客服热线 400 - 820 - 9595

绵密网络 专业服务

中达电通已建立了 71 个分支机构及服务网点，并塑建训练有素的专业团队，提供客户最满意的服务，公司技术人员能在 2 小时内回应您的问题，并在 48 小时内提供所需服务。

上海 电话 :(021)6301-2827	南昌 电话 :(0791)8625-5010	合肥 电话 :(0551)6281-6777	南京 电话 :(025)8334-6585	杭州 电话 :(0571)8882-0610
武汉 电话 :(027)8544-8475	长沙 电话 :(0731)8549-9156	南宁 电话 :(0771)5879-599	厦门 电话 :(0592)5313-601	广州 电话 :(020)3879-2175
济南 电话 :(0531)8690-7277	郑州 电话 :(0371)6384-2772	北京 电话 :(010)8225-3225	天津 电话 :(022)2301-5082	太原 电话 :(0351)4039-475
乌鲁木齐 电话 :(0991)4678-141	西安 电话 :(029)8836-0780	成都 电话 :(028)8434-2075	重庆 电话 :(023)8806-0306	哈尔滨 电话 :(0451)5366-0643
沈阳 电话 :(024)2334-16123	长春 电话 :(0431)8892-5060			

台達 DXMC 系列運動控制使用手冊

台达 DXMC 系列运动控制器使用手册

中达电通股份有限公司

地址：上海市浦东新区民夏路238号
邮编：201209
电话：(021)5863-5678
传真：(021)5863-0003
网址：http://www.deltagreentech.com.cn

扫一扫，关注官方微信

DELTA_IA-GMC_DXMC_UM_SC_20200603

www.deltaww.com

Smarter. Greener. Together.

序言

感谢您使用本产品，本使用手册提供运动控制器 DXMC 系列(以下简称 DXMC 主机) 相关信息。
本手册内容包含

- DXMC 主机产品检查与型号说明
- DXMC 主机安装注意事项
- 配线方法
- DXMC 控制器 PLC 执行原理
- 所有参数说明
- PLC 操作接口与指令说明
- 运动控制应用
- 通讯功能说明
- 异常排除

DXMC 产品特色

DXMC 主机为高阶多轴同步运动控制器，最多可控制 32 轴实体轴与 64 轴虚拟轴。DXMC 系列提供完整的张力控制、旋切、追剪、电子凸轮等功能模块，方便用户以 Multiprog PLC 及台达运动控制模块编辑程序，完整的系统信息都整合在同一个控制核心中，提升了整套系统运算的实时性。产品内建标准 IEC61131-3 五种 PLC 编辑语法及 PLCopen 运动控制的完整功能块，可广泛的应用在包装机、印刷机、卷绕机、工业机器人等等，便于产业「智动化」与「智」造产业升级。

如何使用本操作手册

您可视本手册为学习使用 DXMC 之参考信息，手册将告诉您如何安装、设定、使用及维护本产品。在开始调机或设定前，请先阅读一到三章节。

其他相关手册

DXMC 主机搭配的软件为 DMARS；DMARS 软件操作手册内容包含 DMARS 软件操作方式、配置文件配置方式、电子凸轮编辑、示波器操作、以及 PLC 运动控制程序的编辑方式。

台达电子技术服务

如果您在使用上仍有问题，欢迎洽询经销商或本公司客服中心。

DXMC 主机可使用于工业应用场合上，在接收检验、安装、配线、操作、维护及检查时，应随时注意以下安全注意事项。

标志「危险」、「警告」及「禁止」代表之涵义：

意指可能潜藏危险，若未遵守可能会对人员造成严重或致命的伤害。

意指可能潜藏危险，若未遵守可能会对人员造成中度的伤害，或导致产品严重损坏，或甚至故障。

意指绝对禁止的行动，若未遵守可能会导致产品损坏，或甚至故障而无法使用。

DXMC-S 安全注意事项

安装注意

- 依照手册指定的方式安装控制器，否则可能导致设备损坏。
- 禁止将本产品暴露在有水气、腐蚀性气体、可燃性气体等物质的场所下使用，否则可能会造成产品损坏、触电、火灾或爆炸。
- 请勿将控制器安装在超过规格范围的温度环境中，否则可能造成控制器无法正常运作或损坏。
- 本产品为 KC Class A (商用设备) 产品且通过试验认证，其设计的目的是在商业或是工业环境使用，而非家庭环境中使用。
- 请勿将控制器用于可能会造成人员伤亡、设备损坏或系统停机等警报的机台。

配线注意

- 请将接地线连接到 class-3 (100Ω 以下) 接地，接地不良可能会造成通讯异常、触电或火灾。

操作注意

- 未经规划或确认之 PLC 程序可能会导致运转不正常。为避免操作人员伤害或设备损坏，规划 PLC 程序时，请确保控制器与设备之间的通讯良好，通讯异常将造成设备功能无法正常运作。
- 当机械设备开始运转前，须配合其使用者参数调整设定值。若未调整到相符的正确设定值，可能会导致机械设备运转失去控制或发生故障。

- 当马达运转时，禁止接触任何旋转中的马达零件，否则可能造成人员受伤。
- 不得在开启电源的情况下改变配线，否则可能造成触电或人员受伤。

保养及检查

- 禁止接触控制器内部，否则可能会造成触电或产品损坏。
- 电源关闭 10 分钟内，不得接触接线端子，残余电压可能造成触电。

配线方法

- 请勿使用超过控制器规格范围的电压，否则可能会引起产品损坏、触电或火灾。
- 对于错误强行拔出电线的动作，请重新检查连接电线再启动。

通讯电路的配线

- 用正确的接地回路，以避免通讯不良。

DXMC-P 安全注意事项

安装注意

- 依照手册指定的方式安装控制器，否则可能导致设备损坏。
- 禁止将本产品暴露在有水气、腐蚀性气体、可燃性气体等物质的场所下使用，否则可能会造成产品损坏、触电、火灾或爆炸。
- 请勿将控制器安装在超过规格范围的温度环境中，否则可能造成控制器无法正常运作或损坏。
- 本产品为 KC Class A (商用设备) 产品且通过试验认证，其设计的目的是在商业或是工业环境使用，而非家庭环境中使用。
- 请勿将控制器用于可能会造成人员伤亡、设备损坏或系统停机等警报的机台。

配线注意

- 请将接地线连接到 class-3 (100Ω 以下) 接地，接地不良可能会造成通讯异常、触电或火灾。

操作注意

- 控制器需配合编辑软件规划画面，未经规划或确认之控制器可能会导致不正常的运转结果。为避免操作时造成人身伤害或设备损坏，设置控制器时，要确保控制器及其连接设备之间的通讯故障不会造成设备功能无法正常运作。
- 为避免意外遗失程序，请务必备份规划好的控制器程序。
- 机器开始运转前，请确认是否可以随时启动紧急停机装置。

- 当马达运转时，禁止接触任何旋转中的马达零件，否则可能造成人员受伤。
- 不得在开启电源的情况下改变配线，否则可能造成触电或人员受伤。

保养及检查

- 禁止接触控制器内部，否则可能会造成触电或产品损坏。
- 电源关闭 10 分钟内，不得接触接线端子，残余电压可能造成触电。

配线方法

- 请勿使用超过控制器规格范围的电压，否则可能会引起产品损坏、触电或火灾。
- 对于错误强行拔出电线的动作，请重新检查连接电线再启动。

通讯电路的配线

- 用正确的接地回路，以避免通讯不良。

DXMC-H 安全注意事项

安装注意

- 依照手册指定的方式安装控制器，否则可能导致设备损坏。
- 禁止将本产品暴露在有水气、腐蚀性气体、可燃性气体等物质的场所下使用，否则可能会造成产品损坏、触电、火灾或爆炸。
- 请勿将控制器安装在超过规格范围的温度环境中，否则可能造成控制器无法正常运作或损坏。
- 本产品为 KC Class A (商用设备) 产品且通过试验认证，其设计的目的是在商业或是工业环境使用，而非家庭环境中使用。
- 请勿将控制器用于可能会造成人员伤亡、设备损坏或系统停机等警报的机台。

配线注意

- 请将接地线连接到 class-3 (100Ω 以下) 接地，接地不良可能会造成通讯异常、触电或火灾。
- 请勿任意架接其它线材延长本体线材长度，避免造成通讯或功能不良。

操作注意

- 控制器需配合编辑软件规划画面，未经规划或确认之控制器可能会导致不正常的运转结果。为避免操作时造成人身伤害或设备损坏，设置控制器时，要确保控制器及其连接设备之间的通讯故障不会造成设备功能无法正常运作。
- 为避免意外遗失程序，请务必备份规划好的控制器程序。
- 暂时不操作时，请将控制器平放于平面或利用机体背面挂勾固定，避免不慎造成控制器摔落损坏。

- 不得在开启电源的情况下改变配线，否则可能造成触电或人员受伤。
- 请勿以尖锐物品碰触面板，否则可能导致面板凹陷，进而使显示接口无法正常运作。

保养及检查

- 禁止接触控制器内部，否则可能会造成触电或产品损坏。
- 电源关闭 10 分钟内，不得接触接线端子，残余电压可能造成触电。

配线方法

- 请勿使用超过控制器规格范围的电压，否则可能会引起产品损坏、触电或火灾。
- 对于错误强行拔出电线的动作，请重新检查连接电线再启动。

通讯电路的配线

- 用正确的接地回路，以避免通讯不良。

注：各版本内容若略有差异，请以台达网站(https://www.deltaww.com/) 最新公布信息为主。

(此页有意留为空白)

目录

使用前

1 产品检查与型号说明

1.1 产品检查……1-2
1.2 产品型号对照……1-3
1.2.1 铭牌说明……1-3
1.2.2 型号说明……1-4

2 安装

2.1 注意事项……2-2
2.2 储存环境条件……2-2
2.3 安装环境条件……2-2
2.4 安装方向与空间……2-3
2.4.1 DXMC-S 机种安装方向与空间……2-3
2.4.2 DXMC-P 机种安装方向与空间……2-7
2.5 安装扩展模块 (DXMC-S 机种)……2-9

3 配线

3.1 硬件接口及配线……3-2
3.1.1 各部名称……3-2
3.1.1.1 DXMC-S 机种各部名称……3-2
3.1.1.2 DXMC-P 机种各部名称……3-4
3.1.1.3 DXMC-H 机种各部名称……3-8
3.1.2 动作状态指示灯 (DXMC-S 机种)……3-16
3.1.3 电源配线……3-17
3.1.4 接地方式……3-18
3.2 I/O 端子端口 (DXMC-S 机种)……3-19
3.2.1 I/O 端子说明……3-19
3.2.2 数字输入配线……3-21
3.2.3 数字输出配线图……3-22
3.3 EtherCAT 通信端口……3-23
3.3.1 DXMC-S 机种 EtherCAT 通信端口……3-23
3.3.2 DXMC-P 机种 EtherCAT 通信端口……3-24
3.4 USB 串行通讯端口……3-25

3.4.1 DXMC-S 机种 USB 串行通讯端口 3-25
3.4.2 DXMC-P 机种 USB 串行通讯端口 3-27
3.4.3 DXMC-H 机种 USB 串行通讯端口 3-28
3.5 Ethernet 通信端口 3-29
3.5.1 DXMC-S 机种 Ethernet 通信端口 3-29
3.5.2 DXMC-P 机种 Ethernet 通信端口 3-30
3.5.3 DXMC-H 机种 Ethernet 通信端口 3-31
3.6 外部通信端口 (DXMC-S 机种) 3-32
3.6.1 外部端子说明 3-32
3.6.2 外部端子接线图 3-34
3.7 DMCNET 通信端口 (DXMC-H 机种) 3-36

如何設定基礎配置

4 控制器 PLC 执行原理
4.1 任务 (Task) 4-2
4.1.1 任务类型 4-2
4.1.2 IO 执行原理 4-4
4.1.3 任务优先权 4-6

5 参数与功能
5.1 参数定义 5-2
5.2 控制器参数一览表 5-3
5.3 控制器参数说明 5-7
P0-x 共同设定 5-7
P1-x 版本信息 5-10
P2-x Ethernet 设定 5-13
P4-x USB 设定 5-16
P5-x 串行通信设置 5-17
P9-x 数据安全设定 5-20
P10-x PLC 设定 5-21
P11-x SSI 通讯型编码器设定 5-23
P12-x 脉冲型编码器设定 (第一组) 5-24
P13-x 脉冲型编码器设定 (第二组) 5-25
5.4 轴参数一览表 5-26
5.5 轴参数说明 5-29
P0-x 基本设定 5-29
P1-x 单位设定 5-31
P2-x 速度 / 加速度设定 5-34
P3-x 警告设定 5-35

P4-x 监测设定 ····· 5-37
P5-x 极限设定 ····· 5-39
P6-x 原点复归设定 ····· 5-40
P7-x 位置计数设定 ····· 5-42
P8-x 操作设定 ····· 5-43

如何設計程式

6 PLC 操作接口及功能块介绍

6.1 MULTIPROG 系统介绍 ····· 6-6
6.1.1 MULTIPROG 用户接口简介 ····· 6-7
6.1.1.1 功能区及常用工具栏 ····· 6-8
6.1.1.2 专案树 ····· 6-11
6.1.1.3 程序编辑区 ····· 6-12
6.1.1.4 功能块函式库 ····· 6-13
6.1.1.5 状态显示区 ····· 6-13
6.2 IEC 61131-3 概述与 MULTIPROG 系统 ····· 6-14
6.2.1 MULTIPROG 专案架构 ····· 6-16
6.2.2 Configuration (配置) ····· 6-17
6.2.3 Resource (资源) ····· 6-17
6.2.4 Task (任务) ····· 6-21
6.2.5 POU (程序组织单元) ····· 6-26
6.2.6 Data Type (数据型态) ····· 6-32
6.3 MULTIPROG 编程语言 ····· 6-38
6.3.1 指令表(IL)编程语言 ····· 6-38
6.3.2 梯形图(LD)编程语言 ····· 6-40
6.3.3 功能块图(FBD)编程语言 ····· 6-45
6.3.4 结构化文字(ST)编程语言 ····· 6-50
6.3.5 顺序功能图(SFC)编程语言 ····· 6-53
6.4 PLCopen 介绍 ····· 6-57
6.4.1 PLCopen 组织简介 ····· 6-57
6.4.2 PLCopen Motion Control 简介 ····· 6-58
6.4.3 常用变量定义 ····· 6-59
6.5 运动功能块介绍 ····· 6-64
6.5.1 运动功能块总览 ····· 6-64
6.5.2 单轴功能块 (Motion SingleAxis) ····· 6-68
6.5.2.1 MC_Home ····· 6-68
6.5.2.2 MC_Stop ····· 6-72
6.5.2.3 MC_Halt ····· 6-76
6.5.2.4 MC_Jog ····· 6-81
6.5.2.5 MC_MoveAbsolute ····· 6-85
6.5.2.6 MC_MoveRelative ····· 6-89
6.5.2.7 MC_MoveAdditive ····· 6-93

6.5.2.8 MC_MoveSuperimposed ····· 6-98
6.5.2.9 MC_SetPosition ····· 6-102
6.5.2.10 DMC_Home ····· 6-105
6.5.2.11 MC_MoveVelocity ····· 6-109
6.5.2.12 MC_Power ····· 6-113
6.5.2.13 MC_Reset ····· 6-115
6.5.2.14 MC_SetOverride ····· 6-117
6.5.2.15 MC_ReadParameter ····· 6-120
6.5.2.16 DMC_SetServoGain ····· 6-122
6.5.2.17 DMC_TouchProbe ····· 6-124
6.5.2.18 MC_AbortTrigger ····· 6-126
6.5.2.19 MC_WriteParameter ····· 6-127
6.5.2.20 MC_TorqueControl ····· 6-129
6.5.3 多轴功能块 (Motion MultiAxis) ····· 6-131
6.5.3.1 MC_GearIn ····· 6-131
6.5.3.2 MC_GearOut ····· 6-135
6.5.3.3 DMC_CamIn ····· 6-137
6.5.3.4 MC_CamOut ····· 6-154
6.5.3.5 MC_PhasingAbsolute ····· 6-156
6.5.3.6 MC_PhasingRelative ····· 6-159
6.5.3.7 DMC_GantrySynchornize ····· 6-162
6.5.4 群轴功能块 (Motion Group) ····· 6-164
6.5.4.1 MC_GroupStop ····· 6-164
6.5.4.2 MC_MoveDirectAbsolute ····· 6-167
6.5.4.3 MC_MoveDirectRelative ····· 6-171
6.5.4.4 MC_MoveLinearAbsolute ····· 6-175
6.5.4.5 MC_MoveLinearRelatve ····· 6-179
6.5.4.6 MC_GroupReset ····· 6-183
6.5.4.7 MC_GroupSetOverride ····· 6-185
6.5.5 系统功能块 (System Function) ····· 6-188
6.5.5.1 SYS_ParaRead ····· 6-188
6.5.5.2 SYS_ParaWrite ····· 6-189
6.5.5.3 SYS_GetAlarmCode ····· 6-191
6.5.5.4 SYS_ResetAlarmCode ····· 6-193
6.5.6 自定义系统功能块 (DMC Function) ····· 6-194
6.5.6.1 CEIL ····· 6-194
6.5.6.2 FLOOR ····· 6-195
6.5.6.3 Ring_Queue ····· 6-196
6.5.7 自定义通讯功能块 (DMC FieldBus) ····· 6-199
6.5.7.1 CAN_PDORead ····· 6-199
6.5.7.2 CAN_PDOWrite ····· 6-201
6.5.7.3 CAN_SDORead ····· 6-203
6.5.7.4 CAN_SDOWrite ····· 6-205
6.5.7.5 MB_ReadBits ····· 6-207
6.5.7.6 MB_ReadWords ····· 6-210

6.5.7.7 MB_SerialPort_Set ····· 6-213
6.5.7.8 MB_TCP_Connect ····· 6-215
6.5.7.9 MB_TCP_DisConnect ····· 6-218
6.5.7.10 MB_TCP_ReadBits ····· 6-220
6.5.7.11 MB_TCP_ReadWords ····· 6-223
6.5.7.12 MB_TCP_WriteBit ····· 6-226
6.5.7.13 MB_TCP_WriteBits ····· 6-229
6.5.7.14 MB_TCP_WriteWord ····· 6-233
6.5.7.15 MB_TCP_WriteWords ····· 6-236
6.5.7.16 MB_WriteBit ····· 6-239
6.5.7.17 MB_WriteBits ····· 6-242
6.5.7.18 MB_WriteWord ····· 6-246
6.5.7.19 MB_WriteWords ····· 6-249
6.5.8 凸轮编辑功能块 (ECAM Editor) ····· 6-253
6.5.8.1 ECAM_GenTableByFile ····· 6-253
6.5.8.2 ECAM_GenTableByData ····· 6-257
6.5.8.3 ECAM_GenTableByVel ····· 6-262
6.5.8.4 ECAM_AddPoints ····· 6-265
6.5.8.5 ECAM_DelPoints ····· 6-270
6.5.8.6 ECAM_ModifyPoints ····· 6-273
6.5.8.7 ECAM_ReadPointsByID ····· 6-278
6.5.8.8 ECAM_ReadTableByID ····· 6-282
6.5.8.9 ECAM_SaveTable ····· 6-286
6.5.8.10 ECAM_GenRotaryCutTable ····· 6-290
6.5.8.11 ECAM_GenFlyingShearTable ····· 6-293
6.5.8.12 ECAM_SetCamTappetData ····· 6-297
6.5.8.13 ECAM_GetCamTappetStatus ····· 6-301
6.5.8.14 ECAM_SetConnectVelocity ····· 6-304
6.5.8.15 ECAM_PreviewTable ····· 6-307
6.5.8.16 ECAM_ModRotCutVelRatio ····· 6-310
6.5.8.17 ECAM_SetOnlineChgMode ····· 6-312
6.5.9 自定义数据型态 ····· 6-315
6.5.9.1 自定义数据型态总览 ····· 6-315
6.5.9.2 任务信息数据型态 (Task Information) ····· 6-315
6.5.9.3 PLCOpen 数据型态 (PLCOpenDataType) ····· 6-316
6.5.9.4 ECAM 数据型态 (ECAMEditorDataType) ····· 6-321
6.6 标准功能 ····· 6-323
6.6.1 标准功能总览 ····· 6-323
6.6.2 指令启用说明 (EN 和 ENO 说明) ····· 6-325
6.6.3 数学运算功能 ····· 6-326
6.6.3.1 ABS ····· 6-326
6.6.3.2 ACOS ····· 6-327
6.6.3.3 ASIN ····· 6-328
6.6.3.4 ATAN ····· 6-329
6.6.3.5 COS ····· 6-330

6.6.3.6 EXP ····· 6-330
6.6.3.7 LN ····· 6-331
6.6.3.8 LOG ····· 6-331
6.6.3.9 SIN ····· 6-332
6.6.3.10 SQRT ····· 6-333
6.6.3.11 TAN ····· 6-334
6.6.3.12 ADD ····· 6-335
6.6.3.13 ADD_T_T ····· 6-336
6.6.3.14 DIV ····· 6-337
6.6.3.15 DIV_T_AI ····· 6-338
6.6.3.16 DIV_T_AN ····· 6-339
6.6.3.17 DIV_T_R ····· 6-340
6.6.3.18 EXPT ····· 6-341
6.6.3.19 MOD ····· 6-342
6.6.3.20 MOVE ····· 6-343
6.6.3.21 MUL ····· 6-344
6.6.3.22 MUL_T_AI ····· 6-345
6.6.3.23 MUL_T_AN ····· 6-346
6.6.3.24 MUL_T_R ····· 6-347
6.6.3.25 NEG ····· 6-348
6.6.3.26 SUB ····· 6-349
6.6.3.27 SUB_T_T ····· 6-350
6.6.4 位串功能 ····· 6-351
6.6.4.1 AND ····· 6-351
6.6.4.2 NOT ····· 6-352
6.6.4.3 OR ····· 6-353
6.6.4.4 XOR ····· 6-354
6.6.5 位移功能 ····· 6-355
6.6.5.1 ROL ····· 6-355
6.6.5.2 ROR ····· 6-356
6.6.5.3 SHL ····· 6-357
6.6.5.4 SHR ····· 6-358
6.6.6 选择运算功能 ····· 6-359
6.6.6.1 LIMIT ····· 6-359
6.6.6.2 MAX ····· 6-360
6.6.6.3 MIN ····· 6-361
6.6.6.4 SEL ····· 6-362
6.6.7 比较运算功能 ····· 6-363
6.6.7.1 EQ ····· 6-363
6.6.7.2 GE ····· 6-364
6.6.7.3 GT ····· 6-365
6.6.7.4 LE ····· 6-366
6.6.7.5 LT ····· 6-367
6.6.7.6 NE ····· 6-368
6.6.8 字符串操作功能 ····· 6-369

6.6.8.1 CONCAT ····· 6-369
6.6.8.2 DELETE ····· 6-370
6.6.8.3 EQ_STRING ····· 6-371
6.6.8.4 FIND ····· 6-372
6.6.8.5 GE_STRING ····· 6-373
6.6.8.6 GT_STRING ····· 6-374
6.6.8.7 INSERT ····· 6-375
6.6.8.8 LE_STRING ····· 6-376
6.6.8.9 LEFT ····· 6-377
6.6.8.10 LEN ····· 6-378
6.6.8.11 LT_STRING ····· 6-379
6.6.8.12 MID ····· 6-380
6.6.8.13 NE_STRING ····· 6-381
6.6.8.14 REPLACE ····· 6-382
6.6.8.15 RIGHT ····· 6-383

7 运动控制应用

7.1 抓取 (Capture) 功能说明 ····· 7-2
7.1.1 DMC_TouchProbe 功能块说明 ····· 7-2
7.1.2 高速抓取范例 (Driver Mode) ····· 7-6
7.1.3 高速抓取范例 (Controller Mode) ····· 7-7
7.1.4 低速抓取范例 ····· 7-8
7.2 电子凸轮 (E-CAM) 功能说明 ····· 7-9
7.2.1 飞剪功能 ····· 7-11
7.2.2 追剪功能 ····· 7-13

8 通讯功能

8.1 通讯参数设定 ····· 8-2
8.2 MODBUS 通讯协议 ····· 8-4
8.3 自由口通讯协议 ····· 8-18

如何排除問題

9 报警排除

9.1 报警一览表 ····· 9-3
9.2 报警原因与处置 ····· 9-24
9.3 SDO 终止传输代码 ····· 9-104

附錄

A MODBUS 地址表

A.1 MODBUS 站号说明 ····· A-2
A.2 分区说明 ····· A-2
A.3 DV、SDV 区说明 ····· A-3
A.4 DH、SDH 区说明 ····· A-7
A.5 MODBUS 设定范例 ····· A-8

1 产品检查与型号说明

使用 DXMC 前，请注意此章节所列的注意事项与型号相关说明。不同的机种对应不同的硬件样式，用户可根据需求来选择适合的机种。

1.1 产品检查 …… 1-2
1.2 产品型号对照 …… 1-3
1.2.1 铭牌说明 …… 1-3
1.2.2 型号说明 …… 1-4

1.1 产品检查

1

完整可操作的组件应包括：

(1) DXMC 运动控制器主机。

(2) 使用于 I/O 模块的 40 PIN 接头。

(3) 使用于外部通讯的 20 PIN 接头。

(4) 直流电源输入 3 PIN 快速接头 (24V、0V、GND)。

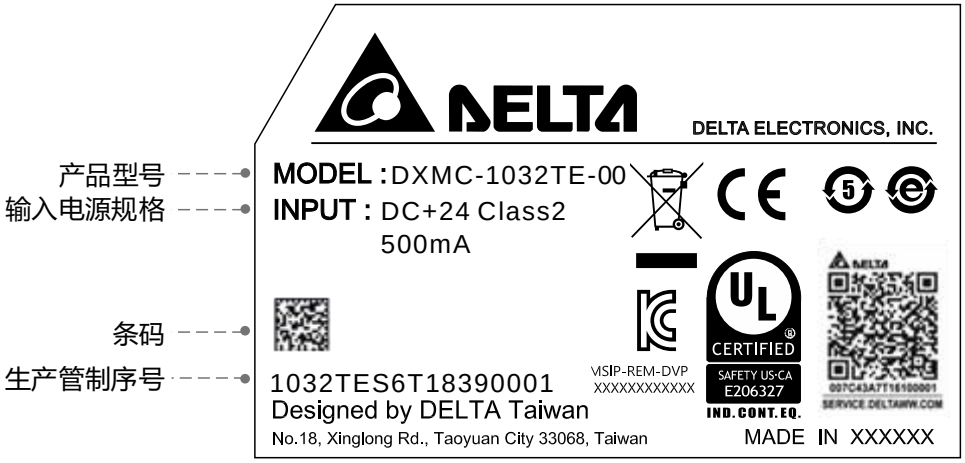
(5) 一个铝轨固定扣。

(6) 一本安装手册。

1.2 产品型号对照

1.2.1 铭牌说明

■ 铭牌说明



■ 序号说明

1032TES6 T 18 39 0001
(1) (2) (3) (4) (5)

(1) 机种型号

(2) 制造工厂 (T：桃园厂；W：吴江厂)

(3) 生产年份 (18：2018 年)

(4) 生产周次 (从 1 至 52)

(5) 制造序号 (一周内制造序号，从 0001 开始)

1.2.2 型号说明

1

DXMC − 1 S 32 T E − 0 0 A
(1) (2)(3)(4) (5)(6) (7)(8)(9)

(1) 主产品名称
DXMC：Delta eXtended Motion Controller

(2) 产品系列
1：1000 系列

(3) 形式
S：标准型
P：显控一体型
H：手持显控一体型

(4) 控制轴数
04：4 轴
06：6 轴
08：8 轴
16：16 轴
32：32 轴

(5) 内建 I/O
N：N/A
T：NPN (晶体输出)
A：手轮 + 三段限动开关 + 2 组急停开关 (NC)
C：手轮 + 二段限动开关 + 2 组急停开关 (NC)

(6) 运动总线
D：DMCNET
E：EtherCAT

(7) 显示屏幕尺寸

0：N/A

4：4 吋 LCM

7：7 吋 LCM

8：8 吋 LCM

A：10 吋 LCM

C：12 吋 LCM

F：15 吋 LCM

(8) 通讯接口

0：具有网络接口及串行通讯端口

S：不具有网络接口，只具备串行通讯端口

(9) 出线长度 (仅适用 DXMC-H)

0：N/A

3：3 米

5：5 米

A：10 米

F：15 米

K：20 米

(此页有意留为空白)

1

2 安装

在安装产品前，用户可依照此章节提到的注意事项、储存及安装环境等条件来进行安装。

2.1 注意事项 …… 2-2
2.2 储存环境条件 …… 2-2
2.3 安装环境条件 …… 2-2
2.4 安装方向与空间 …… 2-3
2.4.1 DXMC-S 机种安装方向与空间 …… 2-3
2.4.2 DXMC-P 机种安装方向与空间 …… 2-7
2.5 安装扩展模块 (DXMC-S 机种) …… 2-9

2.1 注意事项

2

请用户特别注意下列事项：

- 为了防止 DXMC 运动控制器温度上升，请勿安装于控制箱内的底部或顶部。
- 于高频干扰设备、高磁场/电场或电源线通过之场所，请充分采取遮蔽措施。

2.2 储存环境条件

本产品在安装之前必须置于其包装箱内，若暂不使用，为了使该产品能够符合本公司的保固范围及日后的维护，储存时务必注意下列事项：

- 储存位置的环境温度必须在-25°C 到+70°C 范围内。
- 储存位置的相对湿度必须在 10%到 95%范围内，且无结露。
- 避免储存于含有腐蚀性气体之环境中。

2.3 安装环境条件

安装 DXMC 运动控制器与运转环境的条件：无发高热装置、无水滴、蒸气、灰尘及油性灰尘、无腐蚀、易燃性之气、液体、无漂浮性的尘埃及金属微粒、坚固无振动、无电磁噪声干扰之场所。

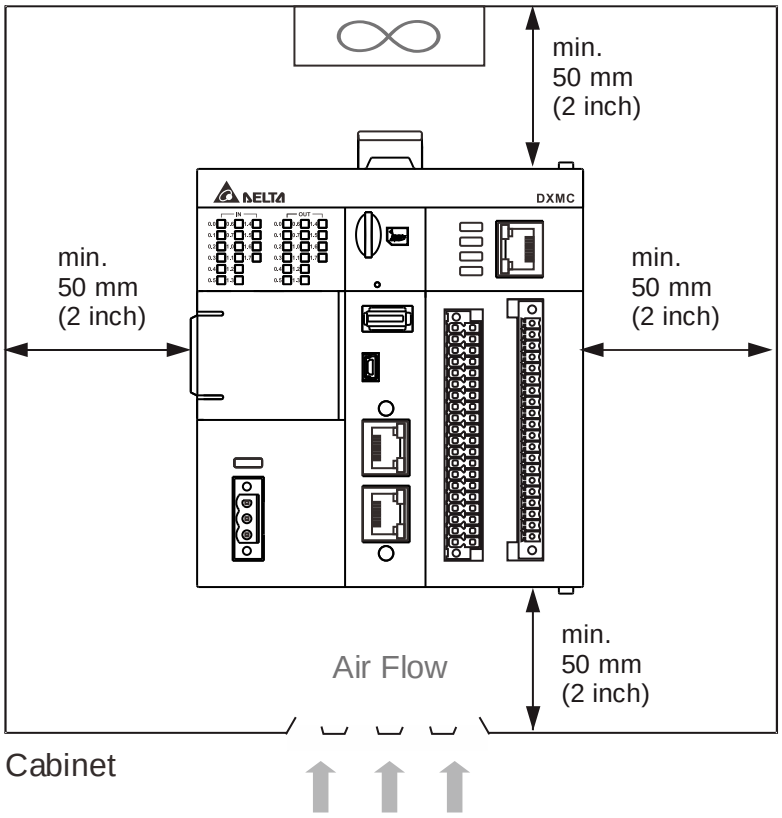
本产品使用环境温度为 0°C ~ 50°C。若环境温度超过 45°C 以上，请置于通风良好之场所。若需长时间运转，建议在 45°C 以下的环境温度，以确保产品性能。

2.4 安装方向与空间

2.4.1 DXMC-S 机种安装方向与空间

注意事项:

- 散热时需要较低的风阻，才能有效地排出热量。安装于控制箱内时，DXMC 运动控制器其周围应保持 50 mm，以确保散热正常。
- 请勿安装于高发热量之设备正上方。
- 若环境温度超过 50°C 时，请务必安装风扇或空调。



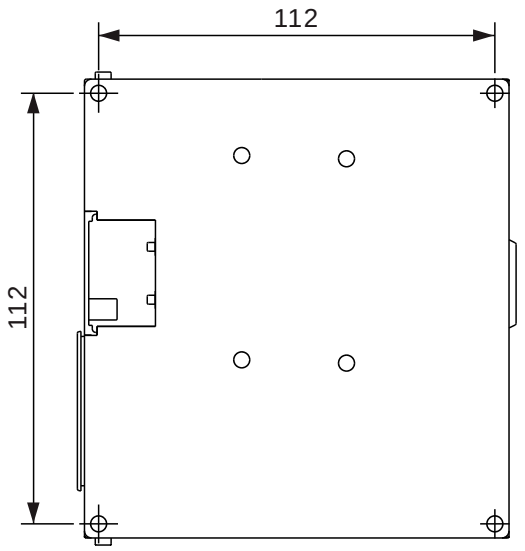
注：安装图文件之间隔距离与文字批注非等比例尺寸，请以文字批注为准。

2

安装示意图

■ 螺丝固定方式

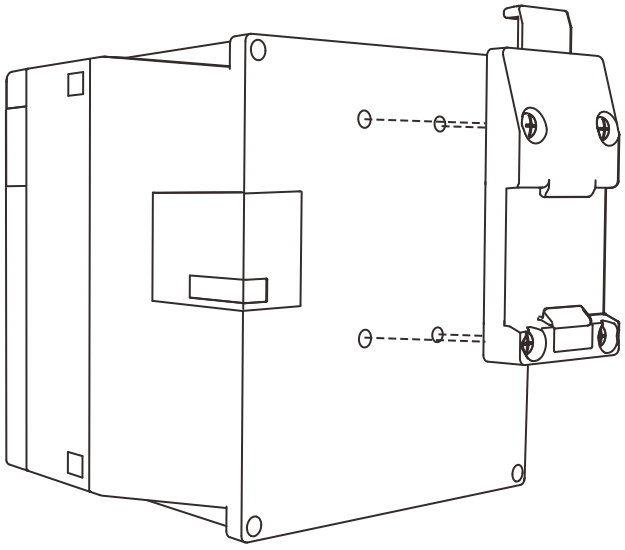
请依照图示中 DXMC 运动控制器的指定孔位，将螺丝（M4）固定于安装平面。
请自行评估螺丝长度、螺纹粗细及螺帽使用与否。



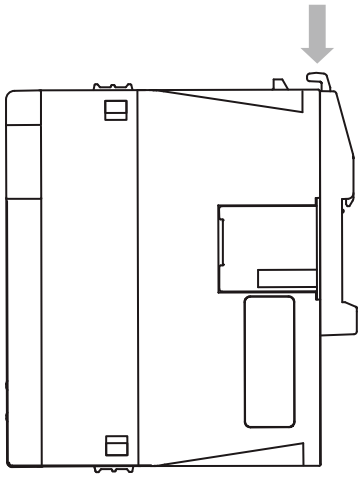
Unit: mm

■ 铝轨固定方式

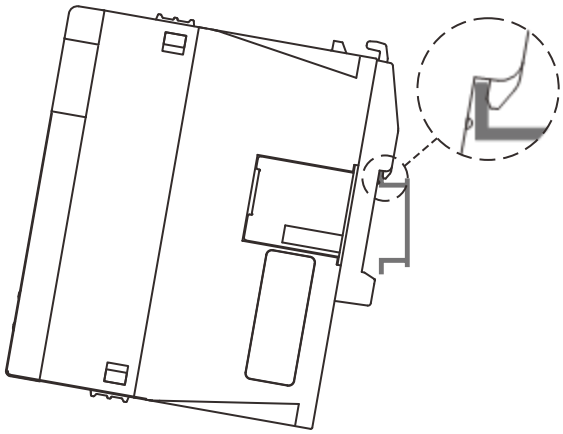
(1) 先将铝轨固定扣安装于 DXMC 运动控制器。



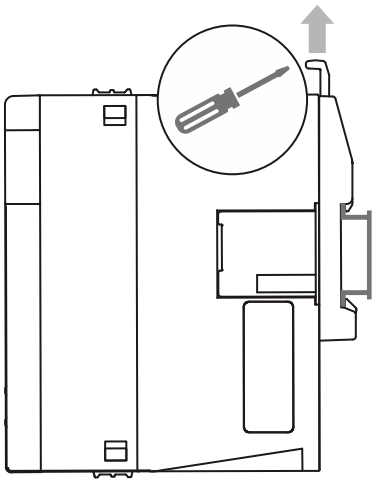
(2) 将铝轨固定扣上方的固定杆往下扣压。



(3) 将 DXMC 运动控制器架在铝轨上。

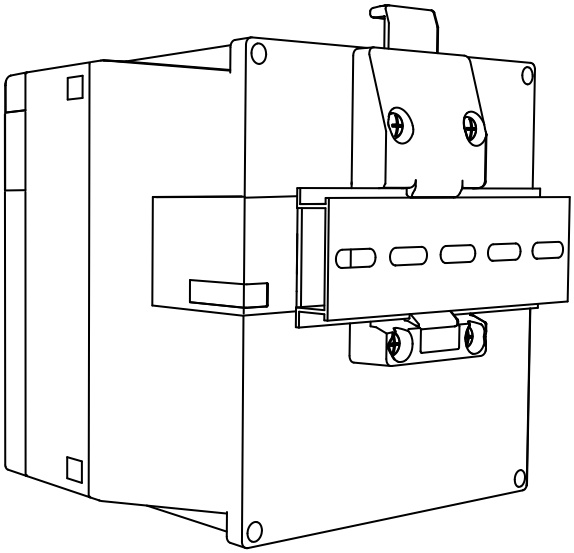


(4) 使用螺丝起子将固定杆往上拉起，完成固定。



2

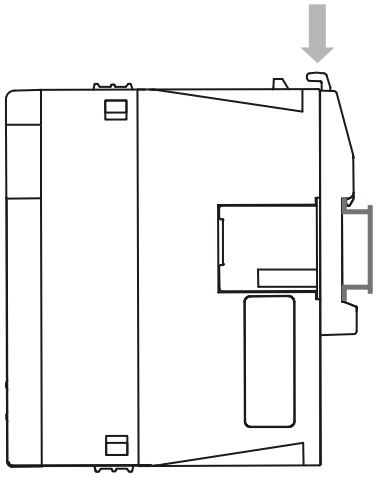
(5) DXMC 运动控制器安装完成示意图。



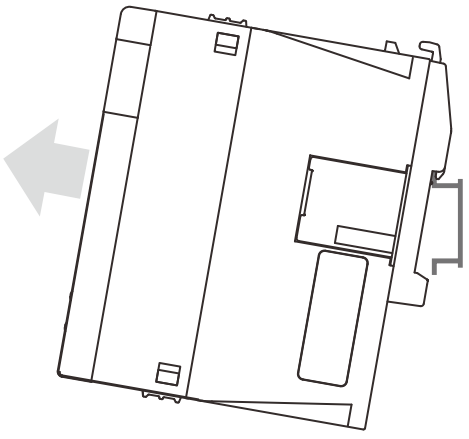
注：本产品提供的铝轨固定扣适用于 35 mm 之铝轨。

■ 将 DXMC 运动控制器从铝轨取下的方法

(1) 将上方固定杆向下扣压。



(2) 将 DXMC 运动控制器取下。



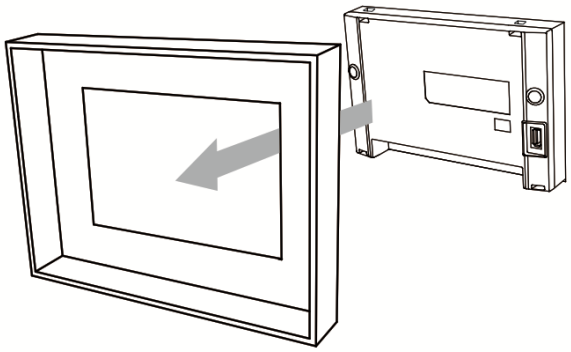
2.4.2 DXMC-P 机种安装方向与空间

注意事项:

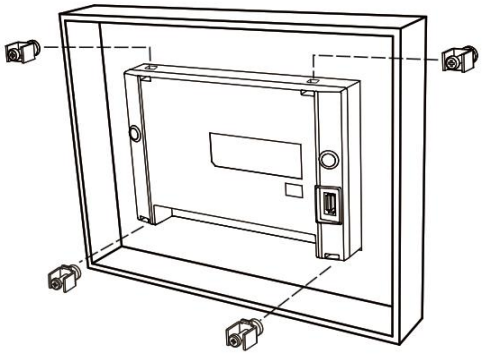
- 安装方向必须依图面所示，否则会造成故障。
- 为了使冷却循环效果良好，安装控制器时，其上下左右与相邻的物品和挡板（墙）必须保持足够的空间，否则会造成散热不良。
- 使用于 Type 4X 室内用等级之外壳平面。
- 安装面板最大板厚请勿超过 5 mm。

安装示意图

(1) 请确认套上防水垫圈，然后再安装控制器。



(2) 请确认将固定片螺丝组装入内，然后下方勾住前盖螺丝头顶住控制箱内侧。



2

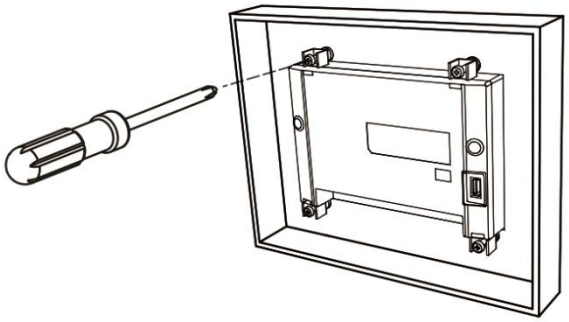
(3) 请以 0.5 ~ 0.7 N-m 的扭力锁紧螺丝。请勿超过此扭力，否则将损坏塑料外壳。

DXMC-1P08NE-70 扭力：6.17 lb-inch (0.7 N-m)

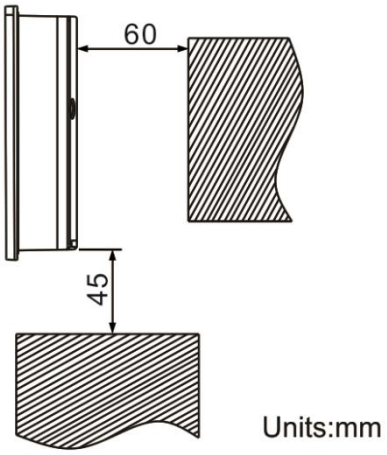
DXMC-1P08NE-7S 扭力：6.17 lb-inch (0.7 N-m)

DXMC-1P08NE-A0 扭力：6.17 lb-inch (0.7 N-m)

DXMC-1P08NE-4S 扭力：4.41 lb-inch (0.5 N-m)



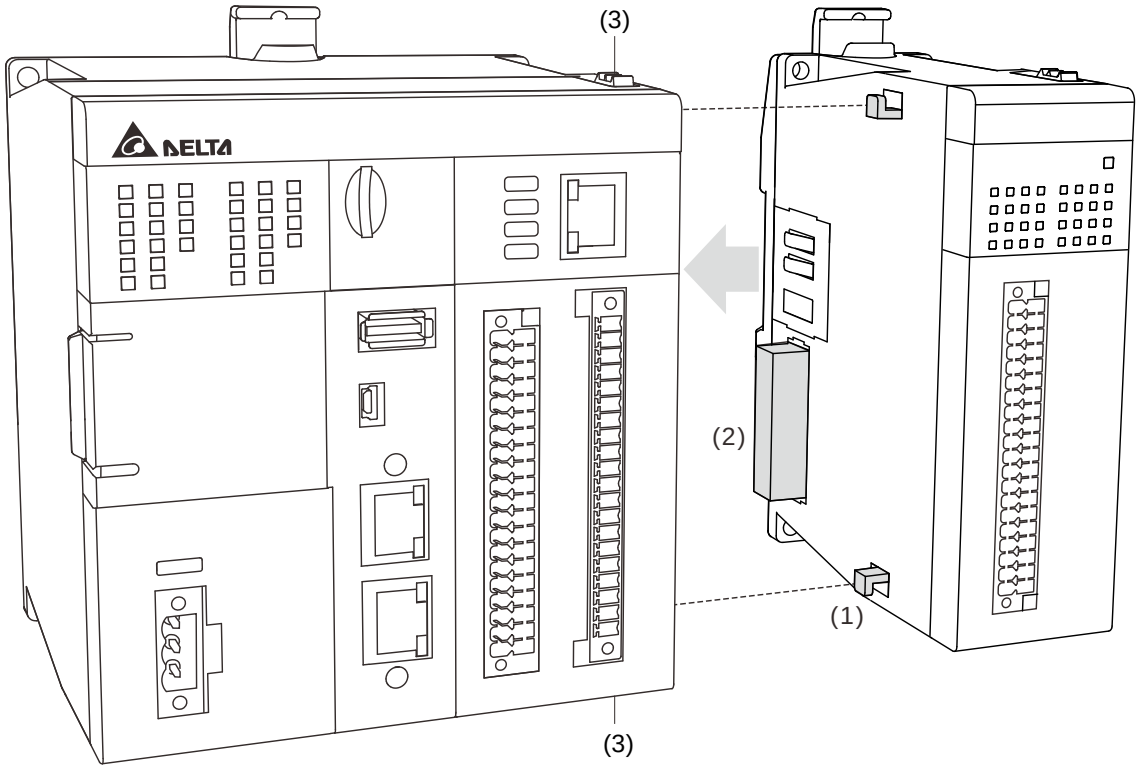
(4) 安装时，请在控制器后方及下方各预留 60 mm 及 45 mm 的散热空间。



2.5 安装扩展模块 (DXMC-S 机种)

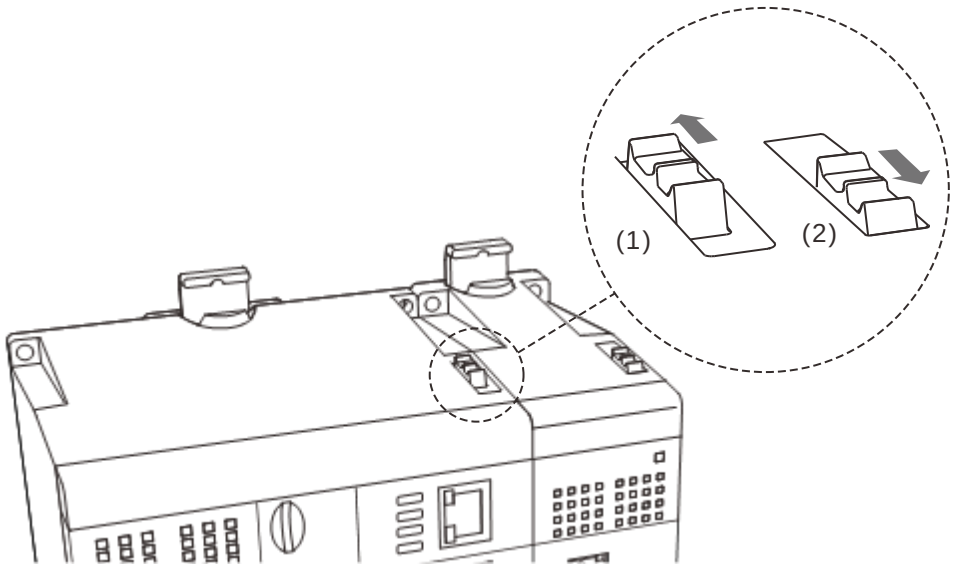
DXMC 运动控制器安装扩展模块的方式如下所述。

(1) 如下图所示，将扩展模块与 DXMC 运动控制器侧面的连接器进行咬合，然后将滑片锁定即可固定。



(1) 挂勾；(2) 连接器；(3) 滑片

(2) 滑动 DXMC 本体上方及下方的白色滑片直到听见滑片卡榫固定的声音即可。



(1) 滑片锁定；(2) 滑片解除

2

请用户特别注意下列事项：

- 安装扩展模块时，请务必切断电源。
- 根据不同的 DXMC 机型，可连接扩展装置的数量也有所不同。若超出数量上限，DXMC 运动控制器将无法正常运作。

配线 3

本章节说明 DXMC 主机之电源回路接线方法与各个接头的定义和配接方式，并列出各种控制模式的标准接线图。

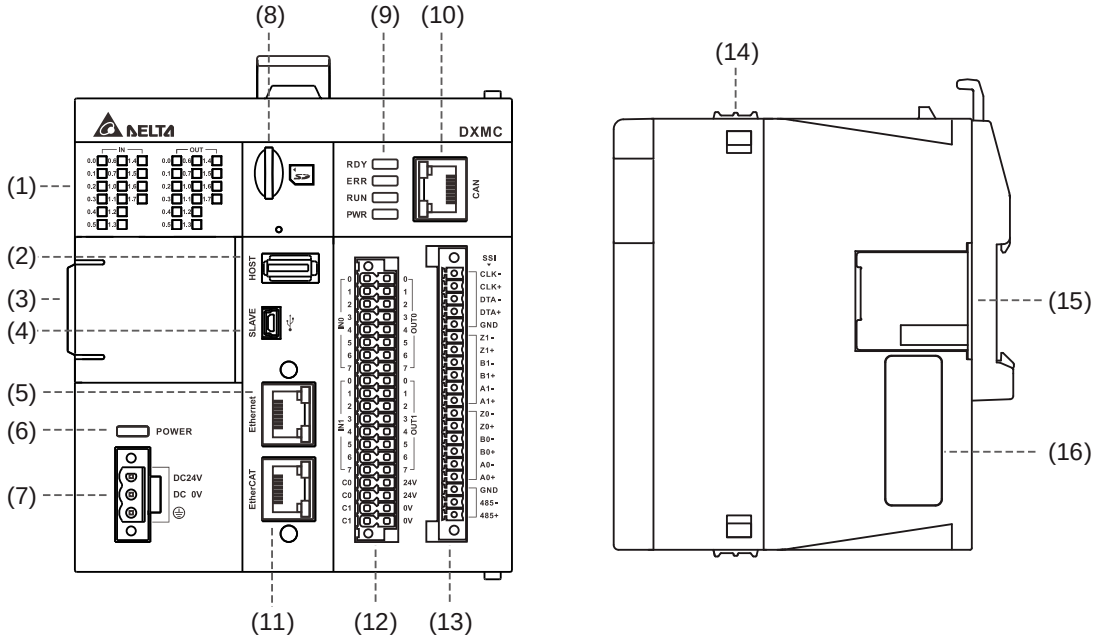
3.1 硬件接口及配线 …… 3-2
3.1.1 各部名称 …… 3-2
3.1.1.1 DXMC-S 机种各部名称 …… 3-2
3.1.1.2 DXMC-P 机种各部名称 …… 3-4
3.1.1.3 DXMC-H 机种各部名称 …… 3-8
3.1.2 动作状态指示灯 (DXMC-S 机种) …… 3-16
3.1.3 电源配线 …… 3-17
3.1.4 接地方式 …… 3-18
3.2 I/O 端子端口 (DXMC-S 机种) …… 3-19
3.2.1 I/O 端子说明 …… 3-19
3.2.2 数字输入配线 …… 3-21
3.2.3 数字输出配线图 …… 3-22
3.3 EtherCAT 通信端口 …… 3-23
3.3.1 DXMC-S 机种 EtherCAT 通信端口 …… 3-23
3.3.2 DXMC-P 机种 EtherCAT 通信端口 …… 3-24
3.4 USB 串行通讯端口 …… 3-25
3.4.1 DXMC-S 机种 USB 串行通讯端口 …… 3-25
3.4.2 DXMC-P 机种 USB 串行通讯端口 …… 3-27
3.4.3 DXMC-H 机种 USB 串行通讯端口 …… 3-28
3.5 Ethernet 通信端口 …… 3-29
3.5.1 DXMC-S 机种 Ethernet 通信端口 …… 3-29
3.5.2 DXMC-P 机种 Ethernet 通信端口 …… 3-30
3.5.3 DXMC-H 机种 Ethernet 通信端口 …… 3-31
3.6 外部通信端口 (DXMC-S 机种) …… 3-32
3.6.1 外部通讯端子说明 …… 3-32
3.6.2 外部通讯端子接线图 …… 3-34
3.7 DMCNET 通信端口 (DXMC-H 机种) …… 3-36

3.1 硬件接口及配线

3

3.1.1 各部名称

3.1.1.1 DXMC-S 机种各部名称



编号	名称	说明
(1)	I/O 端子状态指示灯	每个 LED 指示灯对应显示 DXMC 运动控制器内建 I/O 端子状态。
(2)	USB 串行通讯端 (HOST)	连接 U 盘口。
(3)	3V 锂电池	锂电池用来维持 Real-time clock 运作。若电池电压低于 2.4V，系统时间可能会发生异常。 注：电池电压过低会导致新下载的配置文件无法生效。
(4)	USB 串行通讯端 (SLAVE)	透过 Mini USB 与 PC 连接。
(5)	Ethernet 通信端口	透过 Ethernet 网络与 PC 连接。
(6)	电源指示灯	显示 DC 24V 电源供应情况。
(7)	直流电源输入端	连接 DC 24V 直流电源。 (请使用符合 Class 2 标准之电源。)
(8)	MicroSD 卡槽	连接 MicroSD 记忆卡。
(9)	动作状态指示灯	使用 4 个 LED 显示 DXMC 运动控制器的动作状态。 (详细说明请见章节 3.1.2。)
(10)	CANopen 通信端口	透过 CANopen 与其他装置连接。

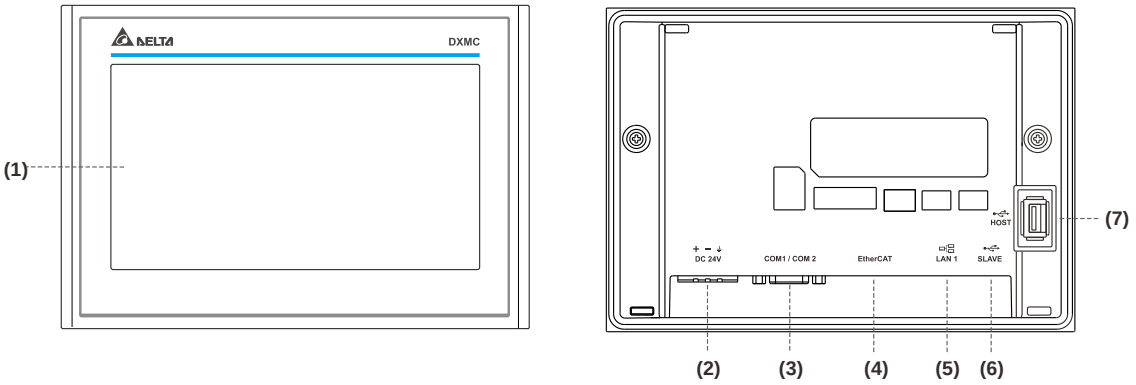
编号	名称	说明
(11)	EtherCAT 通信端口	透过 EtherCAT 与其他装置连接。
(12)	I/O 端子端口	16 组 DI 及 16 组 DO。
(13)	外部通信端口	连接外部编码器或 RS-485 通讯装置。
(14)	滑片	用于固定 DXMC 运动控制器与扩展模块。
(15)	接地弹片	扩展模块接地用。
(16)	扩展模块接口	用于连接扩展模块的连接器。

注：确保 24V、0V 的电源和接线正确。

3.1.1.2 DXMC-P 机种各部名称

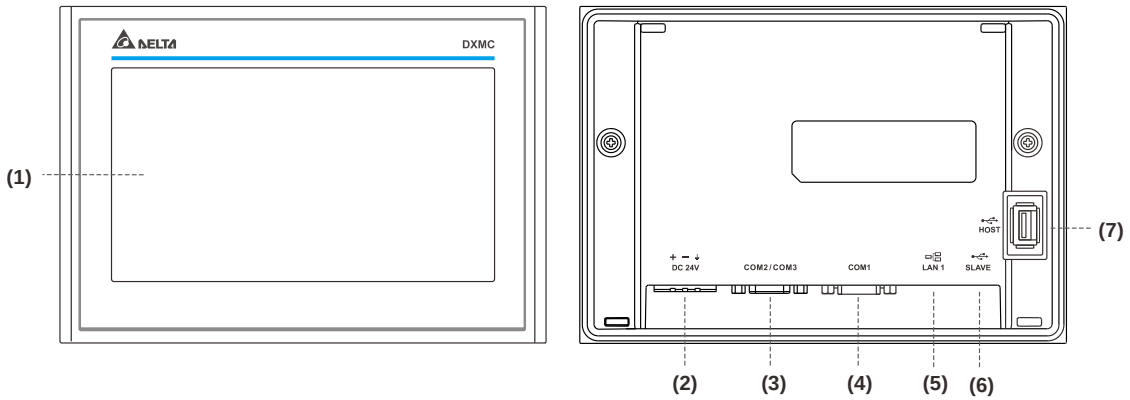
3

■ DXMC-1P08NE-70 机种



编号	名称	说明
(1)	操作 / 显示区域	触控面板可以用来操作与显示。
(2)	电源输入端子 (24AWG wire min.)	连接 DC 24V 直流电源。(请使用符合 Class 2 标准之电源。)
(3)	COM 1 / COM 2	透过 COM port 与其他装置连接。
(4)	EtherCAT 通信端口	透过 EtherCAT 与其他装置连接。
(5)	Ethernet 通信端口	透过 Ethernet 网络与 PC 连接。
(6)	USB Slave	透过 USB Cable 与 PC 连接。
(7)	USB Host	连接 U 盘口。

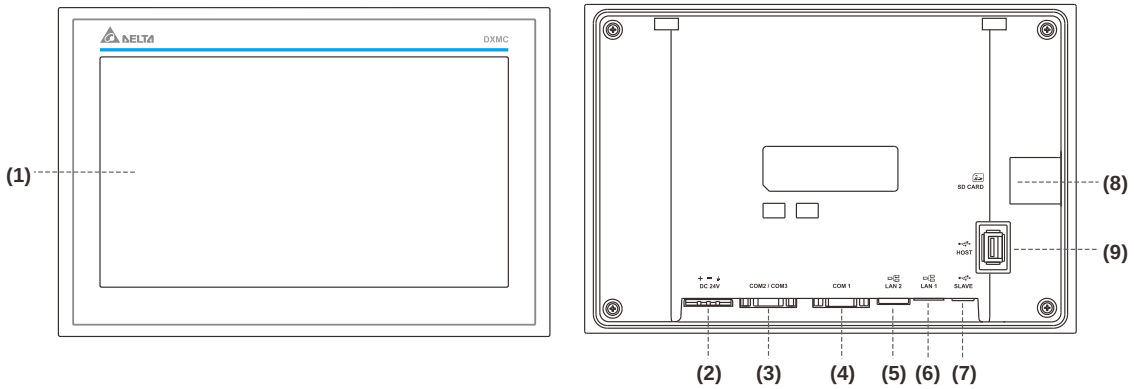
■ DXMC-1P08NE-7S 机种



编号	名称	说明
(1)	操作 / 显示区域	触控面板可以用来操作与显示。
(2)	电源输入端子 (24AWG wire min.)	连接 DC 24V 直流电源。(请使用符合 Class 2 标准之电源。)
(3)	COM 2 / COM 3	透过 COM port 与其他装置连接。
(4)	COM 1	透过 COM port 与其他装置连接。
(5)	EtherCAT 通信端口	透过 EtherCAT 与其他装置连接。
(6)	USB Slave	透过 USB Cable 与 PC 连接。
(7)	USB Host	连接 U 盘口。

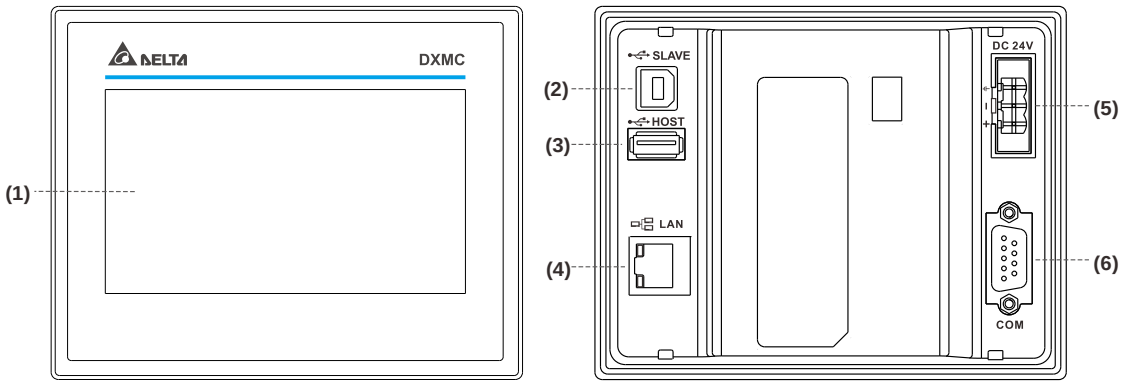
3

■ DXMC-1P08NE-A0 机种



编号	名称	说明
(1)	操作 / 显示区域	触控面板可以用来操作与显示。
(2)	电源输入端子 (24AWG wire min.)	连接 DC 24V 直流电源。(请使用符合 Class 2 标准之电源。)
(3)	COM 2 / COM 3	透过 COM port 与其他装置连接。
(4)	COM 1	透过 COM port 与其他装置连接。
(5)	EtherCAT 通信端口	透过 EtherCAT 与其他装置连接。
(6)	Ethernet 通信端口	透过 Ethernet 网络与 PC 连接。
(7)	USB Slave	透过 USB Cable 与 PC 连接。
(8)	SD 卡槽	连接 SD 记忆卡。
(9)	USB Host	连接 U 盘口。

■ DXMC-1P08NE-4S 机种

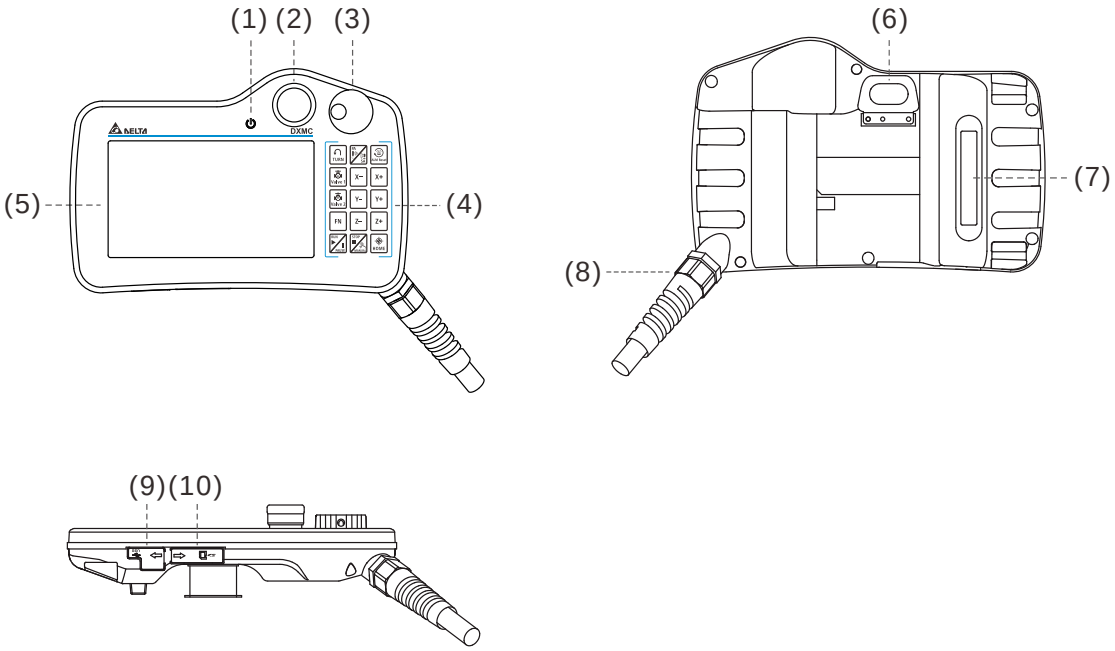


编号	名称	说明
(1)	操作 / 显示区域	触控面板可以用来操作与显示。
(2)	USB Slave	透过 USB Cable 与 PC 连接。
(3)	USB Host	连接 U 盘口。
(4)	EtherCAT 通信端口	透过 EtherCAT 与其他装置连接。
(5)	电源输入端子 (24AWG wire min.)	连接 DC 24V 直流电源。(请使用符合 Class 2 标准之电源。)
(6)	COM	透过 COM port 与其他装置连接。

3.1.1.3 DXMC-H 机种各部名称

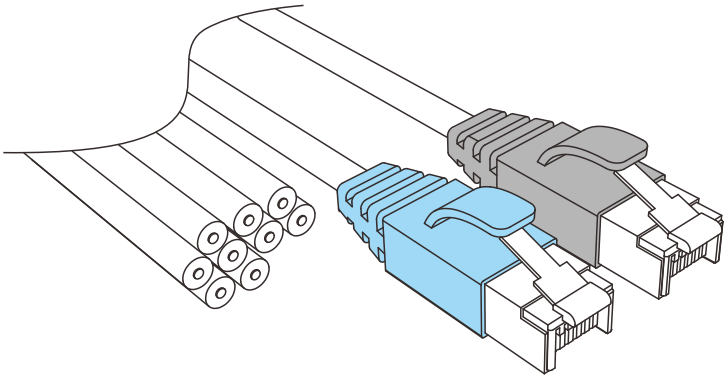
3

■ DXMC-1H08AD-70x 机种



编号	名称	说明
(1)	电源指示灯	系统电源指示灯。
(2)	紧急停止按钮	紧急停止按钮。
(3)	手摇轮	提供手摇轮功能。
(4)	辅助键盘	提供 15 个辅助键，用户可自行定义功能。
(5)	操作 / 显示区域	触控面板可以用来操作与显示。
(6)	挂勾	可使用挂勾固定，避免控制器摔落损坏。
(7)	三段式操作按钮	提供三段式开关，用户可自行定义功能。
(8)	总线	包含 DMCNET 运动总线、紧急停止等连接线。
(9)	USB Slave	透过 USB Cable 与 PC 连接。
(10)	SD Card 卡槽	连接 SD Card。

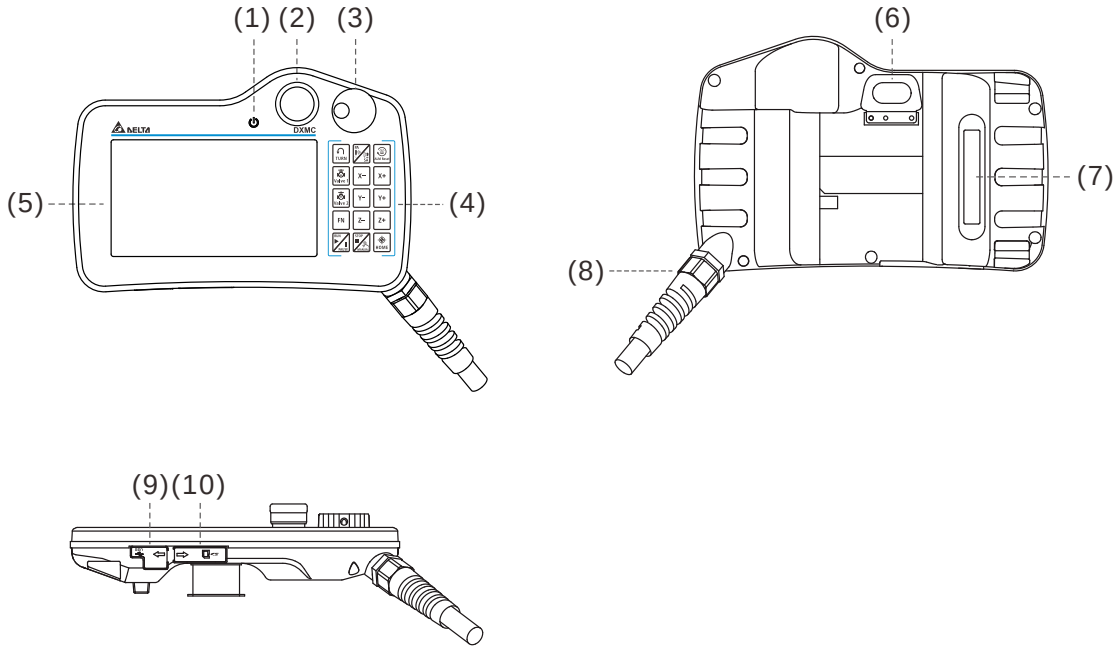
尾端出线型式



线材颜色	功能
RJ-45 (蓝)	DMCNET
RJ-45 (黑)	Ethernet
白 / 紫	紧急停止按钮 1 – NC (B 接点)
白 / 绿	紧急停止按钮 1 – NC (B 接点)
蓝	紧急停止按钮 2 – NC (B 接点)
白 / 蓝	紧急停止按钮 2 – NC (B 接点)
白 / 黑	保留
白 / 红	保留
红	DC 24V
黑	0V
白	FGND (电源接地线)

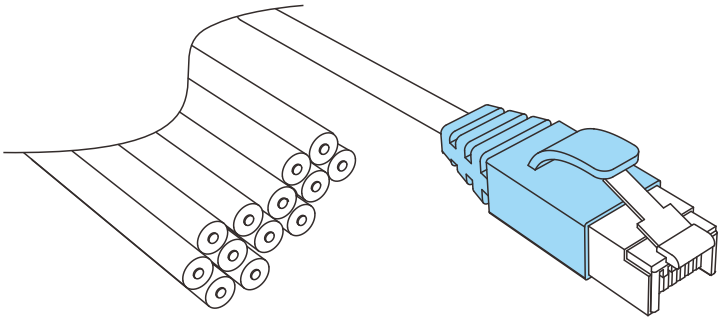
3

■ DXMC-1H08AD-7Sx 机种



编号	名称	说明
(1)	电源指示灯	系统电源指示灯。
(2)	紧急停止按钮	紧急停止按钮。
(3)	手摇轮	提供手摇轮功能。
(4)	辅助键盘	提供 15 个辅助键，用户可自行定义功能。
(5)	操作 / 显示区域	触控面板可以用来操作与显示。
(6)	挂勾	可使用挂勾固定，避免控制器摔落损坏。
(7)	三段式操作按钮	提供三段式开关，用户可自行定义功能。
(8)	总线	包含 DMCNET 运动总线、紧急停止等连接线。
(9)	USB Slave	透过 USB Cable 与 PC 连接。
(10)	SD Card 卡槽	连接 SD Card。

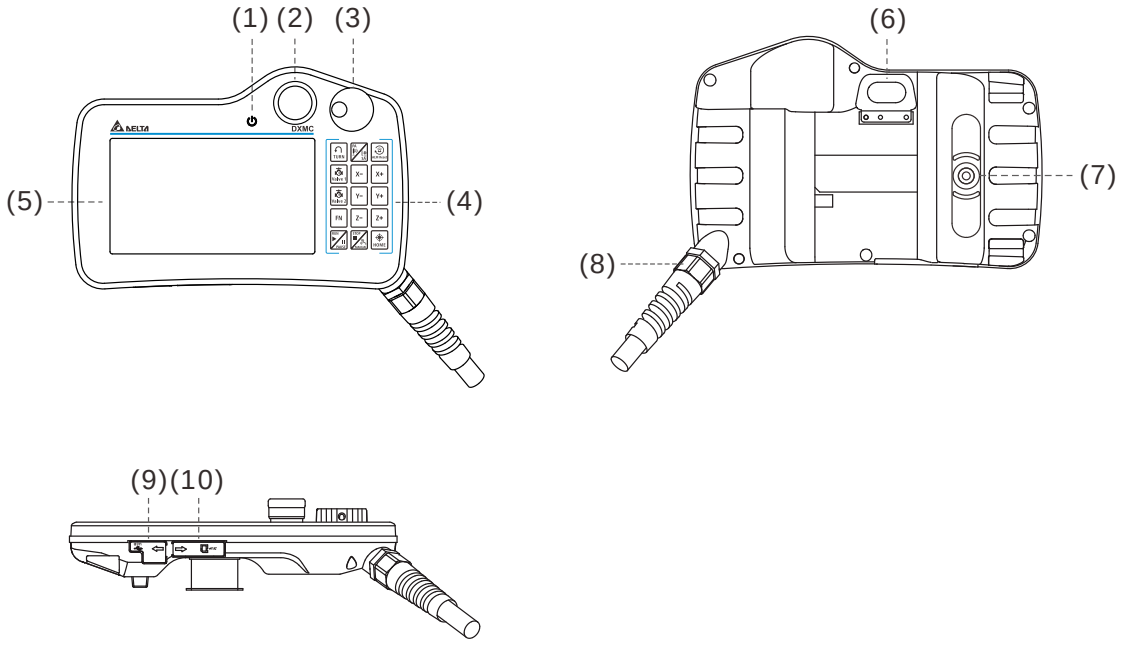
尾端出线型式



线材颜色	功能
RJ-45 (蓝)	DMCNET
黄	RS-485 (+)
白 / 黄	RS-485 (-)
白 / 棕	通讯接地
白 / 红	通讯接地
白 / 紫	紧急停止按钮 1 – NC (B 接点)
白 / 绿	紧急停止按钮 1 – NC (B 接点)
蓝	紧急停止按钮 2 – NC (B 接点)
白 / 蓝	紧急停止按钮 2 – NC (B 接点)
白 / 黑	保留
白 / 黑	保留
红	DC 24V
黑	0V
白	FGND (电源接地线)

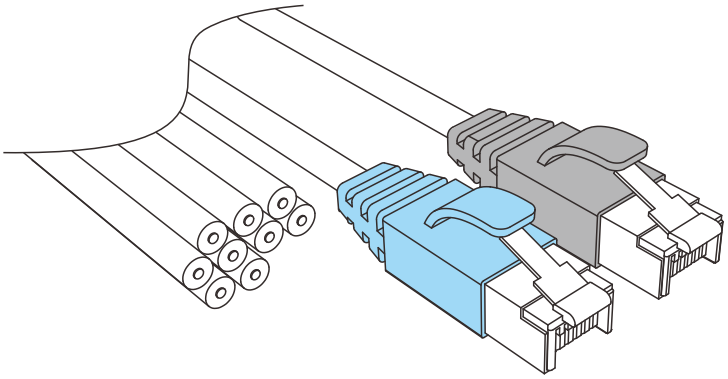
3

■ DXMC-1H08CD-70x 机种



编号	名称	说明
(1)	电源指示灯	系统电源指示灯。
(2)	紧急停止按钮	紧急停止按钮。
(3)	手摇轮	提供手摇轮功能。
(4)	辅助键盘	提供 15 个辅助键，用户可自行定义功能。
(5)	操作 / 显示区域	触控面板可以用来操作与显示。
(6)	挂勾	可使用挂勾固定，避免控制器摔落损坏。
(7)	三段式操作按钮	提供三段式开关，用户可自行定义功能。
(8)	总线	包含 DMCNET 运动总线、紧急停止等连接线。
(9)	USB Slave	透过 USB Cable 与 PC 连接。
(10)	SD Card 卡槽	连接 SD Card。

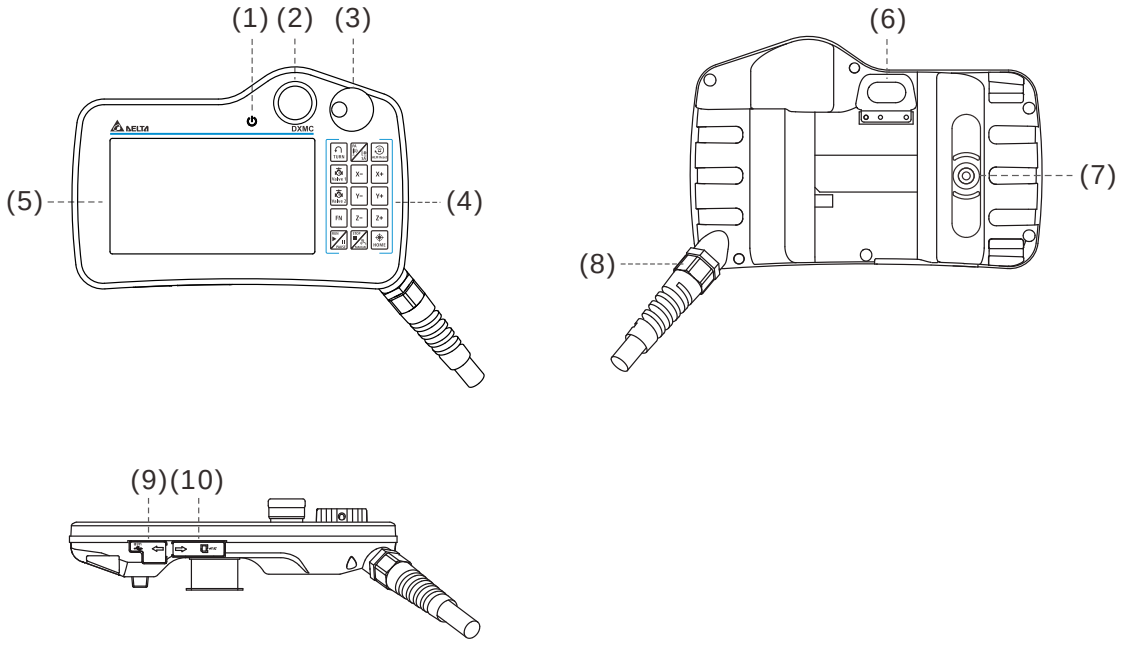
尾端出线型式



线材颜色	功能
RJ-45 (蓝)	DMCNET
RJ-45 (黑)	Ethernet
白 / 紫	紧急停止按钮 1 – NC (B 接点)
白 / 绿	紧急停止按钮 1 – NC (B 接点)
蓝	紧急停止按钮 2 – NC (B 接点)
白 / 蓝	紧急停止按钮 2 – NC (B 接点)
白 / 黑	保留
白 / 黑	保留
红	DC 24V
黑	0V
白	FGND (电源接地线)

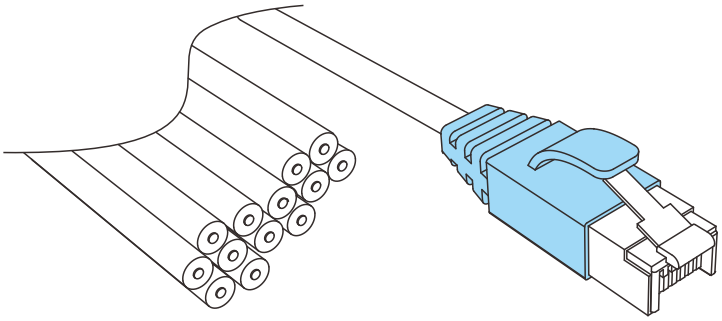
3

■ DXMC-1H08CD-7Sx 机种



编号	名称	说明
(1)	电源指示灯	系统电源指示灯。
(2)	紧急停止按钮	紧急停止按钮。
(3)	手摇轮	提供手摇轮功能。
(4)	辅助键盘	提供 15 个辅助键，用户可自行定义功能。
(5)	操作 / 显示区域	触控面板可以用来操作与显示。
(6)	挂勾	可使用挂勾固定，避免控制器摔落损坏。
(7)	三段式操作按钮	提供三段式开关，用户可自行定义功能。
(8)	总线	包含 DMCNET 运动总线、紧急停止等连接线。
(9)	USB Slave	透过 USB Cable 与 PC 连接。
(10)	SD Card 卡槽	连接 SD Card。

尾端出线型式



线材颜色	功能
RJ-45 (蓝)	DMCNET
黄	RS-485 (+)
白 / 黄	RS-485 (-)
白 / 棕	通讯接地
白 / 红	通讯接地
白 / 紫	紧急停止按钮 1 – NC (B 接点)
白 / 绿	紧急停止按钮 1 – NC (B 接点)
蓝	紧急停止按钮 2 – NC (B 接点)
白 / 蓝	紧急停止按钮 2 – NC (B 接点)
白 / 黑	保留
白 / 黑	保留
红	DC 24V
黑	0V
白	FGND (电源接地线)

3

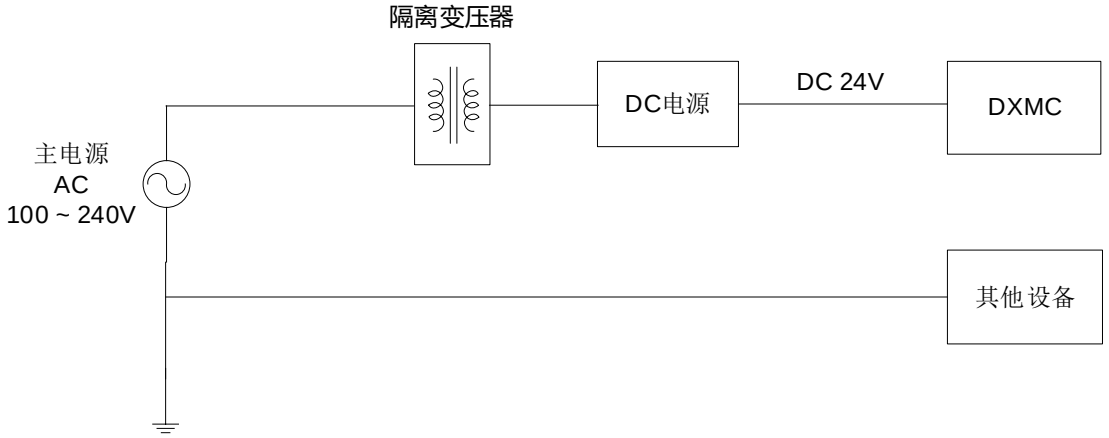
3.1.2 动作状态指示灯（DXMC-S 机种）

使用者可以透过 DXMC 运动控制器正面上方的动作指示灯来确认系统的动作状态。
动作状态指示灯的说明如下表所示。

指示灯名称	显示颜色	状态	说明
RDY	绿色	常亮	系统初始化完成。
		熄灭	正在上电初始化阶段或系统处于失效模式（Fail mode)。
		闪烁	保留
ERR	红色	常亮	系统严重错误发生（Error)。
		熄灭	系统正常动作。
		闪烁	系统非严重错误发生（Warning)。
RUN	绿色	常亮	系统处于操作模式（Operation mode)。
		熄灭	系统处于其他模式。
		闪烁	系统处于诊断模式（Diagnosis mode)。
PWR	绿色	常亮	系统电源正常供电。
		熄灭	系统电源无供电。
		闪烁	保留

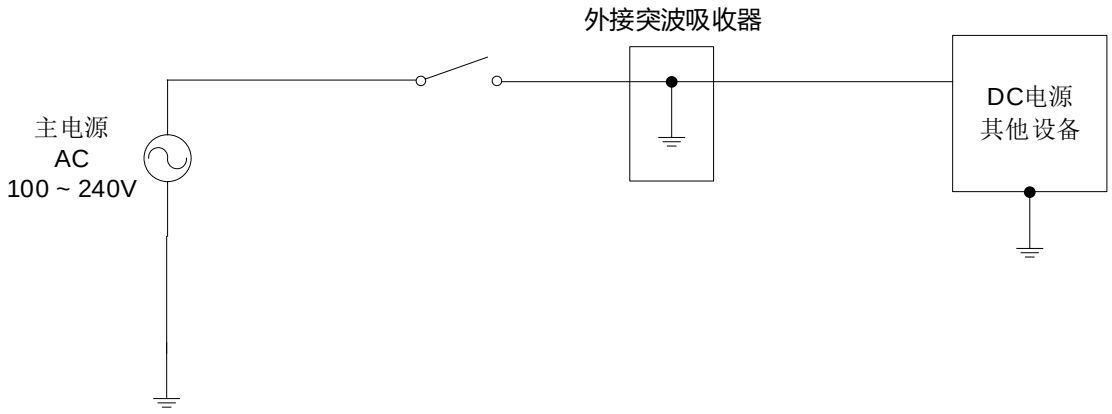
3.1.3 电源配线

DXMC 运动控制器主机电源由独立 DC 电源提供，请将 DC 电源的电源线和其他设备的电源分开。如下图所示，可以根据现场噪声情况考虑是否增加隔离变压器。



下列为接线时的注意事项：

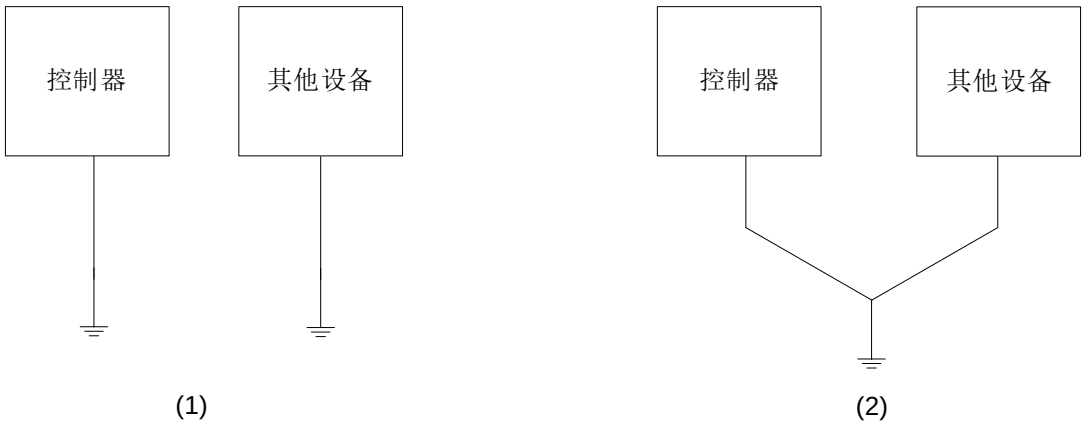
1. 建议交流 AC 电源和直流 DC 电源的电缆线与主回路、I/O 信号线路、其他设备的电缆线路隔开。建议将这些线路尽可能间隔 100 mm 以上。
2. DC 电源容许变化范围为 DC +24V (-10% ~ +15%)。
3. 为了防止雷击引起的突波，建议安装突波吸收器。



3.1.4 接地方式

3

多种设备同时使用时，建议使用专用接地。若无法使用专用接地的情况下，请使用并联共同接地，请参考下图 (2)。



(1) 专用接地；(2) 并联共同接地

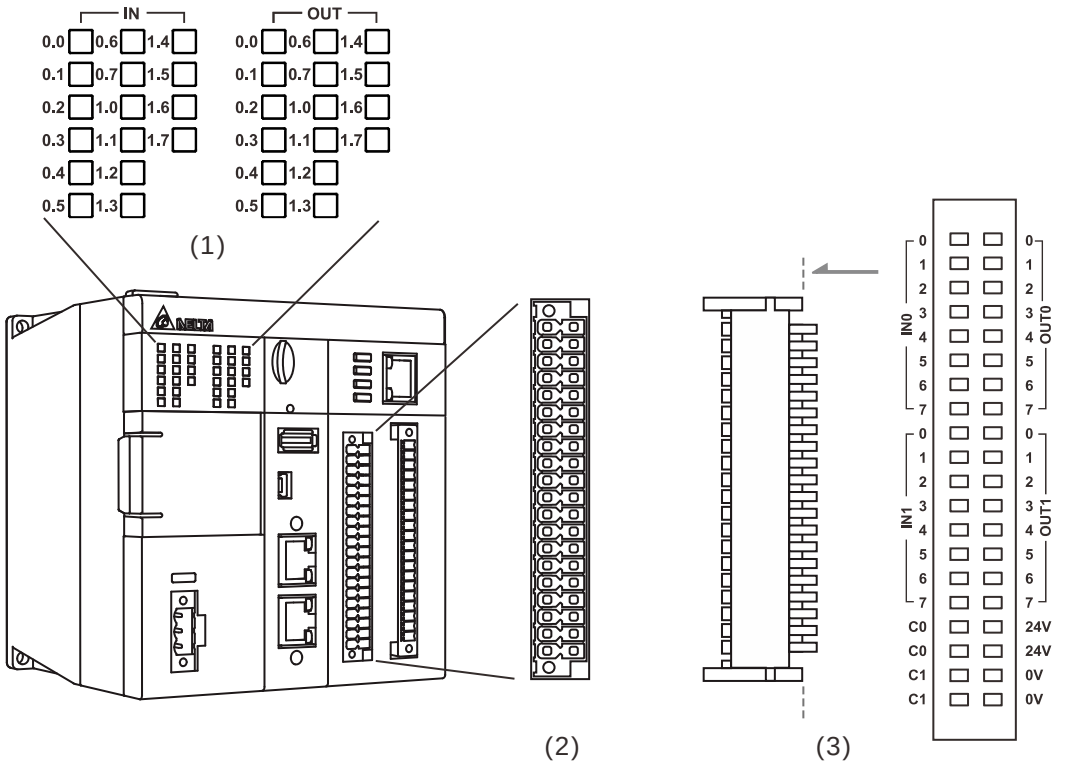
下列为接线时的注意事项：

1. 接地时，请勿使用串联共同接地。
2. DC 电源容许变化范围为 DC +24V (-10% ~ +15%)。
3. 为了防止雷击引起的突波，建议安装突波吸收器。

3.2 I/O 端子端口 (DXMC-S 机种)

3.2.1 I/O 端子说明

为了提供更有弹性的功能控制方式，DXMC-S 提供可任意规划功能的 16 组输出及 16 组输入。使用者可以透过 DXMC-S 左上方的 I/O 端子状态指示灯来确认目前 I/O 状态，亦可透过 DXMC 控制器专用的软件 DMARS 中的**本地 IO 配置**页面来选择需要的功能，详细设定说明请参考 DMARS 软件操作手册。



(1) IO 端子状态指示灯；(2) IO 端子座图；(3) IO 配线定义图

配线定义：

Pin No.	功能说明	Pin No.	功能说明
IN0.0	数字输入	IN1.1	数字输入
IN0.1	数字输入	IN1.2	数字输入
IN0.2	数字输入	IN1.3	数字输入
IN0.3	数字输入	IN1.4	数字输入
IN0.4	数字输入	IN1.5	数字输入
IN0.5	数字输入	IN1.6	数字输入
IN0.6	数字输入	IN1.7	数字输入
IN0.7	数字输入	C0	IN0.0 ~ IN0.7 公共端
IN1.0	数字输入	C0	IN0.0 ~ IN0.7 公共端

Pin No.	功能说明	Pin No.	功能说明
C1	IN1.0 ~ IN1.7 公共端	OUT1.1	数字输出
C1	IN1.0 ~ IN1.7 公共端	OUT1.2	数字输出
OUT0.0	数字输出	OUT1.3	数字输出
OUT0.1	数字输出	OUT1.4	数字输出
OUT0.2	数字输出	OUT1.5	数字输出
OUT0.3	数字输出	OUT1.6	数字输出
OUT0.4	数字输出	OUT1.7	数字输出
OUT0.5	数字输出	24V	I/O 电源 24V
OUT0.6	数字输出	24V	I/O 电源 24V
OUT0.7	数字输出	0V	I/O 电源 0V
OUT1.0	数字输出	0V	I/O 电源 0V

输入点电器规格:

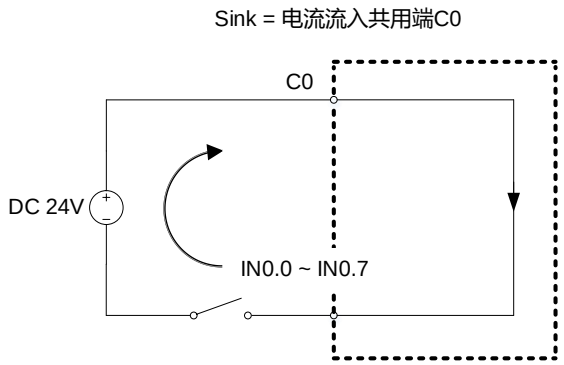
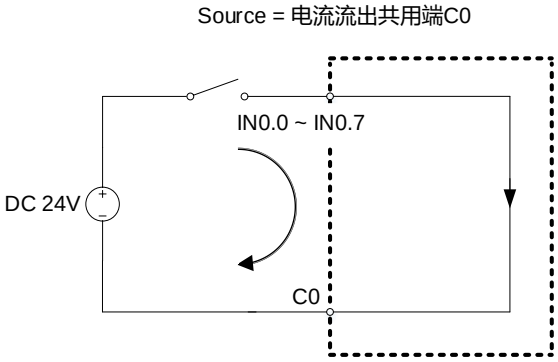
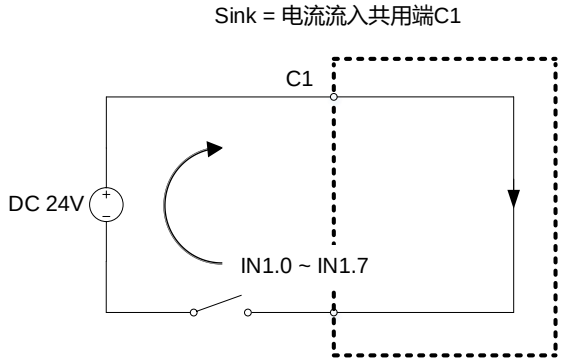
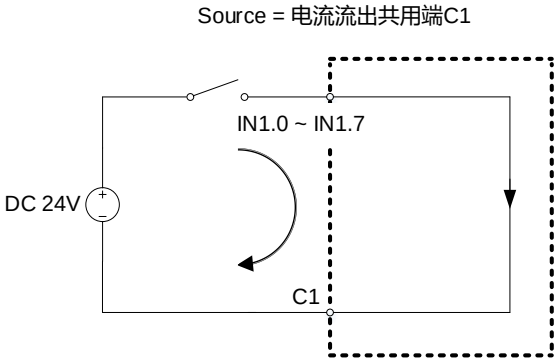
项目	规格
输入通道数	16
输入接线端子	IN 0.0 ~ IN 1.7
输入点公共端	C0 ~ C1
输入点型式	SINK (NPN) / SOURCE (PNP)
输入电流	24 V_{DC}，5 mA
动作准位	OFF -> ON，> 16.5 V_{DC} ON -> OFF，< 5 V_{DC}
工作频率	25 kHz

输出点电器规格:

项目	规格
输出通道数	16
输出接线端子	OUT 0.0 ~ OUT 1.7
输出点公共端	0 V_{DC}
输出点供电电压	24 V_{DC} (-10% ~ +15%)
输出电流	200 mA (环境温度 40°C)
反应时间	OFF -> ON：18 μs ON -> OFF：18 μs
最大切换频率	25 kHz

3.2.2 数字输入配线

数字输入的两种接法：漏型模式（Sink）及源型模式（Source），其定义如下。

NPN 晶体，共射极（E）模式（Sink 模式） IN0.0 ~ IN0.7 接线，外部电源	PNP 晶体，共射极（E）模式（Source 模式） IN0.0 ~ IN0.7 接线，外部电源
	
NPN 晶体，共射极（E）模式（Sink 模式） IN1.0 ~ IN1.7 接线，外部电源	PNP 晶体，共射极（E）模式（Source 模式） IN1.0 ~ IN1.7 接线，外部电源
	

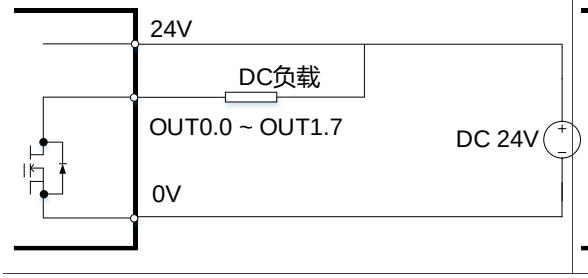
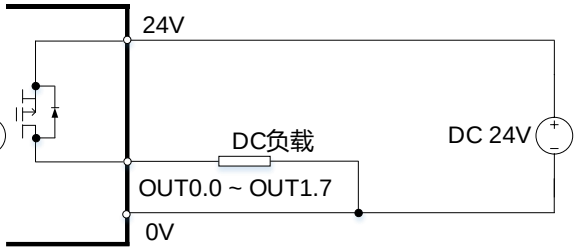
注：

1. 容许电流：200 mA 以下。
2. DC 电源容许变化范围为 DC +24V (-10% ~ +15%)。

3.2.3 数字输出配线图

3

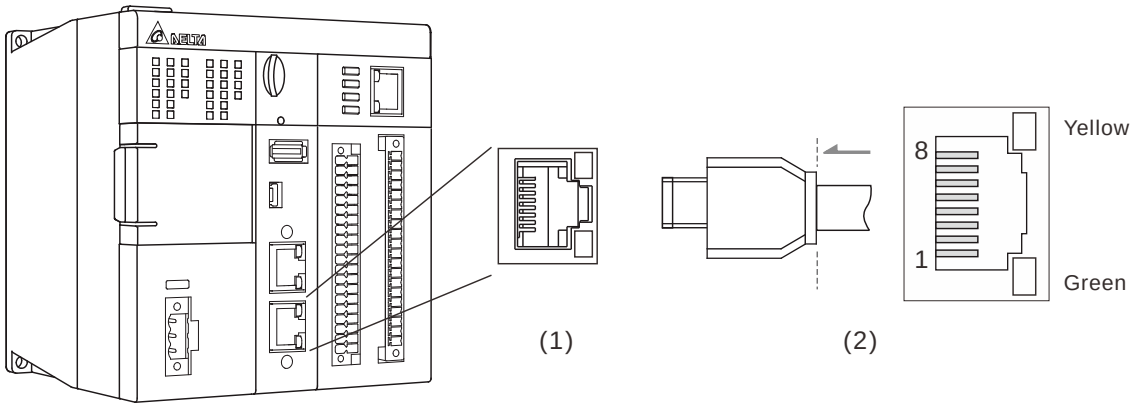
晶体类型输出请接欲控制之装置或负载即可。请注意勿接入 AC 电源以及 AC 负载且超出使用规格。

N 通道输出 (漏型 - Sink type)	P 通道输出 (源型 - Source type)
	

3.3 EtherCAT 通信端口

3.3.1 DXMC-S 机种 EtherCAT 通信端口

DXMC-S 支持 EtherCAT 之通讯功能，可以使用基于 EtherCAT 的通讯以执行运动控制的功能。EtherCAT 通讯功能可以存取与变更 EtherCAT 运动总在线的各种装置，可以控制外挂服务器或是扩充模块。



(2) EtherCAT 端子座图；(2) EtherCAT 配线定义图

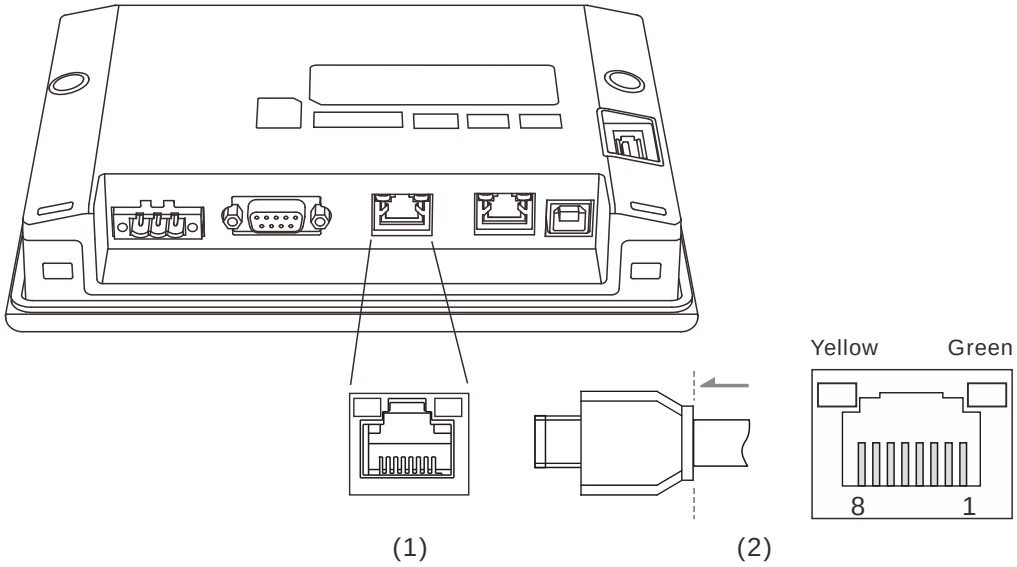
配线定义：

Pin No.	信号名称	功能说明
1	TXP	EtherCAT TX+
2	TXN	EtherCAT TX-
3	RXP	EtherCAT RX+
4	-	保留
5	-	保留
6	RXN	EtherCAT RX-
7	-	保留
8	-	保留

3.3.2 DXMC-P 机种 EtherCAT 通信端口

3

DXMC-P 支持 EtherCAT 之通讯功能，可以使用基于 EtherCAT 的通讯以执行运动控制的功能。EtherCAT 通讯功能可以存取与变更 EtherCAT 运动总在线的各种装置，可以控制外挂服务器或是扩充模块。



(1) EtherCAT 端子座图；(2) EtherCAT 配线定义图

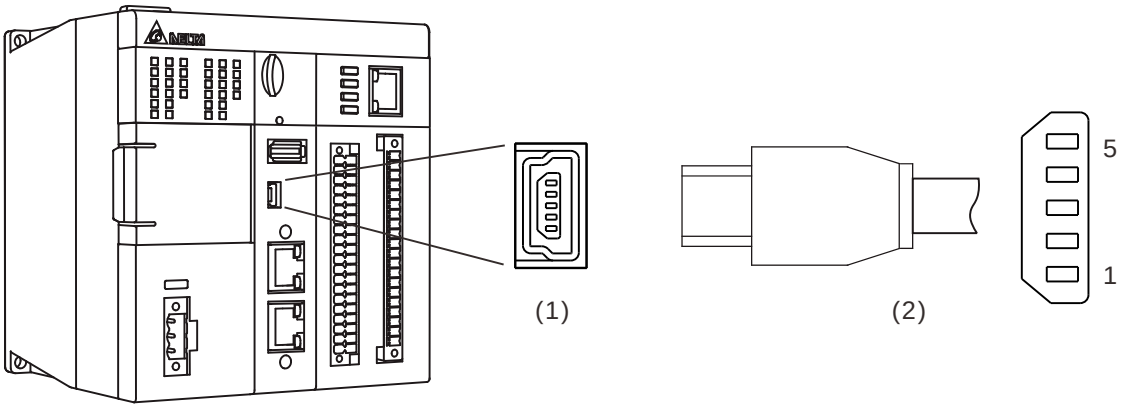
配线定义：

Pin No.	信号名称	功能说明
1	TXP	EtherCAT TX+
2	TXN	EtherCAT TX-
3	RXP	EtherCAT RX+
4	-	保留
5	-	保留
6	RXN	EtherCAT RX-
7	-	保留
8	-	保留

