存时，以 BIN 值存储步号。 • 在 SFC 程序中执行状态锁存时，存储块号及步号。 	
D9072	SD1072	×	可编程控制器通信校验	串行通信模块的数据校验	• 通过串行通信模块的单体自回送测试，串行通信模块自动进行数据的写入 / 读取，进行通信校验。	
D9085	SD1085	×	定时校验时间	1s ～ 65535s	• 设置 MELSECNET/10 用数据链接指令 (ZNRD、ZNWR) 的定时校验时间。 • 设置范围 : 1s ～ 65535s(1 ～ 65535) • 设置单位 : s • 默认值 : 10s(0 的情况下为 10s)	
D9090	SD1090	×	微机子程序 INPUT 数据区起始软元件号	根据各微机元件包	• 有关详细内容请参阅微机元件包的手册。	
D9091	SD1091	×	指令出错	指令出错详细编号	• 以代码存储指令出错发生原因的详细内容。	Qn(H) QnPH QnU*1
D9094	SD1094	SD251	更换输入输出模块的起始 I/O 地址号	更换输入输出模块的起始 I/O 地址号	• 以 BIN 值存储在线（通电状态下）拆装的输入输出模块的起始 I/O 地址号的高 2 位数。 例）输入模块 X2F0 → H2F	Qn(H) QnPH

*1: 以下述模块为对象。
- 序列号的前 5 位数为 “10102” 以后的通用型 QCPU
- Q00UJCPU、Q00UCPU、Q01UCPU

ACPU 的特殊寄存器	转换后的特殊寄存器	修改用的特殊寄存器	名称	内容	详细内容	对应 CPU
D9095	SD1095	SD200	插杆开关信息	插杆开关信息	• 按以下格式存储 CPU 模块的插杆开关信息。 0：OFF 1：ON 	Qn(H) QnPH
D9100	SD1100	–	保险丝熔断模块	保险丝熔断模块的 16 点单位的位模式	• 以位模式输入变为保险丝熔断状态的输出模块编号（16 点单位）。 （在参数中进行了设置时为所设置的编号）  • 对远程 I/O 站的输出模块也进行保险丝熔断状态的检查。 （由于即使恢复正常后也不能被清除，因此需要通过程序进行清除。）	Qn(H) QnPH QnU*1
D9101	SD1101					
D9102	SD1102					
D9103	SD1103					
D9104	SD1104					
D9105	SD1105					
D9106	SD1106					
D9107	SD1107					
D9108	SD1108	–	步移位监视定时器设置	定时器设置值及超时时的 F 编号	• 设置步移位监视定时器的设置值以及由于监视定时器的超时而变为 ON 的报警器编号（F 编号）。  • 通过将 SM1108 ～ SM1114 置为 ON 启动定时器后，在定时器时限内相应步的下一个移位条件不成立时，设置的报警器（F）将变为 ON。	Qn(H) QnPH
D9109	SD1109					
D9110	SD1110					
D9111	SD1111					
D9112	SD1112					
D9113	SD1113					
D9114	SD1114					

*1：以下述模块为对象。

- 序列号的前 5 位数为“10102”以后的通用型 QCPU
- Q00UJCPU、Q00UCPU、Q01UCPU

ACPU 的特殊寄存器	转换后的特殊寄存器	修改用的特殊寄存器	名称	内容	详细内容	对应 CPU
D9116	SD1116	－	输入输出模块校验出错	校验出错模块的 16 点单位的位模式	• 电源 ON 时检测出与所登录的输入输出模块信息不相同的输入输出模块时，输入该输入输出模块的编号（16 点单位）。 （在参数中进行了设置时为所设置的输入输出模块编号）  • 对远程 I/O 站的模块也进行输入输出模块校验。 （由于即使恢复正常后也不能被清除，因此需要通过程序进行清除。）	Qn(H) QnPH QnU*1
D9117	SD1117					
D9118	SD1118					
D9119	SD1119					
D9120	SD1120					
D9121	SD1121					
D9122	SD1122					
D9123	SD1123					
D9124	SD1124	SD63	检测出的报警器个数	检测出的报警器个数	• 通过 SET F 指令使 F0 ～ 2047 中的某一个变为 ON 时，SD63 的值将被 +1。通过 RST F 指令或者 LEDR 指令的执行，SD63 的值将被 -1。 通过 SET F 指令变为 ON 的个数最多可存储 16 个。	
D9125	SD1125	SD64	检测出的报警器编号	检测出的报警器编号	• 通过 SET F 指令使 F0 ～ 2047 中的某一个变为 ON 时，变为 ON 的报警器的编号将依次被登录到 SD1125 ～ SD1132 中。 通过 RST F 指令变为 OFF 的 F 编号将从 SD1125 ～ SD1132 中被删除，存储在被删除的 F 编号后面的 F 编号将依次向前填充对齐。 通过 LEDR 指令的执行将 SD1125 ～ SD1132 的内容向前移动一位。 检测出的报警器个数达到 8 个时，即使检测出第 9 个也不能被存储到 SD1125 ～ SD1132 中。  	
D9126	SD1126	SD65				
D9127	SD1127	SD66				
D9128	SD1128	SD67				
D9129	SD1129	SD68				
D9130	SD1130	SD69				
D9131	SD1131	SD70				
D9132	SD1132	SD71				

*1: 以下述模块为对象。

- 序列号的前 5 位数为 “10102”以后的通用型 QCPU
- Q00UJCPU、Q00UCPU、Q01UCPU

(11)以太网端口内置 QCPU 对应

编号	名称		内容	详细内容	设置方（设置时机）	对应 ACPU D9□□□	对应 CPU
SD1270	时间设置功能	动作结果	存储动作结果	存储时间设置功能的动作结果。 0：未执行 1：成功 0FFFFH：失败	S（状态变化）	新增	QnU*1
SD1271		执行时间	存储执行的时间	以 2 位数的 BCD 码存储执行了时间设置功能的年（公历，低 2 位）、月。 b15 ～ b12 b11 ～ b8 b7 ～ b4 b3 ～ b0 例）1993年，7月 9307H 年 月			
SD1272				以 2 位数的 BCD 码存储执行了时间设置功能的日、时。 b15 ～ b12 b11 ～ b8 b7 ～ b4 b3 ～ b0 例）31日，10时 3110H 日 时			
SD1273				以 2 位数的 BCD 码存储执行了时间设置功能的分、秒。 b15 ～ b12 b11 ～ b8 b7 ～ b4 b3 ～ b0 例）35分，48秒 3548H 分 秒			
SD1274				以 2 位数的 BCD 码存储执行了时间设置功能的（公历，高 2 位）及星期。 b15 ～ b12 b11 ～ b8 b7 ～ b4 b3 ～ b0 例）1993年，星期五 1905H 公历高位(0～99) 星期：0 日，1 一，2 二，3 三，4 四，5 五，6 六			
SD1275		响应所需时间	存储时间的获取所需要的时间	存储从发送至 SNTP 服务器之后至 CPU 时间设置为止所需要的时间。 范围：A0 ～ 0FFFEH（单位：ms） 超出了上述范围的情况下将变为 0FFFFH。			
SD1276	连接强制无效化		指定连接的强制无效化	希望通过用户程序使连接强制无效时进行此指定。 被指定为无效的连接其通信将停止而变为无响应状态。（将使用远程口令时解锁处理异常过多的连接视为正在遭受攻击，希望暂时将其置为访问禁止状态时使用此功能。） SD1276 b15 b14 ～ b1 b0 连接1 连接2 连接15 连接16 SD1277 b15 b14 b13 b12 ～ b4 b3 b2 b1 b0 MELSOFT通信端口(UDP/IP) MELSOFT通信端口(TCP/IP) FTP通信端口 MELSOFT的直接连接 0：有效（默认） 1：无效	U		
SD1277							