3.4 USB 串行通讯端口

3.4.1 DXMC-S 机种 USB 串行通讯端口

USB SLAVE：连接 PC 软件 DMARS 的串行通讯端口。PC 可透过 Mini USB 电缆来操作 DXMC 主机。



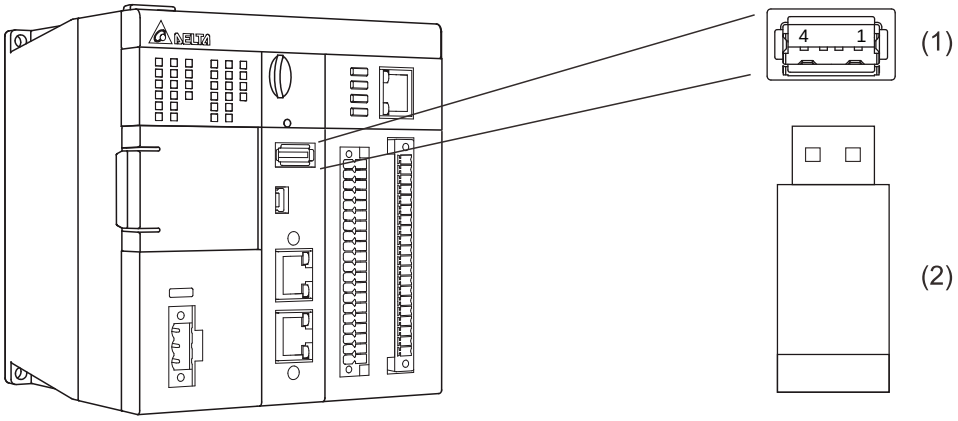
(1) USB SLAVE 端子座图；(2) USB SLAVE 配线定义图

配线定义：

Pin No.	信号名称	功能说明
1	V bus	直流+5V (外部提供)
2	D-	Data-
3	D+	Data+
4	GND	接地
5	GND	接地

3

USB HOST：连接 U 盘专用插槽。



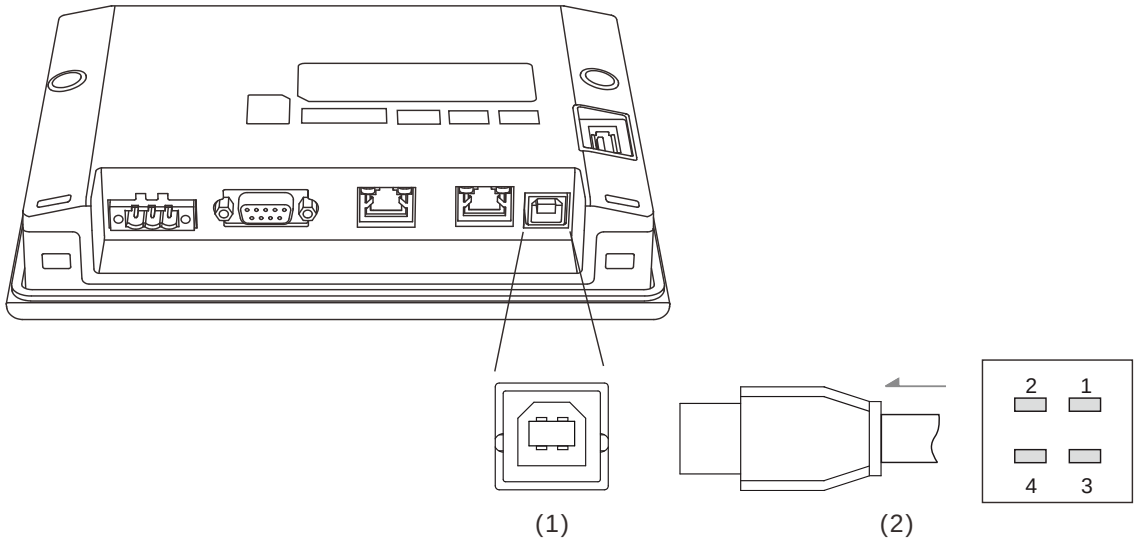
(1) USB HOST 端子座图；(2) USB 接口图

配线定义：

Pin No.	信号名称	功能说明
1	V bus	直流+5V (外部提供)
2	D-	Data-
3	D+	Data+
4	GND	接地

3.4.2 DXMC-P 机种 USB 串行通讯端口

USB SLAVE：连接 PC 软件 DIAScreen 的串行通讯端口。PC 可透过 USB 电缆来传输人机画面。



(1) USB SLAVE 端子座图；(2) USB SLAVE 配线定义图

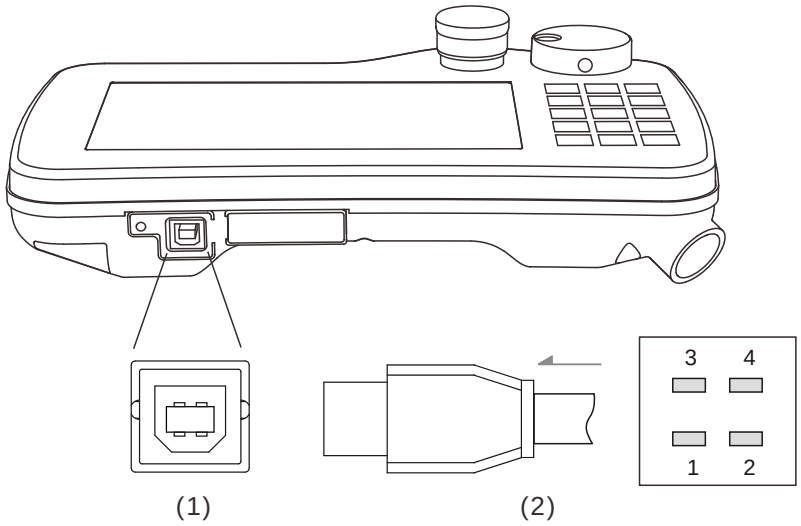
配线定义：

Pin No.	信号名称	功能说明
1	V bus	直流+5V (外部提供)
2	D-	Data-
3	D+	Data+
4	GND	接地

3.4.3 DXMC-H 机种 USB 串行通讯端口

3

USB SLAVE：连接 PC 软件 DIAScreen 的串行通讯端口。PC 可透过 USB 电缆来传输人机画面。



(1) USB SLAVE 端子座图；(2) USB SLAVE 配线定义图

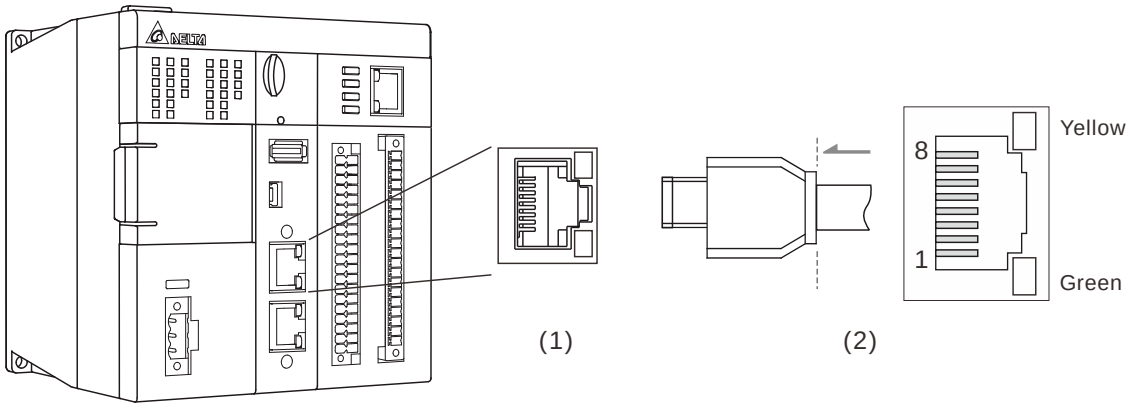
配线定义：

Pin No.	信号名称	功能说明
1	V bus	直流+5V (外部提供)
2	D-	Data-
3	D+	Data+
4	GND	接地

3.5 Ethernet 通信端口

3.5.1 DXMC-S 机种 Ethernet 通信端口

DXMC-S 主机支持 Ethernet 之通讯功能，PC 软件可以透过 Ethernet 操作 DXMC-S，同时透过 Ethernet 通讯功能可以存取与变更 DXMC-S 系统内的参数。



(1) Ethernet 端子座图；(2) Ethernet 配线定义图

配线定义：

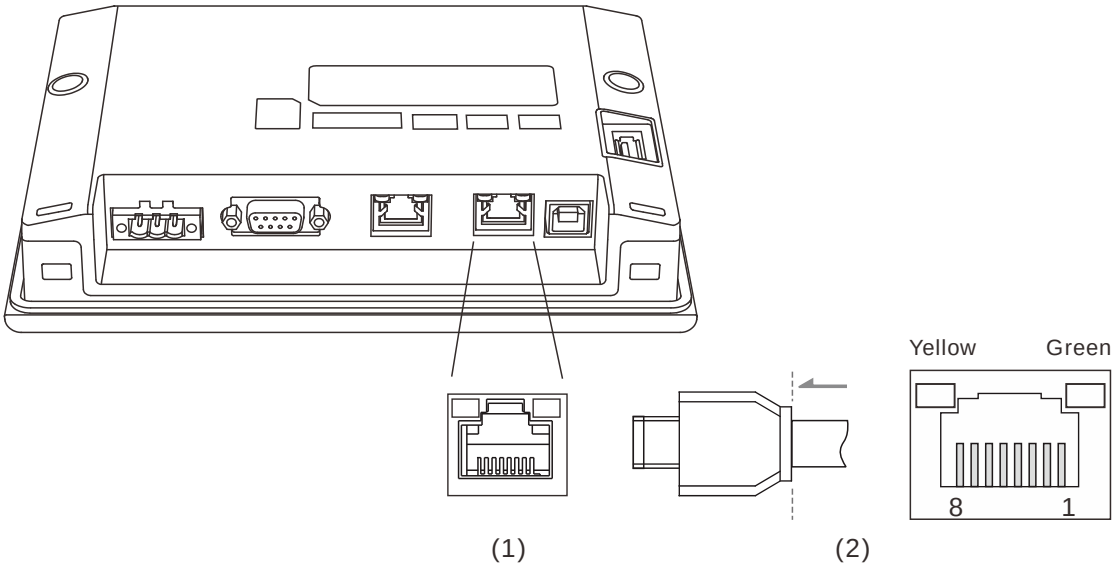
Pin No.	信号名称	功能说明
1	TXP	Ethernet TX+
2	TXN	Ethernet TX-
3	RXP	Ethernet RX+
4	-	保留
5	-	保留
6	RXN	Ethernet RX-
7	-	保留
8	-	保留

注：Ethernet 的默认 IP 地址为 192.168.1.10。

3.5.2 DXMC-P 机种 Ethernet 通信端口

3

DXMC-P 主机支持 Ethernet 之通讯功能，PC 软件可以透过 Ethernet 操作 DXMC-P，同时透过 Ethernet 通讯功能可以存取与变更 DXMC-P 系统内的参数。



(1) Ethernet 端子座图；(2) Ethernet 配线定义图

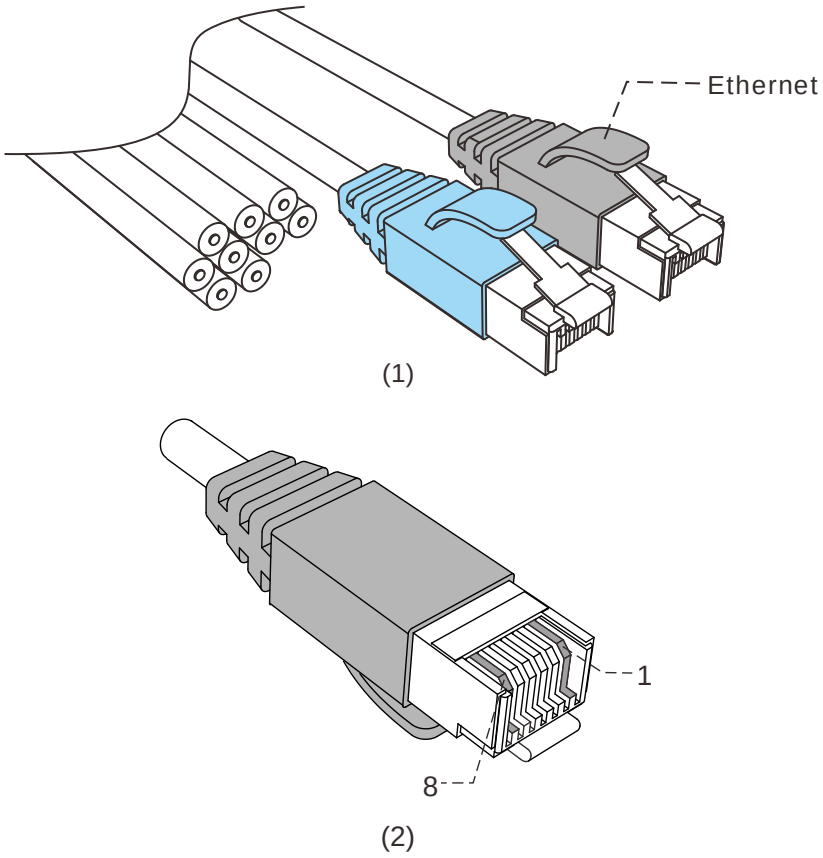
配线定义：

Pin No.	信号名称	功能说明
1	TXP	Ethernet TX+
2	TXN	Ethernet TX-
3	RXP	Ethernet RX+
4	-	保留
5	-	保留
6	RXN	Ethernet RX-
7	-	保留
8	-	保留

注：Ethernet 的默认 IP 地址为 192.168.1.10。

3.5.3 DXMC-H 机种 Ethernet 通信端口

部分 DXMC-H 主机支持 Ethernet 之通讯功能，PC 软件可以透过 Ethernet 操作 DXMC-H，同时透过 Ethernet 通讯功能可以存取与变更 DXMC 系统内的参数。



(1) Ethernet 尾端出线图；(2) Ethernet 配线定义图

Ethernet 配线定义：

Pin No.	信号名称	功能说明
1	TXP	Ethernet TX+
2	TXN	Ethernet TX-
3	RXP	Ethernet RX+
4	-	保留
5	-	保留
6	RXN	Ethernet RX-
7	-	保留
8	-	保留

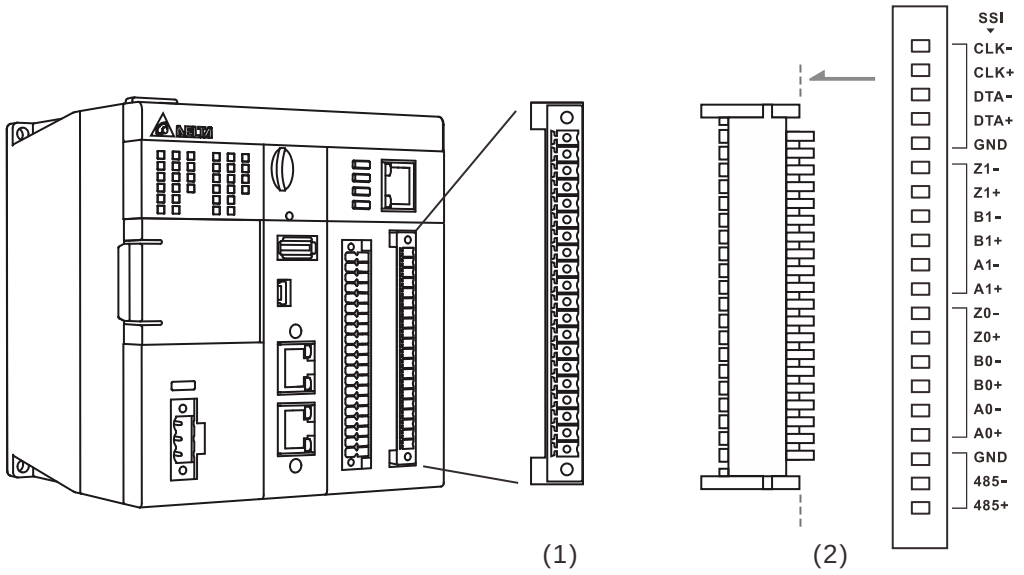
注：Ethernet 的默认 IP 地址为 192.168.1.10。

3.6 外部通信端口 (DXMC-S 机种)

3

3.6.1 外部通讯端子说明

DXMC-S 主机支持多种外部通讯格式。外部通讯端子中包含 1 组 SSI 绝对型编码器输入接口、2 组增量型编码器输入接口及 1 组 RS-485 串行通讯功能。详细设定说明请参考 DMARS 软件操作手册。



(1) 外部通讯连接器端子座图；(2) 外部通讯连接器配线定义图

配线定义：

种类	信号名称	功能说明
SSI 输入	CLK-	Clock「-」端
	CLK+	Clock「+」端
	DTA-	Data「-」端
	DTA+	Data「+」端
	GND	接地
增量型编码器输入 1	Z1-	Z 相位「-」端
	Z1+	Z 相位「+」端
	B1-	B 相位「-」端
	B1+	B 相位「+」端
	A1-	A 相位「-」端
	A1+	A 相位「+」端

种类	信号名称	功能说明
增量型编码器输入 0	Z0-	Z 相位「-」端
	Z0+	Z 相位「+」端
	B0-	B 相位「-」端
	B0+	B 相位「+」端
	A0-	A 相位「-」端
	A0+	A 相位「+」端
RS-485	GND	讯号接地
	485-	数据传送差动「-」端
	485+	数据传送差动「+」端

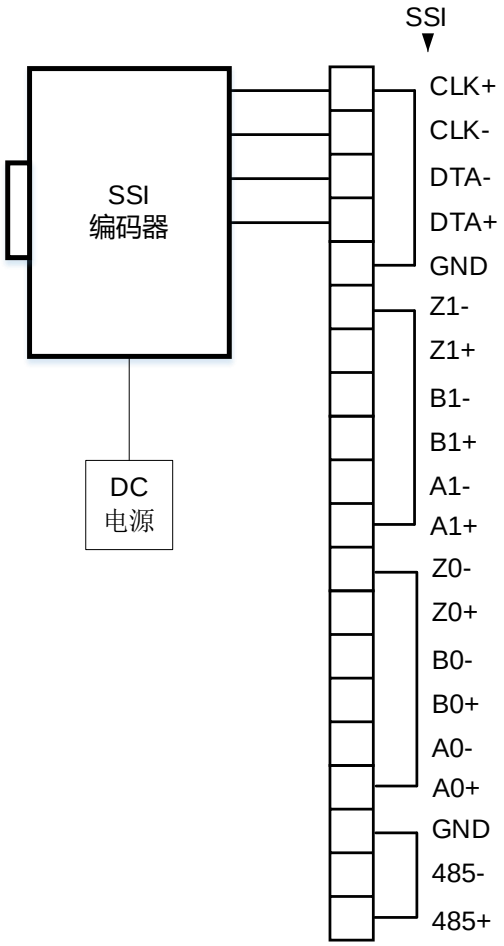
编码器输入电器规格:

项目	规格
输入通道数	2 组
输入类型	差动讯号
输入电压	5 V_{DC} ~ 12 V_{DC}
计数方式	A / B 脉冲
最大工作频率	10 MHz

3.6.2 外部通讯端子接线图

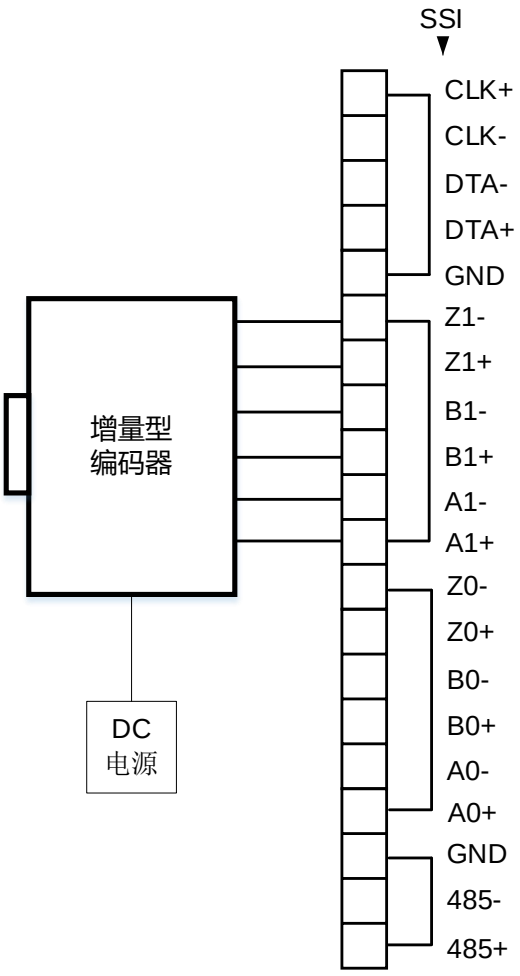
3

■ SSI 输入接线图



注：请根据所连接的 SSI 编码器实际电源规格，对该编码器进行独立供电。

■ 增量型编码器输入接线图



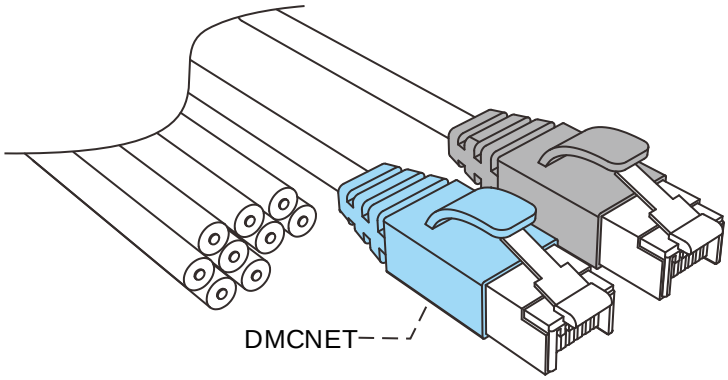
注:

1. 增量型编码器输入 0 和增量型编码器输入 1 接线方式相同。
2. 请根据所连接的增量型编码器实际电源规格，对该编码器进行独立供电。

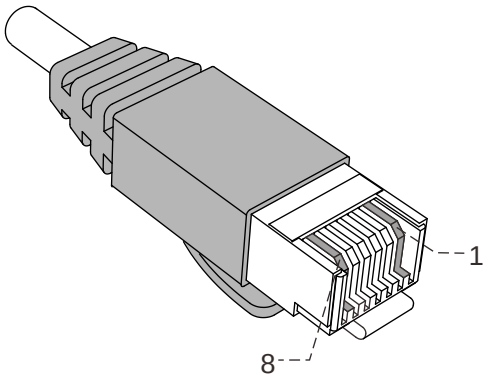
3.7 DMCNET 通信端口 (DXMC-H 机种)

3

DXMC-H 支持 DMCNET 之通讯功能，可以使用基于 DMCNET 的通讯以执行运动控制的功能。DMCNET 通讯功能可以存取与变更 DMCNET 运动总在线的各种装置，可以控制外挂服务器或是扩充模块。



(1)



(2)

(1) DMCNET 尾端出线图；(2) DMCNET 配线定义图

DMCNET 配线定义：

Pin No.	信号名称	功能说明
1	DMC_A1	DMCNET 1+
2	DMC_B1	DMCNET 1-
3	DMC_A2	DMCNET 2+
4	-	保留
5	-	保留
6	DMC_B2	DMCNET 2-
7	-	保留
8	-	保留

控制器 PLC 执行原理 4

本章节主要介绍控制器 PLC 执行原理，包含任务类型和任务之间的优先权及执行顺序。

4.1 任务 (Task) ····· 4-2

4.1.1 任务类型 ····· 4-2

4.1.2 IO 执行原理 ····· 4-4

4.1.3 任务优先权 ····· 4-6

4.1 任务（Task）

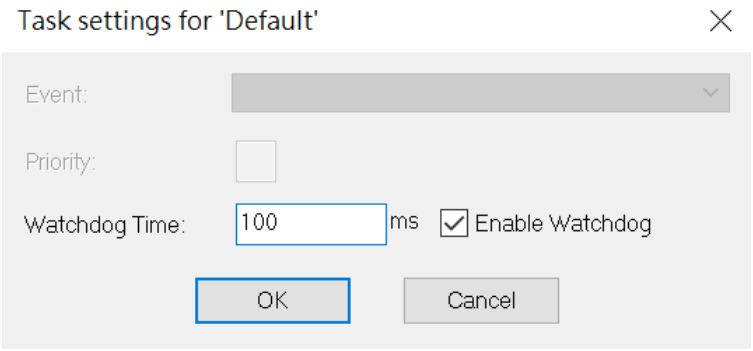
4

在进行 PLC 程序编辑前，需先了解各种任务的类型和优先级。任务（Task）用于管理加载程序组织单位位（POU）的类型和运行周期。使用 POU 时，使用者可自行依照需求设定其任务类型，PLC 将会根据任务类型决定执行任务的周期及优先级。MULTIPROG 定义了三种任务类型，分别是 Default、Cyclic 和 Event，这些任务的优先权由低到高为 Default、Cyclic 和 Event。另外，PLC 中的 IO 装置点位更新周期是可以设置要绑定在哪个任务上，其更新周期跟优先级是根据绑定任务而定。

4.1.1 任务类型

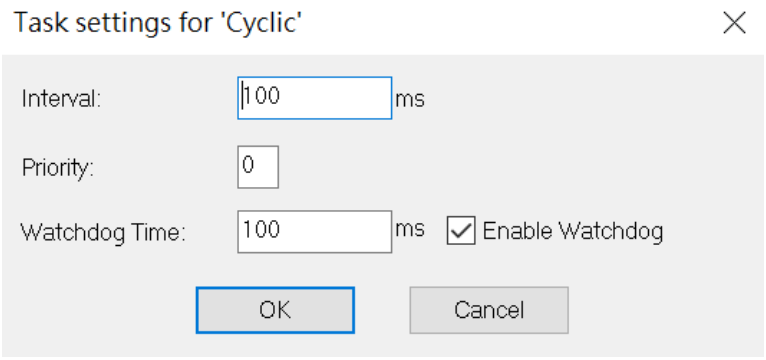
Default Task

MULTIPROG 内仅提供一组 Default Task，与 PLC 中运行的程序相同，当控制器上电后便一直重复循环运行，可以被其他任务插断。下图为 Default Task 任务的参数设定窗口，Default Task 无法设定优先权及执行周期时间。若勾选 **Enable Watchdog**，监视定时器会根据设定的 **Watchdog Time** 进行监视。当 Default Task 没办法在设定时间内完成任务时，监视定时器将认为任务运行异常，此时 PLC 会停止运行并产生一个异常报警。



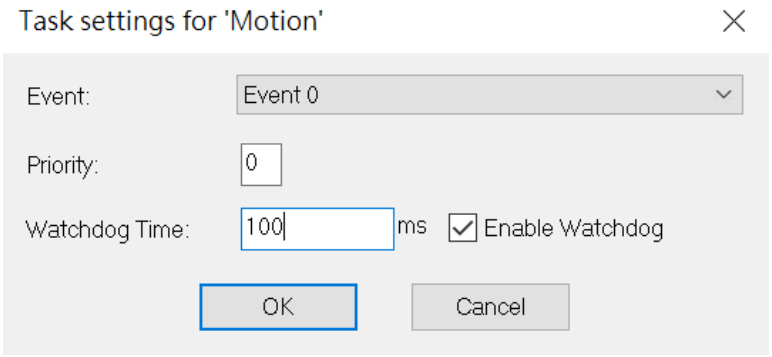
Cyclic Task

此类型任务为周期性的任务，且具有一个重要参数「运行周期」(Interval)，每隔这个间隔时间，任务即被呼叫运行。设定窗口中需要设置的参数有 3 个：**Interval**、**Priority** 和 **Watchdog Time**。**Interval** 为任务的运行周期；**Priority** 为当任务同样都是 Cyclic Task 时，彼此之间的优先权排序，设定范围为 0 ~ 15，其中数值越小代表优先级越大；**Watchdog Time** 用于监视时间内有无执行完任务。Cyclic Task 为周期性任务，可设置运行周期，因此用户可以规划任务需在几个周期内完成，若没完成则跳报警。



Event Task

此任务类型是根据事件而触发的，在控制器运行时该任务处于非运行状态，一旦对应的事件触发，该任务就会开始运行。该类型的任务通常用于处理较紧急的事件。下图为 Event Task 的参数设定窗口，**Event** 主要是设定该任务所对应的事件编号，事件编号可选择 Event 0 ~ Event 5。DXMC 为多核心运动控制器，Event 0 与其他事件编号不同，Event 0 预设为运动核心（Motion）与 EtherCAT 总线是运行在同一个 CPU 下；Event 1 ~ Event 5 的事件及其他任务（Cyclic、Default）则是运行在另外一个 CPU。控制器再依照任务类型或优先权执行任务。



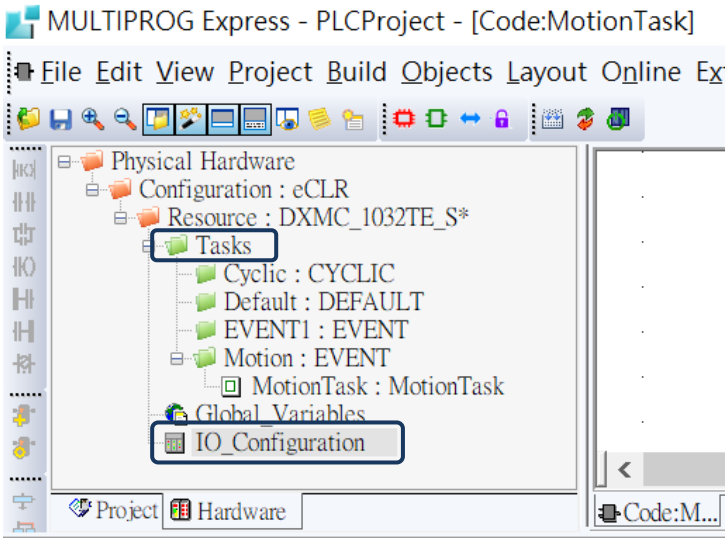
4

4.1.2 IO 执行原理

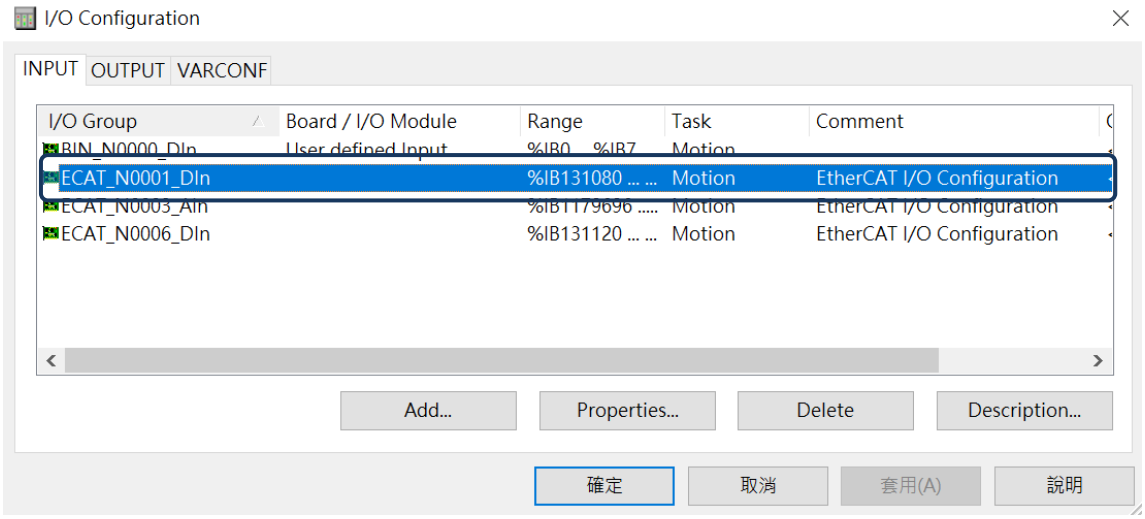
在 MULTIPROG 中，IO 装置的输出/输入点位更新是根据设置在哪个任务上，并且跟随着该任务类型和优先级更新点位。章节 4.1.3 范例中有 IO 点位更新周期及不同任务种类之间执行顺序的关系图。控制器使用的 IO 装置可能不只一个，每个 IO 都能设置不同的任务类型，用户可依照实时性去分配 IO 要设置在哪个任务上。

■ **IO 设置**

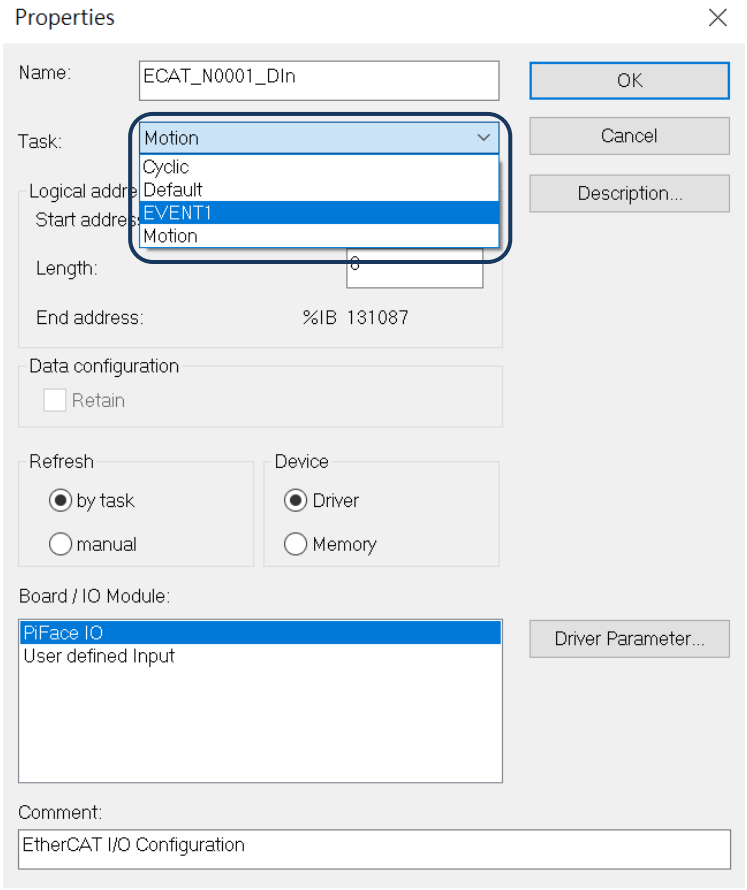
在 IO 设置任务之前，需先在 **Tasks** 节点中建立任务，进入 MULTIPROG 的 **Hardware** 页签中可新增任务（任务新增说明请参考章节 6.2.4），接着点击 **IO_Configuration** 进行设定。



点击 **IO_Configuration** 后将弹出如下窗口，可选择要使用哪个装置的 IO 进行设置，对欲设定装置点击左键两下。



点击两下欲设定装置后将跳出如下 Properties 窗口。**Task** 表示设置的任务类型，用户可以透过下拉式选单来改变任务类型。

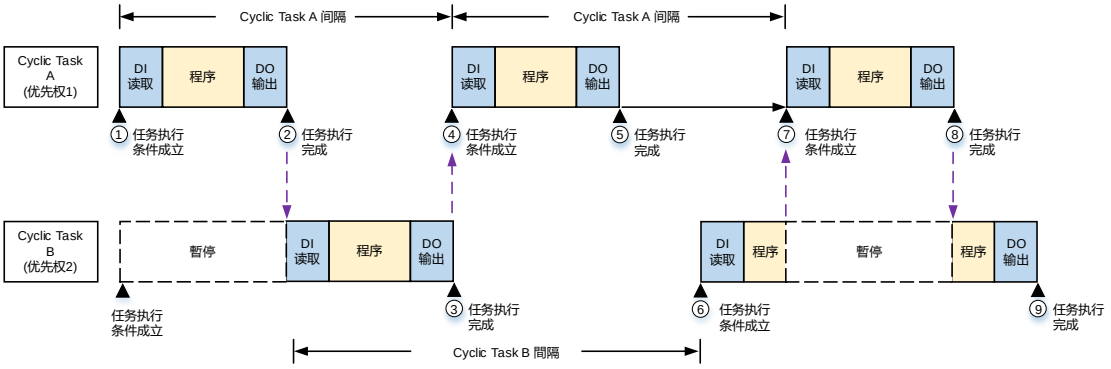


4.1.3 任务优先权

4

■ 两个相同任务同时执行 (优先权不一样)

在使用控制器执行任务时，有时会执行多个相同类型的任务，但即便是相同的任务类型，也会依据不同的优先序设定而有不同的执行顺序。Cyclic Task 可设定的优先权范围为 0 ~ 15，其中优先序数值越小表示优先权越高。下图为两个 Cyclic Task 设置不同的优先权，Cyclic Task A 优先权设为 1、Cyclic Task B 优先权设为 2。假设两个 Cyclic Task 各自有设置 IO 装置，并执行完整一次任务情况。



① Cyclic Task A 和 Cyclic Task B 执行条件同时成立，虽然任务类型相同，但由于 A 的优先权比 B 高，所以先执行 Cyclic Task A 的任务。

② Cyclic Task A 任务执行完成后，Cyclic Task B 开始执行任务。

③ Cyclic Task B 任务执行完成且 Cyclic Task A 有任务需求。

④ Cyclic Task A 开始执行任务。

⑤ Cyclic Task A 执行完任务，此时没有任务请求，所以没有执行任务。

⑥ Cyclic Task B 执行条件成立，开始执行任务。

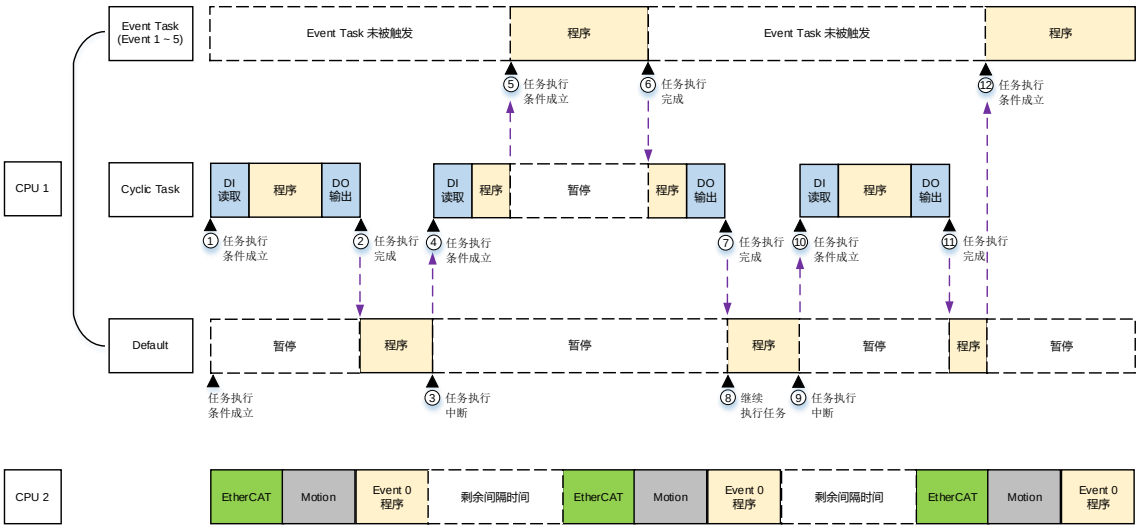
⑦ Cyclic Task B 执行任务过程时，优先权较高的 Cyclic Task A 会中断 Cyclic Task B 的任务，控制器开始执行 Cyclic Task A 的任务。Cyclic Task B 在这次任务时，有执行到 DI 输入，DI 输入讯号的状态会更新，接着执行部分程序后就被中断。此次任务并没有执行到 DO 输出，所以还不会输出讯号。

⑧ Cyclic Task A 执行任务完成，控制器继续执行优先权较低的 Cyclic Task B 未执行的部分。

⑨ Cyclic Task B 任务执行完毕，DO 输出的部分有执行到，此时才有输出讯号。

■ 三个任务混合执行（Event、Cyclic、Default)

DXMC 为多核心控制器，在 DXMC 资源规划中，会使用到两个 CPU，也就是 CPU1 和 CPU2。Event Task、Cyclic Task 和 Default Task 是分配在 CPU1 中。请注意 Event 1 ~ Event 5 任务都会在 CPU1 上执行；而 Event 0 则是独立的，与 EtherCAT 以及运动核心 (Motion) 在 CPU2 上执行任务。在 CPU1 上优先权由低到高分别为 Default、Cyclic、Event。控制器将执行优先权较高的任务，优先权高的任务可以中断优先权低的任务。在需要高及时反应要求的程序，建议设置为高优先权的任务。



① Cyclic Task 和 Default Task 任务条件同时成立，Default Task 为一上电就开始执行的任务，此时刚好也轮到 Cyclic Task 执行任务，因为 Cyclic Task 优先权较 Default Task 高，所以控制器优先执行 Cyclic Task 的任务。

② Cyclic Task 的任务执行完成后，Default Task 开始执行任务。

③ Default Task 执行任务过程中，Cyclic Task 任务触发条件成立 (设置的间隔时间)，Cyclic Task 会中断 Default Task。

④ Cyclic Task 开始执行任务。

⑤ Cyclic Task 执行任务过程中，Event Task 被触发并执行任务，Event Task 为优先序最高的任务，因此 Event Task 中断 Cyclic Task 任务，控制器开始执行 Event Task。Cyclic Task 在这次任务时，有执行到 DI 输入，DI 输入讯号的状态会更新，接着执行部分程序后就被打断，此次任务并没有执行到 DO 输出，所以还不会输出讯号。

⑥ Event Task 执行任务完成，控制器继续执行优先权较低的 Cyclic Task 未执行的部分。

⑦ Cyclic Task 任务执行完毕，DO 输出执行完毕，此时才会输出讯号。

⑧ 此时没有其他优先权更高的任务，控制器则继续执行 Default Task 的任务。

⑨ Default Task 执行任务过程中，Cyclic Task 任务触发条件成立（设置的间隔时间），Cyclic Task 会中断 Default Task，控制器便开始执行 Cyclic Task 任务。

⑩ Cyclic Task 开始执行任务。

⑪ Cyclic Task 任务执行完毕，控制器则继续执行 Default Task 任务。

⑫ Default Task 执行任务过程中，Event Task 任务触发条件成立，Event Task 为优先序最高的任务，因此 Event Task 中断 Default Task，控制器开始执行 Event Task 任务。

参数与功能 5

本章节主要介绍 DXMC 主机具备之运动控制功能以及参数设定说明。使用者可利用不同的参数完成功能设定。

5.1 参数定义 ····· 5-2
5.2 控制器参数一览表 ····· 5-3
5.3 控制器参数说明 ····· 5-7
P0-x 共同设定 ····· 5-7
P1-x 版本信息 ····· 5-10
P2-x Ethernet 设定 ····· 5-13
P4-x USB 设定 ····· 5-16
P5-x 串行通信设置 ····· 5-17
P9-x 数据安全设定 ····· 5-20
P10-x PLC 设定 ····· 5-21
P11-x SSI 通讯型编码器设定 ····· 5-23
P12-x 脉冲型编码器设定 (第一组) ····· 5-24
P13-x 脉冲型编码器设定 (第二组) ····· 5-25
5.4 轴参数一览表 ····· 5-26
5.5 轴参数说明 ····· 5-29
P0-x 基本设定 ····· 5-29
P1-x 单位设定 ····· 5-31
P2-x 速度 / 加速度设定 ····· 5-34
P3-x 警告设定 ····· 5-35
P4-x 监测设定 ····· 5-37
P5-x 极限设定 ····· 5-39
P6-x 原点复归设定 ····· 5-40
P7-x 位置计数设定 ····· 5-42
P8-x 操作设定 ····· 5-43

5.1 参数定义

5

本章节介绍 DXMC 系列的各项控制器端参数的设定说明。这些参数涵盖控制器本体的设定、运动控制的设定，以及总线装置的设定。各项参数除了可以透过软件存取与设定，也能使用功能块（SYS_ParaRead 或 SYS_ParaWrite），透过指定存取对象的类型、编号、参数主编号与参数次编号即可进行存取。不同的存取对象有不同的类型代号，以下是各种对象类型的代号。

类型	代号
轴	0h
群组	4h
机械手臂	5h
控制器本体	8h
EtherCAT Master	9h

本章节使用两种方式进行参数说明：第一种为简易内容说明，为参数一览表；第二种为详细内容说明，为参数说明表格。

参数一览表的内容定义：

参数组名						
编号		功能	初值	格式	单位	属性
主	次					
-	-	功能说明	默认值	参数单位	-	-

参数说明表格的内容定义：

编号	**主** **次**	**功能**		
初值：	默认值	格式：	数据类型与长度	
属性：	参数属性，以符号表示	单位：	参数单位	
数值范围：	最小值 ~ 最大值			
参数说明：	参数的用途与设定方法			

注：

数值范围表示方式：

a ~ b：数值范围在 a 跟 b 间连续。

a、b：数值仅有此二者有效。

参数属性符号说明：

参数属性符号	说明
🔒	参数为只读，只能读取状态值。
⛔	Servo On 时无法设定。
⏻	必须重新启动参数才有效。
🔌	断电后即还原默认值。
Ⓡ	存在于实体轴（Real）。
Ⓥ	存在于虚拟轴（Virtual）。
Ⓔ	存在于编码器轴（Encoder）。

5.2 控制器参数一览表

使用功能块（SYS_ParaRead 或 SYS_ParaWrite）时，控制器参数的对象类型（Type）代号为 8h，由于不需要设定其他站号，因此对象编号（Idx）为 0。控制器参数主编号和次编号请参考下方表格。

共同设定						
编号		功能	初值	格式	单位	属性
主	次					
0	0	Modbus Slave 站号	1	UDINT	-	-
0	1	机器程序空间量配置	4000000	UDINT	byte	🔒
0	2	电子凸轮（ECAM）最大表数配置	256	UDINT	张	🔒
0	3	PLC 程序配置大小	33554432	UDINT	byte	🔒
0	4	PLC 变量配置大小	33554432	UDINT	byte	🔒
0	5	PLC 程序剩余空间	-	UDINT	byte	🔒
0	6	PLC 变量剩余空间	-	UDINT	byte	🔒
0	7	FTP 剩余空间	-	UDINT	byte	🔒
0	8	总开机时数	-	UDINT	小时	🔒
0	9	报警显示与清除	-	DWORD	-	-
0	10	控制器操作模式	-	UDINT	-	-

5

版本信息						
编号		功能	初值	格式	单位	属性
主	次					
1	0	韧体版本	-	UDINT	-	🔒
1	1	用户 PLC 程序版本	-	UDINT	-	🔒
1	2	Motion Kernel 版本	-	UDINT	-	🔒
1	3	PLC Kernel 版本	-	UDINT	-	🔒
1	4	Common Server 版本	-	UDINT	-	🔒
1	5	Fieldbus 版本	-	UDINT	-	🔒
1	6	Lua Server 版本	-	UDINT	-	🔒
1	7	NC Parser 版本	-	UDINT	-	🔒
1	8	Common Lib 版本	-	UDINT	-	🔒
1	9	操作系统版本	-	UDINT	-	🔒

Ethernet 设定						
编号		功能	初值	格式	单位	属性
主	次					
2	0	Ethernet 网络状态	0x0	DWORD	-	🔒
2	1	Ethernet IP 地址	0xC0A8010A	DWORD	-	🔒
2	2	Ethernet 子网掩码	0xFFFFFF00	DWORD	-	🔒
2	3	Ethernet 预设网关	0xC0A8010A	DWORD	-	🔒
2	4	Ethernet 网络设定	0x0	DWORD	-	-
2	5	Ethernet IP 地址设定	0xC0A8010A	DWORD	-	-
2	6	Ethernet 子网掩码设定	0xFFFFFF00	DWORD	-	-
2	7	Ethernet 预设网关设定	0xC0A8010A	DWORD	-	-

USB 设定						
编号		功能	初值	格式	单位	属性
主	次					
4	0	USB 功能设定	0	UDINT	-	-

串行通信设置						
编号		功能	初值	格式	单位	属性
主	次					
5	1	串行通讯协议设定	0x0	DWORD	-	-
5	2	串行通讯主从设定	0x0	DWORD	-	-
5	3	串行通讯接口设定	0x1	DWORD	-	-
5	4	串行通讯波特率设定	0x3	DWORD	bps	-
5	5	串行通讯自由口格式设定	0x0	DWORD	-	-

数据安全设定						
编号		功能	初值	格式	单位	属性
主	次					
9	0	FTP 密码	-	UDINT	-	不可读取
9	1	软件权限密码	-	LINT	-	不可读取

PLC 设定						
编号		功能	初值	格式	单位	属性
主	次					
10	0	开机后 PLC 启动模式	0	DINT	-	-
10	1	自动加载 PLC 程序编号	0	UDINT	-	-
10	2	DI 启动编号	0	UDINT	-	-
10	3	DI 启动模式	0	UDINT	-	-
10	4	PLC 启动模式	0	UDINT	-	-

SSI 通讯型编码器设定						
编号		功能	初值	格式	单位	属性
主	次					
11	0	SSI 编码器频率	0	UDINT	-	🔒
11	1	SSI 编码器多圈分辨率（Turn）	0	UDINT	Turn	🔒
11	2	SSI 编码器单圈分辨率（Step）	1	UDINT	Step	🔒
11	3	SSI 编码器数据格式	0	UDINT	-	🔒

5

脉冲型编码器设定 (第一组)						
编号		功能	初值	格式	单位	属性
主	次					
12	0	脉冲型编码器滤波设定	0	UDINT	-	🔒
12	1	脉冲型编码器输入设定	0	UDINT	-	🔒

脉冲型编码器设定 (第二组)						
编号		功能	初值	格式	单位	属性
主	次					
13	0	脉冲型编码器滤波设定	0	UDINT	-	🔒
13	1	脉冲型编码器输入设定	0	UDINT	-	🔒

5.3 控制器参数说明

P0-x 共同设定

编号	主	0	Modbus Slave 站号		
	次	0			
初值:			1	格式:	UDINT
属性:			-	单位:	-
数值范围:			1 ~ 247		

参数功能:

设定 Modbus Slave 站号。

编号	主	0	机器程序空间量配置		
	次	1			
初值:			4000000	格式:	UDINT
属性:			🔒	单位:	byte
数值范围:			4000000 ~ 128000000		

参数功能:

显示机器程序空间量配置大小。

编号	主	0	电子凸轮（ECAM）最大表数配置		
	次	2			
初值:			256	格式:	UDINT
属性:			🔒	单位:	张
数值范围:			1 ~ 256		

参数功能:

设定电子凸轮（ECAM）最大表数配置数量。

编号	主	0	PLC 程序配置大小		
	次	3			
初值:			33554432	格式:	UDINT
属性:			🔒	单位:	byte
数值范围:			-		

参数功能:

显示 PLC 程序配置大小。

5

编号	主	0	PLC 变量配置大小		
	次	4			
初值:			33554432	格式:	UDINT
属性:			🔒	单位:	byte
数值范围:			-		

参数功能:

显示 PLC 变量配置大小。

编号	主	0	PLC 程序剩余空间		
	次	5			
初值:			-	格式:	UDINT
属性:			🔒	单位:	byte
数值范围:			-		

参数功能:

显示 PLC 程序剩余空间。

编号	主	0	PLC 变量剩余空间		
	次	6			
初值:			-	格式:	UDINT
属性:			🔒	单位:	byte
数值范围:			-		

参数功能:

显示 PLC 变量剩余空间。

编号	主	0	FTP 剩余空间		
	次	7			
初值:			-	格式:	UDINT
属性:			🔒	单位:	byte
数值范围:			-		

参数功能:

显示 FTP 剩余空间。

编号	主	0	总开机时数		
	次	8			
初值:			-	格式:	UDINT
属性:			🔒	单位:	小时
数值范围:			-		

参数功能:

显示控制器出厂至目前启动的总时数。

编号	主	0	报警显示与清除		
	次	9			
初值:			-	格式:	DWORD
属性:			-	单位:	-
数值范围:			-		

参数功能:

读取最后一笔的报警数据或设定 0 来清除报警。

编号	主	0	控制器操作模式		
	次	10			
初值:			-	格式:	UDINT
属性:			-	单位:	-
数值范围:			-		

参数功能:

设定控制器操作模式:

2: 编辑模式

4: 运行模式

5

P1-x 版本信息

编号	主	1	韧体版本		
	次	0			
初值:		-		格式:	UDINT
属性:		🔒		单位:	-
数值范围:		-			

参数功能:

显示韧体版本。

编号	主	1	用户 PLC 程序版本		
	次	1			
初值:		-		格式:	UDINT
属性:		🔒		单位:	-
数值范围:		-			

参数功能:

显示用户 PLC 程序版本。

编号	主	1	Motion Kernel 版本		
	次	2			
初值:		-		格式:	UDINT
属性:		🔒		单位:	-
数值范围:		-			

参数功能:

显示 Motion Kernel 版本。

编号	主	1	PLC Kernel 版本		
	次	3			
初值:		-		格式:	UDINT
属性:		🔒		单位:	-
数值范围:		-			

参数功能:

显示 PLC Kernel 版本。

编号	主	1	Common Server 版本		
	次	4			
初值:			-	格式:	UDINT
属性:			🔒	单位:	-
数值范围:			-		

参数功能:

显示 Common Server 版本。

编号	主	1	Fieldbus 版本		
	次	5			
初值:			-	格式:	UDINT
属性:			🔒	单位:	-
数值范围:			-		

参数功能:

显示 Fieldbus 版本。

编号	主	1	Lua Server 版本		
	次	6			
初值:			-	格式:	UDINT
属性:			🔒	单位:	-
数值范围:			-		

参数功能:

显示 Lua Server 版本。

编号	主	1	NC Parser 版本		
	次	7			
初值:			-	格式:	UDINT
属性:			🔒	单位:	-
数值范围:			-		

参数功能:

显示 NC Parser 版本。

编号	主	1	Common Lib 版本		
	次	8			
初值:			-	格式:	UDINT
属性:			🔒	单位:	-
数值范围:			-		

参数功能:

显示 Common Lib 版本。

编号	主	1	操作系统版本		
	次	9			
初值:			-	格式:	UDINT
属性:			🔒	单位:	-
数值范围:			-		

参数功能:

显示操作系统版本。

P2-x Ethernet 设定

编号	主	2	Ethernet 网络状态		
	次	0			
初值:			0x0	格式:	DWORD
属性:			🔒	单位:	-
数值范围:			0x00000000 ~ 0xFFFFFFFF		

参数功能:

格式: D C B A U Z Y X

X: 连接状态

0: Cable Plugged

1: Cable Unplugged

Y: IP 模式

0: Static IP

1: DHCP

Z: DHCP State

0: 设定中或未开始

1: 已取得 IP

2: 未取得 IP

A U: 保持联机的动作时间间隔

0: 不开启此功能

非 0: 间隔秒数

D C B: 系统保留

编号	主	2	Ethernet IP 地址		
	次	1			
初值:			0xC0A8010A	格式:	DWORD
属性:			🔒	单位:	-
数值范围:			0x00000000 ~ 0xFFFFFFFF		

参数功能:

显示 Ethernet IP 地址。若 IP 为 192.168.1.10，则显示 0xC0A8010A。

编号	主	2	Ethernet 子网掩码		
	次	2			
初值:			0xFFFFFF00	格式:	DWORD
属性:			🔒	单位:	-
数值范围:			0x00000000 ~ 0xFFFFFFFF		

参数功能:

显示 Ethernet 子网掩码。

5

编号	主	2	Ethernet 预设网关		
	次	3			
初值：			0xC0A8010A	格式：	DWORD
属性：			🔒	单位：	-
数值范围：			0x00000000 ~ 0xFFFFFFFF		

参数功能：

显示 Ethernet 默认网关。

编号	主	2	Ethernet 网络设定		
	次	4			
初值：			0x0	格式：	DWORD
属性：			-	单位：	-
数值范围：			0x00000000 ~ 0xFFFFFFFF		

参数功能：

格式：D C B A U Z Y X

X：开启网络功能。上升缘触发时（0→1），Ethernet 会进行初始化。

Y：IP 模式设定

0：Static IP

1：DHCP

Z：系统保留

A U：保持联机的动作时间间隔

0：不开启此功能

非 0：间隔秒数

D C B：系统保留

编号	主	2	Ethernet IP 地址设定		
	次	5			
初值：			0xC0A8010A	格式：	DWORD
属性：			-	单位：	-
数值范围：			0x00000000 ~ 0xFFFFFFFF		

参数功能：

Ethernet IP 地址设定使用 16 进制设定。默认值为 192.168.1.10 = 0xC0A8010A。

编号	主	2	Ethernet 子网掩码设定		
	次	6			
初值:			0xFFFFFF00	格式:	DWORD
属性:			-	单位:	-
数值范围:			0x00000000 ~ 0xFFFFFFFF		

参数功能:

Ethernet 子网掩码设定使用 16 进制设定。默认值为 255.255.255.0 = 0xFFFFFF00。

编号	主	2	Ethernet 预设网关设定		
	次	7			
初值:			0xC0A8010A	格式:	DWORD
属性:			-	单位:	-
数值范围:			0x00000000 ~ 0xFFFFFFFF		

参数功能:

Ethernet 预设网关设定使用 16 进制设定。默认值为 192.168.1.10 = 0xC0A8010A。

5

P4-x USB 设定

编号	主	4	USB 功能设定		
	次	0			
初值:			0	格式:	UDINT
属性:			-	单位:	-
数值范围:			-		

参数功能:

USB 功能设定，选择仿真使用 Serial 或 Ethernet。

0：USB-Serial

1：USB-Ethernet

注:

1. USB-Serial 的 Baud Rate 为 921600 bps。
2. USB-Ethernet 的 IP 为 192.168.240.1，有提供 DHCP Server，PC 端自动分配成 192.168.240.100。

P5-x 串行通信设置

编号	主	5	串行通讯协议设定		
	次	1			
初值：			0x0	格式：	DWORD
属性：			-	单位：	-
数值范围：			-		

参数功能：

设定值的定义如下：

0x00：7，N，2(MODBUS，ASCII)	0x01：7，E，1(MODBUS，ASCII)
0x02：7，O，1(MODBUS，ASCII)	0x03：8，N，2(MODBUS，ASCII)
0x04：8，E，1(MODBUS，ASCII)	0x05：8，O，1(MODBUS，ASCII)
0x06：8，N，2(MODBUS，RTU)	0x07：8，E，1(MODBUS，RTU)
0x08：8，O，1(MODBUS，RUT)	0x09：7，N，2(Freeport)
0x0A：7，E，1(Freeport)	0x0B：7，O，1 (Freeport)
0x0C：8，N，2(Freeport)	0x0D：8，E，1(Freeport)
0x0E：8，O，1(Freeport)	0x0F：7，N，1(MODBUS，ASCII)
0x10：7，N，1(Freeport)	0x11：7，E，2(MODBUS，ASCII)
0x12：7，E，2(Freeport)	0x13：7，O，2(MODBUS，ASCII)
0x14：7，E，2(Freeport)	0x15：8，N，1(MODBUS，ASCII)
0x16：8，N，1(MODBUS，RTU)	0x17：8，N，1(Freeport)
0x18：8，E，2(MODBUS，ASCII)	0x19：8，E，2(MODUBS，RTU)
0x1A：8，E，2(Freeport)	0x1B：8，O，2(MODBUS，ASCII)
0x1C：8，O，2(MODBUS，RTU)	0x1D：8，O，2(Freeport)

编号	主	5	串行通讯主从设定		
	次	2			
初值：			0x0	格式：	DWORD
属性：			-	单位：	-
数值范围：			-		

参数功能：

格式：D C B A U Z Y X

X：连接状态

0：Modbus Slave

1：Modbus Master

D C B A U Z Y：系统保留

5

编号	主	5	串行通讯接口设定		
	次	3			
初值：			0x1	格式：	DWORD
属性：			-	单位：	-
数值范围：			0 ~ 3		

参数功能：

格式：D C B A U Z Y X

X：接口设定

0：RS-232

1：RS-485

2：RS-422

3：SSI-Encoder

注：DXMC 目前仅提供 RS-485 设定。

编号	主	5	串行通讯波特率设定		
	次	4			
初值：			0x3	格式：	DWORD
属性：			-	单位：	bps
数值范围：			0 ~ 5		

参数功能：

设定值的定义如下：

0：4800	1：9600	2：19200
3：38400	4：57600	5：115200

编号	主	5	串行通讯自由口格式设定		
	次	5			
初值：			0x0	格式：	DWORD
属性：			-	单位：	-
数值范围：			0 ~ 0x1021FFFF		

参数功能：

格式：D C B A U Z Y X

YX：定义自由口-串行通讯的起始位。数值范围：0x00 ~ 0xFF

UZ：定义自由口-串行通讯的结束位。数值范围：0x00 ~ 0xFF

A：选择自由口-串行通讯是否存在起始位。

0：无

1：存在起始位，内容由 YX 决定

B：选择自由口-串行通讯是否存在结束位。

0：无

1：存在结束位，内容由 UZ 决定

2：存在结束位，内容固定为 CR+LF (换行符号)

C：系统保留

D：控制自由口-串行通讯缓冲区命令。当命令完成后，自动回复为 0。

0：无动作

1：清除接收缓冲区

5

P9-x 数据安全设定

编号	主	9	FTP 密码		
	次	0			
初值:			-	格式:	UDINT
属性:			不可读取	单位:	-
数值范围:			-		

参数功能:

设定 FTP 密码。

注：此功能暂不开放。

编号	主	9	软件权限密码		
	次	1			
初值:			-	格式:	LINT
属性:			不可读取	单位:	-
数值范围:			-		

参数功能:

设定软件权限密码。

注：此功能暂不开放。

P10-x PLC 设定

编号	主	10	开机后 PLC 启动模式		
	次	0			
初值：			0	格式：	DINT
属性：			-	单位：	-
数值范围：			-		

参数功能：

设定开机后 PLC 启动模式：

0：PLC 停止

1：DI 触发

2：PLC 立即启动

编号	主	10	自动加载 PLC 程序编号		
	次	1			
初值：			0	格式：	UDINT
属性：			-	单位：	-
数值范围：			-		

参数功能：

自动加载 PLC 程序编号。

注：此功能暂不开放。

编号	主	10	DI 启动编号		
	次	2			
初值：			0	格式：	UDINT
属性：			-	单位：	-
数值范围：			0 ~ 15		

参数功能：

使用 DI 来启动 PLC 程序。DI 编号：0 ~ 15 分别对应 DXMC 本体 IN0.0 ~ IN1.7。

编号	主	10	DI 启动模式		
	次	3			
初值：			0	格式：	UDINT
属性：			-	单位：	-
数值范围：			-		

参数功能：

设定 DI 启动模式：

0：上升沿触发

1：下降沿触发

编号	主	10	PLC 启动模式		
	次	4			
初值：			0	格式：	UDINT
属性：			-	单位：	-
数值范围：			-		

参数功能：

设定 PLC 启动模式：

0：Cold Start

1：Warm Start

注：Warm Start 功能暂不开放。

P11-x SSI 通讯型编码器设定

编号	主	11	SSI 编码器频率		
	次	0			
初值:			0	格式:	UDINT
属性:			🔒	单位:	-
数值范围:			0 ~ 4		

参数功能:

设定 SSI 编码器频率:

0: 2 MHz

1: 1 MHz

2: 625 kHz

3: 200 kHz

4: 100 kHz

编号	主	11	SSI 编码器多圈分辨率 (Turn)		
	次	1			
初值:			0	格式:	UDINT
属性:			🔒	单位:	Turn
数值范围:			0 ~ 31		

参数功能:

设定 SSI 编码器多圈分辨率 X 位，即分辨率为 2^x。

编号	主	11	SSI 编码器单圈分辨率 (Step)		
	次	2			
初值:			1	格式:	UDINT
属性:			🔒	单位:	Step
数值范围:			1 ~ 32		

参数功能:

设定 SSI 编码器单圈分辨率 Y 位，即分辨率为 2^y。

编号	主	11	SSI 编码器数据格式		
	次	3			
初值:			0	格式:	UDINT
属性:			🔒	单位:	-
数值范围:			0 ~ 31		

参数功能:

设定 SSI 编码器数据格式:

0: Binary

1: Gray

5

P12-x 脉冲型编码器设定 (第一组)

编号	主	12	脉冲型编码器滤波设定		
	次	0			
初值:			0	格式:	UDINT
属性:			🔒	单位:	-
数值范围:			0 ~ 3		

参数功能:

设定脉冲型编码器滤波设定:

0：无滤波

1：0.2 usec

2：0.3 usec

3：0.4 usec

编号	主	12	脉冲型编码器输入设定		
	次	1			
初值:			0	格式:	UDINT
属性:			🔒	单位:	-
数值范围:			0 ~ 3		

参数功能:

设定脉冲型编码器输入设定:

0：AB Phase

1：Pulse/Dir

2：CW/CCW

3：Disable

注：Pulse/Dir、CW/CCW、Disable 功能暂不开放。

P13-x 脉冲型编码器设定 (第二组)

编号	主	13	脉冲型编码器滤波设定		
	次	0			
初值:			0	格式:	UDINT
属性:			🔒	单位:	-
数值范围:			0 ~ 3		

参数功能:

设定脉冲型编码器滤波设定:

0：无滤波

1：0.2 usec

2：0.3 usec

3：0.4 usec

编号	主	13	脉冲型编码器输入设定		
	次	1			
初值:			0	格式:	UDINT
属性:			🔒	单位:	-
数值范围:			0 ~ 3		

参数功能:

设定脉冲型编码器输入设定:

0：AB Phase

1：Pulse/Dir

2：CW/CCW

3：Disable

注：Pulse/Dir、CW/CCW、Disable 功能暂不开放。

5.4 轴参数一览表

5

使用功能块（SYS_ParaRead 或 SYS_ParaWrite）时，控制器参数的对象类型（Type）代号为 0h，根据 DMARS 上的**使用者自定义编号**来填入对象编号（Idx），轴参数主编号和次编号请参考下方表格。

基本设定						
编号		功能	初值	格式	单位	属性
主	次					
0	0	总线类型	-	DINT	-	🔒
0	1	使用者自定义编号	-	UDINT	-	🔒
0	2	PLC 变量名称	-	STRING	-	🔒
0	3	轴启用标志	-	UDINT	-	🔒
0	4	Vendor ID	-	DWORD	-	🔒
0	5	装置类型	-	DWORD	-	🔒
0	6	装置版本	-	DWORD	-	🔒

单位设定						
编号		功能	初值	格式	单位	属性
主	次					
1	0	电子齿轮比分子 (B)	-	UDINT	-	🔒⊖ⓇⓋⒺ
1	1	电子齿轮比分母 (A)	-	UDINT	-	🔒⊖ⓇⓋⒺ
1	2	机械齿轮比分子 (C)	-	UDINT	-	🔒⊖ⓇⓋⒺ
1	3	机械齿轮比分母 (D)	-	UDINT	-	🔒⊖ⓇⓋⒺ
1	4	机械行程分子 (E)	-	UDINT	-	🔒⊖ⓇⓋⒺ
1	5	机械行程分母 (F)	-	UDINT	-	🔒⊖ⓇⓋⒺ
1	6	单位	-	DWORD	-	🔒ⓇⓋⒺ

速度 / 加速度设定						
编号		功能	初值	格式	单位	属性
主	次					
2	0	最大速度	-	LREAL	-	ⓇⓋⒺ
2	1	最大加速度	-	LREAL	-	ⓇⓋⒺ
2	2	最大减速度	-	LREAL	-	ⓇⓋⒺ
2	3	最大加加速度	-	LREAL	-	ⓇⓋⒺ

警告设定						
编号		功能	初值	格式	单位	属性
主	次					
3	0	速度警告值	-	LREAL	-	ⓇⓋⒺ
3	1	加速度警告值	-	LREAL	-	ⓇⓋⒺ
3	2	减速度警告值	-	LREAL	-	ⓇⓋⒺ
3	3	加加速度警告值	-	LREAL	-	ⓇⓋⒺ
3	4	正扭矩警告值	-	LREAL	%	Ⓡ
3	5	负扭矩警告值	-	LREAL	%	Ⓡ
3	6	位置偏差报警值	-	LREAL	-	Ⓡ
3	7	位置偏差警告值	-	LREAL	-	Ⓡ

监测设定						
编号		功能	初值	格式	单位	属性
主	次					
4	0	定位检查时间	-	UDINT	ms	ⓇⒺ
4	1	目标位置到达判断范围	-	LREAL	-	ⓇⒺ
4	2	目标速度到达判断范围	-	LREAL	-	ⓇⒺ
4	3	目标扭矩到达判断范围	-	LREAL	%	Ⓡ
4	4	零速度检出范围	-	LREAL	-	ⓇⒺ

极限设定						
编号		功能	初值	格式	单位	属性
主	次					
5	0	软件位置极限开关	-	UDINT	-	ⓇⓋⒺ
5	1	正软件位置极限	-	LREAL	-	ⓇⓋⒺ
5	2	负软件位置极限	-	LREAL	-	ⓇⓋⒺ

原点复归设定						
编号		功能	初值	格式	单位	属性
主	次					
6	0	原点复归模式	-	DINT	-	Ⓡ
6	1	原点偏移量	-	LREAL	-	Ⓡ
6	2	高速原点复归速度	-	LREAL	-	Ⓡ
6	3	低速原点复归速度	-	LREAL	-	Ⓡ
6	4	原点复归加速度	-	LREAL	-	Ⓡ

5

位置计数设定						
编号		功能	初值	格式	单位	属性
主	次					
7	0	计数模式	-	DINT	-	🔒ⓇⓋⒺ
7	1	模最大位置设定值	-	LREAL	-	🔒ⓇⓋⒺ
7	2	模最小位置设定值	-	LREAL	-	🔒ⓇⓋⒺ

操作设定						
编号		功能	初值	格式	单位	属性
主	次					
8	0	马达方向	-	DINT	-	⛀ⓇⓋ
8	1	PLC 停止时轴停止方式	-	DINT	-	ⓇⓋ

5.5 轴参数说明

P0-x 基本设定

编号	主	0	总线类型		
	次	0			
初值:			-	格式:	DINT
属性:			🔒	单位:	-
数值范围:			0 ~ 4		

参数功能:

显示总线类型:

0：EtherCAT

1：DMCNET

2：CANopen

3：外部通讯轴

4：虚拟轴

编号	主	0	使用者自定义编号		
	次	1			
初值:			-	格式:	UDINT
属性:			🔒	单位:	-
数值范围:			1 ~ 96		

参数功能:

显示用户自定义编号。

编号	主	0	PLC 变量名称		
	次	2			
初值:			-	格式:	STRING
属性:			🔒	单位:	-
数值范围:			-		

参数功能:

显示 PLC 变量名称。

5

编号	主	0	轴启用标志		
	次	3			
初值:			-	格式:	UDINT
属性:			🔒	单位:	-
数值范围:			0 ~ 1		

参数功能:

显示轴启用标志

0：不启用

1：启用

编号	主	0	Vendor ID		
	次	4			
初值:			-	格式:	DWORD
属性:			🔒	单位:	-
数值范围:			-		

参数功能:

显示 Vendor ID，台达的 Vendor ID 为 0x01DD。

编号	主	0	装置类型		
	次	5			
初值:			-	格式:	DWORD
属性:			🔒	单位:	-
数值范围:			-		

参数功能:

显示设备类型。

编号	主	0	装置版本		
	次	6			
初值:			-	格式:	DWORD
属性:			🔒	单位:	-
数值范围:			-		

参数功能:

显示设备版本。

P1-x 单位设定

编号	主	1	电子齿轮比分子（B）		
	次	0			
初值：			-	格式：	UDINT
属性：			🔒⊖ⓇⓋⒺ	单位：	-
数值范围：			1 ~ (2^{29}-1)		

参数功能：

设定电子齿轮比分子，请参考 P1-5 的参数功能说明。

编号	主	1	电子齿轮比分母（A）		
	次	1			
初值：			-	格式：	UDINT
属性：			🔒⊖ⓇⓋⒺ	单位：	-
数值范围：			1 ~ (2^{31}-1)		

参数功能：

设定电子齿轮比分母，请参考 P1-5 的参数功能说明。

编号	主	1	机械齿轮比分子（C）		
	次	2			
初值：			-	格式：	UDINT
属性：			🔒⊖ⓇⓋⒺ	单位：	-
数值范围：			1 ~ (2^{31}-1)		

参数功能：

设定机械齿轮比分子，请参考 P1-5 的参数功能说明。

编号	主	1	机械齿轮比分母（D）		
	次	3			
初值：			-	格式：	UDINT
属性：			🔒⊖ⓇⓋⒺ	单位：	-
数值范围：			-		

参数功能：

设定机械齿轮比分母，请参考 P1-5 的参数功能说明。

5

编号	主 1 次 4	机械行程分子（E）		
初值：		-	格式：	UDINT
属性：		🔒 ⛔ Ⓡ Ⓥ Ⓔ	单位：	-
数值范围：		-		

参数功能：

设定机械行程分子，请参考 P1-5 的参数功能说明。

编号	主 1 次 5	机械行程分母（F）		
初值：		-	格式：	UDINT
属性：		🔒 ⛔ Ⓡ Ⓥ Ⓔ	单位：	-
数值范围：		-		

参数功能：

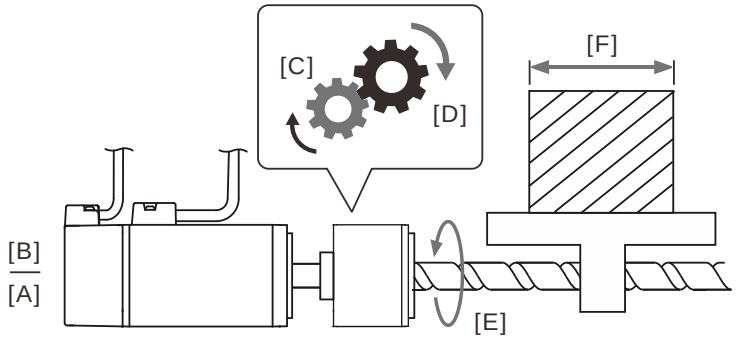
设定机械行程分母。

单位换算成脉冲数的公式如下：

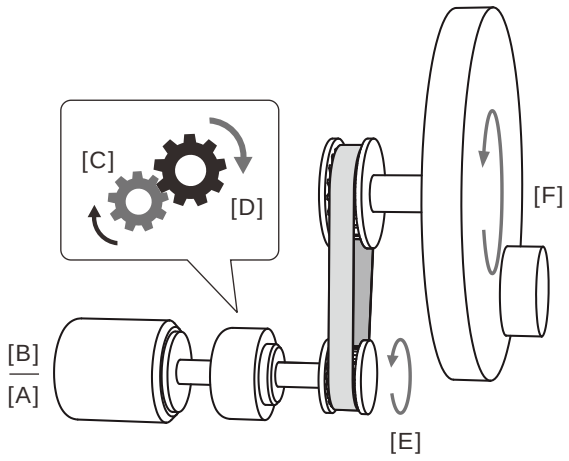
$$运转行程的脉冲数\ (Pulse) = \frac{[B] \times [C] \times [E]}{[A] \times [D] \times [F]} \times 行程$$

注：行程为 PLC 功能块参数使用单位。

线性机构：



旋转机构：



编号	主	1	单位		
	次	6			
初值:			-	格式:	DWORD
属性:			🔒ⓇⓋⒺ	单位:	-
数值范围:			-		

参数功能:

设定单位:

0x00：自定义单位

0x01：mm

0x02：inch

0x03：cm

0x04：um

0x70：0.001 Degree

0x71：Degree

0x72：Rev

5

P2-x 速度 / 加速度设定

编号	主	2	最大速度		
	次	0			
初值:			-	格式:	LREAL
属性:			ⓇⓋⒺ	单位:	-
数值范围:			-		

参数功能:

设定最大速度。

编号	主	2	最大加速度		
	次	1			
初值:			-	格式:	LREAL
属性:			ⓇⓋⒺ	单位:	-
数值范围:			-		

参数功能:

设定最大加速度。

编号	主	2	最大减速度		
	次	2			
初值:			-	格式:	LREAL
属性:			ⓇⓋⒺ	单位:	-
数值范围:			-		

参数功能:

设定最大减速度。

编号	主	2	最大加加速度		
	次	3			
初值:			-	格式:	LREAL
属性:			ⓇⓋⒺ	单位:	-
数值范围:			-		

参数功能:

设定最大加加速度。

P3-x 警告设定

编号	主	3	速度警告值		
	次	0			
初值:			-	格式:	LREAL
属性:			ⓇⓋⒺ	单位:	-
数值范围:			-		

参数功能:

设定速度警告值。

编号	主	3	加速度警告值		
	次	1			
初值:			-	格式:	LREAL
属性:			ⓇⓋⒺ	单位:	-
数值范围:			-		

参数功能:

设定加速度警告值。

编号	主	3	减速度警告值		
	次	2			
初值:			-	格式:	LREAL
属性:			ⓇⓋⒺ	单位:	-
数值范围:			-		

参数功能:

设定减速度警告值。

编号	主	3	加加速度警告值		
	次	3			
初值:			-	格式:	LREAL
属性:			ⓇⓋⒺ	单位:	-
数值范围:			-		

参数功能:

设定加加速度警告值。

5

编号	主	3	正扭矩警告值		
	次	4			
初值:			-	格式:	LREAL
属性:			Ⓡ	单位:	%
数值范围:			-		

参数功能:

设定正扭矩警告值。

编号	主	3	负扭矩警告值		
	次	5			
初值:			-	格式:	LREAL
属性:			Ⓡ	单位:	%
数值范围:			-		

参数功能:

设定负扭矩警告值。

编号	主	3	位置偏差报警值		
	次	6			
初值:			-	格式:	LREAL
属性:			Ⓡ	单位:	-
数值范围:			-		

参数功能:

设定位置偏差报警值。

编号	主	3	位置偏差警告值		
	次	7			
初值:			-	格式:	LREAL
属性:			Ⓡ	单位:	-
数值范围:			-		

参数功能:

设定位置偏差警告值。

P4-x 监测设定

编号	主	4	定位检查时间		
	次	0			
初值:			-	格式:	UDINT
属性:			ⓇⒺ	单位:	ms
数值范围:			-		

参数功能:

设定定位检查时间。

编号	主	4	目标位置到达判断范围		
	次	1			
初值:			-	格式:	LREAL
属性:			ⓇⒺ	单位:	-
数值范围:			-		

参数功能:

设定目标位置到达判断范围。

编号	主	4	目标速度到达判断范围		
	次	2			
初值:			-	格式:	LREAL
属性:			ⓇⒺ	单位:	-
数值范围:			-		

参数功能:

设定目标速度到达判断范围。

编号	主	4	目标扭矩到达判断范围		
	次	3			
初值:			-	格式:	LREAL
属性:			Ⓡ	单位:	%
数值范围:			-		

参数功能:

设定目标扭矩到达判断范围。

5

编号	主	4	零速度检出范围		
	次	4			
初值:			-	格式:	LREAL
属性:			ⓇⒺ	单位:	-
数值范围:			-		

参数功能:

设定零速度检出范围。

P5-x 极限设定

编号	主	5	软件位置极限开关		
	次	0			
初值:			-	格式:	UDINT
属性:			ⓇⓋⒺ	单位:	-
数值范围:			-		

参数功能:

设定软件位置极限开关的有效标志。

0：不启用

1：启用

编号	主	5	正软件位置极限		
	次	1			
初值:			-	格式:	LREAL
属性:			ⓇⓋⒺ	单位:	-
数值范围:			-		

参数功能:

设定正软件位置极限。

编号	主	5	负软件位置极限		
	次	2			
初值:			-	格式:	LREAL
属性:			ⓇⓋⒺ	单位:	-
数值范围:			-		

参数功能:

设定负软件位置极限。

5

P6-x 原点复归设定

编号	主	6	原点复归模式		
	次	0			
初值:			-	格式:	DINT
属性:			Ⓡ	单位:	-
数值范围:			-		

参数功能:

设定原点复归模式。

编号	主	6	原点偏移量		
	次	1			
初值:			-	格式:	LREAL
属性:			Ⓡ	单位:	-
数值范围:			-		

参数功能:

设定原点偏移量之默认值。

注：此功能暂不开放。

编号	主	6	高速原点复归速度		
	次	2			
初值:			-	格式:	LREAL
属性:			Ⓡ	单位:	-
数值范围:			-		

参数功能:

设定高速原点复归速度。

编号	主	6	低速原点复归速度		
	次	3			
初值:			-	格式:	LREAL
属性:			Ⓡ	单位:	-
数值范围:			-		

参数功能:

设定低速原点复归速度。

编号	主	6	原点复归加速度		
	次	4			
初值:			-	格式:	LREAL
属性:			Ⓡ	单位:	-
数值范围:			-		

参数功能:

设定原点复归加速度。

5

P7-x 位置计数设定

编号	主	7	计数模式		
	次	0			
初值:		-	格式:	DINT	
属性:		🔒ⓇⓋⒺ	单位:	-	
数值范围:		-			

参数功能:

设定计数模式:

0：线性

1：循环

编号	主	7	模最大位置设定值		
	次	1			
初值:		-	格式:	LREAL	
属性:		🔒ⓇⓋⒺ	单位:	-	
数值范围:		-			

参数功能:

设定模最大位置设定值。

编号	主	7	模最小位置设定值		
	次	2			
初值:		-	格式:	LREAL	
属性:		🔒ⓇⓋⒺ	单位:	-	
数值范围:		-			

参数功能:

设定模最小位置设定值。

P8-x 操作设定

编号	主	8	马达方向		
	次	0			
初值:			-	格式:	DINT
属性:			ⓇⓋ	单位:	-
数值范围:			-		

参数功能:

设定马达方向:

1: 正转

-1: 反转

编号	主	8	PLC 停止时轴停止方式		
	次	1			
初值:			-	格式:	DINT
属性:			ⓇⓋ	单位:	-
数值范围:			-		

参数功能:

设定 PLC 停止时轴停止方式:

0: 不做任何动作

1: 立即停止

(此页有意留为空白)

PLC 操作接口及功能块介绍 6

此章节详细介绍 MULTIPROG 系统所提供的项目架构、编程语言、变量定义、台达依照 PLCOpen 所建立的功能块、台达自定义数据型态以及 MULTIPROG 系统提供之标准功能。用户可依据自己的需求在 MULTIPROG 系统的基础上进行开发。

6.1 MULTIPROG 系统介绍 ····· 6-6
6.1.1 MULTIPROG 用户接口简介 ····· 6-7
6.1.1.1 功能区及常用工具栏 ····· 6-8
6.1.1.2 专案树 ····· 6-11
6.1.1.3 程序编辑区 ····· 6-12
6.1.1.4 功能块函式库 ····· 6-13
6.1.1.5 状态显示区 ····· 6-13
6.2 IEC 61131-3 概述与 MULTIPROG 系统 ····· 6-14
6.2.1 MULTIPROG 专案架构 ····· 6-16
6.2.2 Configuration (配置) ····· 6-17
6.2.3 Resource (资源) ····· 6-17
6.2.4 Task (任务) ····· 6-21
6.2.5 POU (程序组织单元) ····· 6-26
6.2.6 Data Type (数据型态) ····· 6-32
6.3 MULTIPROG 编程语言 ····· 6-38
6.3.1 指令表(IL)编程语言 ····· 6-38
6.3.2 梯形图(LD)编程语言 ····· 6-40
6.3.3 功能块图(FBD)编程语言 ····· 6-45
6.3.4 结构化文字(ST)编程语言 ····· 6-50
6.3.5 顺序功能图(SFC)编程语言 ····· 6-53
6.4 PLCopen 介绍 ····· 6-57
6.4.1 PLCopen 组织简介 ····· 6-57
6.4.2 PLCopen Motion Control 简介 ····· 6-58
6.4.3 常用变量定义 ····· 6-59
6.5 运动功能块介绍 ····· 6-64
6.5.1 运动功能块总览 ····· 6-64
6.5.2 单轴功能块 (Motion SingleAxis) ····· 6-68
6.5.2.1 MC_Home ····· 6-68
6.5.2.2 MC_Stop ····· 6-72

6

6.5.2.3 MC_Halt ····· 6-76
6.5.2.4 MC_Jog ····· 6-81
6.5.2.5 MC_MoveAbsolute ····· 6-85
6.5.2.6 MC_MoveRelative ····· 6-89
6.5.2.7 MC_MoveAdditive ····· 6-93
6.5.2.8 MC_MoveSuperimposed ····· 6-98
6.5.2.9 MC_SetPosition ····· 6-102
6.5.2.10 DMC_Home ····· 6-105
6.5.2.11 MC_MoveVelocity ····· 6-109
6.5.2.12 MC_Power ····· 6-113
6.5.2.13 MC_Reset ····· 6-115
6.5.2.14 MC_SetOverride ····· 6-117
6.5.2.15 MC_ReadParameter ····· 6-120
6.5.2.16 DMC_SetServoGain ····· 6-122
6.5.2.17 DMC_TouchProbe ····· 6-124
6.5.2.18 MC_AbortTrigger ····· 6-126
6.5.2.19 MC_WriteParameter ····· 6-127
6.5.2.20 MC_TorqueControl ····· 6-129
6.5.3 多轴功能块 (Motion MultiAxis) ····· 6-131
6.5.3.1 MC_GearIn ····· 6-131
6.5.3.2 MC_GearOut ····· 6-135
6.5.3.3 DMC_CamIn ····· 6-137
6.5.3.4 MC_CamOut ····· 6-154
6.5.3.5 MC_PhasingAbsolute ····· 6-156
6.5.3.6 MC_PhasingRelative ····· 6-159
6.5.3.7 DMC_GantrySynchornize ····· 6-162
6.5.4 群轴功能块 (Motion Group) ····· 6-164
6.5.4.1 MC_GroupStop ····· 6-164
6.5.4.2 MC_MoveDirectAbsolute ····· 6-167
6.5.4.3 MC_MoveDirectRelative ····· 6-171
6.5.4.4 MC_MoveLinearAbsolute ····· 6-175
6.5.4.5 MC_MoveLinearRelatve ····· 6-179
6.5.4.6 MC_GroupReset ····· 6-183
6.5.4.7 MC_GroupSetOverride ····· 6-185
6.5.5 系统功能块 (System Function) ····· 6-188
6.5.5.1 SYS_ParaRead ····· 6-188
6.5.5.2 SYS_ParaWrite ····· 6-189
6.5.5.3 SYS_GetAlarmCode ····· 6-191
6.5.5.4 SYS_ResetAlarmCode ····· 6-193

6.5.6 自定义系统功能块 (DMC Function) 6-194
6.5.6.1 CEIL 6-194
6.5.6.2 FLOOR 6-195
6.5.6.3 Ring_Queue 6-196
6.5.7 自定义通讯功能块 (DMC FieldBus) 6-199
6.5.7.1 CAN_PDORead 6-199
6.5.7.2 CAN_PDOWrite 6-201
6.5.7.3 CAN_SDORead 6-203
6.5.7.4 CAN_SDOWrite 6-205
6.5.7.5 MB_ReadBits 6-207
6.5.7.6 MB_ReadWords 6-210
6.5.7.7 MB_SerialPort_Set 6-213
6.5.7.8 MB_TCP_Connect 6-215
6.5.7.9 MB_TCP_DisConnect 6-218
6.5.7.10 MB_TCP_ReadBits 6-220
6.5.7.11 MB_TCP_ReadWords 6-223
6.5.7.12 MB_TCP_WriteBit 6-226
6.5.7.13 MB_TCP_WriteBits 6-229
6.5.7.14 MB_TCP_WriteWord 6-233
6.5.7.15 MB_TCP_WriteWords 6-236
6.5.7.16 MB_WriteBit 6-239
6.5.7.17 MB_WriteBits 6-242
6.5.7.18 MB_WriteWord 6-246
6.5.7.19 MB_WriteWords 6-249
6.5.8 凸轮编辑功能块 (ECAM Editor) 6-253
6.5.8.1 ECAM_GenTableByFile 6-253
6.5.8.2 ECAM_GenTableByData 6-257
6.5.8.3 ECAM_GenTableByVel 6-262
6.5.8.4 ECAM_AddPoints 6-265
6.5.8.5 ECAM_DelPoints 6-270
6.5.8.6 ECAM_ModifyPoints 6-273
6.5.8.7 ECAM_ReadPointsByID 6-278
6.5.8.8 ECAM_ReadTableByID 6-282
6.5.8.9 ECAM_SaveTable 6-286
6.5.8.10 ECAM_GenRotaryCutTable 6-290
6.5.8.11 ECAM_GenFlyingShearTable 6-293
6.5.8.12 ECAM_SetCamTappetData 6-297
6.5.8.13 ECAM_GetCamTappetStatus 6-301
6.5.8.14 ECAM_SetConnectVelocity 6-304

6.5.8.15 ECAM_PreviewTable ····· 6-307
6.5.8.16 ECAM_ModRotCutVelRatio ····· 6-310
6.5.8.17 ECAM_SetOnlineChgMode ····· 6-312
6.5.9 自定义数据型态 ····· 6-315
6.5.9.1 自定义数据型态总览 ····· 6-315
6.5.9.2 任务信息数据型态 (Task Information) ····· 6-315
6.5.9.3 PLCOpen 数据型态 (PLCOpenDataType) ····· 6-316
6.5.9.4 ECAM 数据型态 (ECAMEditorDataType) ····· 6-321
6.6 标准功能 ····· 6-323
6.6.1 标准功能总览 ····· 6-323
6.6.2 指令启用说明 (EN 和 ENO 说明) ····· 6-325
6.6.3 数学运算功能 ····· 6-326
6.6.3.1 ABS ····· 6-326
6.6.3.2 ACOS ····· 6-327
6.6.3.3 ASIN ····· 6-328
6.6.3.4 ATAN ····· 6-329
6.6.3.5 COS ····· 6-330
6.6.3.6 EXP ····· 6-330
6.6.3.7 LN ····· 6-331
6.6.3.8 LOG ····· 6-331
6.6.3.9 SIN ····· 6-332
6.6.3.10 SQRT ····· 6-333
6.6.3.11 TAN ····· 6-334
6.6.3.12 ADD ····· 6-335
6.6.3.13 ADD_T_T ····· 6-336
6.6.3.14 DIV ····· 6-337
6.6.3.15 DIV_T_AI ····· 6-338
6.6.3.16 DIV_T_AN ····· 6-339
6.6.3.17 DIV_T_R ····· 6-340
6.6.3.18 EXPT ····· 6-341
6.6.3.19 MOD ····· 6-342
6.6.3.20 MOVE ····· 6-343
6.6.3.21 MUL ····· 6-344
6.6.3.22 MUL_T_AI ····· 6-345
6.6.3.23 MUL_T_AN ····· 6-346
6.6.3.24 MUL_T_R ····· 6-347
6.6.3.25 NEG ····· 6-348
6.6.3.26 SUB ····· 6-349
6.6.3.27 SUB_T_T ····· 6-350

6.6.4 位串功能 6-351
6.6.4.1 AND 6-351
6.6.4.2 NOT 6-352
6.6.4.3 OR 6-353
6.6.4.4 XOR 6-354
6.6.5 位移功能 6-355
6.6.5.1 ROL 6-355
6.6.5.2 ROR 6-356
6.6.5.3 SHL 6-357
6.6.5.4 SHR 6-358
6.6.6 选择运算功能 6-359
6.6.6.1 LIMIT 6-359
6.6.6.2 MAX 6-360
6.6.6.3 MIN 6-361
6.6.6.4 SEL 6-362
6.6.7 比较运算功能 6-363
6.6.7.1 EQ 6-363
6.6.7.2 GE 6-364
6.6.7.3 GT 6-365
6.6.7.4 LE 6-366
6.6.7.5 LT 6-367
6.6.7.6 NE 6-368
6.6.8 字符串操作功能 6-369
6.6.8.1 CONCAT 6-369
6.6.8.2 DELETE 6-370
6.6.8.3 EQ_STRING 6-371
6.6.8.4 FIND 6-372
6.6.8.5 GE_STRING 6-373
6.6.8.6 GT_STRING 6-374
6.6.8.7 INSERT 6-375
6.6.8.8 LE_STRING 6-376
6.6.8.9 LEFT 6-377
6.6.8.10 LEN 6-378
6.6.8.11 LT_STRING 6-379
6.6.8.12 MID 6-380
6.6.8.13 NE_STRING 6-381
6.6.8.14 REPLACE 6-382
6.6.8.15 RIGHT 6-383

6.1 MULTIPROG 系统介绍

6

DXMC 的 PLC 编辑接口为 MULTIPROG 系统，而 MULTIPROG 系统是基于 IEC61131-3 标准 PLC 语言规范所建立的软件编辑系统，此系统更具备变量状态监控、示波器、PLC 指令编写等功能整合软件。

MULTIPROG 是 Phoenix Contact 公司针对自动化应用开发的通用型 PLC 程序设计软件。主要针对中大型控制应用场合，可以广泛的应用在机械制造、传统产业、自动化等行业。

此软件是基于 COM / DCOM 的技术架构，适用于 Windows 7 以上操作系统，其结构符合 IEC61131-3 标准，支持标准定义的五种 PLC 程序语言，且可以让使用者自行定义函式库及数据结构。该程序编辑工具可以应用在现有的控制系统，也可同时使用多个任务来对 PLC 控制器进行统一配置、程序编辑和操作。

MULTIPROG 更提供了丰富的操作命令和人机互动接口，支持功能块拖曳编辑且所有功能皆可透过键盘来操作。亦提供变量实时监控，强制设定及覆写功能，并有完整侦错模式，可以在程序设置断点和单步执行，同时有逻辑分析功能，便利的记录输入输出波型。在特殊应用上，可以进行原始码保护和不停机的在线下载功能。针对不同国家的程序编辑习惯，提供包含变量名称在内的多国语言，目前支持英文、简体中文、日文、德文。

6.1.1 MULTIPROG 用户接口简介

MULTIPROG 操作接口主要可分为五大部分，依序为(1)功能区及常用工具栏、(2)项目树、(3)程序编辑区、(4)功能块函式库以及(5)状态显示区。

图 6.1.1.1 MULTIPROG 用户接口

功能区包括 11 个下拉菜单，以及常用的编译工具。

项目树分为[Project]和[Hardware]页签，若是使用台达所开发的样版所建立的 Project，则[Project]页签中内含预设的 DXMC PLC 样版和变量型态宣告，而[Hardware]页签则说明整个项目配置。工作区是编辑程序的区块，其中具有五种程序语言可选择，针对各种不同需求进行开发。功能块函式库则提供基础运算、逻辑判断以及符合 PLCopen 规范的运动功能块，状态显示区则显示编译后的错误、警报等信息，同时也可以用来观察变量的变化。

6

6.1.1.1 功能区及常用工具栏

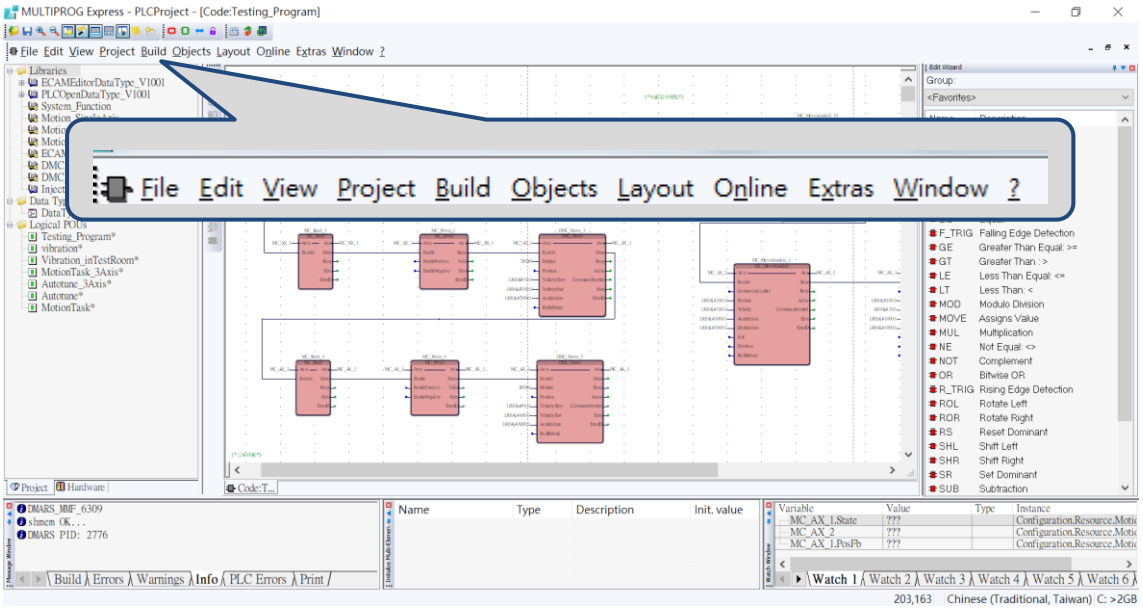
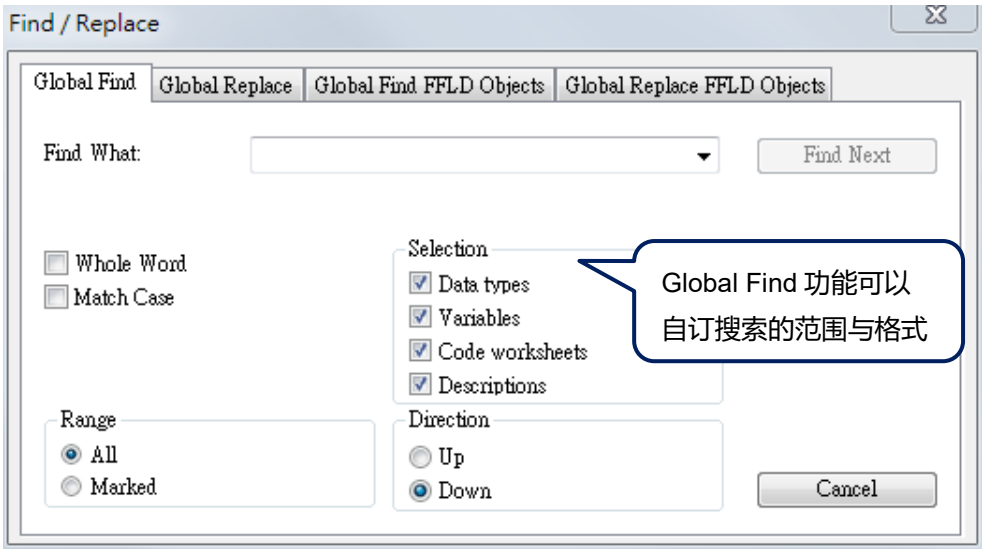
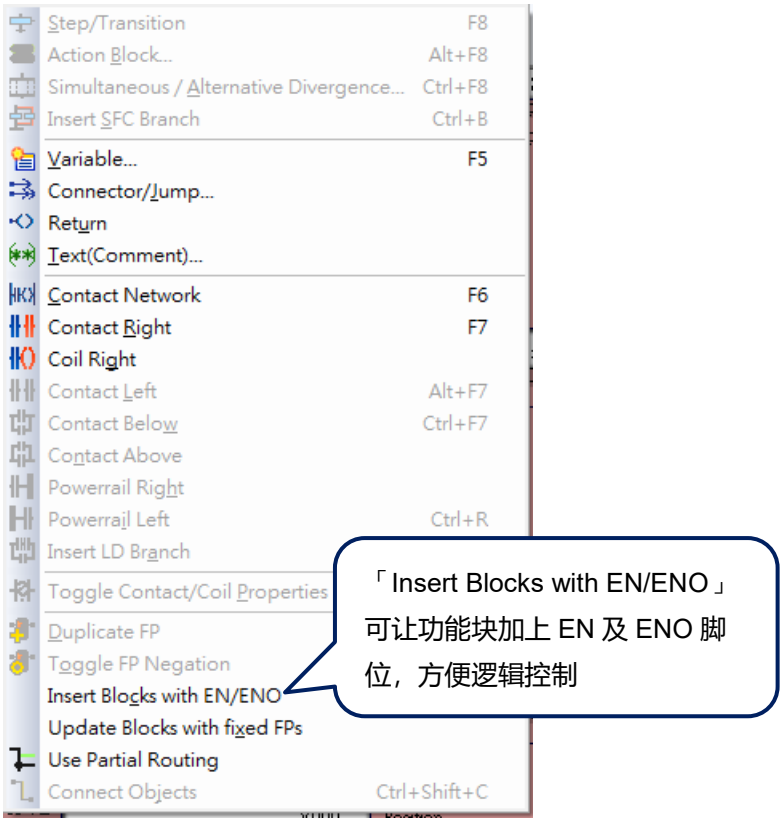


图 6.1.1.1.1 功能区

选单	叙述
File	包含新建、开启、储存、关闭、删除项目、储存、删除样板、打印、打印设定以及打印预览等功能。
Edit	包含常用的剪下、复制以及贴上等基本编辑功能，以及全局变量搜寻及取代选项(Edit → Global Find)，可透过自定义搜寻限制范围或吻合度，方便大范围的搜寻。 
View	用于编辑操作接口。用户可选择区块的显示或隐藏并自行调整接口。
Project	可新增函式库、变量型态和程序管理单元。
Build	提供编译、协助除错和移除未使用到的变量和功能块等功能。
Objects	只有在编辑程序会显示。其中[Variable]功能用来插入新变量和编辑图形化语言时，可以直接插入 Network、Coil 等梯形图组件，其他不支持的组件则会显示为灰色，表示无法点选。

选单	叙述
Layout	提供页面放大缩小功能。实际选项会根据当前程序 POU 是图形化还是文本式的编辑语言会有些许不同。[Insert Block with EN/ENO] 功能如下图： 
Online	提供项目侦错指令。子选单中的[Logic Analyzer]可以记录变量的数值，并将多组数值绘制成曲线供用户比较观察。
Extras	选单中的[PageLayoutEditor]可用来预览及编辑打印的画面布局，其中 Option 功能则可以让用户自定义软件的外观。
Window	可安排窗口及编辑按钮。

6

选单	叙述
?	供所有指令和功能的协助。软件的基本操作可以参考 PLC Help，基础功能块的定义则可以查询 Help on Standard FB/FU，如下图。

除了下拉选单外，还有常用按钮列，提供档案存取、接口设计、编辑除错等常用功能。

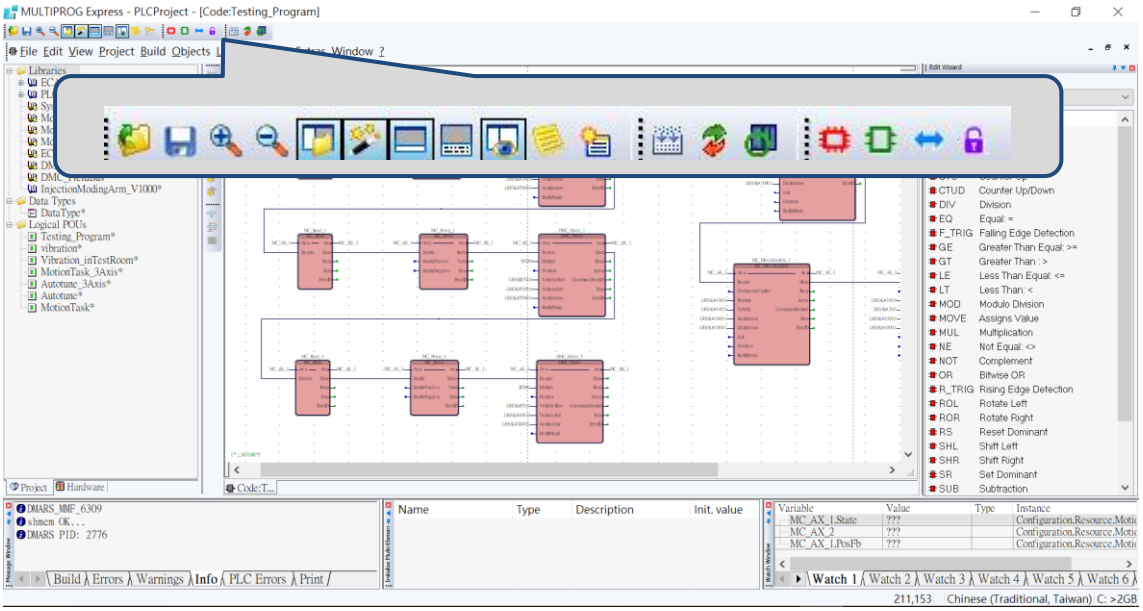


图 6.1.1.1.2 常用工具栏

6.1.1.2 专案树

项目树分为[Project]和[Hardware]两个子页签。[Project]页签包含了样版提供的函式库，而函式库可再细分成变量型态(Data Type)与样版提供的基本程序单元 Logical POUs。用户可点开这两个子项目并开查看内容，但样板无提供程序编辑修改。有钥匙图案的项目，如 System_Function 等七个项目，此项目为韧体提供的函式库，无提供查看。使用者可以自行在最下方的 Logical POUs 新增需要的功能，但要尽量避免和样版提供的程序有所冲突，以免发生错误。