*1 ：以以太网端口内置的 QCPU 为对象。

(12)保险丝熔断模块

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU D9□□□	对应 CPU
SD1300	保险丝熔断模块	保险丝熔断模块的16点单位的位模式 0: 无保险丝熔断 1: 有保险丝熔断	• 以位模式输入变为保险丝熔断状态的输出模块编号（16点单位）。（在参数中进行了设置时为所设置的编号） • 对远程站的输出模块也进行保险丝熔断状态的检测。 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 显示保险丝熔断状态 • 即使恢复正常后也不能被清除。可通过出错解除进行清除。	S（发生出错）	D9100	Qn(H) QnPH QnPRH QnU
SD1301					D9101	
SD1302					D9102	
SD1303					D9103	
SD1304					D9104	
SD1305					D9105	
SD1306					D9106	
SD1307					D9107	
SD1308					新增	
SD1309～SD1330					新增	
SD1331					新增	

(13)输入输出模块校验

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU D9□□□	对应 CPU
SD1400	输入输出模块校验出错	校验出错模块的16点单位的位模式 0: 无输入输出校验出错 1: 有输入输出校验出错	• 电源ON时检测出与所登录的输入输出模块信息不相同的输入输出模块时，以位模式输入该输入输出模块的编号。（在参数中进行了设置时为所设置的输入输出模块编号） • 对远程站的输入输出模块信息也进行检测。 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/Y0) SD1401 0 0 0 0 0 0 1(X/Y90) 0 0 0 0 0 0 0 0 0 SD1431 0 1(X/Y1F0) 0 0 0 0 0 0 0 0 0 0 0 0 0 0 显示输入输出模块校验出错 • 即使恢复正常后也不能被清除。可通过出错解除进行清除。	S（发生出错）	D9116	Qn(H) QnPH QnPRH QnU
SD1401					D9117	
SD1402					D9118	
SD1403					D9119	
SD1404					D9120	
SD1405					D9121	
SD1406					D9122	
SD1407					D9123	
SD1408					新增	
SD1409～SD1430					新增	
SD1431					新增	

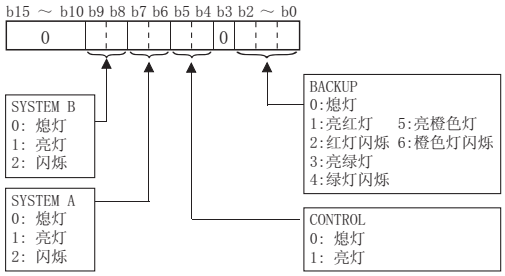
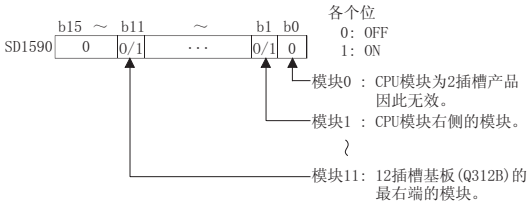
(14)过程控制指令

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU D9□□□	对应 CPU
SD1500 SD1501	执行周期	执行周期时间	• 以浮点数据设置过程控制指令用的执行周期（单位：秒）。 浮点数据= SD1501 | SD1500	U	新增	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	• 显示过程控制指令中发生的出错的详细内容。	S（发生出错）	新增	
SD1503	过程控制指令发生出错位置	过程控制指令发生出错位置	• 显示过程控制器指令中发生的出错处理块。	S（发生出错）	新增	
SD1506 SD1507	虚拟软元件	虚拟软元件	• 用于在过程控制指令中指定虚拟软元件。	U	新增	QnPH QnPRH
SD1508	过程控制指令功能选择	b0 S.PIDP 控制的无冲击切换功能 0：有效 /1：无效 （默认：0）	• 选择在过程控制指令中各功能是否有效。 b15 b14 ～ b2 b1 b0 SD1508 | 0 | 0 | 0 | I/O | S.PIDP控制的无冲击切换功能有效/无效	U	新增	