图 6.1.1.2.1 专案树 Project 页面

[Hardware]页面介绍整个项目的配置，可以针对需求将 POU 放置到不同的 Tasks 中。
[Global Variables]可以查看所有的变量。
[IO_Configuration]可以查看目前选择产品的 IO 配置。
关于 Hardware 详细说明，请参考第 6.2 章。

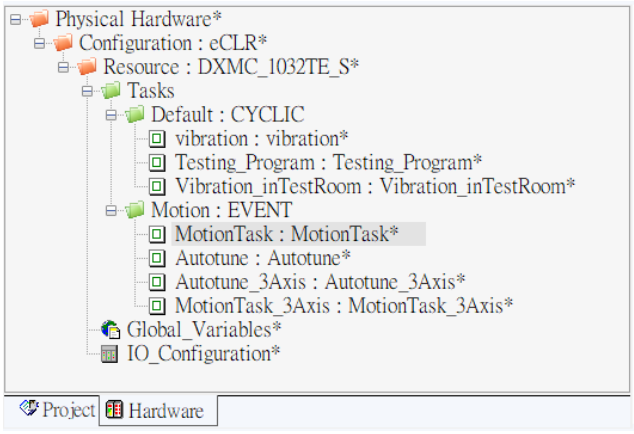


图 6.1.1.2.2 专案树 Hardware 页面

6

6.1.1.3 程序编辑区

此窗口用来编写 PLC 程序，下方的页签可以切换当前选择的程序或是变量页面，图形化语言可以互相配合使用。不同的程序可以透过不同的 PLC 语言撰写，透过不同语言的优势来实现所需功能。

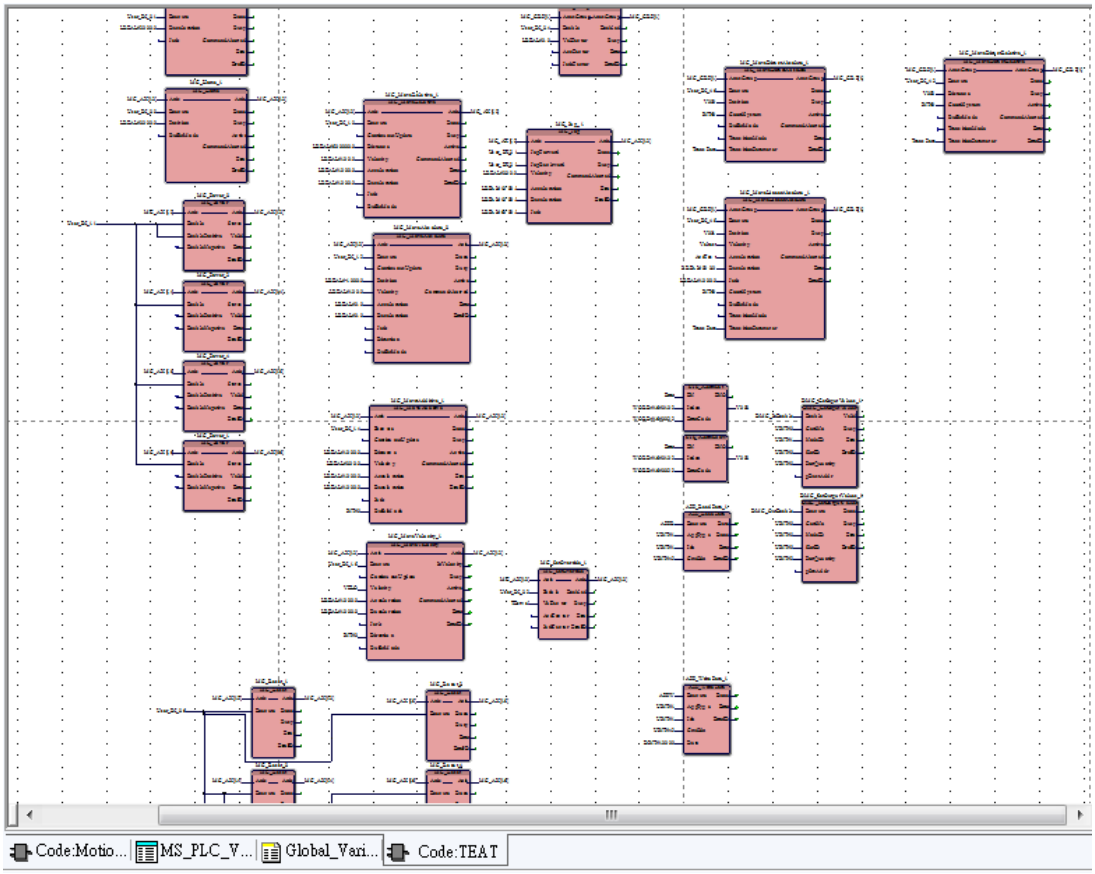


图 6.1.1.3.1 程序编辑区

6.1.1.4 功能块函式库

此窗口显示目前使用的函式库中所有的功能块。从函式库直接拖曳功能块进编辑区直接编辑使用。点选[Object]选单中的[Insert Blocks with EN / ENO]选项，可以让功能块增加 EN / ENO 脚位，方便逻辑控制。基础功能块定义可以参考 [?] 选单中的 [Help on Standard FB/FU]，符合 PLCOpen 定义的功能块，详细说明可参考本手册第 6.4 章。

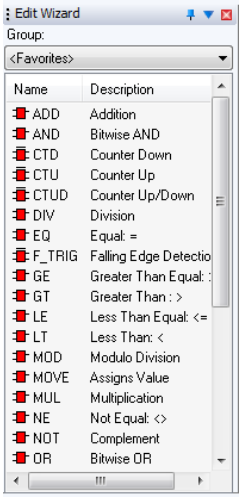


图 6.1.1.4.1 功能块函式库

6.1.1.5 状态显示区

此区块主要进行状态的显示，[Message Window]提供编译的错误以及警告信息，[Watch Window]监控变量的数值变化，[Logic Analyzer]可以同时观察多个变量，并将变量绘制成曲线图，方便观察各变量变化状况。最底下状态栏显示目前的语言，及磁盘目前剩余空间。

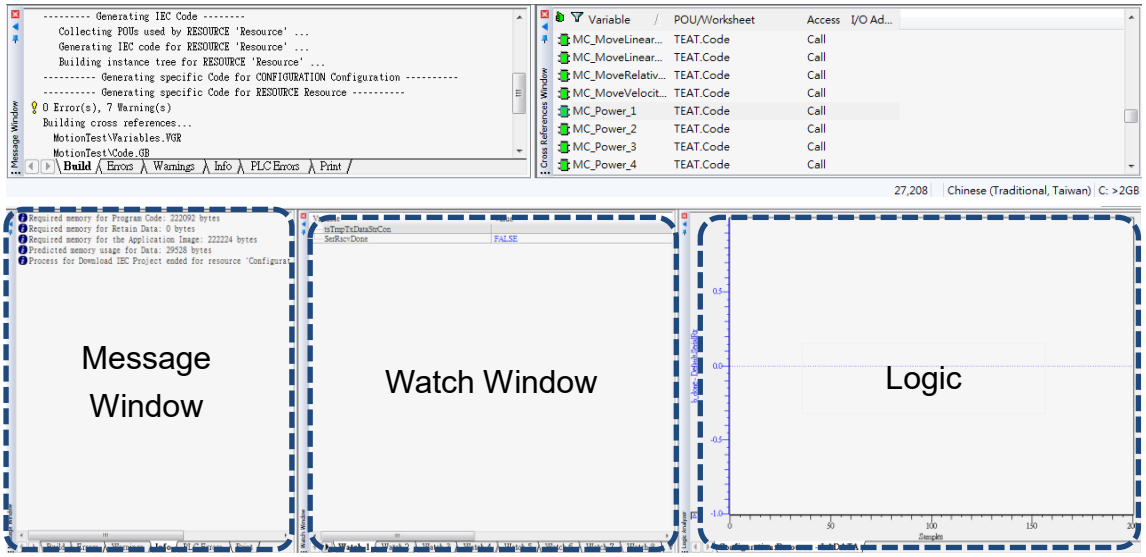


图 6.1.1.5.1 状态显示区

6.2 IEC 61131-3 概述与 MULTIPROG 系统

6

早期 PLC 控制程序软件大多使用阶梯图程序(Ladder Diagram, LD)语言，其语法简单易学，因此被广泛使用于 PLC 应用程序之开发。

随着众多厂商纷纷投入可程控器的开发，使得可程控器的语法越来越多，造成使用者在不同厂牌之间程序转换不便。因此，1993 年国际电工委员会(IEC)制定了可编程控制器的国际标准 IEC 61131，其中的第三部分关于编程语言的标准，规范了可编程控制器的编程语言及其基本元素。

IEC61131-3 标准的软件模型采用层次结构来表示，各层的关系如图所示。

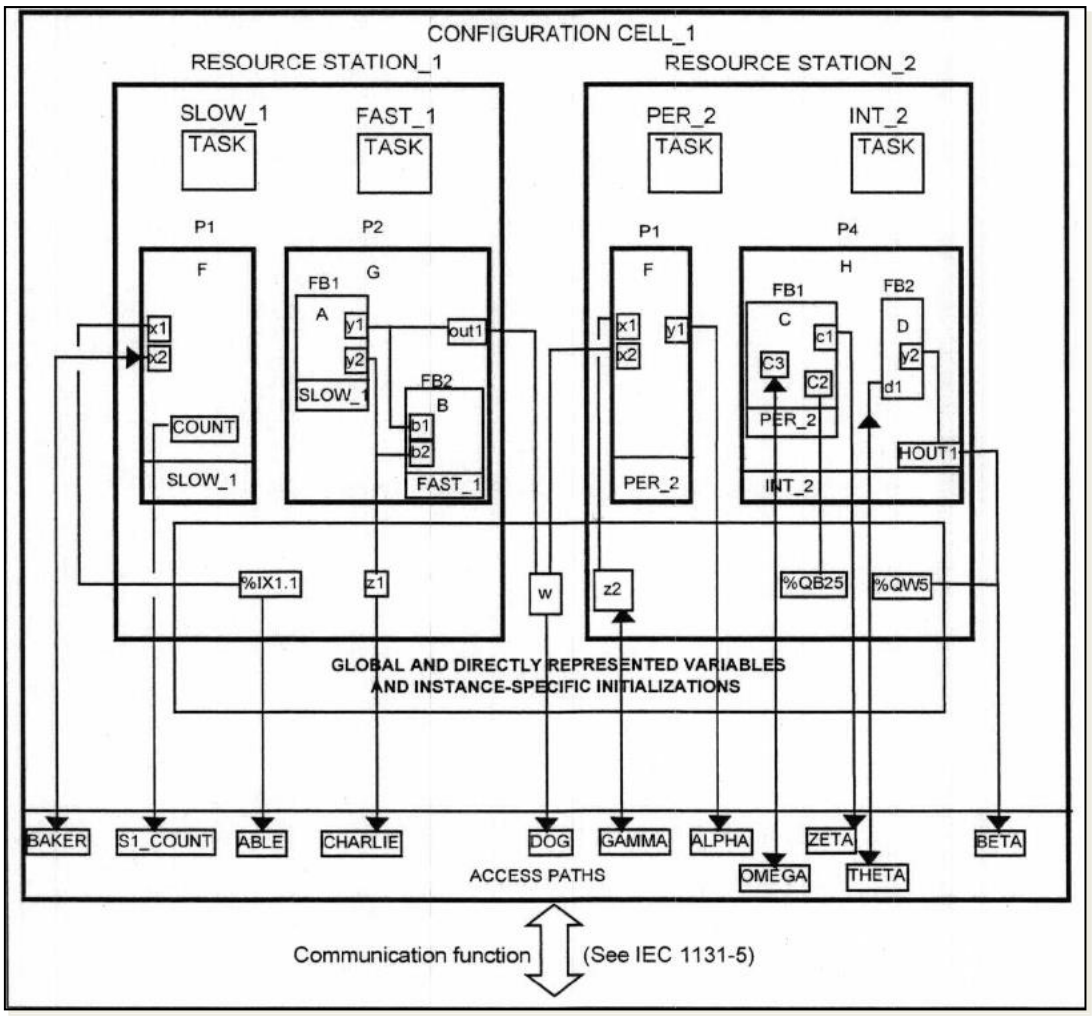


图 6.2.1.1 IEC61131-3 软件架构各层关系图(出处: INTERNATIONAL STANDARD IEC 61131-3)

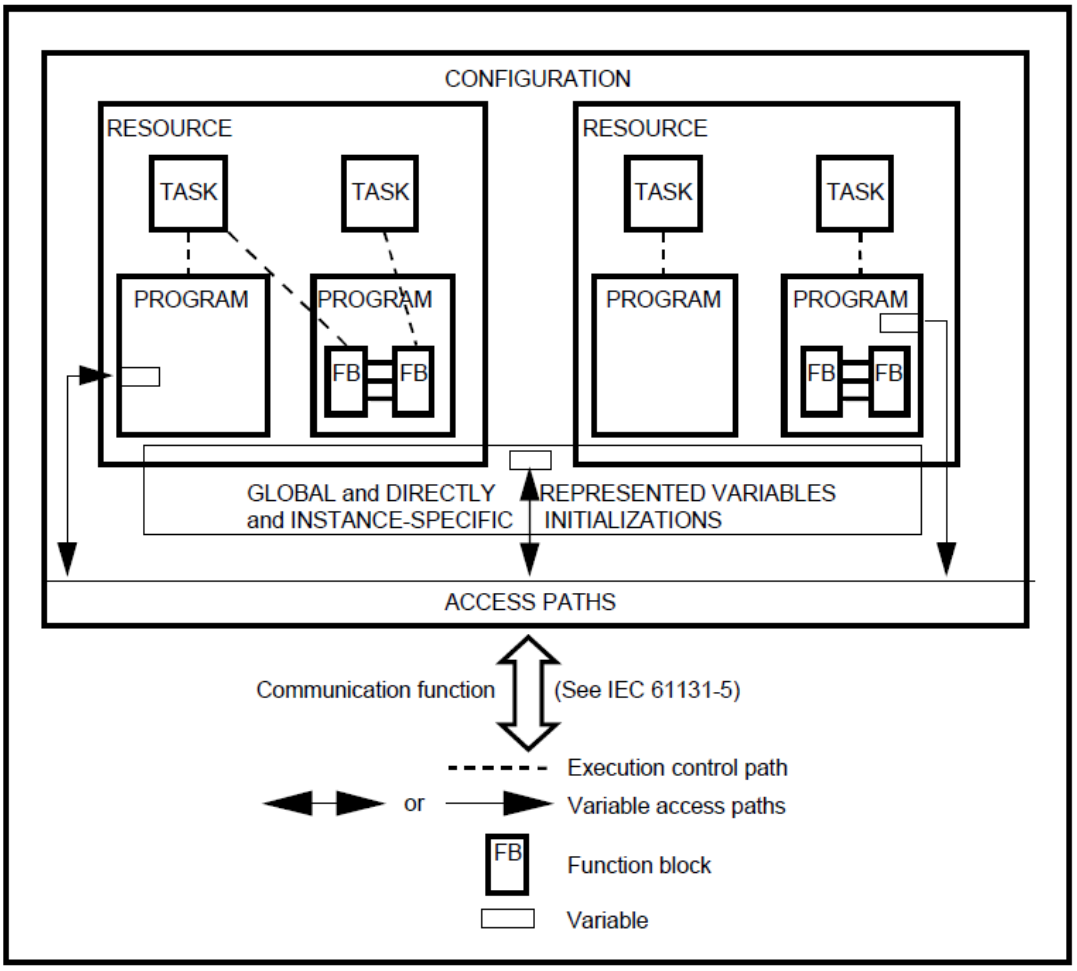


图 6.2.1.2 IEC61131-3 软件架构示意图 (出处: INTERNATIONAL STANDARD IEC 61131-3)

6

6.2.1 MULTIPROG 专案架构

MULTIPROG 遵循了 IEC61131-31 标准定义的层次结构。透过 MULTIPROG 开启项目后，左侧的项目树窗口中会列出当前项目所用到的所有节点信息，如图所示。

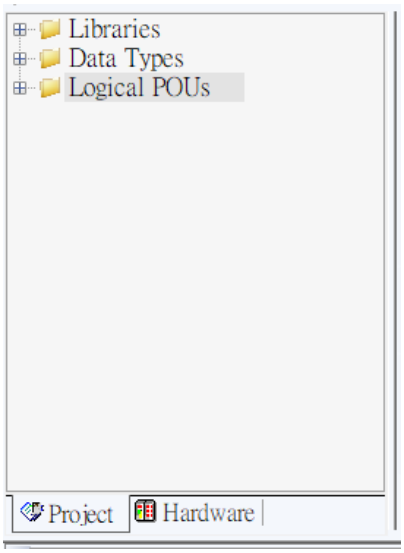


图 6.2.1.3 专案树窗口 Project 页面

点击项目树窗口下方的 Hardware 选项，切换至[Hardware]页面，并点击 Physical Hardware 节点，即可展开 Physical Hardware 的配置(Configuration)。用户可以看到 Configuration (配置)、Resource (资源)、Task (任务)、Global_Variables 和 IO_Configuration 这几个节点，如图所示。其中 Task 节点默认包含一个 Motion 任务。

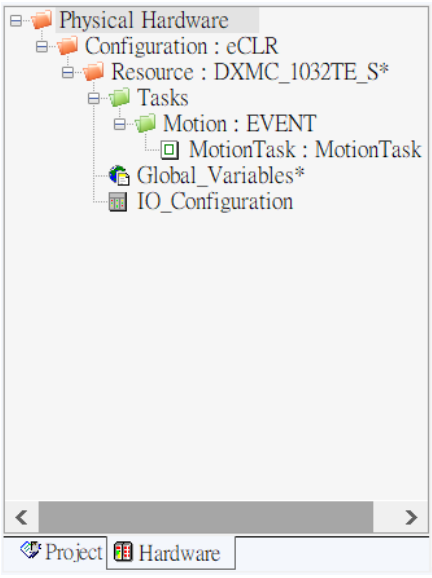


图 6.2.1.4 专案树窗口 Hardware 页面

6.2.2 Configuration (配置)

Configuration(配置)对应于 IEC61131-3 标准中最外层的框架，用来定义控制系统的特性，包括硬设备、中央处理器资源、I/O 信道、储存和通信地址等。通常一个配置对应一台实际的控制器。当需要处理的控制问题较复杂时，则需要几台控制器的相互配合，此时可以定义多个配置，每个配置都是相互独立的个体，不过可以通过通信接口实现信息的交换。

目前 DXMC 配合的 MULTIPROG 5.51 Express 版本仅提供一组配置 eCLR 设定。

6.2.3 Resource (资源)

Resource(资源)节点位于 IEC61131-3 软件架构的第二层，可支持多个运行的程序。资源反映了控制器的物理结构，为程序和 PLC 的物理输入输出通道提供一个接口。一个资源可以增加并执行多个相互独立的程序。

资源可以通过资源名称来加以区别，一个资源相当于一个 CPU，所以资源也可以简单理解为控制器里的中央处理器单元。

目前 DXMC 配合的 MULTIPROG 5.51 Express 版本仅提供一组资源设定。

■ Settings

鼠标右击 Resource 节点，在弹出的选单中选择 Settings…，如图所示，其弹出的窗口如图所示。

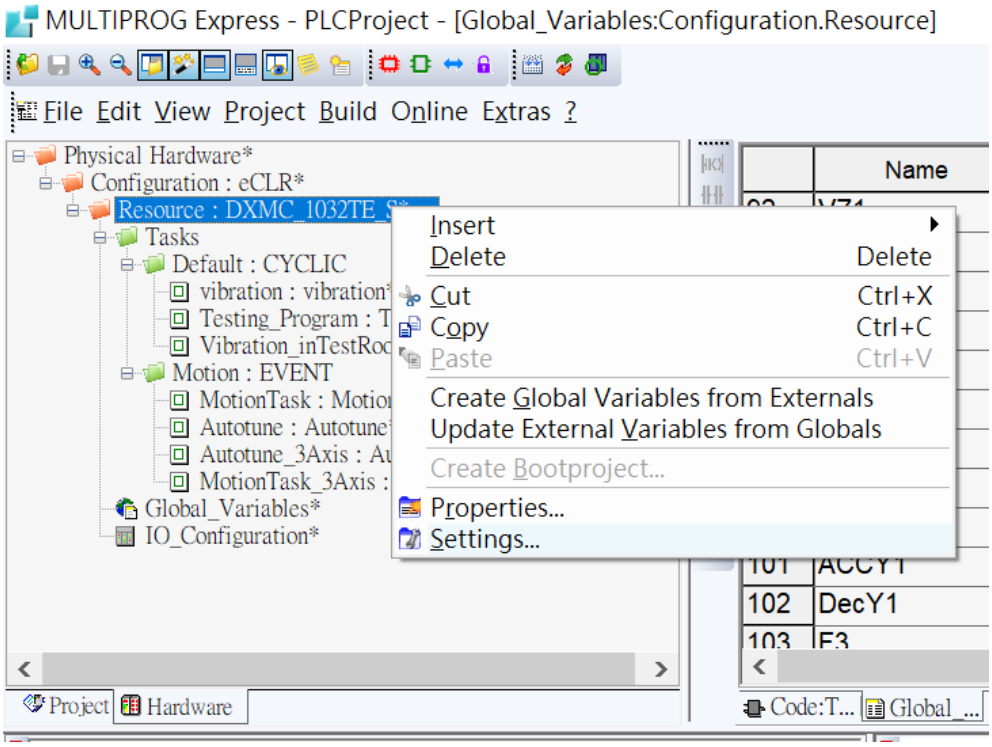


图 6.2.2.1 Resource Setting 路径

6

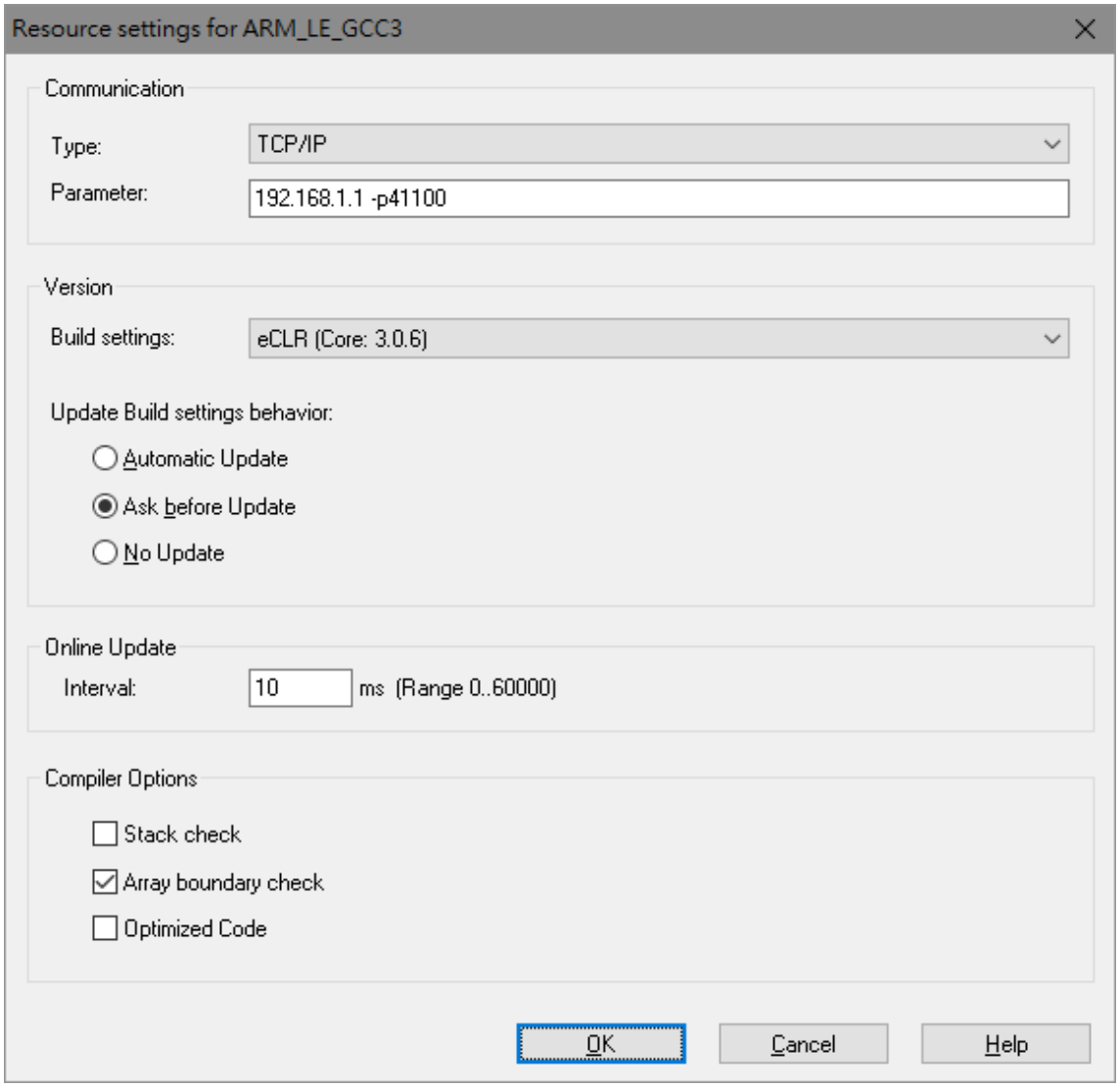


图 6.2.2.2 Resource Setting 页面

MULTIPROG 透过 TCP/IP 模式与 DXMC 通讯，用户可以透过更改 Parameter 的 IP 地址来连接不同的 DXMC 设备。

Online Update 的时间间隔指的是在使用 MULTIPROG 的 Debug 模式时，控制器内部变量的数据在 MULTIPROG 界面上显示的更新周期。

■ Global Variables

每个资源(Resource)中仅提供一组的「Global variables」节点。双击该节点即可开启全局变量表，如图所示。

	Name	Type	Usage	Description	Address	Init	Retain	DeviceMapping
1	⊟ SV							
2	Axis1ServoON	BOOL	VAR_GL...	Axis Servo On Co...	%MX3.0.0		☐	☐
3	Axis2ServoON	BOOL	VAR_GL...		%MX3.1.0		☐	☐
4	Axis3ServoON	BOOL	VAR_GL...		%MX3.2.0		☐	☐
5	Axis4ServoON	BOOL	VAR_GL...		%MX3.3.0		☐	☐
6	Axis5ServoON	BOOL	VAR_GL...		%MX3.4.0		☐	☐
7	Axis6ServoON	BOOL	VAR_GL...		%MX3.5.0		☐	☐
8	Axis13ServoON	BOOL	VAR_GL...		%MX3.12.0		☐	☐
9	Axis14ServoON	BOOL	VAR_GL...		%MX3.13.0		☐	☐
10	Axis15ServoON	BOOL	VAR_GL...		%MX3.14.0		☐	☐
11	Axis16ServoON	BOOL	VAR_GL...		%MX3.15.0		☐	☐
12	Axis1FaultRst	BOOL	VAR_GL...	Axis Fault Reset ...	%MX3.64.0		☐	☐
13	Axis2FaultRst	BOOL	VAR_GL...		%MX3.65.0		☐	☐
14	Axis3FaultRst	BOOL	VAR_GL...		%MX3.66.0		☐	☐
15	Axis4FaultRst	BOOL	VAR_GL...		%MX3.67.0		☐	☐

图 6.2.2.3 Global Variables 页面

每个资源中仅提供一组全局变量表，在其所属的资源中存在的任何一个程序都可以存取全局变量表中的变量，使同一资源中的不同程序可以交换数据和共享数据。

全局变量工作单中比较常用的是 Name、Type、Address 和 Init4 个属性。

Address 用来表示该变量是否和特定的物理地址相关联，关联有两种类型：一种是和实体 I/O 模块相关联，另外一种是和内存中的某一地址关联。

以「%M」开头表示该变量存在于控制器的共享内存中，后面紧跟着的一个字母表示数据的类型。第一位数字表示共享内存的第几个区域，目前只支持 0、1、3 这三个数字，其中 M0 与 M1 为系统所使用，用户只能使用 M3 区域。第一个小数点后的一串数字表示该变量在共享内存区域的字节(Byte)偏移量。如果数据的类型是位(对应字母为 X)，则有第二个小数点，其后的一个数字代表在前面一串数字表示的偏移量所定位的一个字节中的第几位，范围为 0 ~ 7。

以「%I」开头的表示该全局变量关联到实体上的一个输入，后面紧跟着的一个字母表示数据类型，其后的一串数字表示从输入起始到该变量所在地址的字节偏移量。如果数据类型为一个位，则有一个小数点，其后的数字表示位的偏移量。

关联到实体上输出的全局变量的地址命名规则和关联到输入的变量相同，只是以「%Q」开始。

6

■ 实例说明

宣告 BOOL 型态变量%MX3.0.1，「%M」开头表示该变量存在于控制器的共享内存中，「X」则为 BOOL 型态使用，「3」代表为 M3 区域，「0」为 PLC 地址，根据使用地址范围，可分为 DV 区、DH 区、SDV 区及 SDH 区，最后「1」只有在变量为位型态时候才有，其后的一个数字代表在前面一串数字表示的偏移量所定位的一个字节中的第几位，范围为 0 ~ 7。

	Name	Type	Usage	Description	Address
1	⊟ **Var**				
2	Var_Bool	BOOL	VAR_GLOBAL		%MX3.0.1

宣告 Byte 型态变量%MB3.0，「%M」开头表示该变量存在于控制器的共享内存中，「B」则为 BYTE 使用，「3」代表为 M3 区域，「0」为 PLC 地址，根据使用地址范围，可分为 DV 区、DH 区、SDV 区及 SDH 区。

	Name	Type	Usage	Description	Address
1	⊟ **Var**				
2	Var_Byte	BYTE	VAR_GLOBAL		%MB3.0

宣告 16 位型态的变量%MW3.0，如 INT、WORD、UDINT 等…，「%M」开头表示该变量存在于控制器的共享内存中，「W」则为 16 位所使用，「3」代表为 M3 区域，「0」为 PLC 地址，根据使用地址范围，可分为 DV 区、DH 区、SDV 区及 SDH 区。

	Name	Type	Usage	Description	Address
1	⊟ **Var**				
2	Var_WORD	INT	VAR_GLOBAL		%MW3.0

宣告 32 位型态的变量%MD3.0，如 REAL、DWORD、DINT 等…，「%M」开头表示该变量存在于控制器的共享内存中，「D」则为 32 位所使用，「3」代表为 M3 区域，「0」为 PLC 地址，根据使用地址范围，可分为 DV 区、DH 区、SDV 区及 SDH 区。

	Name	Type	Usage	Description	Address
1	⊟ **Var**				
2	Var_DWORD	REAL	VAR_GLOBAL		%MD3.0

宣告 64 位型态的变量%ML3.0，如 LREAL、LINT 等…，「%M」开头表示该变量存在于控制器的共享内存中，「L」则为 64 位所使用，「3」代表为 M3 区域，「0」为 PLC 地址，根据使用地址范围，可分为 DV 区、DH 区、SDV 区及 SDH 区。

	Name	Type	Usage	Description	Address
1	⊟ **Var**				
2	Var_LINT	LREAL	VAR_GLOBAL		%ML3.0

■ IO Configuration

当有变量与实体输入输出地址相关时，就需要使用 I/O 设置。双击项目树中 I/O_Configuration 节点，就可以弹出 I/O 设置窗口，目前可以检视设定结果，可对这些变量的属性和描述进行更改。

设定配置主要是搭配 DXMC 内部芯片参数设定，目前 DXMC 仅提供两组 Input 和两组 Output 的定义。

6.2.4 Task (任务)

任务(Task) 位于 IEC61131-3 软件架构的第三层，是程序规划的最小单位，其用于管理加载程序组织单元(POU)的类型和运行周期(POU 的介绍将在下一章说明)。为了能更清楚地区分不同功能的任务，方便编程人员使用，MULTIPROG 定义了 3 种任务类型，分别是 Default、Cyclic、Event，这些任务的优先权由低到高分别为 Default、Cyclic、Event。

■ Task 任务类型

Default 任务：与传统 PLC 中运行的程序相同，当控制器上电之后便一直重复循环运行，且可以被其他任务插断。

Cyclic 任务：此为周期性的任务，包含「运行间隔」(Interval)的参数设定，每隔设定的间隔时间，任务即被呼叫运行，且可以被 Event 任务插断。

Event 任务：此类型任务是根据事件而触发的，在控制器运行时该任务处于非运行状态，一旦对应的事件被触发，该任务就会开始被执行。该类型的任务通常用于处理较紧急的事件。目前 DXMC 仅提供一组 Event 任务，因此用户无法自行增加 Event 任务。

6

■ 增加删除任务

一个资源可以建立多个任务的运行，这些任务可以同时运行。用户可以通过 Tasks 节点上点击鼠标右键，然后在选单中选择 [Insert] > [Task] 来增加任务，如图所示。

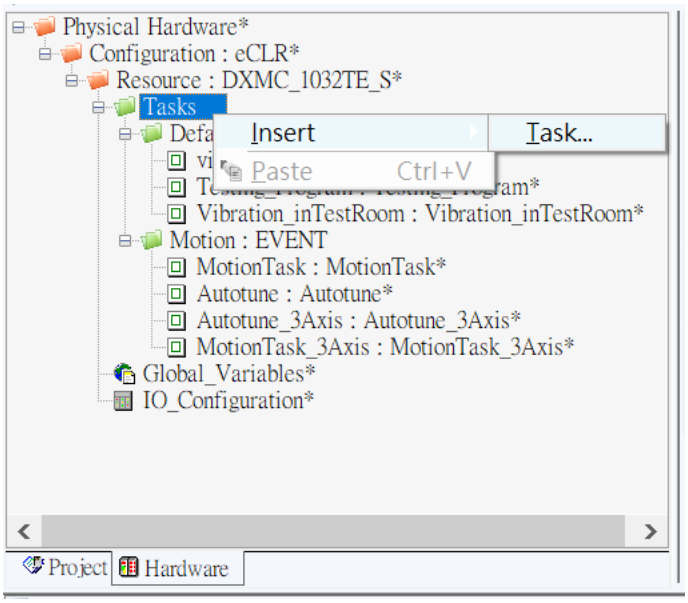


图 6.2.4.1 新增任务

在弹出的窗口中写入任务的名称并选择任务的类型，点击 **OK**，如图所示。

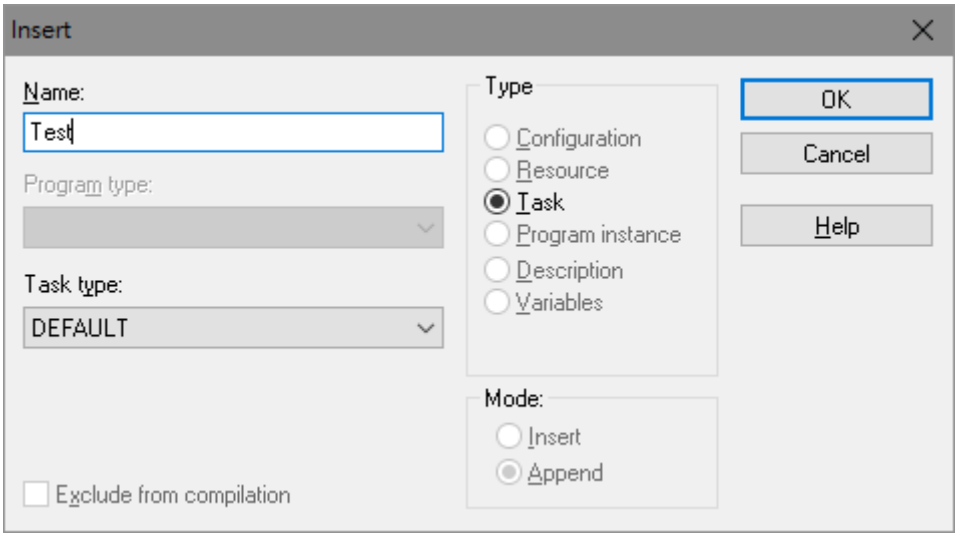


图 6.2.4.2 任务属性设置

当任务不再需要时，可以在项目树节点上点击右键，在弹出的选单中选择 Delete 将任务删除。

■ Task 属性修改

鼠标右键点击项目树中 Tasks 节点中的某个任务的节点，在选单中选择 Properties…，如图所示。

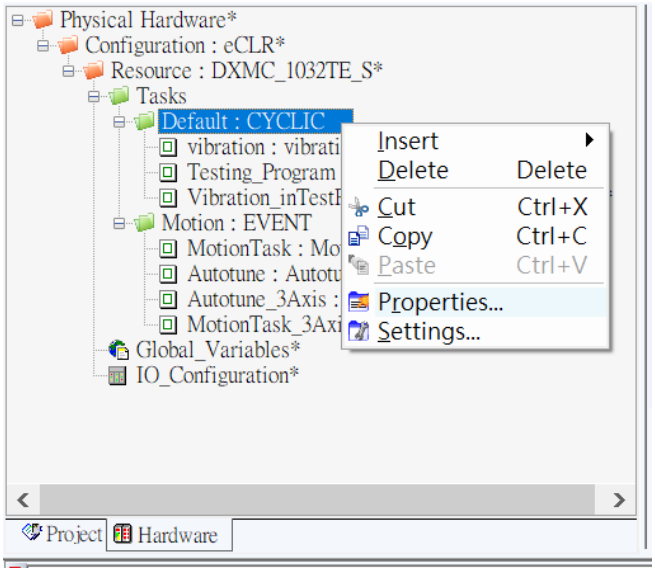


图 6.2.4.3 修改任务属性

在弹出窗口中选择 Type 卷标页，在 Task Type 下拉选单中即可选择任务的类型，如图所示。

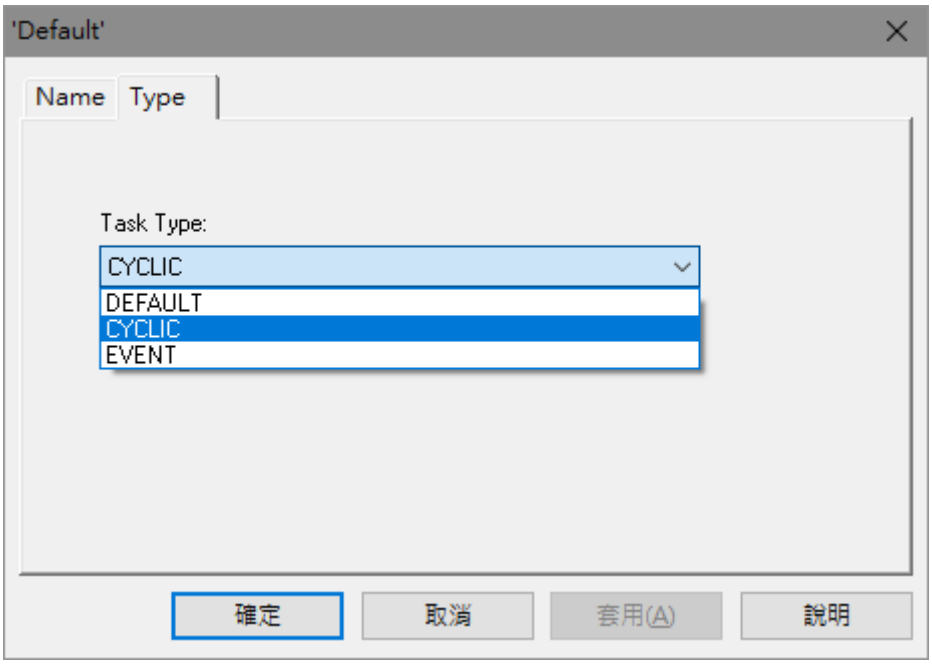


图 6.2.4.4 任务属性窗口

6

不同的任务类型，所需要设置的参数也不同，下面分别进行说明。在需要更改参数的任务节点上点击鼠标右键，在弹出选单中选择 Settings…，如图所示。

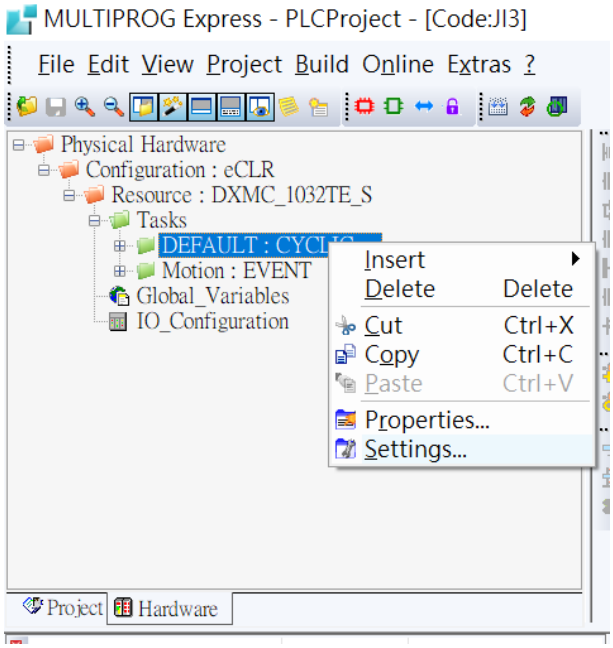


图 6.2.4.5 任务属性路径

Default Task：

图标为 Default 任务的参数设定窗口。Watchdog Time 为监视定时器的计时时间，当勾选了 Enable Watchdog 监视定时器后，监视定时器就会根据设定的 Watchdog Time 开始计时。当计时到达之后，监视定时器就会对运行中的任务发出信号，若运行的任务没有响应，则监视定时器会认为任务运行发生异常，此时就会产生一个运行期异常，对应的 System Task 就会被触发。(System Task 为处理紧急事件，不开放给使用者编辑，在此就不赘述相关设定。)

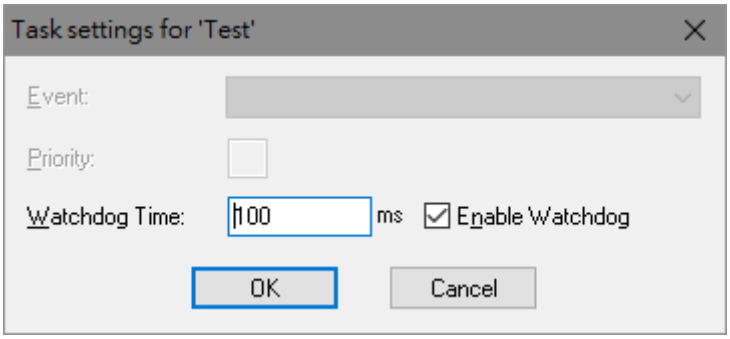


图 6.2.4.6 Default 任务设定

Cyclic Task：

图标为 Cyclic Task 的参数设置窗口，其中需要设置的参数有 3 个：Interval、Priority 和 Watchdog Time。Interval 为任务的运行周期，Priority 为任务的优先级，只是这个优先级是指所有的 Cyclic Task。可以设定的优先权范围是 0 ~ 15，其中优先序数值愈小表示优先权愈高。

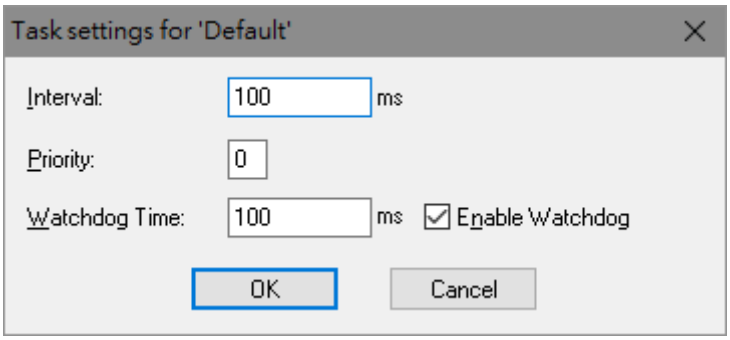


图 6.2.4.7 Cyclic 任务设定

Event 任务：

图标为 Event Task 的参数设定窗口，和 Cyclic Task 不一样，Event Task 主要是设定该任务所对应的事件编号，DXMC 预设 Motion Task 对应的事件编号为 0，由于 DXMC 仅提供一组 Motion Event Task，在此就不赘述相关设定。

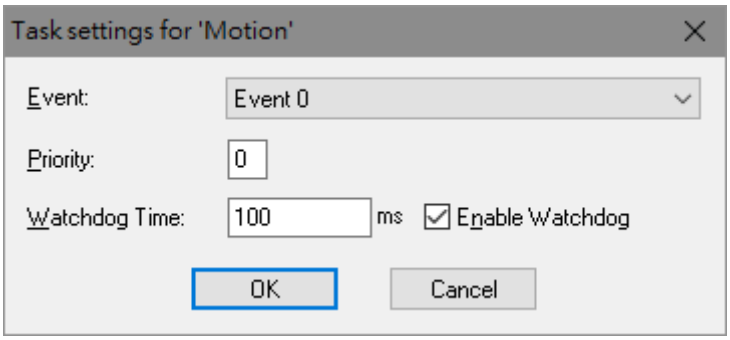


图 6.2.4.8 Event 任务设定

6

6.2.5 POU (程序组织单元)

程序组织单元(Programming Organization Unit，POU)是 IEC61131-3 标准中最基本的程序单元，其中包括了程序(Program)、功能块(Function Block)和功能(Function)三种，各有不同的特性和权限。

程序(Program)：属于撰写程序的主程序，内容可以包含 IO 的使用配置、全局变量和局部变量的定义，是程序组织单元的最大形式。同时程序也是三种 POU 中拥有最大的呼叫权限，可以呼叫功能块(Function Block)与功能(Function)在程序中使用。

功能块(Function Block)：功能块是一种函数，一个功能块拥有专属于自己的内存空间，可以存放变量值。功能块可以呼叫其他功能块及功能来使用。

功能(Function)：功能是只有一个回传值的函数，而且没有自己的内存空间。只能计算某些经过设定的运算，功能只能呼叫功能，而不能呼叫功能块。

■ 增加程序、功能块、功能

鼠标右键点击项目树中 Logical POUs 节点，在选单中选择[Insert] > [Function Block]或是其他想要增加的 POU 类型，如图所示。

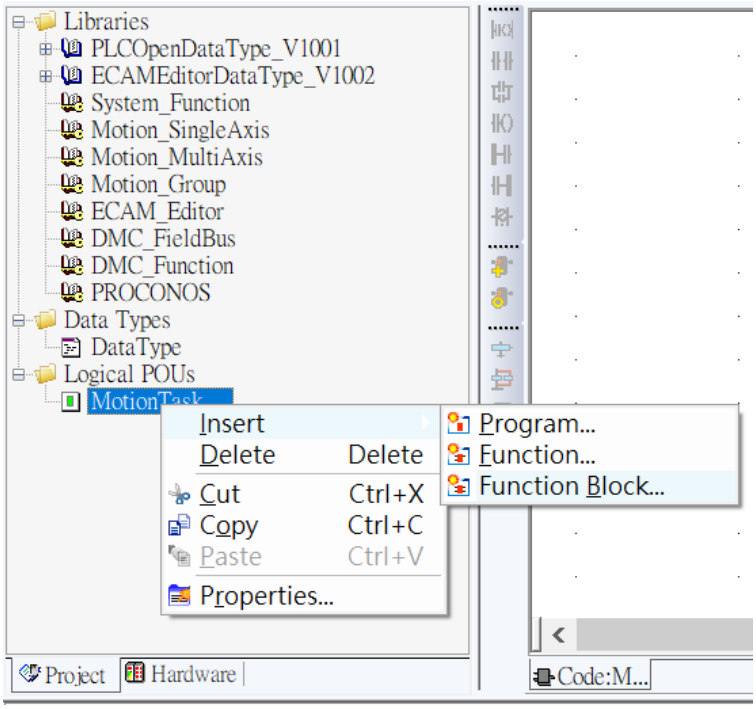


图 6.2.5.1 建立 POU

在弹出的窗口中输入功能块的名称，选择需要的编程语言，点击 **OK**，如下图所示。

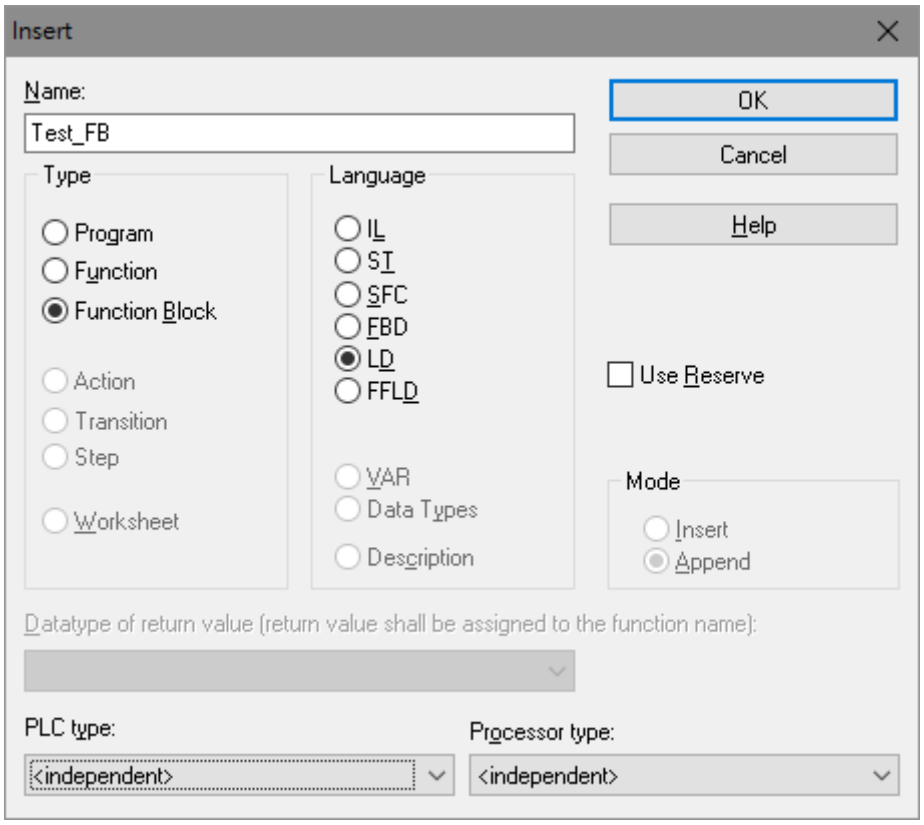


图 6.2.5.1 POU 设定

删除 POU 也是一样，在不需要使用的 POU 节点上点击鼠标右键，在选单中选择 Delete 即可。

6

■ 功能块实例化

在 Logical POUs 中使用功能块的窗体编写完一个函数之后，仅代表宣告了此功能块的定义，并不能直接在控制器中运行。如果要使其能真正运行，就必须要实例化(Instance)。功能块实例化之后就产生这个功能块所需要的专属于内存空间。

也就是说功能块的实例化是通过程序对功能块的呼叫来实现的。在 MULTIPROG 中进行实例化功能块的步骤很简单，以 FBD 语言的程序说明，如图 3.5.3 所示。

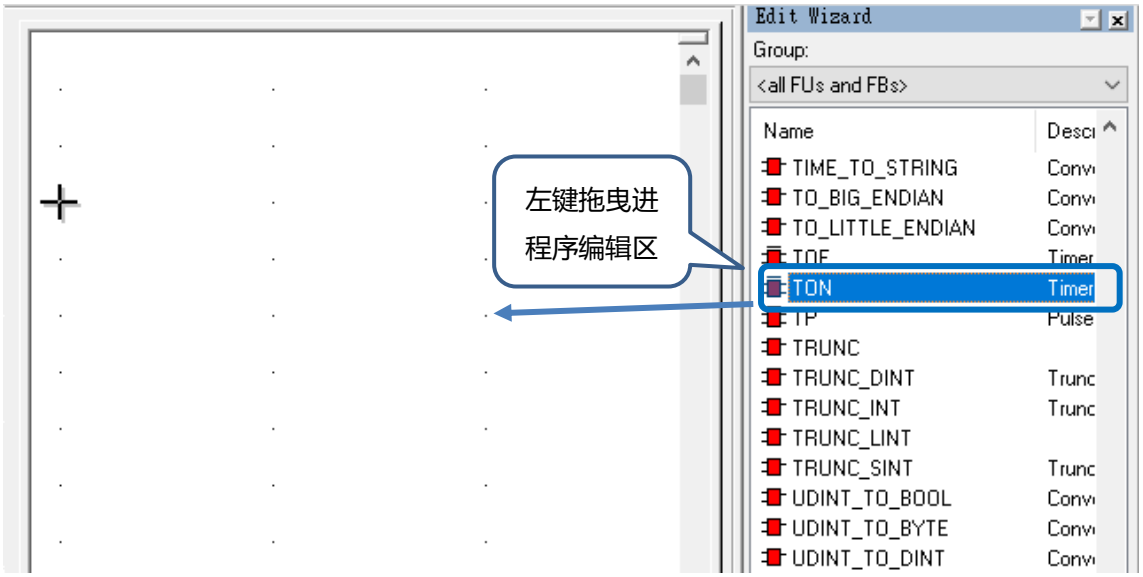


图 6.2.5.3 功能块应用案例

在 MULTIPROG 右侧的 Edit Wizard 窗口中，通过下拉式选单选择 all FUs and FBs，在任意一个功能块图标(以 TON 为例)上按住鼠标左键不放，将图标拖曳至程序编辑区内，然后松开鼠标左键，会弹出属性设定窗口，如下图所示。

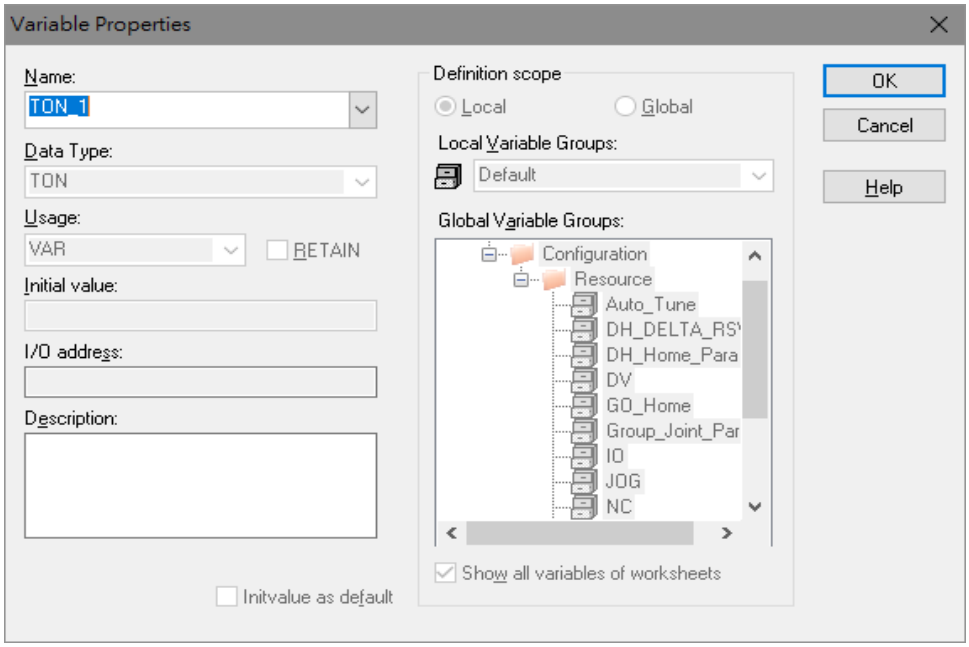


图 6.2.5.4 功能块设定

在 Name 默认名称为功能块名称加上编号，如下图中的 TON_1，功能块实例的名称可以自行定义，不一定要包含有功能块名称，但不可与其他功能块名称相同。

点击窗口中的 **OK** 按钮，在程序的工作单中就出现了功能块的一个实例，如下图所示。在功能块实例 TON_1 图标的顶部可以看到 TON 文字，表示此功能块是通过哪个功能块实例化产生的。

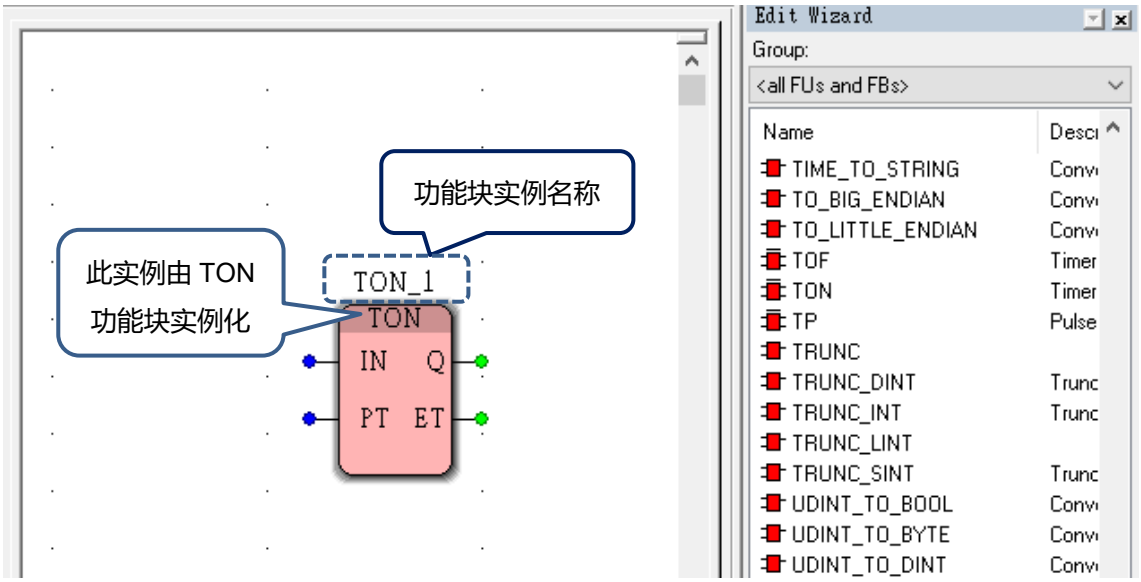


图 6.2.5.5 功能块实际范例

6

■ 程序实例化

和功能块的情形一样，程序仅是宣告了数据的结构和类型，不能在实际的控制器中运行。如果要使其能真正运行，同样地要实例化(Instance)。要使程序能实际运行，同样需要一个实例化的过程。程序的实例化是与任务相关的，MULTIPROG 会根据选择的任务类型来分配其在控制器上所需要的内存空间。

在项目树中右击需要运行程序的任务节点，选择选单中的[Insert] > [Program instance…]，如图所示。

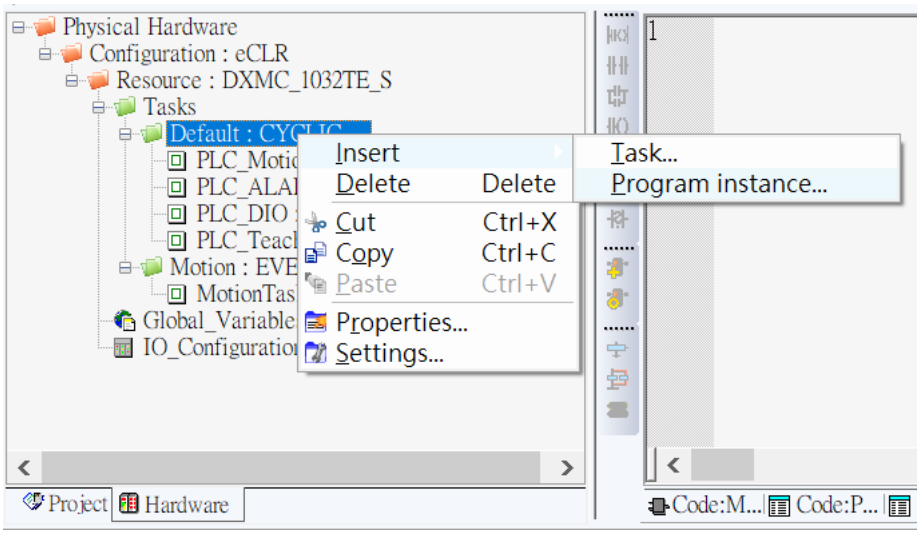


图 6.2.5.6 插入程序实例路径

在弹出窗口中输入程序实例名称(Program instance)，Program Type 选择刚刚新增的程序 Test_PROG，并在右方 Type 字段选择「Program instance」，如下图所示。

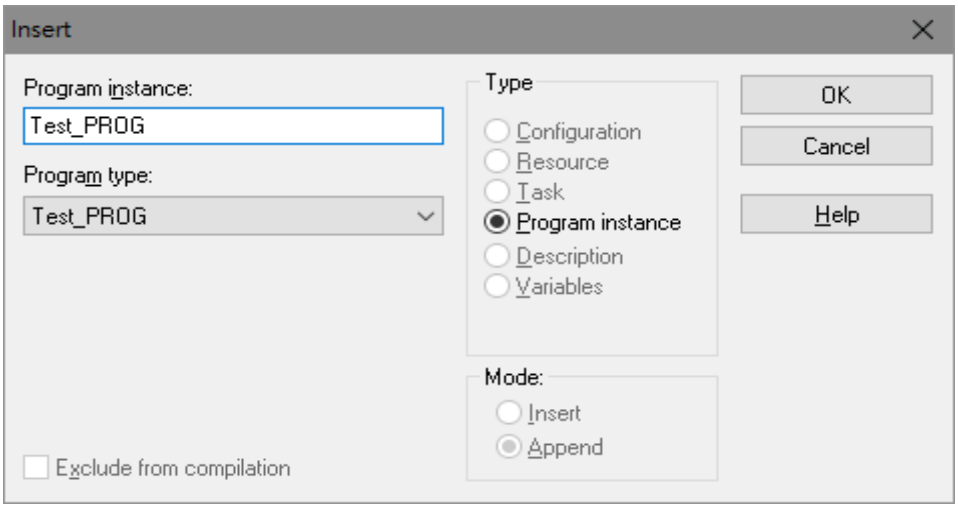


图 6.2.5.7 程序实例设定

点击 **OK** 按钮后，就可以看到在项目树任务节点「Default：CYCLIC」下出现节点「Test_PROG：Test_PROG」，如下图所示。

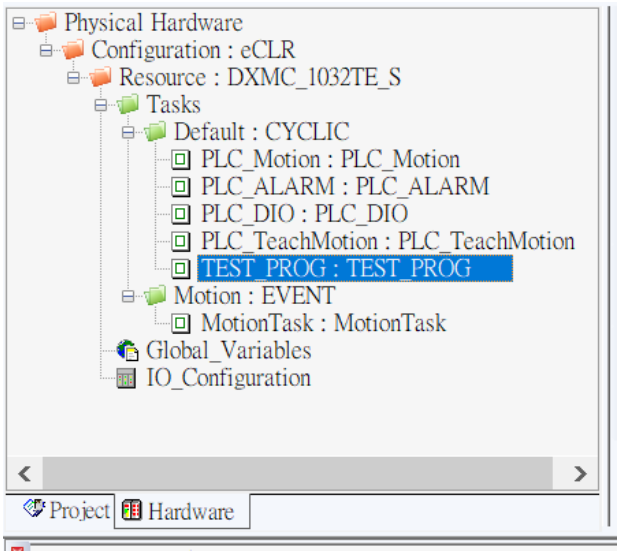


图 6.2.5.8 程序实例范例

注：程序的实例和程序的名称可以重复，但同一个任务中不能有重复的程序实例。

6.2.6 Data Type (数据型态)

6

■ 基本数据型态

基本数据型态是在 IEC61131-3 标准中定义好的标准化的数据类型，包括数据型态、数据长度、数据范围和初始值，详细如下表所示。

表 6.2.6.1 基本数据型态

数据型态	关键词	数据长度(bit)	范围	初始值
布尔	BOOL	1	0和1	0
字节	BYTE	8	0 ~ 0xFF	0
字组	WORD	16	0 ~ 0xFFFF	0
双字组	DWORD	32	0 ~ 0xFFFFFFFF	0
长字组	LWORD	64	0 ~ 0xFFFFFFFFFFFFFFFF	0
短整数	SINT	8	-128 ~ +127	0
整数	INT	16	-32768 ~ +32767	0
双整数	DINT	32	-2^{31} ~ $+2^{31}$-1	0
长整数	LINT	64	-2^{63} ~ $+2^{63}$-1	0
无号短整数	USINT	8	0 ~ +255	0
无号整数	UINT	16	0 ~ +65535	0
无号双整数	UDINT	32	0 ~ $+2^{32}$-1	0
无号长整数	ULINT	64	0 ~ $+2^{64}$-1	0
实数	REAL	32	$1.5e^{-45}$ ~ $3.4e^{38}$	0.0
长实数	LREAL	64	$5.0e^{-324}$ ~ $1.7e^{308}$	0.0
时间	TIME	-	-	T#0s
日期	DAY	-	-	D#0001-01-01
时刻	TOD	-	-	TOD#00:00:00
日期和时刻	DT	-	-	DT#0001-01-01-00:00:00
字符串	STRING	80	-	空字符串

■ 衍生数据型态

衍生数据型态是在基本数据型态的基础上，是由用户自行建立的数据型态，也就是用户自定义的数据型态。

在 MULTIPROG 中插入用户自定义的数据型态步骤如下：

项目树中右击 Data Types 数据型态，选择选单的 [Insert] > [Datatypes]，如图所示。

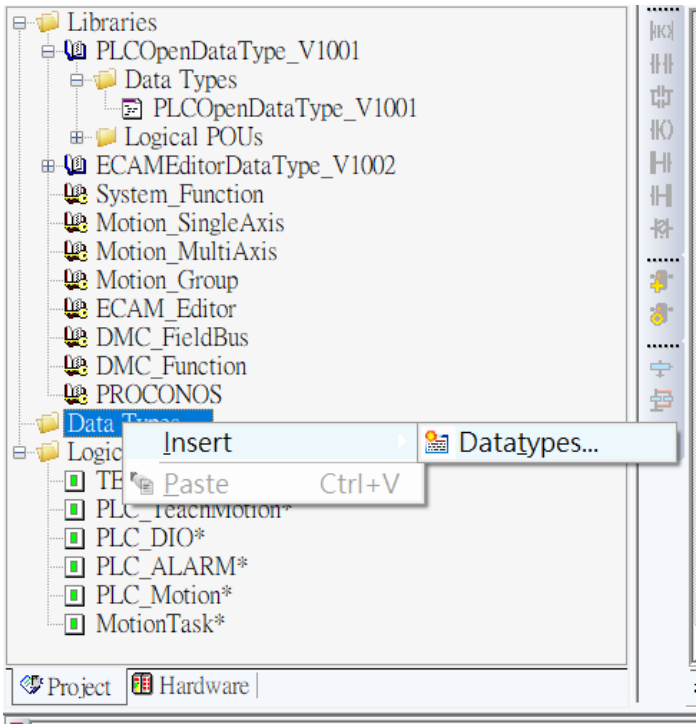


图 6.2.6.1 数据型态插入路径

6

在弹出的窗口中输入名称，点击确定 **OK**，如图所示。

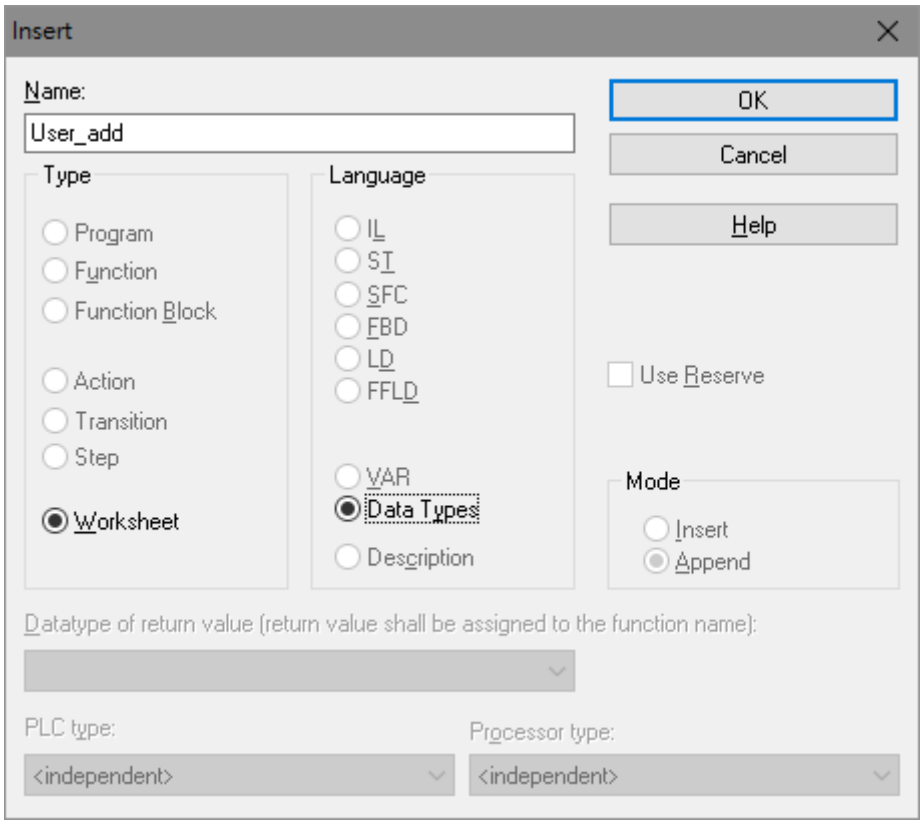


图 6.2.6.2 数据型态设定

注：此处的名称(Name)只是数据型态工作单的名称，与数据型态没有关系。

在数据型态工作单中使用描述式语言进行编辑，依照 IEC61131-3 标准的规定，所有的衍生数据型态的定义都使用「TYPE」关键词开头，以「END_TYPE」关键词结束。MULTIPROG 支持 4 种基本的用户自定义数据型态，分别是列举、范围、数组和结构，其对应的范例如图 6.2.6.3 所示。


```
TYPE
    TEST_Enum : (Value1, Value2 , Value3);
END_TYPE


TYPE
    TEST_Range : INT(-18..100);
END_TYPE


TYPE
    TEST_Array : ARRAY [1..4] OF LREAL;
END_TYPE

TYPE
    ROBOT_POSTURE :
    STRUCT
        Shoulder    : BOOL;
        Elbow       : BOOL;
        Flip        : BOOL;
        PS          : BOOL;
        UF          : USINT;
        TF          : USINT;
        RSVD1       : USINT;
        RSVD2       : USINT;
    END_STRUCT;
END_TYPE
```

User_add

图 6.2.6.3 数据型态范例

■ Initialize Multi-Element Variable Window

宣告变量时候，可以赋予全局变量初始值，对于像基本数据型态(Data Type)，只需要在“init”字段输入初始值即可，但对于较复杂的自定义数据型态 (详细参阅 DXMC 手册 6.5.9.1)，如 S_CAM_BASIC_POINT、S_CAM_POINT_ARR 等，欲赋予初始值就需要透过 Initalize Multi_Element Varible Window。

6

使用自定义数据型态数据 S_CAM_BASIC_POINT 时，在数据型态工作单中使用描述式语言进行编辑，可看到 S_CAM_BASIC_POINT 中包含数个数据型态数据，MasterPos、SlavePos、SlaveVel、SlaveAcc、SlaveJerk。

```
TYPE
    S_CAM_POINT_RSV_1 : ARRAY [1..3] OF UINT;
    S_CAM_POINT_RSV_2 : ARRAY [1..4] OF REAL;

    S_CAM_BASIC_POINT:
    STRUCT
        FuncType        : UINT;                         (*(1)straight line
                                                          (2)parabola
                                                          (3)3rd degree polynomial
                                                          (4)3rd degree polynomial(velocity adjustalbe)
                                                          (5)5th degree polynomial
                                                          (6)5rd degree polynomial(velocity adjustalbe)
                                                          (7)simple harmonic
                                                          (8)7th degree polynomial
                                                          (9)7rd degree polynomial(velocity adjustalbe)
                                                          *)
        Rsv1            : S_CAM_POINT_RSV_1;
        MasterPos       : LREAL;
        SlavePos        : LREAL;
        SlaveVel        : LREAL;
        SlaveAcc        : LREAL;
        SlaveJerk       : LREAL;
        Rsv2            : S_CAM_POINT_RSV_2;
    END_STRUCT;
END_TYPE

TYPE
    S_CAM_POINT_ARR: ARRAY [1..32767] OF S_CAM_BASIC_POINT;
    S_CAM_POINT_ARR_10: ARRAY [1..10] OF S_CAM_BASIC_POINT;
    S_CAM_POINT_ARR_100: ARRAY [1..100] OF S_CAM_BASIC_POINT;
    S_CAM_POINT_ARR_1000: ARRAY [1..1000] OF S_CAM_BASIC_POINT;
    S_CAM_POINT_ARR_10000: ARRAY [1..10000] OF S_CAM_BASIC_POINT;
END_TYPE
```

图 6.2.6.4 数据型态范例

进入【View】→【Initialize Multi-Element Varible Window】。

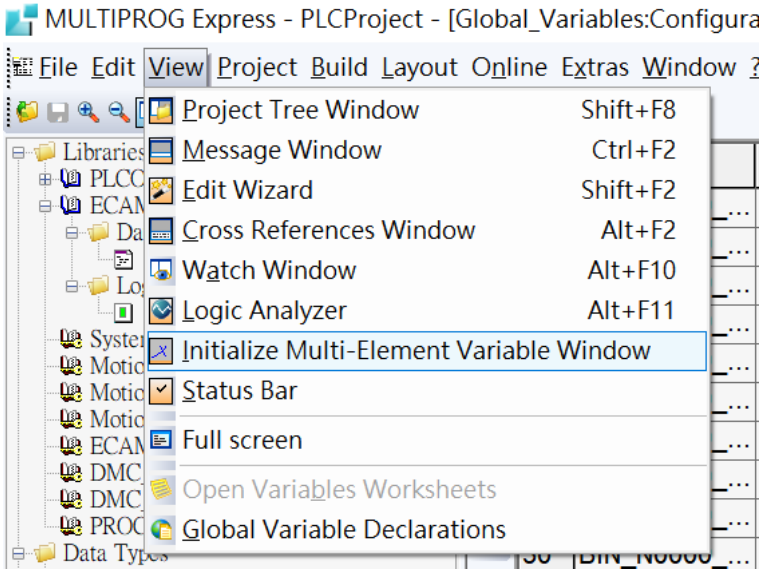
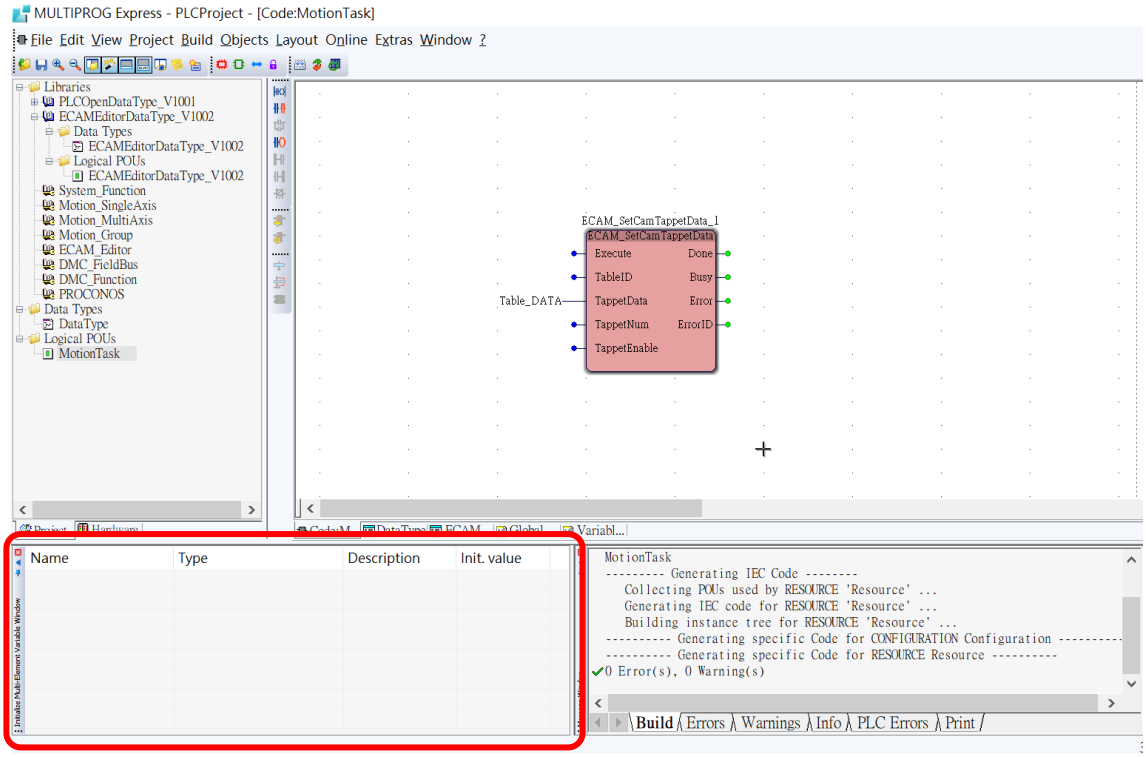


图 3.2.6.5 Initialize Multi-Element Varible Window

【Initialize Multi-Element Varible Window】将出现在讯息窗口中。



按鼠标左键功能块引脚上的变量，窗口将显示该变量数据内容，可透过 init.value 字段中赋予所有数据初始值。若变量为全局变量，则需先进入 Global Variable Declarations 中，点击该变量才能对初始值进行编辑。

Name	Type	Description	Init. value
Table_DATA	S_CAM_BASIC_POINT		(MasterPos := 200.0, SlavePos
FuncType	UINT	(1)straight line ...	
Rsv1	S_CAM_POINT_RSV_1		
MasterPos	LREAL		200.0
SlavePos	LREAL		700.0
SlaveVel	LREAL		500.0
SlaveAcc	LREAL		500.0
SlaveJerk	LREAL		500.0
Rsv2	S_CAM_POINT_RSV_2		

6

6.3 MULTIPROG 编程语言

MULTIPROG 系统支持所有符合 IEC61131-3 标准的 PLC 编程语言，此章节透过范例方式分别针对此五种标准的 PLC 编程语言进行说明，另外 MULTIPROG 系统还可以支持混合编程(如：功能块图与梯形图混合使用)，使用者可以根据使用需求或习惯来混合适合的编程语言。

6.3.1 指令表(IL)编程语言

指令表编程语言是 PLC 程序设计最基本的语言；具有容易储存、便于操作的特点，同时其占用系统资源也比较少，但同时可读性较差，比较适合在功能不复杂的小型控制系统中使用。

范例：实现四则运算 A+B×(C+D)。

1. 在项目树 POU 节点中插入功能块(Function Block)，名称为 IL_Example，Language 选择为 IL。

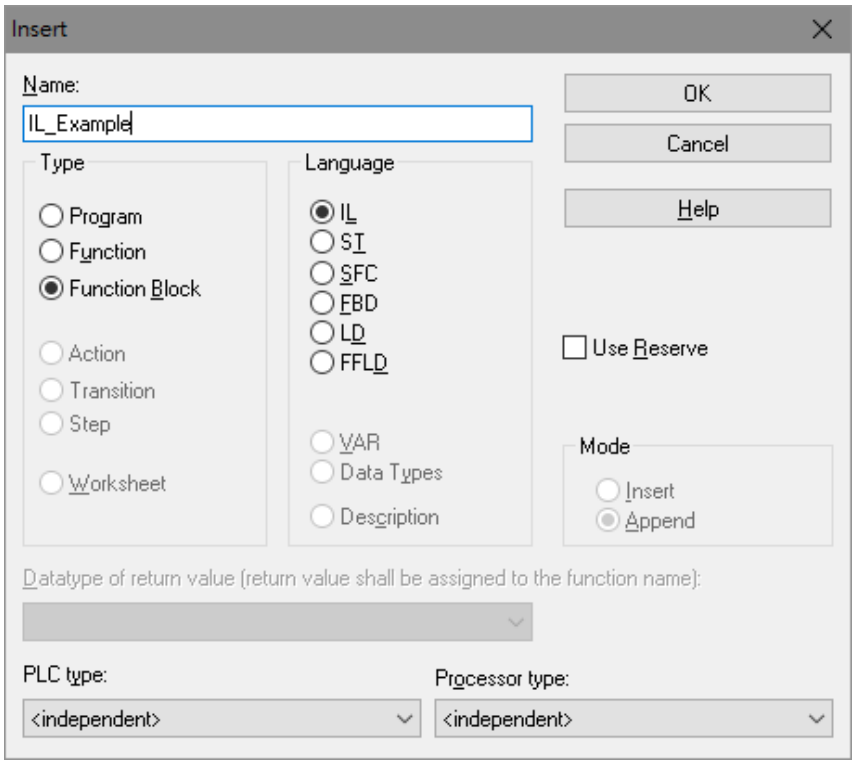


图 6.3.1.1 新增 IL 范例

2. 按下 **OK** 后，在其中编写如图所示程序。

```
1    LD D
2    ADD C
3    MUL B
4    ADD A
5    ST X
```

图 6.3.1.2 IL 程序范例

3. 在程序编辑区中，点击鼠标右键，在选单中选择 Open Variables Worksheets 后，在其中定义如图所示的变量。

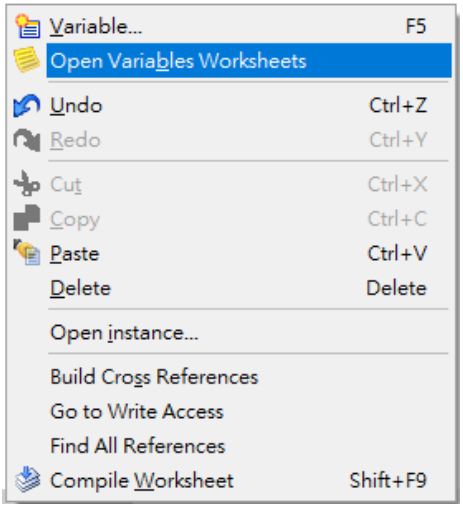


图 6.3.1.3 新增变量路径

	Name	Type	Usage	Description
1	⊟ **Default**			
2	A	INT	VAR_INPUT	
3	B	INT	VAR_INPUT	
4	C	INT	VAR_INPUT	
5	D	INT	VAR_INPUT	
6	X	INT	VAR_OUTPUT	

图 6.3.1.4 变量宣告

4. 点击常用工具栏中的编译(Make)按钮图标或是功能区 Build 选单中的 Make 功能。
5. 在程序(Program)中可以透过拖曳 IL_Example 功能块来实例化此功能块，如下图所示。

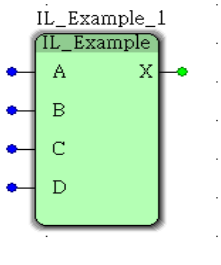


图 6.3.1.5 IL 范例功能块

6

6.3.2 梯形图(LD)编程语言

梯形图是 PLC 使用得最多的图形编程语言，被称为 PLC 的第一编程语言。根据梯形图中各接点的状态和逻辑关系，从左到右、自上而下的顺序排列。

范例：伺服的启动、保持和停止控制

1. 在项目树 POU 节点中插入功能块(Function Block)，名称为 LD_Example，Language 选择为 LD。

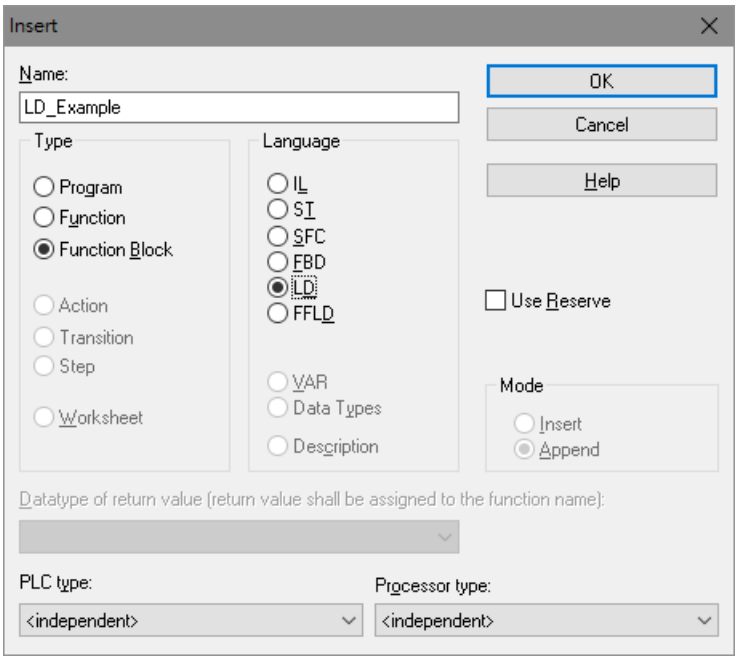


图 6.3.2.1 新增 LD 范例

2. 按下 **OK** 后，在其中工作单中，点击鼠标右键，在选单中选择 Contact Network，如图所示程序。

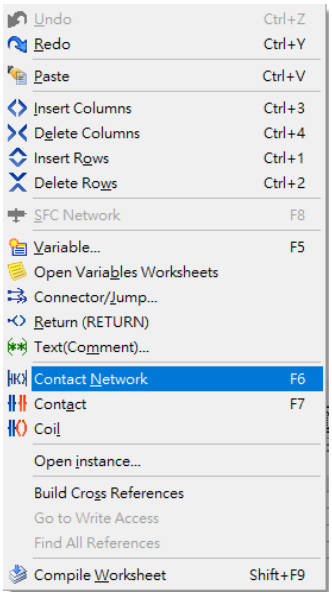


图 6.3.2.2 新增 LD 组件路径

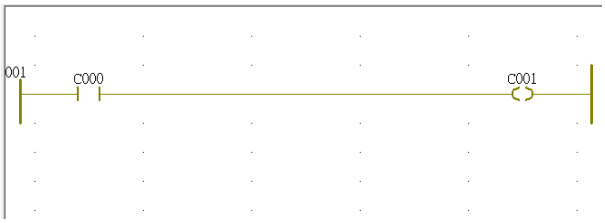


图 6.3.2.3 LD 范例

3. 鼠标双击 C000 接点，在弹出窗口中将其名称修改为 Start，Usage 选择为 VAR_INPUT，如图所示，然后点击 **OK**

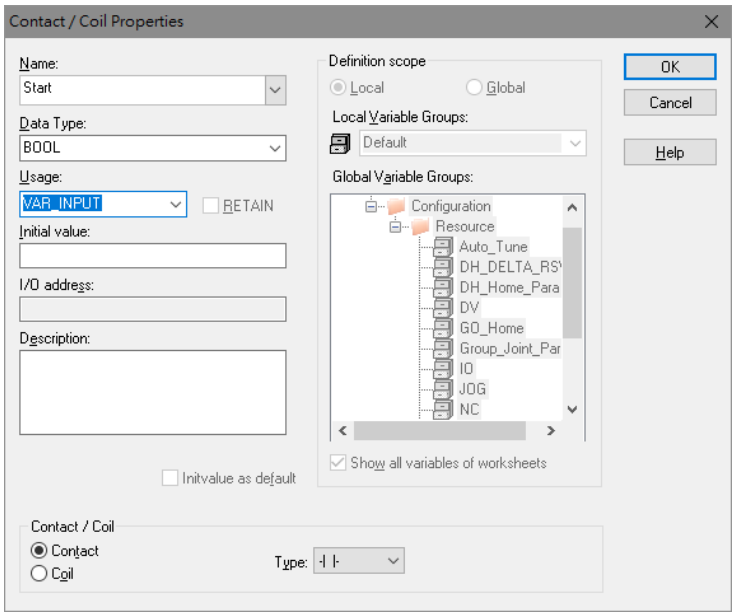


图 6.3.2.4 设定接点变量

4. 鼠标双击 C001 线圈，在弹出窗口中将其名称修改为 Run，Usage 选择为 VAR_OUTPUT，如图所示，然后点击 **OK。**

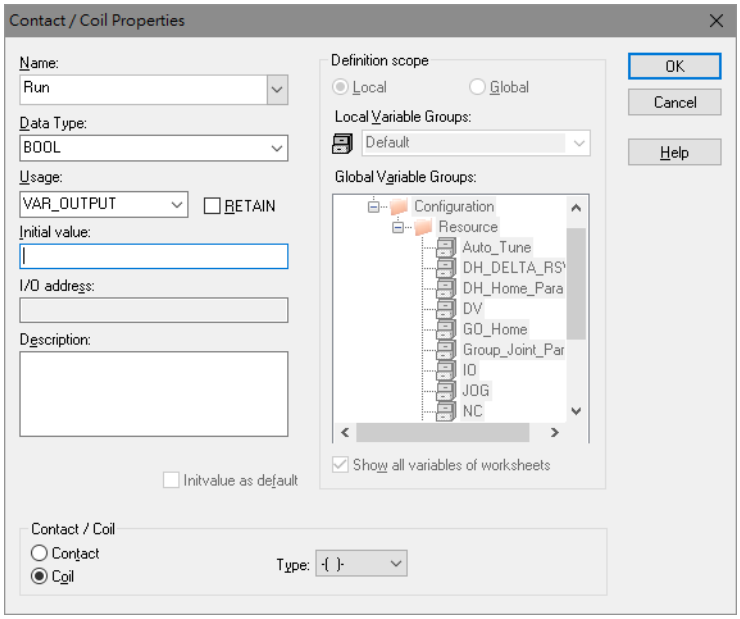


图 6.3.2.5 设定接点变量

6

5. 点击工作单中的 Start 接点，然后再点击鼠标右键选择 Contact Below，在其下面插入一个平行的接点 C002，并将该接点改名为 RUN，如图所示。

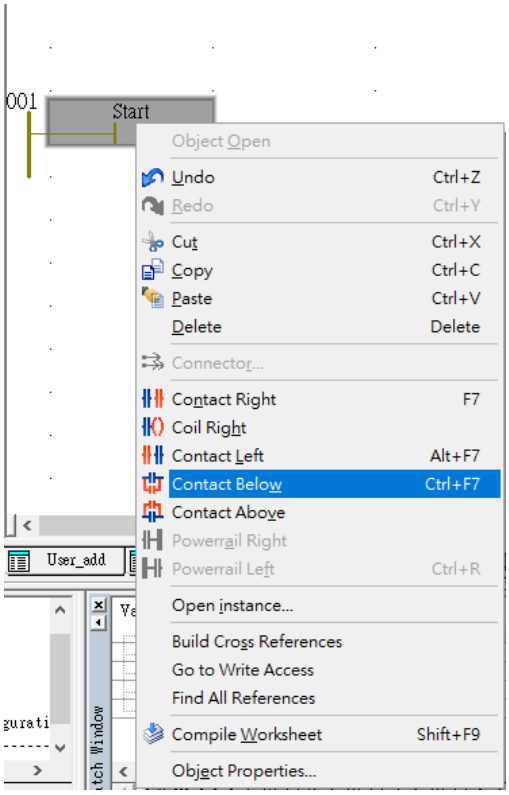


图 6.3.2.6 新增 LD 组件路径

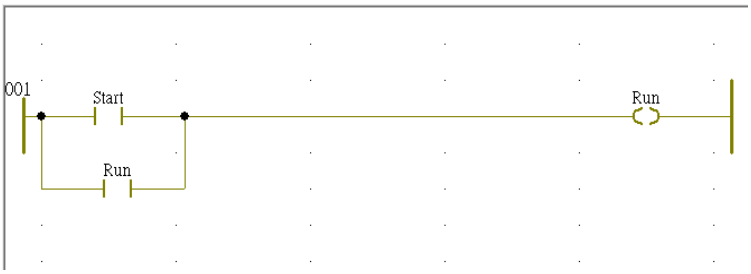


图 6.3.2.7 LD 范例

6. 点击 RUN 线圈，然后再点击鼠标右键选择 Contact Left，在左边插入一个接点 C003，如图所示。

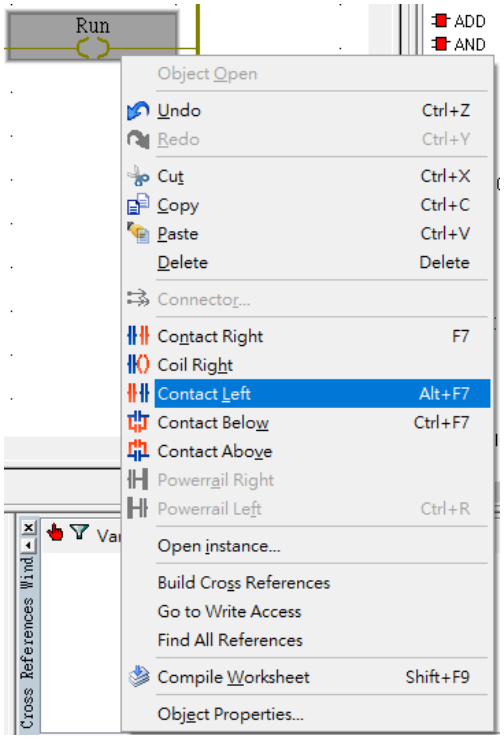


图 6.3.2.8 新增 LD 组件路径

7. 双击接点 C003，并在弹出的窗口中将其名称改为 Stop，Usage 选择为 VAR_INPUT，并在窗口下方选择 Type 为常闭接点，如图所示。

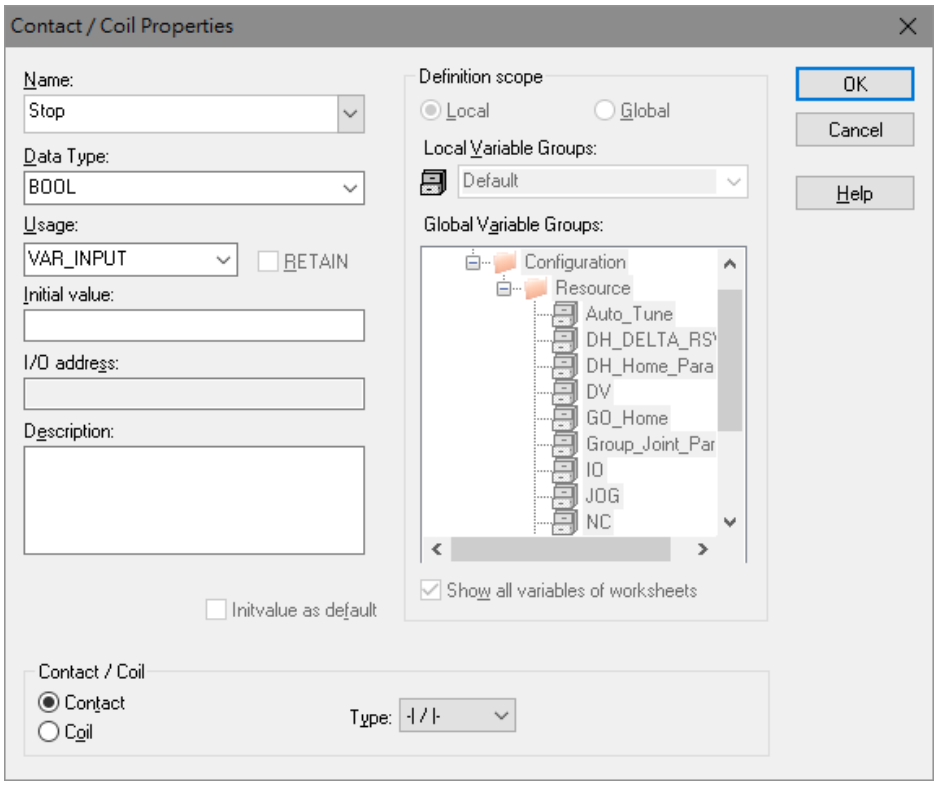


图 6.3.2.9 设定接点变量

8. 点击常用工具栏中的编译(Make)按钮图标或是功能区 Build 选单中的 Make 功能。
9. 在程序(Program)中可以透过拖曳 LD_Example 功能块来实例化此功能块，如图所示。

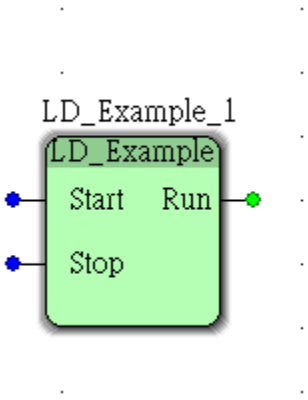


图 6.3.2.10 LD 范例功能块

6.3.3 功能块图(FBD)编程语言

功能块图(Function Block Diagram，简称 FBD)是一种图形化的编程语言，实际上是用代表逻辑或算数符号的功能块所组成，各方块图的输入或输出是由方块图之间的连接线来连接，类似绘制电路图的方式来进行编程。

范例：方波产生器，负载周期(Duty Cycle) 50%，周期可根据输入参数调整。

1. 在项目树 POU 节点中插入功能块(Function Block)，名称为 FBD_Example，Language 选择为 FBD。

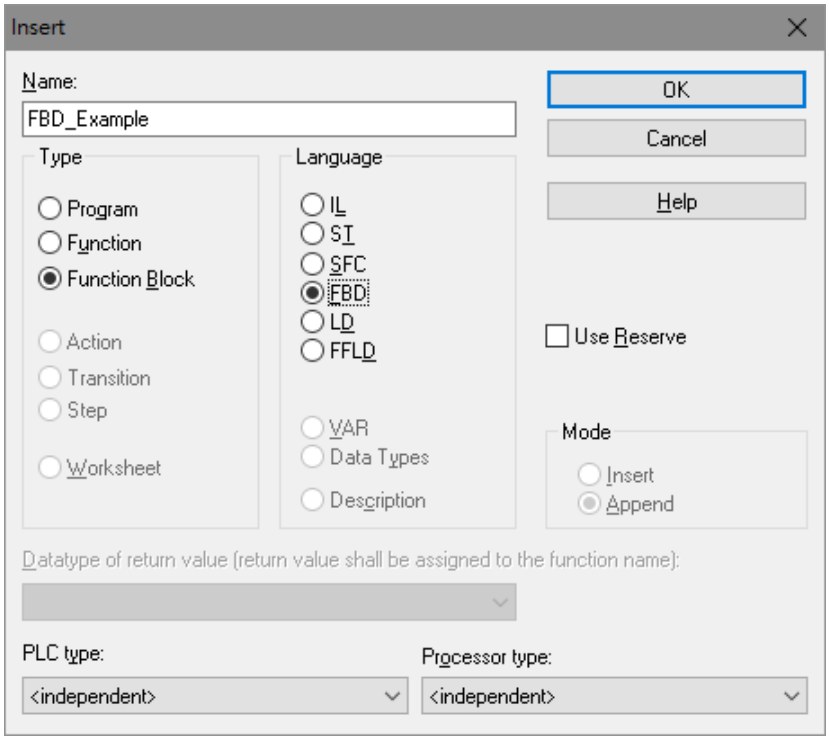


图 6.3.3.1 新增 FBD 范例

6

2. 按下 **OK** 后，在其中工作单中，从右侧的 Edit Wizard 窗口中选择 TON 功能块，并把 TON 功能块拖曳至其工作单，在弹跳的窗口中将其命名为 TON_Front，点击 **OK**。使用同样的方法实例化第二个 TON 功能块，取名为 TON_Back。

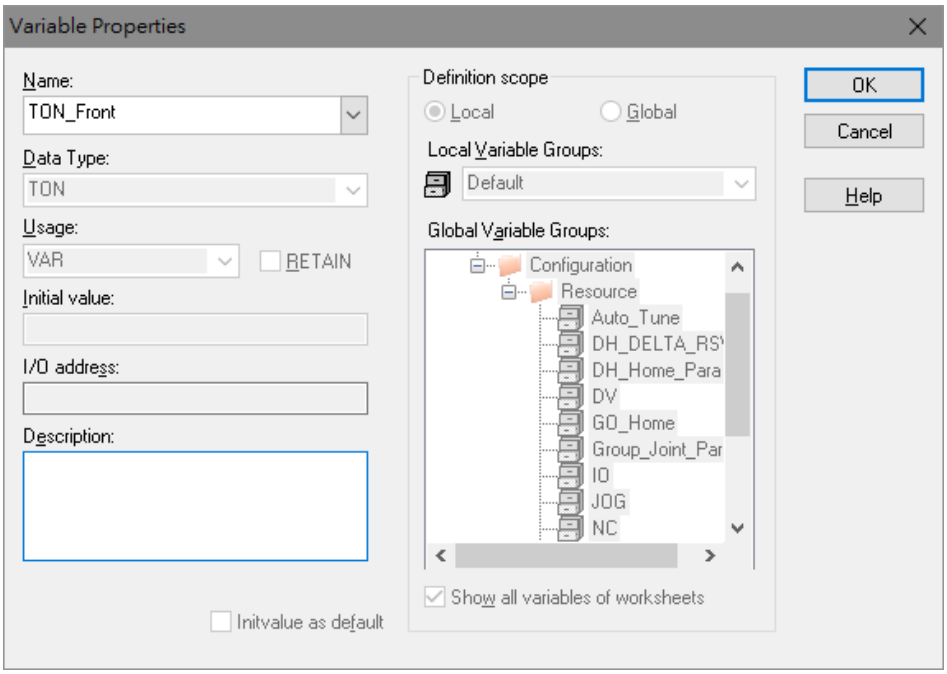


图 6.3.3.2 FBD 接点设定

3. 将鼠标移至 TON_Back 的 Q 输出端，如图所示。此时按住鼠标左键不放，将鼠标移动到 TON_Front 的 IN 输入端，等待连接线变为绿色后，松开鼠标左键。

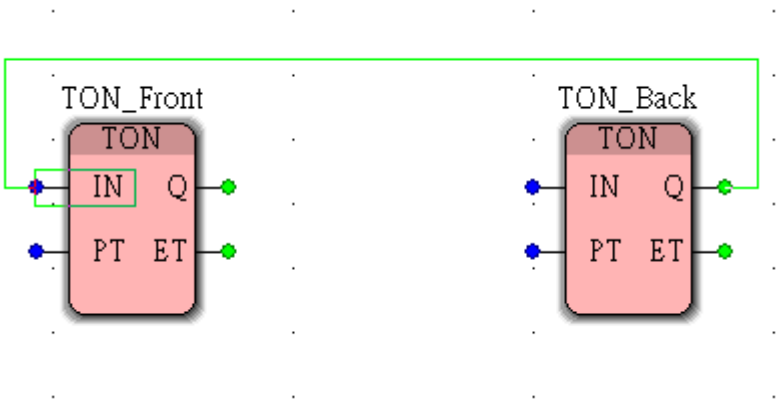


图 6.3.3.3 FBD 范例

4. 同步骤 3，连接 TON_Front 的 Q 和 TON_Back 的 IN。

5. 双击 TON_Front 功能块，在弹出窗口中的 Formal Parameters 中 IN 选项勾选 Negated，如图所示。点击 **OK**，此时工作单中的 TON_Front 的 IN 输入端会增加了一个小圆圈，如图所示。

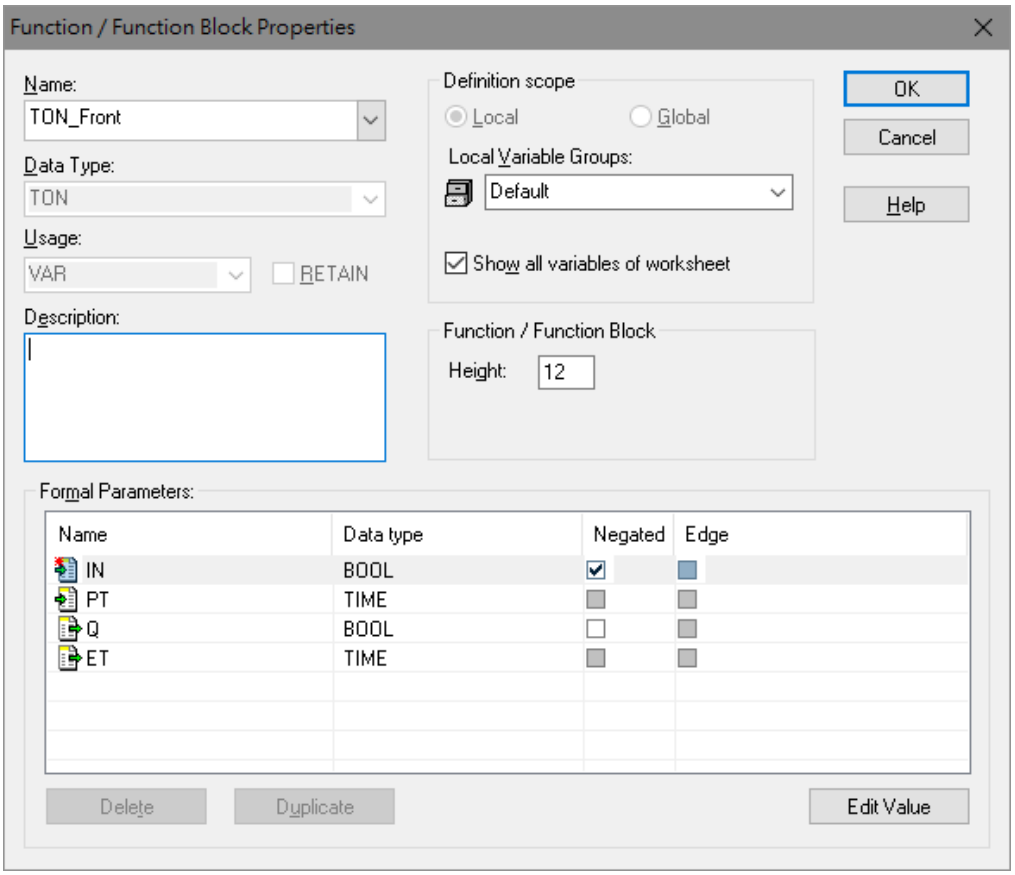


图 6.3.3.4 FBD 接点设定

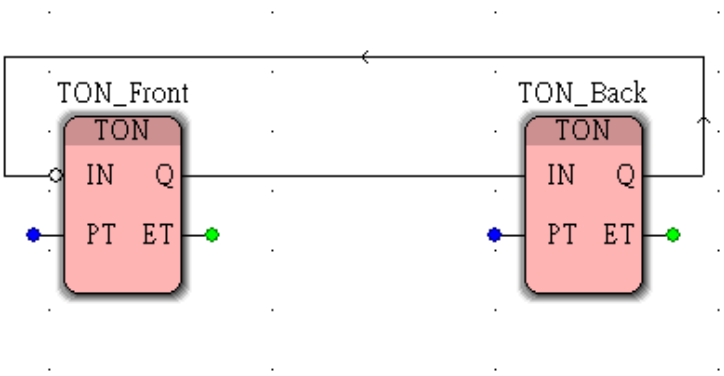


图 6.3.3.5 FBD 范例

6

6. 在 TON_Front 的输入端 PT 上点击鼠标右键，选择选单中的 Variable，如图所示。在弹出的窗口中输入名称 Cycle_Time，Usage 选择 VAR_INPUT，并点击 **OK**。

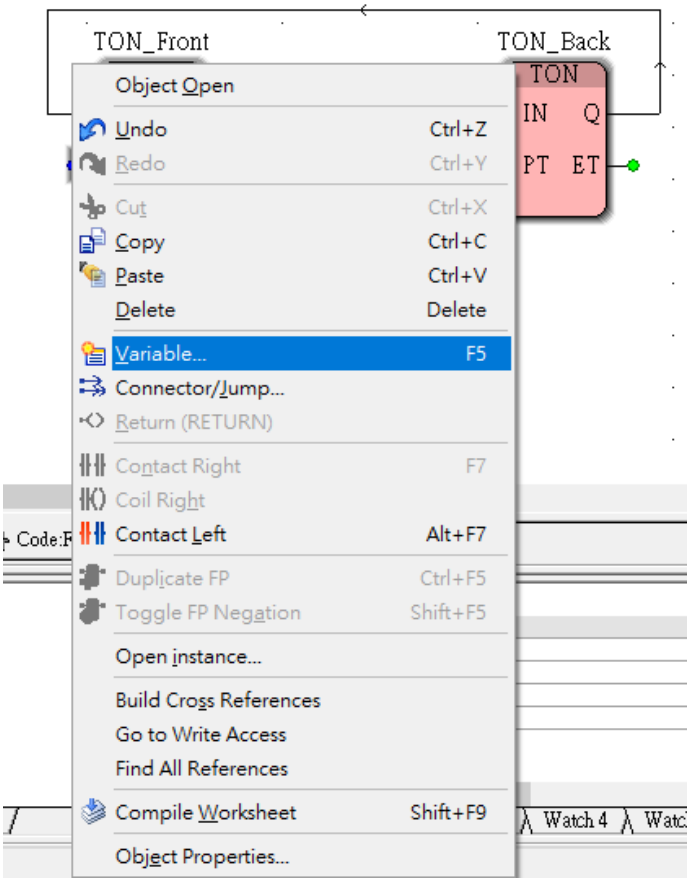


图 6.3.3.6 变量设定路径

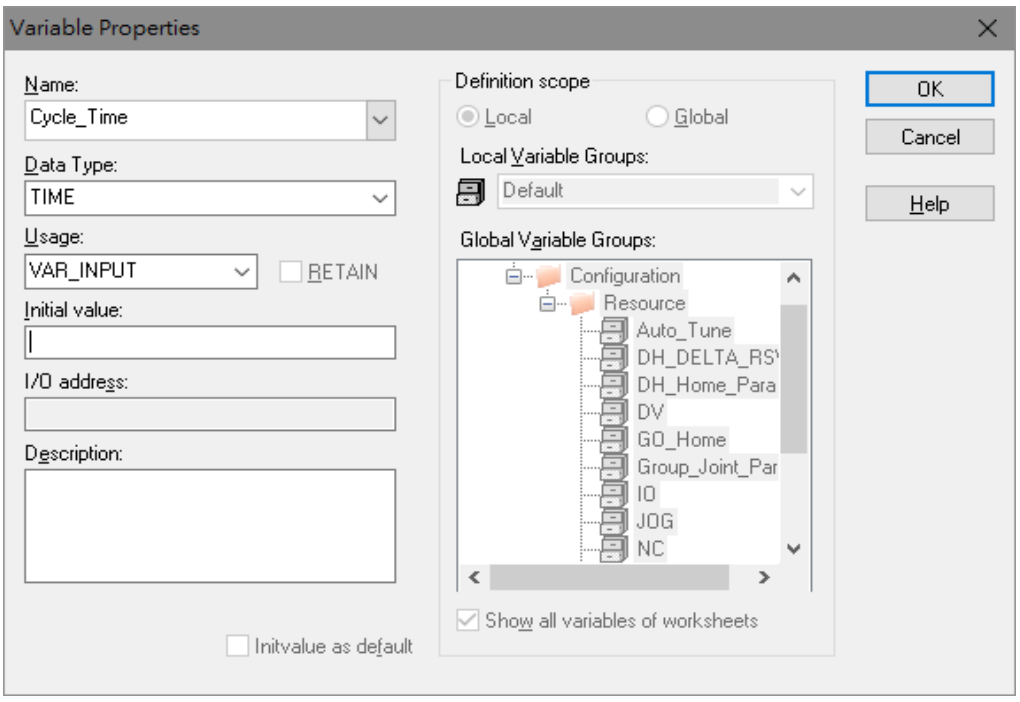


图 6.3.3.7 变量设定

7. 连接 TON_Back 的 PT 输入端和变量 Cycle_Time，如图 4.3.8 所示。

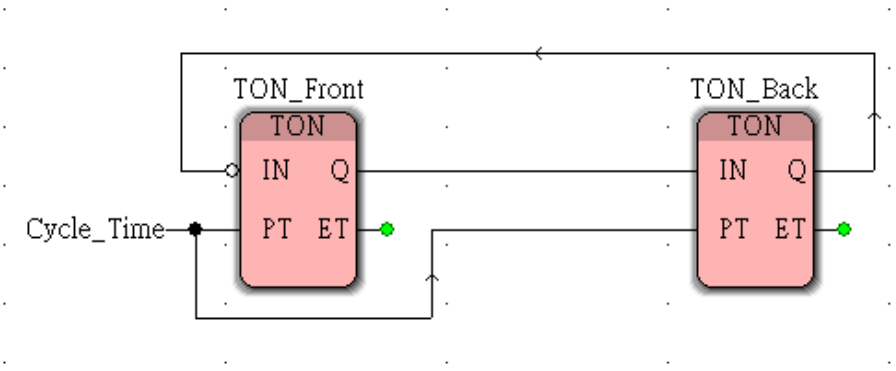


图 6.3.3.8 FBD 范例

8. 在空白处点选鼠标右键，在选单中选择 Variable，然后插入一个变量 WaveOut，Data Type 选择为 BOOL，Usage 为 VAR_OUTPUT。
9. 连接 WaveOut 到 TON_Front 的输出端 Q 和 TON_Back 的 IN 的连接在线，如图所示。

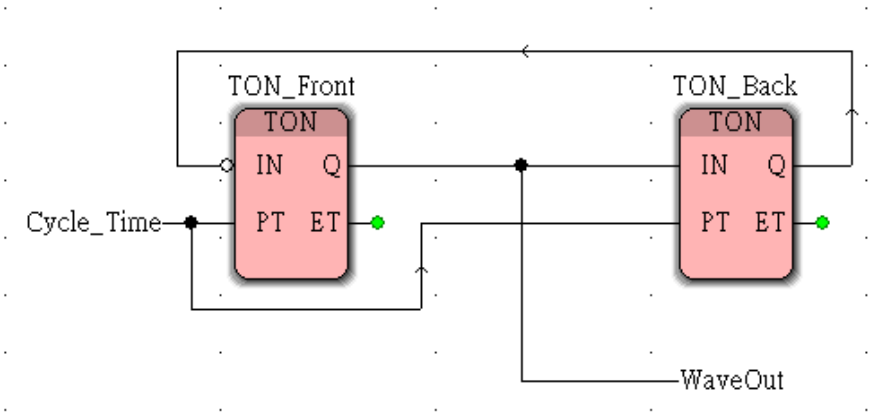


图 6.3.3.9 FBD 范例

10. 点击常用工具栏中的编译(Make)按钮图标，或是功能区 Build 选单中的 Make 功能。
11. 在程序(Program)中，可以透过拖曳 FBD_Example 功能块来实例化此功能块，如图所示。

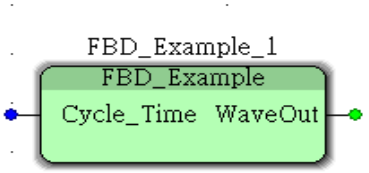


图 6.3.3.10 FBD 范例功能块

6.3.4 结构化文字(ST)编程语言

6

结构化文字(Structured Text,简称ST)是一种支持块状结构(Block Structured)的高级语言，以 Pascal 为基础，语法也类似 Pascal。具有和其他高阶编程语言相同的程序流程控制能力，编写的代码结构清晰，适合用于解决复杂的控制问题，结构式文件编程语言支持复杂的叙述及巢状指令。

范例：求自然数的累加和阶乘。

1. 在项目树 POU 节点中插入功能块(Function Block)，名称为 ST_Example，Language 选择为 ST。

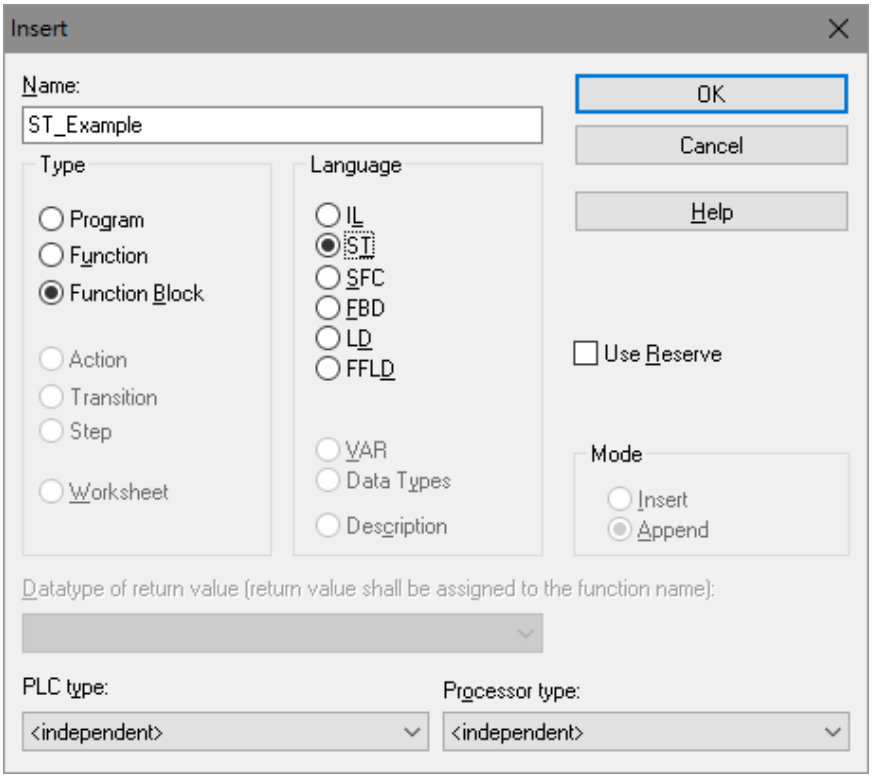


图 6.3.4.1 新增 ST 范例

2. 按下 **OK** 后，在其中工作单中，点击鼠标右键，在选单中选择 Open Variables Worksheets，并输入如图所示的变量。

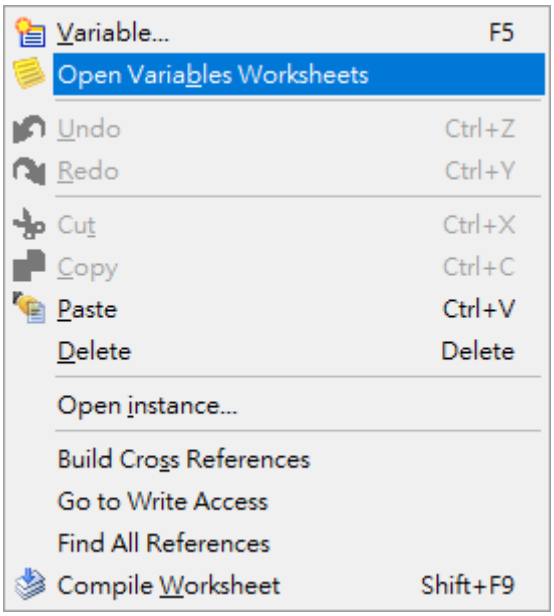


图 6.3.4.2 变量设定路径

	Name	Type	Usage	Description
1	⊟ **Default**			
2	Sum	INT	VAR_OUTPUT	
3	Factorial	INT	VAR_OUTPUT	
4	Begin	INT	VAR_INPUT	
5	End	INT	VAR_INPUT	

图 6.3.4.3 变量设定

3. 在工作单输入如图所示之程序内容。

```
1    Sum := 0;
2    Factorial := 1;
3    FOR I:= Begin TO End BY 1 DO
4        Sum := Sum + I;
5        Factorial:= Factorial * I;
6    END_FOR;
7
8
```

图 6.3.4.4 ST 范例

4. 点击常用工具栏中的编译(Make)按钮图标或是功能区 Build 选单中的 Make 功能。

6

5. 在程序(Program)中可以透过拖曳 ST_Example 功能块来实例化此功能块，如下图所示。

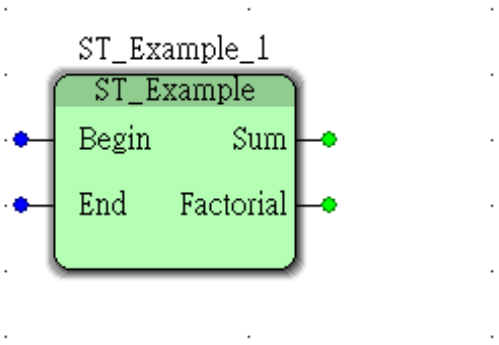


图 6.3.4.5 ST 范例功能块

6.3.5 顺序功能图(SFC)编程语言

顺序功能图(Sequence Function Chart，简称 SFC)，也叫状态转移图，是一种描述顺序控制系统控制条件和过程的方法。在顺序功能图中，将系统的工作过程分为多个阶段，每个阶段称为一个「步」(Step)，每一步中可以完成一个或者多个特定的动作，同时也采用了文字叙述和图形符号相结合的方式。SFC 程序代码占用的存储空间较多，运行期间所需的资源也比较多，不适合解决简单的问题。

范例：产线流程控制程序

产在线有一个控制开关，当开关启动后，进入自动切换状态。自动状态下，首先工序 1，Y1 动作，接着自动切换，工序 2，Y2 动作，接着自动切换，工序 3，延时计数，在三个工序结束后，用重复的方式，直接返回工序 1，形成序列的循环。

1. 在项目树 POU 节点中插入功能块(Function Block)，名称为 SFC_Example，Language 选择为 SFC。

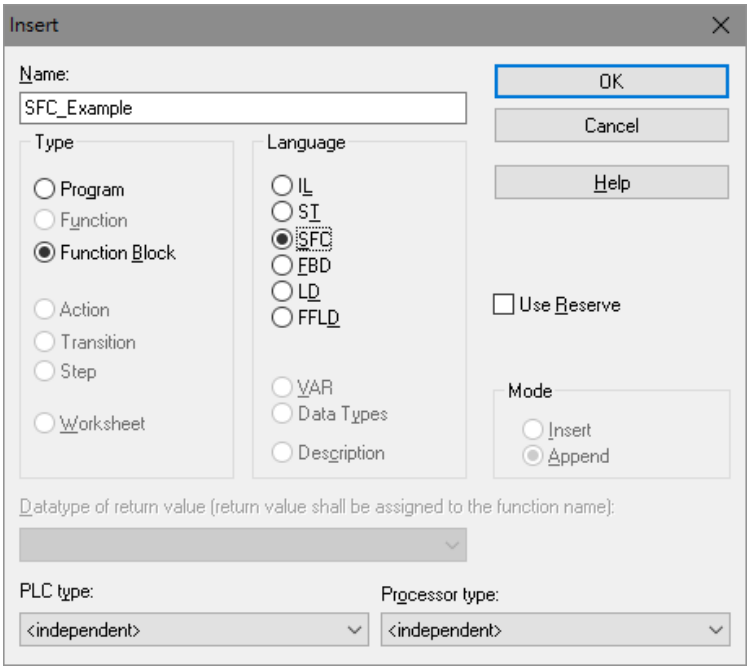


图 6.3.5.1 新增 SFC 范例

2. 按下 **OK** 后，在其中工作单中，点击鼠标右键，在选单中选择 Open Variables Worksheets，并输入如图所示的变量。

	Name	Type	Usage	Description
1	⊟ **Default**			
2	Start	BOOL	VAR_INPUT	
3	Process_1	BOOL	VAR_OUTPUT	
4	Process_2	BOOL	VAR_OUTPUT	
5	Y1_forward	BOOL	VAR_OUTPUT	
6	Y2_Back	BOOL	VAR_OUTPUT	
7	Delay_03	BOOL	VAR_OUTPUT	

图 6.3.5.2 变量设定

6

3. 返回工作单，点击鼠标右键，在选单中选择 SFC Network，如图所示。

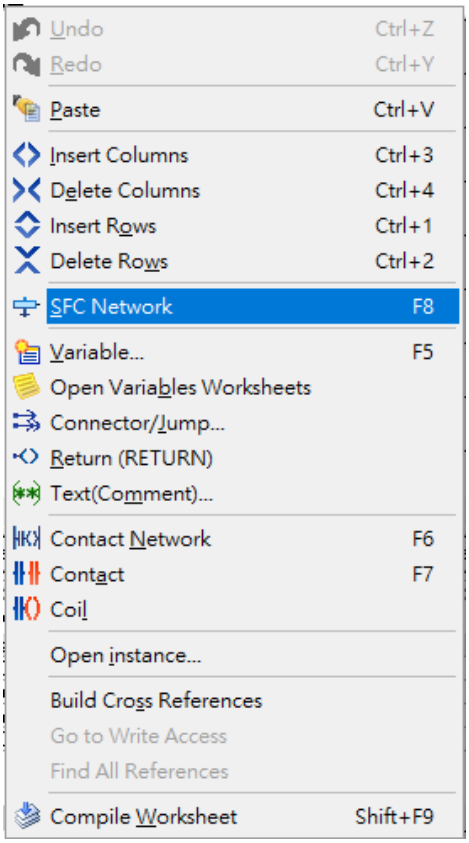


图 6.3.5.3 SFC 新增组件路径

4. 建立 SFC Network 之后，鼠标左键点击 S001 节点，然后点击鼠标右键，在选单中可以看到三个与 SFC 相关的指令。

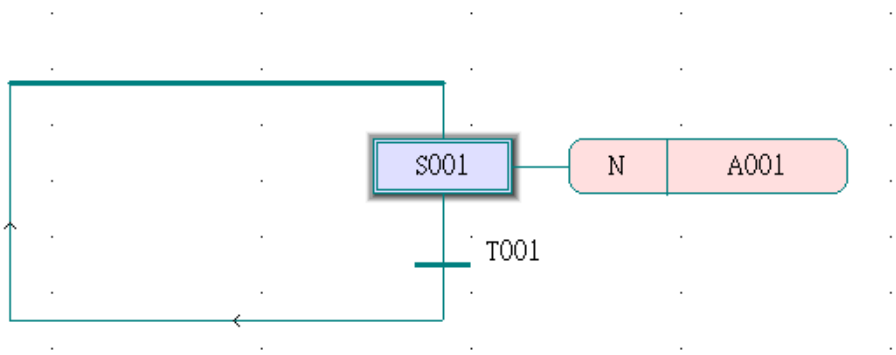


图 6.3.5.4 SFC 范例

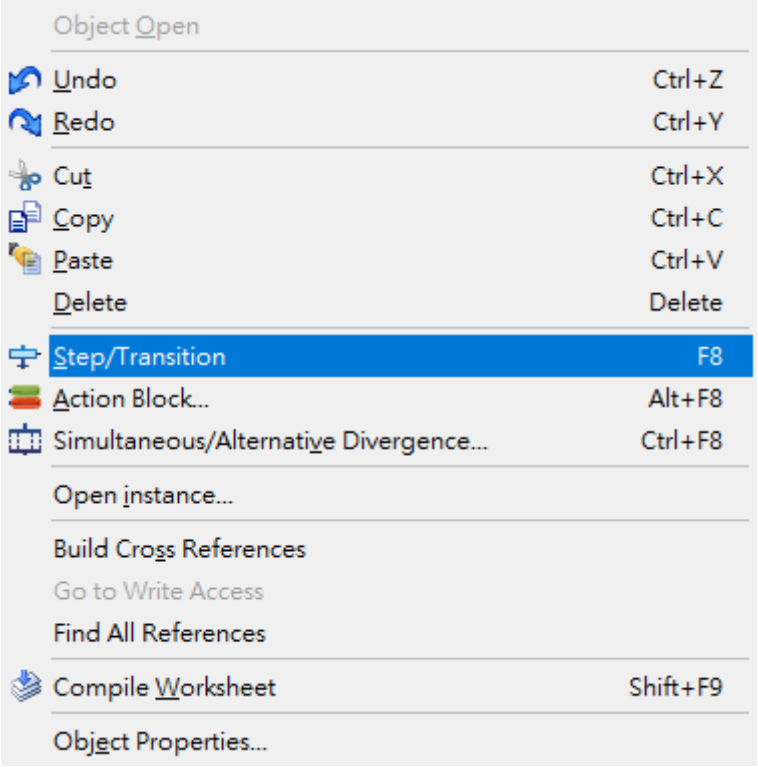


图 6.3.5.5 SFC 新增组件路径

Step/Transition：在选择 Step 的情况下，可以增加一个工作 Step。

Action Block：在选择 Action Block 或 Step 的情况下，可以增加一个动作。

Simultaneous/Alternative Divergence：在选转换(T001)的情况下，则会在该转换下增加一个 2 条支路的并行结构。如果在选择 Step 的情况下点击，则会在该步下增加 2 条支路的选择结构。

5. 在工作单输入如图所示之程序内容。

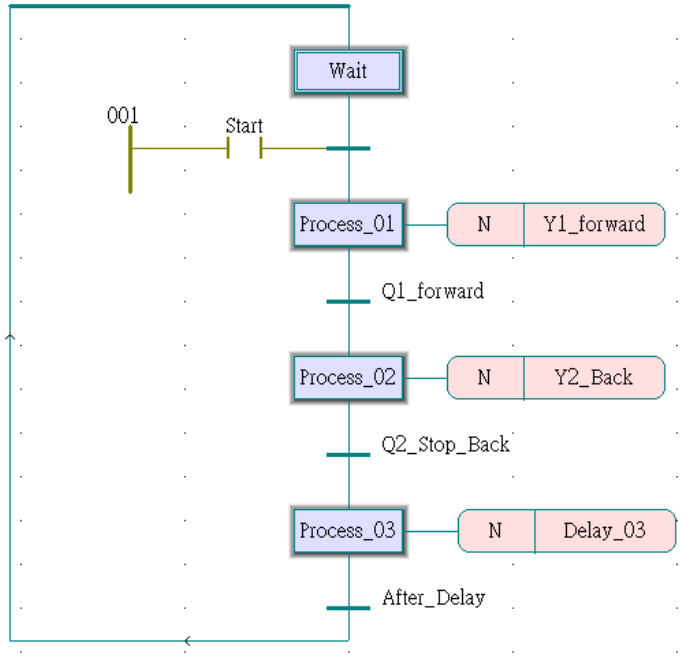


图 6.3.5.6 SFC 范例

6

6. 双击 Q1_forward 转换，开启工作单，在其中编写 FBD 程序，如图所示。

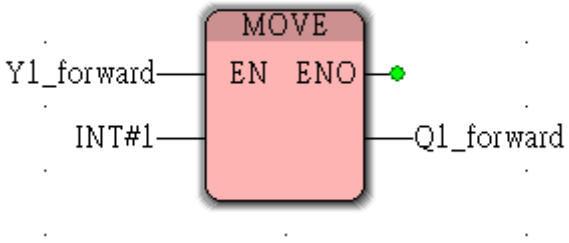


图 6.3.5.7 SFC 范例功能块

注：转换程序可以采用 LD、ST、IL、FBD 四种编程语言中的任何一种实现。

7. 点击常用工具栏中的编译(Make)按钮图标或是功能区 Build 选单中的 Make 功能。
8. 在程序(Program)中可以透过拖曳 ST_Example 功能块来实例化此功能块，如图所示。

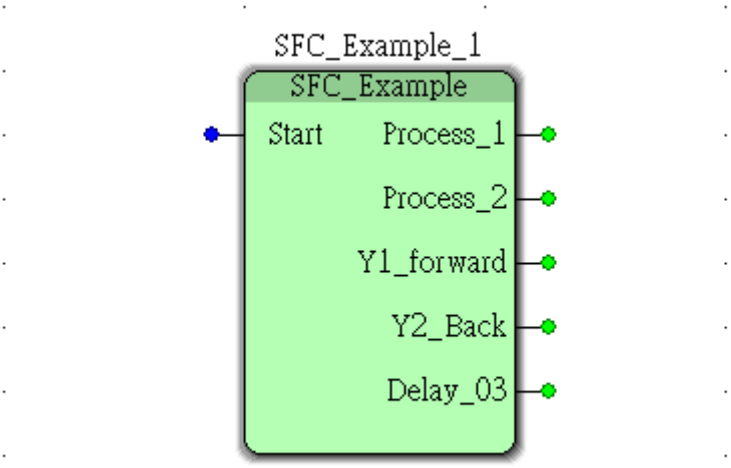


图 6.3.5.8 SFC 范例功能块

6.4 PLCopen 介绍

简介 PLCopen 组织，并提供常用的功能块说明。

6.4.1 PLCopen 组织简介

PLCopen 组织是独立于制造商和产品的国际组织，此组织的宗旨是促进 PLC 兼容软件的开发和使用，提倡使用 PLC 国际标准(IEC 61131-3)作为使用规范，其目的以「解决与控制编程相关的问题」和「支持该领域内国际标准的使用」为使命。用户可通过在众多程序开发环境中应用此标准，于不同品牌产品和不同类型的控制器之间移植控制程序，实现互换功能，因此 PLCopen 为此设下了技术与推广组织，如下图。

PLCopen 组织的一项主要工作是使用 IEC-61131-3 工业控制编程(Programming)国际标准，将各 Function Block 接脚统一化，使设计者可依照此标准设计，也让使用方式与逻辑更加一致。PLCopen 组织利用 IEC-61131-3 标准共制定了五项的 PLC 程序语言；Ladder Diagram(LD)阶梯图、Sequential Function Chart (SFC) 顺序功能流程图、Function Block Diagram (FBD) 功能区块图(具有结构性，属面向对象)、Structured Text (ST)似 Pascal 语言、Instruction List (IL)简单文本 PLC 语言。而每一个程序则由逻辑组件、模块化以及软件技术组成，从而提高使用率、减少错误并有效的编辑程序。

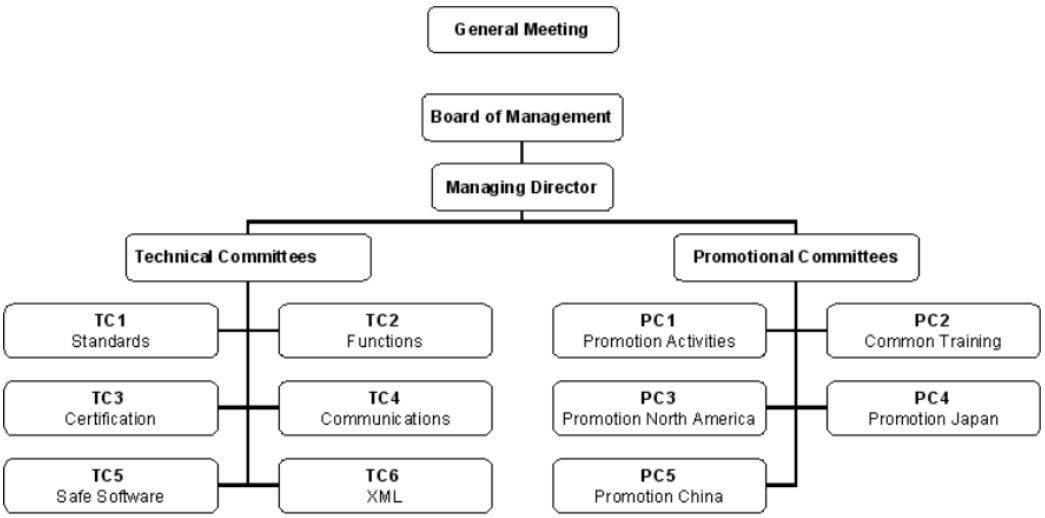


图 6.4.1.1 PLCopen 组织架构 (出处: http://www.plcopen.org/index.html)

6

6.4.2 PLCopen Motion Control 简介

PLCopen 运动控制主要由 PLCopen 组织底下的 TC2 负责规划，TC2 基于应用需求和工程项目规格，设计工程师需要使用或选择范围广泛的运动控制器来完成。过去这些应用开发需要使用个别应用软件来创建，即使功能相同，也会因为软件的不同而无法共享。 PLCopen 运动标准提供了一种标准的应用程序库，可重复使用在多种硬件平台上，这样的方式可大幅降低开发和维护成本。除此之外，编辑程序变得更加容易简单，培训成本降低、并且统一设计的方式，使得功能可做跨平台使用。实际上，其标准化是通过定义可重复使用的 Block 来完成的。透过此方式，用户无须依赖硬件的编程，应用软件亦可重复增加，以减少人员培训和功能维护的成本，使各种控制解决方案变得更有弹性。由于数据隐藏和封装可使用不同的体系，例如集中或分布式的控制，不仅能运用在单一的应用功能上，更能作为一个基本层，广泛的用于不同领域中。

PLCopen 为可开放式，可应用在现有或未来技术上，故 PLCopen 所定义的运动控制功能主要是以 Function Block 为主，PLCopen 运动控制功能所开发的标准文件如下：

Part 1 - Function Blocks for Motion Control
Part 2 - Extensions(in the new release 2.0 merged with Part 1)
Part 3 - User Guidelines
Part 4 - Coordinated Motion
Part 5 - Homing Procedures
Part 6 - Fluid Power Extensions

经 PLCopen 认证以后符合标准的标示如下，由图示便可得知其组织定义的 Motion 功能是以 Function Block 的概念为设计基础。

图 6.4.2.1 PLCopen motion control 认证标记图

6.4.3 常用变量定义

下表为功能块常见的输入及输出变量列表，依照不同功能块的特性而定，通常会包含至少一个或是一组变量，其变量如下：

输入变量			
名称	数据型态	设定值（默认值）	功能
Enable	BOOL	True / False (False)	启用功能块
Execute	BOOL	True / False (False)	启动功能块
输出变数			
名称	数据型态	设定值（默认值）	功能
Done	BOOL	True / False (False)	功能块动作完成
Valid	BOOL	True / False (False)	输出值有效
Busy	BOOL	True / False (False)	功能块动作已被触发
Active	BOOL	True / False (False)	功能块动作执行中
CommandAborted	BOOL	True / False (False)	功能块被其他命令中断
Error	BOOL	True / False (False)	功能块产生错误

一个运动功能块中必定会包含 Enable 或是 Execute 其中一个输入变量，用于执行此功能块。而显示运动功能块执行状态时，则会包含 Busy 和 Done 两种输出变量。Active 变量则是有 Buffer 输入变量的功能块才提供。对于同一个运动轴，可同时有多个 Busy 输出，但只会有一个功能块 Active 输出。如果功能块在执行过程可被其他功能块中断，则会包含 CommandAborted 变量。Error 变量则用来显示此运动功能块在启动过程中发生错误。

单一运动功能块的输入变量除了 Enable 和 Execute 以外，会包含其他运动数据输入端，这些数据 / 状态会有以下特性：

- 输入数据取用时机
  1. 具有 Enable 输入时：参数会在 Enable 上升沿触发时被取用。与 Execute 相比，Enable 较常被设计为执行中持续更新的形式，例如读取参数功能块。
  2. 具有 Execute 输入时：参数会在 Execute 上升沿触发时被取用。若要让变更数据再次生效，必须在修改输入参数后，再次让 Execute 上升沿触发。
- 输入数据超出范围

  若于运动功能块输入超出允许范围的数值且本功能块被启动时，会造成输入的数据被限制或是产生错误。此时产生的轴错误结果，是运动功能块的应用错误所造成的，因此用户必须确保输入的数据正确。

6

- 输入时缺少参数

  根据 IEC-61131-3 定义，假使功能块输入参数缺少时，则输入参数维持上次输入值，第一次使用则是使用系统默认值。

- 输出状态互斥

  1. 功能块有 Execute 输入变量时，Busy、Done、Error 和 CommandAborted 会彼此互斥。运动功能块中同时只有一个能为 True，且其中一个必定为 True。
  2. 功能块有 Buffer 输入变量时，Active、Error、Done 和 CommandAborted 同时只能有一个被设置。
  3. 功能块有 Enable 输入变量时，输出 Valid 与 Error 彼此互斥，同时只能有一个被设置。

- 输出数据 / 状态有效时机

  1. 功能块有 Execute 变量时，Done、Error、ErrorID、CommandAborted 以及数据输出会在 Execute 下降沿时被重置，然而 Execute 下降沿不会停止，也不影响功能块实际的执行，即使在功能块完成前 Execute 就被重置，相对应的输出状态仍会产生，并保持一个周期。如果功能块在完成之前就收到新的 Execute，功能块不会对之前动作的 Done 与 CommandAborted 有任何反馈，且可能产生功能块错误。
  2. 功能块有 Enable 变量时，Valid、Enable、Busy、Error 和 ErrorID 输出将跟着 Enable 下降沿被重置。

- 各输出端特性

  1. Done 输出特性：Done 输出会在被命令的运动成功完成时被设置。
  2. Busy 输出特性：
     2.1 功能块有 Execute 变量时，每个运动功能块会有一个 Busy 输出，用来反映运动功能块尚未完成，并且输出状态值将会被更新。Busy 在 Execute 上升沿被设置时，在 Done、CommandAborted 和 Error 被设置时将被重置。
     2.2 功能块有 Enable 变量时，每个运动功能块会有一个 Busy 输出，用来反应运动功能块尚未完成，并且输出状态值将会被更新，Busy 在 Enable 上升沿被设置时，只要运动功能还在执行，动作就会保持住，同时，对应的输出仍会有变化。
  3. Active 输出特性：
     3.1 有 Buffer 缓冲输入的功能块需要 Active 输出变量，当功能块实际控制对应的运动轴时，此输出会被设置，没有 Buffer 缓冲输入的功能块，其 Active 与 Busy 输出行为相同。
     3.2 对于同一个运动轴，可以有多个功能块的 Busy 变量同时输出，但在一个时刻只能有一个功能块的 Active 变量输出，表示正在执行此功能块。

CommandAborted 输出特性：CommandAborted 被启动时，会实时停止所有尚未完成的运动。

4. Enable 与 Valid 的关系：搭配 Enable 时，若运动功能块包含输出状态或是数值的变量，则会由 Valid 输出变量来表示这些输出是否有效，Valid 输出只在 Enable 为真及输出有效时为真，若运动功能块有错误会让输出无效，Valid 会为 False，直到错误状况消失，输出值重新生效时，Valid 才会再次被设置。
5. 正负号规则：Acceleration、Deceleration 和 Jerk 永远为正。Velocity、Position 或 Distance 则可能包含正负号。

■ 输入/输出端的变化时序

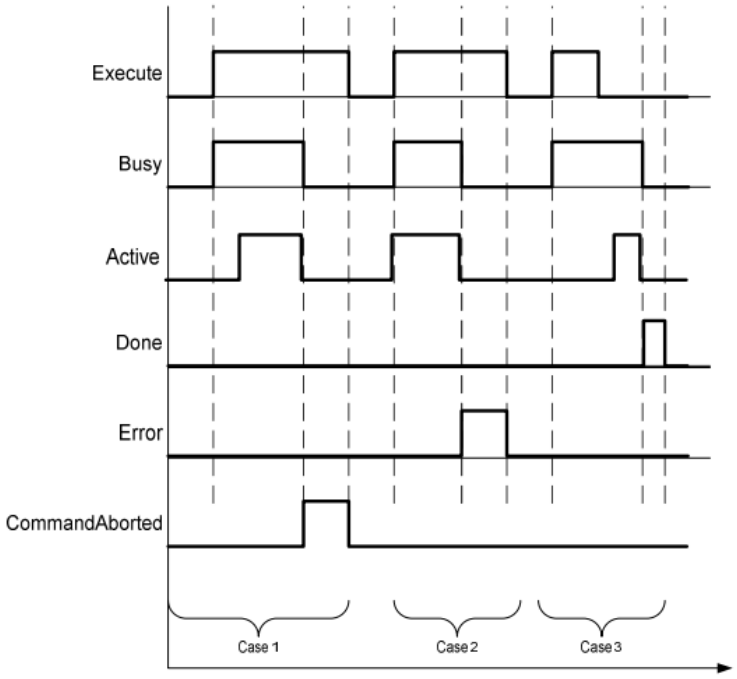


图 6.4.3.1 Execute 功能块输入输出端的变化时序范例（出处: MULTIPROG 快速入门指南）

6

Case 1：含 Buffer 输入的运动功能块执行时，被其他功能块中断。

Case 2：运动功能块执行时发生异常。

Case 3：含 Buffer 输入的运动功能块完成正常动作。(即使 Execute 已为 False，功能块仍要将动作完成，完成时 Done 讯号输出至少要维持一个 PLC 扫描周期。

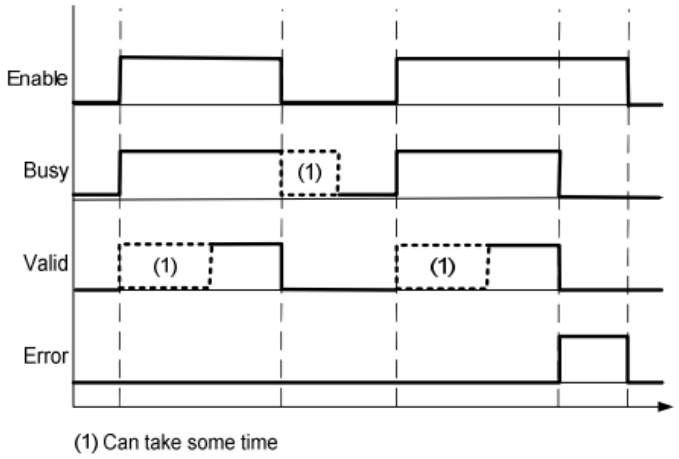


图 6.4.3.2 Enable 功能块输入输出端的变化时序范例（出处: MULTIPROG 快速入门指南）

注：(1)表示需要一段时间。

■ 缓冲模式(BufferMode)

若运动功能块包含 BufferMode 输入接脚，则 BufferMode 决定了当前指令与前一笔指令的缓冲模式，支持的缓冲模式如下：

缓冲模式	功能
0：Aborting	中断前一笔指令，立即执行当前指令
1：Buffered	前一笔指令结束后，自动执行当前指令
2：BlendingLow	比较前一笔指令与当前指令的速度，然后以较低的速度作为中继速度
3：BlendingPrevious	以前一笔指令的速度作为中继速度
4：BlendingNext	以当前指令的速度作为中继速度
5：BlendingHigh	比较前一笔指令与当前指令的速度，然后以较高的速度作为中继速度

■ TransitionMode 接脚

若运动功能块包含 TransitionMode 输入接脚且 BufferMode 选择 Blending 模式(Blending Low、Blending Previous、Blending Next、Blending High)时，可选择重迭模式来设定群组合成路径的行为。

其中 BufferMode 有以下几种选择：

None：无迭合模式。

Corner Distance：转角迭合。当群组的回授位置距离目标位置等于设定值时，执行下一笔群组命令。

Immdiate：立即迭合。当下一笔命令被触发时立即执行路径迭合。

6.5 运动功能块介绍

6.5.1 运动功能块总览

6

分类		名称	说明
单轴运动控制	定位控制	MC_Home	根据原点复归设定的参数来启动指定轴的原点复归流程。
		MC_Stop	依照使用者输入的减速度值来执行一个受控的单轴运动停止指令。
		MC_Halt	依照使用者输入的减速度值来使单轴运动停止，将轴状态切换至DiscreteMotion，指令完成后再将状态切换至Standstill。
		MC_Jog	对指定轴执行连续型吋动指令。
		MC_MoveAbsolute	依照用户所规划之速度、加速度、减速度等运动参数，移动到用户设定的绝对目标位置。
		MC_MoveRelative	依照用户者所规划的目标速度、加速度、减速度与加加速度等运动参数，移动到使用者设定的相对距离位置。
		MC_MoveAdditive	依照用户输入的目标速度、加速度、减速度与加加速度，来执行一个相对距离运动命令，此命令的目标位置会附加到前一笔命令指定的目标位置上。
		MC_MoveSuperimposed	依照用户输入的目标速度、加速度、减速度与加加速度，迭加设定距离在当前运动命令上。
		MC_SetPosition	依照输入的指定位置来设定轴的当前位置。
		DMC_Home	根据功能块所设定的参数，启动原点复归流程。
	速度控制	MC_MoveVelocity	使控制轴按照设定的加减速运动至设定速度，并等速运行。
	扭力控制	MC_TorqueControl	单轴扭力控制。
	管理型	MC_Power	控制指定轴进行伺服使能或关闭。
		MC_Reset	清除或重置所有指定轴相关的错误。
		MC_SetOverride	透过超驰控制(override control)系数并依加(减)速度或是加加速度来改变轴的速度。
		MC_ReadParameter	读取指定轴的信息。

分类		名称	说明
		MC_WriteParameter	写入指定轴的信息
		DMC_SetServoGain	根据使用者输入的增益值设置前馈控制器，并将其输出作为速度前馈输入至伺服。
	资料撷取	DMC_TouchProbe	依据使用者之设定，在触发事件中撷取所需要的数据。
		MC_AbortTrigger	中断指定触发事件。
多轴运动控制	同步控制	MC_GearIn	依照输入的齿轮比，建立主从轴的速度比例关系。
		MC_GearOut	解除主从轴间的速度比例关系，并维持当下速度。
		DMC_CamIn	使主轴与从轴建立凸轮关系，并根据凸轮表的描述运动。
		MC_CamOut	解除已建立的电子凸轮关系，解除后从轴维持当前速度运动。
		MC_PhasingAbsolute	同步运动中，以绝对值设定调整主轴跟从轴间的相位关系。
		MC_PhasingRelative	同步运动中，以相对值设定调整主轴跟从轴间的相位关系。
		DMC_GantrySynchornize	根据使用者输入的增益值设计PID控制器，并将其输出作为速度前馈输入至伺服。
群组运动控制	定位控制	MC_GroupStop	使所有群组成员轴依设定减速度减速至停止。
		MC_MoveDirectAbsolute	控制指定群组轴执行无规划路径运动，并移动至用户设定的绝对目标位置。
		MC_MoveDirectRelative	控制指定群组轴执行无规划路径运动，并移动至用户设定的相对目标位置。
		MC_MoveLinearAbsolute	控制指定群组轴执行直线运动，移动至用户设定的绝对目标位置。
		MC_MoveLinearRelatve	控制指定群组轴执行直线运动，移动至用户设定的相对目标位置。
	管理型	MC_GroupReset	清除或重置所有群组轴及其成员轴的相关错误。
		MC_GroupSetOverride	透过超驰控制(Override control)系数并依所设定的加减速度及加加速度来改变群组成员轴的速度。

6

分类		名称	说明
群组运动控制	参数设定	SYS_ParaRead	读取各种系统参数，控制器、轴、群组、总线装置等。
		SYS_ParaWrite	写入各种系统参数，控制器、轴、群组、总线装置等。
	警报设定	SYS_GetAlarmCode	读取当前控制器错误码。
		SYS_ResetAlarmCode	清除当前控制器错误码。
	自定义系统功能块	CEIL	将使用者输入的数值无条件进位至整数。
		FLOOR	将使用者输入的数值无条件舍去至整数。
		Ring_Queue	实现一长度为64的环状数据序列。
	自定义通讯功能块	CAN_PDORead	进行PDO读取功能。
		CAN_PDOWrite	进行PDO写入功能。
		CAN_SDORead	进行SDO读取功能。
		CAN_SDOWrite	进行SDO写入功能。
		MB_ReadBits	使用串行端口通讯读取指定位置的一组位长度的数据串。
		MB_ReadWords	使用串行端口通讯读取指定位置的一组word长度的数据串。
		MB_SerialPort_Set	控制器串行端口通讯参数更改。
		MB_TCP_Connect	藉由用户输入的IP建立一个TCP Socket通讯。
		MB_TCP_DisConnect	解除已建立的TCP Socket通讯。
		MB_TCP_ReadBits	使用Socket通讯读取指定位置的位元值。
		MB_TCP_ReadWords	使用Socket通讯读取指定位置的一组word长度的数据串。
		MB_TCP_WriteBit	使用Socket通讯写入一个位值至指定位置。
		MB_TCP_WriteBits	使用Socket通讯写入一组位型态数据串至指定位置。
		MB_TCP_WriteWord	使用Socket通讯写入一个word长度的数据至指定位置。
		MB_TCP_WriteWords	使用Socket通讯写入一组word长度的数据串至指定位置。
		MB_WriteBit	使用串行端口通讯写入一个位值至指定位置。
		MB_WriteBits	使用串行端口通讯写入一组位长度的数据串至指定位置。

分类		名称	说明
群组运动控制	自定义通讯功能块	MB_WriteWord	使用串行端口通讯写入一个word长度的数据至指定位置。
		MB_WriteWords	使用串行端口通讯写入一组word长度的数据串至指定位置。
凸轮表控制	凸轮表操作	ECAM_GenTableByFile	藉由DMARS所产生的.bin档案来建立凸轮表。
		ECAM_GenTableByData	根据输入的凸轮数据来建立凸轮表。
		ECAM_GenTableByVel	根据输入的主轴长度、从轴长度以及等待、加速、等速、减速、停止速度的百分比，来建立凸轮表。
		ECAM_AddPoints	为目前系统中的凸轮表加入数据点。
		ECAM_DelPoints	删除目前系统中的凸轮表中的关键点。
		ECAM_ModifyPoints	为目前系统中的凸轮表修改特定关键点。
		ECAM_ReadPointsByID	藉由输入凸轮表的TableID和参数来读取凸轮表中关键点信息。
		ECAM_ReadTableByID	藉由输入凸轮表的TableID来读取整个凸轮表的数据。
		ECAM_SaveTable	根据输入凸轮表的TableID，将编辑过后的凸轮表储存至控制器的断电保持区。
		ECAM_GenRotaryCutTable	由使用者输入的参数生成一张飞剪凸轮表。
		ECAM_GenFlyingShear Table	根据输入的参数生成一张追剪凸轮表。
		ECAM_SetCamTappetData	设定挺杆点数据。
		ECAM_GetCamTappet Status	取得挺杆点状态。
		ECAM_SetConnectVelocity	根据输入凸轮表的TableID，设定是否自动速度平滑凸轮表，开启AutoSetVel功能之后，凸轮表会自动填入连接速度，让凸轮在移动的过程中，速度曲线不会减速至0。
		ECAM_PreviewTable	预览凸轮表的各项信息。
		ECAM_ModRotCutVelRatio	调整飞剪凸轮表中的同步速度比例。
		ECAM_SetOnlineChgMode	设定变更凸轮表生效时机。

6.5.2 单轴功能块（Motion SingleAxis)

6.5.2.1 MC_Home

6

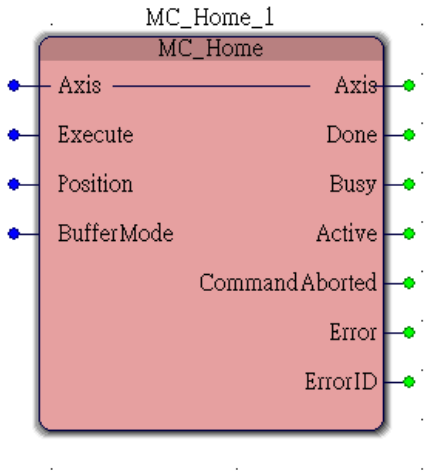
■ 类型

Function Block

■ 功能描述

根据原点复归设定的参数来启动指定轴的原点复归流程。例如：寻找极限、传感器和扭力等模式来执行回原流程。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值（默认值)	功能
Axis	AXIS_REF	None	指定运动轴编号

■ 输入脚位

脚位名称	数据型态	设定值（默认值)	功能
Execute	BOOL	True / False (False)	当Execute为上升缘时，此功能即开始执行
Position	LREAL	正数、负数或0.0 (0.0)	原点距离目前位置的偏差值
BufferMode	INT	None	保留脚位

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	回原完成时为True
Busy	BOOL	True / False (False)	回原指令被执行时为True
Active	BOOL	True / False (False)	轴正在回原时为True
Command Aborted	BOOL	True / False (False)	指令被中断时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 详细说明请参考第9章

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	绝对寻址完成时	■ Execute转为False时 ■ 若Execute为False而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
Active	当轴运动开始时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
CommandAborted	此功能块指令被MC_Stop中断时	■ 在Execute下降沿时 ■ 当Execute为False而CommandAborted转为True，此时CommandAborted 维持一个扫描周期的True状态后，立刻转为False
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

6

■ 输出脚位的变化时序

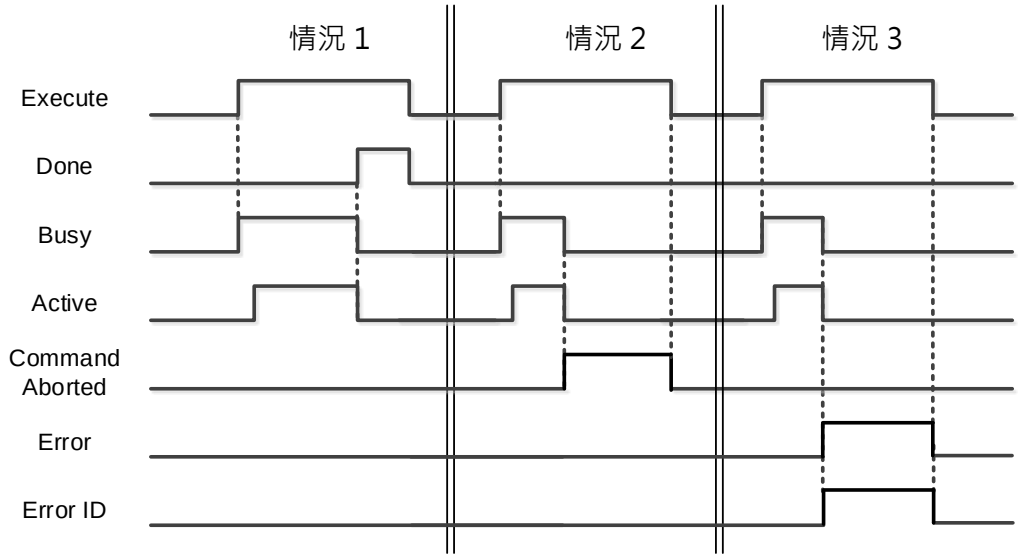


图 6.5.2.1.1 MC_Home 脚位时序图 (时序详细说明请参考 6.4.3 节)

情况 1：当此指令第一次执行(Execute 由 FALSE 变为 TRUE)时，Busy 脚位在同一时间变为 TRUE，表示指令已被执行。Active 由 FALSE 变为 TRUE 时，代表轴正在执行回原流程。等待回原流程完成，Done 脚位变为 TRUE，同时 Busy 与 Active 变为 FALSE。当 Execute 由 TRUE 变为 FALSE 时，Done 同时变为 FALSE。

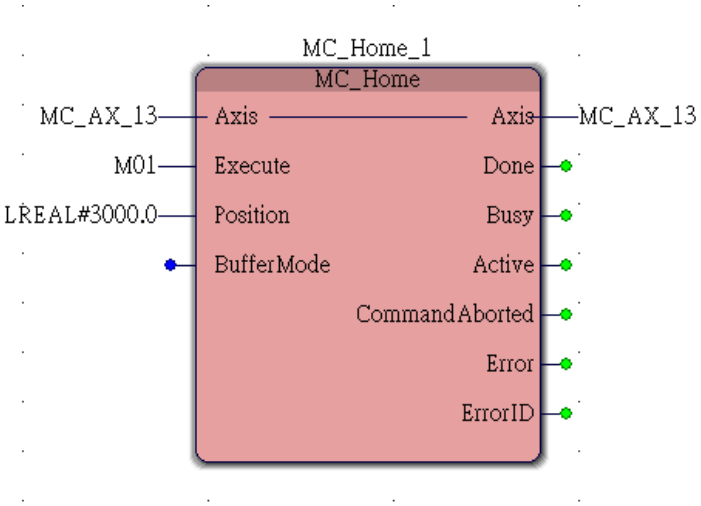
情况 2：当此指令执行的过程中，执行 MC_Stop 命令，此时 CommandAborted 脚位由 FALSE 变为 TRUE，同时 Busy 与 Active 脚位变为 FALSE，回原流程被中断。当 Execute 由 TRUE 变为 FALSE 时，CommandAborted 同时变为 FALSE。

情况 3：当此指令在执行过程中，有错误产生，则 Error 变为 TRUE，同时 Busy 与 Active 变为 FALSE，此时请观察 ErrorID 脚位所显示的错误码，并了解详细错误讯息。Error 与 ErrorID 脚位将一直维持，直到 Execute 脚位变为 FALSE。

- 回原模式设定

  回原模式的流程将依据用户所在 DMARS 软件上所设定之回原模式进行。回原模式详细说明请参照伺服手册。

- 参考范例



此范例将依下列条件来设计

I. M01 设置触发脚位，可以为某个数字输入讯号。
II. 目前位置 10000 (自定义单位)。
III. 功能块输入脚位 Position 设计为 3000.0(自定义单位)。

预计获得动作与结果

I. 将目前位置设为 0 自定义单位 (原点)。
II. 将相对目前位置 3000.0 自定义单位位置设为 0 自定义单位(原点)。
III. 目前点位将变更为 -3000.0 自定义单位。

- 图形表示

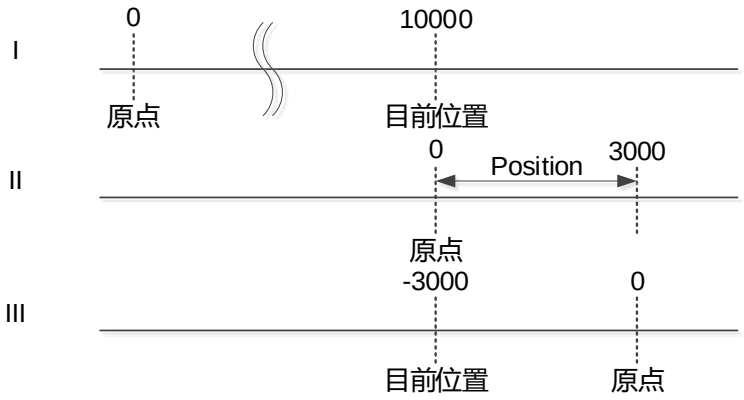


图 6.5.2.1.2 MC_Home 范例说明

6.5.2.2 MC_Stop

6

■ 类型

Function Block

■ 功能描述

依照使用者输入的减速度值来执行一个受控的单轴运动停止指令。

■ 图形表示

■ 输入输出共享变量

名称	数据型态	设定值 (默认值)	功能
Axis	AXIS_REF	None	指定运动轴编号

■ 输入变量

名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	当Execute为上升缘时，此功能即开始执行
Deceleration	LREAL	正数(0.0)	运动停止命令之减速度数值
Jerk	LREAL	正数或0.0 (0.0)	运动停止命令之减减速度数值，将影响运动的平滑程度 (单位：用户单位/秒^3)

■ 输出变数

名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	绝对寻址完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Command Aborted	BOOL	True / False (False)	指令被中断时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000~0xFFFF (0)	指令错误发生时记录的错误码 详细说明请参考第9章

■ 输出变量的变化时序

名称	输出变数上升沿时机	输出变数下降沿时机
Done	绝对寻址完成时	■ 当Execute转为False时 ■ 若Execute为False而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
Command Aborted	■ 此功能块指令被其他缓冲模式设为Aborting的指令中断时 ■ 此功能块指令被MC_Stop中断时	■ 在Execute下降沿时 ■ 当Execute为False而CommandAborted转为True，此时CommandAborted维持一个扫描周期的True状态后，立刻转为False
Error / ErrorID	当指令的执行条件或输入值发生错误时(错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

■ 输出变量的变化时序

下图为功能块正确完成的时序。

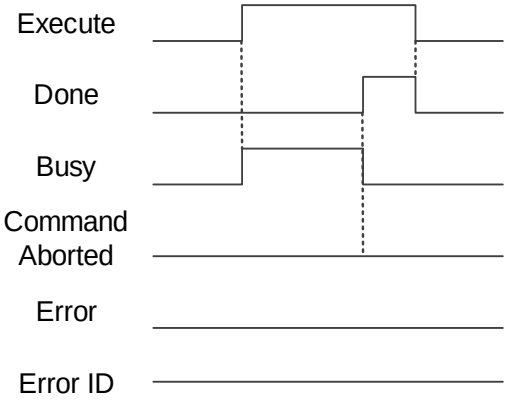


图 6.5.2.2.1 MC_Stop 变量时序图(时序详细说明请参考 6.4.3 节)

6

■ 速度时间图

若单轴正在运动中，启动 MC_Stop 功能中断此单轴运动命令，其速度图的变化如下。

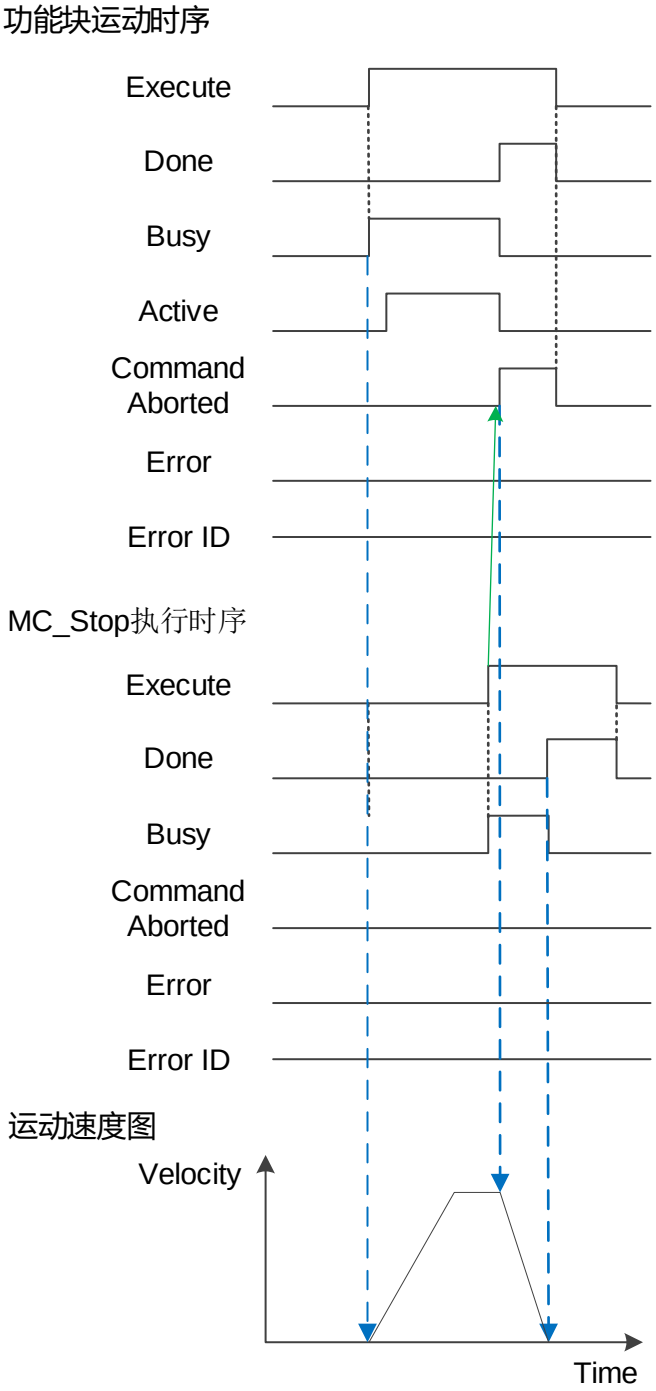
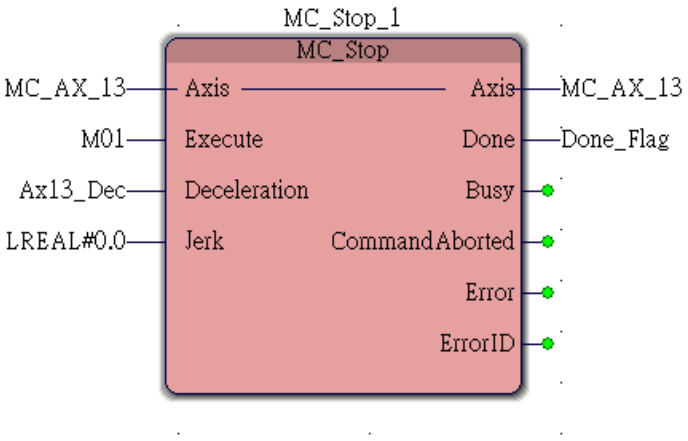


图 6.5.2.2.2 MC_Stop 速度时间图

■ 参考范例

以下为 13 号轴被启动 Stop 的运动命令的时序图，其中包含两种 Execute 的执行方式，一是保持住执行，直到完成后才释放(a)，另一个是触发后即释放(b)。



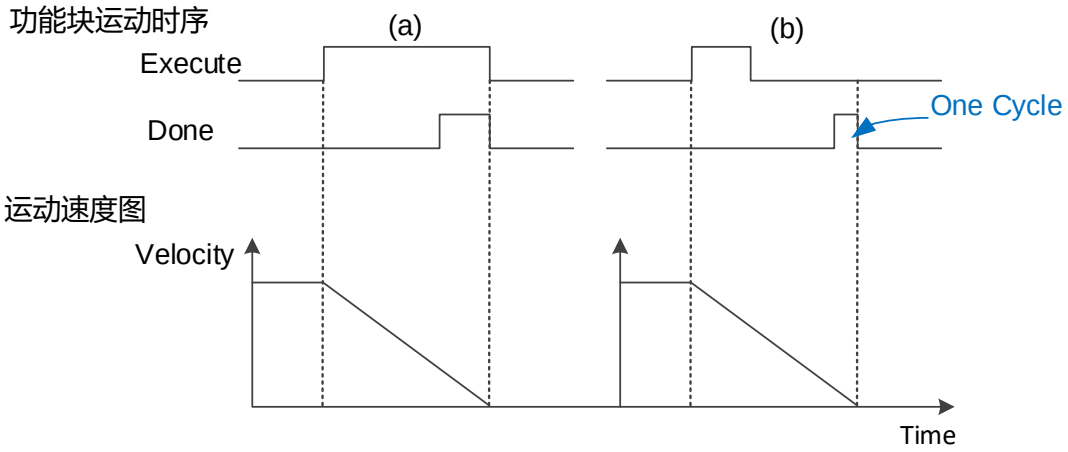


图 6.5.2.2.3 MC_Stop 范例说明

6.5.2.3 MC_Halt

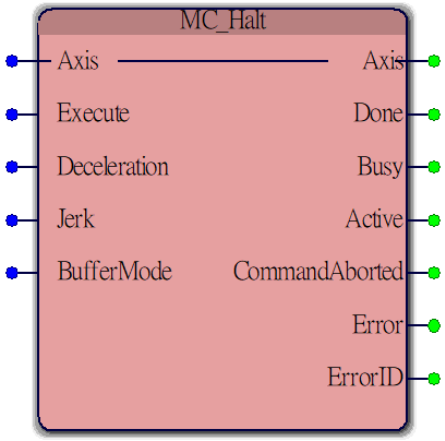
■ 类型

Function Block

■ 功能描述

依照使用者输入的减速度值来使单轴运动停止，将轴状态切换至 Discrete Motion，指令完成后再将状态切换至 Standstill。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
Axis	AXIS_REF	-	指定运动轴编号

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	当Execute为上升缘时，此功能即开始执行
Deceleration	LREAL	正数 (0.0)	运动停止命令之减速度数值 (单位: 用户单位/秒^2)
Jerk	LREAL	正数或0.0 (0.0)	运动停止命令之减减速度数值，将影响运动的平滑程度 (单位: 用户单位/秒^3)
BufferMode	INT	0 ~ 5 (1)	指定此功能块的缓冲行为 0：中断(Aborting) 1：等待(Buffered)

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	速度为零时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Command Aborted	BOOL	True / False (False)	指令被中断时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF(0)	指令错误发生时记录的错误码 详细说明请参考第9章

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	轴停止时	■ Execute转为False时 ■ 若Execute为False，而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
Command Aborted	■ 此功能块指令被其他缓冲模式设为Aborting的指令中断时 ■ 此功能块指令被 MC_Stop中断时	■ 在Execute下降沿时 ■ 当Execute为False而CommandAborted转为True，此时 CommandAborted 维持一个扫描周期的True状态后，立刻转为False
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

■ 输出脚位的变化时序

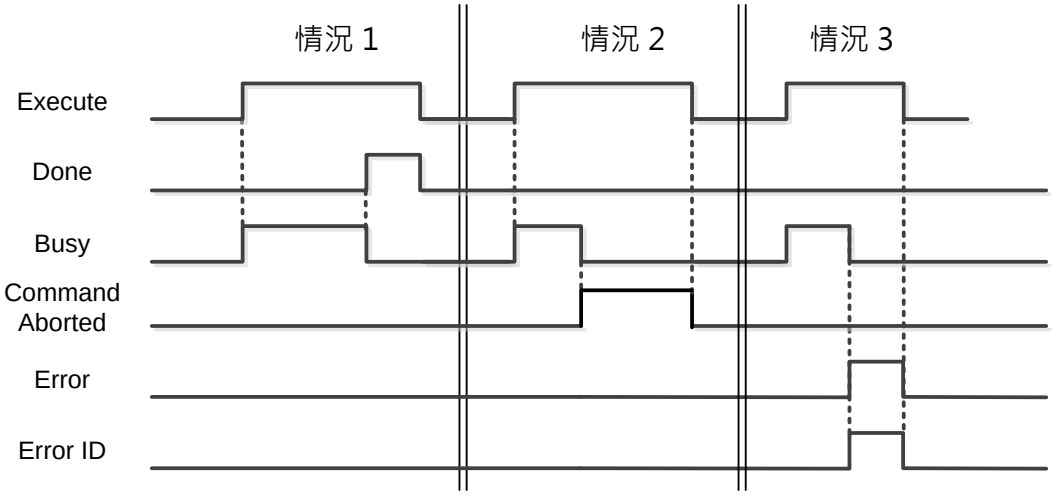


图 6.5.2.3.1 MC_Halt 脚位时序图 (时序详细说明请参考 6.4.3 节)

6

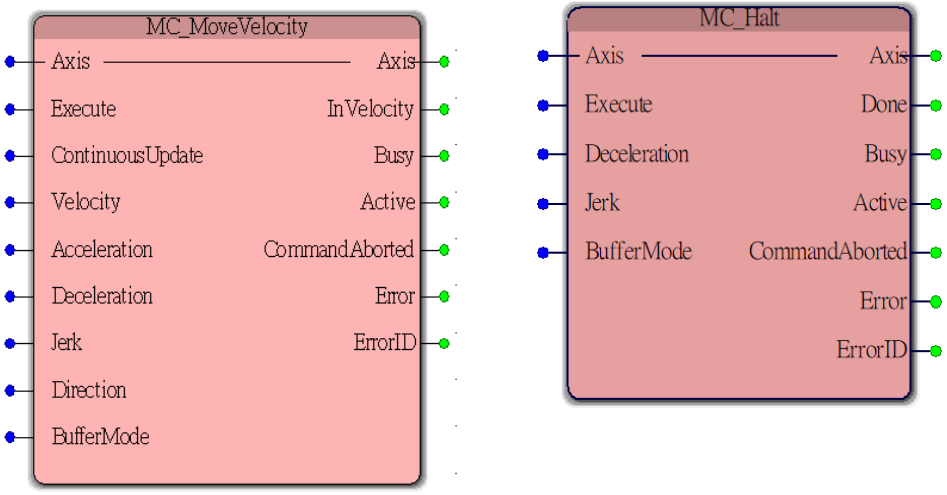
情况 1：当此指令第一次执行(Execute 由 FALSE 变为 TRUE)时，Busy 脚位在同一时间变为 TRUE，表示指令已被执行。等待伺服减速至零速度，Done 脚位变为 TRUE，同时 Busy 变为 FALSE。当 Execute 由 TRUE 变为 FALSE 时，Done 同时变为 FALSE。

情况 2 :当此指令执行的过程中，被其他运动指令中断，此时 CommandAborted 脚位由 FALSE 变为 TRUE，同时 Busy 脚位变为 FALSE。当 Execute 由 TRUE 变为 FALSE 时，CommandAborted 同时变为 FALSE。

情况 3：当此指令在执行过程中，有错误产生时，Error 变为 TRUE，同时 Busy 变为 FALSE，此时请观察 ErrorID 脚位所显示的错误码，了解详细错误讯息。Error 与 ErrorID 脚位将一直维持直到 Execute 脚位变为 FALSE。

■ 速度时间图

单轴正在运动中，若启动 MC_Halt 功能中断此单轴运动命令，其速度图的变化如下图。



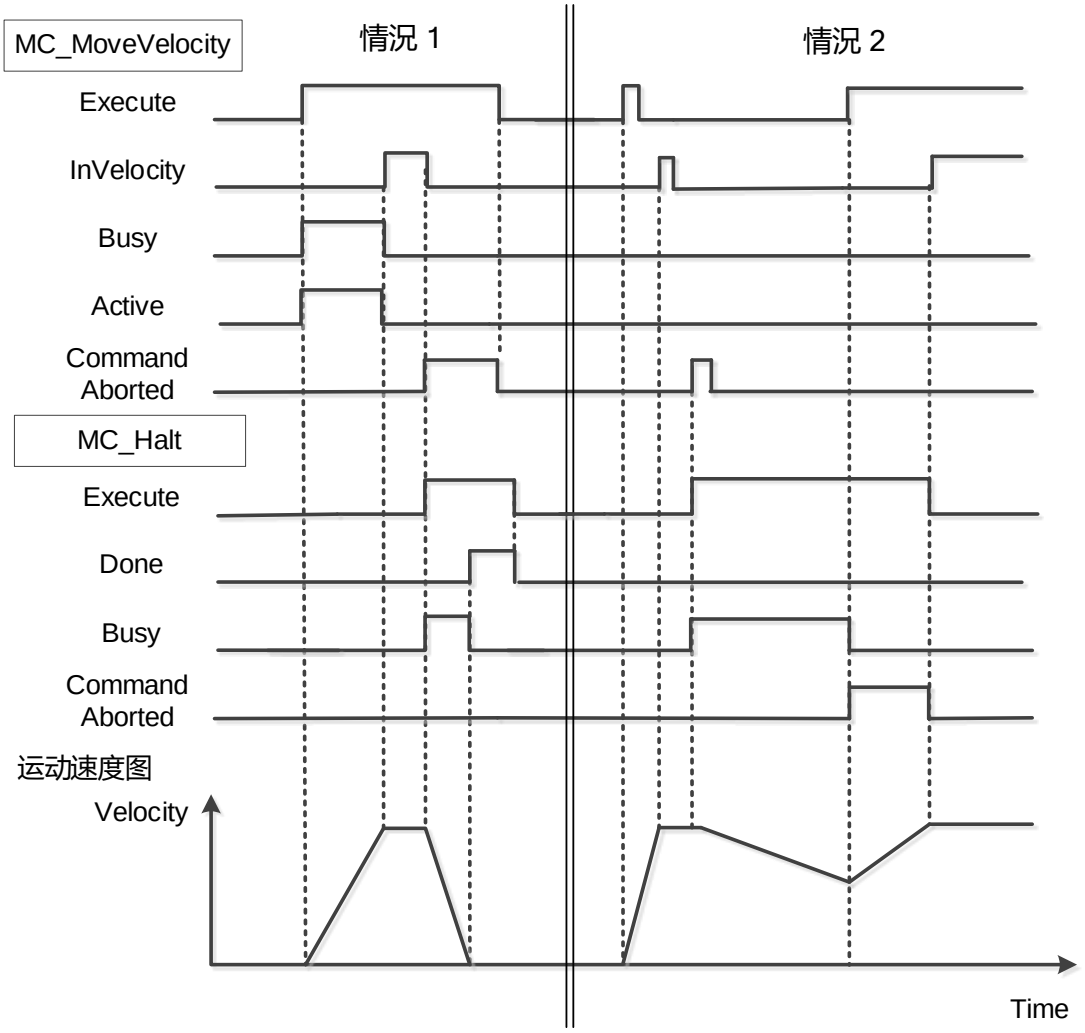


图 6.5.2.3.2 MC_Halt 速度时间图

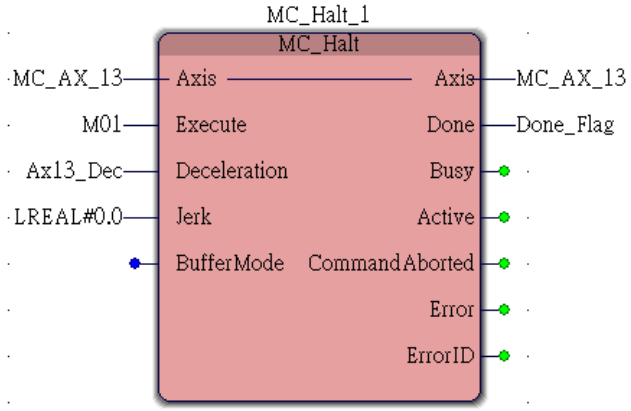
情况 1：当轴正在执行运动指令时，执行 MC_Halt，此时轴将以 MC_ Halt 所设定的减速度及减减速度进行减速运动直至轴速度为 0。

情况 2：MC_Halt 执行的过程中，可被其他运动指令插断。

6

■ 参考范例

以下为 13 号轴被启动 MC_Halt 的运动命令的时序图，其中包含两种 Execute 的执行方式，一是保持住执行，直到完成后才释放(a)，另一个是触发后即释放(b)。



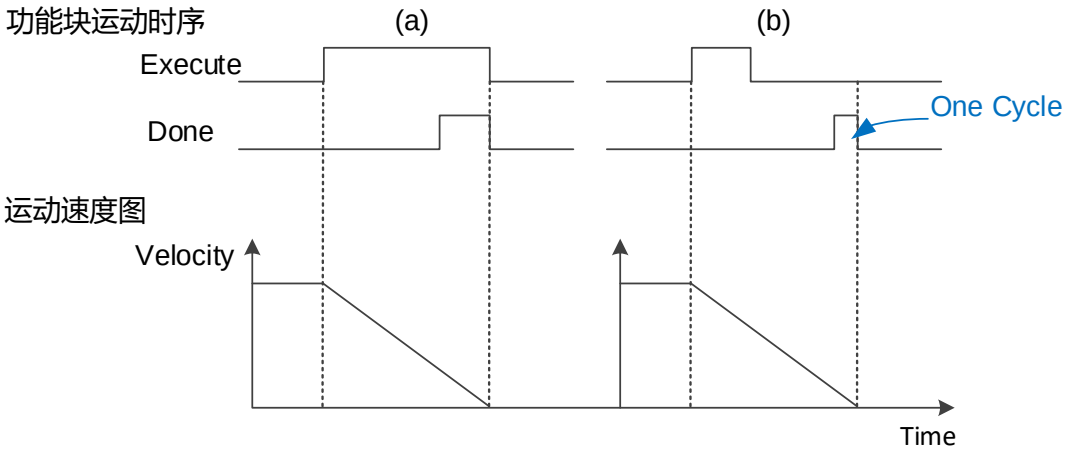


图 6.5.2.3.3 MC_Halt 范例说明

6.5.2.4 MC_Jog

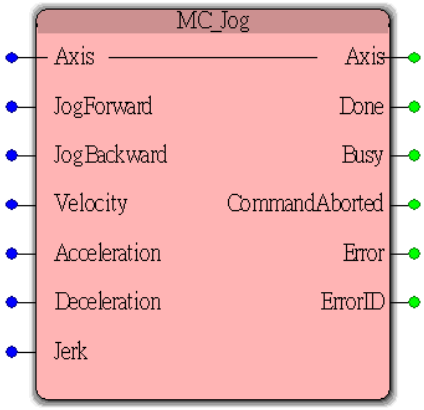
■ 类型

Function Block

■ 功能描述

对指定轴执行连续型时动指令。

■ 图形表示



■ 输入输出共享变量

名称	数据型态	设定值 (默认值)	功能
Axis	AXIS_REF	None	指定运动轴编号

■ 输入变量

名称	数据型态	设定值 (默认值)	功能
JogForward	BOOL	True / False (False)	JogForward为True时，开始执行连续时动指令；反之则停止。
JogBackward	BOOL	True / False (False)	JogBackward为True时，开始执行连续时动指令；反之则停止。
Velocity	LREAL	正数或0.0 (0.0)	最大速度数值 (用户单位/秒)
Acceleration	LREAL	正数(0.0)	加速度数值 (用户单位/秒^2)
Deceleration	LREAL	正数(0.0)	减速度数值 (用户单位/秒^2)
Jerk	LREAL	正数或0.0 (0.0)	加加速度数值 (用户单位/秒^3)

6

■ 输出变数

名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	绝对寻址完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Command Aborted	BOOL	True / False (False)	指令被中断时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 详细说明请参考第9章

■ 输出变量的变化时序

名称	输出变数上升沿时机	输出变数下降沿时机
Done	绝对寻址完成时	■ Execute转为False时 ■ 若Execute为False，而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
Command Aborted	■ 此功能块指令被其他缓冲模式设为Aborting的指令中断时 ■ 此功能块指令被MC_Stop中断时	■ 在Execute下降沿时 ■ 当Execute为False，而CommandAborted转为True，此时CommandAborted维持一个扫描周期的True状态后，立刻转为False
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

■ 输出变量的变化时序

此时序图说明一个完整的 Jog 运动命令由启动至停止的时序。

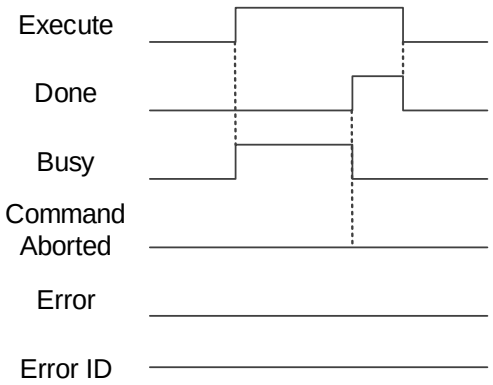


图 6.5.2.4.1 MC_Jog 变量时序图(时序详细说明请参考 6.4.3 节)

■ 速度时间图

以下提供了两种时序，一是完整完成 Jog (时动)运动命令(a)，另一个是在 Jog 运动命令执行时，因为错误产生而中断(b)。

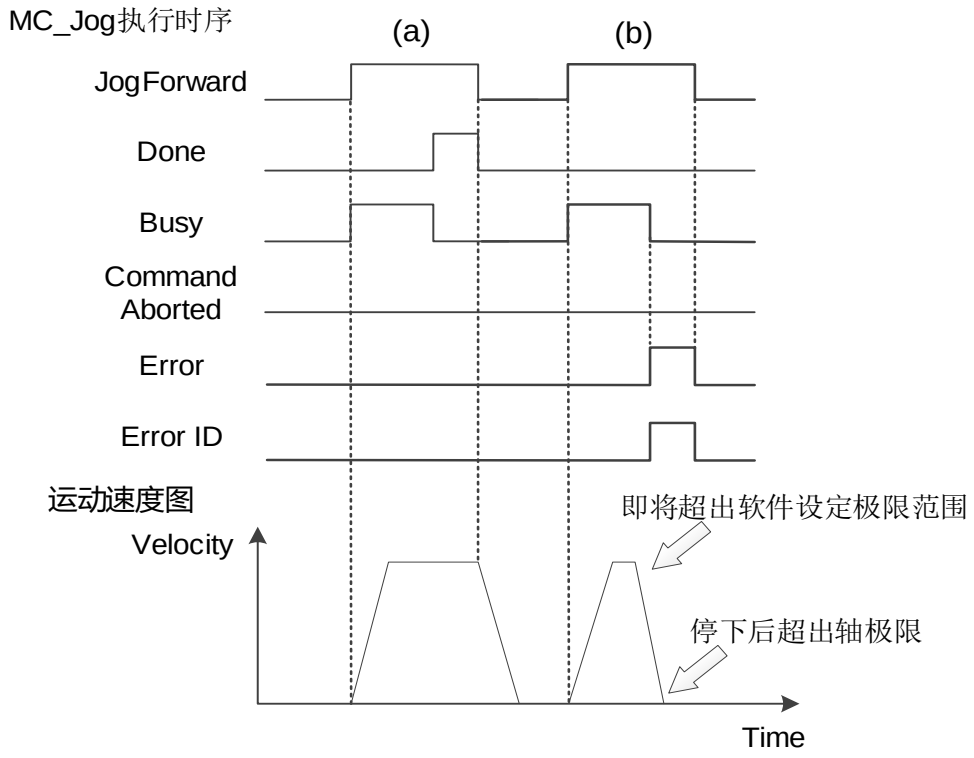
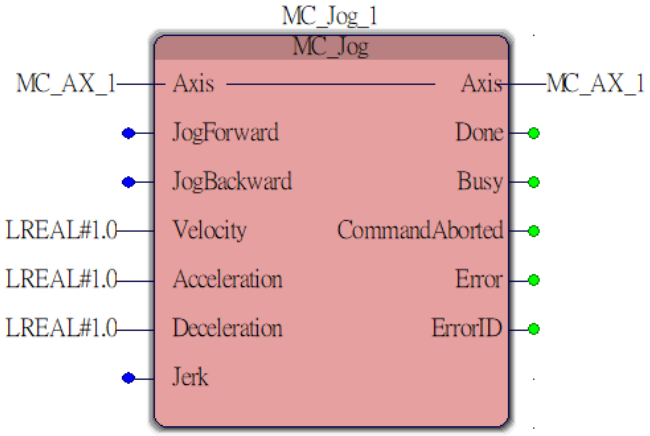


图 6.5.2.4.2 MC_Jog 速度时间图

■ 参考范例

此范例为 MC_Jog 的各项参数输入。



步骤如下:

1. 设定时动速度为 1.0 PUU/S
2. 设定时动加速度为 1.0 PUU/S^2
3. 设定时动减速度为 1.0 PUU/S^2
4. 设定时动加加速度为 0 PUU/S^3

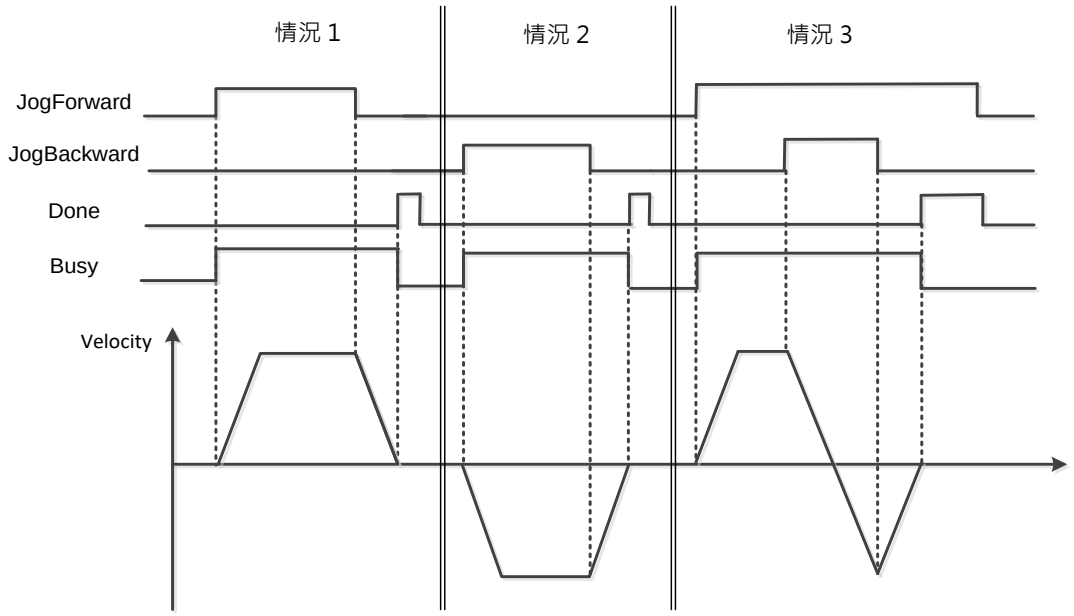


图 6.5.2.4.3 MC_Jog 脚位时序与轴速度图

情况 1 :当 JogForward 由 FALSE 变为 TRUE 时，Busy 同时变为 TRUE。轴将以 Acceleration 设定的加速度加速至 Velocity 所给定的速度并维持等速。当 JogForward 由 TRUE 变为 FALSE，轴速度将以 Deceleration 所设定的减速度减速至零。等待轴速度为零时，Busy 脚为由 TRUE 变为 FALSE，同时 Done 由 FALSE 变为 TRUE，维持一周期后变为 FALSE。

情况 2 :当 JogBackward 由 FALSE 变为 TRUE 时，Busy 同时变为 TRUE。轴将以 Deceleration 设定的减速度减速至 Velocity 脚位的负值（例：Velocity = 10，则减速至-10)并维持等速。当 JogBackward 由 TRUE 变为 FALSE，轴速度将以 Acceleration 所设定的加速度加速至零。等待轴速度为零时，Busy 脚为由 TRUE 变为 FALSE，同时 Done 由 FALSE 变为 TRUE，维持一周期后变为 FALSE。

情况 3 :当 JogForward 由 FALSE 变为 TRUE 时，Busy 同时变为 TRUE。轴将以 Acceleration 设定的加速度加速至 Velocity 所给定的速度并维持等速。当 JogBackward 由 FALSE 变为 TRUE 时，轴将以 Deceleration 设定的减速度减速，目标速度为 Velocity 脚位的负值(ex.Velocity = 10，则目标速度为-10)。减速的过程中，JogBackward 由 TRUE 变为 FALSE，轴速度将以 Acceleration 所设定的加速度加速至零。等待轴速度为零时，Busy 脚为由 TRUE 变为 FALSE，同时 Done 由 FALSE 变为 TRUE，Done 脚位将维持直到 JogForward 由 TRUE 变为 FALSE。

6.5.2.5 MC_MoveAbsolute

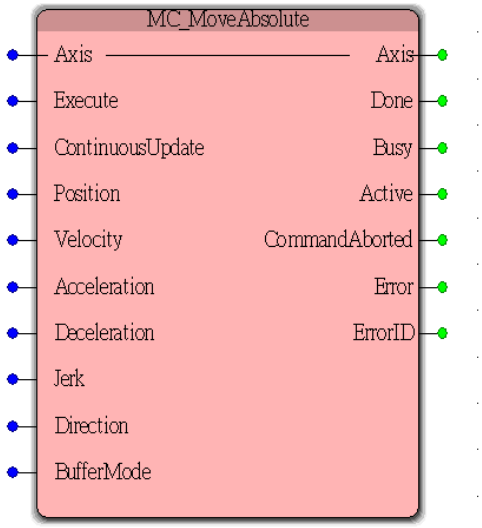
■ 类型

Function Block

■ 功能描述

依照用户所规划之速度、加速度、减速度等运动参数，移动到用户设定的绝对目标位置。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
Axis	AXIS_REF	-	指定运动轴编号

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute上升沿时，执行此指令
Continuous Update	BOOL	None	保留脚位
Position	LREAL	正数、负数或0.0 (0.0)	绝对位置 (用户单位)
Velocity	LREAL	正数 (0.0)	目标最大速度数值 (用户单位/秒)
Acceleration	LREAL	正数 (0.0)	加速度数值 (用户单位/秒^2)
Deceleration	LREAL	正数 (0.0)	减速度数值 (用户单位/秒^2)
Jerk	LREAL	正数或0.0 (0.0)	加加速度数值 (用户单位)/秒^3)
Direction	MC_DIRECTION	0 ~ 3 (0)	指定轴伺服马达运转方向 0：正向 1：最短距离 2：负向 3：目前方向

6

脚位名称	数据型态	设定值 (默认值)	功能
BufferMode	INT	0 ~ 5 (1)	指定此功能块的缓冲行为*注 0：中断(Aborting) 1：等待(Buffered) 2：以较低的速度作为中继速度(BlendingLow) 3：以前一笔指令的速度作为中继速度(BlendingPrevious) 4：以后一笔指令的速度作为交接(BlendingNext) 5：以较低的高速作为中继速度(BlendingHigh)

注：BufferMode 详细请参考 6.4.3 节

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	绝对寻址完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Active	BOOL	True / False (False)	指定轴受控制时为True
Command Aborted	BOOL	True / False (False)	指令被中断时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 详细说明请参考第9章

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	绝对寻址完成时	■ Execute转为False时 ■ 若Execute为False而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
Active	轴运动开始时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted 上升沿时

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
CommandAborted	■ 此功能块指令被其他缓冲模式设为Aborting的指令中断时 ■ 此功能块指令被 MC_Stop 中断时	■ 在Execute下降沿时 ■ 当Execute为False而CommandAborted转为True，此时CommandAborted维持一个扫描周期的True状态后，立刻转为False
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

■ 输出脚位的变化时序

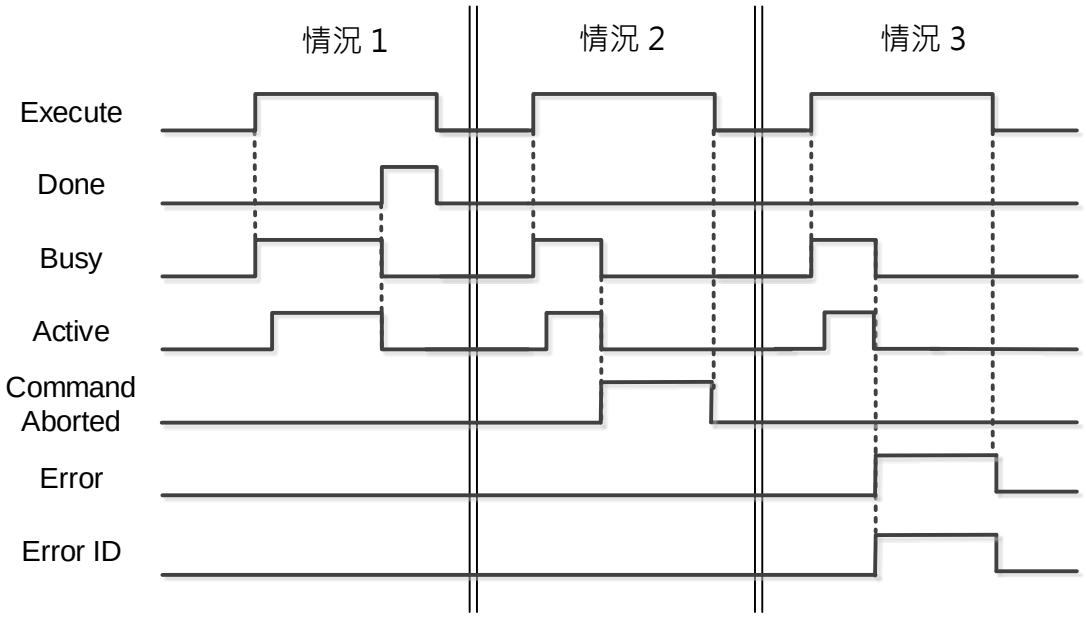


图 6.5.2.5.1 MC_MoveAbsolute 脚位时序图 (时序详细说明请参考 6.4.3 节)

情况 1：当此指令第一次执行(Execute 由 FALSE 变为 TRUE)时，Busy 脚位在同一时间变为 TRUE，表示指令已被执行。Active 由 FALSE 变为 TRUE 时，代表轴正在执行绝对寻址运动。目标位置到达时，Done 脚位变为 TRUE，同时 Busy 与 Active 变为 FALSE。当 Execute 由 TRUE 变为 FALSE 时，Done 同时变为 FALSE。

情况 2：当此指令执行的过程中，被 MC_Stop 或其他运动指令中断，此时 CommandAborted 脚位由 FALSE 变为 TRUE，同时 Busy 与 Active 脚位变为 FALSE，绝对寻址运动被中断。当 Execute 由 TRUE 变为 FALSE 时，CommandAborted 同时变为 FALSE。

情况 3：当此指令在执行过程中，若有错误产生，则 Error 变为 TRUE，同时 Busy 与 Active 变为 FALSE，此时请观察 ErrorID 脚位所显示的错误码，了解详细错误讯息。Error 与 ErrorID 脚位将一直维持直到 Execute 脚位变为 FALSE。

6

■ 参考范例

此范例示范透过 MC_MoveAbsolute 的各项参数输入，使轴移动至绝对位置 6000 PUU。

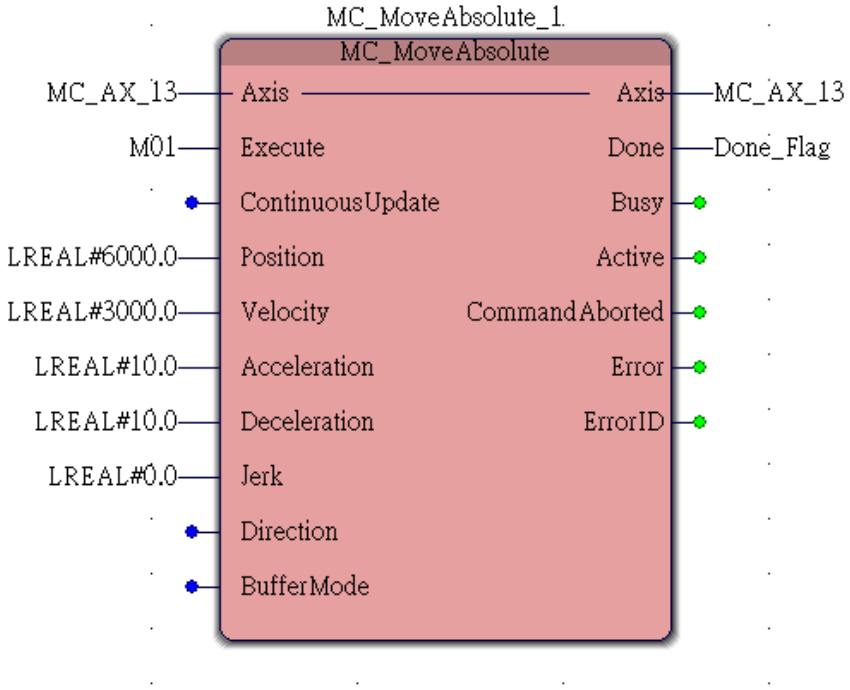


图 6.5.2.5.2 MC_MoveAbsolute 使用参考范例

1. 以 M01 触发此功能
2. 设定运动参数位置为 6000 PUU
3. 设定运动参数速度为 3000 PUU/S
4. 设定运动参数加速度为 10 PUU/S^2
5. 设定运动参数减速度为 10 PUU/S^2
6. 设定运动参数加加速度为 0 PUU/S^3

6.5.2.6 MC_MoveRelative

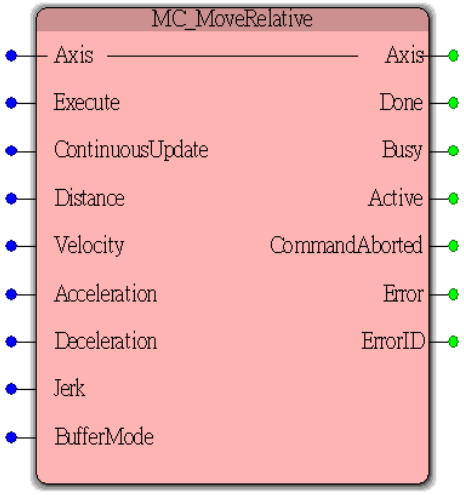
■ 类型

Function Block

■ 功能描述

依照用户者所规划的目标速度、加速度、减速度与加加速度等运动参数，移动到使用者设定的相对距离位置

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
Axis	AXIS_REF	-	指定运动轴编号

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute上升沿时，执行此指令
Continuous Update	BOOL	None	脚位保留
Distance	LREAL	正数、负数或0.0 (0.0)	相对位置(用户单位)
Velocity	LREAL	正数 (0.0)	目标最大速度数值 (用户单位/秒)
Acceleration	LREAL	正数 (0.0)	加速度数值(用户单位/秒^2)
Deceleration	LREAL	正数 (0.0)	减速度数值(用户单位/秒^2)
Jerk	LREAL	正数或0.0 (0.0)	加加速度数值 (用户单位/秒^3)
BufferMode	INT	0 ~ 5 (1)	指定此功能块的缓冲行为*注 0：中断(Aborting) 1：等待(Buffered) 2：以较低的速度作为中继速度(BlendingLow)

6

脚位名称	数据型态	设定值 (默认值)	功能
			3：以前一笔指令的速度作为中继速度(BlendingPrevious) 4：以后一笔指令的速度作为交接(BlendingNext) 5：以较低的高速作为中继速度(BlendingHigh)

注：BufferMode 详细请参考 6.4.3 节。

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	相对定位完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Active	BOOL	True / False (False)	指定轴受控制时为True
Command Aborted	BOOL	True / False (False)	指令被中断时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 详细说明请参考第9章

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	绝对寻址完成时	■ Execute转为False时 ■ 若Execute为False而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
Active	轴运动开始时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
Command Aborted	■ 此功能块指令被其他缓冲模式设为Aborting的指令中断时 ■ 此功能块指令被MC_Stop中断时	■ 在Execute下降沿时 ■ 当Execute为False，而CommandAborted转为True，此时CommandAborted维持一个扫描周期的True状态后，立刻转为False
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

■ 输出脚位的变化时序

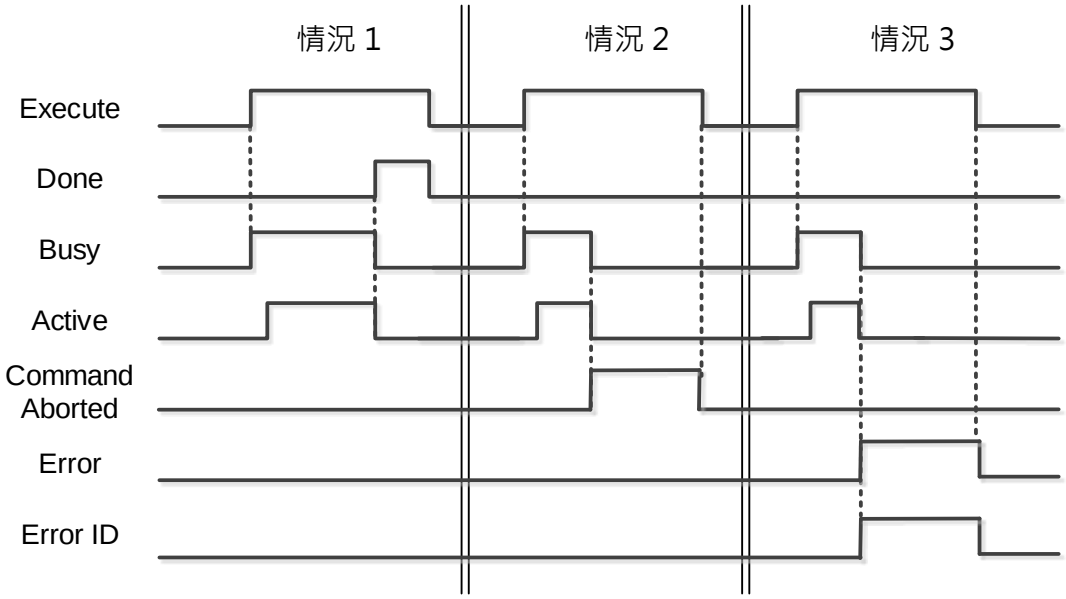


图 6.5.2.6.1 MC_MoveRelative 脚位时序图 (时序详细说明请参考 6.4.3 节)

情况 1：当此指令第一次执行(Execute 由 FALSE 变为 TRUE)时，Busy 脚位在同一时间变为 TRUE 表示指令已被执行。Active 由 FALSE 变为 TRUE 时，代表轴正在执行相对定位运动。目标位置到达时，Done 脚位变为 TRUE，同时 Busy 与 Active 变为 FALSE。当 Execute 由 TRUE 变为 FALSE 时，Done 同时变为 FALSE。

情况 2：当此指令执行的过程中，被 MC_Stop 或其他运动指令中断，此时 CommandAborted 脚位由 FALSE 变为 TRUE，同时 Busy 与 Active 脚位变为 FALSE，相对定位运动被中断。当 Execute 由 TRUE 变为 FALSE 时，CommandAborted 同时变为 FALSE。

情况 3：当此指令在执行过程中，若有错误产生，则 Error 变为 TRUE。同时 Busy 与 Active 变为 FALSE，此时请观察 ErrorID 脚位所显示的错误码，了解详细错误讯息。Error 与 ErrorID 脚位将一直维持直到 Execute 脚位变为 FALSE。