(15)冗余对应（自站系统 CPU 信息 *1）

SD1510 ～ SD1599 仅在冗余系统时有效。

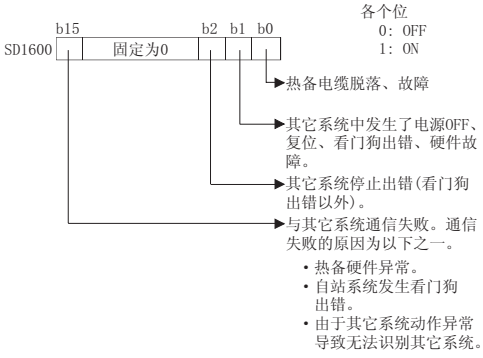
单系统时全部为 0。

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU D9□□□	对应 CPU
SD1585	冗余对应 LED 状态	以下 4 个 LED 状态 ・BACKUP ・CONTROL ・SYSTEM A ・SYSTEM B	按以下格式存储 BACKUP、CONTROL、SYSTEM A、SYSTEM B 的 LED 状态。 	S（状态变化）	新增	QnPRH
SD1588	系统切换原因	自站系统中发生的系统切换原因	存储自站系统中发生的系统切换原因。 即使由于系统切换禁止原因导致未能进行系统切换时，系统切换原因也将被存储到本寄存器中。 电源 OFF → ON/ 复位→复位解除时初始化为 0。 0：初始值（系统切换一次也未发生过）。 1：硬件故障、看门狗出错。 2：停止出错（看门狗出错除外）。 3：通过网络模块发出的系统切换请求。 16：控制系统切换指令。 17：来自于 GX Developer 的系统切换请求。	S（发生原因时）	○	
SD1589	系统切换禁止原因	系统切换禁止原因编号	・由于发生了系统切换原因导致系统切换时，如果系统切换失败，则以下述值存储系统切换失败的原因。 0：正常切换结束（默认）。 1：热备电缆异常（电缆脱落、电缆异常、内部电路异常）。 2：待机系统中发生了硬件故障、电源 OFF、复位、看门狗出错。 3：控制系统中发生了硬件故障、电源 OFF、复位、看门狗出错。 4：热备通信准备中。 5：时间超时。 6：待机系统停止出错（除看门狗出错以外）。 7：两个系统的动作不相同（仅在备份模式时进行检测） 8：正在进行从控制系统至待机系统的存储器拷贝。 9：正在进行 RUN 中写入。 10：从待机系统中检测出网络模块的异常 11：正在进行系统切换。 ・自站系统电源 ON 时以 0 进行初始化。 ・系统切换正常结束时存储 0。	S（系统切换时）	○	
SD1590	通过自站系统网络模块发出系统切换请求的模块号	通过自站系统网络模块发出系统切换请求的模块号	・通过自站系统网络模块发出了系统切换请求的模块号的下述位将变为 ON。 ・由用户将相应模块的异常消除后，由系统将相应位变为 OFF。  ・关于从其它系统网络模块发出系统切换请求的模块号，请参阅 SD1690。	S（发生出错 / 状态变化）	新增	
SD1595	存储器拷贝目标 I/O 地址号	存储器拷贝目标 I/O 地址号	・在 SM1595 由 OFF 变为 ON 之前存储存储器拷贝目标 I/O 地址号（待机系统 CPU 模块：3D1H）。	U	新增	
SD1596	存储器拷贝状态	存储器拷贝状态	・存储存储器拷贝的状态。 0 ：正常结束。 4241H ：待机系统的电源 OFF。 4242H ：热备电缆脱落、异常。 4247H ：正在执行存储器拷贝。 4248H ：不支持的拷贝目标 I/O 地址号。	S（状态变化）	新增	