6

■ 参考范例

此范例示范透过输入 MC_MoveRelative 的各项参数，使轴由当前位置开始移动 6000 PUU。

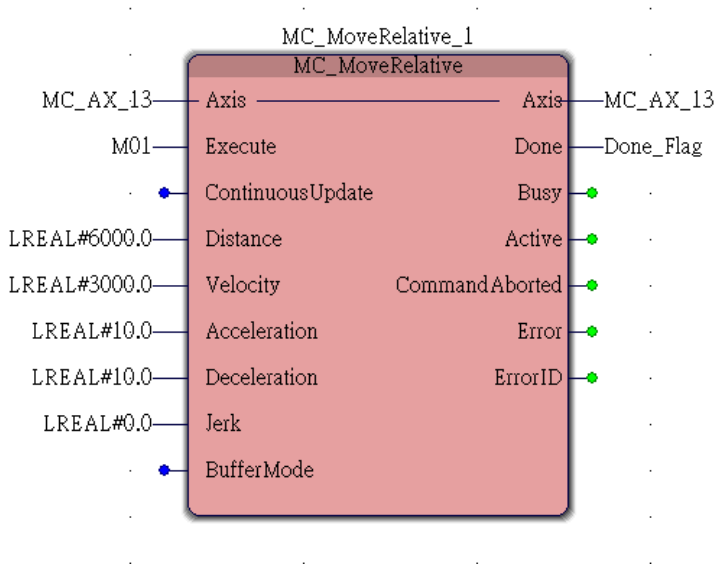


图 6.5.2.6.2 MC_MoveRelative 使用参考范例

1. 以 M01 触发此功能
2. 设定运动参数相对位置为 6000 PUU
3. 设定运动参数速度为 3000 PUU/S
4. 设定运动参数加速度为 10 PUU/S^2
5. 设定运动参数减速度为 10 PUU/S^2
6. 设定运动参数加加速度为 0 PUU/S^3

6.5.2.7 MC_MoveAdditive

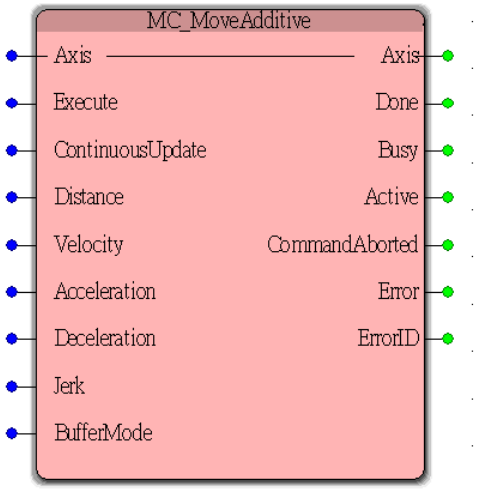
■ 类型

Function Block

■ 功能描述

依照用户输入的目标速度、加速度、减速度与加加速度，来执行一个相对距离运动命令，此命令的目标位置会附加到前一笔命令指定的目标位置上。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
Axis	AXIS_REF	-	指定运动轴编号

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute上升沿时，执行此指令
Continuous Update	BOOL	None	脚位保留
Distance	LREAL	正数，负数或0.0 (0.0)	相对位置 (用户单位)
Velocity	LREAL	正数 (0.0)	目标最大速度数值 (用户单位/秒)
Acceleration	LREAL	正数 (0.0)	加速度数值 (用户单位/秒^2)
Deceleration	LREAL	正数 (0.0)	减速度数值 (用户单位/秒^2)
Jerk	LREAL	正数或0.0 (0.0)	加加速度数值 (用户单位/秒^3)

6

脚位名称	数据型态	设定值 (默认值)	功能
BufferMode	INT	0 ~ 5 (1)	指定此功能块的缓冲行为*注 0：中断(Aborting) 1：等待(Buffered) 2：以较低的速度作为中继速度(BlendingLow) 3：以前一笔指令的速度作为中继速度(BlendingPrevious) 4：以后一笔指令的速度作为交接(BlendingNext) 5：以较低的高速作为中继速度(BlendingHigh)

注：BufferMode 详细请参考 6.4.3 节。

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	指令完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Active	BOOL	True / False (False)	指定轴受控制时为True
Command Aborted	BOOL	True / False (False)	指令被中断时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 详细说明请参考第9章

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	绝对寻址完成时	■ Execute转为False时 ■ 若Execute为False，而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
Active	轴运动开始时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
CommandAborted	■ 此功能块指令被其他缓冲模式设为Aborting的指令中断时 ■ 此功能块指令被MC_Stop中断时	■ 在Execute下降沿时 ■ 当Execute为False而CommandAborted转为True，此时CommandAborted维持一个扫描周期的True状态后，立刻转为False
Error / ErrorID	指令的执行条件或输入值发生错误时(错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

■ 输出脚位的变化时序

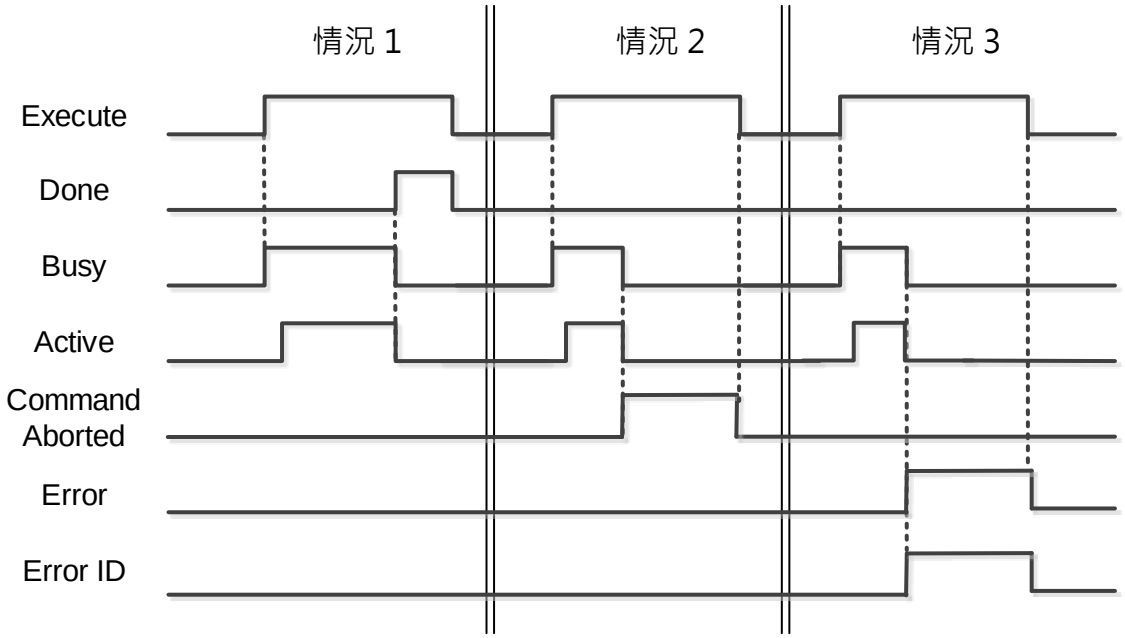


图 6.5.2.7.1 MC_MoveAdditive 脚位时序图 (时序详细说明请参考 6.4.3 节)

情况 1：当此指令第一次执行(Execute 由 FALSE 变为 TRUE)时，Busy 脚位在同一时间变为 TRUE，表示指令已被执行。Active 由 FALSE 变为 TRUE 时，代表轴正在执行迭加运动。目标位置到达时，Done 脚位变为 TRUE，同时 Busy 与 Active 变为 FALSE。当 Execute 由 TRUE 变为 FALSE 时，Done 同时变为 FALSE。

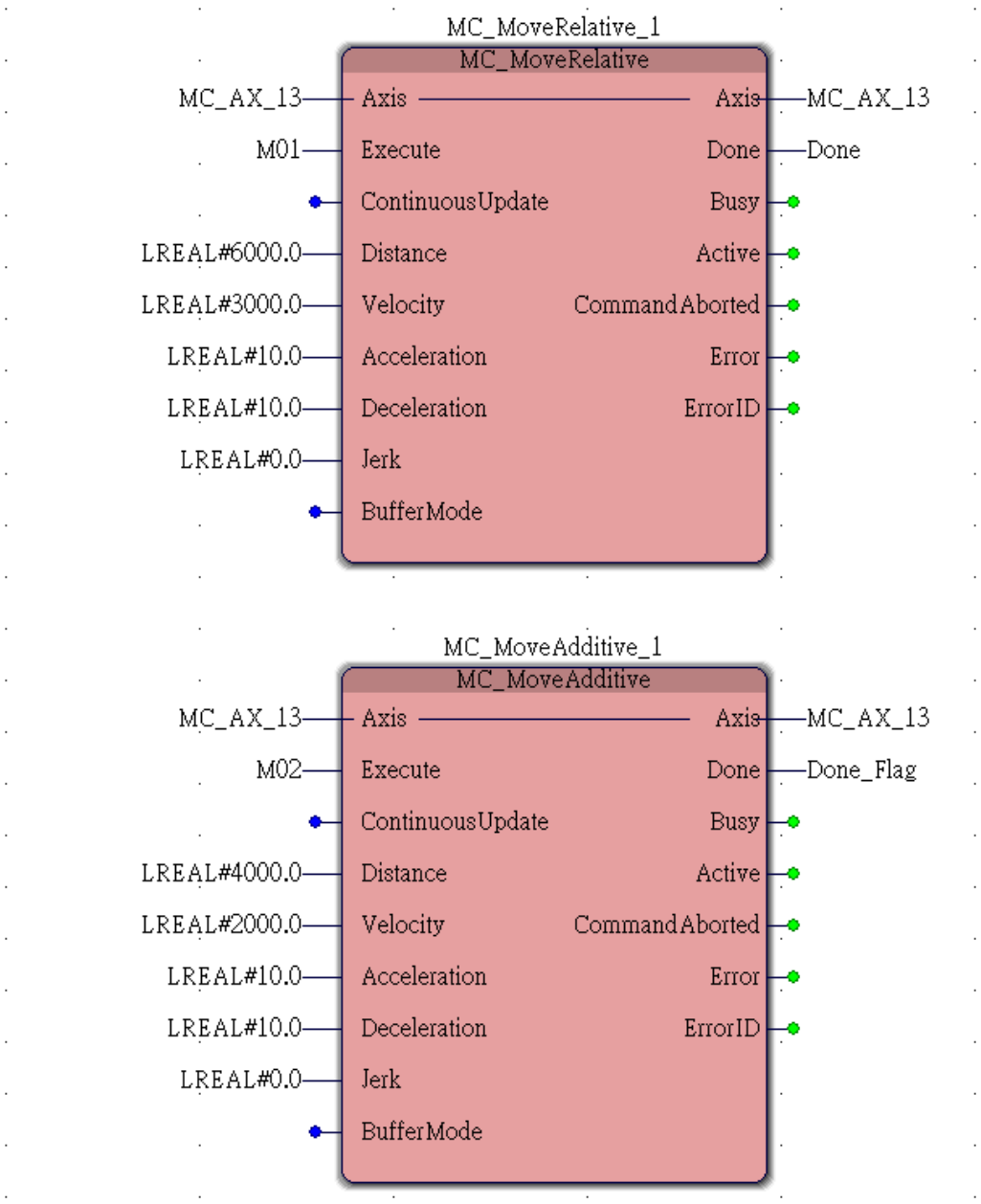
情况 2：当此指令执行的过程中，被 MC_Stop 或其他运动指令中断，此时 CommandAborted 脚位由 FALSE 变为 TRUE，同时 Busy 与 Active 脚位变为 FALSE，MC_MoveAdditive 运动被中断。当 Execute 由 TRUE 变为 FALSE 时，CommandAborted 同时变为 FALSE。

情况 3：当此指令在执行过程中有错误产生，则 Error 变为 TRUE。同时 Busy 与 Active 变为 FALSE，此时请观察 ErrorID 脚位所显示的错误码，了解详细错误讯息。Error 与 ErrorID 脚位将一直维持直到 Execute 脚位变为 FALSE。

6

■ 参考范例

以下范例中，在触发 M01 执行了第一个功能块 MC_MoveRelative 后，当速度到达 3000，再触发 M02 执行 MC_MoveAdditive 功能块来附加一段移动距离。



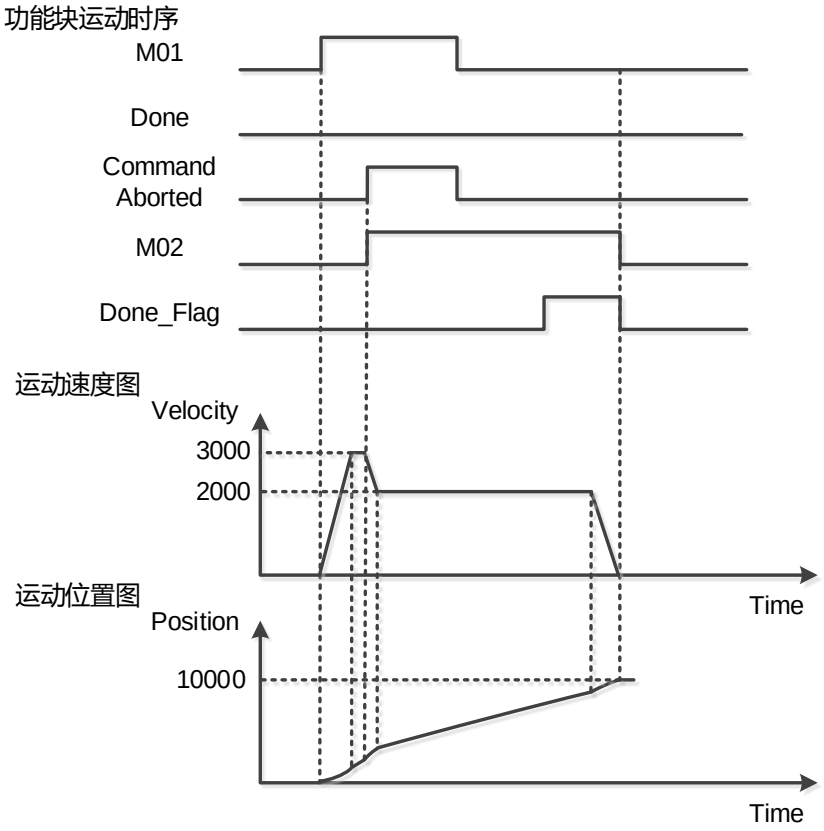


图 6.5.2.7.2 MC_MoveAdditive 使用参考范例

6

6.5.2.8 MC_MoveSuperimposed

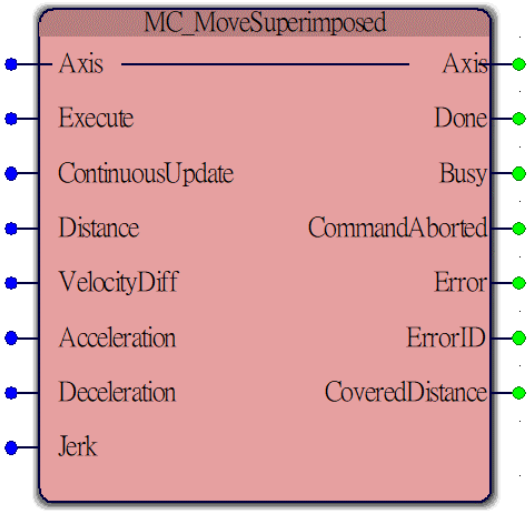
■ 类型

Function Block

■ 功能描述

依照用户输入的目标速度、加速度、减速度与加加速度，迭加设定距离在当前运动命令上。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
Axis	AXIS_REF	-	指定运动轴编号

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute上升沿时，执行此指令
Continuous Update	BOOL	None	脚位保留
Distance	LREAL	负数、正数或0 (0)	相对位置 (用户单位)
VelocityDiff	LREAL	正数 (0)	目标最大速度数值 (用户单位/秒)
Acceleration	LREAL	正数 (0)	加速度数值 (用户单位/秒^2)
Deceleration	LREAL	正数 (0)	减速度数值 (用户单位/秒^2)
Jerk	LREAL	正数或0 (0)	加加速度数值 (用户单位/秒^3)

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	绝对寻址完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Command Aborted	BOOL	True / False (False)	指令被中断时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 错误码详细说明请参考第9章
Covered Distance	LREAL	负数、正数或0 (0)	连续显示自此指令启动后所涵盖的移动距离

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	绝对寻址完成时	■ Execute转为False时 ■ 若Execute为False而Done 转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted 上升沿时
Active	轴运动开始时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
Command Aborted	■ 此功能块指令被其他缓冲模式设为Aborting的指令中断时 ■ 此功能块指令被 MC_Stop中断时	■ 在Execute下降沿时 ■ 当Execute为False而CommandAborted转为True，此时CommandAborted维持一个扫描周期的True状态后，立刻转为False
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

6

■ 输出脚位的变化时序

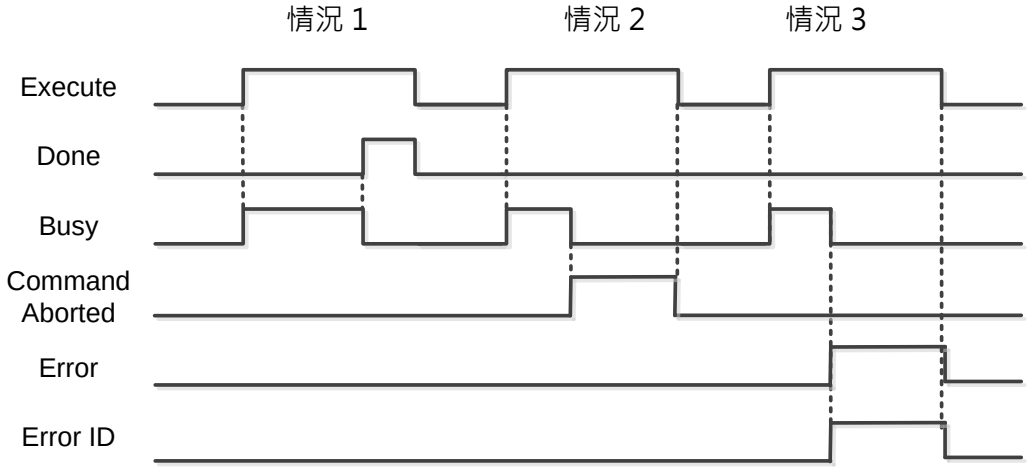


图 6.5.2.8.1 MC_MoveSuperimposed 脚位时序图 (时序详细说明请参考 6.4.3 节)

情况 1：当此指令第一次执行(Execute 由 FALSE 变为 TRUE)时，Busy 脚位在同一时间变为 TRUE 表示轴正在执行迭加运动。目标位置到达时，Done 脚位变为 TRUE，同时 Busy 变为 FALSE。当 Execute 由 TRUE 变为 FALSE 时，Done 同时变为 FALSE。

情况 2：当此指令执行的过程中，被 MC_Stop 或其他运动指令中断，此时 CommandAborted 脚位由 FALSE 变为 TRUE，同时 Busy 脚位变为 FALSE，迭加运动被中断。当 Execute 由 TRUE 变为 FALSE 时，CommandAborted 同时变为 FALSE。

情况 3：当此指令在执行过程中有错误产生，则 Error 变为 TRUE。同时 Busy 变为 FALSE，此时请观察 ErrorID 脚位所显示的错误码，了解详细错误讯息。Error 与 ErrorID 脚位将一直维持直到 Execute 脚位变为 FALSE。

■ 参考范例

以下范例中，在触发 Execute 执行了第一个功能块 MC_MoveRelative 后，当位置已到达 5000，再触发 Execute 执行第二个 MC_MoveSuperimposed 功能块来迭加一段移动距离。

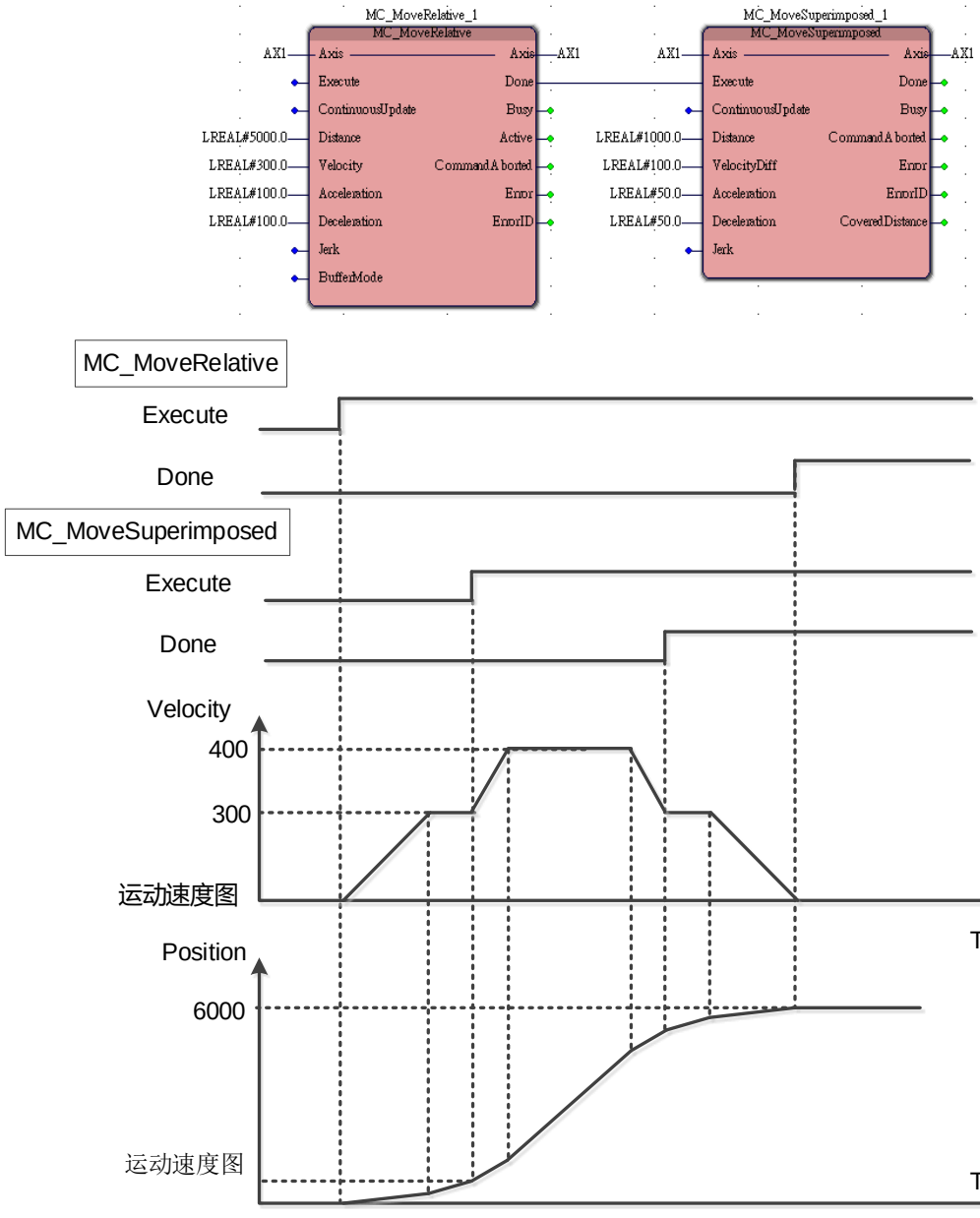


图 6.5.2.8.2 MC_MoveSuperimposed 使用参考范例

6

6.5.2.9 MC_SetPosition

■ 类型

Function Block

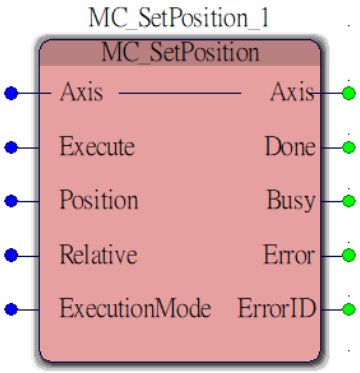
■ 功能描述

依照输入的指定位置来设定轴的当前位置。

注:

1. 建议在轴停止时启用。
2. 请勿在 CamIn 指令执行中的状态下对主轴启用本功能块，以避免不可预期的状况。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
Axis	AXIS_REF	-	指定运动轴编号

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute上升沿时，执行此指令
Position	LREAL	正数、负数或 0.0 (0.0)	默认定之位置
Relative	BOOL	True / False (False)	若为True，则Position为相对位置；反之，则Position为绝对位置
Execution Mode	INT	0 ~ 1 (0)	0：中断 1：等待

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	定位完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 详细说明请参考第9章

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Enabled	设定值成立后	■ Execute转为False时 ■ 若Execute为False而Done 转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
Error / ErrorID	指令的执行条件或输入值发生错误时(错误码记录在ErrorID)	在Execute下降沿时(清除ErrorID记录之错误码)

■ 输出脚位的变化时序

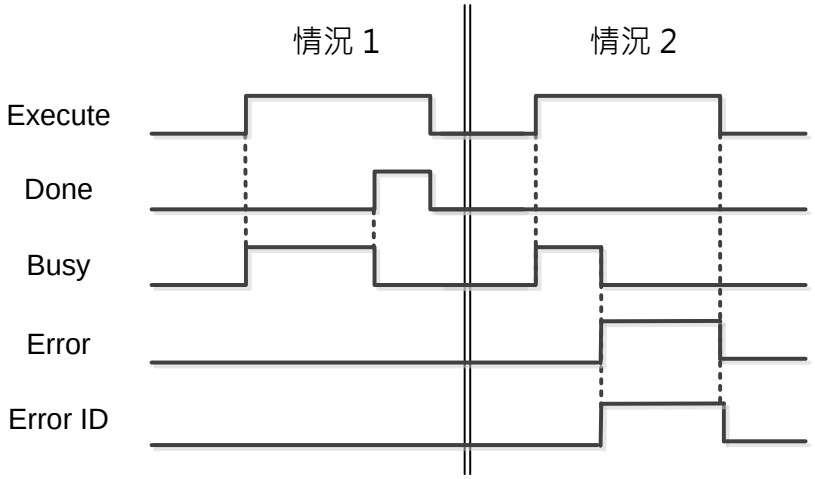


图 6.5.2.9.1 MC_SetPosition 脚位时序图 (时序详细说明请参考 6.4.3 节)

情况 1：当此指令第一次执行(Execute 由 FALSE 变为 TRUE)时，Busy 脚位在同一时间变为 TRUE 表示指令已被执行。轴位置成功被设定时，Done 脚位变为 TRUE，同时 Busy 变为 FALSE。当 Execute 由 TRUE 变为 FALSE 时，Done 同时变为 FALSE。

情况 2：当此指令执行的过程中发生错误(轴不在停止状态、脚位输入错误等等…)，则 Error 变为 TRUE，同时 Busy 变为 FALSE，此时请观察 ErrorID 脚位所显示的错误码，了解详细错误讯息。Error 与 ErrorID 脚位将一直维持直到 Execute 脚位变为 FALSE。

6

■ 参考范例

此范例提供了 MC_SetPosition 功能块的 Relative 脚位为绝对和相对模式时候，触发 MC_SetPosition 功能块之后的轴位置结果。

第一个 MC_SetPosition 功能块为绝对模式（Relative 为 False）：

在触发 Ex1 后，伺服 AX1 轴不进行移动，但是控制器将当前位置设定为 1000.0。

第二个 MC_SetPosition 功能块为相对模式（Relative 为 True）：

在触发 Ex2 后，伺服 AX1 轴不进行移动，但是控制器将当前位置设定为 2000.0。

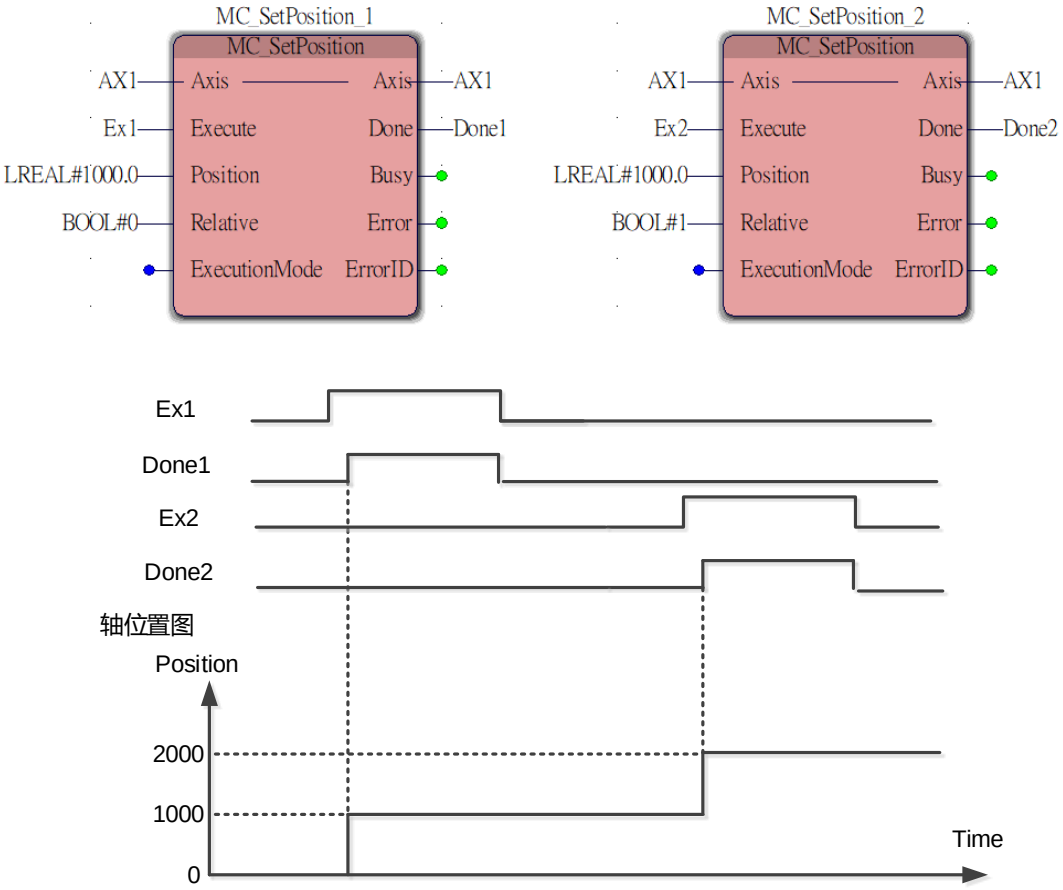


图 6.5.2.9.2 MC_SetPosition 使用参考范例

6.5.2.10 DMC_Home

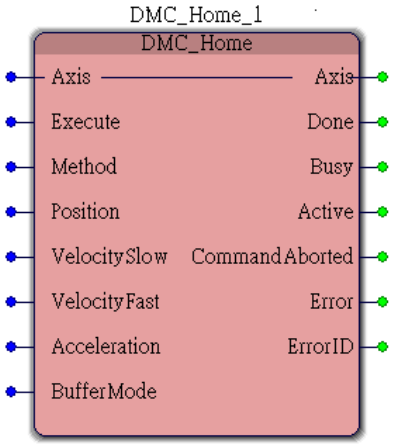
■ 类型

Function Block

■ 功能描述

根据功能块所设定的参数启动原点复归流程。例如：寻找极限、传感器和扭力等模式来执行回原流程。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
Axis	AXIS_REF	-	指定运动轴编号

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	当Execute为上升缘时，此功能即开始执行
Method	USINT	0 ~ 35 (0)	设定回原模式
Position	LREAL	正数、负数或0.0 (0.0)	原点距离目前位置的偏差值
VelocitySlow	LREAL	正数 (0.0)	慢速段搜寻Z脉冲之速度 (第二段复归速度)，该速度最大值视伺服机种而定 (用户单位/秒)
VelocityFast	LREAL	正数 (0.0)	快速段搜寻原点开关之速度，速度最大值视伺服机种而定 (用户单位/秒)
Acceleration	LREAL	正数 (0.0)	加速度数值，最大值视伺服机种而定(用户单位/秒^2)
BufferMode	INT	None	保留

6

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	回原完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Active	BOOL	True / False (False)	指定轴受控制时为True
Command Aborted	BOOL	True / False (False)	指令被中断时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000~0xFFFF (0)	指令错误发生时记录的错误码 详细说明请参考第9章

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	绝对寻址完成时	■ Execute转为False时 ■ 若Execute为False而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
Active	轴运动开始时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
Command Aborted	此功能块指令被MC_Stop中断时	■ 在Execute下降沿时 ■ 在Execute为False而CommandAborted转为True，此时CommandAborted维持一个扫描周期的True状态后，立刻转为False
Error / ErrorID	指令的执行条件或输入值发生错误时（错误码记录在ErrorID ）	在Execute下降沿时 (清除ErrorID记录之错误码)

■ 输出脚位的变化时序

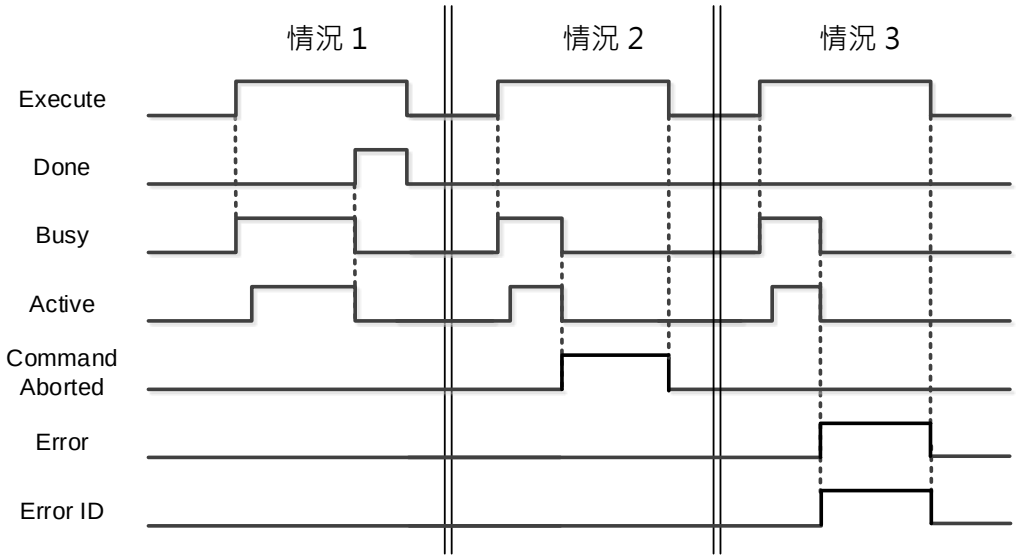


图 6.5.2.10.1 DMC_Home 脚位时序图 (时序详细说明请参考 6.4.3 节)

情况 1：当此指令第一次执行(Execute 由 FALSE 变为 TRUE)时，Busy 脚位在同一时间变为 TRUE 表示指令已被执行，Active 由 FALSE 变为 TRUE 时代表轴正在执行回原流程，等待回原流程完成时，Done 脚位变为 TRUE，同时 Busy 与 Active 变为 FALSE。当 Execute 由 TRUE 变为 FALSE 时，Done 同时变为 FALSE。

情况 2：当此指令执行的过程中，执行 MC_Stop 命令，此时 CommandAborted 脚位由 FALSE 变为 TRUE，同时 Busy 与 Active 脚位变为 FALSE，回原流程被中断。当 Execute 由 TRUE 变为 FALSE 时，CommandAborted 同时变为 FALSE。

情况 3：当此指令在执行过程中有错误产生时，则 Error 变为 TRUE，同时 Busy 与 Active 变为 FALSE，此时请观察 ErrorID 脚位所显示的错误码，了解详细错误讯息。Error 与 ErrorID 脚位将一直维持直到 Execute 脚位变为 FALSE。

6

■ 参考范例

此范例提供了 DMC_Home 功能块使用回原模式 1 来进行回原动作。

在触发 Ex1 后，伺服 MC_AX_1 轴会先以速度 200.0 PUU/s 反转寻找硬件极限，遇反向硬件极限后，再以速度 50.0 PUU/s 正转，找到第一个 Z 脉冲为原点。

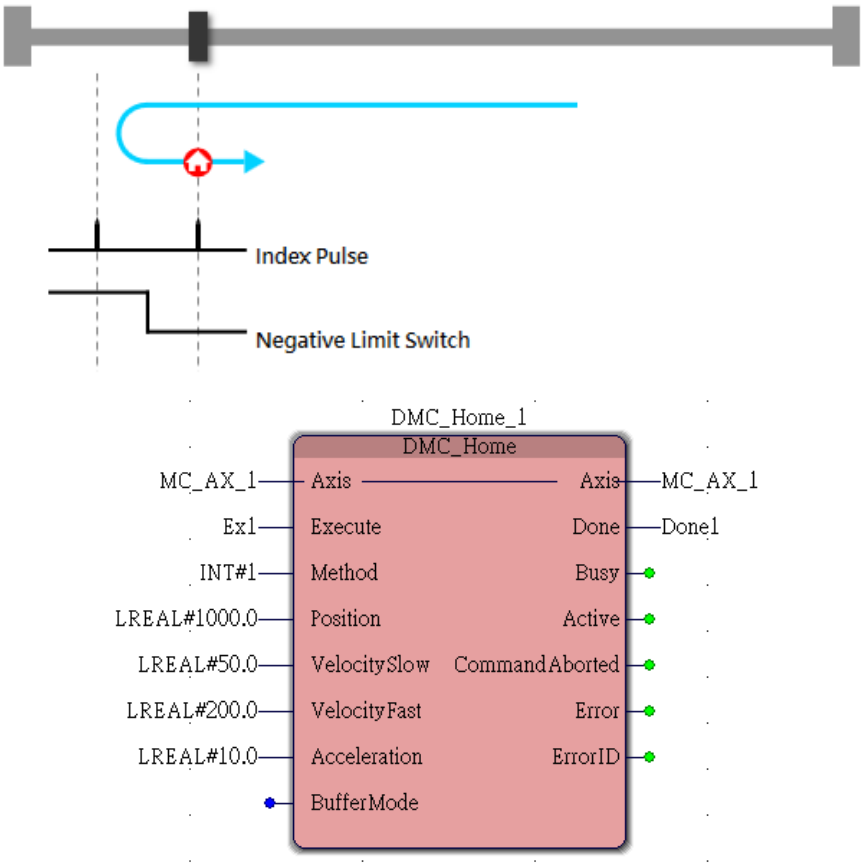


图 6.5.2.10.2 DMC_Home 使用参考范例

6.5.2.11 MC_MoveVelocity

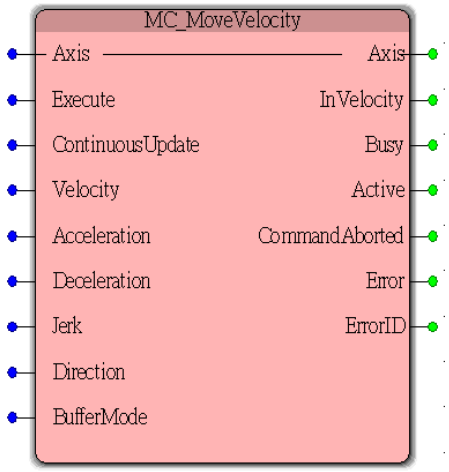
■ 类型

Function Block

■ 功能描述

使控制轴按照设定的加减速运动至目标速度，并等速运行。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
Axis	AXIS_REF	-	指定运动轴编号

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute上升沿时，执行此指令
Continuous Update	BOOL	None	保留脚位
Velocity	LREAL	正数或0.0 (0.0)	目标最大速度数值 (用户单位/秒)
Acceleration	LREAL	正数 (0.0)	加速度数值 (用户单位/秒^2)
Deceleration	LREAL	正数 (0.0)	减速度数值 (用户单位/秒^2)
Jerk	LREAL	正数或0.0 (0.0)	加加速度数值 (用户单位/秒^3)
Direction	MC_DIRECTION	0 ~ 3 (0)	指定轴伺服马达运转方向 0：正向 1：最短距离 (不支持) 2：负向 3：目前方向

6

脚位名称	数据型态	设定值 (默认值)	功能
BufferMode	INT	0 ~ 5 (0)	指定此功能块的缓冲行为(注) 0：中断(Aborting) 1：等待(Buffered) 2：以较低的速度作为中继速度(BlendingLow) 3：以前一笔指令的速度作为中继速度(BlendingPrevious) 4：以后一笔指令的速度作为交接(BlendingNext) 5：以较低的高速作为中继速度(BlendingHigh)

注：BufferMode 详细请参考 6.4.3 节。

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
InVelocity	BOOL	True / False (False)	达到目标速度时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Active	BOOL	True / False (False)	指定轴受控制时为True
Command Aborted	BOOL	True / False (False)	指令被中断时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 详细说明请参考第9章

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
InVelocity	速度达到设定值时	■ 当错误发生时 ■ 当中断发生时 ■ 当执行MC_Stop命令时
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
Active	轴运动开始时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
Command Aborted	■ 此功能块指令被其他缓冲模式设为Aborting的指令中断时 ■ 此功能块指令被MC_Stop中断时	■ 在Execute下降沿时 ■ 当Execute为False而CommandAborted转为True，此时CommandAborted维持一个扫描周期的True状态后，立刻转为False

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

■ 输出脚位的变化时序

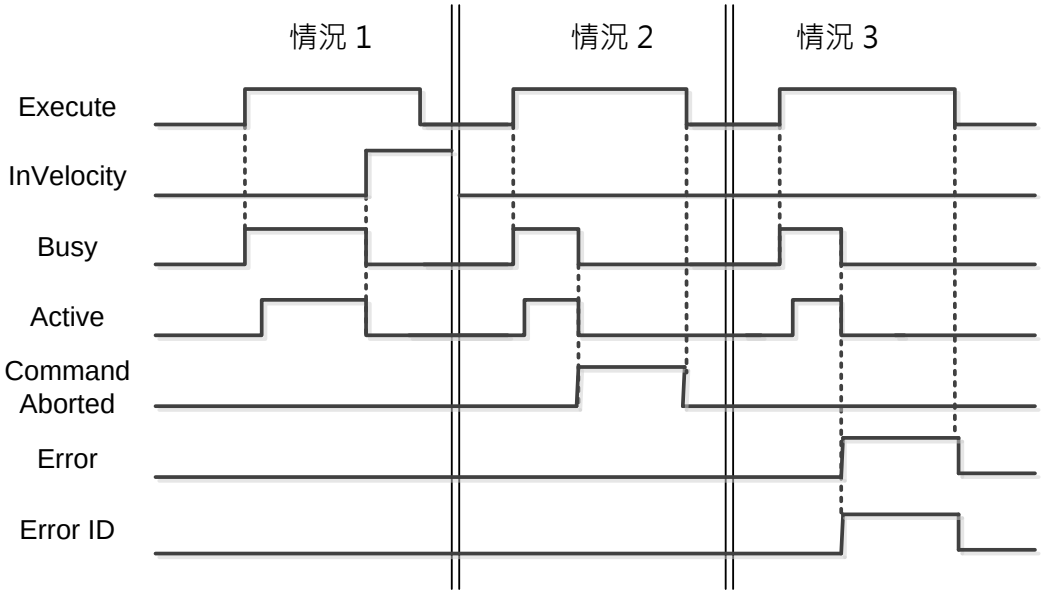


图 6.5.2.11.1 MC_MoveVelocity 脚位时序图 (时序详细说明请参考 6.4.3 节)

情况 1：当此指令第一次执行(Execute 由 FALSE 变为 TRUE)时，Busy 脚位在同一时间变为 TRUE 表示指令已被执行，Active 由 FALSE 变为 TRUE 时代表轴正在执行定速运动，目标速度到达时，InVelocity 脚位变为 TRUE，同时 Busy 与 Active 变为 FALSE。

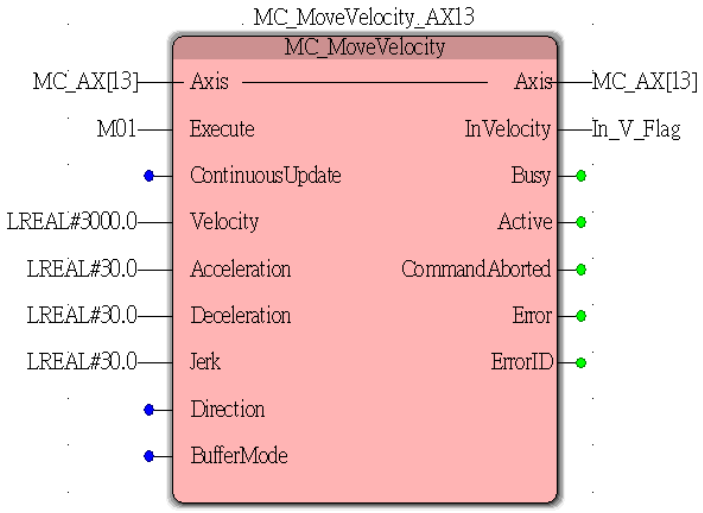
情况 2：当此指令执行的过程中被 MC_Stop 或其他运动指令中断，此时 CommandAborted 脚位由 FALSE 变为 TRUE，同时 Busy 与 Active 脚位变为 FALSE，MC_MoveVelocity 运动被中断。当 Execute 由 TRUE 变为 FALSE 时，CommandAborted 同时变为 FALSE。

情况 3：当此指令在执行过程中有错误产生，则 Error 变为 TRUE，同时 Busy 与 Active 变为 FALSE，此时请观察 ErrorID 脚位所显示的错误码，了解详细错误讯息。Error 与 ErrorID 脚位将一直维持直到 Execute 脚位变为 FALSE。

6

■ 参考范例

在此范例中，透过 MC_MoveVelocity 的命令下达轴的目标速度，与利用其他功能块插断，使轴减速至停止。



功能块时序

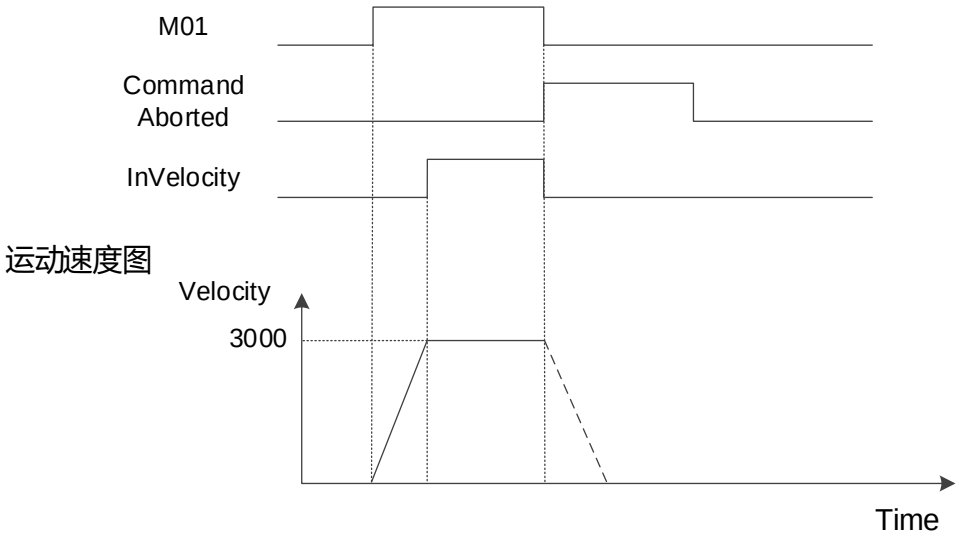


图 6.5.2.11.2 MC_MoveVelocity 使用参考范例

6.5.2.12 MC_Power

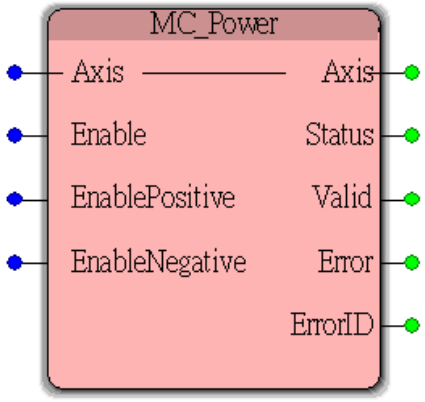
■ 类型

Function Block

■ 功能描述

控制指定轴进行伺服使能或关闭。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
Axis	AXIS_REF	-	指定运动轴编号

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Enable	BOOL	True / False (False)	Enable为True时，启用此功能
Enable Positive	BOOL	None	保留
Ebable Negative	BOOL	None	保留

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Status	BOOL	True / False (False)	伺服状态，伺服启动时为True；反之则为False
Valid	BOOL	True / False (False)	成立时表示功能块有效。
Error	BOOL	True / False (False)	错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码详细说明请参考第9章

6

■ 输出脚位的变化时序

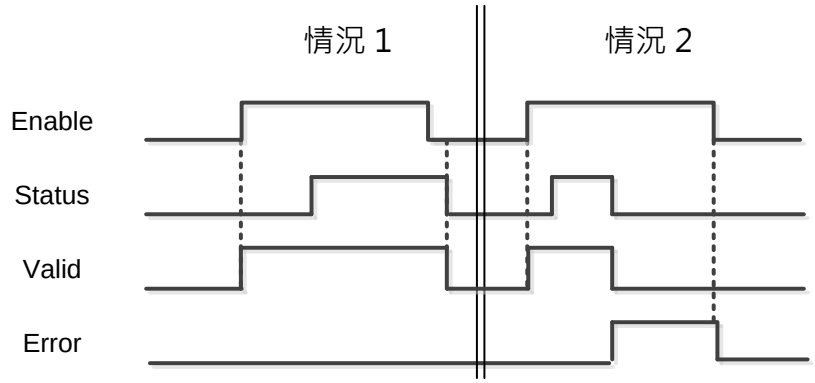


图 6.5.2.12.1 MC_Power 脚位时序图 (时序详细说明请参考 6.4.3 节)

情况 1：当此指令第一次执行(Enable 脚位由 FALSE 变为 TRUE)时，Valid 脚位与 Enable 脚位同时为 TRUE，等待伺服成功使能(1 周期以上)，Status 为 TRUE。当 Enable 脚位由 TRUE 变为 FALSE，Valid 与 Status 脚位维持一周期之后便为 FALSE，此时伺服解除使能。

情况 2：当此指令在执行过程中有错误产生，则 Error 变为 TRUE，同时 Valid 与 Status 变为 FALSE，此时请观察 ErrorID 脚位所显示的错误码，了解详细错误讯息。Error 与 ErrorID 脚位将一直维持直到 Enable 脚位变为 FALSE。

■ 参考范例

此范例示范透过外部命令来控制伺服启动功能。

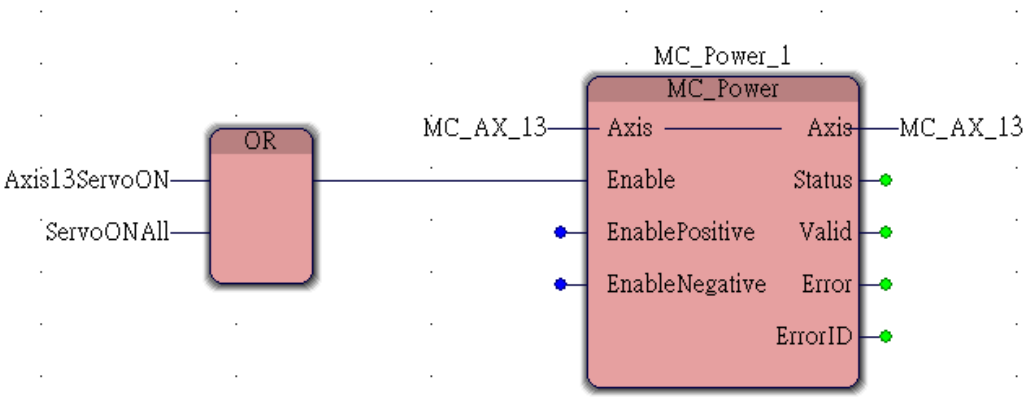


图 6.5.2.12.2 MC_Power 使用参考范例

1. Axis13ServoON 连接外部命令(如数字输入脚位或内存位置)。
2. ServoONAll 连接外部命令(如数字输入脚位或内存位置)。

6.5.2.13 MC_Reset

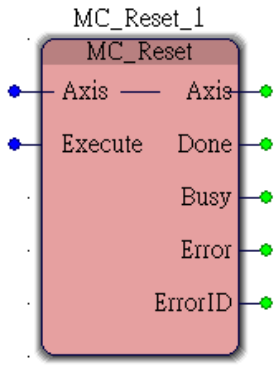
■ 类型

Function Block

■ 功能描述

清除或重置所有指定轴相关的错误。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
Axis	AXIS_REF	-	指定运动轴编号

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute上升沿时，执行此指令

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	绝对寻址完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 详细说明请参考第9章

6

■ 输出脚位的变化时序

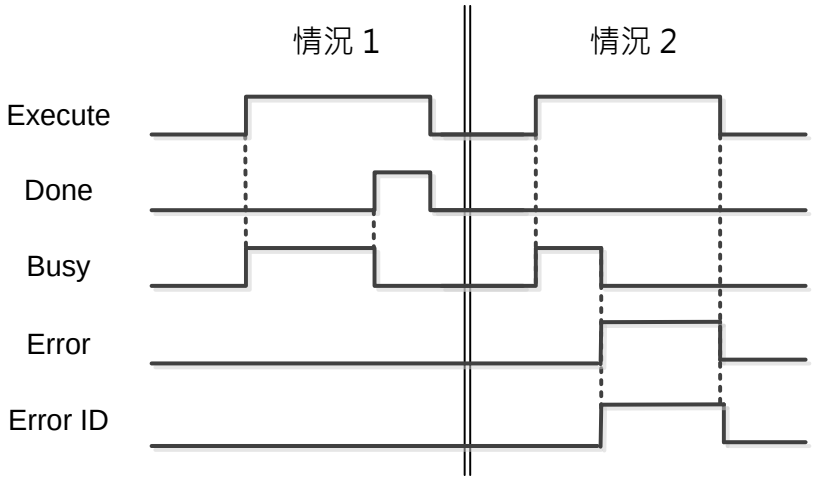


图 6.5.2.13.1 MC_Reset 脚位时序图 (时序详细说明请参考 6.4.3 节)

情况 1：当此指令第一次执行(Execute 由 FALSE 变为 TRUE)时，Busy 在同一时间变为 TRUE，表示指令已被执行。当轴成功执行错误清除后，Done 脚位变为 TRUE。当 Enable 由 TRUE 变为 FALSE 时，Done 维持一个周期后变为 FALSE。

情况 2：当此指令执行的过程中发生错误，则 Error 变为 TRUE，同时 Busy 变为 FALSE，此时请观察 ErrorID 脚位所显示的错误码，了解详细错误讯息。Error 与 ErrorID 脚位将一直维持 TRUE 直到 Execute 脚位变为 FALSE。

■ 参考范例

此范例示范透过外部讯号清除轴错误。

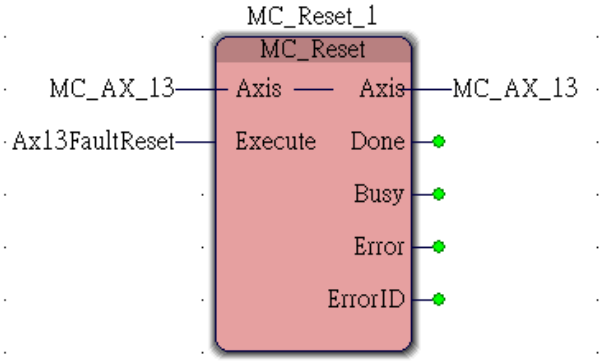


图 6.5.2.13.2 MC_Reset 使用参考范例

将外部讯号命名为 Ax13FaultReset，触发 Ax13FaultReset 时，伺服 MC_AX_13 轴会清除轴错误。

6.5.2.14 MC_SetOverride

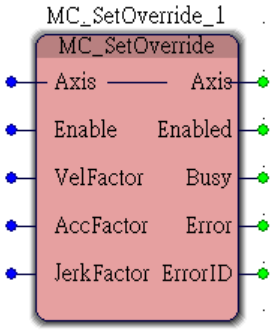
■ 类型

Function Block

■ 功能描述

透过超驰控制(override control)系数并依加(减)速度或是加加速度来改变轴的速度。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值(默认值)	功能
Axis	AXIS_REF	-	指定运动轴编号

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Enable	BOOL	True / False (False)	Enable为True时，启用此功能
VelFactor	REAL	正数或0 (1.0)	超驰控制速度系数
AccFactor	REAL	正数 (1.0)	超驰控制系数，正数为加速度系数，负数为减速度系数
JerkFactor	REAL	正数 (1.0)	超驰控制加加速度系数

■ 输出脚位

脚位名称	数据型态	设定值(默认值)	功能
Enabled	BOOL	True / False (False)	True时表示超驰控制被设立成功
Busy	BOOL	True / False (False)	指令执行时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	错误发生时记录的错误码 详细说明请参考第9章

6

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Enabled	设定值成立后	■ Execute转为False时 ■ 若Execute为False而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
Error / ErrorID	当指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

■ 输出脚位的变化时序

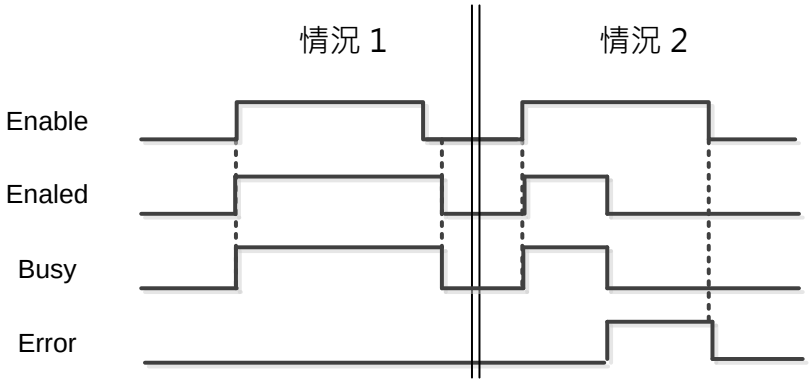


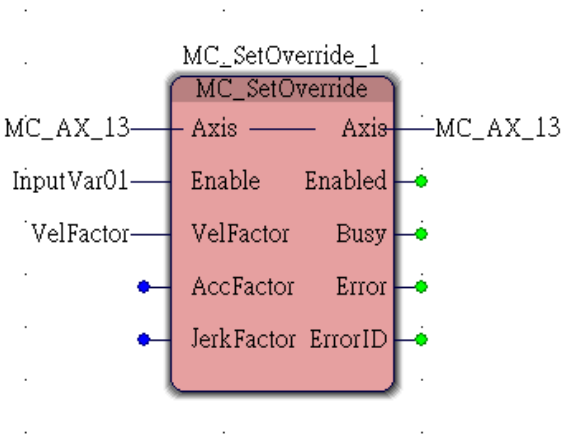
图 6.5.2.14.1 MC_SetOverride 脚位时序图 (时序详细说明请参考 6.4.3 节)

情况 1：当此指令第一次执行(Enable 由 FALSE 变为 TRUE)时，Busy 与 Enabled 脚位在同一时间变为 TRUE 表示指令已被执行。当 Enable 由 TRUE 变为 FALSE 时，Done 维持一个周期后变为 FALSE。

情况 2：当此指令执行的过程中发生错误，则 Error 变为 TRUE，同时 Busy 变为 FALSE，此时请观察 ErrorID 脚位所显示的错误码，了解详细错误讯息。Error 与 ErrorID 脚位将一直维持 TRUE 直到 Enabled 脚位变为 FALSE。

■ 参考范例

此范例示范透过外部讯号控制，使运动速度的比例值生效。



功能块时序图

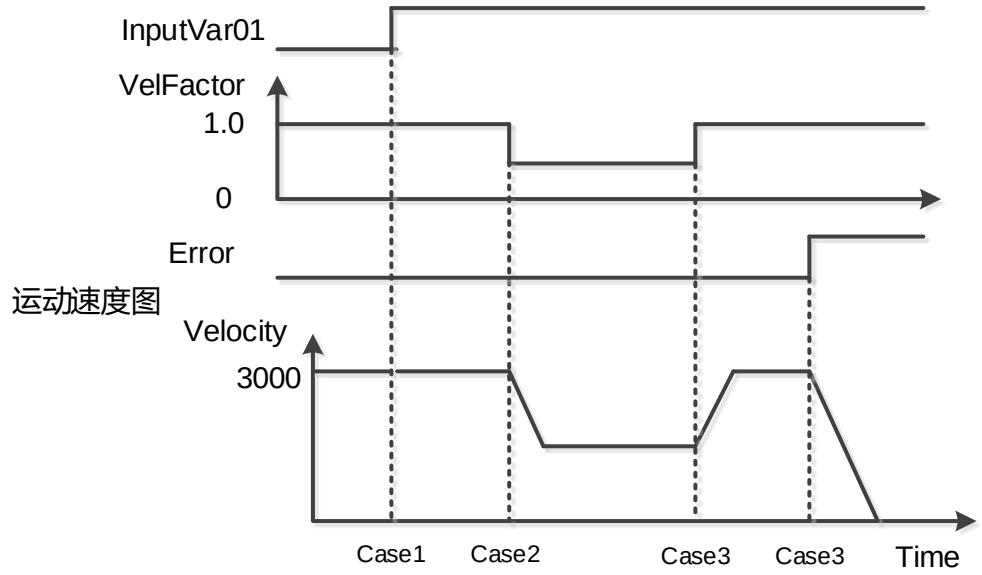


图 6.5.2.14.2 MC_SetOverride 使用参考范例

1. InputVar01 连接外部命令 (如数字输入脚位或内存位置)。
2. VelFactor 连接外部命令 (如内存位置)。
3. VelFactor 变动影响轴运动速度。

   Case1：VelFactor = 1，轴速度维持 100%

   Case2：VelFactor = 0.5，轴速度降速为 50%

   Case3：VelFactor = 1，轴速度升速至 100%

   Case4：VelFactor = 0，轴速度降速至 0%
4. 轴速度因为错误产生，导致不会变更至 100%而保持在 0%。

6

6.5.2.15 MC_ReadParameter

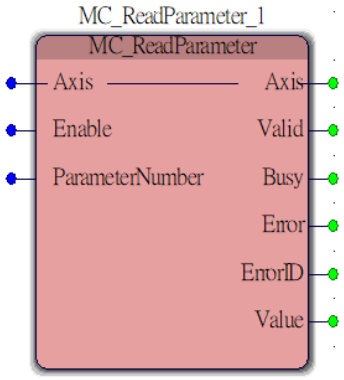
■ 类型

Function Block

■ 功能描述

读取指定轴的信息。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
Axis	AXIS_REF	-	指定轴编号

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute上升沿时，执行此指令
Parameter Number	INT	输入范围请参考下表 (0)	参数编号

■ 参数编号(Parameter Number)说明

Parameter Number	名称	数据型态	R/W	详细说明
1	位置命令	LREAL	R	读取位置命令
2	正极限位置	LREAL	R/W	读写正极限位置
3	负极限位置	LREAL	R/W	读写负极限位置
4	正极限开关	BOOL	R/W	读写正极限开关
5	负极限开关	BOOL	R/W	读写负极限开关
6	位置误差监控开关	BOOL	R/W	读写位置误差监控开关
7	最大位置误差	LREAL	R/W	读写最大位置误差
8	最大速度	LREAL	R	读取最大速度(用户单位)
9	最大速度	LREAL	R/W	读写最大速度(用户单位)
10	速度回授	LREAL	R	读取速度回授(用户单位)
11	速度命令	LREAL	R	读取速度命令(用户单位)
12	最大加速度	LREAL	R	读取最大加速度(用户单位)

Parameter Number	名称	数据型态	R/W	详细说明
13	最大加速度	LREAL	R/W	读写最大加速度(用户单位)
14	最大减速度	LREAL	R	读取最大减速度(用户单位)
15	最大减速度	LREAL	R/W	读写最大减速度(用户单位)
16	最大加加速度	LREAL	R	读取最大加加速度 (用户单位)
17	最大加加速度	LREAL	R/W	读写最大加加速度 (用户单位)
1000	速度回授	LREAL	R	读取速度回授(RPM)
1001	RPM单位 转换系数	LREAL	R	读取单位转换系数 (用户单位 → RPM)
1002	马达端PUU单位 转换系数	LREAL	R	读取单位转换系数 (用户单位→ 马达端PUU)
1003	用户单位 转换系数	LREAL	R	读取单位转换系数 (马达端PUU → 用户单位)
1004	扭矩限制开关	BOOL	R/W	读写扭矩限制开关
1005	AMF时间常数	LREAL	R/W	读写AMF时间常数(ms)

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Valid	BOOL	True / False (False)	功能块完成时为True
Busy	BOOL	True / False (False)	功能块作用时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 详细说明请参考第9章
Value	LREAL	负数、正数或0 (0)	读取的参数数据

6

6.5.2.16 DMC_SetServoGain

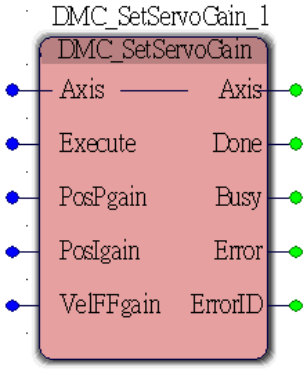
■ 类型

Function Block

■ 功能描述

根据使用者输入的增益值设置前馈控制器，并将其输出作为速度前馈输入至伺服。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
Axis	AXIS_REF	-	指定运动轴编号

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	当Execute为上升缘时，改写前馈控制器的增益值
PosPgain	LREAL	正数、负数或0.0 (0.0)	前馈控制器位置环比例增益值
PosIain	LREAL	正数、负数或0.0 (0.0)	前馈控制器位置环积分增益值
VelFFgain	LREAL	正数、负数或0.0 (0.0)	前馈控制器速度前馈增益值

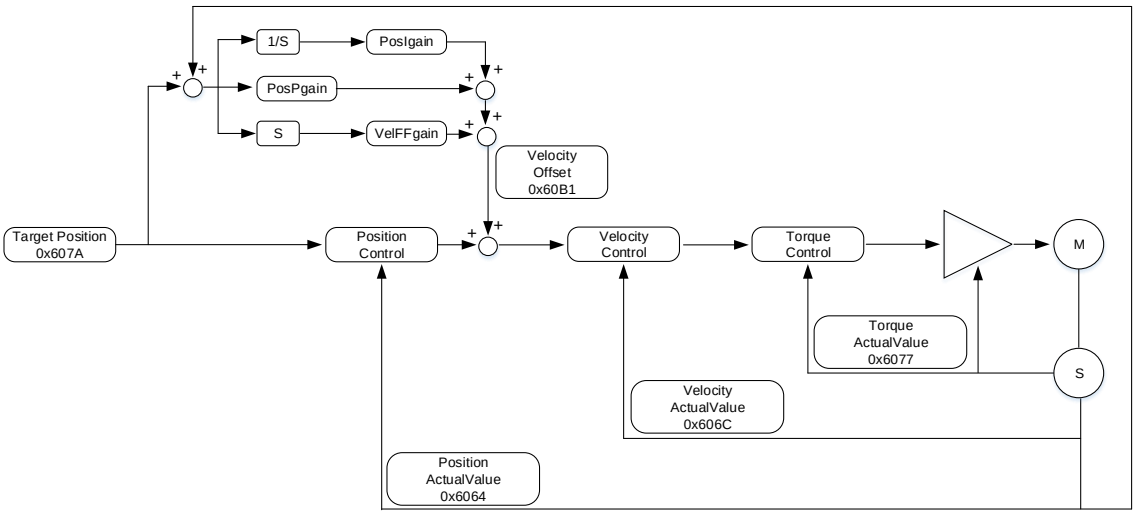
■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	增益写入完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	错误发生时记录的错误码 详细说明请参考第9章

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	增益写入完成时	■ Execute转为False时 ■ 若Execute为False而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在 ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

■ 控制方块图



6

6.5.2.17 DMC_TouchProbe

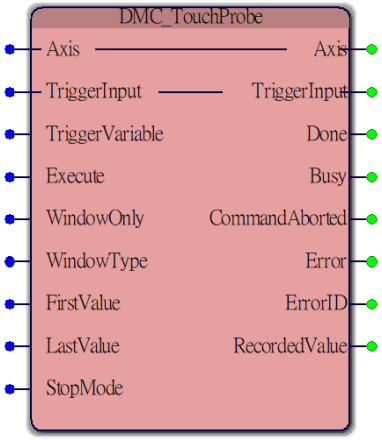
■ 类型

Function Block

■ 功能描述

依据用户之设定，在触发事件中记录所需要的数据。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
Axis	AXIS_REF	-	指定触发事件轴编号
TriggerInput	MC_TRIGGER_REF	详细说明请参考6.5.9.3节	设定触发事件，详细设定请参考7.1节 Capture功能说明

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Trigger Variable	BOOL	True / False (False)	PLC型触发事件之处发信号
Execute	BOOL	True / False (False)	Execute上升沿时，执行此指令
WindowOnly	BOOL	True / False (False)	若为True，开启Window功能
WindowType	USINT	0 ~ 6 (0)	设定Window功能开启时，Window范围参考来源 0：RECORDED_DATA 1：POS_CMD 2：POS_FB 3：VEL_CMD 4：VEL_FB 5：ACCDEC_CMD 6：ACCDEC_FB
FirstValue	LREAL	正数、负数或0.0 (0.0)	Window起始值
LastValue	LREAL	正数、负数或0.0 (0.0)	Window末端值

脚位名称	数据型态	设定值 (默认值)	功能
StopMode	BOOL	None	保留

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	触发事件完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Command Aborted	BOOL	True / False (False)	指令被中断时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 详细说明请参考第9章
Recorded Value	LREAL	负数、正数或0 (0)	触发事件发生时所记录的数据

■ 功能描述

详细使用说明请参照 7.1 节 Capture 功能说明

6

6.5.2.18 MC_AbortTrigger

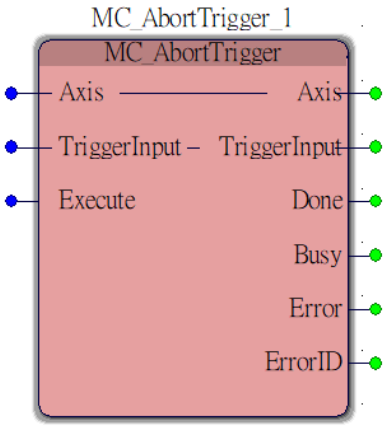
■ 类型

Function Block

■ 功能描述

中断指定触发事件，例如 DMC_TouchProbe。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
Axis	AXIS_REF	-	指定触发事件轴编号
TriggerInput	MC_TRIGGER_REF	详细说明请参考6.5.9.3节	设定触发事件

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute上升沿时，执行此指令

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	触发事件中断完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 详细说明请参考第9章

6.5.2.19 MC_WriteParameter

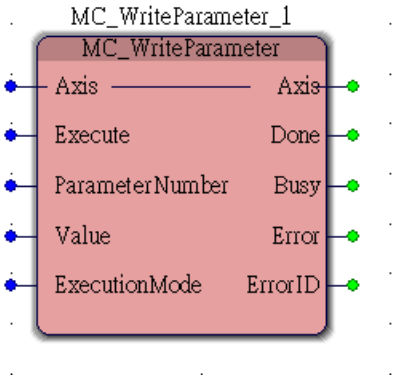
■ 类型

Function Block

■ 功能描述

写入指定轴的信息。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
Axis	AXIS_REF	-	指定轴编号

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute上升沿时，执行此指令
Parameter Number	INT	输入范围请参考下表 (0)	参数编号
Value	LREAL	负数、正数或0 (0)	写入的参数资料
ExecutionMode	INT	0、1 (0)	目前不支持

■ 参数编号(Parameter Number)说明

Parameter Number	名称	数据型态	R/W	详细说明
1	位置命令	LREAL	R	读取位置命令
2	正极限位置	LREAL	R/W	读写正极限位置
3	负极限位置	LREAL	R/W	读写负极限位置
4	正极限开关	BOOL	R/W	读写正极限开关
5	负极限开关	BOOL	R/W	读写负极限开关
6	位置误差监控开关	BOOL	R/W	读写位置误差监控开关
7	最大位置误差	LREAL	R/W	读写最大位置误差
8	最大速度	LREAL	R	读取最大速度(用户单位)
9	最大速度	LREAL	R/W	读写最大速度(用户单位)

6

Parameter Number	名称	数据型态	R/W	详细说明
10	速度回授	LREAL	R	读取速度回授(用户单位)
11	速度命令	LREAL	R	读取速度命令(用户单位)
12	最大加速度	LREAL	R	读取最大加速度(用户单位)
13	最大加速度	LREAL	R/W	读写最大加速度(用户单位)
14	最大减速度	LREAL	R	读取最大减速度(用户单位)
15	最大减速度	LREAL	R/W	读写最大减速度(用户单位)
16	最大加加速度	LREAL	R	读取最大加加速度 (用户单位)
17	最大加加速度	LREAL	R/W	读写最大加加速度 (用户单位)
1000	速度回授	LREAL	R	读取速度回授(RPM)
1001	RPM 单位转换系数	LREAL	R	读取单位转换系数 (用户单位 → RPM)
1002	马达端PUU 单位转换系数	LREAL	R	读取单位转换系数 (用户单位→ 马达端PUU)
1003	用户单位 单位转换系数	LREAL	R	读取单位转换系数 (马达端PUU → 用户单位)
1004	扭矩限制开关	BOOL	R/W	读写扭矩限制开关
1005	AMF时间常数	LREAL	R/W	读写AMF时间常数 (ms)

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Valid	BOOL	True / False (False)	功能块完成时为True
Busy	BOOL	True / False (False)	功能块作用时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 详细说明请参考第9章

6.5.2.20 MC_TorqueControl

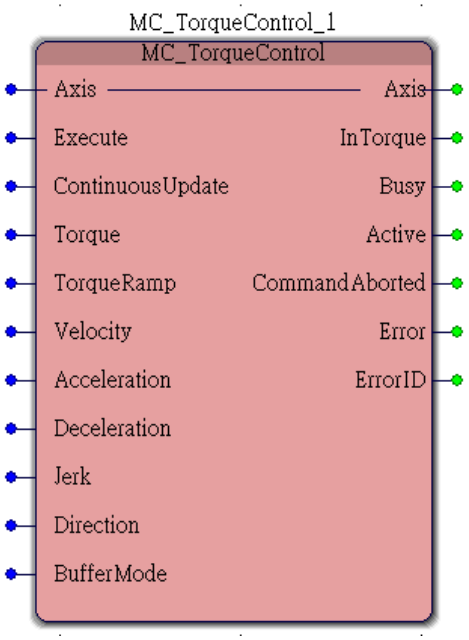
■ 类型

Function Block

■ 功能描述

单轴扭力控制。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
Axis	AXIS_REF	-	指定轴编号

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute上升沿时，执行此指令
Continuous Update	BOOL	None	保留脚位
Torque	LREAL	负数、正数或0 (0)	设定扭力值 [%]
TorqueRamp	LREAL	正数 (0.0)	扭力的设定值的最大时间导数 [%/s]
Velocity	LREAL	正数或0.0 (0.0)	最大速度数值 (用户单位/秒)
Acceleration	LREAL	正数 (0.0)	加速度数值 (用户单位/秒^2)
Deceleration	LREAL	正数 (0.0)	减速度数值 (用户单位/秒^2)
Jerk	LREAL	正数或0.0 (0.0)	加加速度数值 (用户单位/秒^3)

6

脚位名称	数据型态	设定值 (默认值)	功能
Direction	MC_ DIRECTION	0, 2 (0)	指定轴伺服马达运转方向 0：正向 2：负向
BufferMode	INT	0, 1 (1)	0：中断 1：等待

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
InTorque	BOOL	True / False (False)	扭力的设定值等于指令值 功能块完成时为True
Busy	BOOL	True / False (False)	功能块作用时为True
Active	BOOL	True / False (False)	功能块动作执行中
Command Aborted	BOOL	True / False (False)	功能块被其他命令中断
Error	BOOL	True / False (False)	当指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录错的误码 详细说明请参考第9章

6.5.3 多轴功能块（Motion MultiAxis）

6.5.3.1 MC_GearIn

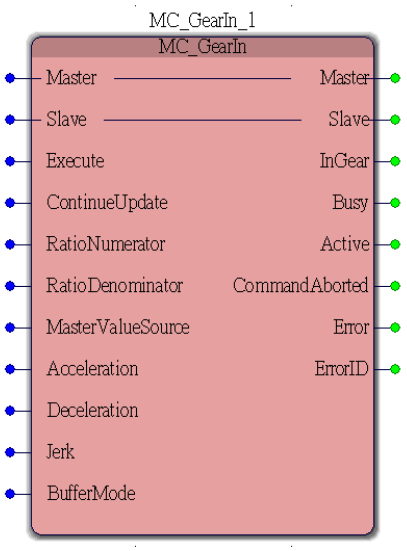
■ 类型

Function Block

■ 功能描述

依照输入的齿轮比，建立主从轴的速度比例关系。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值（默认值）	功能
Master	AXIS_REF	-	指定主轴编号
Slave	AXIS_REF	-	指定从轴编号

■ 输入脚位

脚位名称	数据型态	设定值（默认值）	功能
Execute	BOOL	True / False (False)	当Execute为上升缘时，此功能即开始执行
ContinueUpdate	BOOL	None	保留脚位
RatioNumerator	INT	整数（0）	主从轴齿轮比之分子
Ratio Denominator	UINT	正整数（0）	主从轴齿轮比之分母
MasterValue Source	INT	0 ~ 1 (0)	主轴速度的来源 0：主轴速度命令 1：主轴回授速度
Acceleration	LREAL	正数（0.0）	从轴追赶时的加速度
Deceleration	LREAL	正数（0.0）	从轴追赶时的减速度
Jerk	LREAL	正数或0.0 (0.0)	从轴追赶时的加加速度

6

脚位名称	数据型态	设定值 (默认值)	功能
BufferMode	INT	0 ~ 5 (0)	指定此功能块的缓冲行为*注 0：中断(Aborting) 1：等待(Buffered) 2：以较低的速度作为中继速度(BlendingLow) 3：以前一笔指令的速度作为中继速度(BlendingPrevious) 4：以后一笔指令的速度作为交接(BlendingNext) 5：以较低的高速作为中继速度(BlendingHigh)

注：BufferMode 详细请参考 6.4.3 节。

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
InGear	BOOL	True / False (False)	主从轴速度维持输入比例时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Active	BOOL	True / False (False)	指定轴受控制时为True
Command Aborted	BOOL	True / False (False)	指令被中断时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码详细说明请参考第9章

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
InGear	主从轴速度维持输入比例时	■ 在CommandAborted上升沿时 ■ Execute为False且在Error上升沿时
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
Active	轴运动开始时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
CommandAborted	■ 此功能块指令被其他缓冲模式设为Aborting的指令中断时 ■ 此功能块指令被MC_Stop中断时	■ 在Execute下降沿时 ■ 当Execute为False而CommandAborted转为True，此时CommandAborted 维持一个扫描周期的True状态后，立刻转为False

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Error / ErrorID	指令的执行条件或输入值发生错误时（错误码记录在ErrorID）	在Execute下降沿时(清除ErrorID记录之错误码)

■ 输出脚位的变化时序

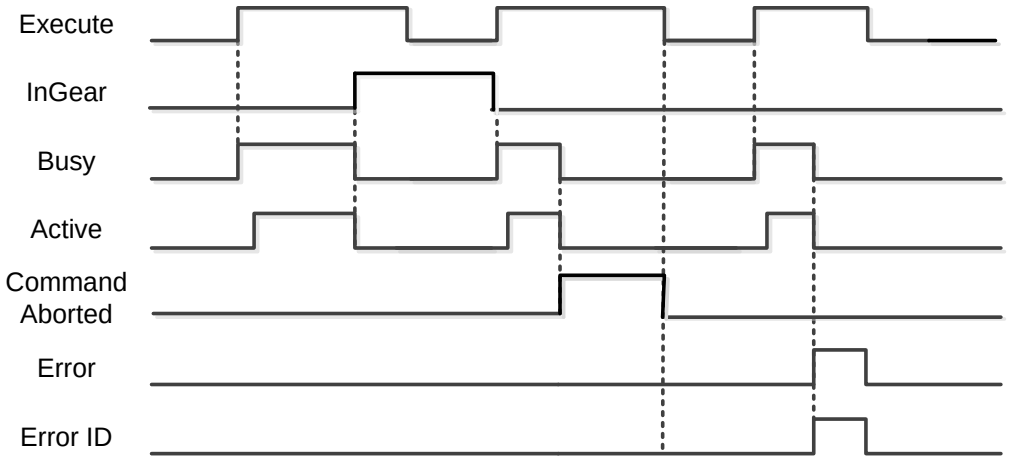
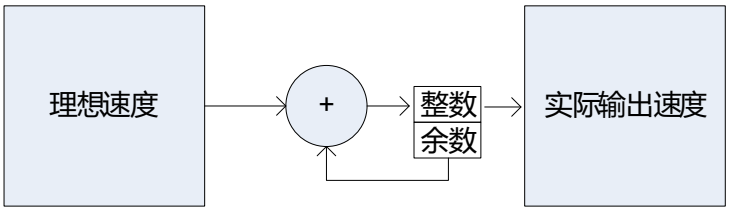


图 6.5.3.1.1 MC_GearIn 脚位时序图 (时序详细说明请参考 6.4.3 节)

■ 功能描述

余数保留：由于理想速度可能不是整数，因此系统会累积剩余的余数，到达整数时，补偿在实际输出速度。



速度比例：当输出脚位 InGear 变为 True 时，从轴的速度会与主轴成使用者输入的比例，如果主轴速度改变，从轴也会呈现固定比例的改变。

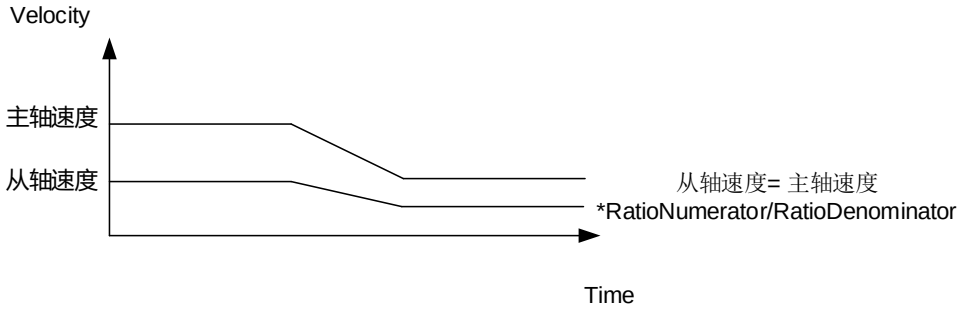


图 6.5.3.1.2 MC_GearIn 速度时间图

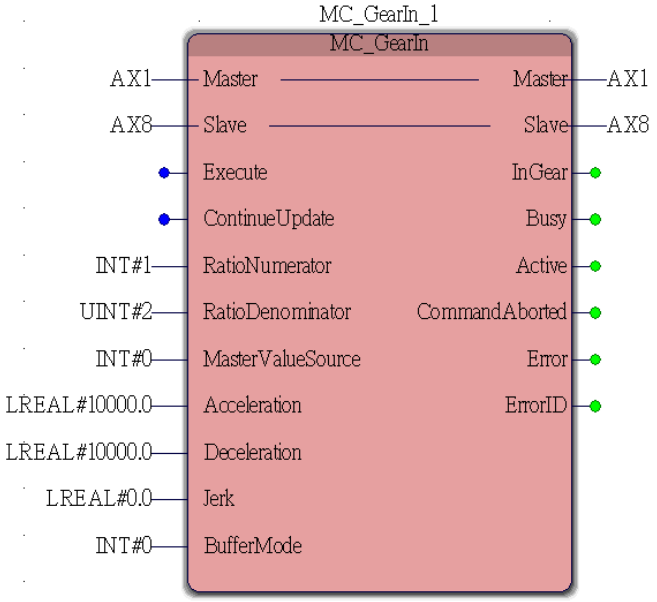
6

■ 参考范例

此范例依下列条件来设计：

1. 主从轴比例为 1 : 2。
2. 加速度与减速度为 10000.0 PUU/S^2。
3. 主轴参考速度为主轴速度命令。

实际动作结果为主轴与从轴的速度维持在 1 : 2 的关系。



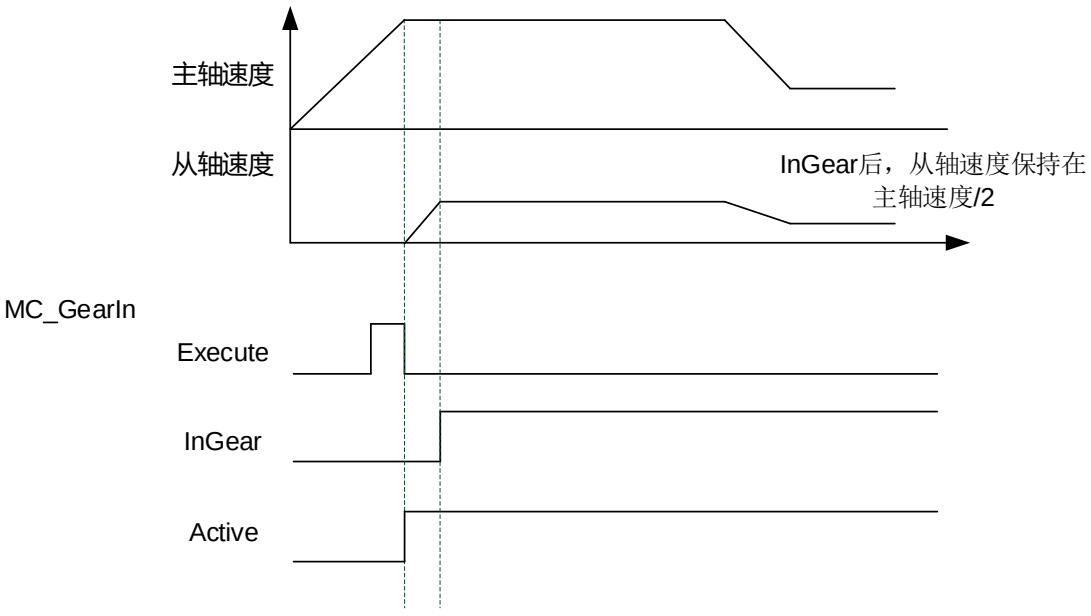


图 6.5.3.1.3 MC_GearIn 范例说明

6.5.3.2 MC_GearOut

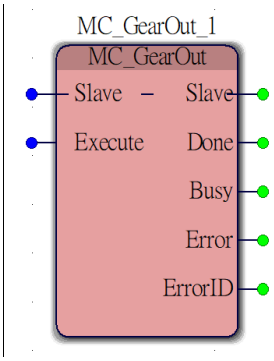
■ 类型

Function Block

■ 功能描述

解除主从轴间的速度比例关系，并维持当下速度。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
Slave	AXIS_REF	-	指定运动轴编号

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	当Execute为上升缘时，此功能即开始执行

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	绝对寻址完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 详细说明请参考第9章

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	主从轴关系解除时	■ Execute转为False时 ■ 若Execute为False而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

6

■ 输出脚位的变化时序

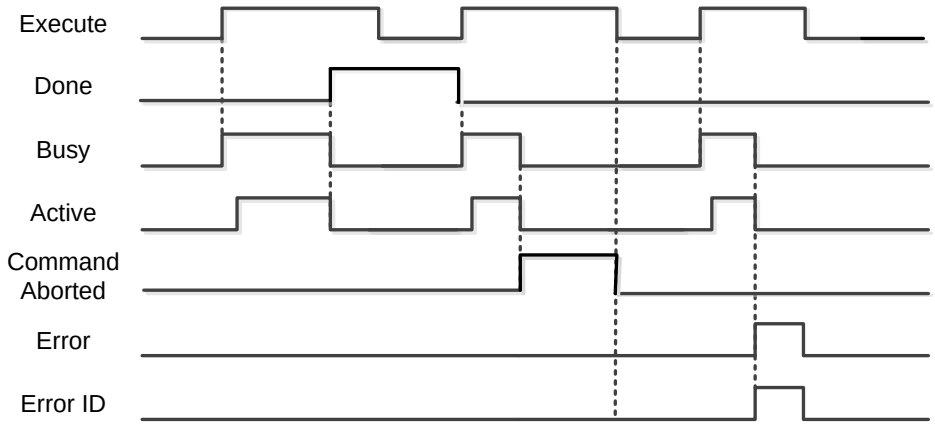


图 6.5.3.2.1 MC_GearOut 脚位时序图 (时序详细说明请参考 6.4.3 节)

■ 参考范例

当从轴在齿轮脱离(GearOut)后，这时若变更主轴的速度，从轴将保持原本的速度。

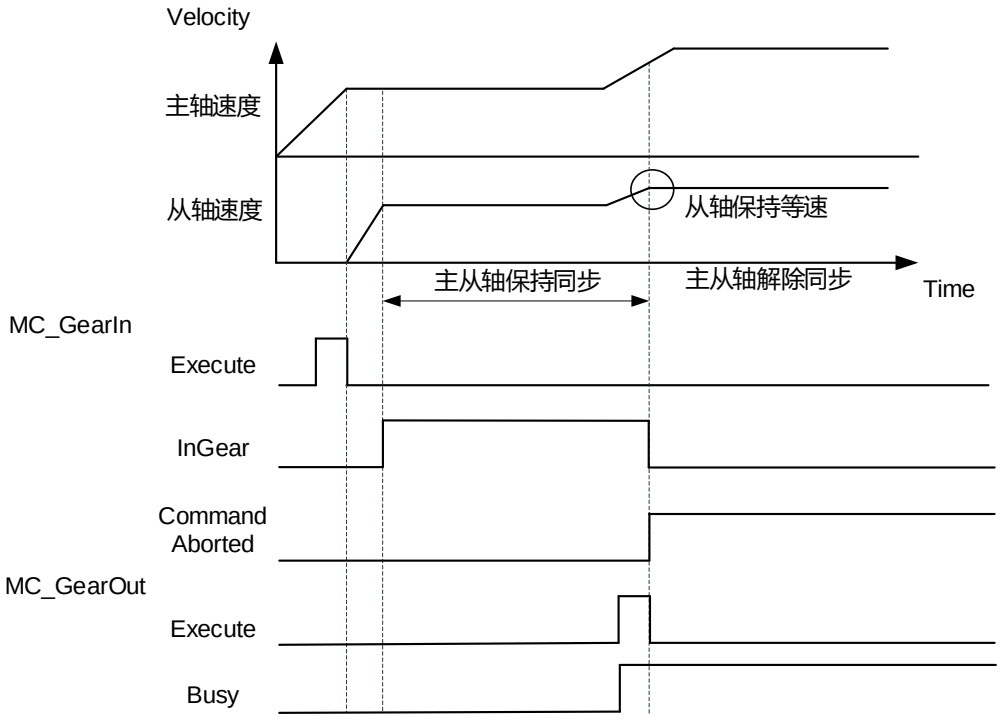


图 6.5.3.2.2 MC_GearOut 范例说明

6.5.3.3 DMC_CamIn

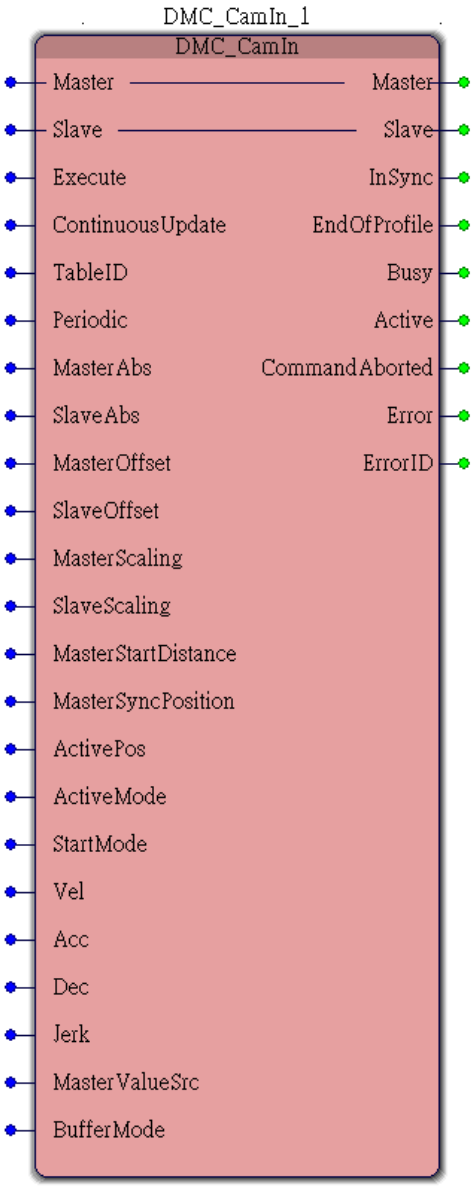
■ 类型

Function Block

■ 功能描述

使主轴与从轴建立凸轮关系，并根据凸轮表的描述运动。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
Master	AXIS_REF	-	指定主轴编号
Slave	AXIS_REF	-	指定从轴编号

6

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	当Execute为上升缘时，此功能即开始执行
Continue Update	BOOL	None	保留脚位
TableID	UINT	0 ~ 65535 (0)	欲使用的凸轮表编号
Periodic	BOOL	True / False (False)	是否重复执行凸轮表
MasterAbs	BOOL	True / False (False)	设定主轴模式 True：绝对 False：相对
SlaveAbs	BOOL	True / False (False)	设定从轴模式 True：绝对 False：相对
MasterOffset	LREAL	正数、负数或0.0 (0.0)	主轴的偏移量
SlaveOffset	LREAL	正数、负数或0.0 (0.0)	从轴的偏移量
MasterScaling	LREAL	正数 (1.0)	主轴相位的比例缩放
SlaveScaling	LREAL	正数 (1.0)	从轴位移的比例缩放
MasterStrat Distance	LREAL	None	保留脚位
MasterSync Distance	LREAL	None	保留脚位
ActivePos	LREAL	正数、负数或0.0 (0.0)	设定凸轮啮合启动时主轴位置
ActiveMode	UINT	0 ~ 4 (0)	设定执行跟随凸轮表时机模式 0：立即执行从轴跟随凸轮表运动。 1：主轴绝对位置。 当主轴到达ActivePos设定的位置时，从轴开始跟随凸轮表运动。 2：主轴绝对相位。 将ActivePos对应到主轴凸轮表格中的指定位置，当主轴运行到指定位置时，从轴开始根据凸轮表运动。 3：当主轴运动到凸轮表下个周期后，从轴开始跟随凸轮表运动。 4：从轴完全跟随凸轮表运动。

脚位名称	数据型态	设定值 (默认值)	功能
StartMode	UINT	0 ~ 3 (0)	选择啮合时机时从轴动作 0：正向跳变 1：最短距离 2：正向 3：反向
Vel	LREAL	正数 (0.0)	啮合时从轴追上主轴的速度
Acc	LREAL	正数 (0.0)	啮合时从轴追上主轴的加速度
Dec	LREAL	正数 (0.0)	啮合时从轴追上主轴的减速度
Jerk	LREAL	正数或0.0 (0.0)	啮合时从轴追上主轴的加加速度
MasterValue Source	UINT	0 ~ 1 (0)	主轴速度的来源 0：主轴速度命令 1：主轴回授速度
BufferMode	UINT	0 ~ 5 (0)	指定此功能块的缓冲行为*注 0：中断(Aborting) 1：等待(Buffered) 2：以较低的速度作为中继速度(BlendingLow) 3：以前一笔指令的速度作为中继速度(BlendingPrevious) 4：以后一笔指令的速度作为交接(BlendingNext) 5：以较低的高速作为中继速度(BlendingHigh)

注：BufferMode 详细请参考 6.4.3 节。

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
InSync	BOOL	True / False (False)	主从轴速度同步时为True
EndOfProfile	BOOL	True / False (False)	凸轮周期是否完成
Busy	BOOL	True / False (False)	指令被触发执行时为True
Active	BOOL	True / False (False)	指定轴受控制时为True
Command Aborted	BOOL	True / False (False)	指令被中断时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 详细说明请参考第9章

6

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
InGear	主从轴速度同步时	■ 在CommandAborted上升沿时 ■ 若Execute为False而Error上升沿时
EndOfProfile	完成周期运动时	-
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
Active	轴运动开始时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
CommandAborted	■ 此功能块指令被其他缓冲模式设为Aborting的指令中断时 ■ 此功能块指令被MC_Stop中断时	■ 在Execute下降沿时 ■ 当Execute为False而CommandAborted转为True，此时CommandAborted 维持一个扫描周期的True状态后，立刻转为False
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

■ 输出脚位的变化时序

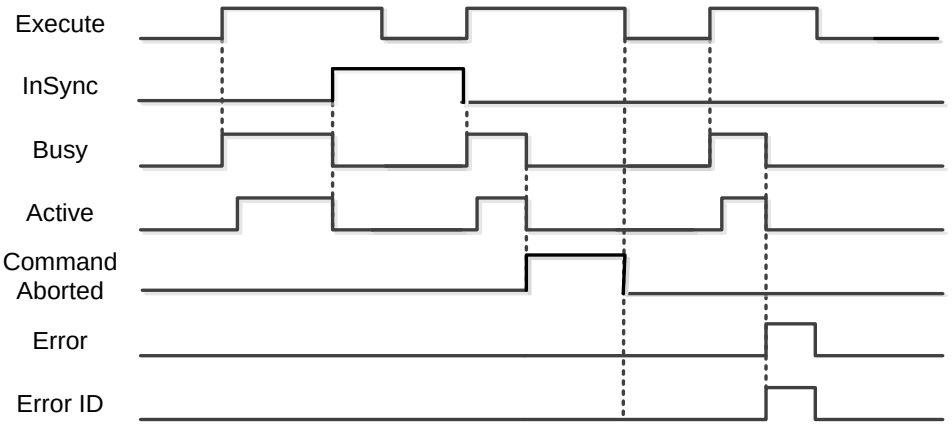
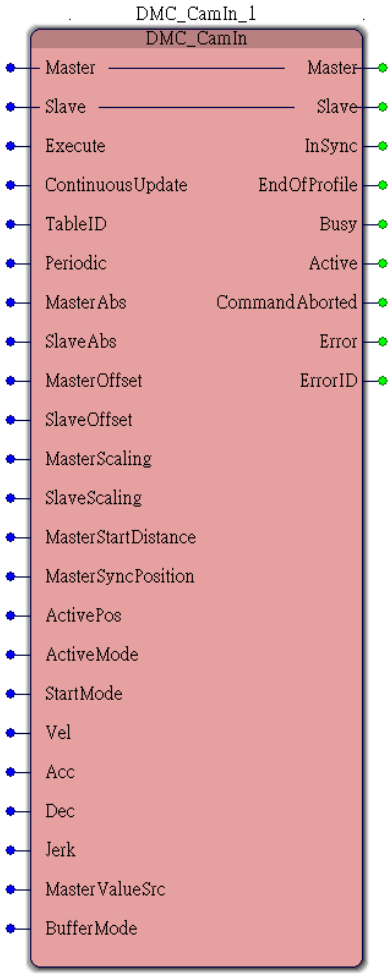


图 6.5.3.3.1 DMC_CamIn 脚位时序图 (时序详细说明请参考 6.4.3 节)

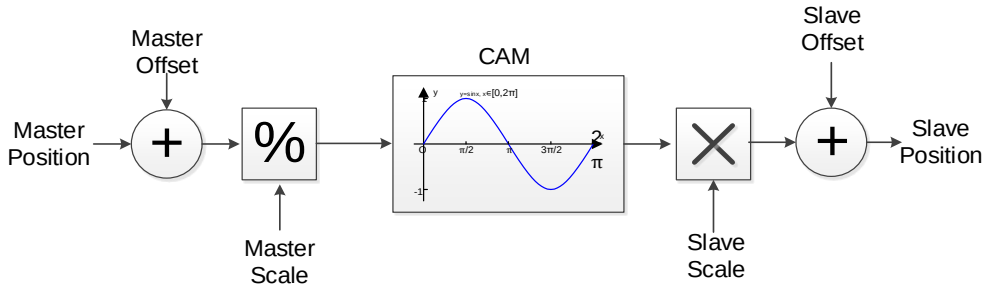
■ 功能描述

凸轮表输入：

输入完主轴与从轴信息后，请直接输入凸轮表编号(TableID)当初设定的名称，如果控制器中凸轮表不存在，则会发生错误。



Offset 与 Scaling 说明：



从轴位置 = f[(主轴位置+ 主轴偏移量)/主轴比例] × 从轴比例 + 从轴偏移量

6

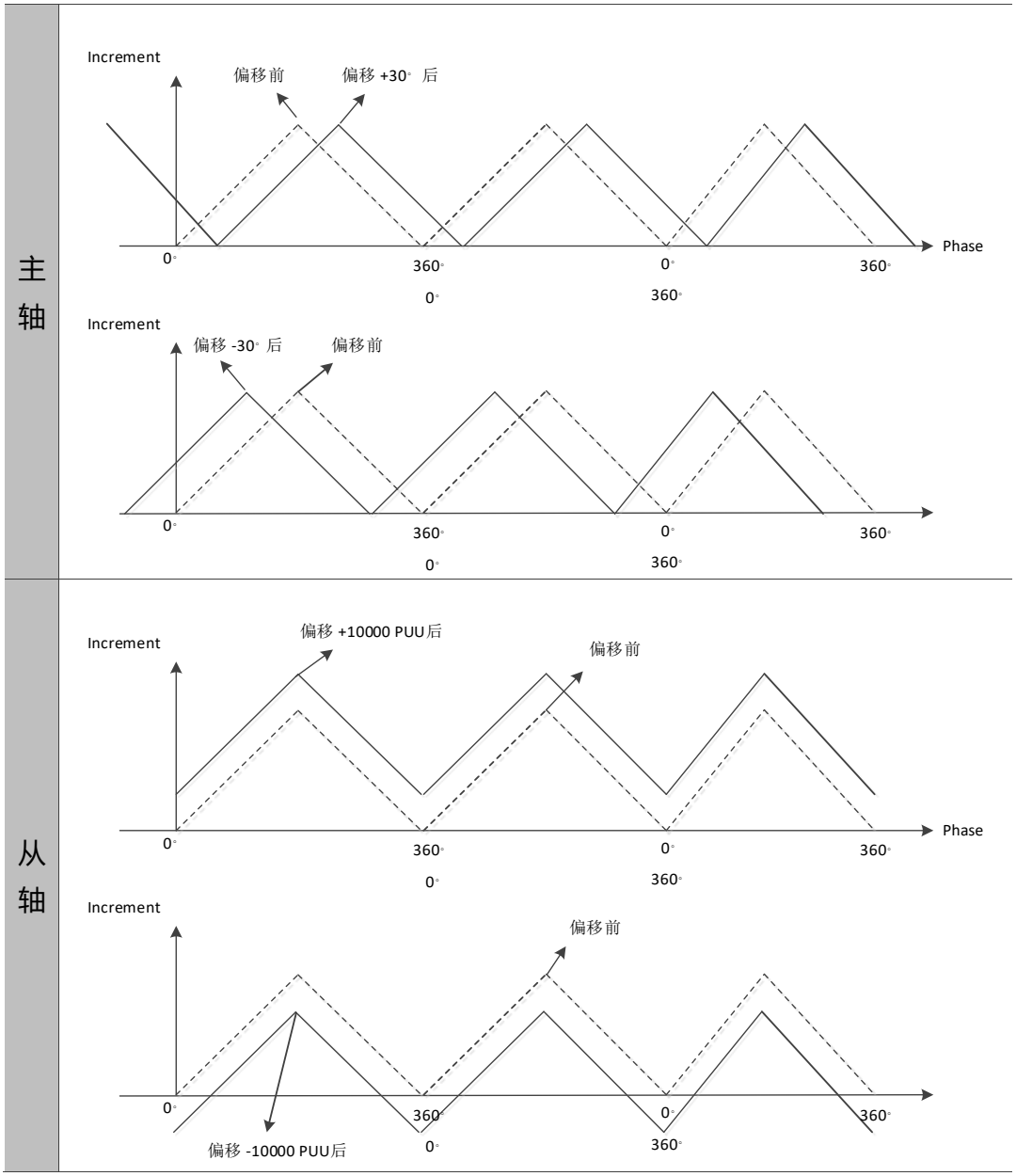
主轴比例范例说明：

假设一个凸轮主轴走一圈 300 mm，从轴走一圈 200 mm，主轴比例 = 2。

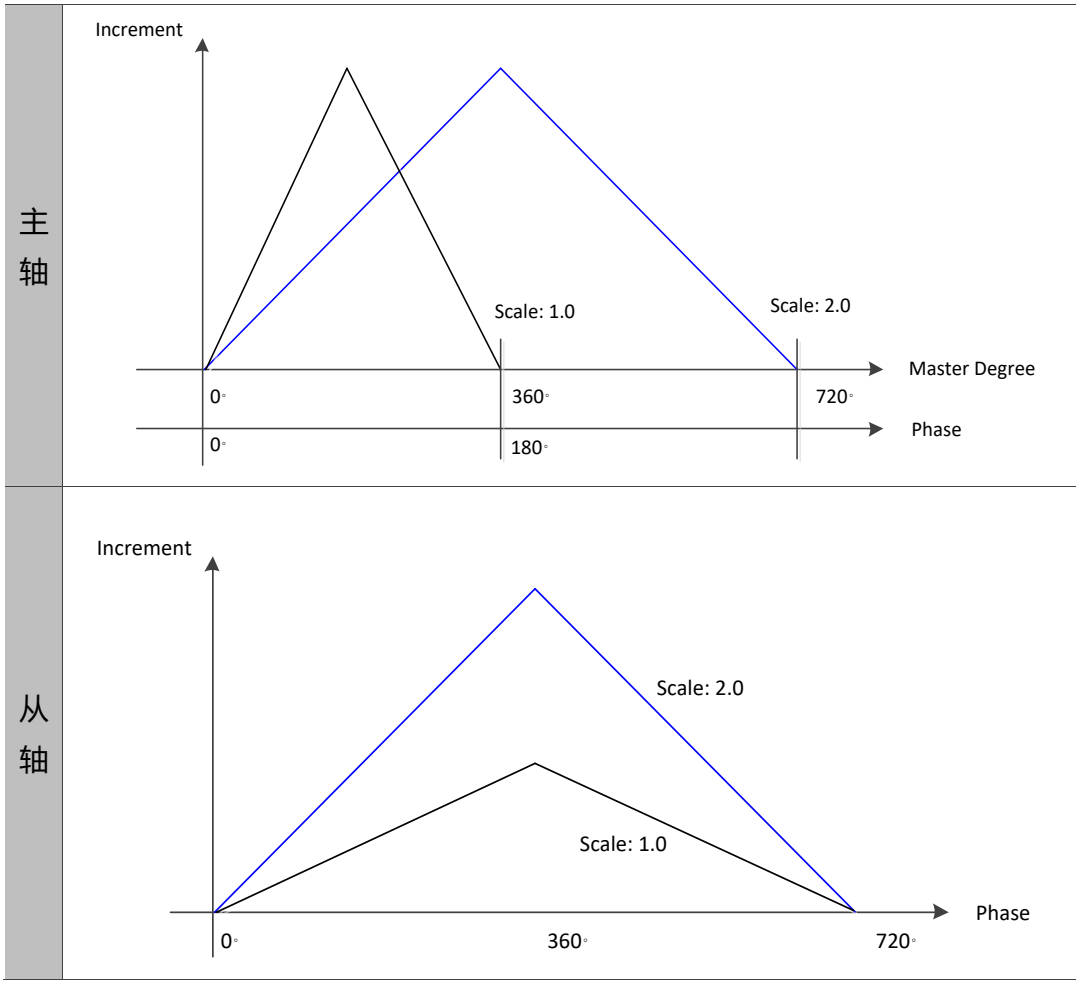
当主轴走到 300mm，从轴位置 = $f\left(\frac{300}{2}\right) = f(150)$