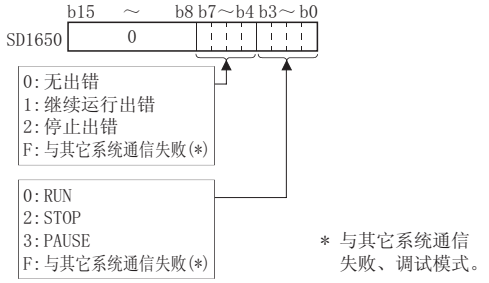
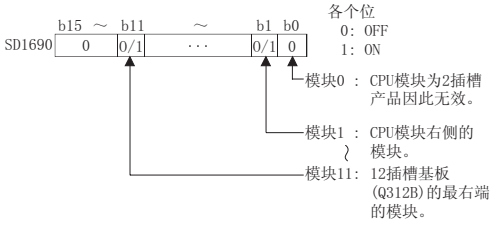
*1：存储自站系统 CPU 模块的信息。

(16)冗余对应（其它 CPU 信息 *1）

SD1600 ～ SD1650 仅在冗余系统时的备份模式下有效，在分开模式下将不被刷新。
SD1651 ～ SD1699 在备份模式及分开模式下均有效。
在单系统时，SD1600 ～ SD1699 全部为 0。

编号	名称	内容	详细内容	设置方（设置时机）	对应帧宰 SD□□ *2	对应 CPU
SD1600	系统异常信息	系统异常信息	• 在冗余系统用出错检查中检测出出错时，下述相应位将变为 ON。此后如果出错被解除则相应位变为 OFF。  • 在 b0、b1、b2、b15 中某个为 ON 时，其它则均变为 OFF。 • 调试模式时，b0、b1、b2、b15 将全部变为 OFF。	S（每次 END）	–	QnPRH
SD1601	系统切换结果	系统切换原因	存储系统切换原因。 • 系统切换时两个系统的 SD1601 中存储系统切换原因。 • 电源 OFF → ON/ 复位→复位解除时初始化为 0。 • 本寄存器中存储的值如下所示。 0：初始值（系统切换一次也未发生过。） 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：通过网络模块发出的系统切换请求。 16：控制系统切换指令。 17：来自于 GX Developer 的系统切换请求。 *：由于控制系统的电源 OFF/ 复位导致发生了系统切换时，新待机系统的 SD1601 中不存储 “1”。	S（系统切换时）		
SD1602	控制系统切换指令变量	控制系统切换指令变量	• 通过 SP.CONTSW 指令发生了系统切换时，存储指令的变量。（在系统切换时两个系统的 SD1602 中将存储 SP.CONTSW 指令的变量。） • SD1602 只有在 SD1601 中存储了 “16：控制系统切换指令” 时有效。 • SD1602 仅在通过控制系统切换指令执行了系统切换时进行更新。	S（系统切换时）		
SD1610	其它系统诊断出错	诊断出错代码	• 以 BIN 值存储其它系统中发生的出错的出错代码。 • 反映其它系统 CPU 模块的 SD0。	S（每次 END）	SD0	
SD1611	其它系统诊断出错发生时间	诊断出错发生时间	• 存储其它系统中发生的出错的发生时间。 • 数据的构成与 SD1 ～ SD3 相同。 • 反映其它系统 CPU 模块的 SD1 ～ SD3。	S（每次 END）	SD1 ～ SD3	
SD1612						
SD1613						