当主轴走到 600mm，从轴位置 = $f\left(\frac{600}{2}\right) = f(300)$ = 200 (凸轮转一圈)

Offset 的结果：



Scaling 的结果：



主从轴绝对模式与相对模式说明

- 假设主轴模数：凸轮表格中主轴一圈的距离。如下面表格中的 400
  从轴模数：凸轮表格中从轴点位和 0 的距离最大值。如下面表格中的 380

主轴位置	0	20	70	150	250	400
从轴位置	50	70	380	250	200	260

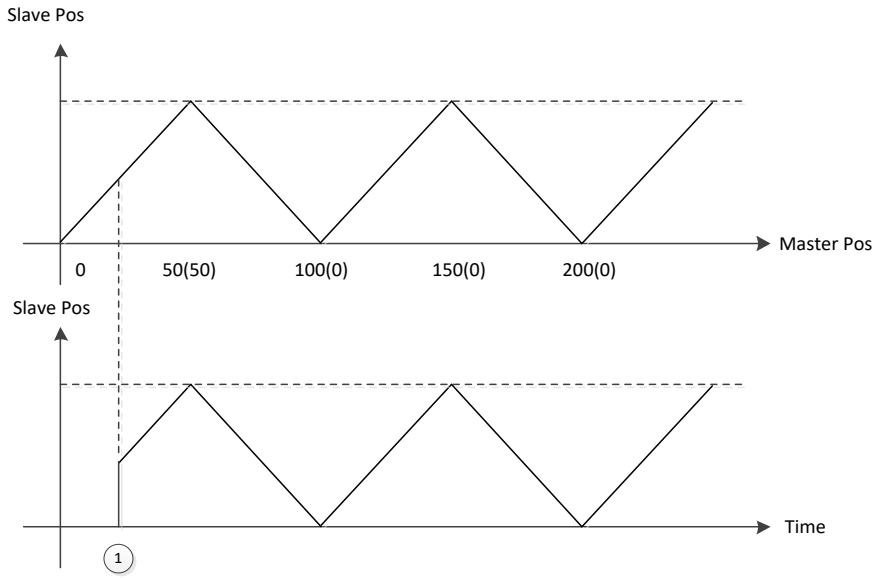
- 如果主轴为相对模式，则凸轮启动时，无论主轴当时在何位置，主轴当时位置即为凸轮曲线的起始位置(0)。
- 如果主轴为绝对模式，则凸轮启动时，主轴的位置为当时位置除以主轴模数后的余数。
  例：主轴位置 = 650，则凸轮启动时主轴的位置 = 650 mod 400 = 250
- 如果从轴为相对模式，则凸轮启动时，从轴位置为当时主轴位置相对应的凸轮位置。
  例：主轴位置 = 70，则凸轮启动时从的位置 = 380
- 如果从轴为绝对模式，则凸轮启动时从轴位置为当时位置除以从轴模数后的余数。
  例：从轴位置 = 700，则凸轮启动时从轴的位置 = 700 mod 380 = 320

6

ActiveMode 说明

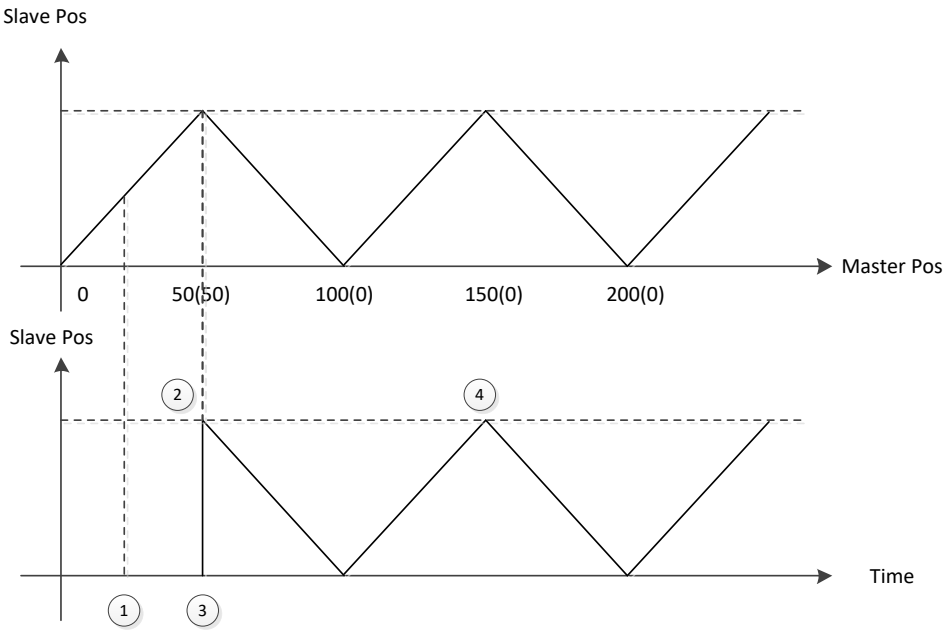
透过 ActiveMode 与 ActivePos 设定从轴开始啮合的时机，ActiveMode 共有 5 种模式。

1. ActiveMode = 0 (立即执行)，ActivePos (无效)



① ：DMC_CamIn Execute 脚位上升沿触发，从轴立即开始啮合 (InSync 脚位为 TRUE)

2. ActiveMode = 1 (主轴绝对位置)，ActivePos = 50



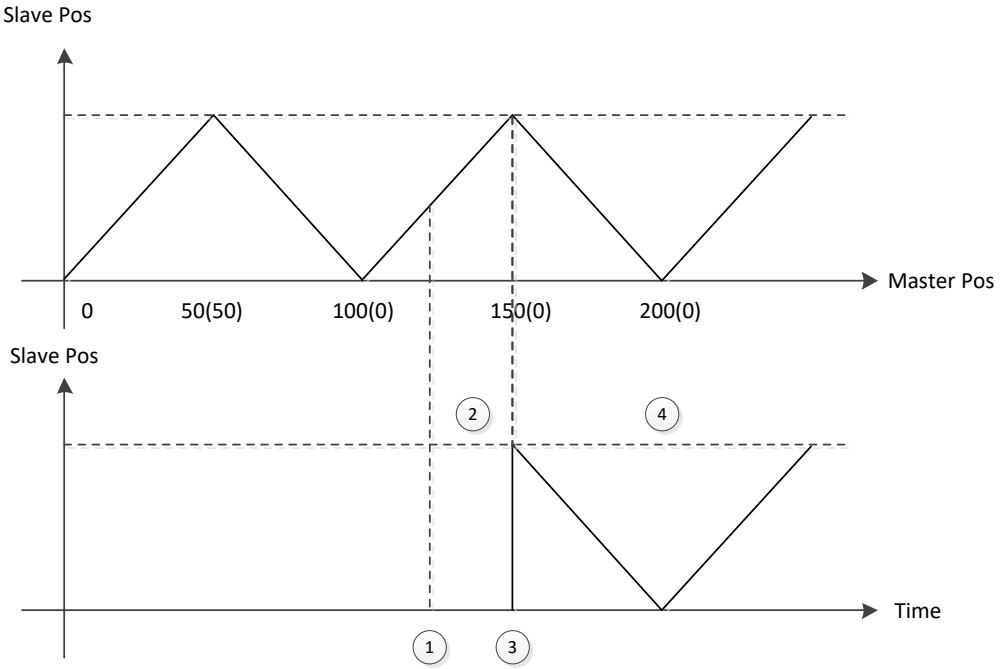
①：DMC_CamIn Execute 脚位上升沿触发，此时主轴位置为 25 (25)

②：从轴静止等待啮合

③：当主轴位置到达 50 (ActivePos = 50) 时，从轴开始啮合

④：主从轴完成啮合 (InSync 脚位为 TRUE)

3. ActiveMode = 2 (主轴绝对相位)，ActivePos = 50



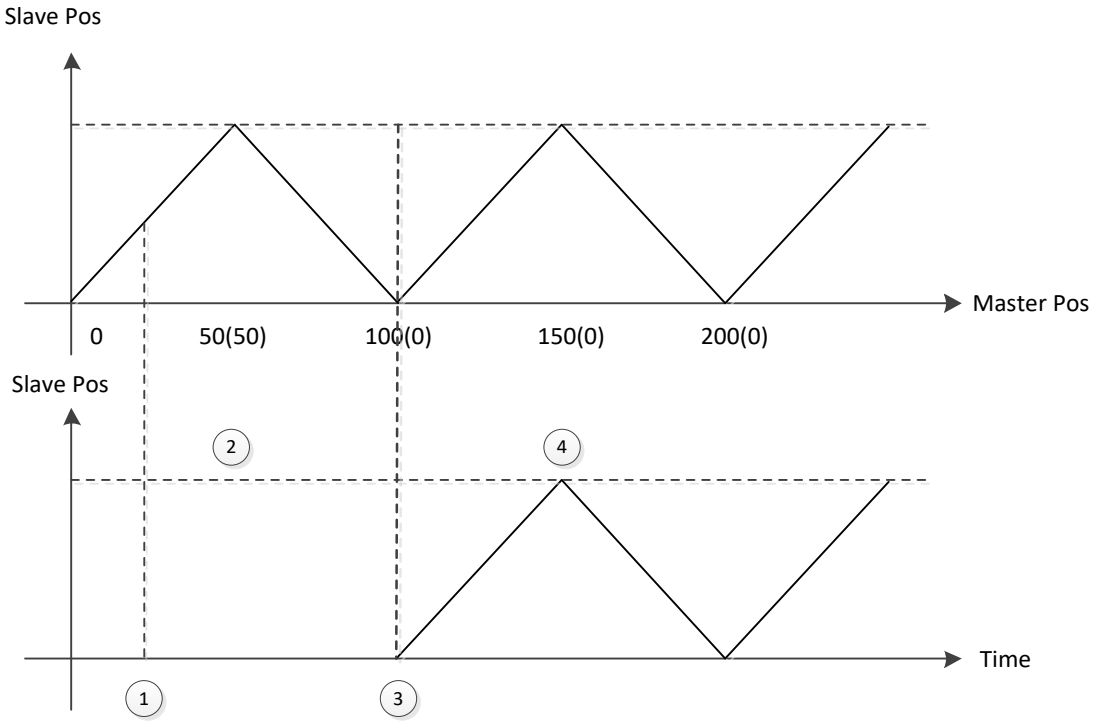
①：DMC_CamIn Execute 脚位上升沿触发，此时主轴位置为 150 (25)

②：从轴静止等待啮合

③：当主轴相位到达 50(ActivePos = 50)时，从轴开始啮合

④：主从轴完成啮合(InSync 脚位为 TRUE)

4. ActiveMode = 3 (下一周期开始啮合)，ActivePos (无效)



①：DMC_CamIn Execute 脚位上升沿触发，此时主轴位置为 25(25)

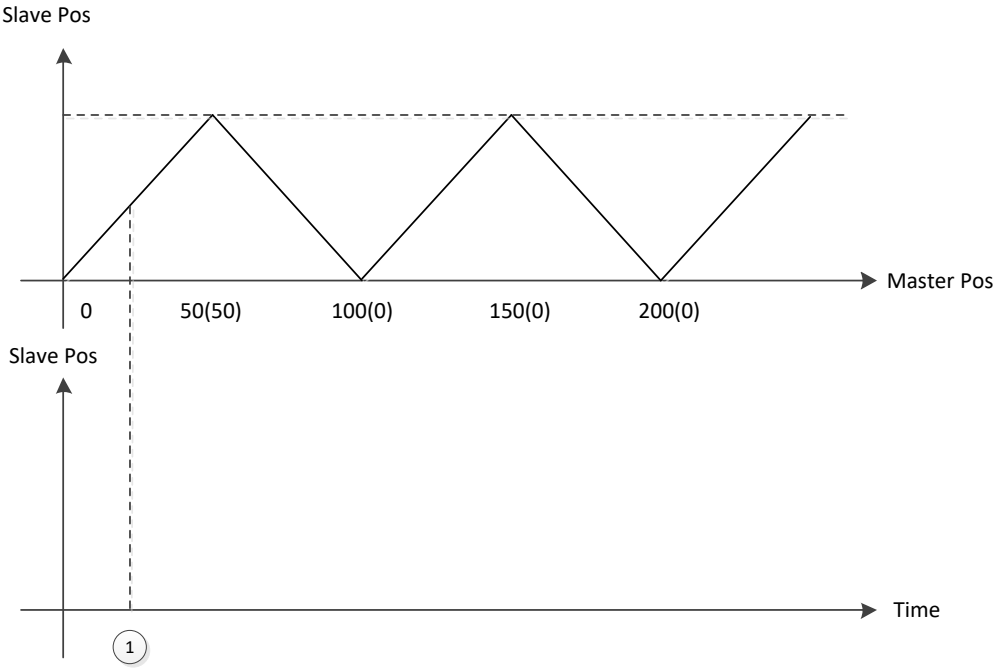
②：从轴静止等待啮合

③：当主轴相位到达下一凸轮周期时，从轴开始啮合

④：主从轴完成啮合 (InSync 脚位为 TRUE)

6

5. ActiveMode = 4 (完全不运行)，ActivePos (无效)



① ：DMC_CamIn Execute 脚位上升沿触发，但从轴完全不啮合

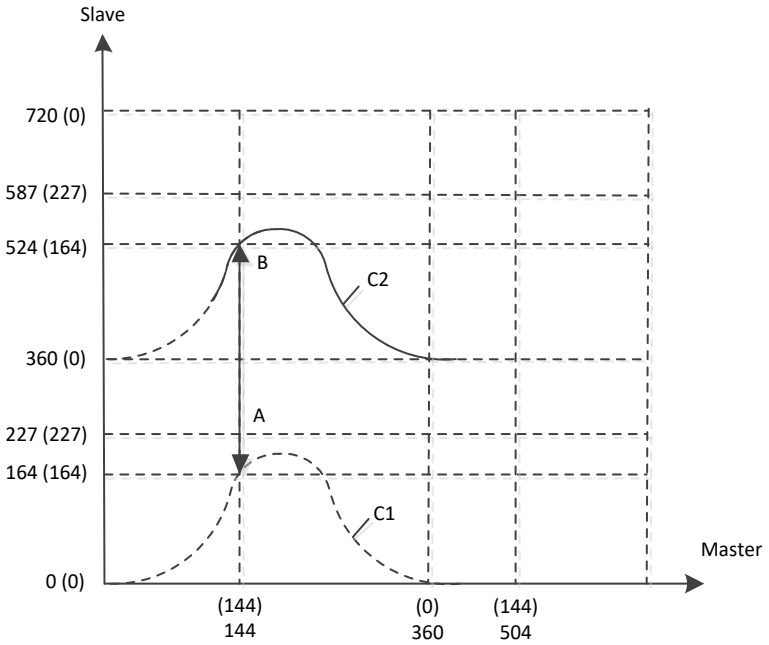
StartMode 说明

Start Mode 会根据主轴、从轴的绝对(相对)模式不同，而有不同的行为，总共有 4 种组合。

1. 主轴绝对 & 从轴绝对

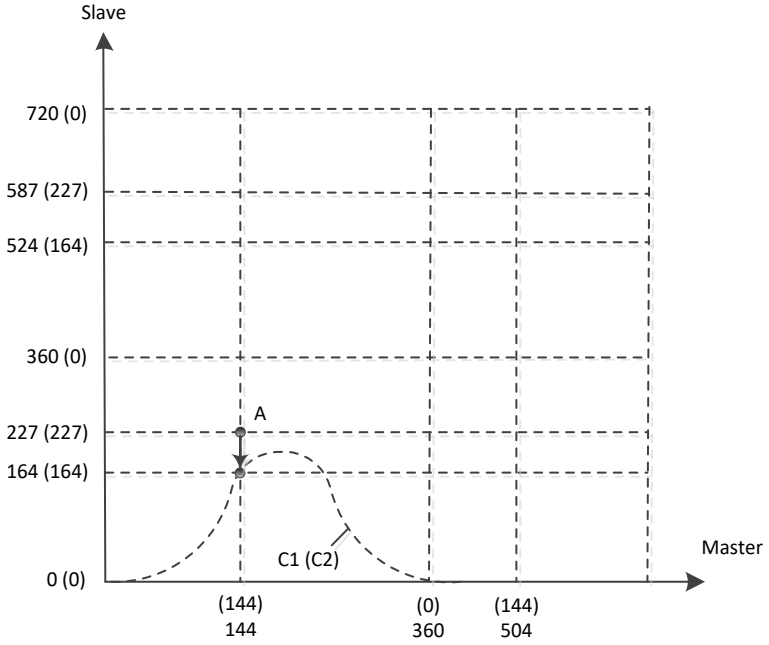
- 启动模式为 0：正向跳变

  在一个同步周期内，从轴从 A 点正向移动到 B 点与主轴啮合。A 点为啮合前的主从轴位置，B 点为啮合点，主轴为绝对模式，凸轮曲线中的主轴位置 = 144。由凸轮曲线可知，当主轴位置为 144 时，凸轮曲线中的从轴位置为 164。从轴为绝对模式，因启动模式 0 为正向跳变启动，故从轴需在一个同步周期内从当前位置正转运行到其下个周期的 164，即实际位置 524，故啮合点 B 的坐标为(144,524)。当主轴运转时，从轴跟随主轴按照 C2 曲线从 B 点开始运转。在一个周期中从轴补偿的距离为 297 (524 – 227 = 297)。

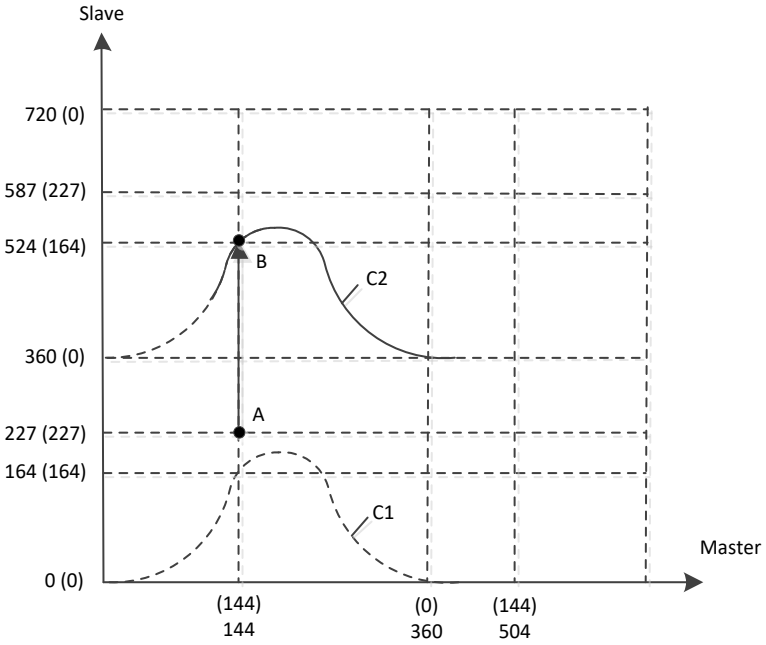


- 启动模式为 1：最短距离 (从轴根据当前位置和目标位置的距离关系判断正向啮合或者反向啮合)。

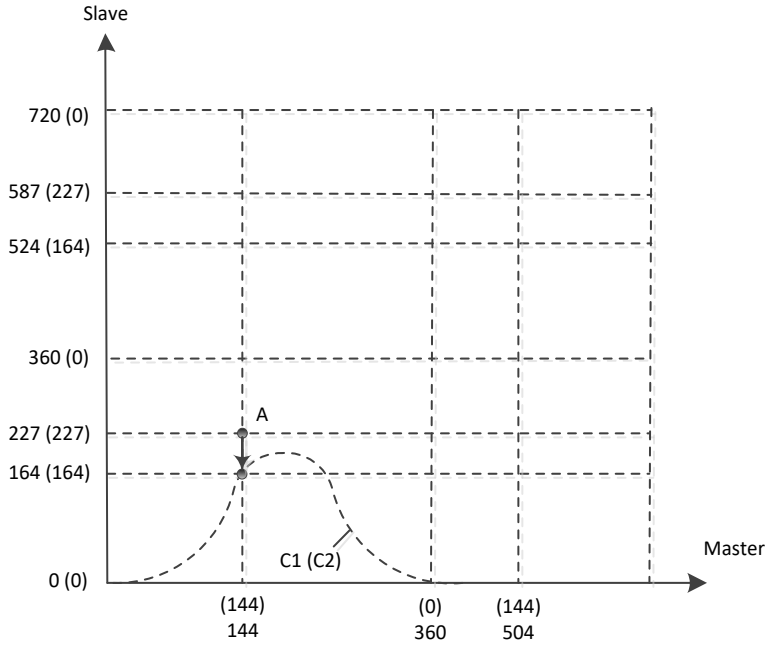
  从轴以轴参数中的最大速度、最大加速度、最大减速度从 A 点正向移动到 B 点与主轴啮合。从轴当前位置和本周期 164 的距离最近，故从轴需从当前位置反转运行到本周期的 164，即实际位置 164，故啮合点 B 的坐标为(144,164)。当主轴运转时，从轴跟随主轴按照 C2 曲线从 B 点开始运转。从轴迭加的距离为-63 (164 - 227 = - 63)。



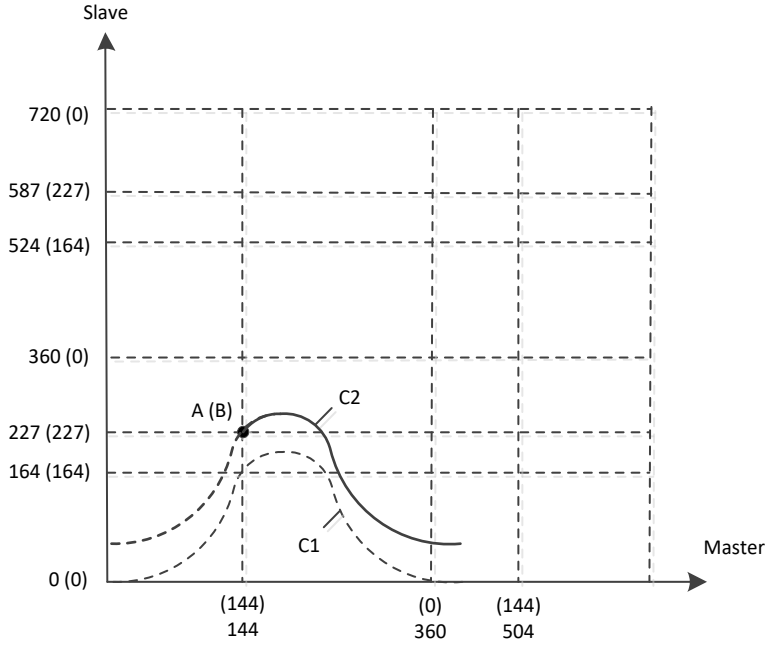
- 启动模式为 2：正向启动。
  从轴以轴参数中的最大速度、最大加速度、最大减速度从 A 点正向移动到 B 点与主轴啮合。因从轴当前位置(227)已超过本周期的位置(164)，启动模式为 2，所以从轴会正向迭加一段距离啮合。从轴迭加的距离为 297 (524 - 227 = 297)。



- 启动模式为 3：凸轮反方向启动。
  从轴以轴参数中的最大速度、最大加速度、最大减速度从 A 点正向移动到 B 点与主轴啮合。因从轴当前位置(227)已超过本周期的位置(164)，启动模式为 3，所以从轴会反向迭加一段距离啮合。从轴迭加的距离为-63 (164 – 227 = - 63)。



2. 主轴绝对 & 从轴相对
   从轴为相对模式时，在任何一种启动模式下，从轴在 B 点的实际物理位置为 227，在凸轮曲线中的位置为 164，所以啮合点 B 的坐标为(144,227)。当主轴运转时，从轴跟随主轴按照 C2 曲线从 B 点开始运转，不做任何迭加。

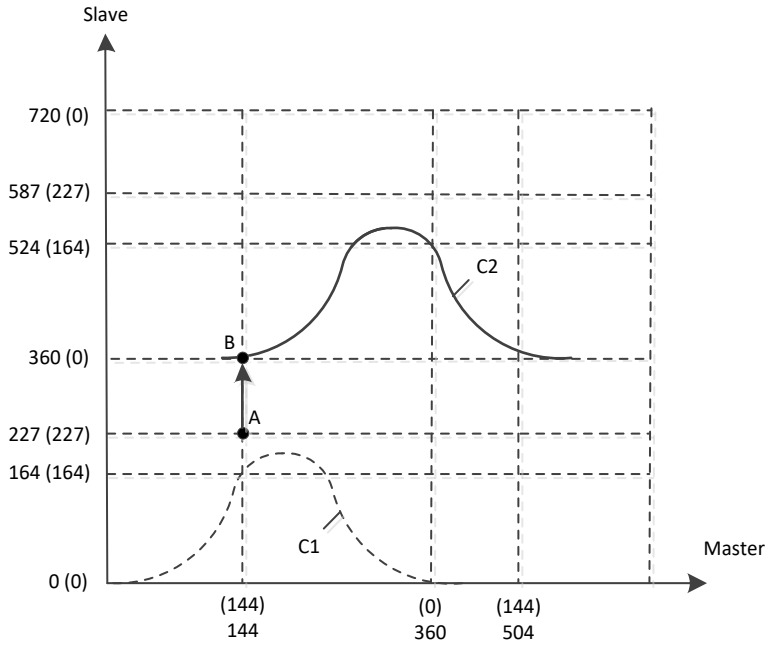


6

3. 主轴相对 & 从轴绝对

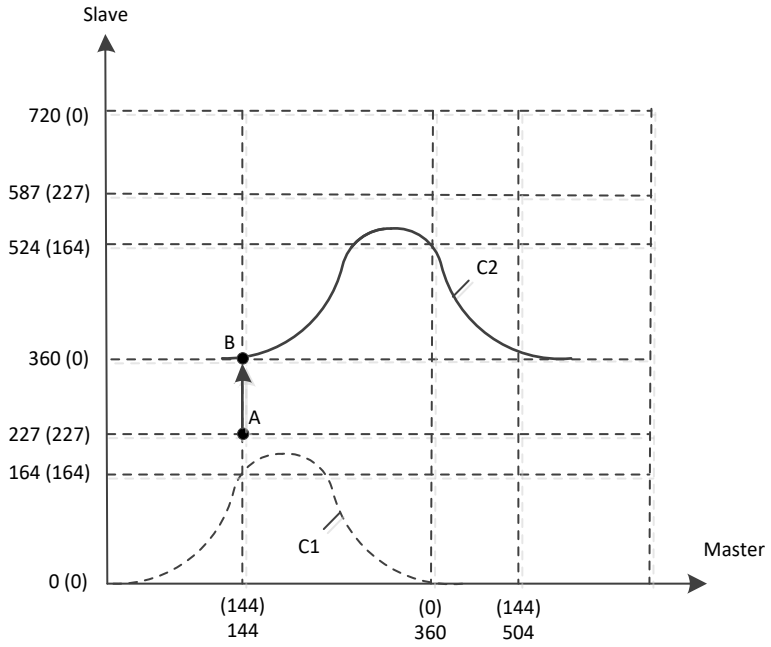
- 启动模式为 0：正向跳变

  从轴需在一个同步周期内从当前位置正转运行到其下个周期的 0，即实际位置 360，故啮合点 B 的坐标为（144,360)。当主轴运转时，从轴跟随主轴按照 C2 曲线从 B 点开始运转。在一个周期中从轴补偿的距离为 133 (360 – 227 = 133)。



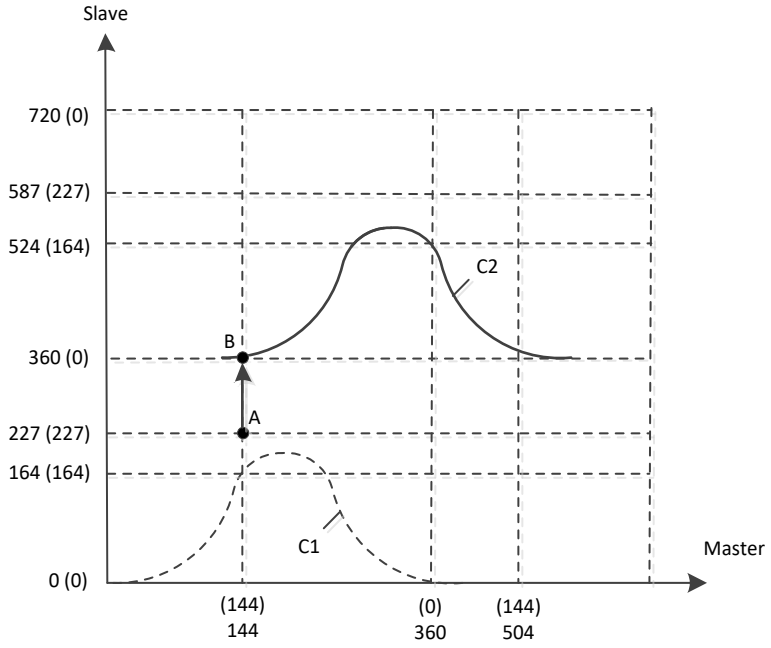
- 启动模式为 1：正向跳变

  从轴当前位置和下一周期 0 的距离最近，故从轴需从当前位置运行到下一周期的 0，即实际位置 360，故啮合点 B 的坐标为(144,360)。当主轴运转时，从轴跟随主轴，按照 C2 曲线从 B 点开始运转。在一个周期中从轴补偿的距离为 133 (360 – 227 = 133)。



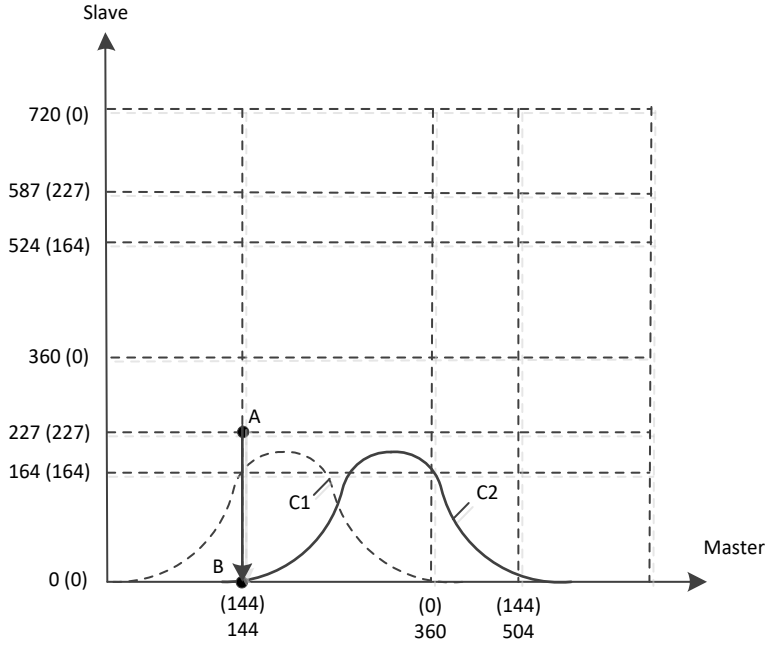
- 启动模式为 2：正向。

  从轴以轴参数中的最大速度、最大加速度、最大减速度从 A 点正向移动到 B 点与主轴啮合。因从轴当前位置(227)已超过本周期的位置(0)，所以从轴会正向迭加一段距离啮合。从轴迭加的距离为 133 (360 - 227 = 133)。



- 启动模式为 3：凸轮反方向。

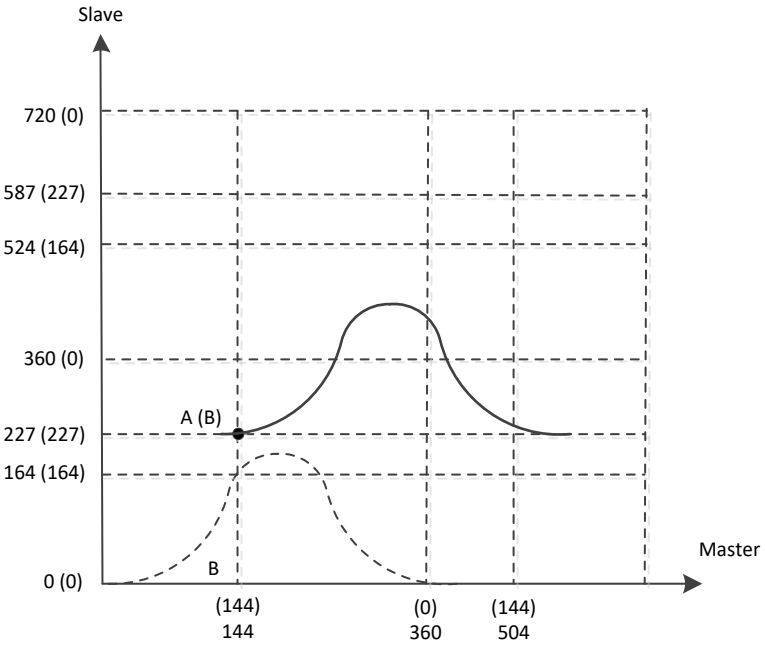
  从轴以轴参数中的最大速度、最大加速度、最大减速度从 A 点正向移动到 B 点与主轴啮合。因从轴当前位置(227)已超过本周期的位置(0)，所以从轴会反向迭加一段距离啮合。



6

4. 主轴绝对 &从轴相对

- 从轴为相对模式时，各种启动模式下，从轴在 B 点的实际物理位置为 227，在凸轮曲线中的位置为 0，所以啮合点 B 的坐标为(144,227)。当主轴运转时，从轴跟随主轴按照 C2 曲线从 B 点开始运转，不做任何迭加。



■ 参考范例

此范例依照下列条件来设计：

1. 使用 MC_MoveVel 使主轴开始运动。
2. 当主轴到达等速时，从轴开始执行 DMC_CamIn。
3. 执行两周期后将从轴关闭(Servo off)，使其触发 error。

实际动作结果为 InSync 时，主轴与从轴会根据凸轮表运动。直到 Error 发生后，从轴将不再随凸轮表改变位置。

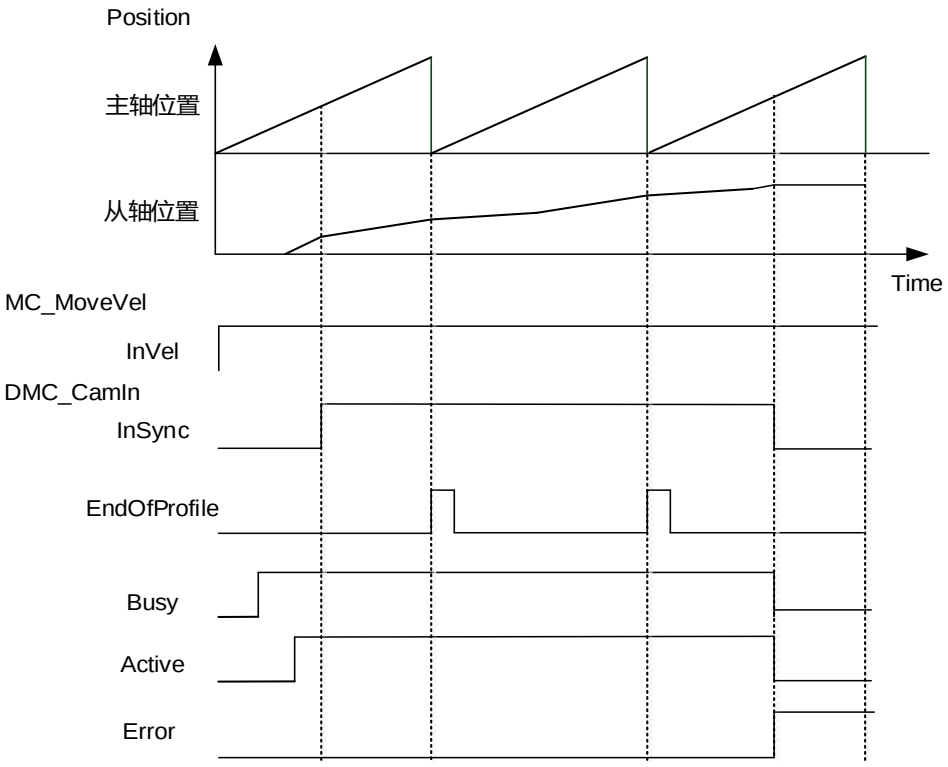


图 6.5.3.3.2 DMC_CamIn 范例说明

6.5.3.4 MC_CamOut

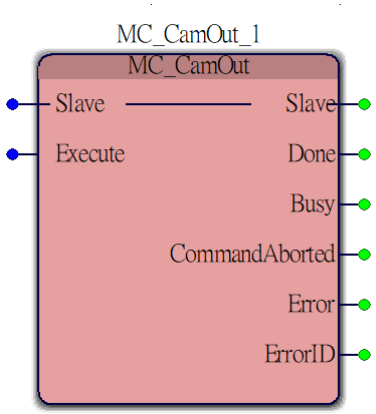
■ 类型

Function Block

■ 功能描述

解除已建立的电子凸轮关系，解除后从轴维持当前速度运动。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
Slave	AXIS_REF	-	指定运动轴编号

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	当Execute为上升缘时，此功能即开始执行

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	绝对寻址完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	错误发生时记录的错误码 详细说明请参考第9章

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	主从轴关系解除时	■ Execute转为False时 ■ 若Execute为False而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为False

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

■ 输出脚位的变化时序

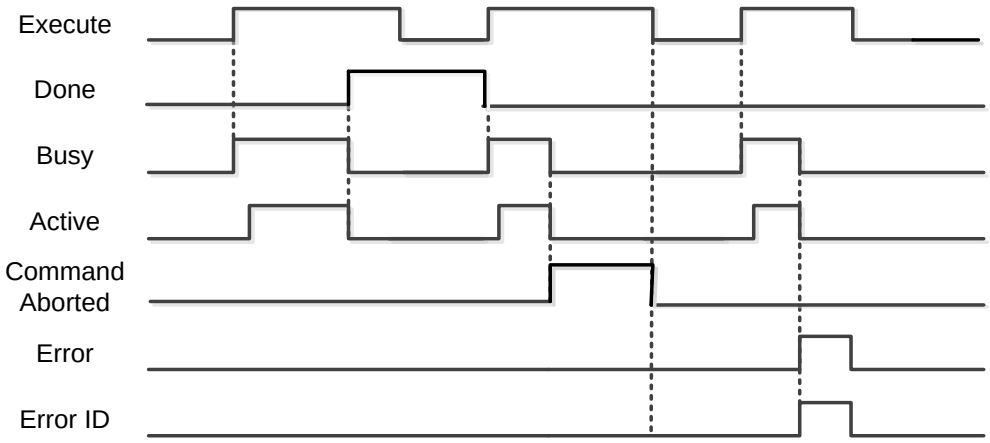


图 6.5.3.4.1 MC_CamOut 脚位时序图 (时序详细说明请参考 6.4.3 节)

■ 参考范例

当从轴执行 CamOut 功能后，这时若变更主轴的速度，从轴速度将不会跟着改变。

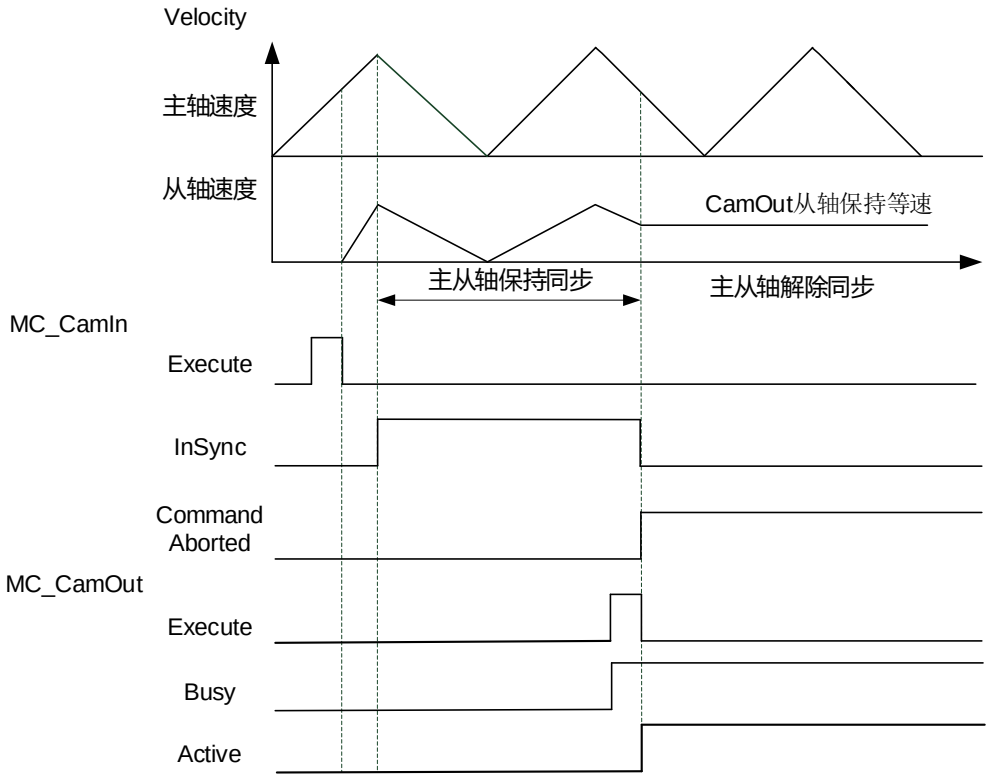


图 6.5.3.4.2 MC_CamOut 脚位时序图 (时序详细说明请参考 6.4.3 节)

6

6.5.3.5 MC_PhasingAbsolute

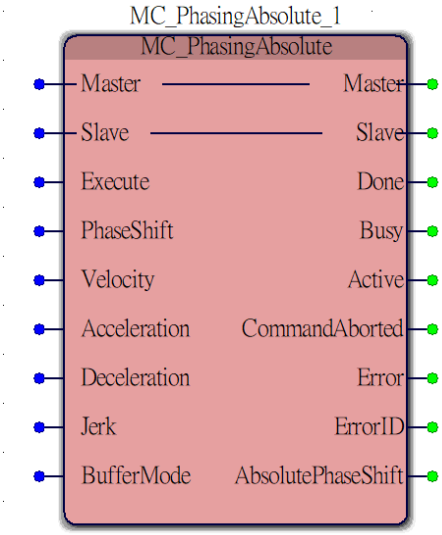
■ 类型

Function Block

■ 功能描述

同步运动中，以绝对值设定调整主轴跟从轴间的相位关系。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
Master	AXIS_REF	-	指定主轴编号
Slave	AXIS_REF	-	指定从轴编号

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	当Execute为上升缘时，此功能即开始执行
PhaseShift	LREAL	正数、负数或0.0 (0.0)	主轴与从轴的相位差距
Velocity	LREAL	正数 (0.0)	最大速度数值 (用户单位/秒)
Acceleration	LREAL	正数 (0.0)	加速度数值 (用户单位/秒^2)
Deceleration	LREAL	正数 (0.0)	减速度数值 (用户单位/秒^2)
Jerk	LREAL	正数或0.0 (0.0)	加加速度数值 (用户单位/秒^3)

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	绝对寻址完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Command Aborted	BOOL	True / False (False)	指令被中断时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 详细说明请参考第9章
Absolute PhaseShift	LREAL	正数、负数或0.0 (0.0)	主从轴当前绝对相位差值

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	绝对寻址完成时	■ Execute转为False时 ■ 若Execute为False而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
CommandAborted	■ 此功能块指令被其他缓冲模式设为Aborting的指令中断时 ■ 此功能块指令被MC_Stop中断时	■ 在Execute下降沿时 ■ 当Execute为False而CommandAborted转为True，此时CommandAborted维持一个扫描周期的True状态后，立刻转为False
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

6

■ 输出脚位的变化时序

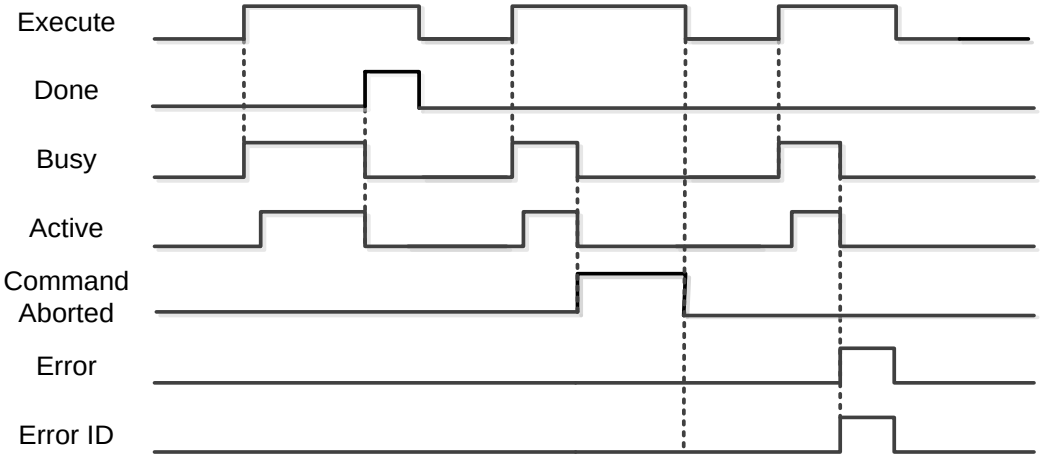


图 6.5.3.5.1 MC_ PhasingAbsolute 脚位时序图 (时序详细说明请参考 6.4.3 节)

■ 参考范例

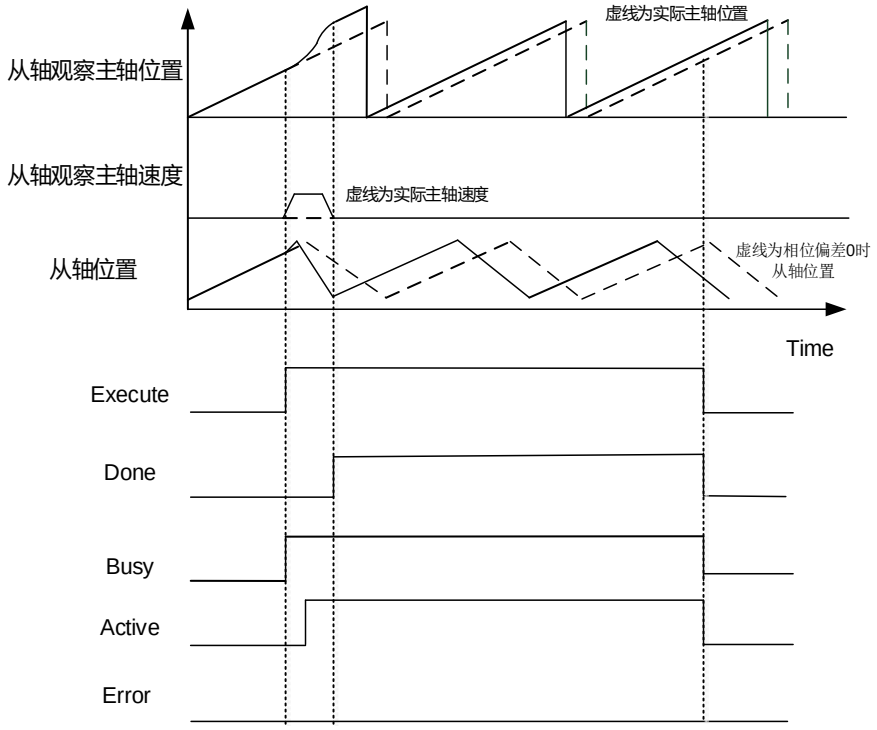


图 6.5.3.5.2 MC_ PhasingAbsolute 范例说明

6.5.3.6 MC_PhasingRelative

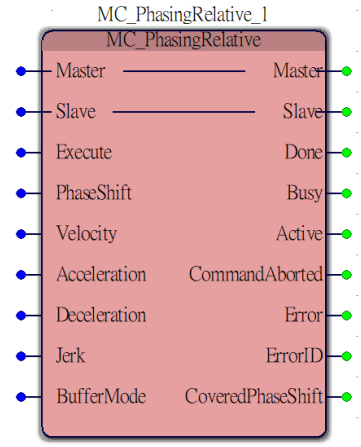
■ 类型

Function Block

■ 功能描述

同步运动中，以相对值设定调整主轴跟从轴间的相位关系。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
Master	AXIS_REF	-	指定主轴编号
Slave	AXIS_REF	-	指定从轴编号

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	当Execute为上升缘时，此功能即开始执行
PhaseShift	LREAL	正数、负数或0.0 (0.0)	主轴与从轴的相位差距
Velocity	LREAL	正数 (0.0)	最大速度数值 (用户单位/秒)
Acceleration	LREAL	正数 (0.0)	加速度数值 (用户单位/秒^2)
Deceleration	LREAL	正数 (0.0)	减速度数值 (用户单位/秒^2)
Jerk	LREAL	正数或0.0 (0.0)	加加速度数值 (用户单位/秒^3)

6

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	当绝对寻址完成时为True
Busy	BOOL	True / False (False)	当指令被触发执行时为True
Command Aborted	BOOL	True / False (False)	当指令被中断时为True
Error	BOOL	True / False (False)	当指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 详细说明请参考第9章
Covered PhaseShift	LREAL	正数、负数或0.0 (0.0)	执行本功能块后主从轴的相位差值

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	绝对寻址完成时	■ Execute转为False时 ■ 若Execute为False而Done 转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
CommandAborted	■ 此功能块指令被其他缓冲模式设为Aborting的指令中断时 ■ 此功能块指令被 MC_Stop 中断时	■ 在Execute下降沿时 ■ 在Execute为False而CommandAborted转为True，此时CommandAborted维持一个扫描周期的True状态后，立刻转为False
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

■ 输出脚位的变化时序

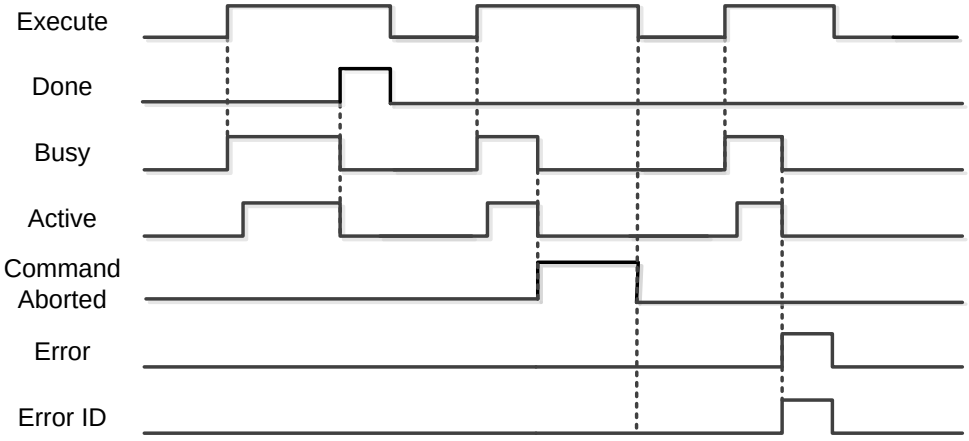


图 6.5.3.5.1 MC_ PhasingRelative 脚位时序图 (时序详细说明请参考 6.4.3 节)

■ 参考范例

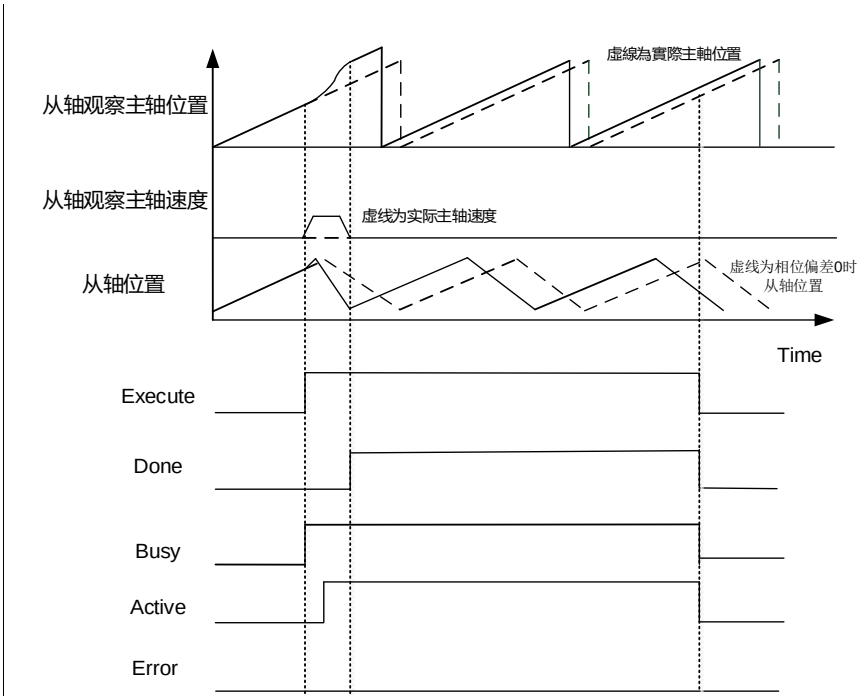


图 6.5.3.5.1 MC_ PhasingRelative 范例说明

6

6.5.3.7 DMC_GantrySynchornize

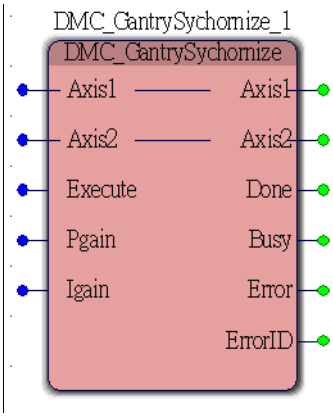
■ 类型

Function Block

■ 功能描述

根据使用者输入的增益值设计 PID 控制器，并将其输出作为速度前馈输入至伺服。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
Axis1	AXIS_REF	-	指定运动轴编号
Axis2	AXIS_REF	-	指定运动轴编号

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	当Execute为上升缘时，改写前馈控制器的增益值
Pgain	LREAL	正数、负数或0.0 (0.0)	PID控制器比例增益值
Igain	LREAL	正数、负数或0.0 (0.0)	PID控制器积分增益值

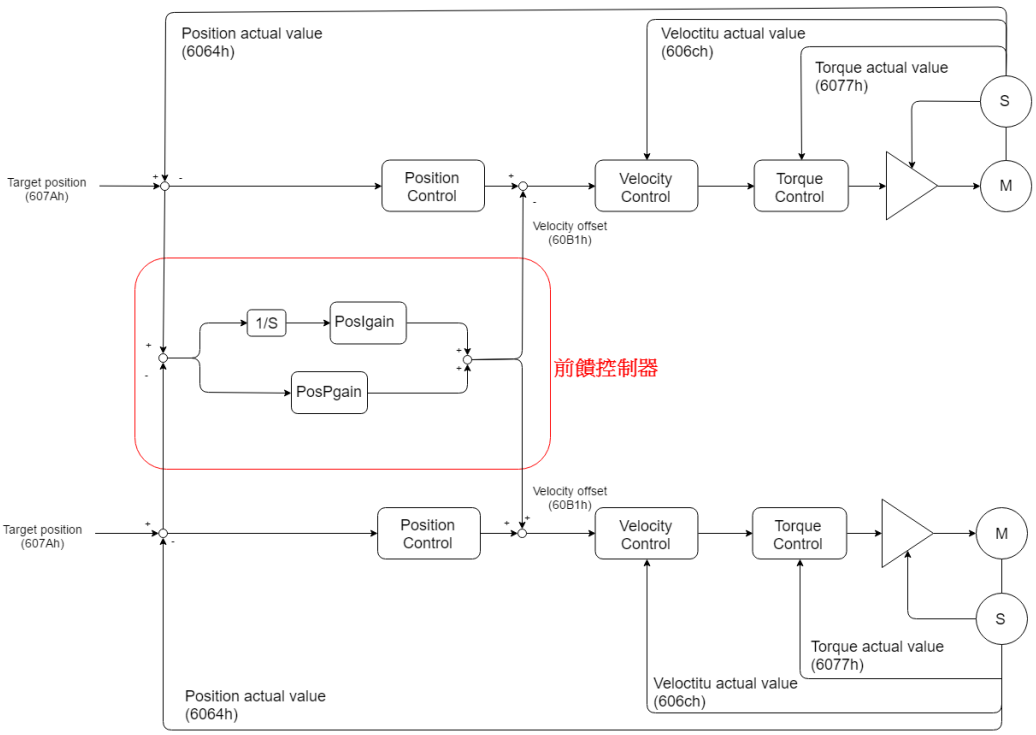
■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	增益写入完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 详细说明请参考第9章

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	增益写入完成时	■ Execute转为False时 ■ 若Execute为False而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
Error / ErrorID	指令的执行条件或输入值发生错误时（错误码记录在ErrorID ）	在Execute下降沿时 (清除ErrorID记录之错误码)

■ 控制方块图



6

6.5.4 群轴功能块（Motion Group）

6.5.4.1 MC_GroupStop

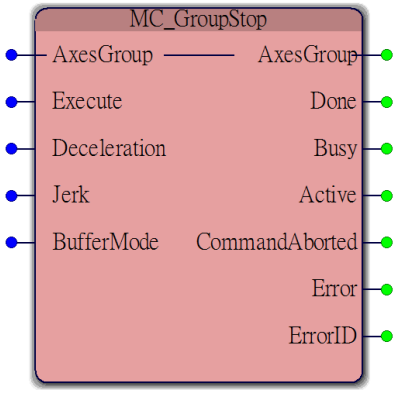
■ 类型

Function Block

■ 功能描述

使所有群组成员轴依设定的减速度减速至停止。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值（默认值）	功能
AxesGroup	GROUP_REF	-	指定轴群组编号

■ 输入脚位

脚位名称	数据型态	设定值（默认值）	功能
Execute	BOOL	True / False (False)	Execute上升沿时，执行此指令
Deceleration	LREAL	正数或0.0 (0.0)	减速度(um/s^2)
Jerk	LREAL	正数或0.0 (0.0)	加加速度(um/s^2)
BufferMode	INT	0 (0)	指定此功能块的缓冲行为 0：中断

■ 输出脚位

脚位名称	数据型态	设定值（默认值）	功能
Done	BOOL	True / False (False)	群组成员轴停止运动，且群组轴状态进入Stopping状态时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Active	BOOL	True / False (False)	群组轴受控制时为True
CommandAborted	BOOL	True / False (False)	指令被中断时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 详细说明请参考第9章

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	群组轴减速至0时	■ Execute由True转为False时 ■ 若Execute为False而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
Active	轴开始减速时	■ 在Execute下降沿且Done为True ■ 在Done上升沿且Execute为False ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
CommandAborted	■ 此功能块指令被其他缓冲模式设为Aborting的指令中断时 ■ 此功能块指令被MC_Stop中断时	■ 在Execute下降沿时 ■ 当Execute为False而CommandAborted转为True，此CommandAborted维持一个扫描周期的True状态后，立刻转为 False
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在 ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

■ 输出脚位的变化时序

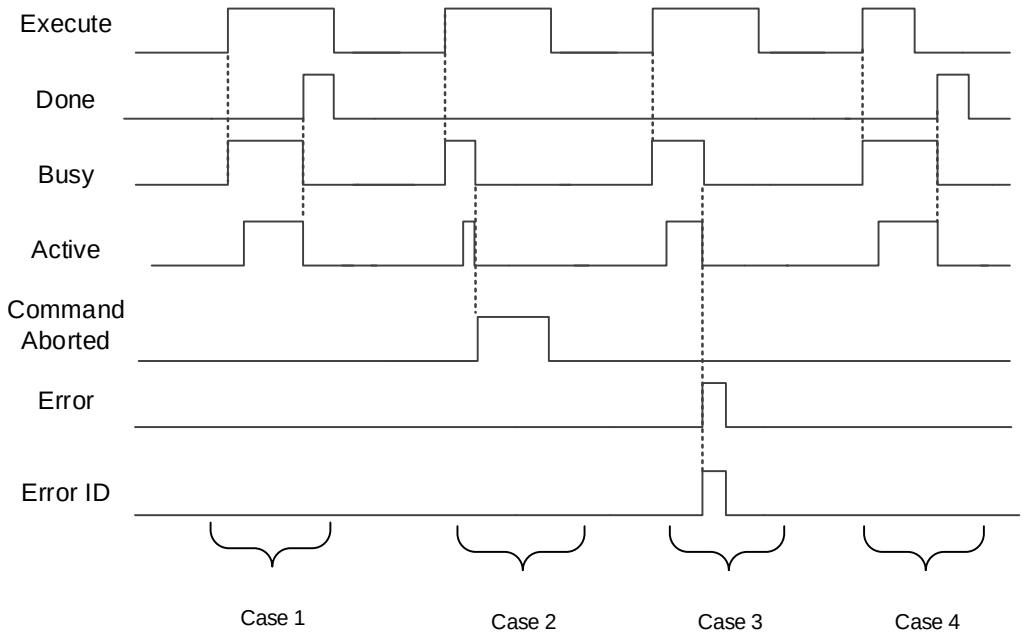


图 6.5.4.1.1 MC_GroupStop 脚位时序图 (时序详细说明请参考 6.4.3 节)

6

Case1：运动功能块完全执行。

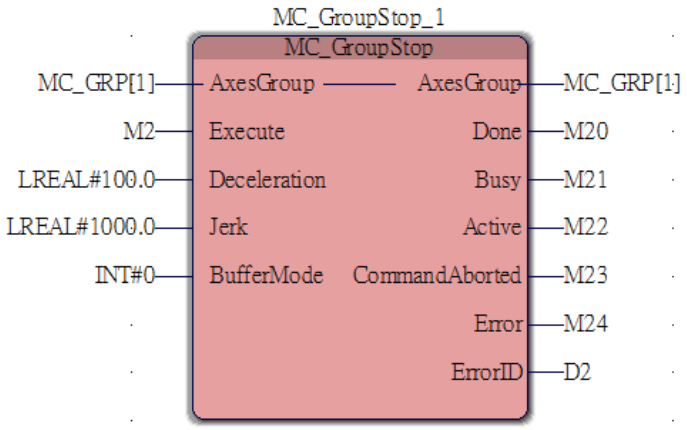
Case2：运动功能块执行过程中被其他运动功能块插断。

Case3：运动功能块执行过程中，有错误产生。

Case4：有 Buffer 输入的运动功能块完全执行。(即使 Execute 已经是 False，功能块仍要完成执行动作，完成时 Done 讯号输出至少要维持一个 PLC 扫描周期。)

■ 参考范例

此范例说明 MC_GroupStop 的运行方式，及执行时的运动轨迹。



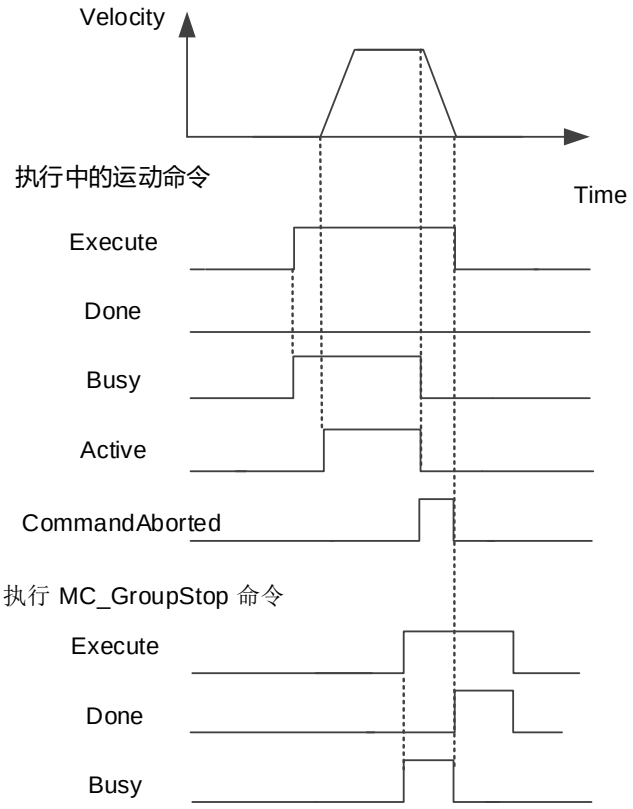


图 6.5.4.1.2 MC_GroupStop 范例说明

6.5.4.2 MC_MoveDirectAbsolute

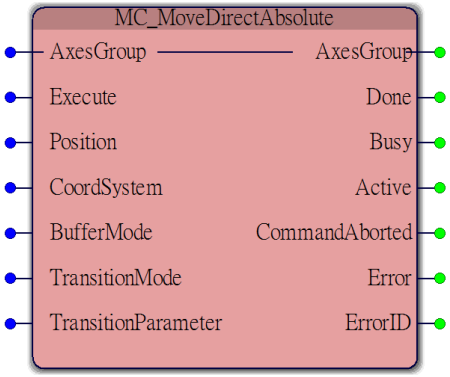
■ 类型

Function Block

■ 功能描述

控制指定群组轴执行无规划路径运动，并移动至指定的绝对目标位置。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
AxesGroup	GROUP_REF	-	指定轴群组编号

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute上升沿时，执行此指令
Position	S_GRP_POS	-	绝对位置。型态为一数组，设定目标空间坐标 X, Y, Z, A, B, C。 *1
CoordSystem	INT	整数 (0)	指定坐标系 1：：大地坐标系 (MCS) 2：：使用者坐标系 (PCS) 3：：工具坐标系 (TCS) 4：：关节坐标系 (ACS)
BufferMode	INT	整数 (0)	指定此功能块的缓冲行为*3 0：中断(Aborting) 1：等待(Buffered) 2：以较低的速度作为中继速度(BlendingLow) 3：以前一笔指令的速度作为中继速度(BlendingPrevious) 4：以后一笔指令的速度作为交接(BlendingNext) 5：以较低的高速作为中继速度(BlendingHigh)

脚位名称	数据型态	设定值 (默认值)	功能
Transition Mode	INT	整数 (0)	指定重迭路径模式 0：无迭合模式 3：转角迭合 A：立即迭合
Transition Parameter	S_MC_TRANSITION_PARAMETER_REF	-	重迭路径参数 (Pulse或%) *2

注:

1. S_GRP_POS 为自定义数据型态，设定目标空间坐标 X, Y, Z, A, B, C，详细请参考 6.5.9 节。
2. S_MC_TRANSITION_PARAMETER_REF 为自定义数据型态，详细请参考 6.5.9 节。
3. BufferMode 详细请参考 6.4.3 节

■ 输出脚位

脚位名称	数据型态	设定值(默认值)	功能
Done	BOOL	True / False (False)	当群组命令执行完成，且群组进入Stopping状态时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Active	BOOL	True / False (False)	群组轴受控制时为True
CommandAborted	BOOL	True / False (False)	指令被中断时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000~0xFFFF (0)	指令错误发生时记录的错误码 错误码详细说明请参考第9章

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	当群组轴减速至0时	■ Execute由True转为False时 ■ 若Execute为False而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	当Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 当CommandAborted上升沿时
Active	当轴开始减速时	■ Execute下降沿且Done为True ■ Done上升沿且Execute为False ■ Error上升沿时 ■ 在CommandAborted上升沿时
CommandAborted	■ 此功能块指令被其他缓冲模式设为Aborting的指令中断时 ■ 此功能块指令被MC_Stop中断时	■ 在Execute下降沿时 ■ 当Execute为False而CommandAborted转为True，此时CommandAborted维持一个扫描周期的True状态后，立刻转为False

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Error / ErrorID	当指令的执行条件或输入值发生错误时(错误码记录在ErrorID)	在Execute下降沿时(清除ErrorID记录之错误码)

■ 输出脚位的变化时序

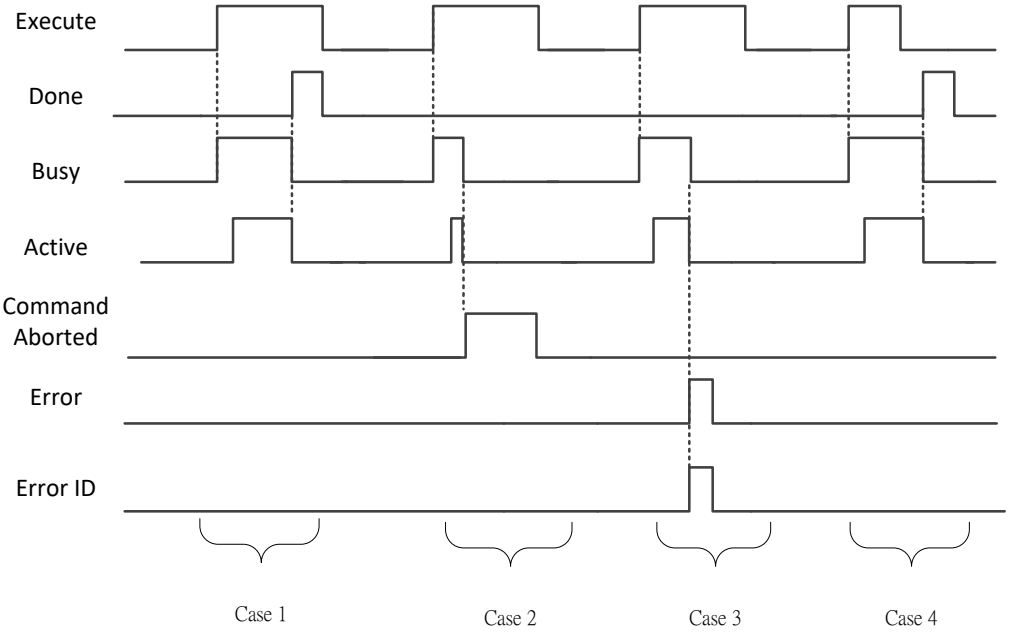


图 6.5.4.2.1 MC_MoveDirectAbsolute 脚位时序图 (时序详细说明请参考 6.4.3 节)

Case1：运动功能块完全执行。

Case2：运动功能块执行过程中被其他运动功能块插断。

Case3：运动功能块执行过程中，有错误产生。

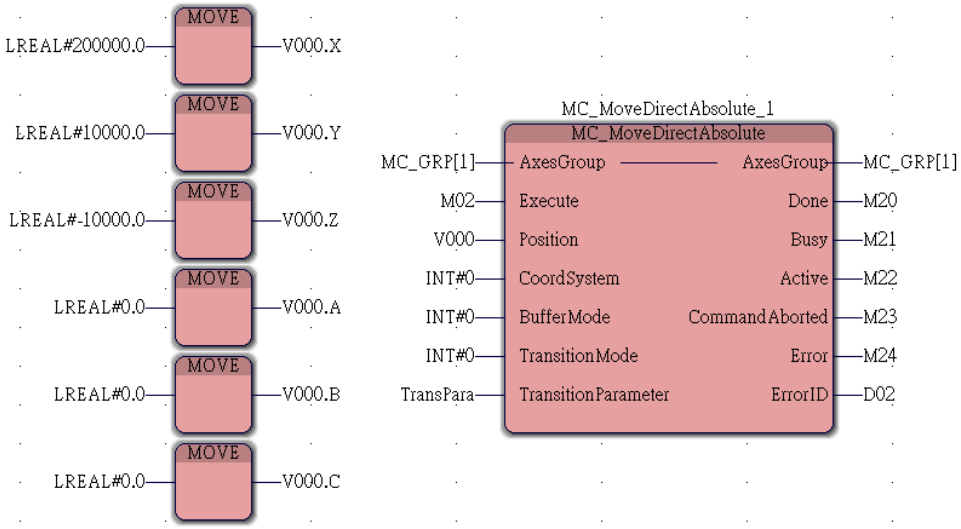
Case4：有 Buffer 输入的运动功能块完全执行。(即使 Execute 已经是 False，功能块仍要完成执行动作，完成时 Done 讯号输出至少要维持一个 PLC 扫描周期。)

6

■ 参考范例

此范例说明 MC_MoveDirectAbsolute 的运行方式及执行时的运动描述。

1. Position 脚位为自定义数据型态 S_GRP_POS，这是一个数组，必须先将欲移动的目标绝对位置以及姿态填入此数组。在此范例中，分别对机器手臂末端点 X 轴坐标 (V00.X)、Y 轴坐标(V00.Y) 、Z 轴坐标(V00.Z) 、A 轴坐标(V00.A) 、B 轴坐标(V00.B) 、C 轴坐标(V00.C) 赋值 200000、10000、-10000、0、0、0。



2. 当 M02 脚位由 False 设定为 True，MC_MoveDirectAbsolute 即控件组轴往设定的绝对位置做直线移动。

3. 当群组轴已抵达指定的绝对位置后，若重复执行此运动指令，群组轴将不做任何移动。

6.5.4.3 MC_MoveDirectRelative

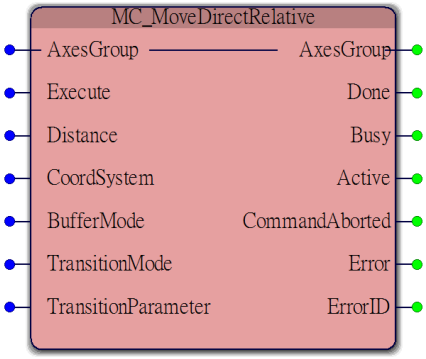
■ 类型

Function Block

■ 功能描述

控制指定群组轴执行无规划路径运动，并移动至指定的相对目标位置。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
AxesGroup	GROUP_REF	-	指定轴群组编号

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute上升沿时，执行此指令
Distance	S_GRP_POSS_GRP_POS	-	相对位置。型态为一数组，设定目标空间坐标 X, Y, Z, A, B, C。*1
CoordSystem	INT	整数 (0)	指定坐标系 1：大地坐标系(MCS) 2：使用者坐标系(PCS) 3：工具坐标系 (TCS) 4：关节坐标系 (ACS)
BufferMode	INT	整数 (0)	指定此功能块的缓冲行为*3 0：中断(Aborting) 1：等待(Buffered) 2：以较低的速度作为中继速度(BlendingLow) 3：以前一笔指令的速度作为中继速度(BlendingPrevious) 4：以后一笔指令的速度作为交接(BlendingNext) 5：以较低的高速作为中继速度(BlendingHigh)

6

脚位名称	数据型态	设定值 (默认值)	功能
Transition Mode	INT	整数 (0)	指定重迭路径模式 0：无迭合模式 3：转角迭合 10：立即迭合
Transition Parameter	S_MC_TRANSITION_PARAMETER_REF	-	重迭路径参数 (Pulse或%) *2

注:

4. S_GRP_POS 为自定义数据型态，设定目标空间坐标 X, Y, Z, A, B, C，详细请参考 6.5.9 节。
5. S_MC_TRANSITION_PARAMETER_REF 为自定义数据型态，详细请参考 6.5.9 节。
6. BufferMode 详细请参考 6.4.3 节

■ 输出脚位

脚位名称	数据型态	设定值(默认值)	功能
Done	BOOL	True / False (False)	当群组命令执行完成，且群组进入Stopping状态时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Active	BOOL	True / False (False)	群组轴受控制时为True
CommandAborted	BOOL	True / False (False)	指令被中断时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码详细说明请参考第9章

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	当群组轴减速至 0 时	■ Execute由True转为False时 ■ 若Execute为False而Done 转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	当 Execute 上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
Active	当轴开始减速时	■ 在Execute下降沿且Done为True ■ 当Done上升沿且Execute为False ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
CommandAborted	■ 此功能块指令被其他缓冲模式设为Aborting的指令中断时 ■ 此功能块指令被MC_Stop中断时	■ 在Execute下降沿时 ■ 当Execute为False而CommandAborted转为True，此时CommandAborted维持一个扫描周期的True状态后，立刻转为False

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在 ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

■ 输出脚位的变化时序

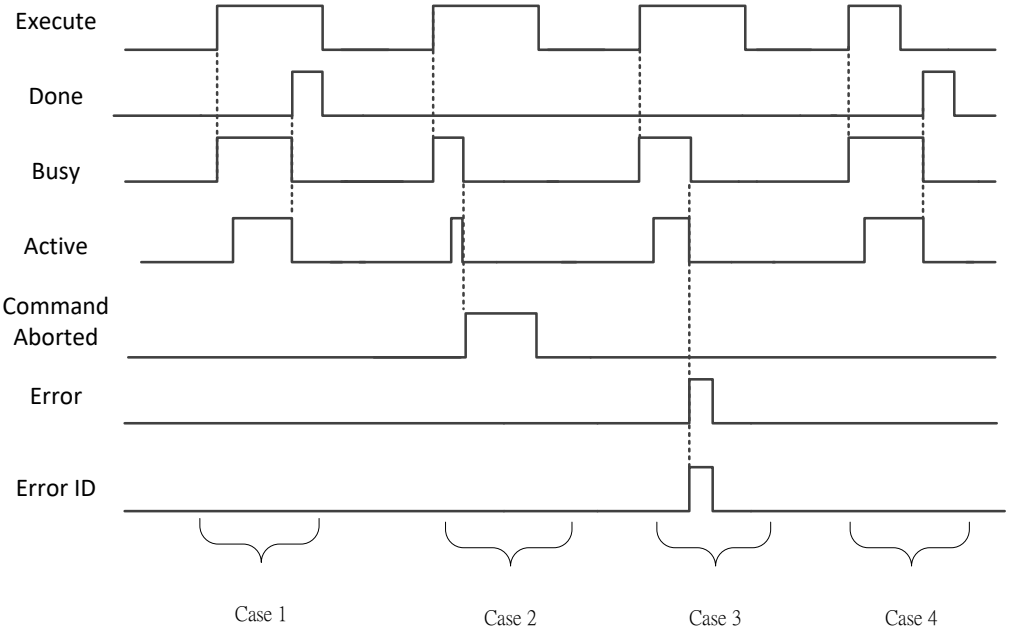


图 6.5.4.3.1 MC_MoveDirectRelative 脚位时序图 (时序详细说明请参考 6.4.3 节)

Case1：运动功能块完全执行。

Case2：运动功能块执行过程中被其他运动功能块插断。

Case3：运动功能块执行过程中，有错误产生。

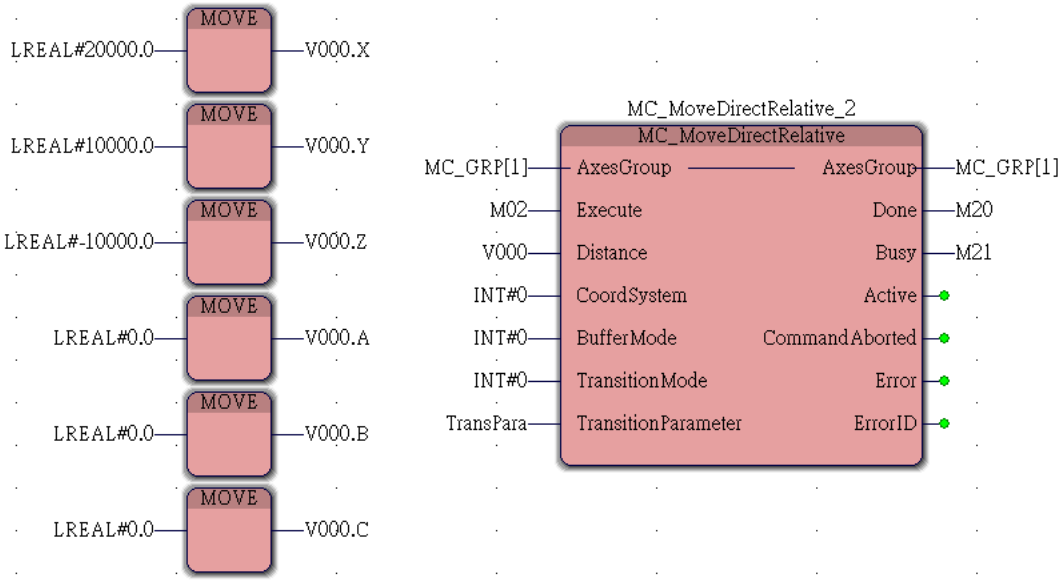
Case4：有 Buffer 输入的运动功能块完全执行。(即使 Execute 已经是 False，功能块仍要执行动作完成，完成时 Done 讯号输出至少要维持一个 PLC 扫描周期。)

6

■ 参考范例

此范例说明 MC_MoveDirectRelative 的运行方式，及执行时的运动描述。

1. Position 脚位为自定义数据型态 S_GRP_POS，这是一个数组，必须先将欲移动的目标相对位置以及姿态填入此数组。在此范例中，分别对机器手臂末端点 X 轴坐标 (V00.X)、Y 轴坐标(V00.Y)、Z 轴坐标(V00.Z)、A 轴坐标(V00.A)、B 轴坐标(V00.B)、C 轴坐标(V00.C) 赋值 200000、10000、-10000、0、0、0。



2. 当 M02 脚位为由 False 设定为 True，MC_MoveDirectRelative 即控件组轴往设定的相对位置做直线移动。
3. 当群组轴移动了相对距离 (20000, 10000, -10000) 后，再次重复触发 M02，运动指令会再次执行一次相对距离。

6.5.4.4 MC_MoveLinearAbsolute

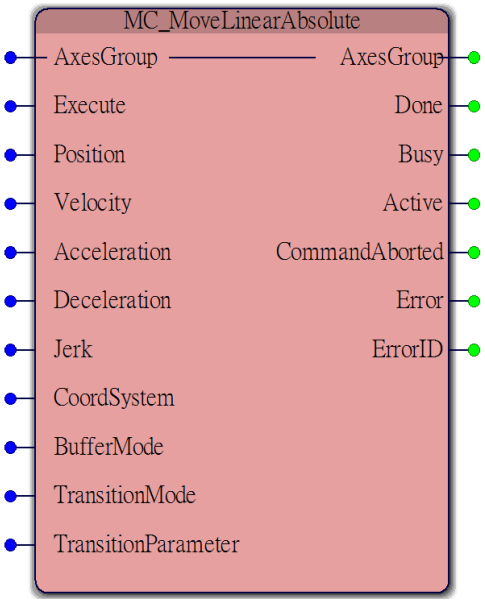
■ 类型

Function Block

■ 功能描述

控制指定群组轴执行直线运动，移动至指定的绝对目标位置。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
AxisGroup	GROUP_REF	-	指定轴群组编号

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute 上升沿时，执行此指令
Position	S_GRP_POS	-	相对位置。型态为一数组，设定目标空间坐标 X, Y, Z, A, B, C。*1
Velocity	LREAL	正整数或0 (0)	目标速度 (um/s)
Acceleration	LREAL	正整数或0 (0)	加速度 (um/s^2)
Deceleration	LREAL	正整数或0 (0)	减速度 (um/s^2)
Jerk	LREAL	正整数或0 (0)	加加速度 (um/s^3)
CoordSystem	INT	整数 (0)	指定坐标系 1：大地坐标系 (MCS) 2：使用者坐标系 (PCS) 3：工具坐标系 (TCS) 4：关节坐标系 (ACS)

6

脚位名称	数据型态	设定值 (默认值)	功能
BufferMode	INT	整数 (0)	指定此功能块的缓冲行为*3 0：中断(Aborting) 1：等待(Buffered) 2：以较低的速度作为中继速度(BlendingLow) 3：以前一笔指令的速度作为中继速度(BlendingPrevious) 4：以后一笔指令的速度作为交接(BlendingNext) 5：以较低的高速作为中继速度(BlendingHigh)
Transition Mode	INT	整数 (0)	指定重迭路径模式 0：无迭合模式 3：转角迭合 10：立即迭合
Transition Parameter	S_MC_TRANSITION_PARAMETER_REF	-	重迭路径参数 (Pulse或%)*2

注:

1. S_GRP_POS 为自定义数据型态，设定目标空间坐标 X, Y, Z, A, B, C，详细请参考 6.5.9 节。
2. S_MC_TRANSITION_PARAMETER_REF 为自定义数据型态，详细请参考 6.5.9 节。
3. BufferMode 详细请参考 6.4.3 节。

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	当群组命令执行完成，且群组进入Stopping状态时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Active	BOOL	True / False (False)	群组轴受控制时为True
Command Aborted	BOOL	True / False (False)	指令被中断时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码详细说明请参考第9章

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	群组轴减速至0时	■ Execute由True转为False时 ■ 若Execute为False而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Active	轴开始减速时	■ Execute下降沿且Done为True ■ 在Done上升沿且Execute为False ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
CommandAborted	■ 此功能块指令被其他缓冲模式设为Aborting的指令中断时 ■ 此功能块指令被MC_Stop中断时	■ 在Execute下降沿时 ■ Execute为False而CommandAborted转为True，此时CommandAborted维持一个扫描周期的True状态后，立刻转为False
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在 ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

■ 输出脚位的变化时序

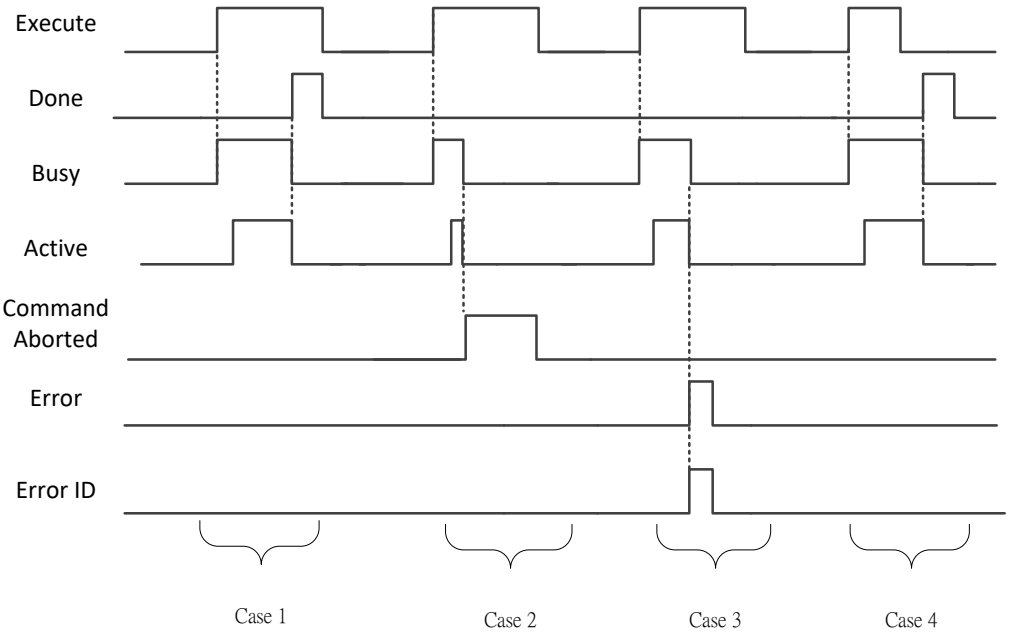


图 6.5.4.4.1 MC_MoveLinearAbsolute 脚位时序图 (时序详细说明请参考 6.4.3 节)

Case1：运动功能块完全执行。

Case2：运动功能块执行过程中被其他运动功能块插断。

Case3：运动功能块执行过程中，有错误产生。

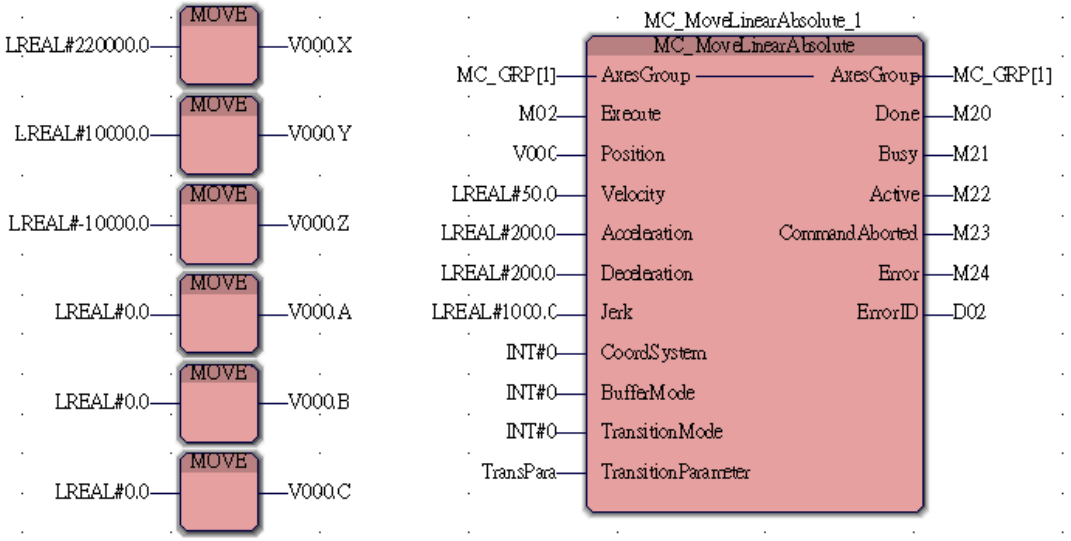
Case4：有 Buffer 输入的运动功能块完全执行。(即使 Execute 已经是 False，功能块仍要完成执行动作，完成时 Done 讯号输出至少要维持一个 PLC 扫描周期。)

6

■ 参考范例

此范例说明 MC_MoveLinearAbsolute 的运行方式及执行时的运动描述。

1. Position 脚位为自定义数据型态 S_GRP_POS，这是一个数组，必须先将欲移动的目标绝对位置以及姿态填入此数组。在此范例中，分别对机器手臂末端点 X 轴坐标 (V00.X)、Y 轴坐标(V00.Y)、Z 轴坐标(V00.Z)、A 轴坐标(V00.A)、B 轴坐标(V00.B)、C 轴坐标(V00.C)赋值 200000、10000、-10000、0、0、0。



2. 当 M02 脚位由 False 设定为 True，MC_MoveLinearAbsolute 即控件组轴往设定的绝对位置做直线移动。
3. 当群组轴已抵达指定的绝对位置后，若重复执行此运动指令，群组轴将不做任何移动。

6.5.4.5 MC_MoveLinearRelatve

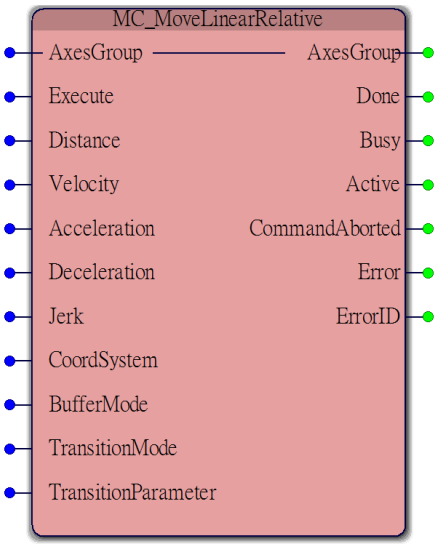
■ 类型

Function Block

■ 功能描述

控制指定群组轴执行直线运动，移动至指定的相对目标位置。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
AxisGroup	GROUP_REF	-	指定轴群组编号

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute上升沿时，执行此指令
Position	S_GRP_POS	-	相对位置。型态为一数组，设定目标空间坐标 X, Y, Z, A, B, C。*1
Velocity	LREAL	正整数或0 (0)	目标速度 (um/s)
Acceleration	LREAL	正整数或0 (0)	加速度 (um/s^2)
Deceleration	LREAL	正整数或0 (0)	减速度 (um/s^2)
Jerk	LREAL	正整数或0 (0)	加加速度 (um/s^3)
CoordSystem	INT	整数 (0)	指定坐标系 1：大地坐标系 (MCS) 2：使用者坐标系 (PCS) 3：工具坐标系 (TCS) 4：关节坐标系 (ACS)

脚位名称	数据型态	设定值 (默认值)	功能
BufferMode	INT	整数 (0)	指定此功能块的缓冲行为*3 0：中断 (Aborting) 1：等待 (Buffered) 2：以较低的速度作为中继速度 (BlendingLow) 3：以前一笔指令的速度作为中继速度 (BlendingPrevious) 4：以后一笔指令的速度作为交接(BlendingNext) 5：以较低的高速作为中继速度 (BlendingHigh)
Transition Mode	INT	整数 (0)	指定重迭路径模式 0：无迭合模式 3：转角迭合 10：立即迭合
Transition Parameter	S_MC_TRANSITION_PARAMETER_REF	-	重迭路径参数 (Pulse或%) *2

注:

1. S_GRP_POS 为自定义数据型态，设定目标空间坐标 X, Y, Z, A, B, C，详细请参考 6.5.9 节。
2. S_MC_TRANSITION_PARAMETER_REF 为自定义数据型态，详细请参考 6.5.9 节。
3. BufferMode 详细请参考 6.4.3 节。

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	当群组命令执行完成，且群组进入Stopping状态时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Active	BOOL	True / False (False)	群组轴受控制时为True
Command Aborted	BOOL	True / False (False)	指令被中断时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 错误码详细说明请参考第9章

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	群组轴减速至0时	■ Execute由True转为False时 ■ 若 Execute 为 False 而 Done 转为 True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	当Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
Active	当轴开始减速时	■ 在Execute下降沿且Done为True ■ 当Done上升沿且Execute为False ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
CommandAborted	■ 此功能块指令被其他缓冲模式设为Aborting的指令中断时 ■ 此功能块指令被MC_Stop中断时	■ 在Execute下降沿时 ■ 当Execute为False而CommandAborted转为True，此时CommandAborted 维持一个扫描周期的True状态后，立刻转为False
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

■ 输出脚位的变化时序

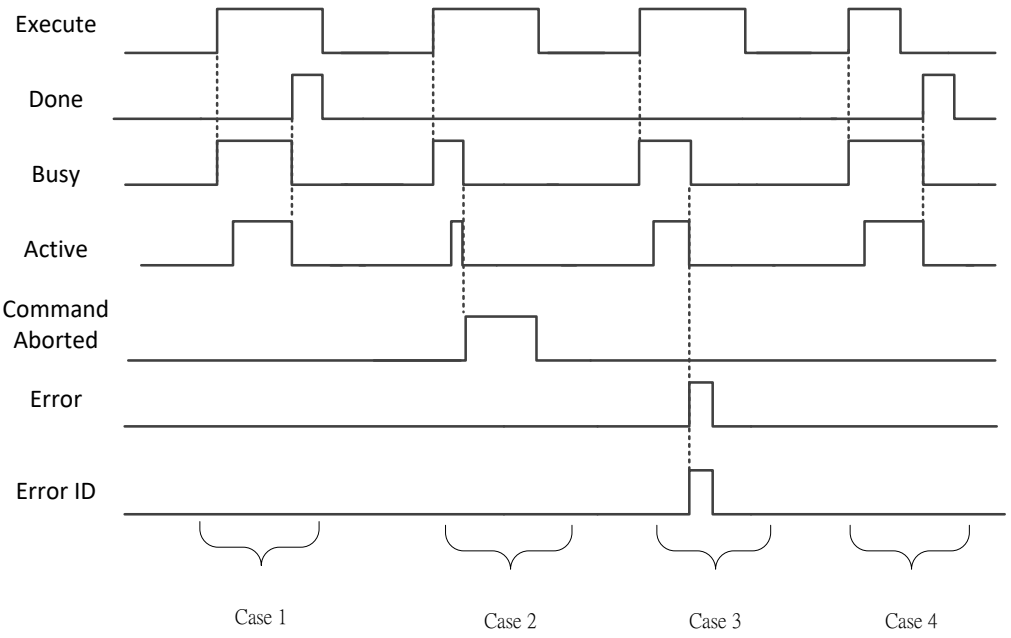


图 6.5.4.5.1 MC_MoveLinearRelative 脚位时序图 (时序详细说明请参考 6.4.3 节)

Case1：运动功能块完全执行。

Case2：运动功能块执行过程中被其他运动功能块插断。

Case3：运动功能块执行过程中，有错误产生。

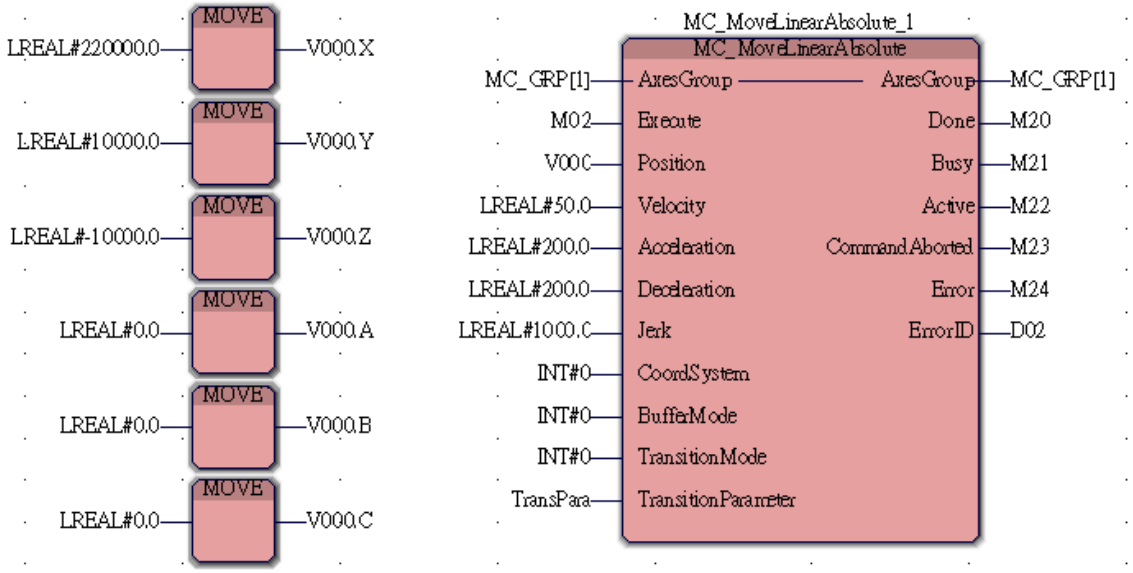
Case4：有 Buffer 输入的运动功能块完全执行。(即使 Execute 已经是 False，功能块仍要完成执行动作，完成时 Done 讯号输出至少要维持一个 PLC 扫描周期。)

6

■ 参考范例

此范例说明 MC_MoveLinearRelative 的运行方式，及执行时的运动描述。

1. Position 脚位为自定义数据型态 S_GRP_POS，这是一个数组，必须先将欲移动的目标相对位置以及姿态填入此数组。在此范例中，分别对机器手臂末端点 X 轴坐标 (V00.X)、Y 轴坐标(V00.Y)、Z 轴坐标(V00.Z)、A 轴坐标(V00.A)、B 轴坐标(V00.B)、C 轴坐标(V00.C)赋值 200000、10000、-10000、0、0、0。



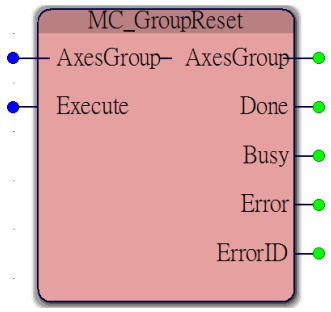
2. 当 M02 脚位为由 Fasle 设定为 True，MC_MoveLinearRelative 即控件组轴往设定的相对位置做直线移动。
3. 当群组轴移动了相对距离 (6000, 7000, -8000)后，再次重复触发 M2，运动指令将会再次执行一次相对距离移动。

6.5.4.6 MC_GroupReset

- 类型
  Function Block

- 功能描述
  清除或重置所有群组以及群组成员轴的错误。

- 图形表示



- 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
AxisGroup	GROUP_REF	-	指定轴群组编号

- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute上升沿时，执行此指令

- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	完成错误清除时为True；反之则为False
Busy	BOOL	True / False (False)	指令被触发执行时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 错误码详细说明请参考第9章

6

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	当群组轴减速至0时	■ Execute由True转为False时 ■ 若Execute为False而Done 转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

■ 输出脚位的变化时序

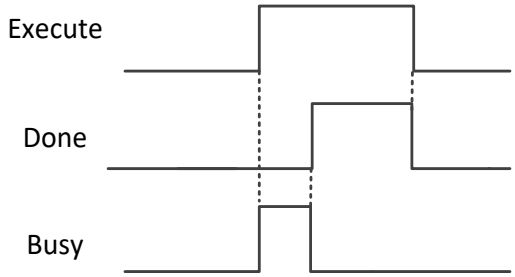


图 6.5.4.6.1 MC_GroupReset 脚位时序图 (时序详细说明请参考 6.4.3 节)