*1：存储其它系统 CPU 模块的诊断信息、系统信息。
*2：表示自站系统 CPU 模块中对应的特殊寄存器（SD□□）。

编号	名称	内容	详细内容	设置方（设置时机）	对应帧寄存器 SD□□ *2	对应 CPU
SD1614	其它系统出错信息分类	出错信息分类代码	• 存储其它系统中发生的出错的个别信息 • 公共信息的分类代码。 • 数据的构成与 SD4 的相同。 • 反映其它系统 CPU 模块的 SD4。	S（每次 END）	SD4	
SD1615 ～ SD1625	其它系统出错公共信息	出错公共信息	• 存储其它系统中发生的出错的公共信息。 • 数据构成与 SD5 ～ SD15 的相同。 • 反映其它系统 CPU 模块的 SD5 ～ SD15。	S（每次 END）	SD5 ～ SD15	
SD1626 ～ SD1636	其它系统出错个别信息	出错个别信息	• 存储其它系统中发生的出错的个别信息。 • 数据构成与 SD16 ～ SD26 的相同。 • 反映其它系统 CPU 模块的 SD16 ～ SD26。	S（每次 END）	SD16 ～ SD26	
SD1649	待机系统出错解除指令	解除的出错的出错代码	• 存储通过待机系统出错解除所要解除的出错的出错代码。 • 在本寄存器中存储要解除的出错的出错代码后，通过对进行 SM1649 OFF → ON 操作，对待机系统的出错进行解除。 • 对于本寄存器中存储的出错代码，低一位（个数的位）的值将被忽略。（通过在本寄存器中存储 4100 并进行出错解除，4100 ～ 4109 的出错均可被解除。）	S（每次 END）		
SD1650	其它系统动作信息	其它系统动作信息	按以下格式存储其它系统 CPU 模块的动作信息。 与其它系统通信失败或者处于调试模式时将存储 00FFH。  〈补充说明〉在以下状态下将无法与其它系统通信。 • 其它系统处于电源 OFF/ 复位状态。 • 自站系统或者其它系统中发生了硬件故障。 • 自站系统或者其它系统中发生了看门狗出错。 • 未安装热备电缆，或者热备电缆断线 / 故障。	S（每次 END）		QnPRH
SD1690	通过其它系统网络模块发出了系统切换请求的模块号	通过其它系统网络模块发出了系统切换请求的模块号	• 通过其它系统网络模块发出了系统切换请求的模块号的下述位将变为 ON。 • 由用户将相应模块的异常消除后，由系统将相应位变为 OFF。  • 关于从自站系统网络模块发出系统切换请求的模块号，请参阅 SD1590。	S（每次 END）		

*2: 表示自站 CPU 模块中对应的特殊寄存器（SD□□）。

(17)冗余对应（热备信息）

SD1700 ～ SD1779 仅在冗余系统时有效。

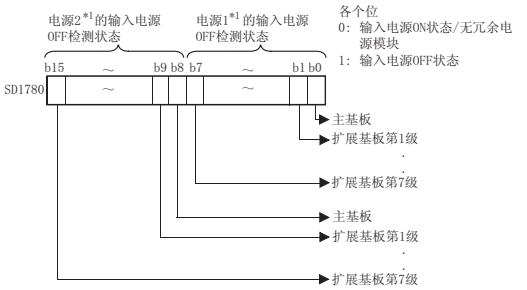
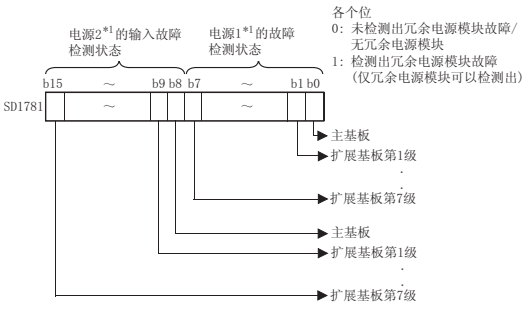
单系统时全部为 0。

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU D9□□□	对应 CPU
SD1700	检测出热备异常的次数	检测出热备异常的次数	• 检测出热备异常时进行 +1。 • 计数按 0 → 32767 → -32768 → 0 的循环反复进行。	S（发生出错）	新增	QnPRH
SD1710	待机系统 RUN 中写入开始等待时间	待机系统 RUN 中写入开始等待时间	• 在 RUN 中写入冗余追踪功能中，以秒为单位设置从控制系统 CPU 模块的 RUN 中写入结束起，至待机系统 CPU 模块的 RUN 中写入开始为止的待机系统 CPU 模块的等待时间。 • 控制系统 CPU 模块的 RUN 中写入结束后，如果在设置时间以内未发出待机系统 CPU 模块的 RUN 中写入请求，则被判断为两个系统 CPU 模块 RUN 中写入冗余追踪异常结束。在这种情况下，两个系统 CPU 模块将重新进行由于 RUN 中写入而停止的两系统一致性检查。此外，将控制系统 CPU 模块置为可受理新的 RUN 中写入冗余追踪请求的状态。 • 两个系统的电源 ON 时，将 90 秒作为默认值设置到 SD1710 中。 • 可设置范围为 90 ～ 3600 秒。如果设置为 0 ～ 89 秒，则按设置值 90 秒执行动作。如果设置值超出了 0 ～ 3600 秒的范围，则按设置值 3600 秒执行动作。 • 执行多个块 RUN 中写入冗余追踪、文件批量 RUN 中写入冗余追踪时，也将根据 SD1710 的设置值，对待机系统 CPU 模块的 RUN 中写入开始的等待时间进行检查。	S（初始化）/U		

(18)冗余电源模块信息

SD1780 ～ SD1789 仅在电源冗余系统时有效。

在单电源系统时全部为 0。

编号	名称	内容	详细内容	设置方（设置时机）	对应 ACPU D9□□□	对应 CPU
SD1780	电源 OFF 检测状态	电源 OFF 检测状态	• 按下述位模式存储输入电源变为 OFF 状态的冗余电源模块（Q64RP）。 • 在主基板不是电源冗余主基板（Q38RB）的情况下，在本寄存器中将存储 0。  • 在多 CPU 系统配置时，仅在 1 号机的 CPU 模块中存储电源 OFF 检测状态。	S（每次 END）	新增	Qn(H)*2 QnPH*2 QnPRH QnU*3
SD1781	电源故障检测状态	电源故障检测状态	• 按下述位模式存储冗余电源模块（Q64RP）的故障检测状态。（检测出冗余电源模块的故障后，故障的冗余电源模块的输入电源 OFF 时将相应位置为 0。） • 在主基板不是电源冗余主基板（Q38RB）的情况下，在本寄存器中将存储 0。  • 在多 CPU 系统配置时，仅在 1 号机的 CPU 模块中存储故障检测状态。	S（每次 END）	新增	
SD1782	电源 1*1 用瞬间掉电检测次数	检测出电源 1 的瞬间掉电次数	• 对电源 1/ 电源 2 的瞬间掉电次数进行计数。 • 对电源冗余主基板（Q38RB）上安装的电源 1/ 电源 2 的状态进行监视及计数。 对电源冗余扩展基板、冗余扩展基板上安装的电源 1/ 电源 2 的状态不进行监视。 • CPU 模块启动时，电源 1/ 电源 2 的计数器将被清 0。 • 一侧的冗余电源模块的输入电源为 OFF 时，对输入电源 OFF 的冗余电源模块对应的计数器进行清 0。 • 每检测出一次电源 1/ 电源 2 的瞬间掉电时将进行 +1。 （计数按 0 → 32767 → -32768 → 0 的循环反复进行。（在 GX Developer 的系统监视中，在 0 ～ 65535 的范围内显示。）） • 在主基板不是电源冗余主基板（Q38RB）的情况下，在本寄存器中将存储 0。 • 多 CPU 系统配置时，仅在 1 号机的 CPU 模块中存储瞬间掉电次数。	S（每次 END）	新增	
SD1783	电源 2*1 用瞬间掉电检测次数	检测出电源 2 的瞬间掉电次数		S（每次 END）	新增	

*1: “电源 1”表示冗余基板（Q38RB/Q68RB/Q65WRB）的 POWER1 插槽中安装的冗余电源模块。
“电源 2”表示冗余基板（Q38RB/Q68RB/Q65WRB）的 POWER2 插槽中安装的冗余电源模块。

*2: 以序列号的前 5 位数为 “07032”以后的模块为对象。
但是，多 CPU 系统配置时，所有的 CPU 模块均以序列号的前 5 位数为 “07032”以后的模块为对象。