■ 参考范例

此范例说明 MC_GroupReset 使用方式，M2 为被启动后，功能块会进行清除编号一号群组及群组中成员轴的报警。

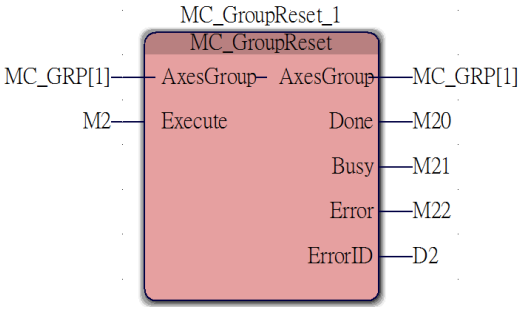


图 6.5.4.1.2 MC_GroupStop 范例说明

6.5.4.7 MC_GroupSetOverride

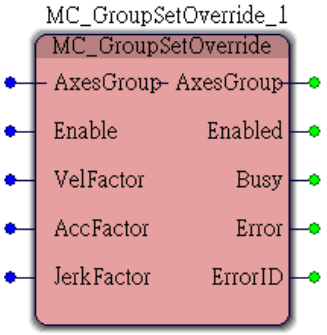
■ 类型

Function Block

■ 功能描述

透过超驰控制(Override control)系数并依所设定的加减速度及加加速度来改变群组成员轴的速度。

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
AxisGroup	GROUP_REF	-	指定轴群组编号

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Enable	BOOL	True / False (False)	Enable为True时，启用此功能
VelFactor	REAL	正数或0 (1.0)	超驰控制速度系数
AccFactor	REAL	正数 (1.0)	超驰控制系数，正数为加速度系数，负数为减速度系数
JerkFactor	REAL	正数 (1.0)	超驰控制加加速度系数

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Enabled	BOOL	True / False (False)	True时表示超驰控制设立成功
Busy	BOOL	True / False (False)	指令执行时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 错误码详细说明请参考第9章

6

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Enabled	设定值成立后	■ 当Execute转为False时 ■ 若Execute为False而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

■ 输出脚位的变化时序

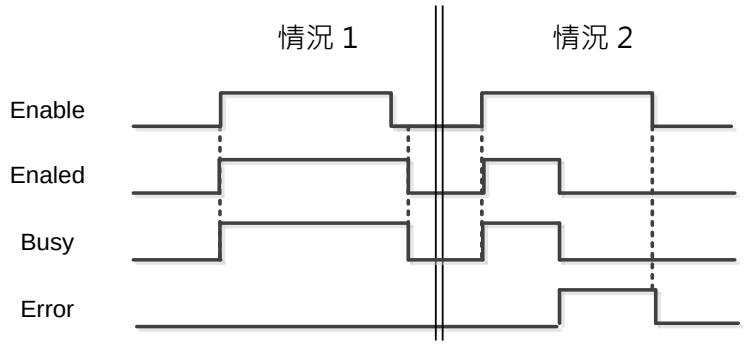


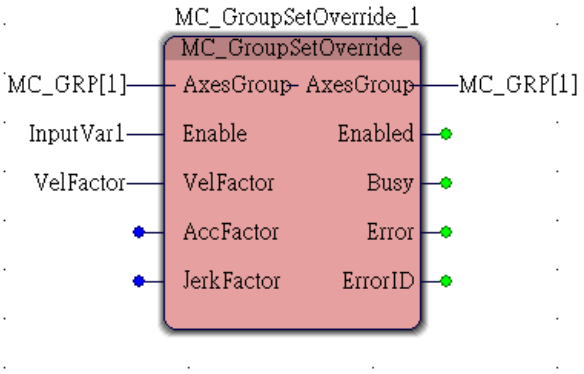
图 6.5.2.14.1 MC_SetOverride 脚位时序图 (时序详细说明请参考 6.4.3 节)

情况 1：当此指令第一次执行(Enable 由 FALSE 变为 TRUE)时，Busy 与 Enabled 脚位在同一时间变为 TRUE 表示指令已被执行。当 Enable 由 TRUE 变为 FALSE 时，Done 维持一个周期后变为 FALSE。

情况 2：当此指令执行的过程中发生错误，则 Error 变为 TRUE，同时 Busy 变为 FALSE，此时请观察 ErrorID 脚位所显示的错误码，了解详细错误讯息。Error 与 ErrorID 脚位将一直维持 TRUE 直到 Enabled 脚位变为 FALSE。

■ 参考范例

此范例透过外部讯号控制，来使群组运动速度的比例值生效。



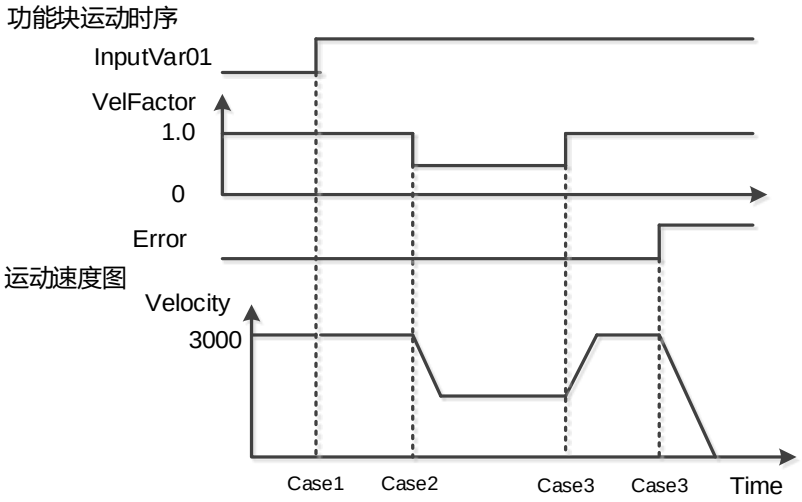


图 6.5.2.14.2 MC_SetOverride 使用参考范例

1. InputVar01 连接外部命令 (如数字输入脚位或内存位置)。
2. VelFactor 连接外部命令 (如内存位置)。
3. VelFactor 变动影响轴运动速度

   Case1：VelFactor = 1，群组轴速度维持 100%

   Case2：VelFactor = 0.5，群组轴速度降速为 50%

   Case3：VelFactor = 1，群组轴速度升速至 100%

   Case4：VelFactor = 0，群组轴速度降速至 0%
4. 群组轴速度因为错误产生，导致不会变更至 100%而保持在 0%

6

6.5.5 系统功能块（System Function)

6.5.5.1 SYS_ParaRead

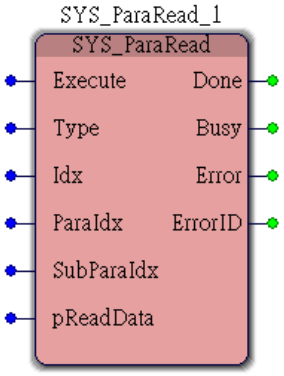
■ 类型

Function Block

■ 功能描述

读取各种系统参数，如控制器、轴、群组、总线装置等。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值（默认值）	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，触发此功能
Type	INT	整数（0)	对象类型代号 详细请参考第5章
Idx	INT	整数（0)	对象编号(站号) 详细请参考第5章
ParaIdx	INT	整数（0)	参数主编号 详细请参考第5章
SubParaIdx	INT	整数（0)	参数次编号 详细请参考第5章

■ 输出脚位

脚位名称	数据型态	设定值（默认值）	功能
Done	BOOL	True / False (False)	完成时为True；反之则为False
Busy	BOOL	True / False (False)	成立时表示此功能块有效
Error	BOOL	True / False (False)	当错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	错误发生时所记录的错误码 错误码详细说明请参考第9章

脚位名称	数据型态	设定值 (默认值)	功能
pReadData	ANY	-	读取结果数值。 根据读取对象的不同，需要设定不同的数据型态。详细说明请参考第5章注

注：pReadData 虽然放在功能块的左边，但是此脚位为输出脚位。

■ 参考范例

此范例示范利用 SYS_ParaRead 功能块读取控制器 IP 地址。

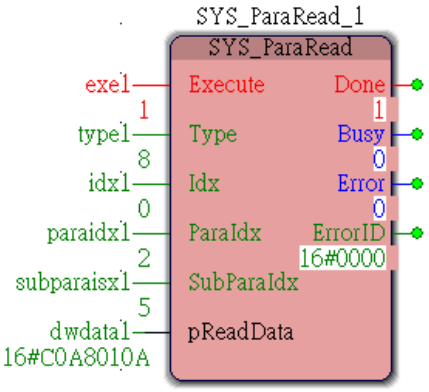


图 6.5.5.1.1 SYS_ParaRead 使用参考范例

6.5.5.2 SYS_ParaWrite

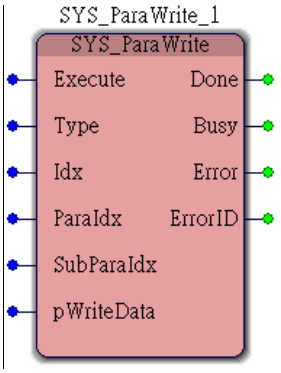
■ 类型

Function Block

■ 功能描述

写入各种系统参数，如控制器、轴、群组、总线装置等。

■ 图形表示



6

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，触发此功能
Type	INT	整数 (0)	对象类型代号 详细请参考第5章
Idx	INT	整数 (0)	对象编号(站号) 详细请参考第5章
ParaIdx	INT	整数 (0)	参数主编号 详细请参考第5章
SubParaIdx	INT	整数 (0)	参数次编号 详细请参考第5章

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	参数写入完成时为True；反之则为False
Busy	BOOL	True / False (False)	Busy脚位为True时，表示此功能块有效
Error	BOOL	True / False (False)	当错误发生时为True
ErrorID	WORD	0x0000~0xFFFF (0)	错误发生时记录的错误码 错误码详细说明请参考第9章
pWriteData	ANY	-	写入数值来源。根据写入对象的不同，需要设定不同的数据型态。详细说明请参考第5章注

注：pWriteData 虽然放在功能块的左边，但是此脚位为输出脚位。

■ 参考范例

此范例示范利用 SYS_ParaWrite 功能块写入控制器 IP 地址。

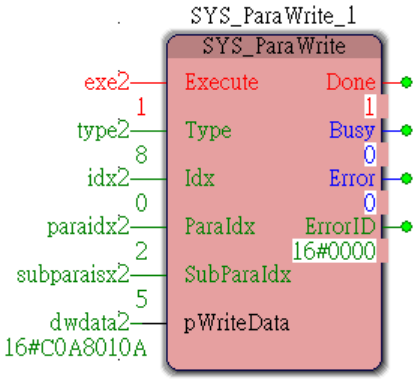


图 6.5.5.1.1 SYS_ParaWrite 使用参考范例

6.5.5.3 SYS_GetAlarmCode

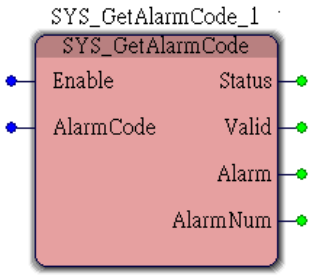
■ 类型

Function Block

■ 功能描述

读取当前控制器错误码。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，触发此功能

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Status	BOOL	True / False (False)	伺服状态，伺服启动时为True；反之则为False
Valid	BOOL	True / False (False)	Valid脚位为True时，表示此功能块有效。
Alarm	BOOL	True / False (False)	当错误发生时为True
AlarmNum	INT	0 ~ 32 (0)	目前错误的数量
AlarmCode	ANY	-	目前控制器的错误码 由于控制器错误码数据型态为DWORD，此脚位的数据型态建议为DWORD或是DWORD数组[注]

注：AlarmCode 虽然放在功能块的左边，但是此脚位为输出脚位。

6

■ 参考范例

此范例示范利用 SYS_GetAlarmCode 功能块读取控制器的错误码。

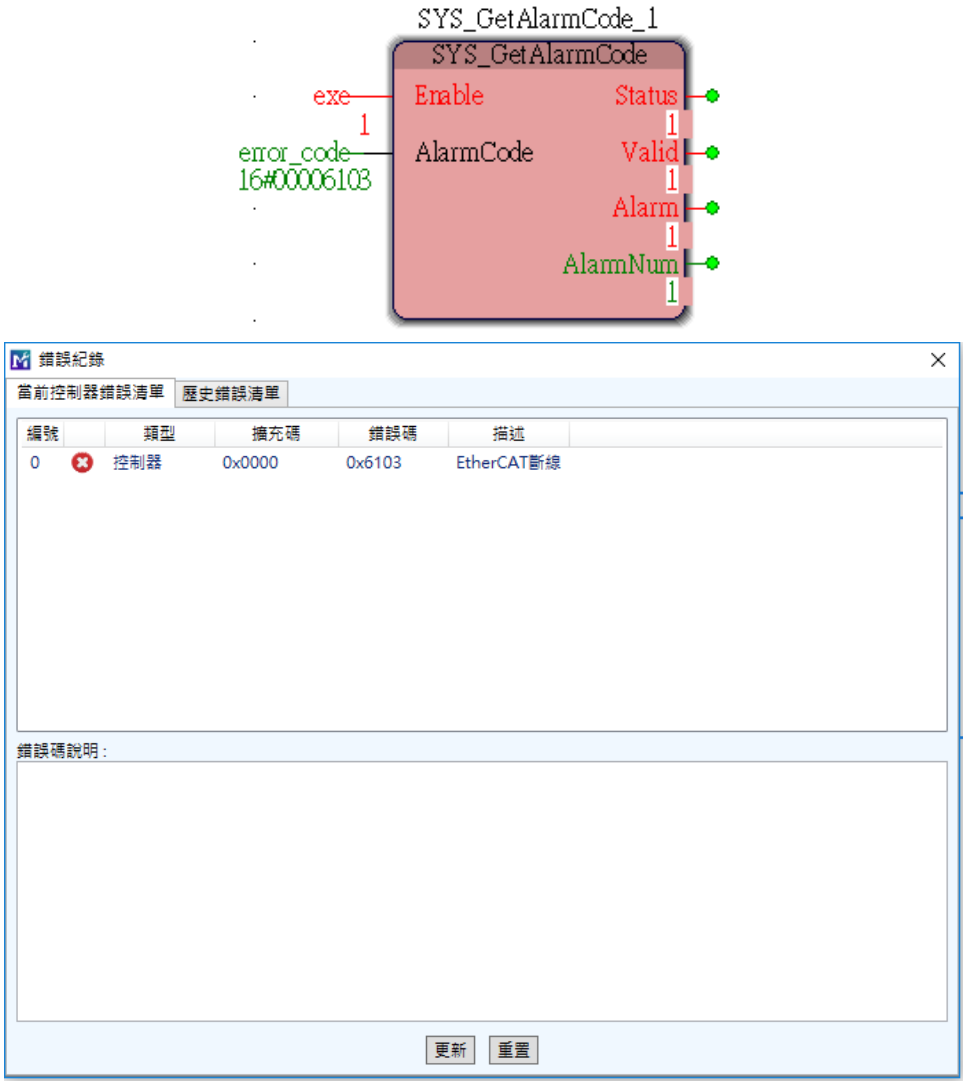


图 6.5.5.3.1 SYS_ GetAlarmCode 使用参考范例

6.5.5.4 SYS_ResetAlarmCode

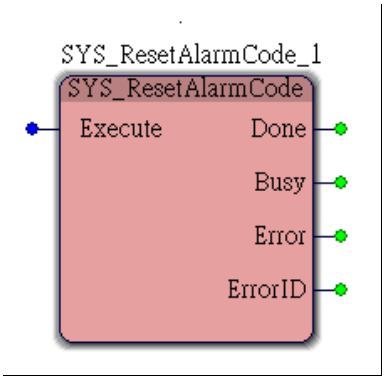
■ 类型

Function Block

■ 功能描述

清除当前控制器错误码。

■ 图形表示



■ 输入脚位

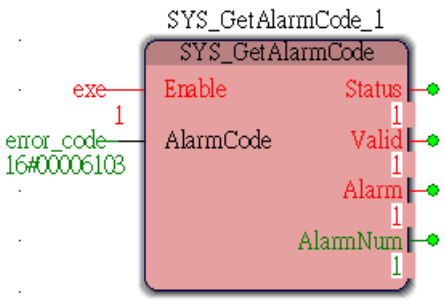
脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，触发此功能

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	完成时为True；反之则为False
Busy	BOOL	True / False (False)	Busy脚位为True时，表示此功能块有效
Error	BOOL	True / False (False)	当错误发生时为True
ErrorID	WORD	0x0000~0xFFFF (0)	错误发生时记录的错误码 错误码详细说明请参考第9章

■ 参考范例

以下示范利用 SYS_ResetAlarmCode 功能块读取控制器错误码。



6

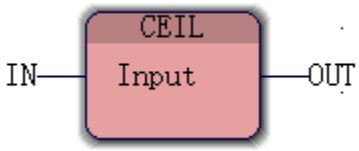
6.5.6 自定义系统功能块（DMC Function)

6.5.6.1 CEIL

- 类型
  Function

- 功能描述
  将使用者输入的数值无条件进位至整数。

- 图形表示



- 输入脚位

脚位名称	数据型态	设定值（默认值)	功能
IN	LREAL	正数、负数或0.0 (0.0)	输入的数值

- 输出脚位

脚位名称	数据型态	设定值（默认值)	功能
OUT	LREAL	正数、负数	经计算后之结果

- 参考范例
  此范例中，CEIL 的输入数值为 3.1，经过 CEIL 运算后，输出结果(OUT)为 4.0。

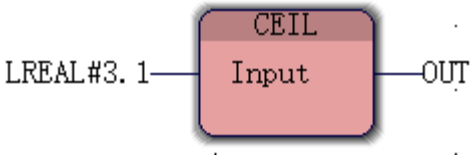


图 6.5.6.1.1 CEIL 范例说明

6.5.6.2 FLOOR

- 类型
  Function
- 功能描述
  将使用者输入的数值无条件舍去至整数。
- 图形表示

- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN	LREAL	正数、负数或0.0 (0.0)	输入的数值

- 输出脚位

脚位名称	数据型态	设定值(默认值)	功能
OUT	LREAL	正数、负数或0.0 (0.0)	经计算后输出之数值

- 参考范例
  此范例中，FLOOR 的输入数值为 3.6，经过 FLOOR 后的运算结果 OutData 为 3.0。

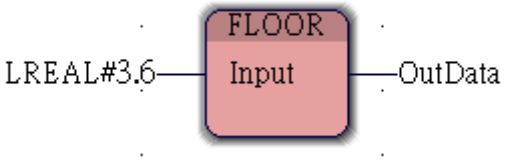
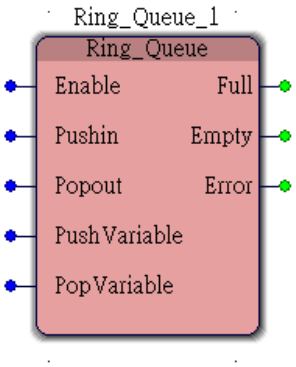


图 6.5.6.2.1 FLOOR 范例说明

6.5.6.3 Ring_Queue

- 类型
  Function block
- 功能描述
  实现一长度为 64 的环状数据序列，存取该环状数据序列采用先进先出法（First In, First Out，FIFO)。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Enable	BOOL	True / False (False)	■ Enable为上升沿时，初始化环状资料序列 ■ Enable为True时，启用环状数据序列功能 ■ Enable为False时，关死循环状数据序列功能
Pushin	BOOL	True / False (False)	■ 若Enable为True，此变量上升沿时，将PushVariable变量所储存的数据塞入环状数据序列尾端 ■ 若Enable为False时，此变数无作用
Popout	BOOL	True / False (False)	■ 若Enable为True时，此变数上升沿时，将环状资料序列前端的数据抛至变量 PopVariable ■ 若Enable为False时，此变数无作用
PushVariable	Any	-	连接欲塞入环状数据序列尾端的资料变数
PopVariable	Any	-	连接抛出环状数据序列数据的目标变量

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Full	BOOL	True / False (False)	当Pushin脚位上升沿时，环状数据序列已塞满64笔数据，Full则为True
Empty	BOOL	True / False (False)	当Popout脚位上升沿时，环状资料序列无任何资料，Empty则为True
Error	BOOL	True / False (False)	当错误发生时为True

■ 参考范例

以下为 Ring_Queue 各个脚位时序及实际环状数据序列变化示意图，并以布尔变量作为范例。

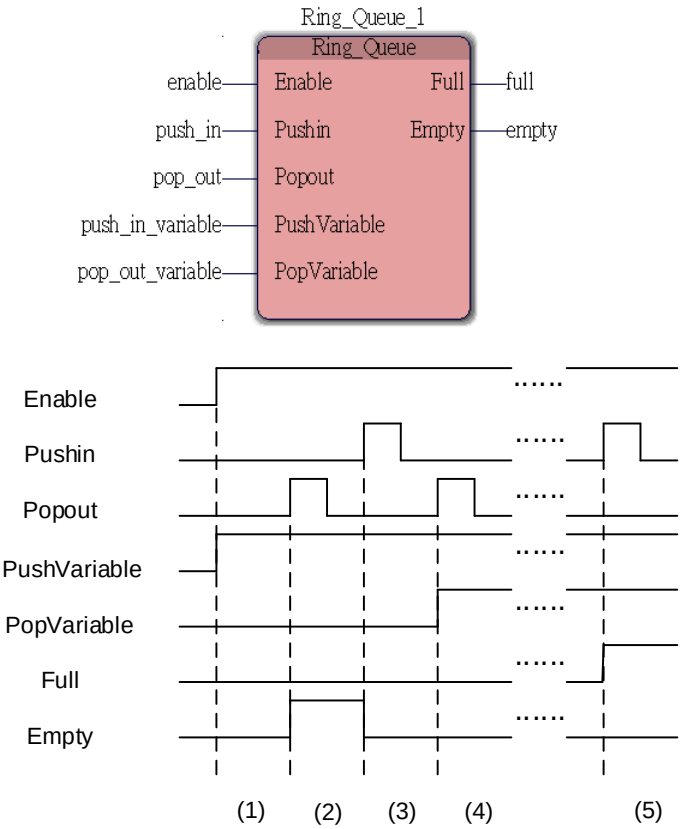


图 6.5.6.3.1 Ring_Queue 范例说明

(1) 初始化环状资料序列

0	0	0	…	…	0	0	0

(2) 当 Popout 脚位上升沿时，环状资料序列无任何资料，则 Empty 脚位为 True

0	0	0	…	…	0	0	0

(3) 当 Pushin 脚位上升沿时，将 PushVariable 变量所储存的数据塞入环状数据序列尾端

1	0	0	…	…	0	0	0

(4) 当 Popout 脚位上升沿时，将环状资料序列前端的数据抛至变量 PopVariable (FIFO)

0	0	0	…	…	0	0	0

6

(5) 若环状数据序列已塞满 64 笔数据，当 Pushin 脚位上升沿时，则 Full 脚位为 True

1	1	1	…	…	1	1	1

6.5.7 自定义通讯功能块（DMC FieldBus）

6.5.7.1 CAN_PDORead

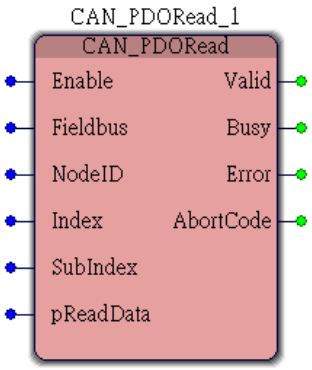
■ 类型

Function Block

■ 功能描述

进行 PDO 读取功能。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值（默认值）	功能
Enable	BOOL	True / False (False)	Enable为True时，执行PDO读取功能。
Fieldbus	UINT	0 ~ 2 (0)	总线类型。 0：EtherCAT 1：DMCNET (台达总线) 2：CANOpen
NodeID	UINT	整数（0）	连接装置之指定站号。
Index	UINT	整数（0）	连接装置之OD主编号。注
SubIndex	USINT	整数（0）	连接装置之OD次编号。注

注：关于 OD (Object Dictionary) 主编号、次编号以及数据型态，请参考该连接装置的手册。

■ 输出脚位

脚位名称	数据型态	设定值（默认值）	功能
Valid	BOOL	True / False (False)	输出有效时为True；反之则为False。
Busy	BOOL	True / False (False)	脚位为True时，表示此功能块进行中。
Error	BOOL	True / False (False)	错误发生时为True。
AbortCode	DWORD	0x00000000 ~ 0xFFFFFFFF (0)	错误发生时记录的错误码 详细说明请参考第9章

脚位名称	数据型态	设定值 (默认值)	功能
pReadData	ANY	-	读取数值。根据写入的OD码对象的不同，需要设定不同的数据型态。注

注：pReadData 虽然放在功能块的左边，但是此脚位为输出脚位。该 OD 码所对应的数据型态，请参考该连接装置的手册。若读取的数据型态不符合该 OD 码，则 AbortCode 会发出报警 0x06070013。

■ 参考范例

此范例示范透过 EtherCAT 总线，读取台达伺服参数 OD 码。

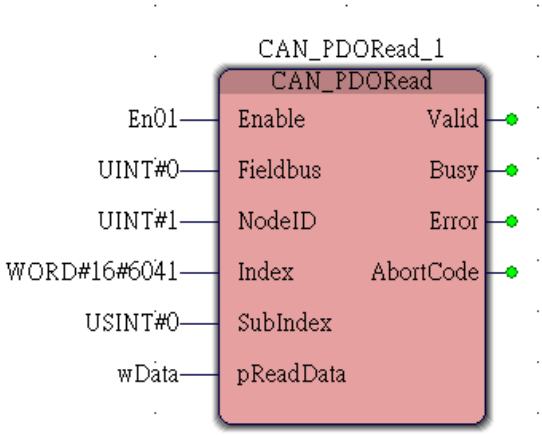


图 6.5.7.1.1 CAN_PDORead 使用参考范例

1. 根据欲读取的数据格式将 wData 的变量格式设为 DWORD。
2. 将 En01 脚位由 Fasle 设定为 True。
3. 此功能块执行成功，输出脚位 Valid 变为 True，CAN_PDORead 功能块读取 EtherCAT 总在线的 OD 码内容，其指定站号为 1、主编号为 0x6041、次编号为 0。(此 OD 码为 Statusword)
4. 此 OD 码动作执行成功，Vaild 和 Busy 脚位变为 True，wData 读回的内容为 0x0650

6.5.7.2 CAN_PDOWrite

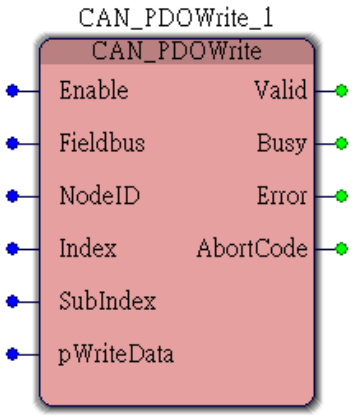
■ 类型

Function Block

■ 功能描述

进行 PDO 写入功能。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Enable	BOOL	True / False (False)	Enable为True时，执行PDO写入此功能。
Fieldbus	UINT	0 ~ 2 (0)	总线类型。 0：EtherCAT 1：DMCNET(台达总线) 2：CANOpen
NodeID	UINT	整数 (0)	连接装置之指定站号。
Index	UINT	整数 (0)	连接装置之OD主编号。
SubIndex	USINT	整数 (0)	连接装置之OD次编号。
pWriteData	ANY	-	写入数值。根据写入的OD码对象的不同，需要设定不同的数据型态。

注：关于 OD (Object Dictionary) 的主编号、次编号以及数据型态，请参考该连接装置的手册。

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Valid	BOOL	True / False (False)	输出有效时为True；反之则为False。
Busy	BOOL	True / False (False)	脚位为True时，表示此功能块进行中。
Error	BOOL	True / False (False)	错误发生时为True。
AbortCode	DWORD	0x00000000 ~ 0xFFFFFFFF (0)	错误发生时记录的错误码 详细说明请参考第9章

6

■ 参考范例

此范例示范使用 EtherCAT 总线对连接的台达伺服，写入所指定 OD 码，使伺服进行 Servo Off 的动作。

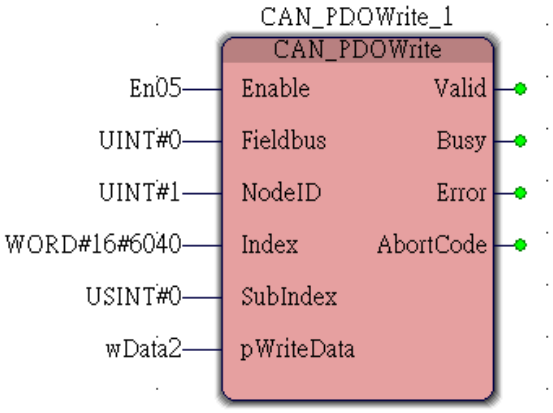
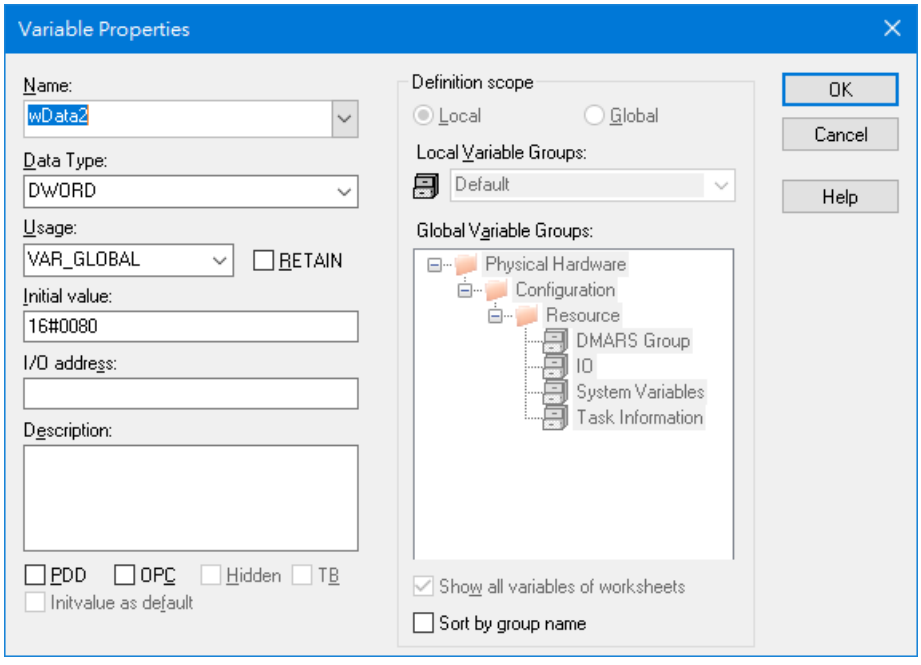


图 6.5.7.2.1 CAN_PDOWrite 使用参考范例

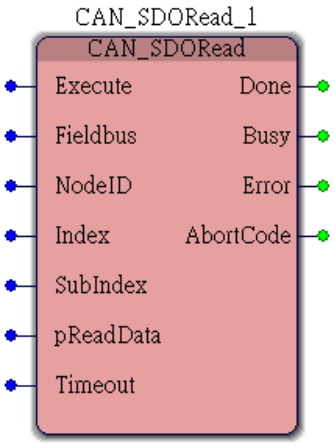
1. 设定 wData2 变量格式为 DWORD 并设定初始值(Initial value)为 16#0080。



2. 将 En05 脚位由 False 设定为 True
3. CAN_PDOWrite 功能块将 OD 码内容 0x0080 透过 EtherCAT 总线写入，其指定站号为 1、主编号 0x6040、次编号为 0 (此 OD 码为 Controlword)。
4. 此 OD 码动作执行成功时，Vaild 和 Busy 脚位变为 True，并将指定站号为 1 的伺服做 Servo Off 的动作。

6.5.7.3 CAN_SDORead

- 类型
  Function Block
- 功能描述
  进行 SDO 读取功能。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值(默认值)	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，触发此功能。
Fieldbus	UINT	0 ~ 2 (0)	总线类型。 0：EtherCAT 1：DMCNET(台达总线) 2：CANOpen
NodeID	UINT	整数 (0)	连接装置之指定站号。
Index	UINT	整数 (0)	连接装置之OD主编号。注
SubIndex	USINT	整数 (0)	连接装置之OD次编号。注
Timeout	TIME	None	保留脚位

注：关于 OD (Object Dictionary) 主编号和次编号以及数据型态，请参考该连接装置的手册。

6

■ 输出脚位

脚位名称	数据型态	设定值(默认值)	功能
Done	BOOL	True / False (False)	完成时为True；反之则为False。
Busy	BOOL	True / False (False)	成立时表示此功能块进行中。
Error	BOOL	True / False (False)	当错误发生时为True。
AbortCode	DWORD	0x00000000~ 0xFFFFFFFF (0)	错误发生时记录的错误码 错误码详细说明请参考第9章
pReadData	ANY	-	写入数值来源。根据写入的OD码对象的不同，需要设定不同的数据型态。注

注:

1. pReadData 虽然放在功能块的左边，但是此脚位为输出脚位。该 OD 码所对应的数据型态，请参考该连接装置的手册。
2. 若读取的数据型态不符合该 OD 码，则 AbortCode 会报警 0x06070013

■ 参考范例

此范例示范使用 EtherCAT 总线对连接的台达伺服，读取所指定的 OD 码，并获得欲读取的伺服装置参数数值。

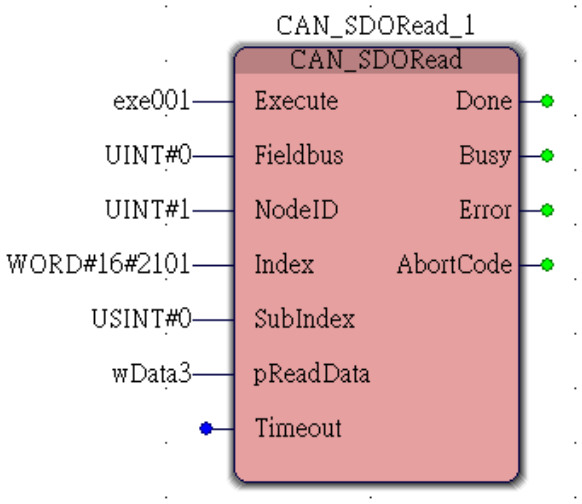


图 6.5.7.3.1 CAN_SDORead 使用参考范例

1. 设定 wData3 变量格式为 DWORD。
2. 将 exe001 脚位由 Fasle 设定为 True。
3. CAN_SDORead 功能块透过 EtherCAT 总线读取指定站号为 1、主编号 0x2101、次编号为 0 的 OD 码内容。(此 OD 码为 P1-01 参数)
4. 此 OD 码动作执行成功时，Done 脚位变为 True，wData3 读回的内容为 0x000C。

6.5.7.4 CAN_SDOWrite

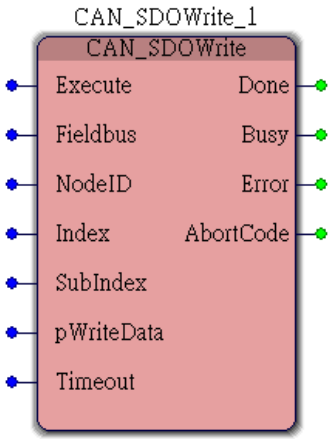
■ 类型

Function Block

■ 功能描述

进行 SDO 读取功能。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，触发此功能。
Fieldbus	UINT	0 ~ 2 (0)	总线类型。 0：EtherCAT 1：DMCNET(台达总线) 2：CANOPEN
NodeID	UINT	整数 (0)	连接装置之指定站号。
Index	UINT	整数 (0)	连接装置之OD主编号。注
SubIndex	USINT	整数 (0)	连接装置之OD次编号。注
pWriteData	ANY	-	写入数值。根据写入的OD码对象的不同，需要设定不同的数据型态。注
Timeout	TIME	None	保留

注：关于 OD (Object Dictionary) 主编号和次编号以及数据型态，请参考该连接装置的手册。

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	完成时为True；反之则为False。
Busy	BOOL	True / False (False)	成立时表示此功能块执行中。
Error	BOOL	True / False (False)	错误发生时为True。

脚位名称	数据型态	设定值 (默认值)	功能
AbortCode	DWORD	0x00000000 ~ 0xFFFFFFFF (0)	错误发生时记录的错误码 详细说明请参考第9章

6

■ 参考范例

此范例示范使用 EtherCAT 总线对连接的台达伺服，写入所指定的 OD 码以修改伺服 P1-44 参数。

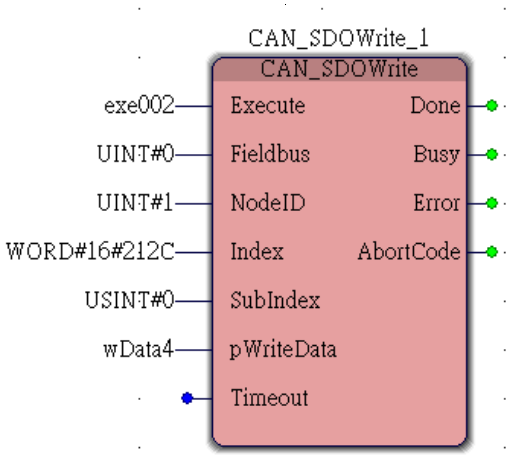
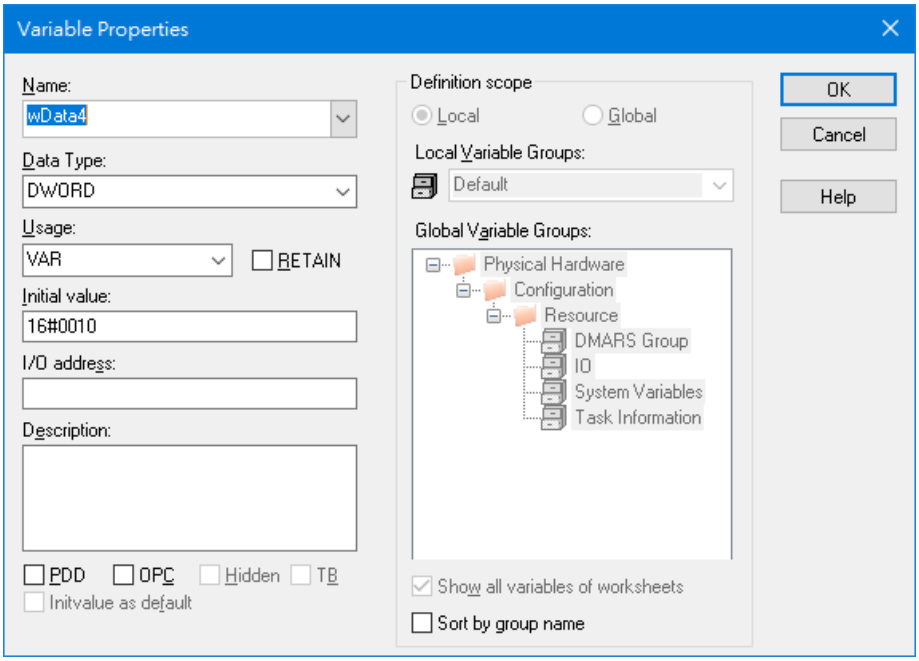


图 6.5.7.4.1 CAN_SDOWrite 使用参考范例

1. 设定 wData4 变量格式为 DWORD 并设定初始值 (Initial value)为 16#0010。



2. 将 exe002 脚位由 Fasle 设定为 True。
3. CAN_SDOWrite 功能块将 wData4 数据透过 EtherCAT 总线写入 OD 码中，其指定站号为 1、主编号为 0x212C、次编号为 0。(此 OD 码为 P1-44 参数)
4. 此 OD 码动作执行成功时，Done 变为 True，P1-44 参数内容修改为 0x0010

6.5.7.5 MB_ReadBits

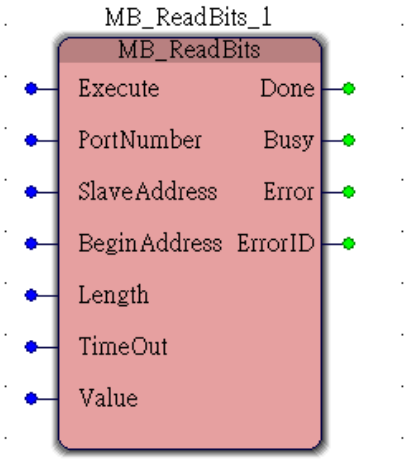
■ 类型

Function Block

■ 功能描述

使用串行端口通讯读取指定位置的一组位长度的数据串。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，触发此功能。
PortNumber	UINT	0 ~ 1 (0)	串行端口编号，目前仅支持DXMC本体上的串行端口 0：COM0 (不支援) 1：COM1
SlaveAddress	UINT	0 ~ 65535 (0)	欲读取数据之从站站号
BeginAddress	UINT	0 ~ 65535 (0)	欲读取数据之起始位置
Length	UINT	0 ~ 65535 (0)	欲读取数据之读取长度
TimeOut	UINT	0 ~ 65535 (0)	设定逾时时间；单位：ms

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	完成时为True；反之则为False。
Busy	BOOL	True / False (False)	成立时表示此功能块进行中。
Error	BOOL	True / False (False)	当错误发生时为True。
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	错误发生时记录的错误码 错误码详细说明请参考第9章

6

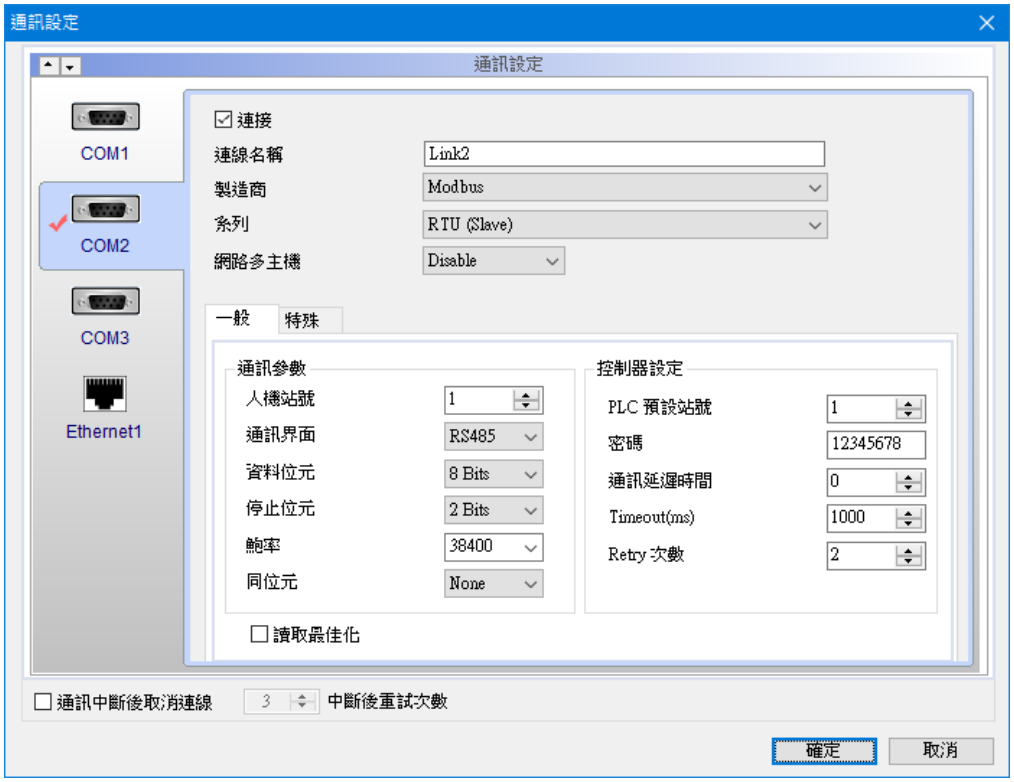
脚位名称	数据型态	设定值 (默认值)	功能
Value	ANY	-	读取数值。根据读取长度的不同，需要设定不同的数据型态。注

注：Value 虽然放在功能块的左边，但是此脚位为输出脚位。

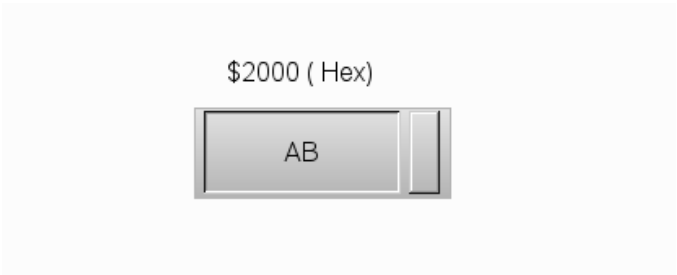
■ 参考范例

此范例示范使用 DXMC 控制器透过串行端口通讯来对台达人机 DOP-110WS 读取数据，其中 DXMC 当作主站 (Master)、人机 DOP-110WS 当作从站 (Slave)。

1. 作为从站的台达人机 DOP-110WS 通信设置如下：

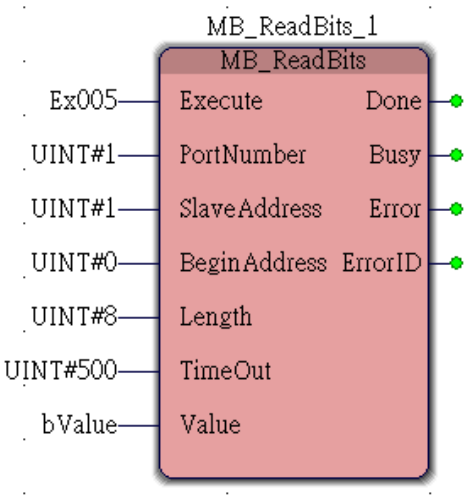


2. 台达人机 DOP-110WS 画面建立数值输入组件$2000 格式为 Hexadecimal：(台达人机的 $2000.0 ~ $2063.15 对应 Modbus 地址为 B00001 ~ B01024，详细请参考台达人机联机手册)。



3. 透过 DMARS 软件对串行端口进行通信设置。将 DXMC 设为 MODBUS 主站，其波特率为 38400、数据位为 8 bits、同位检查为 None、停止位为 2 bits，协议为 RTU。

4. MB_ReadBits 功能块设定如下图，串行端口为 COM1、从站站号为 1、读取位置为 0、读取长度为 8、Timeout 时间为 500 ms。



5. 将读取数据存放至 bValue 变量，并设定 bValue 变量格式为 WORD。
6. 将 Ex005 脚位由 False 设定为 True。
7. MB_ReadBits 功能块将读取串行端口为 COM1、从站站号为 1、读取位置为 0，读取长度为 8 bits 的内容至 bValue 变量。
8. 此功能块执行成功时，Done 脚位变为 True，bValue 读回的内容为 0x00AB。

6.5.7.6 MB_ReadWords

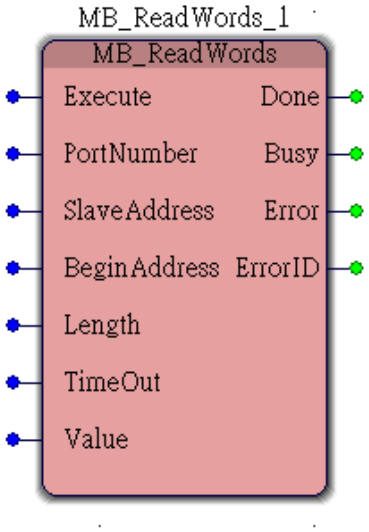
■ 类型

Function Block

■ 功能描述

使用串行端口通讯读取指定位置的一组 word 长度的数据串。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，触发此功能。
PortNumber	UINT	0 ~ 1 (0)	串行埠编号，目前仅支持DXMC本体上的序串埠 0：COM0 (不支援) 1：COM1
SlaveAddress	UINT	0 ~ 65535 (0)	欲读取数据之从站站号
BeginAddress	UINT	0 ~ 65535 (0)	欲读取数据之起始位置
Length	UINT	0 ~ 65535 (0)	欲读取数据之读取长度
TimeOut	UINT	0 ~ 65535 (0)	设定逾时时间；单位：ms

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	完成时为True；反之则为False。
Busy	BOOL	True / False (False)	成立时表示此功能块进行中。
Error	BOOL	True / False (False)	错误发生时为True。

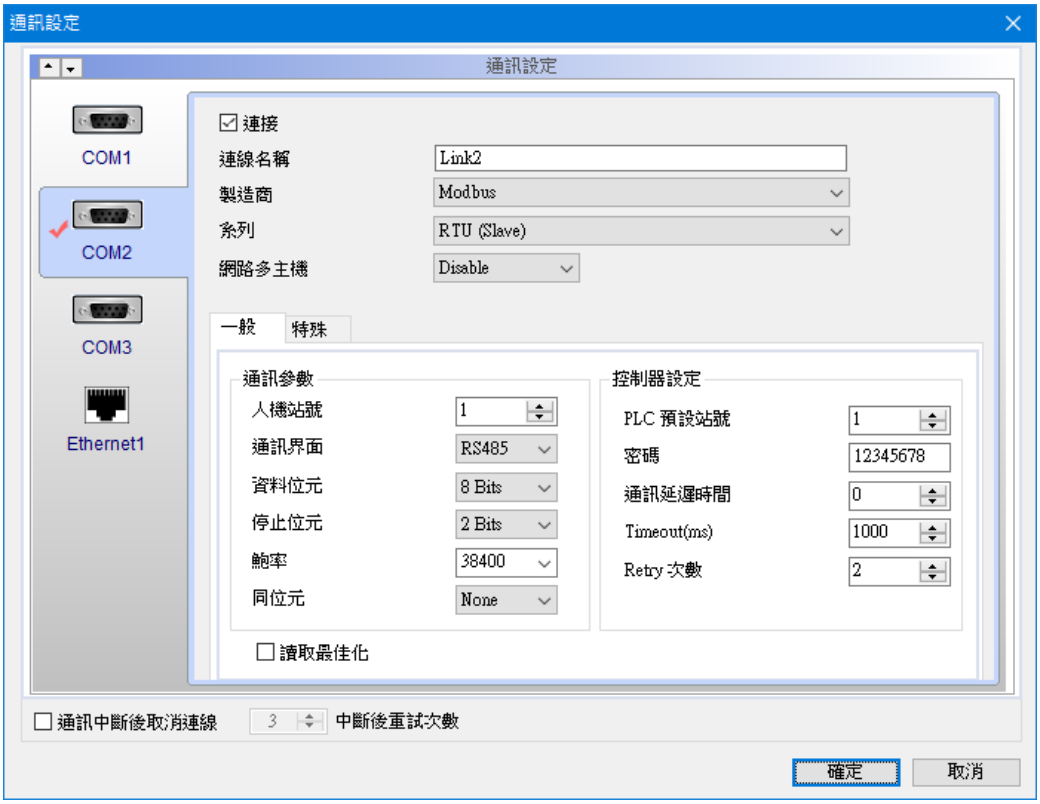
脚位名称	数据型态	设定值 (默认值)	功能
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	错误发生时记录的错误码 详细说明请参考第9章
Value	ANY	-	读取数值。根据读取长度的不同，需要设定不同的数据型态。注

注：Value 虽然放在功能块的左边，但是此脚位为输出脚位。

■ 参考范例

此范例说明如何使用 DXMC 控制器透过串行埠读取台达人机 DOP-110WS 数据，其中 DXMC 控制器作主站 (Master)、人机 DOP-110WS 为从站 (Slave)。

1. 作为从站的台达人机 DOP-110WS 通信设置如下：

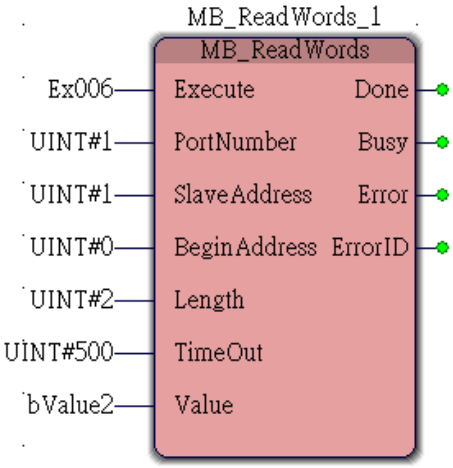


2. 台达人机 DOP-110WS 画面建立数值输入组件$0 格式为 Hexadecimal：(台达人机的$0 ~ $1023 对应 Modbus 地址为 W40001 ~ W41024，详细请参考台达人机联机手册)。

6

3. 透过 DMARS 软件来对串行端口进行通信设置。将 DXMC 设为 MODBUS 主站，其波特率为 38400，数据位为 8 bits，同位检查为 None，停止位为 2 bits，协议为 RTU。

4. MB_ReadBits 功能块设定如下图，串行埠为 COM1、从站站号为 1、读取位置为 0、读取长度为 8、Timeout 时间为 500 ms。



5. 读取数据存放至 bValue2 变量，并设定 bValue2 变量格式为 DWORD。
6. 将 Ex006 脚位由 False 设定为 True。
7. MB_ReadBits 功能块将读取串行端口为 COM1、从站站号为 1、读取位置 0、读取长度为 2 words 的内容至 bValue2 变量。
8. 此功能块执行成功时，Done 脚位变为 True，bValue2 读回的内容为 0x89ABCDEF

6.5.7.7 MB_SerialPort_Set

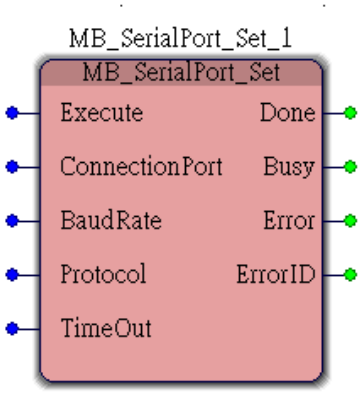
■ 类型

Function Block

■ 功能描述

控制器串行端口通讯参数更改。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，触发此功能。
Connection Port	UINT	0 ~ 1 (0)	串行埠编号 0：COM0 (不支援) 1：COM1
BaudRate	UINT	0 ~ 5 (0)	波特率编号 0：4800 1：9600 2：19200 3：38400 4：57600 5：115200
Protocol	UINT	0 ~ 8 (0)	通讯协议编号 0：7,N,2 (MODBUS，ASCII) 1：7,E,1 (MODBUS，ASCII) 2：7,O,1 (MODBUS，ASCII) 3：8,N,2 (MODBUS，ASCII) 4：8,E,1 (MODBUS，ASCII) 5：8,O,1 (MODBUS，ASCII) 6：8,N,2 (MODBUS，RTU) 7：8,E,1 (MODBUS，RTU) 8：8,O,1 (MODBUS，RTU)

脚位名称	数据型态	设定值 (默认值)	功能
TimeOut	UINT	0 ~ 65535 (0)	设定逾时时间；单位：ms

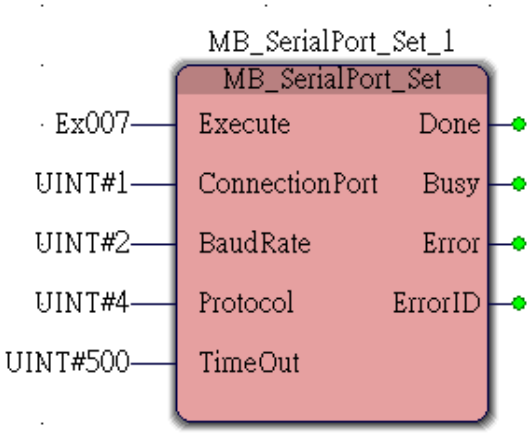
6

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	完成时为True；反之则为False。
Busy	BOOL	True / False (False)	成立时表示此功能块进行中。
Error	BOOL	True / False (False)	当错误发生时为True。
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码详细说明请参考第9章

■ 参考范例

此范例示范透过功能块来变更 DXMC 控制器中串行埠通信设置。



1. 将 Ex007 脚位由 Fasle 设定为 True。
2. MB_SerialPort_Set 功能块将串行端口 COM1 的通信设置，其变更波特率为 19200、数据位为 8 bits、同位检查为 Even、停止位为 1 bit、协议为 ASCII。
3. 此功能块执行成功时，Done 脚位变为 True。

6.5.7.8 MB_TCP_Connect

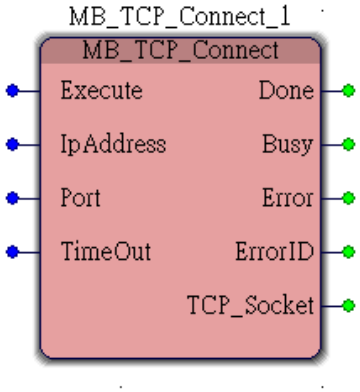
■ 类型

Function Block

■ 功能描述

藉由用户输入的 IP 建立一个 TCP Socket 通讯。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，触发此功能。
IpAddress	STRING	‘(0~255).(0~255).(0~255).(0~255)”(‘)	字符串型态的IP地址。 共有四段，每段IP地址的数字需小于255。
Port	UINT	0 ~ 65535 (0)	Modbus TCP数据交换的编号 (DXMC预设Port为502)
TimeOut	UINT	0 ~ 65535 (0)	设定逾时时间；单位：ms

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	完成时为True；反之则为False。
Busy	BOOL	True / False (False)	成立时表示此功能块进行中。
Error	BOOL	True / False (False)	错误发生时为True。
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	错误发生时记录的错误码 详细说明请参考第9章
TCP_Socket	UINT	17 ~ 22 (0)	回传TCP Socket 编号注

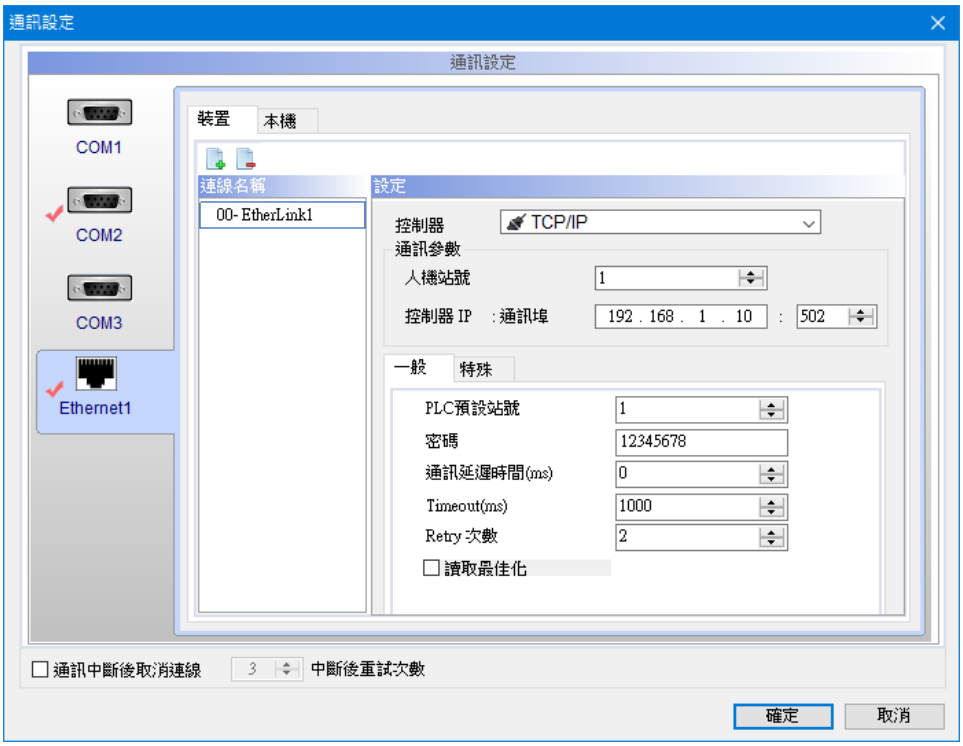
注：DXCM 最大支持的 TPC 联机数为 6 组，Socket 编号从 17 开始。

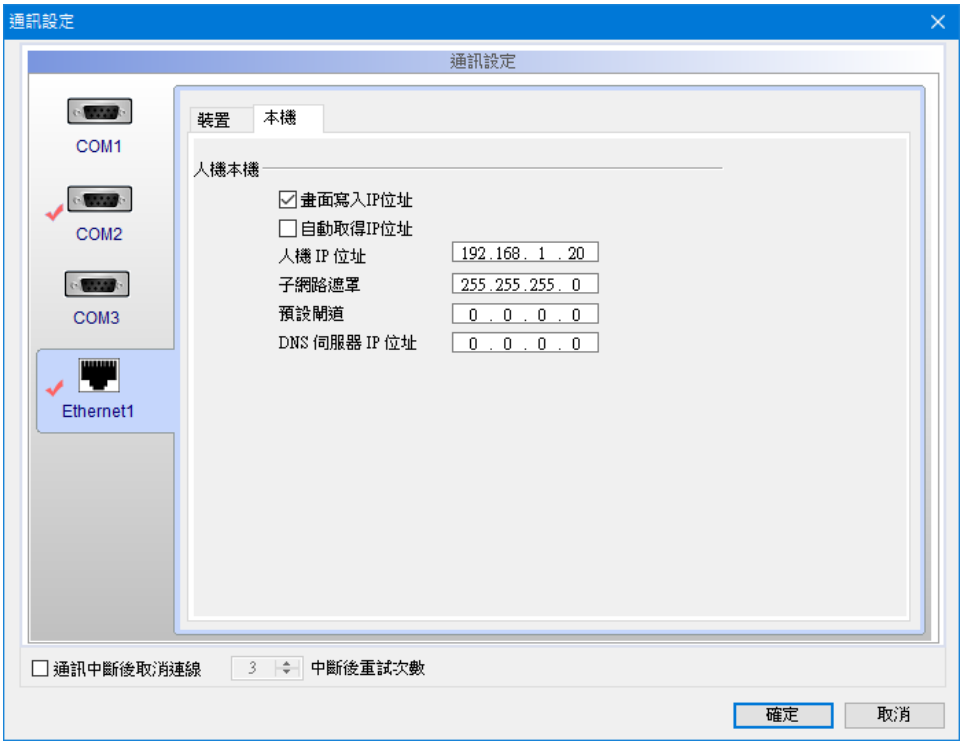
6

■ 参考范例

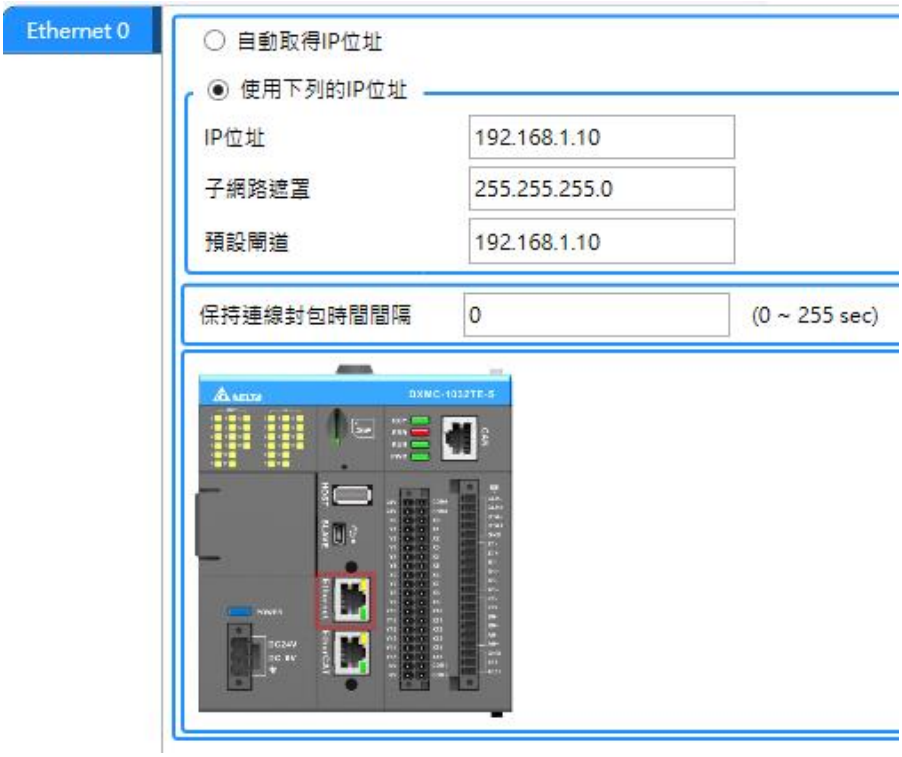
此范例说明如何使用此功能块建立一组 TCP Socket，透过 DXMC 控制器中的网络端口连接台达人机 DOP-110WS，其中 DXMC 控制器 IP 地址为 192.168.1.10、人机 DOP-110WS IP 地址为 192.168.1.20。

1. 台达人机 DOP-110WS 通信设置如下：

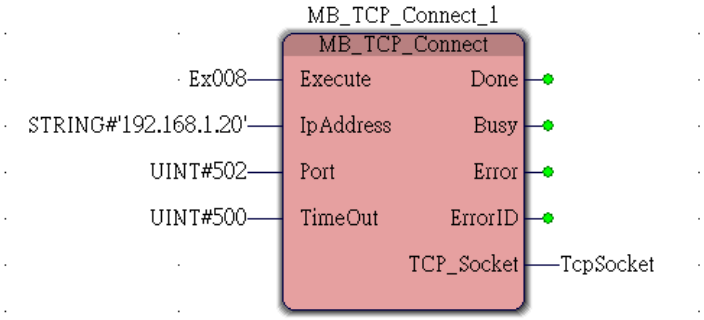




2. 透过 DMARS 设定 DXMC IP 地址为 192.168.1.10



3. MB_TCP_Connect 功能块设定如下：



4. 将 Ex008 脚位由 Fasle 设定为 True。
5. MB_TCP_Connect 功能块根据设定对 IP 地址为 192.168.1..20、Port 为 502 的装置，建立其 Socket 窗口，并将产生的 Socket 编号储存于变量 TcpSocket。
6. 此功能块执行成功时，Done 脚位变为 True，变量 TcpSocket 内容为 17。

注：台达人机的 Socket 建立之后，如果 16 秒之内 DXMC 没有与人机进行任何读写动作，该 Socket 会被台达人机主动释放(断开)，此时 DXMC 也必须使用 MB_TCP_DisConnect 功能块来释放该 Socket，DXMC 才能再次使用 Socket。

6

6.5.7.9 MB_TCP_DisConnect

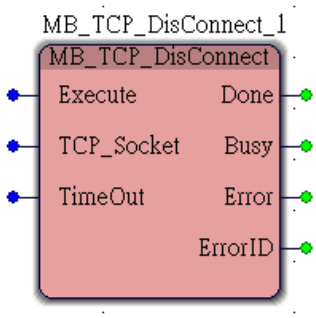
■ 类型

Function Block

■ 功能描述

解除已建立的 TCP Socket 通讯。

■ 图形表示



■ 输入脚位

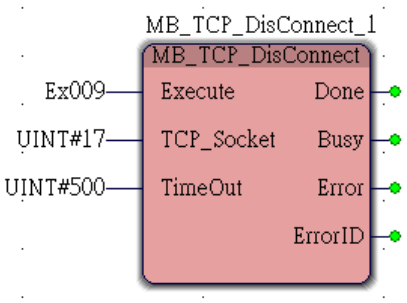
脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，触发此功能。
TCP_Socket	UINT	17 ~ 22 (0)	解除此Socket所建立的联机
TimeOut	UINT	0 ~ 65535 (0)	设定逾时时间；单位：ms

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	完成时为True；反之则为False。
Busy	BOOL	True / False (False)	成立时表示此功能块进行中。
Error	BOOL	True / False (False)	指令错误发生时为True。
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码详细说明请参考第9章

■ 参考范例

此范例利用此功能块解除已建立联机的 TCP Socket 通讯，使其通讯中断。



1. 将 Ex009 脚位由 False 设定为 True。
2. MB_TCP_DisConnect 功能块解除 TCP Socket 编号为 17 的联机。
3. 此功能块执行成功时，Done 脚位变为 True。

6

6.5.7.10 MB_TCP_ReadBits

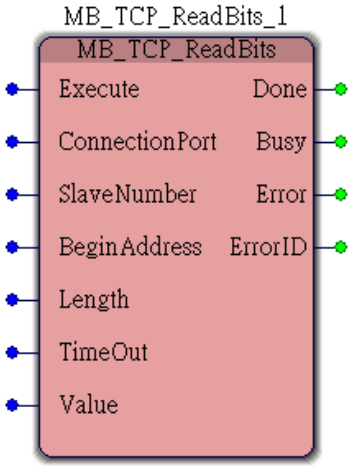
■ 类型

Function Block

■ 功能描述

使用 Socket 通讯读取指定位置的位值 (在使用此功能块之前，必须使用 MC_TCP_Connect 功能块来建立 TCP Socket 通讯)。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，触发此功能。
Connection Port	UINT	17 ~ 22 (0)	设定已经建立的Socket编号
SlaveNumber	UINT	0 ~ 65535 (0)	欲读取数据之从站站号
BeginAddress	UINT	0 ~ 65535 (0)	欲读取数据之起始位置
Length	UINT	0 ~ 65535 (0)	欲读取数据之读取长度
TimeOut	UINT	0 ~ 65535 (0)	设定逾时时间；单位：ms

■ 输出脚位

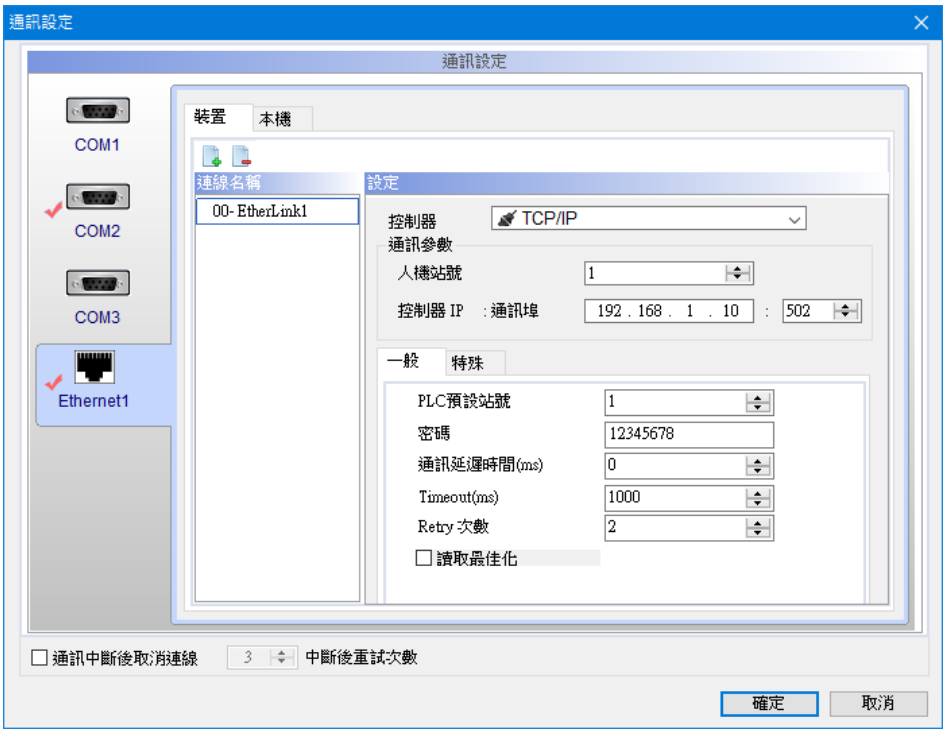
脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	完成时为True；反之则为False。
Busy	BOOL	True / False (False)	成立时表示此功能块进行中。
Error	BOOL	True / False (False)	当错误发生时为True。
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码详细说明请参考第9章
Value	ANY	-	读取数值。根据读取长度的不同，需要设定不同的数据型态。注

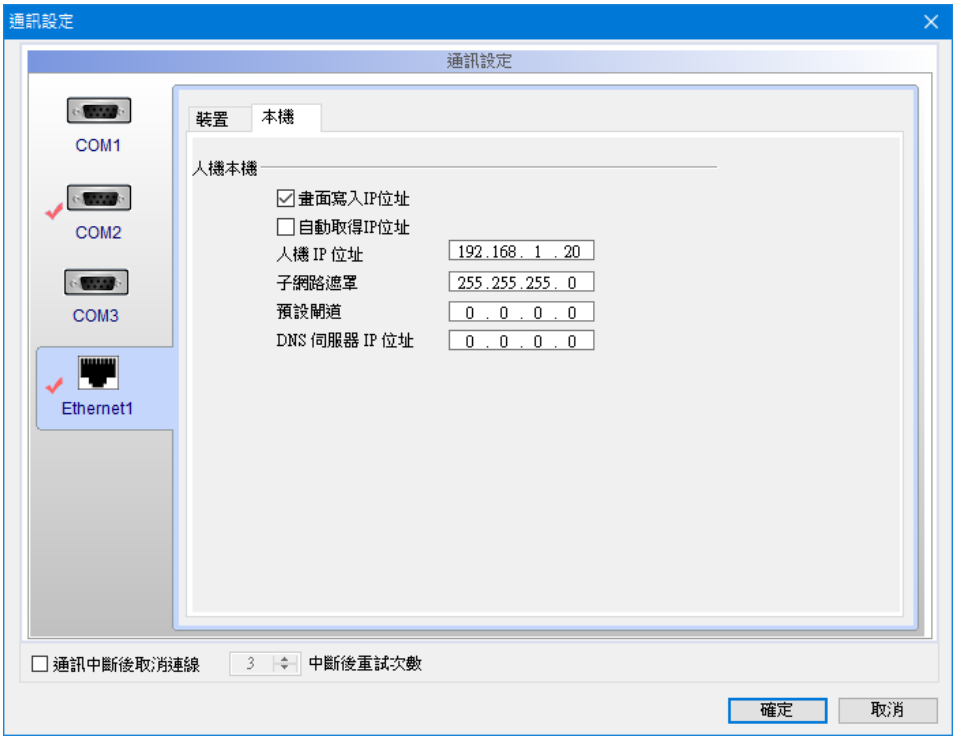
注：Value 虽然放在功能块的左边，但是此脚位为输出脚位。

■ 参考范例

此范例说明 DXMC 控制器如何透过利用一组 TCP Socket，来读取台达人机 DOP-110WS 装置内数据，其中 DXMC 控制器 IP 地址为 192.168.1.10、人机 DOP-110WS IP 地址为 192.168.1.20。

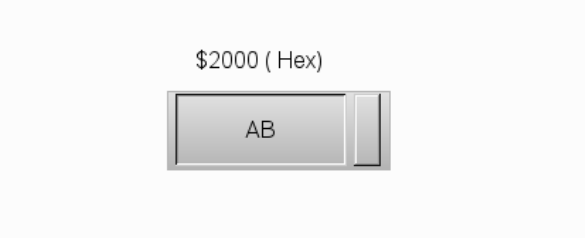
1. 台达人机 DOP-110WS 通信设置如下：



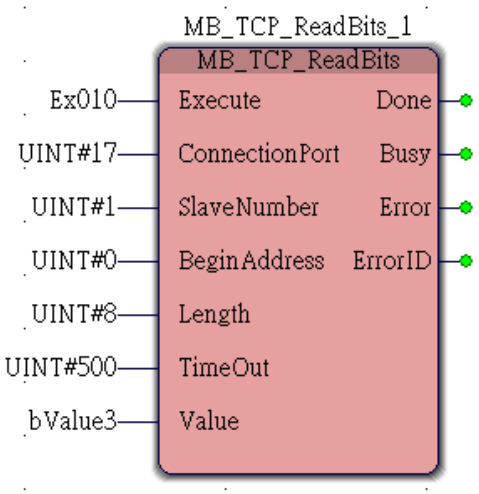


6

2. 台达人机 DOP-110WS 画面建立数值输入组件$2000 格式为 Hexadecimal：(台达人机的$2000.0 ~ $2063.15 对应 Modbus 地址为 B00001 ~ B01024，详细请参考台达人机联机手册)



3. 先使用 MB_TCP_Connect 功能块来建立 Socket，详细请参考章节 6.5.7.8。
4. MB_TCP_ReadBits 功能块设定如下：



5. 读取数据存放至 bValue3 变量，并设定 bValue3 变量格式为 WORD。
6. 将 Ex010 脚位由 Fasle 设定为 True。
7. MB_TCP_ReadBits 功能块将读取 Socket 编号 17、从站站号为 1、读取位置 0、读取长度为 8 bits 的内容至 bValue3 变量。
8. 此功能块执行成功时，Done 脚位变为 True，bValue3 读回的内容为 0x00AB。

6.5.7.11 MB_TCP_ReadWords

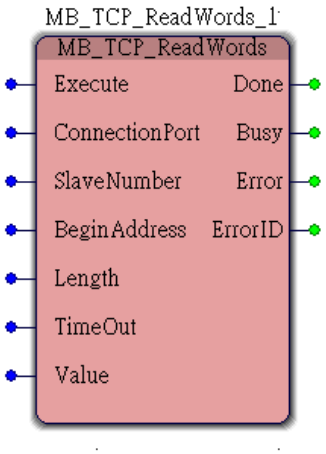
■ 类型

Function Block

■ 功能描述

使用 Socket 通讯读取指定位置的一组 word 长度的数据串。 (在使用此功能块之前，必须使用 MC_TCP_Connect 功能块来建立 TCP Socket 通讯)。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，触发此功能。
Connection Port	UINT	17 ~ 22 (0)	设定已经建立的Socket编号
SlaveNumber	UINT	0 ~ 65535 (0)	欲读取数据之从站站号
BeginAddress	UINT	0 ~ 65535 (0)	欲读取数据之起始位置
Length	UINT	0 ~ 65535 (0)	欲读取数据之读取长度
TimeOut	UINT	0 ~ 65535 (0)	设定逾时时间；单位：ms

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	完成时为True；反之则为False。
Busy	BOOL	True / False (False)	成立时表示此功能块进行中。
Error	BOOL	True / False (False)	指令当错误发生时为True。
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码详细说明请参考第9章

6

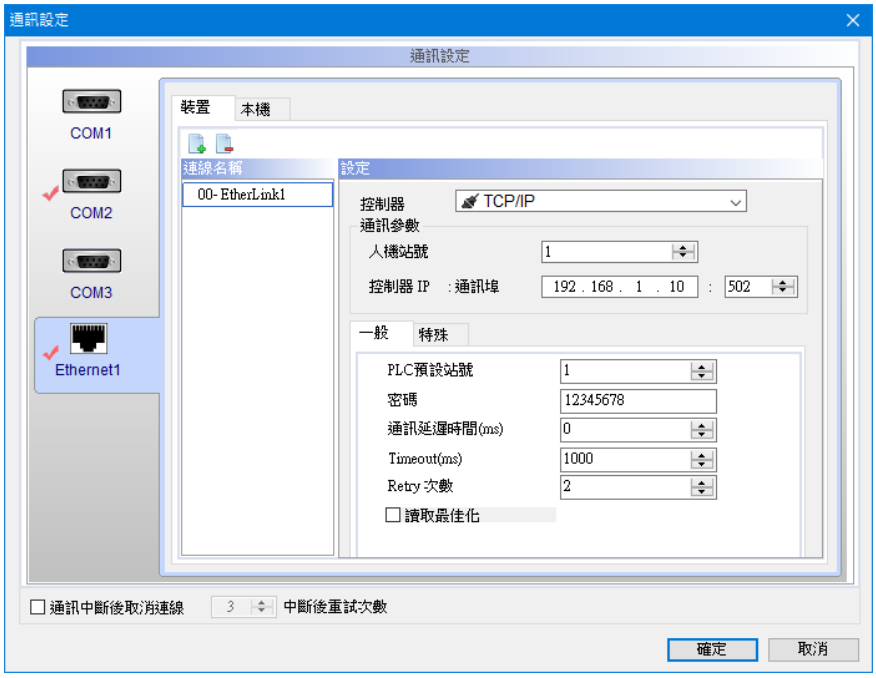
脚位名称	数据型态	设定值 (默认值)	功能
Value	ANY	-	读取数值。根据读取长度的不同，需要设定不同的数据型态。注

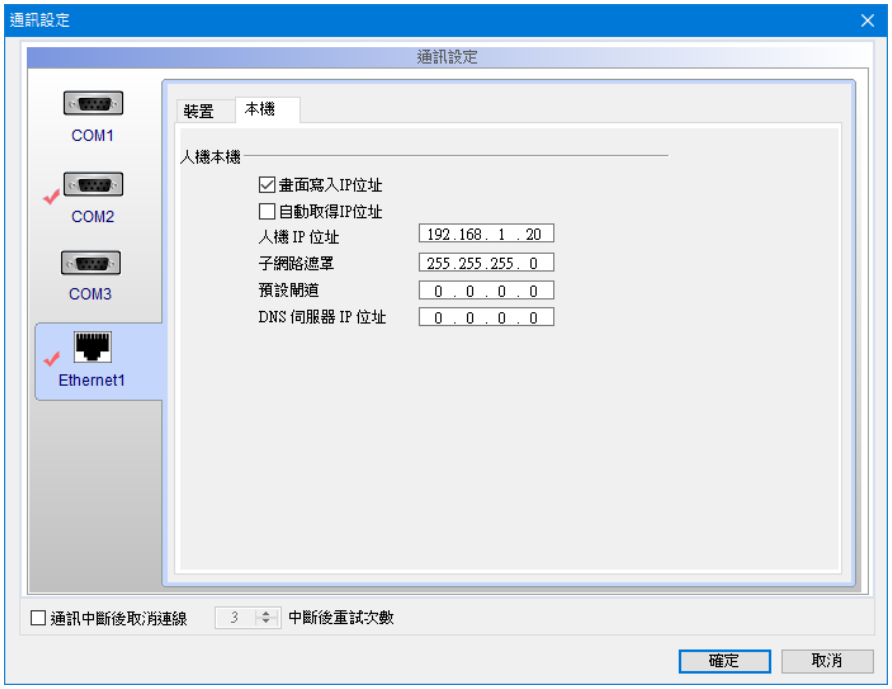
注：Value 虽然放在功能块的左边，但是此脚位为输出脚位。

■ 参考范例

此范例说明 DXMC 控制器如何透过利用 TCP Socket 通讯，来读取台达人机 DOP-110WS 装置内数据，其中 DXMC 控制器 IP 地址为 192.168.1.10、人机 DOP-110WS IP 地址为 192.168.1.20。

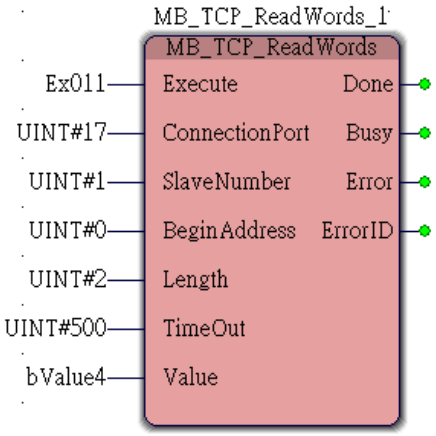
1. 台达人机 DOP-110WS 通信设置如下：





2. 台达人机 DOP-110WS 画面建立数值输入组件 $0 格式为 Hexadecimal：(台达人机的 $0 ~ $1023 对应 Modbus 地址为 W40001 ~ W41024，详细请参考台达人机联机手册)

3. 先使用 MB_TCP_Connect 功能块来建立 Socket，详细请参考章节 6.5.7.8。
4. MB_TCP_ReadWords 功能块设定如下：



5. 读取数据存放至 bValue4 变量，并设定 bValue4 变量格式为 DWORD。
6. 将 Ex011 脚位由 Fasle 设定为 True。
7. MB_TCP_ReadWords 功能块将读取 Socket 编号 17、从站站号为 1、读取位置为 0、读取长度为 8 bits 的内容至 bValue4 变量。
8. 此功能块执行成功时，Done 脚位变为 True，bValue4 读回的内容为 0x89ABCDEF。

6

6.5.7.12 MB_TCP_WriteBit

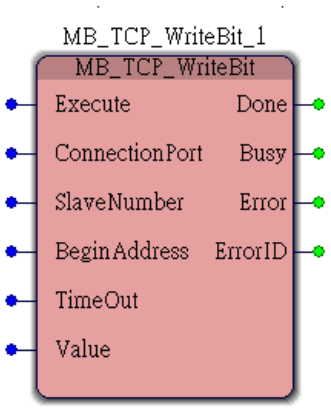
■ 类型

Function Block

■ 功能描述

使用 Socket 通讯写入一个位值至指定位置（在使用此功能块之前，必须先使用 MC_TCP_Connect 功能块来建立 TCP Socket 通讯）。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值(默认值)	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，触发此功能。
Connection Port	UINT	17 ~ 22 (0)	设定已经建立的Socket编号
SlaveNumber	UINT	0 ~ 65535 (0)	欲写入资料之从站站号
BeginAddress	UINT	0 ~ 65535 (0)	欲写入数据之起始位置
TimeOut	UINT	0 ~ 65535 (0)	设定逾时时间；单位：ms
Value	BOOL	True / False (False)	写入位值。

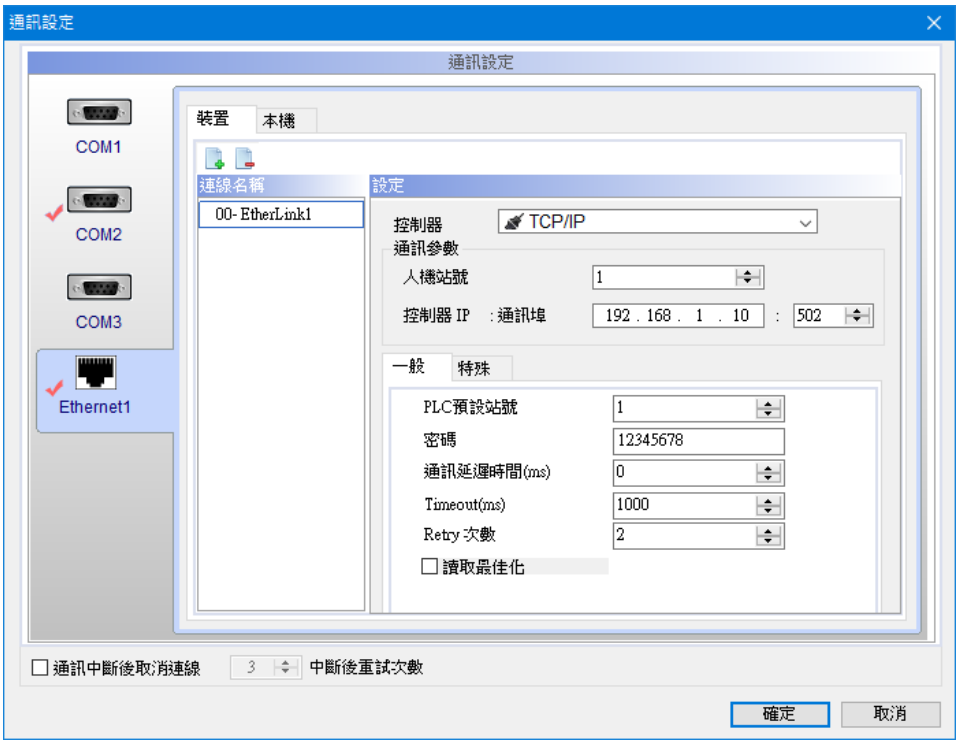
■ 输出脚位

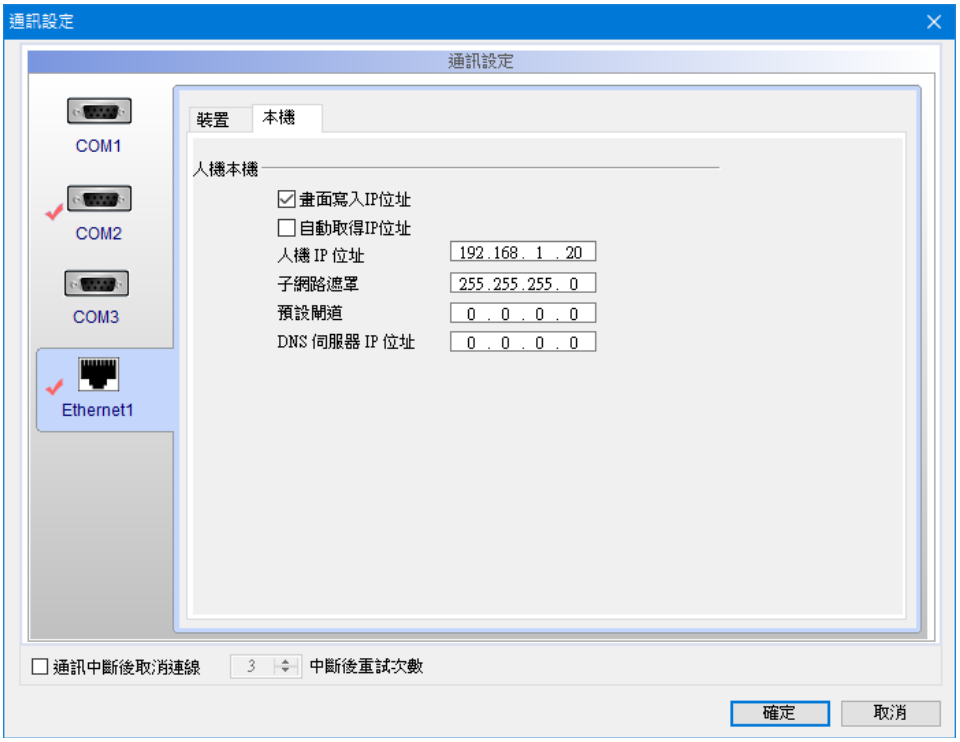
脚位名称	数据型态	设定值(默认值)	功能
Done	BOOL	True / False (False)	完成时为 True;反之则为 False。
Busy	BOOL	True / False (False)	成立时表示此功能块进行中。
Error	BOOL	True / False (False)	当错误发生时为True。
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	错误发生时记录的错误码 错误码详细说明请参考第9章

■ 参考范例

此范例说明 DXMC 控制器如何透过 TCP Socket 通讯写入一位值至台达人机 DOP-110WS 内存，其中 DXMC 控制器 IP 地址为 192.168.1.10、人机 DOP-110WS IP 地址为 192.168.1.20。

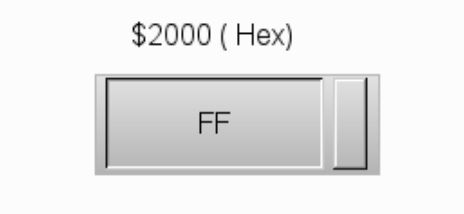
1. 台达人机 DOP-110WS 通信设置如下：



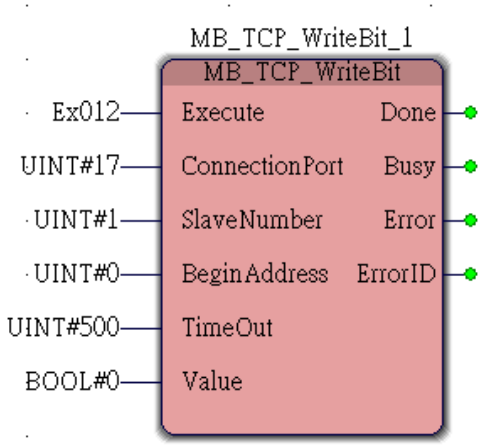


6

2. 台达人机 DOP-110WS 画面建立数值输入组件$2000 格式为 Hexadecimal：(台达人机的$2000.0 ~ $2063.15 对应 Modbus 地址为 B00001 ~ B01024，详细请参考台达人机联机手册)。



3. 先使用 MB_TCP_Connect 功能块来建立 Socket，详细请参考章节 6.5.7.8。
4. MB_TCP_WriteBit 功能块设定如下：



5. 写入位值 Value 设为 0。
6. 将 Ex012 脚位由 False 设定为 True。
7. MB_TCP_WriteBit 功能块将位值 0 写入至 Socket 编号 17，从站站号为 1、起始位置为 0。
8. 此功能块执行成功时，Done 脚位变为 True，台达人机 $2000 的内容变更为 0x00FE。

6.5.7.13 MB_TCP_WriteBits

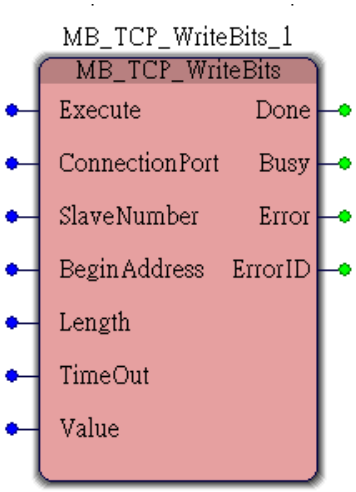
■ 类型

Function Block

■ 功能描述

使用 Socket 通讯写入一组位型态的数据串至指定位置 (在使用此功能块之前，必须先使用 MC_TCP_Connect 功能块来建立 TCP Socket 通讯)。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，触发此功能。
Connection Port	UINT	17 ~ 22 (0)	设定已经建立的Socket编号
SlaveNumber	UINT	0 ~ 65535 (0)	欲写入之从站站号
BeginAddress	UINT	0 ~ 65535 (0)	欲写入数据之起始位置
Length	UINT	0 ~ 65535 (0)	欲写入数据之读取长度
TimeOut	UINT	0 ~ 65535 (0)	设定逾时时间；单位：ms
Value	ANY	-	写入数值。根据写入长度的不同，需要设定不同的数据型态。

■ 输出脚位

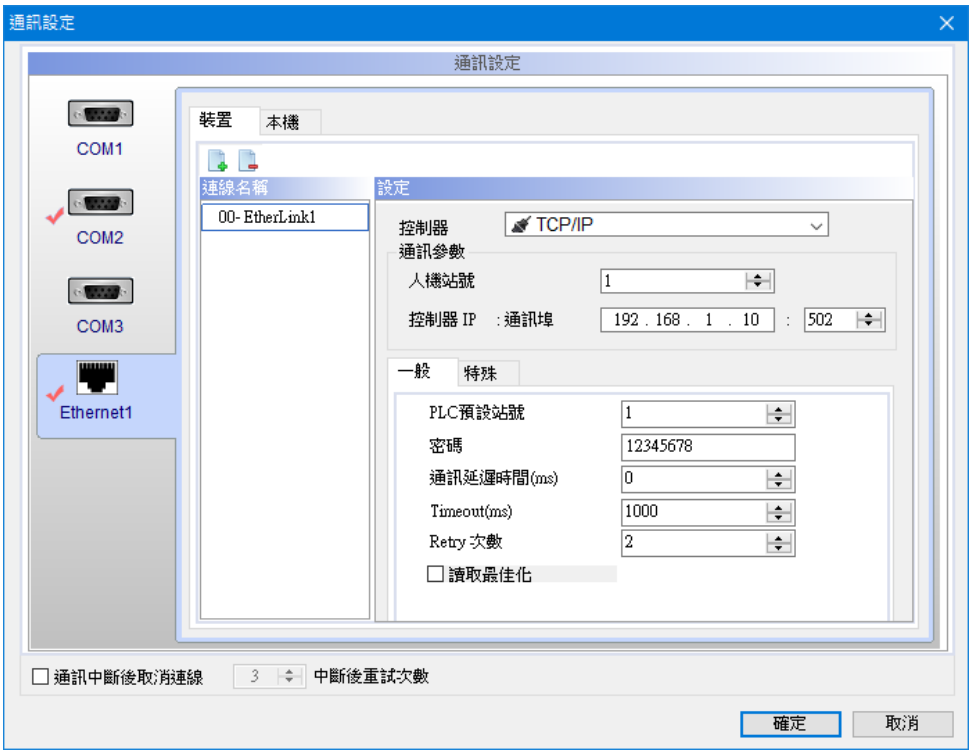
脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	完成时为True；反之则为False。
Busy	BOOL	True / False (False)	成立时表示此功能块进行中。
Error	BOOL	True / False (False)	当错误发生时为True。
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	错误发生时记录的错误码 错误码详细说明请参考第9章

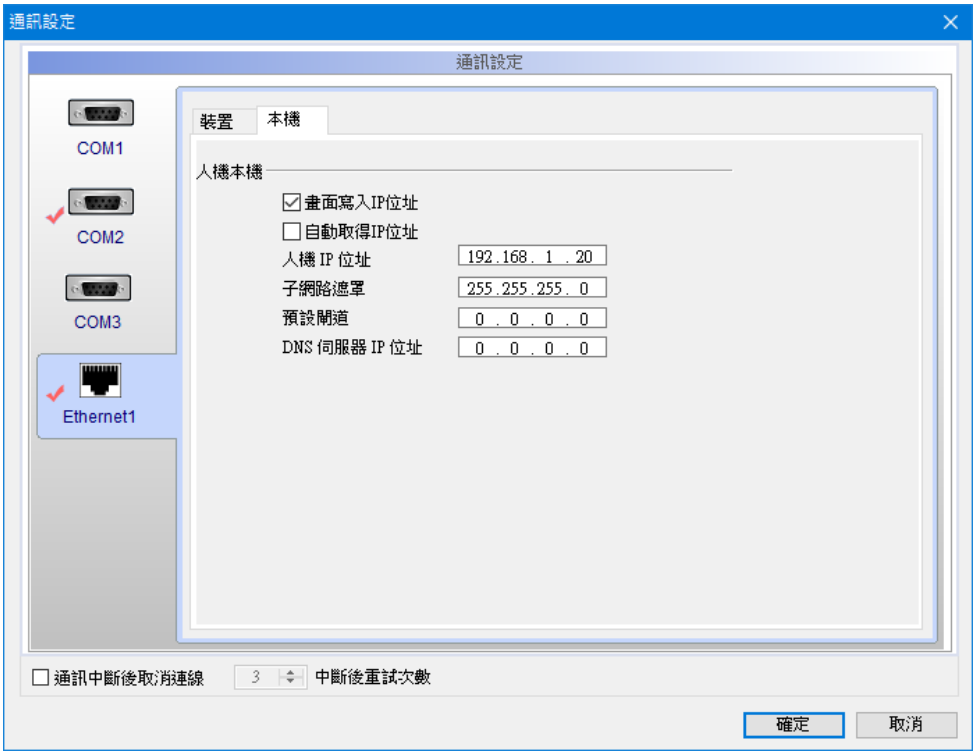
6

■ 参考范例

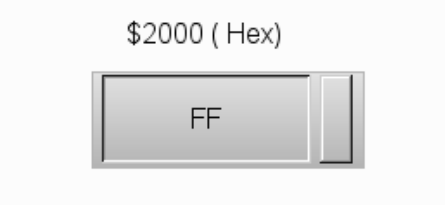
此范例说明 DXMC 控制器如何透过 TCP Socket 通讯来写入一串位值数据至台达人机 DOP-110WS，其中 DXMC IP 地址为 192.168.1.10、人机 DOP-110WS IP 地址为 192.168.1.20。

1. 台达人机 DOP-110WS 通信设置如下：

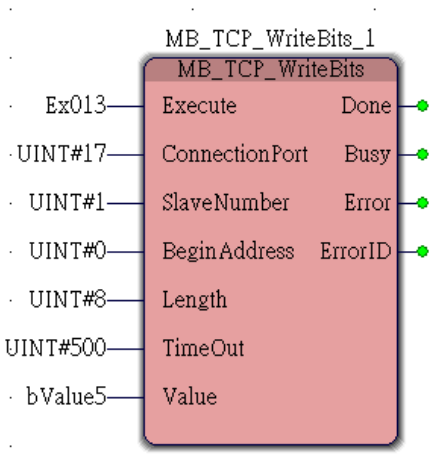




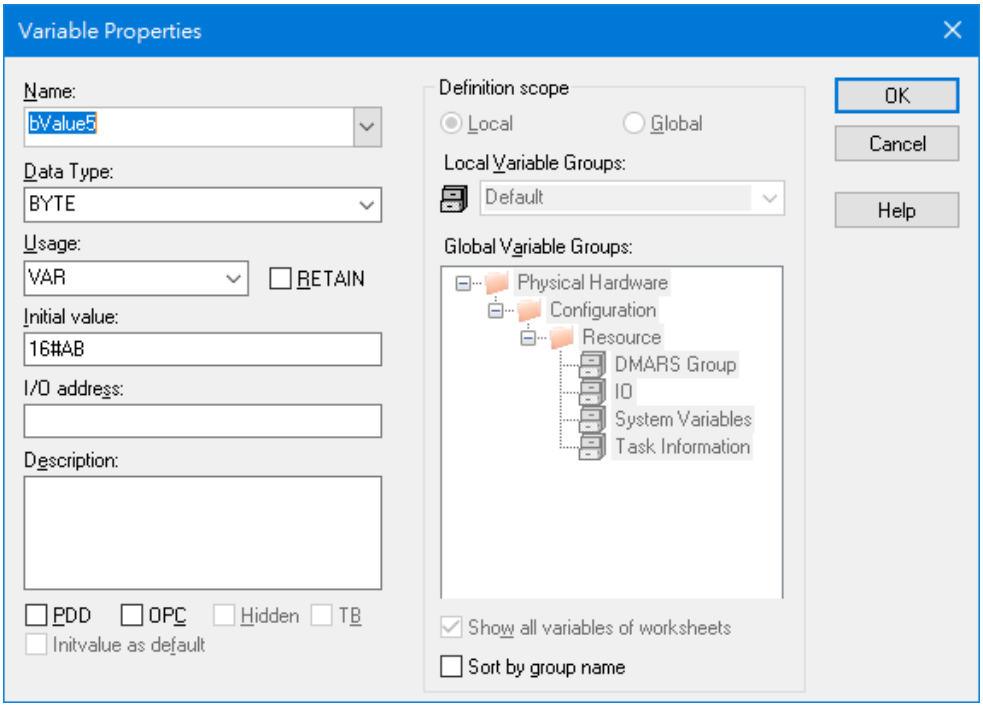
2. 台达人机 DOP-110WS 画面建立数值输入组件$2000 格式为 Hexadecimal：(台达人机的 $2000.0 ~ $2063.15 对应 Modbus 地址为 B00001 ~ B01024，详细请参考台达人机联机手册)



3. 先使用 MB_TCP_Connect 功能块来建立 Socket，详细请参考章节 6.5.7.8。
4. MB_TCP_WriteBit 功能块设定如下：



5. 写入位值存放于 bValue5 变量，并设定 bValue5 变量格式为 BYTE，并设定初始值设为 0x00AB。



6. 将 Ex013 脚位由 False 设定为 True。
7. MB_TCP_WriteBits 功能块将 bValue5 内容 0x00AB 写入至 Socket 编号 17、从站站号为 1、起始位置为 0、写入长度为 8 bits 的位置。
8. 此功能块执行成功时，Done 脚位变为 True，台达人机$2000 的内容变更为 0x00AB

6.5.7.14 MB_TCP_WriteWord

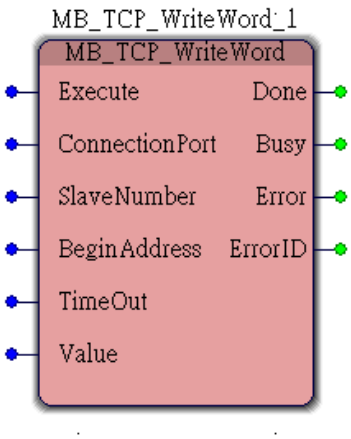
■ 类型

Function Block

■ 功能描述

使用 Socket 通讯写入一个 word 长度的数据至指定位置(在使用此功能块之前，必须先使用 MC_TCP_Connect 功能块来建立 TCP Socket 通讯)

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，触发此功能。
Connection Port	UINT	17 ~ 22 (0)	设定已经建立的Socket编号
SlaveNumber	UINT	0 ~ 65535 (0)	欲写入资料之从站站号
BeginAddress	UINT	0 ~ 65535 (0)	欲写入数据之起始位置
TimeOut	UINT	0 ~ 65535 (0)	设定逾时时间；单位：ms
Value	UINT	0 ~ 65535 (0)	写入字组 (word)。