*3: 以序列号的前 5 位数为 “10042”以后的模块为对象。

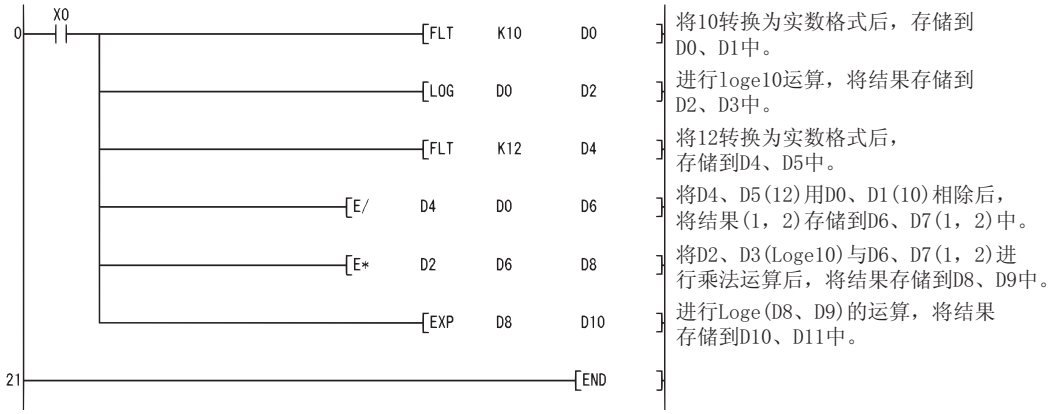
附录 5.1　执行 X^n、$\sqrt[n]{X}$ 运算的程序的思路

在 QCPU 与 QnACPU 中仍然没有执行 X^n、$\sqrt[n]{X}$ 运算的指令，但可以通过 LOG 指令与 EXP 指令的组合执行 X^n、$\sqrt[n]{X}$ 运算。

(1) 执行 X^n 运算的程序的思路

X^n 可以通过 $e^{(n\log eX)}$ 进行运算。

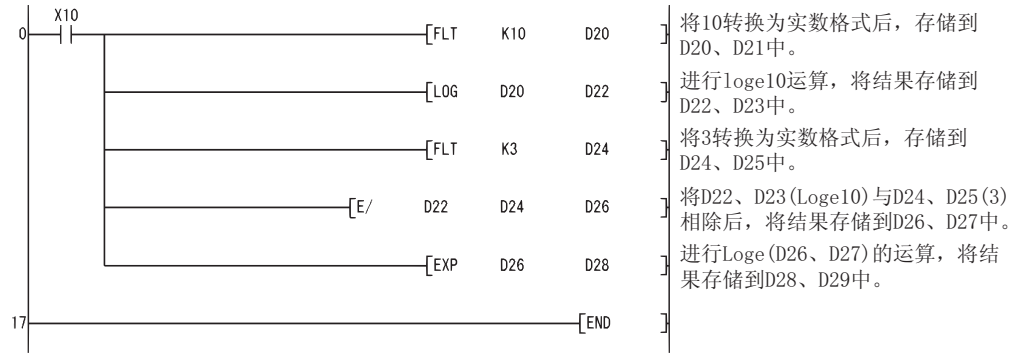
例如，$10^{1.2}$ 的运算等同于 $e^{(1.2\times \log e10)}$，通过顺控程序进行运算时的情况如下所示。



(2) 执行$\sqrt[n]{X}$ 运算的程序的思路

$\sqrt[n]{X}$ 可以通过 $e^{(\frac{1}{n}\log eX)}$ 进行运算。

例如，$\sqrt[3]{10}$ 的运算等同于 $e^{(\frac{1}{3}\times \log e10)}$，通过顺控程序进行运算时的情况如下所示。



备忘录

索引

索

索

索

索

索

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。

QCPU
编程手册

公共指令篇2/2

三菱电机自动化
400-821-3030
CALL CENTER 技术支持热线
周一至周五 9:00~17:00(法定节假日除外)

三菱电机自动化(上海)有限公司
地址：上海市黄浦区南京西路288号创兴金融中心17楼
邮编：200003
电话：021-23223030　传真：021-23223000
网址：www.meas.cn

书号	SH(NA)-080814CHN(2/2)-A(0903)STC
印号	STC-QCPU-CI(2/2)-PM(0903)