■ 输出脚位

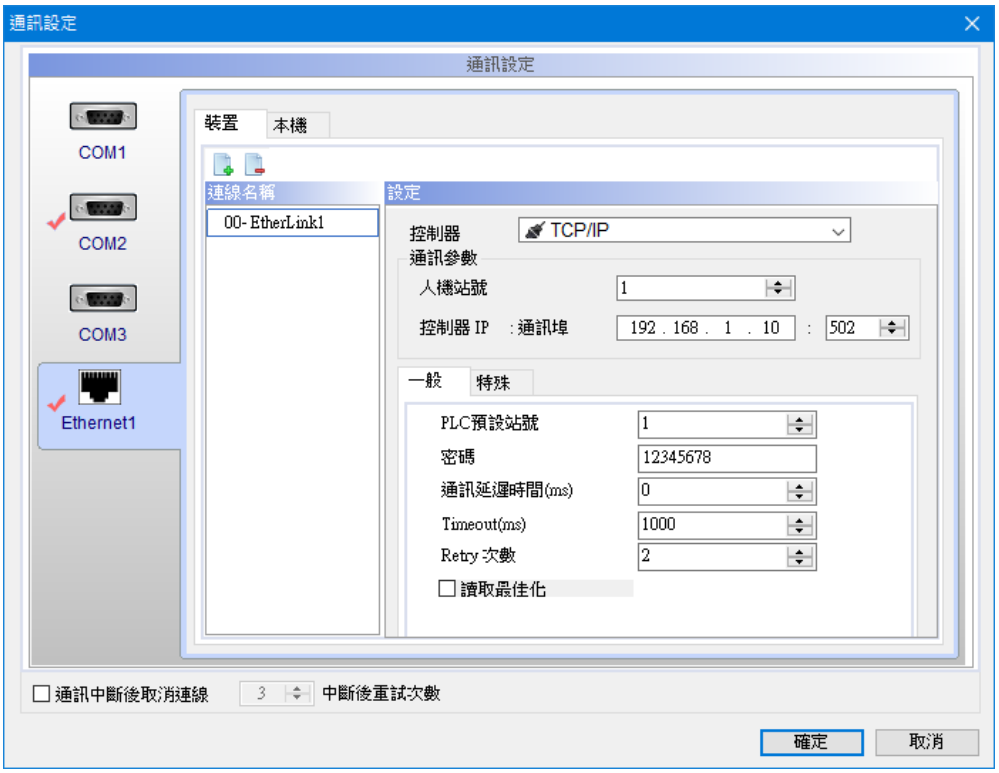
脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	完成时为True；反之则为False。
Busy	BOOL	True / False (False)	成立时表示此功能块进行中。
Error	BOOL	True / False (False)	当错误发生时为True。
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	当错误发生时记录的错误码 错误码详细说明请参考第9章

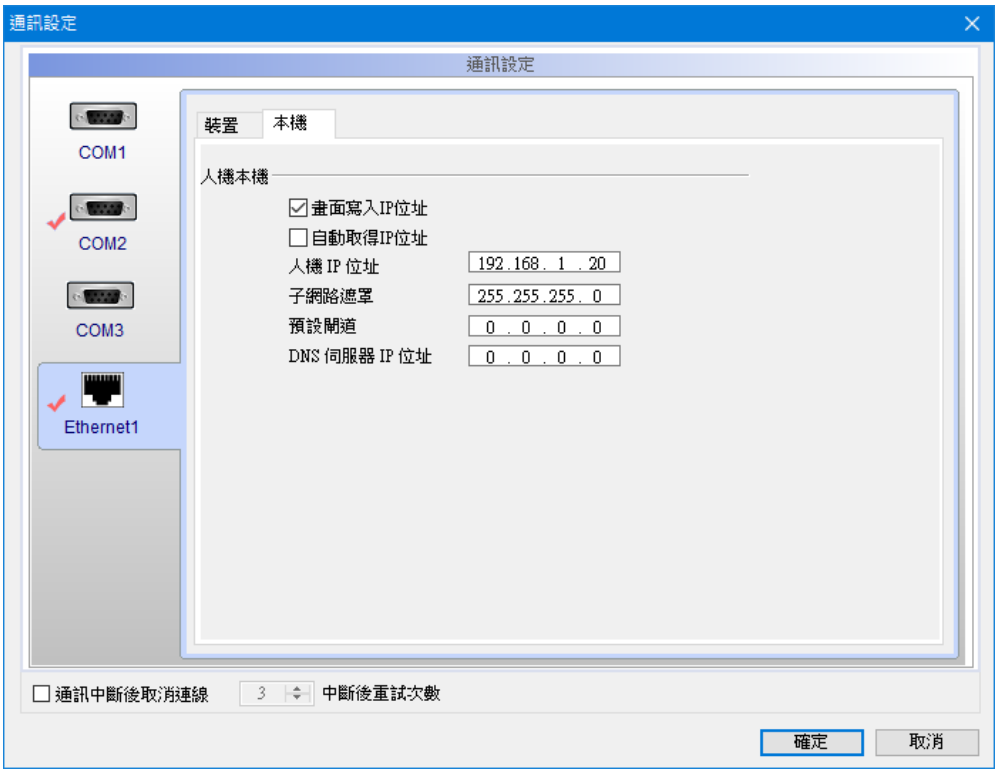
6

■ 参考范例

此范例说明 DXMC 控制器如何透过利用 TCP Socket 通讯，来写入位值至台达人机 DOP-110WS 装置内存，其中 DXMC IP 地址为 192.168.1.10、人机 DOP-110WS IP 地址为 192.168.1.20。

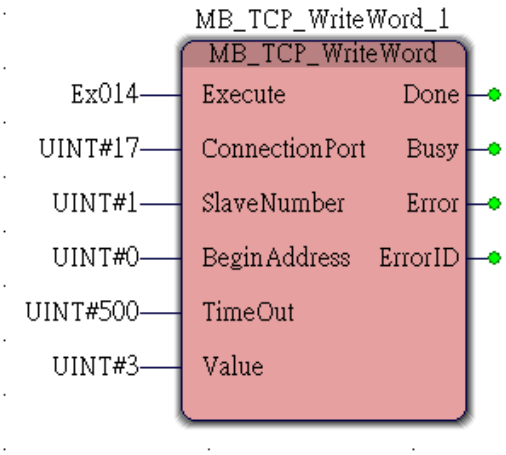
1. 台达人机 DOP-110WS 通信设置如下：





2. 台达人机 DOP-110WS 画面建立数值输入组件 $0 格式为 Hexadecimal：(台达人机的 $0 ~ $1023 对应 Modbus 地址为 W40001 ~ W41024，详细请参考台达人机联机手册)

3. 先使用 MB_TCP_Connect 功能块来建立 Socket，详细请参考章节 6.5.7.8。
4. MB_TCP_WriteWord 功能块设定如下：



5. 写入字组设为 3。
6. 将 Ex014 脚位由 Fasle 设定为 True。
7. MB_TCP_WriteWord 功能块将字组 3 写入至 Socket 编号 17，其从站站号为 1、起始位置为 0。
8. 此功能块执行成功时，Done 脚位变为 True，台达人机 $0 的内容变更为 0x89AB0003。

6

6.5.7.15 MB_TCP_WriteWords

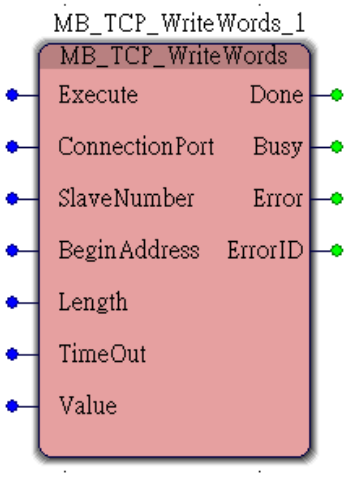
■ 类型

Function Block

■ 功能描述

使用 Socket 通讯写入一组 word 长度的数据串至指定位置 (在使用此功能块之前，必须先使用 MC_TCP_Connect 功能块来建立 TCP Socket 通讯)。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，触发此功能。
Connection Port	UINT	17 ~ 22 (0)	设定已经建立的Socket编号
SlaveNumber	UINT	0 ~ 65535 (0)	欲写入资料之从站站号
BeginAddress	UINT	0 ~ 65535 (0)	欲写入数据之起始位置
Length	UINT	0 ~ 65535 (0)	欲写入数据之读取长度
TimeOut	UINT	0 ~ 65535 (0)	设定逾时时间；单位：ms
Value	ANY	-	写入数值。根据写入长度的不同，需要设定不同的数据型态。

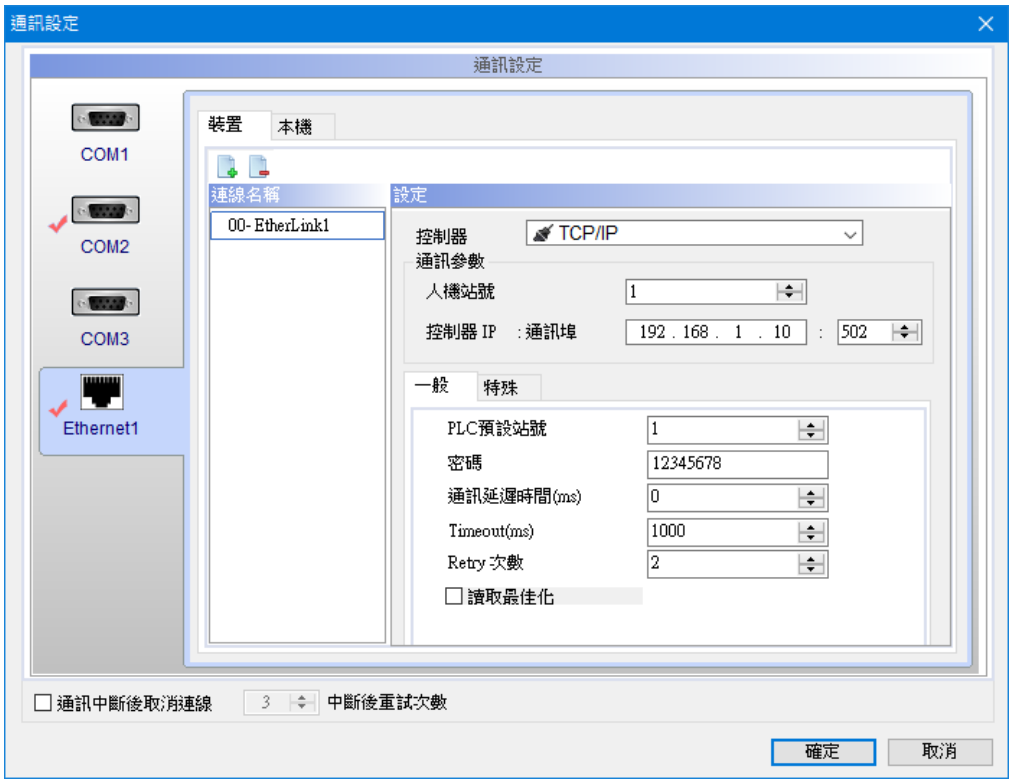
■ 输出脚位

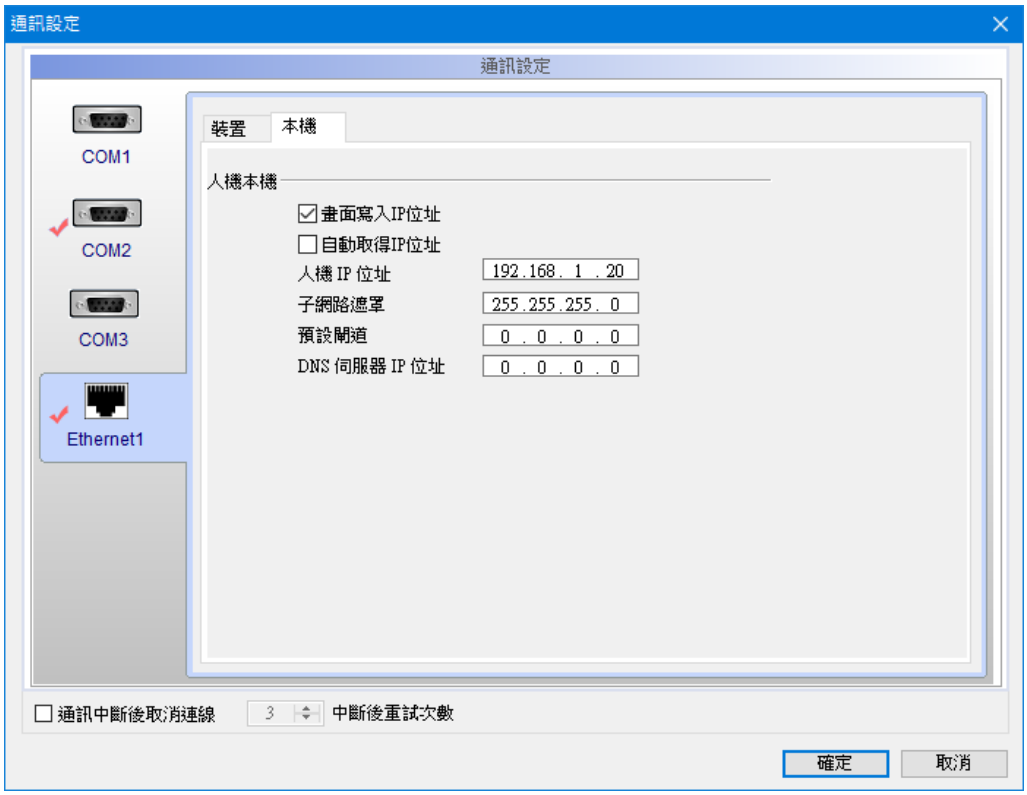
脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	完成时为True；反之则为False。
Busy	BOOL	True / False (False)	成立时表示此功能块进行中。
Error	BOOL	True / False (False)	错误发生时为True。
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	错误发生时记录的错误码 详细说明请参考第9章

■ 参考范例

此范例说明 DXMC 控制器如何透过 TCP Socket 通讯来写入位值至台达人机 DOP-110WS 装置内存储器，其中 DXMC IP 地址为 192.168.1.10、人机 DOP-110WS IP 地址为 192.168.1.20。

1. 台达人机 DOP-110WS 通信设置如下：

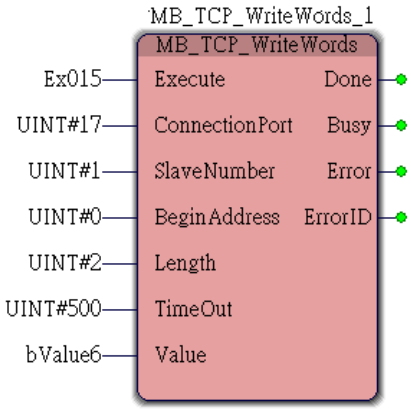




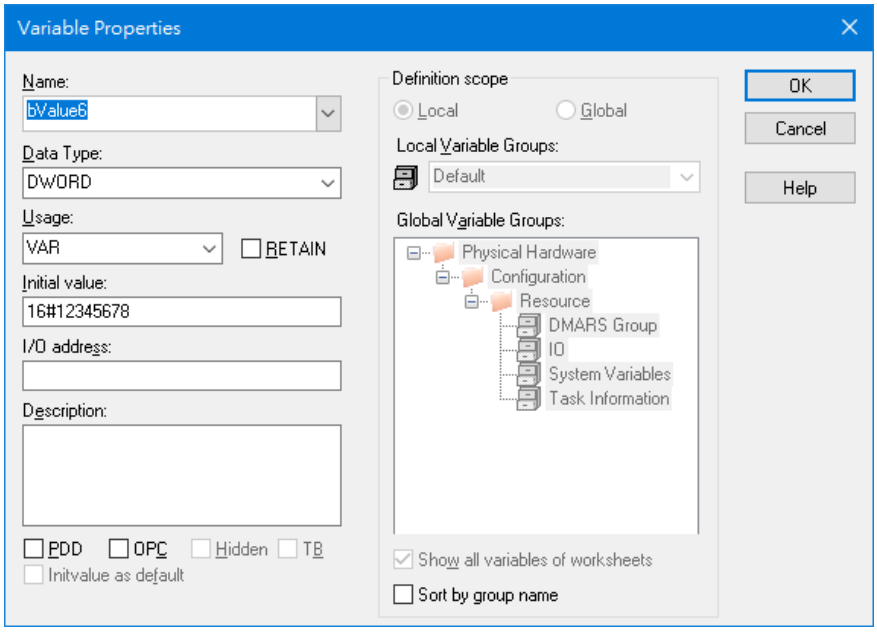
6

2. 台达人机 DOP-110WS 画面建立数值输入组件$0，其格式为 Hexadecimal：(台达人机的 $0 ~ $1023 对应 Modbus 地址为 W40001 ~ W41024，详细请参考台达人机联机手册)。

3. 先使用 MB_TCP_Connect 功能块来建立 Socket，详细请参考章节 6.5.7.8。
4. MB_TCP_WriteWords 功能块设定如下：



5. 写入数据存放至 bValue6 变量，并设定 bValue6 变量格式为 DWORD，并设定初始值为 0x12345678。



6. 将 Ex015 脚位由 False 设定为 True。
7. MB_TCP_WriteWords 功能块将 bValue6 变量写入至 Socket 编号 17，其从站站号为 1、起始位置为 0、长度为 2 words 的位置。
8. 此功能块执行成功时，Done 脚位变为 True，台达人机 $0 的内容变更为 0x12345678。

6.5.7.16 MB_WriteBit

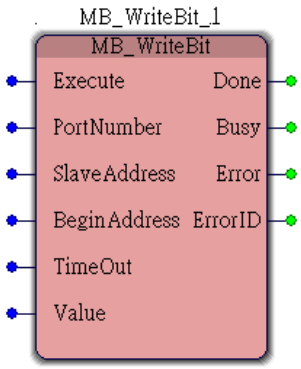
■ 类型

Function Block

■ 功能描述

使用串行端口通讯写入一个位值至指定位置。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，触发此功能。
PortNumber	UINT	0 ~ 1 (0)	串行埠编号，目前仅支持DXMC本体上的串行埠 0：COM0 (不支援) 1：COM1
SlaveAddress	UINT	0 ~ 65535 (0)	欲写入资料之从站站号
BeginAddress	UINT	0 ~ 65535 (0)	欲写入数据之起始位置
TimeOut	UINT	0 ~ 65535 (0)	设定逾时时间；单位：ms
Value	BOOL	True / False (False)	写入位值

■ 输出脚位

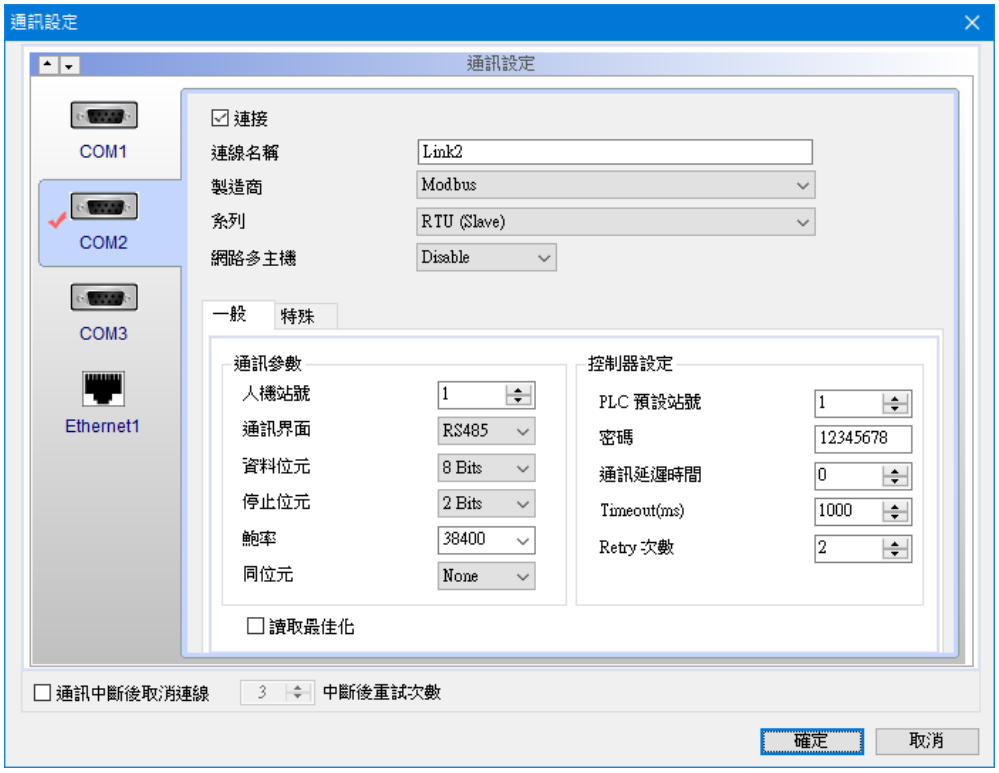
脚位名称	数据型态	设定值(默认值)	功能
Done	BOOL	True / False (False)	完成时为True;反之则为False。
Busy	BOOL	True / False (False)	成立时表示此功能块进行中。
Error	BOOL	True / False (False)	当错误发生时为True
ErrorID	WORD	0x0000~0xFFFF (0)	当错误发生时记录的错误码 详细说明请参考第9章节

6

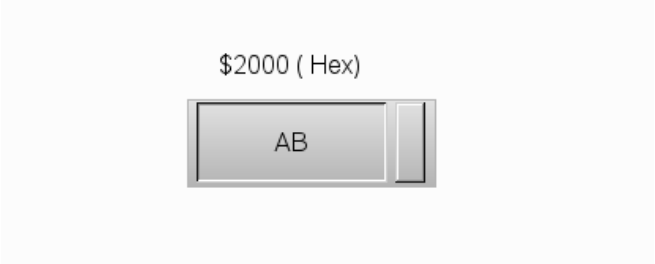
■ 参考范例

DXMC 控制器透过串行埠读取台达人机 DOP-110WS 的内存数据，其中 DXMC 当作主站(Master)、人机 DOP-110WS 当作从站(Slave)。

1. 作为从站的台达人机 DOP-110WS 通信设置如下：

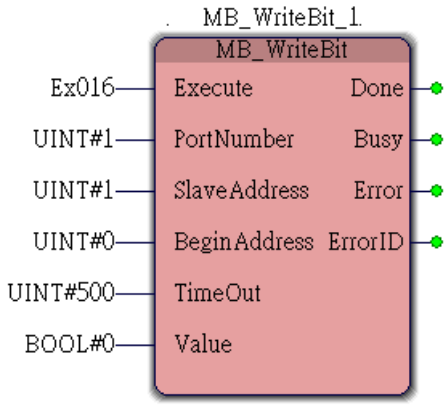


2. 台达人机 DOP-110WS 画面建立数值输入组件$2000 格式为 Hexadecimal：(台达人机的 $2000.0 ~ $2063.15 对应 Modbus 地址为 B00001 ~ B01024，详细请参考台达人机联机手册)。



3. 透过 DMARS 软件来对串行端口进行通信设置。将 DXMC 设为 MODBUS 主站；波特率 38400 数据位 8 bits，同位检查 None，停止位 2 bits，协议为 RTU。

4. MB_WriteBit 功能块设定如下图，串行埠为 COM1、从站站号为 1、起始位置为 0、Timeout 时间为 500 ms。



5. 写入位值 0。
6. 将 Ex016 脚位由 Fasle 设定为 True。
7. MB_WriteBit 功能块将位值 0 写入串行埠 COM1，从站站号为 1，起始位置 0。
8. 此功能块执行成功时，Done 脚位变为 True，台达人机 $2000 的内容变更为 0x00AA。

6

6.5.7.17 MB_WriteBits

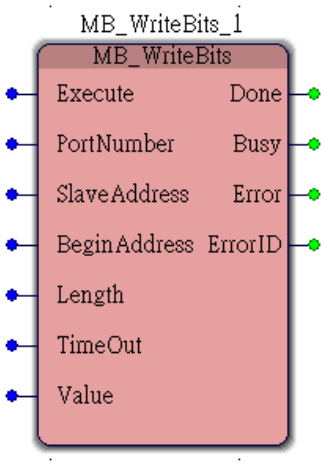
■ 类型

Function Block

■ 功能描述

使用串行端口通讯写入一组位长度的数据串至指定位置。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，触发此功能。
PortNumber	UINT	0 ~ 1 (0)	串行埠编号，目前仅支持DXMC本体上的串行埠 0：COM0 (不支援) 1：COM1
SlaveAddress	UINT	0 ~ 65535 (0)	欲写入资料之从站站号
BeginAddress	UINT	0 ~ 65535 (0)	欲写入数据之起始位置
TimeOut	UINT	0 ~ 65535 (0)	设定逾时时间；单位：ms
Value	ANY	-	写入数值。根据写入长度的不同，需要设定不同的数据型态。

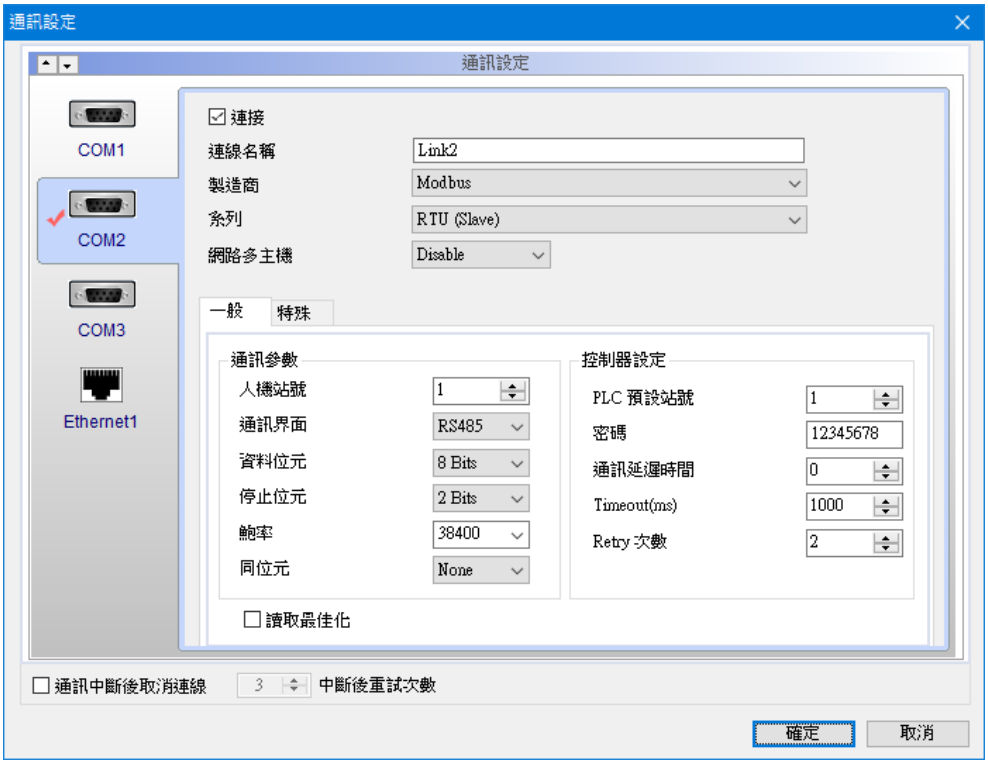
■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	完成时为True；反之则为False。
Busy	BOOL	True / False (False)	成立时表示此功能块执行中。
Error	BOOL	True / False (False)	错误发生时为True。
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	错误发生时记录的错误码 详细说明请参考第9章

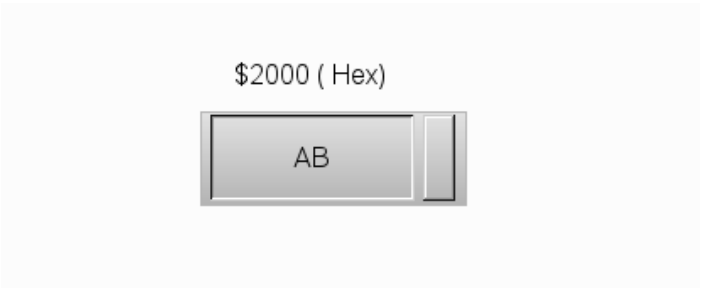
■ 参考范例

DXMC 控制器透过串行埠读取台达人机 DOP-110WS 装置的内存数据，其中 DXMC 当作主站(Master)、人机 DOP-110WS 当作从站(Slave)。

1. 作为从站的台达人机 DOP-110WS 通信设置如下：



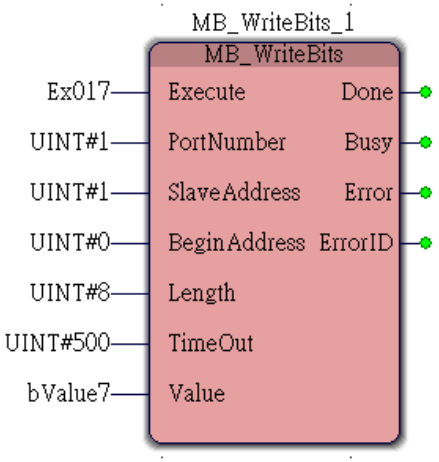
2. 台达人机 DOP-110WS 画面建立数值输入组件$2000，其格式为 Hexadecimal：(台达人机的$2000.0 ~ $2063.15 对应 Modbus 地址为 B00001 ~ B01024，详细请参考台达人机联机手册)。



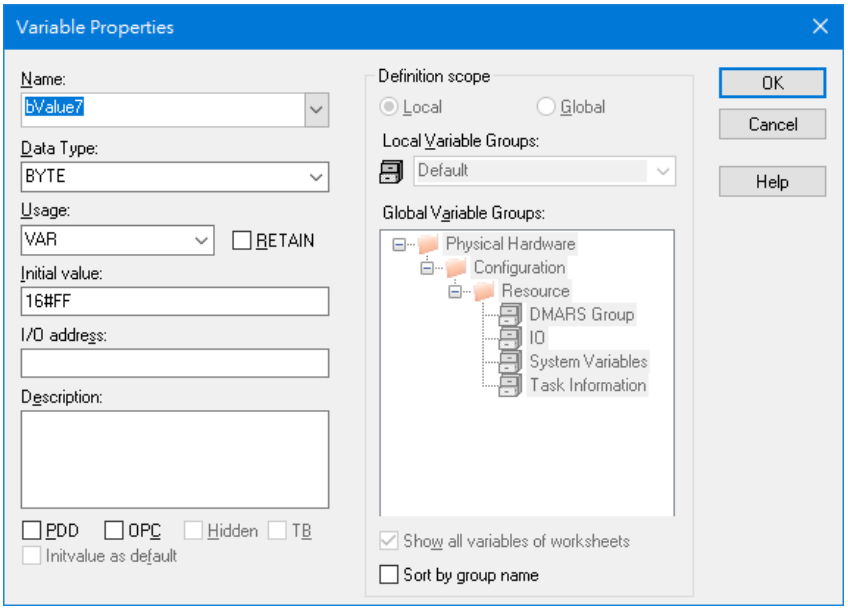
6

3. 透过 DMARS 软件来对串行端口进行通信设置。将 DXMC 设为 MODBUS 主站，其波特率为 38400，数据位为 8 bits，同位检查为 None，停止位为 2 bits，协议为 RTU。

4. MB_WriteBits 功能块设定如下图，串行埠为 COM1、从站站号为 1、起始位置为 0、写入长度为 8、Timeout 时间为 500 ms。



5. 设定写入位值存放于 bValue7 变量，其 bValue7 变量格式为 BYTE、初始值为 0x00FF。



6. 将 Ex017 脚位由 Fasle 设定为 True。
7. MB_WriteBits 功能块将 bValue7 变量写入串行端口 COM1、从站站号为 1、写入位置为 0、长度为 8 bits 的位置。
8. 此功能块执行成功时，Done 脚位变为 True，台达人机$2000 的内容变更为 0x00FF。

6

6.5.7.18 MB_WriteWord

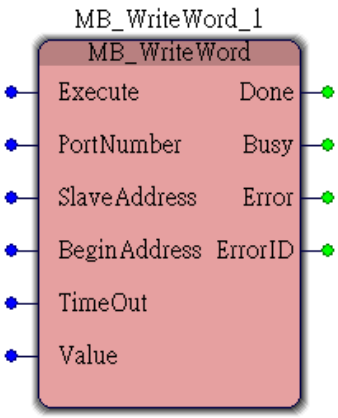
■ 类型

Function Block

■ 功能描述

使用串行端口通讯写入一个 word 长度的数据至指定位置。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，触发此功能。
PortNumber	UINT	0 ~ 1 (0)	串行埠编号，目前仅支持DXMC本体上的串行埠 0：COM0 (不支援) 1：COM1
SlaveAddress	UINT	0 ~ 65535 (0)	欲写入资料之从站站号
BeginAddress	UINT	0 ~ 65535 (0)	欲写入数据之起始位置
TimeOut	UINT	0 ~ 65535 (0)	设定逾时时间；单位：ms
Value	UINT	0 ~ 65535 (0)	写入字组(word)。

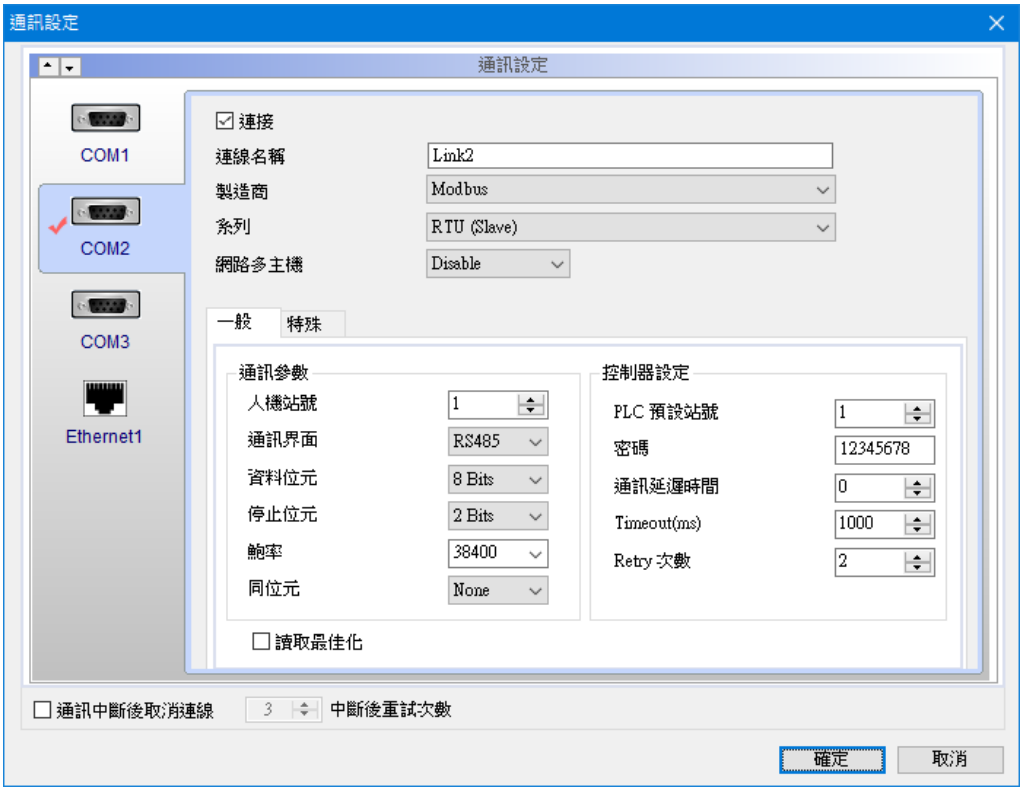
■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	完成时为True；反之则为False。
Busy	BOOL	True / False (False)	成立时表示此功能块执行中。
Error	BOOL	True / False (False)	错误发生时为True。
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	错误发生时记录的错误码 详细说明请参考第9章

■ 参考范例

DXMC 控制器透过串行端口通讯读取台达人机 DOP-110WS 装置之内存数据，其中 DXMC 当作主站(Master)、人机 DOP-110WS 当作从站(Slave)。

1. 作为从站的台达人机 DOP-110WS 通信设置如下：

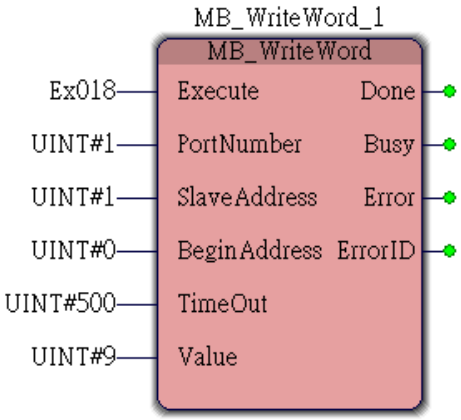


2. 台达人机 DOP-110WS 画面建立数值输入组件$0，其格式为 Hexadecimal：(台达人机的 $0 ~ $1023 对应 Modbus 地址为 W40001 ~ W41024，详细请参考台达人机联机手册)。

6

3. 透过 DMARS 软件来对串行端口进行通信设置。将 DXMC 设为 MODBUS 主站，其波特率 38400，数据位 8 bits，同位检查 None，停止位为 2 bits，协议为 RTU。

4. MB_WriteWord 功能块设定如下图，串行埠为 COM1、从站站号为 1、起始位置为 0、Timeout 时间为 500 ms。



5. 写入字组 9。
6. 将 Ex018 脚位由 Fasle 设定为 True。
7. MB_WriteWord 功能块将字组 9 写入串行埠 COM1、从站站号为 1、起始位置为 0 的位置。
8. 此功能块执行成功时，Done 脚位变为 True，台达人机$0 的内容变更为 0x89AB0009。

6.5.7.19 MB_WriteWords

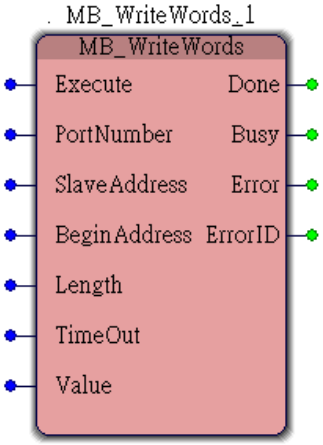
■ 类型

Function Block

■ 功能描述

使用串行端口通讯写入一组 word 长度的数据串至指定位置。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，触发此功能。
PortNumber	UINT	0 ~ 1 (0)	串行埠编号，目前仅支持DXMC本体上的串行埠 0：COM0 (不支援) 1：COM1
SlaveAddress	UINT	0 ~ 65535 (0)	欲写入资料之从站站号
BeginAddress	UINT	0 ~ 65535 (0)	欲写入数据之起始位置
Length	UINT	0 ~ 65535 (0)	欲写入数据之读取长度
TimeOut	UINT	0 ~ 65535 (0)	设定逾时时间；单位：ms
Value	ANY	-	写入数值。根据写入长度的不同，需要设定不同的数据型态。

■ 输出脚位

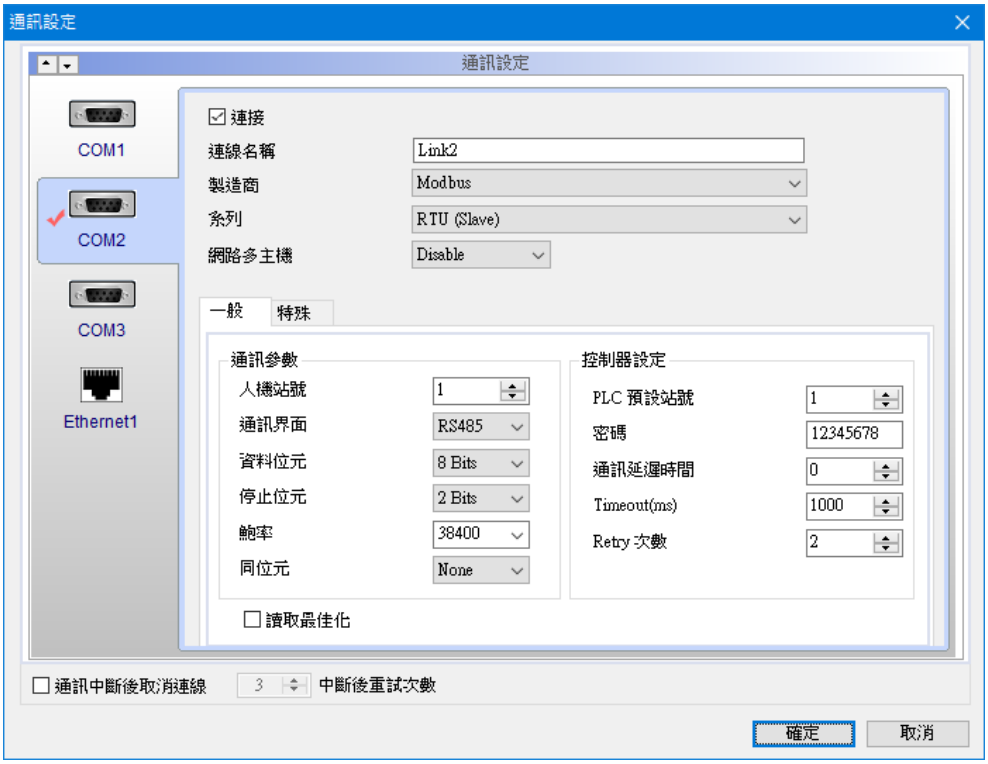
脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	完成时为True；反之则为False。
Busy	BOOL	True / False (False)	成立时表示此功能块进行中。
Error	BOOL	True / False (False)	错误发生时为True。
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	错误发生时记录的错误码 详细说明请参考第9章

6

■ 参考范例

DXMC 控制器透过串行端口通讯，读取台达人机 DOP-110WS 装置之内存数据，其中 DXMC 当作主站（Master)、人机 DOP-110WS 当作从站（Slave)。

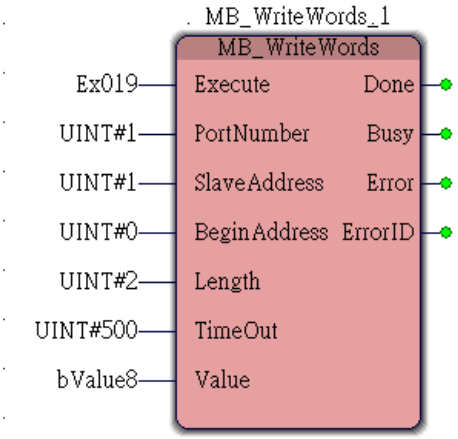
1. 作为从站的台达人机 DOP-110WS 通信设置如下：



2. 台达人机 DOP-110WS 画面建立数值输入组件$0，其格式为 Hexadecimal：(台达人机的$0 ~ $1023 对应 Modbus 地址为 W40001 ~ W41024，详细请参考台达人机联机手册)。

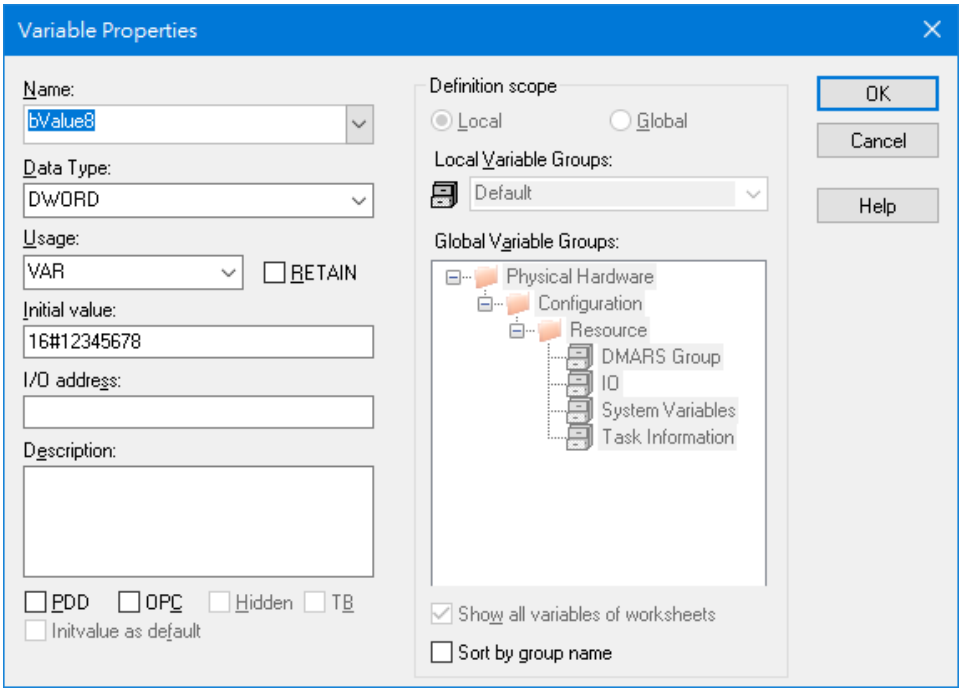
3. 透过 DMARS 软件来对串行端口进行通信设置。将 DXMC 设为 MODBUS 主站，其波特率为 38400，数据位为 8 bits，同位检查为 None，停止位为 2 bits，协议为 RTU。

4. MB_WriteWord 功能块设定如下图，串行埠 COM1、从站站号为 1、起始位置为 0 、写入长度为 2 words、Timeout 时间为 500 ms。



6

5. 设定写入位值存放于 bValue8 变量，其 bValue8 变量格式为 DWORD、初始值为 0x12345678。



6. 将 Ex019 脚位由 False 设定为 True。
7. MB_WriteWords 功能块将 bValue8 变量写入串行端口 COM1、从站站号为 1、写入位置为 0、长度为 2 words 的位置。
8. 此功能块执行成功时，Done 脚位变为 True，台达人机$0 的内容变更为 0x12345678。

6.5.8 凸轮编辑功能块 (ECAM Editor)

6.5.8.1 ECAM_GenTableByFile

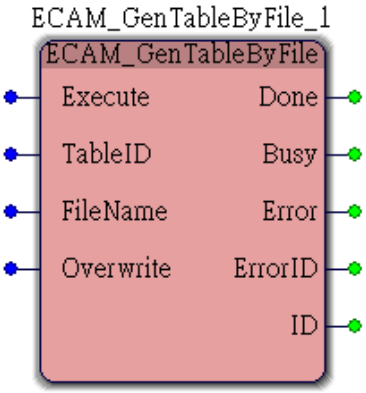
■ 类型

Function Block

■ 功能描述

藉由 DMARS 所产生的.bin 档案来建立凸轮表。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	当Execute为上升缘时，此功能即开始执行
TableID	UINT	0 ~ 65535 (0)	欲产生的凸轮表编号注
FileName	STRING	字符串	凸轮表bin文件名
Overwrite	BOOL	True / False (False)	TableID存在时，是否覆写 False：不覆写 True：覆写

注：用户如果未输入 TableID，则系统会自动将 TableID 设为 0。

■ 输出脚位

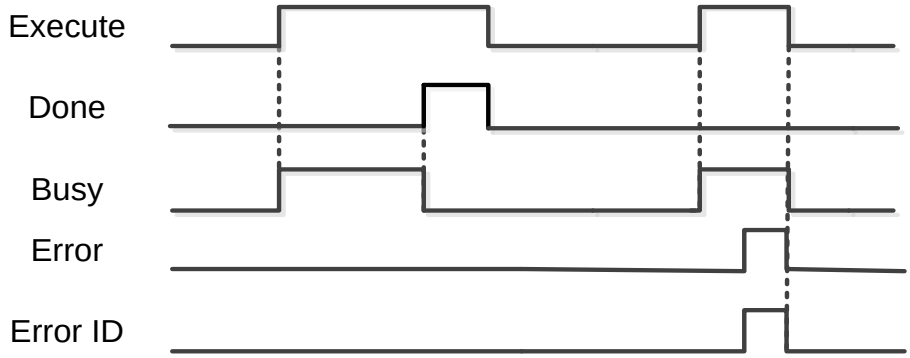
脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	绝对寻址完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	错误发生时记录的错误码 详细说明请参考第9章
ID	UINT	0 ~ 65535 (0)	产生的凸轮表编号

6

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	新增点数据传入系统时	■ Execute转为False时 ■ 若Execute为False而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时
Error / ErrorID	指令的执行条件或输入值发生错误时（错误码记录在ErrorID）	在Execute下降沿时 (清除ErrorID记录之错误码)

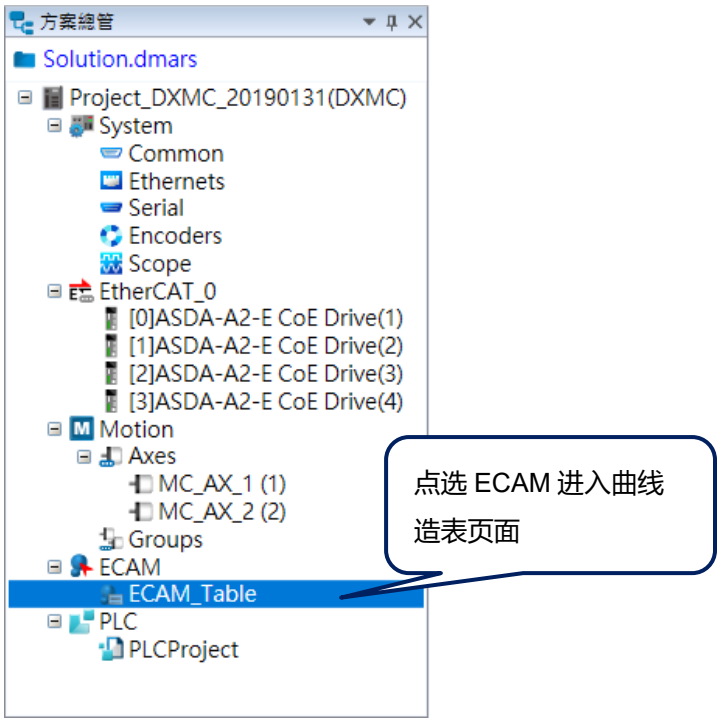
■ 输出脚位的变化时序



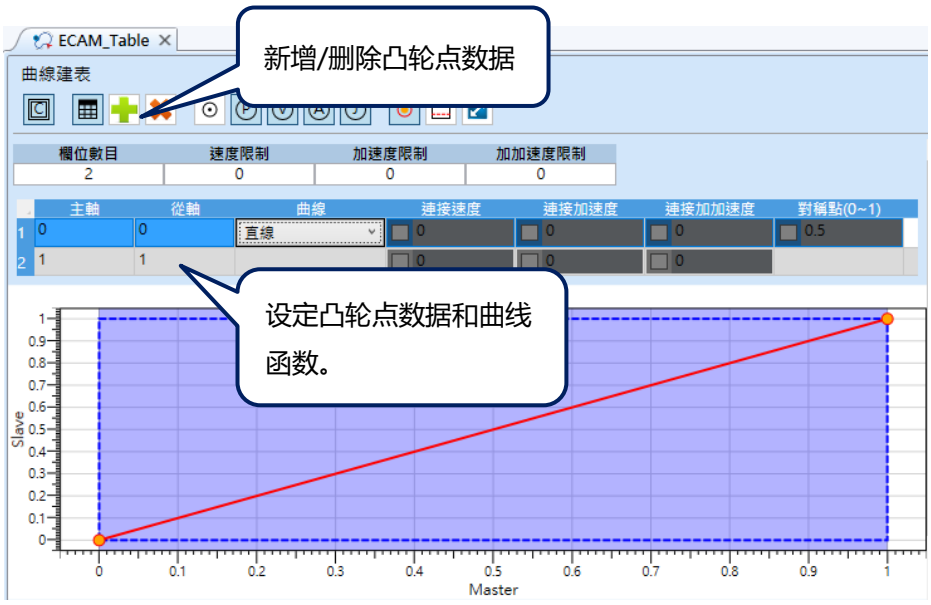
■ 参考范例

此范例说明如何透过 ECAM_GenTableByFile 来产生凸轮表。

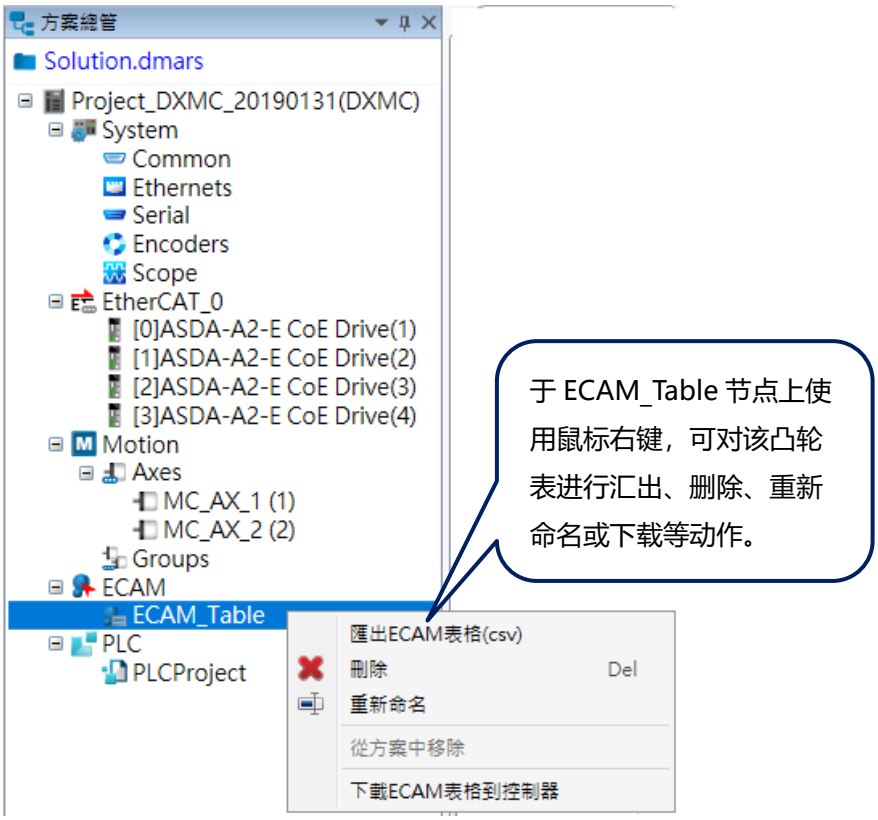
1. 先使用 DMARS 项目树中的 ECAM 造表功能来产生凸轮表数据



2. 设定第一点凸轮点数据：主轴 0、从轴 0、曲线函数为直线
设定第二点凸轮点数据：主轴 1、从轴 1 如下图所示：



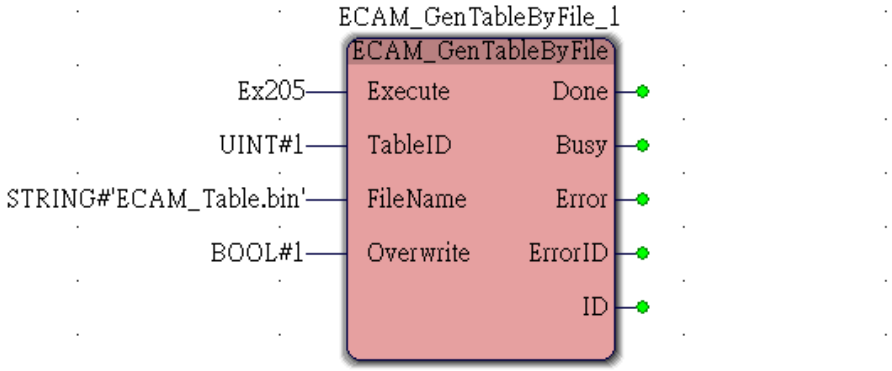
3. 凸轮表设定完成之后，于项目树 ECAM_Table 节点上点选鼠标右键，可以将该表格下载至控制器。



6

4. 以上图为例，ECAM 表格下载之后，文件名为 ECAM_Table.bin，使用者也可以在 ECAM 节点右键单击进入「电子凸轮项目管理」窗口来确认存在于控制器内的凸轮表。

5. ECAM_GenTableByFile 功能块设定如下图：
凸轮表编号为 1；文件名为 ECAM_Table.bin；覆写状态为 1。



6. 将 Ex205 脚位由 Fasle 设定为 True。
7. 此功能块执行成功时，Done 脚位变为 True，ECAM_GenTableByFile 功能块会根据 ECAM_Table.bin 档案内的凸轮点数据生成凸轮表，ID 脚位输出 1。
(欲查看变更后的凸轮表内容可以使用 ECAM_ReadTableByID 或 ECAM_ReadPointsByID)

6.5.8.2 ECAM_GenTableByData

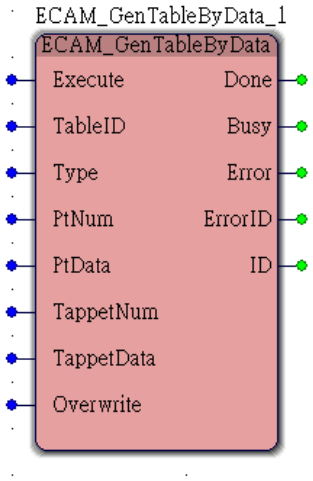
■ 类型

Function Block

■ 功能描述

根据输入的凸轮数据来建立凸轮表。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，此功能即开始执行
TableID	UINT	0 ~ 65535 (0)	欲产生的凸轮表编号*1
Type	UINT	0 ~ 1 (0)	0：手动建表(目前不支持) 1：曲线建表
PtNum	UINT	0 ~ 32767 (0)	凸轮数据点数
PtData	S_CAM_POINT_ARR	-	凸轮点资料*2
TappetNum	UINT	0 ~ 65535 (0)	欲新增挺杆点数量
TappetData	S_CAM_TAPPET_ARR	-	欲新增挺杆点资料
Overwrite	BOOL	True / False (False)	TableID存在时，是否覆写 False：不覆写 True：覆写

注：

1. 用户如果没有输入 TableID，则系统会自动 TableID 设为 0。
2. S_CAM_POINT_ARR 为自定义数据型态，该数组包含 10,000 个 S_CAM_BASIC_POINT 结构，详细请参考 6.5.9 节。

6

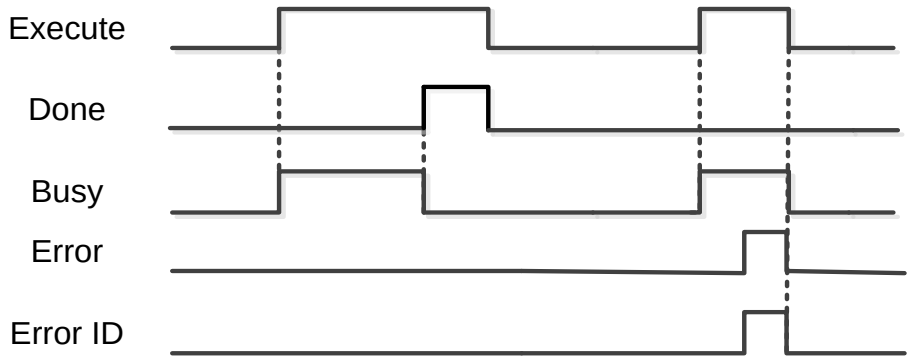
■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	绝对寻址完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 详细说明请参考第9章
ID	UINT	0 ~ 65535 (0)	使用的凸轮表编号

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	新增点数据传入系统时	■ Execute转为False时 ■ 若Execute为False而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为 False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在 ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

■ 输出脚位的变化时序

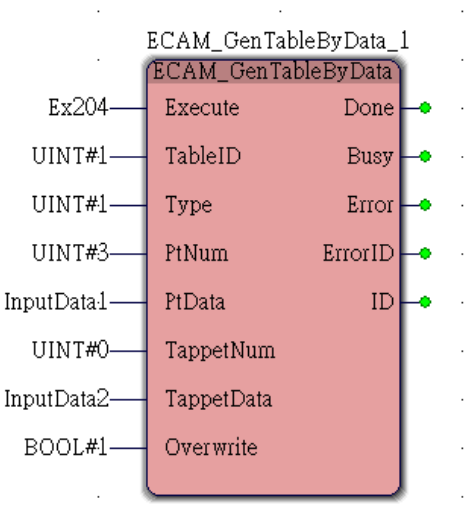


■ 参考范例

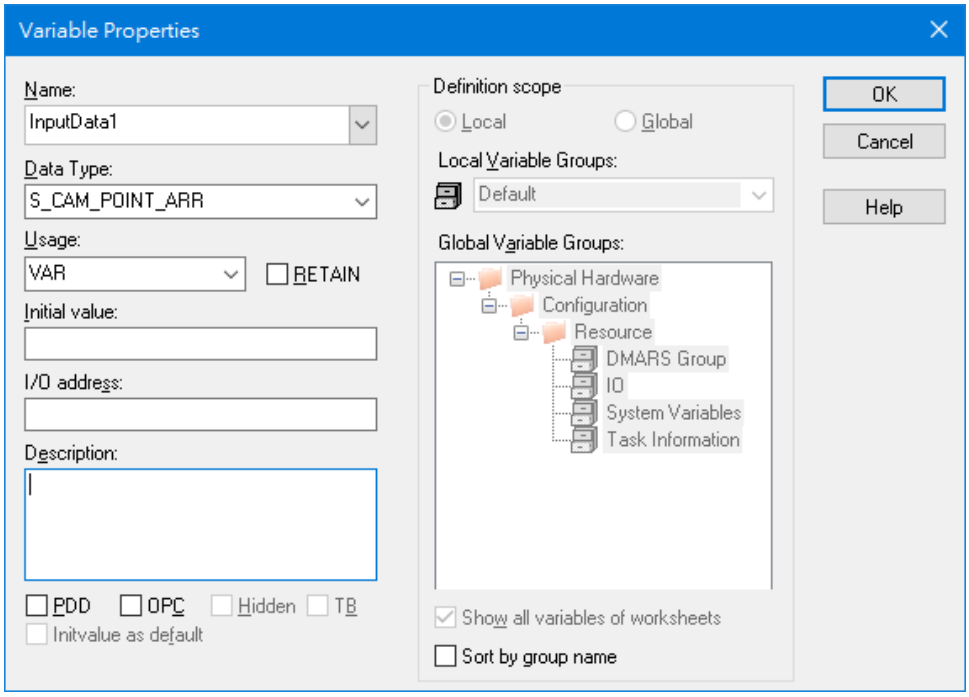
此范例说明如何透过 ECAM_GenTableByData 来产生凸轮表。

1. ECAM_GenTableByData 功能块设定如下图：

   凸轮表编号为 1；曲线造表；凸轮点数量为 3；凸轮表数据数组为 InputData1；挺杆点数量为 0；挺杆点资料为 Inputdata2；覆写状态为 1。

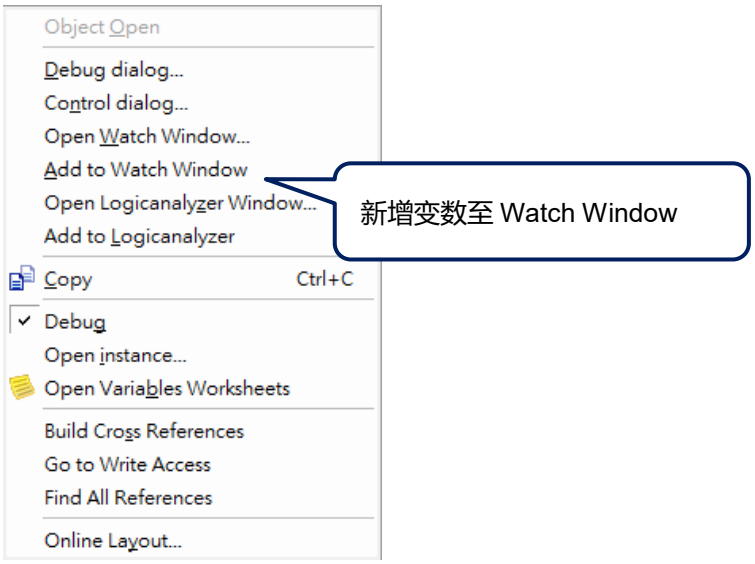


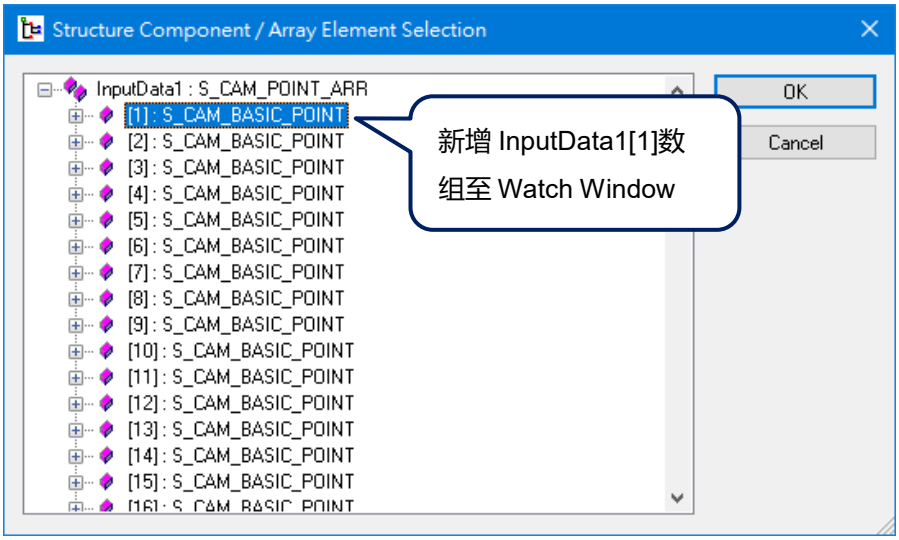
2. InputData1 变量数据型态为 S_CAM_POINT_ARR



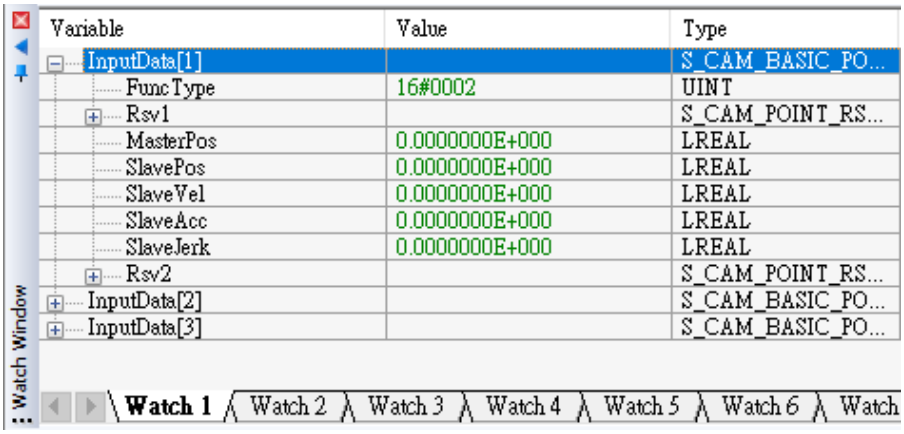
6

3. 在在线监控模式  中对变量 InputData1 按鼠标右键，可以将该变量新增至 Watch Window。

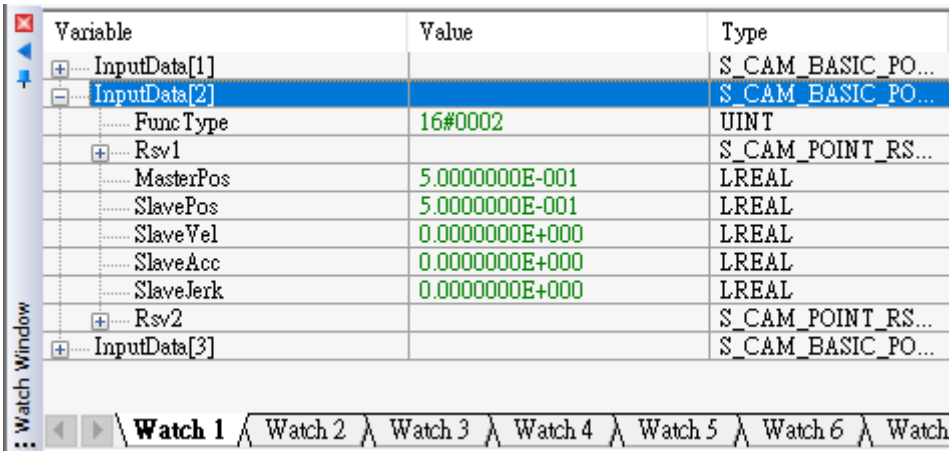




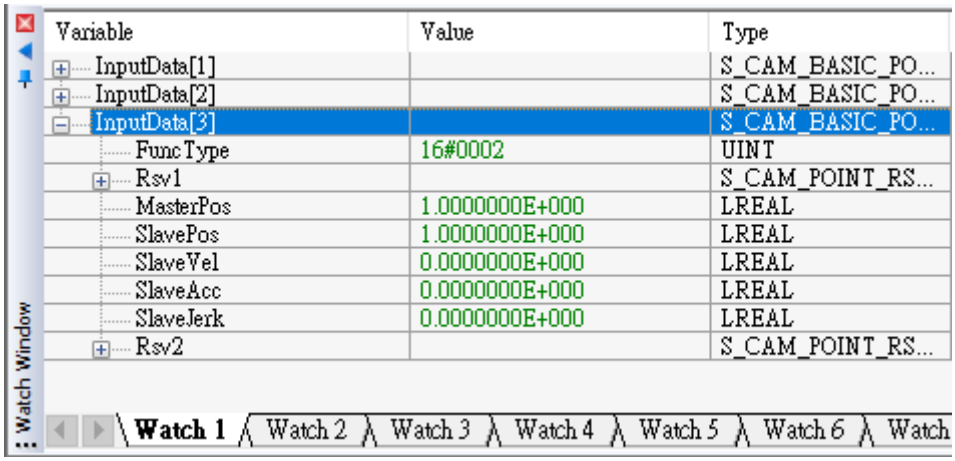
4. 在 Watch Window 中可以对变量 InputData1[1] 填写欲增加的凸轮点信息。FuncType 填入 2 (抛物线)，MasterPos 填入 0，SlavePos 填入 0。



5. 重复步骤 3 和 4，在 Watch Window 中可以对变量 InputData1[2]填写欲增加的凸轮点信息。FuncType 填入 2 (抛物线)，MasterPos 填入 0.5，SlavePos 填入 0.5。



6. 重复步骤 3 和 4，在 Watch Window 中可以对变量 InputData1[3]填写欲增加的凸轮点信息。FuncType 填入 2 (抛物线)，MasterPos 填入 1.0，SlavePos 填入 1.0。



7. 将 Ex204 脚位由 False 设定为 True。
8. 此功能块执行成功时，Done 脚位变为 True，ID 脚位变为 1。
   ECAM_GenTableByData 功能块根据 InputData1[1]、InputData1[2]、InputData1[3] 三笔凸轮点资料生成凸轮表。
   (欲查看变更后的凸轮表内容，可以使用 ECAM_ReadTableByID 或 ECAM_ReadPointsByID)

6

6.5.8.3 ECAM_GenTableByVel

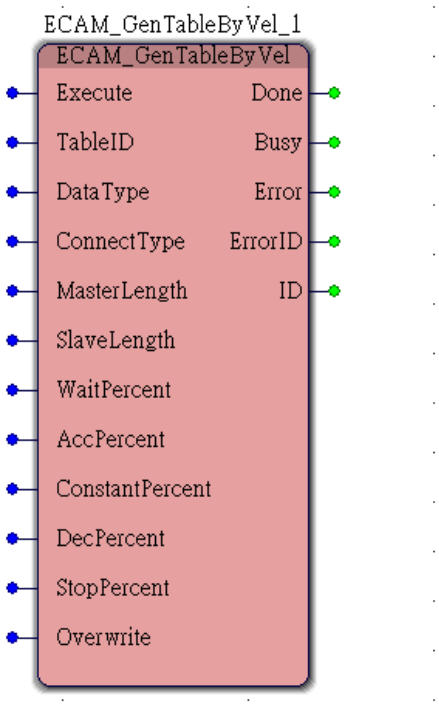
■ 类型

Function Block

■ 功能描述

根据输入的主轴长度、从轴长度以及等待、加速、等速、减速、停止速度的百分比，来建立凸轮表。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，此功能即开始执行
TableID	UINT	0 ~ 65535 (0)	欲产生的凸轮表编号*1
DataType	UINT	0 ~ 1 (0)	0：手动建表(目前不支持) 1：曲线建表
ConnectType	UINT	0 ~ 2 (0)	加减速时的连接曲线 0：三次曲线 1：五次曲线 2：七次曲线
MaterLength	LREAL	正数或0.0 (0.0)	凸轮表的主轴周期长度 (单位：用户单位)
SlaveLength	LREAL	正数或0.0 (0.0)	凸轮表的从轴周期长度（单位:用户单位)
WaitPercent	USINT	0 ~ 100 (0)	等待区的百分比*2

脚位名称	数据型态	设定值 (默认值)	功能
AccPercent	USINT	0 ~ 100 (0)	加速区的百分比*2
Constant Percent	USINT	0 ~ 100 (0)	等速区的百分比*2
DecPercent	USINT	0 ~ 100 (0)	减速区的百分比*2
StopPercent	USINT	0 ~ 100 (0)	停止区的百分比*2
Overwrite	BOOL	True / False (False)	TableID存在时，是否覆写 False：不覆写 True：覆写

注:

1. 用户如果未输入 TableID，则系统会自动将 TableID 设为 0。
2. 等待区+加速区+等速区+减速区+停止区的百分比需等于 100%，若不等于 100%则会产生错误报警。

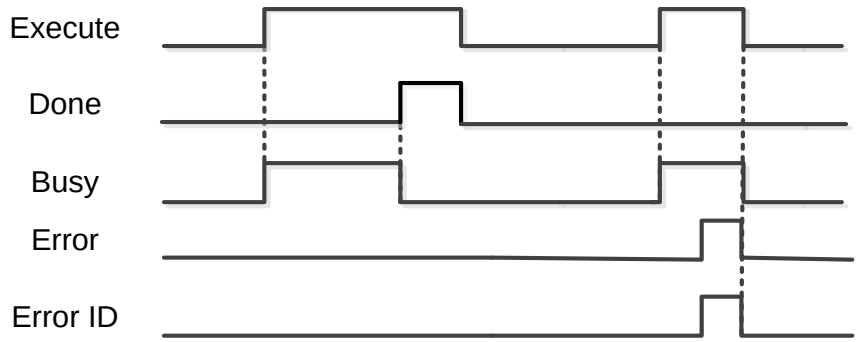
■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	绝对寻址完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 详细说明请参考第9章
ID	UINT	0 ~ 65535 (0)	使用的凸轮表编号

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	新增点数据传入系统时	■ Execute转为False时 ■ 若Execute为False，而Done 转为True，此时 Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	Execute下降沿时 (清除ErrorID记录之错误码)

■ 输出脚位的变化时序

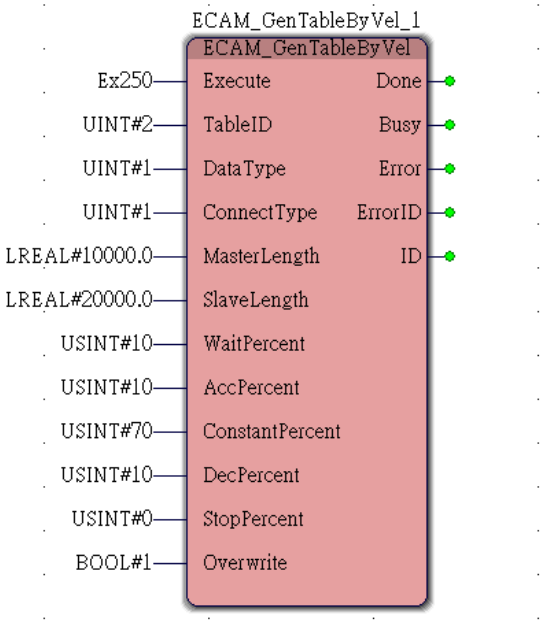


6

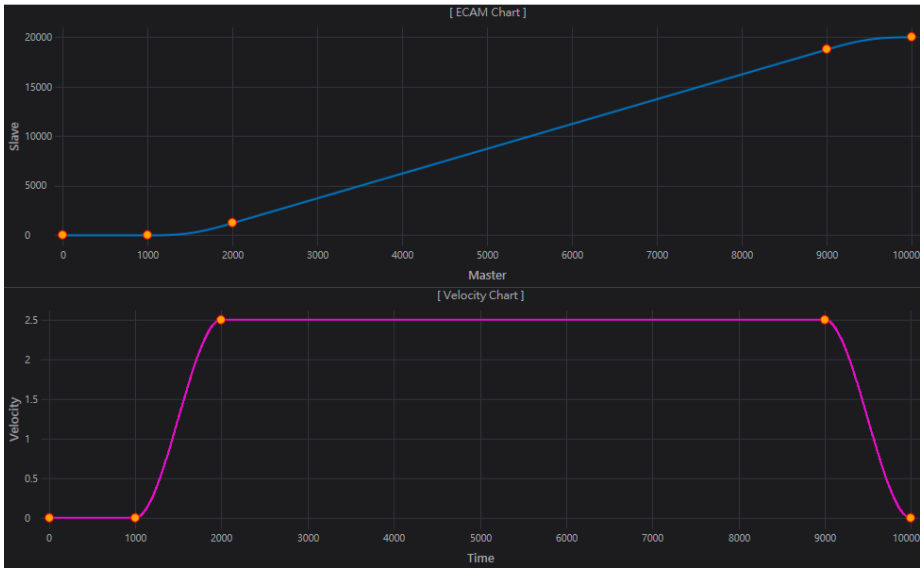
■ 参考范例

此范例说明如何透过 ECAM_GenTableByVel 来产生凸轮表。

1. ECAM_GenTableByVel 功能块设定如下图：
   凸轮表 ID 为 2；曲线造表 (DataType=1)；连接方式为五次曲线 (ConnectType=1)；主轴周期长度为 10000.0；从轴周期长度为 20000.0；等待区占全部的 10%；加速区占全部的 10%；等速区占全部的 70%；减速区占全部的 10%；停止区占全部的 0%，代表没有停止区。并且确认每一区的加总为 100% (若各区的加总不等于 100%，则功能块无法正确执行并且报错)。



2. 将 Ex250 脚位由 False 设定为 True。
3. 此功能块执行成功时，Done 脚位变为 True，ID 脚位变为 2。使用者可以利用 ECAM_SaveTable 将凸轮表储存之后，由 DMARS 读取该凸轮表。如下图所示，主轴距离与从轴的距离分别为使用者输入的 10000.0 与 20000.0，并且根据各个区段的百分比生成凸轮表。



6.5.8.4 ECAM_AddPoints

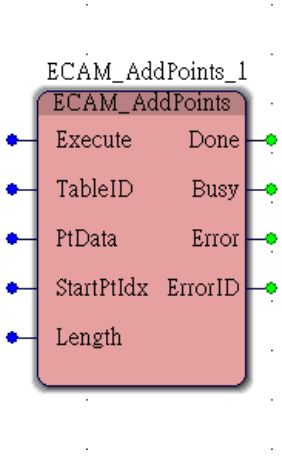
■ 类型

Function Block

■ 功能描述

为目前系统中的凸轮表加入数据点。(在使用此功能块之前，必须设有在线凸轮表，若无凸轮表，则需使用 ECAM_GenTableByData 或者 ECAM_GenTableByVel 等可产生凸轮表之功能块产生在线凸轮表)

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值(默认值)	功能
Execute	BOOL	True / False (False)	当Execute为上升缘时，此功能即开始执行
TableID	UINT	0 ~ 65535 (0)	欲使用的凸轮表编号
PtData	S_CAM_POINT_ARR	-	新增凸轮点资料注
StartPtIdx	UINT	2 ~ 65535 (2)	设定凸轮表新增起始点
Length	UINT	0 ~ 65535 (0)	设定新增点数长度

注：S_CAM_POINT_ARR 为自定义数据型态，该数组包含 10,000 个 S_CAM_BASIC_POINT 结构，详细请参考第 6.5.9 节。

■ 输出脚位

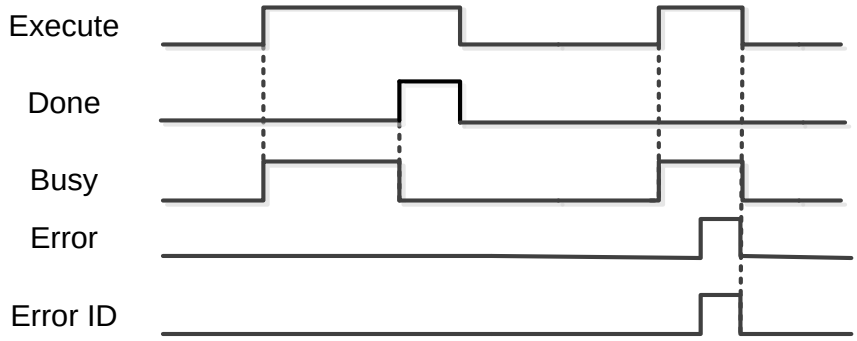
脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	绝对寻址完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	错误发生时记录的错误码 详细说明请参考第9章

6

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	新增点数据传入系统时	■ Execute转为False时 ■ 若Execute为False而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为 False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在 ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

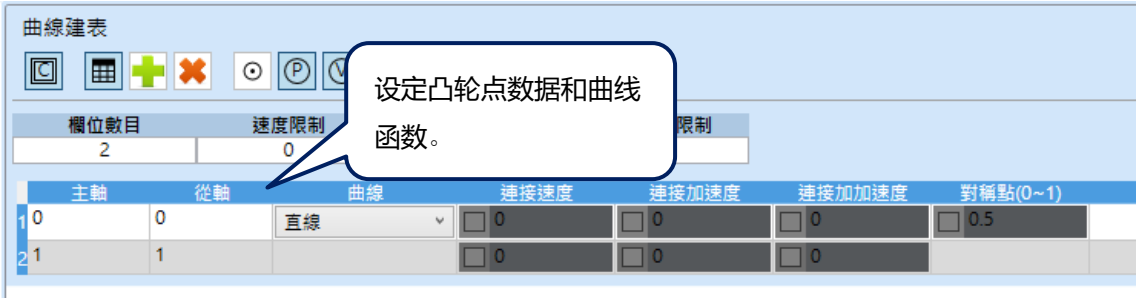
■ 输出脚位的变化时序



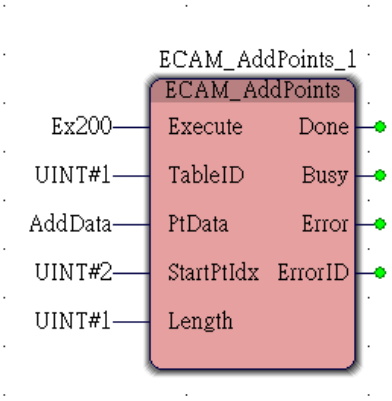
■ 参考范例

此范例说明如何透过 ECAM_AddPoints 来新增点位至现有的凸轮表。

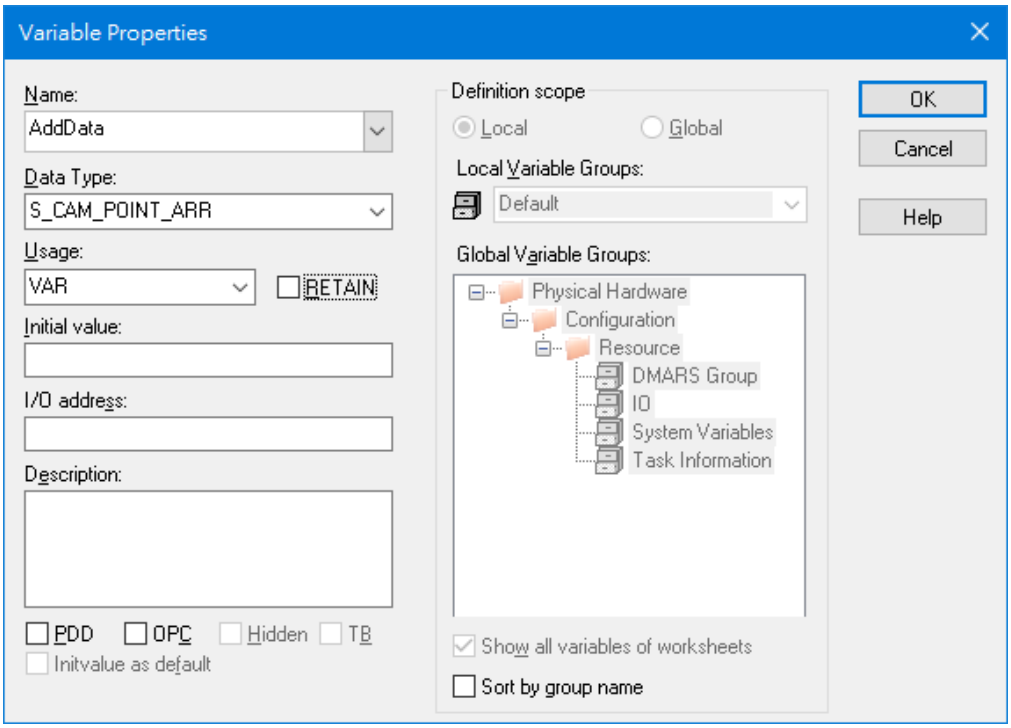
1. 先使用其他功能块来建立凸轮表，该凸轮表包含两点信息，详细请参考章节 6.5.8.1。



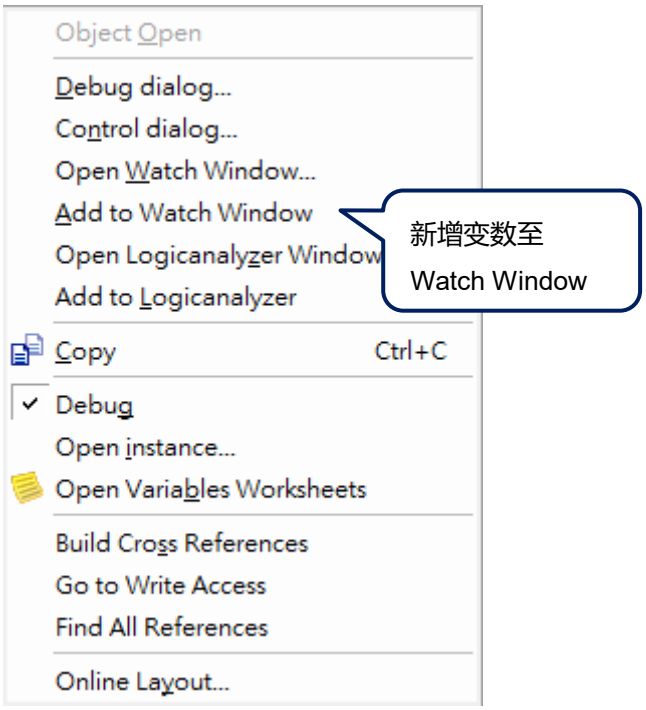
2. ECAM_AddPoints 功能块设定如下图：

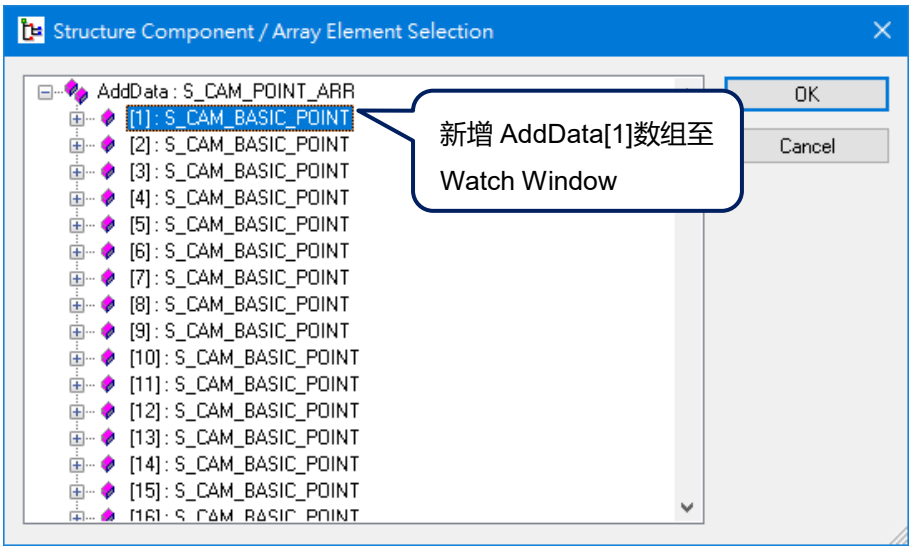


3. AddData 变量数据型态为 S_CAM_POINT_ARR。

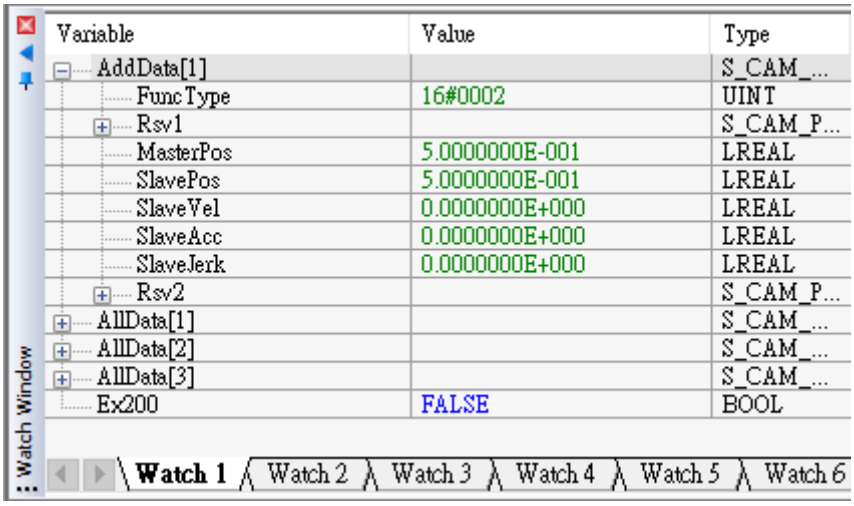


4. 在在线监控模式  中于变量 AddData 按鼠标右键，可以将该变量新增至 Watch Window。





5. 在 Watch Window 中可以对变量 AddData[1]填写欲增加的凸轮点信息；FuncType 填入 2 (抛物线)，MasterPos 填入 0.5，SlavePos 填入 0.5。



6. 将 Ex200 脚位由 False 设定为 True。
7. ECAM_AddPoints 功能块将该凸轮点信息写入凸轮表中第二点。
8. 此功能块执行成功时，Done 脚位变为 True，凸轮表总点数更为三点。(欲查看变更后的凸轮表内容，可以使用 ECAM_ReadTableByID 或 ECAM_ReadPointsByID)
9. 新的凸轮表生效时机会参考 ECAM_SetOnlineChgMode 功能块的 ActiveMode 脚位，详细请参考章节 6.5.8.17。

6

6.5.8.5 ECAM_DelPoints

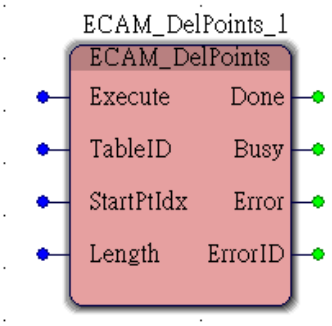
■ 类型

Function Block

■ 功能描述

删除目前系统中的凸轮表中的关键点。(使用此功能块之前，必须在设有在线凸轮表，若无凸轮表则需使用 ECAM_GenTableByData 或者 ECAM_GenTableByVel 等可产生凸轮表之功能块产生在线凸轮表)

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	当Execute为上升缘时，此功能即开始执行
TableID	UINT	0 ~ 65535 (0)	欲使用的凸轮表编号
StartPtIdx	UINT	2 ~ 65535 (2)	设定凸轮表删除起始点
Length	UINT	0 ~ 65535 (0)	设定删除点数长度

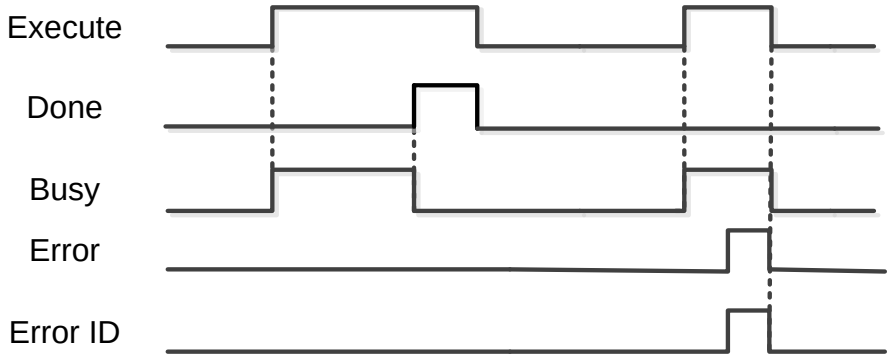
■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	绝对寻址完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码详细说明请参考第9章

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	新增点数据传入系统时	■ Execute转为False时 ■ Execute为False而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

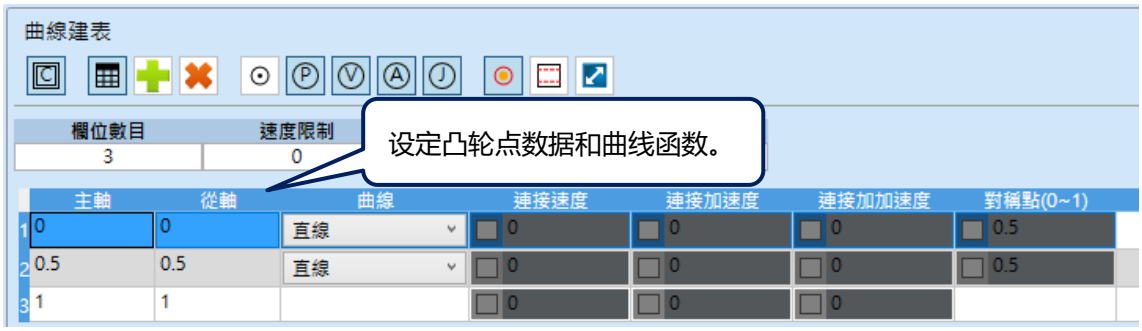
■ 输出脚位的变化时序



■ 参考范例

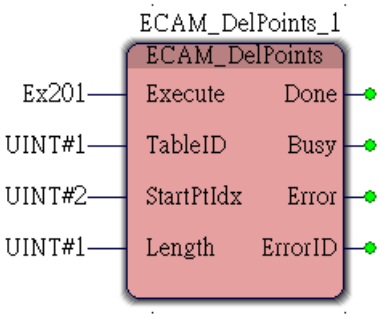
此范例说明如何透过 ECAM_DelPoints 来将现有凸轮表中删除关键点。

1. 先使用其他功能块来建立凸轮表，该凸轮表包含三点信息，详细请参考章节6.5.8.1。



6

2. ECAM_DelPoints 功能块设定如下图：
   凸轮表编号为 1；删除点起始地址为 2；删除长度为 1。



3. 将 Ex201 脚位由 False 设定为 True。
4. ECAM_DelPoints 功能块将删除凸轮表中第二点信息。
5. 此功能块执行成功时，Done 脚位变为 True，凸轮表总点数变更为两点。(欲查看变更后的凸轮表内容，可以使用 ECAM_ReadTableByID 或 ECAM_ReadPointsByID)
6. 新的凸轮表生效时机会参考 ECAM_SetOnlineChgMode 功能块的 ActiveMode 脚位，详细请参考章节 6.5.8.17。

6.5.8.6 ECAM_ModifyPoints

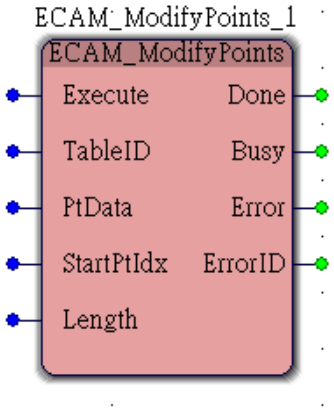
■ 类型

Function Block

■ 功能描述

为目前系统中的凸轮表修改特定关键点。(使用此功能块之前，必须设有在线凸轮表，若无凸轮表则需使用 ECAM_GenTableByData 或者 ECAM_GenTableByVel 等可产生凸轮表之功能块产生在线凸轮表)。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	当Execute为上升缘时，此功能即开始执行
TableID	UINT	0 ~ 65535 (0)	欲使用的凸轮表编号
PtData	S_CAM_POINT_ARR	-	新增凸轮点资料[注]
StartPtIdx	UINT	1 ~ 65535 (1)	设定凸轮表修改起始点
Length	UINT	0 ~ 65535 (0)	设定修改点数长度

注：S_CAM_POINT_ARR 为自定义数据型态，该数组包含 10,000 个 S_CAM_BASIC_POINT 结构，详细请参考第 6.5.9 节。

■ 输出脚位

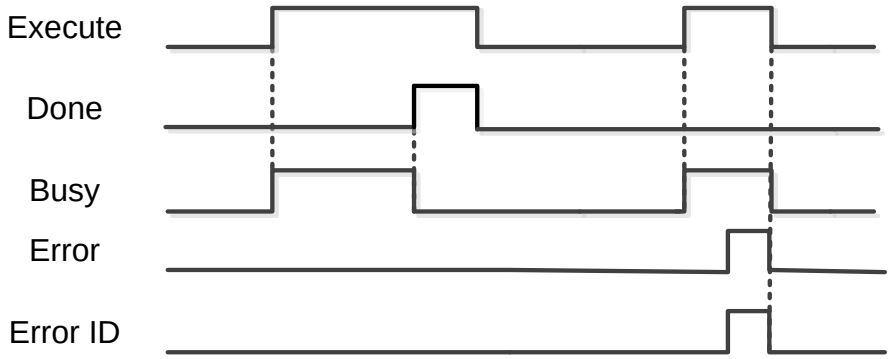
脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	绝对寻址完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码详细说明请参考第9章

6

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	新增点数据传入系统时	■ Execute转为False时 ■ Execute为False而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	Execute下降沿时 (清除ErrorID记录之错误码)

■ 输出脚位的变化时序

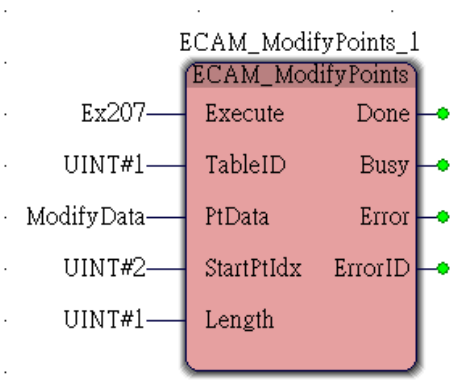


■ 参考范例

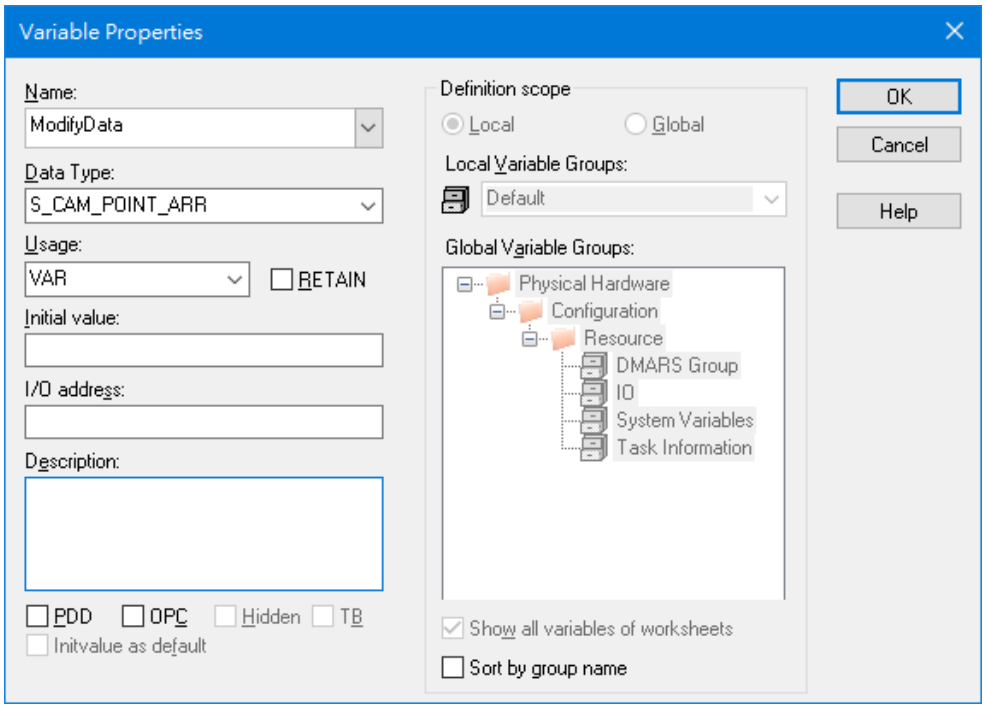
此范例说明如何透过 ECAM_ModifyIPoints 来修改现有凸轮表中的特定点信息。

1. 先使用其他功能块来建立凸轮表，该凸轮表包含三点信息，详细请参考章节 6.5.8.1。

2. ECAM_ModifyIPoints 功能块设定如下图：

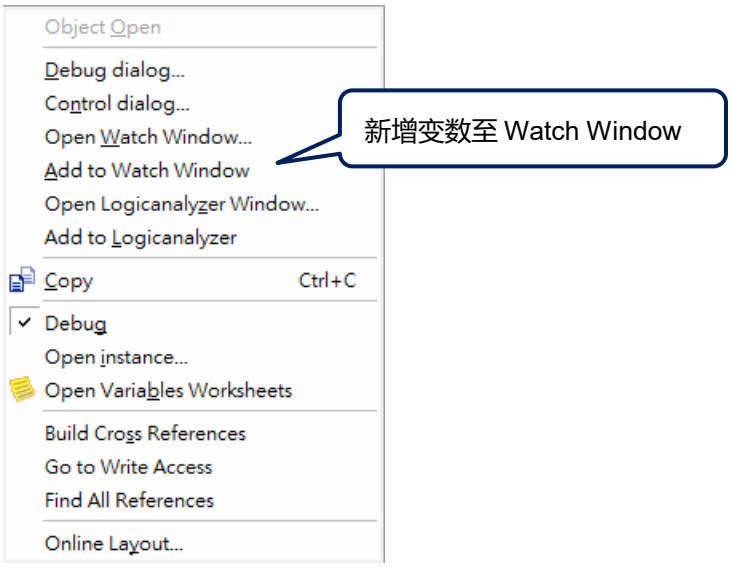


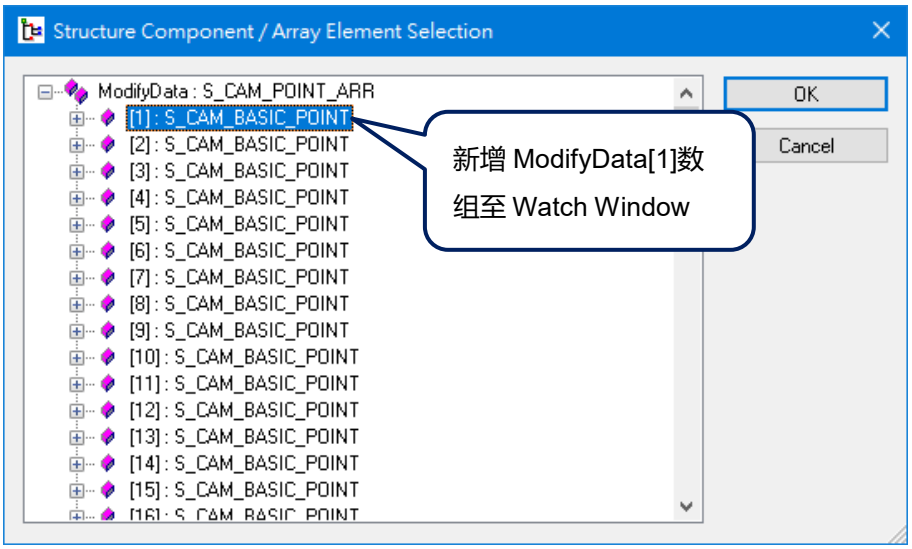
3. ModifyData 变量数据型态为 S_CAM_POINT_ARR



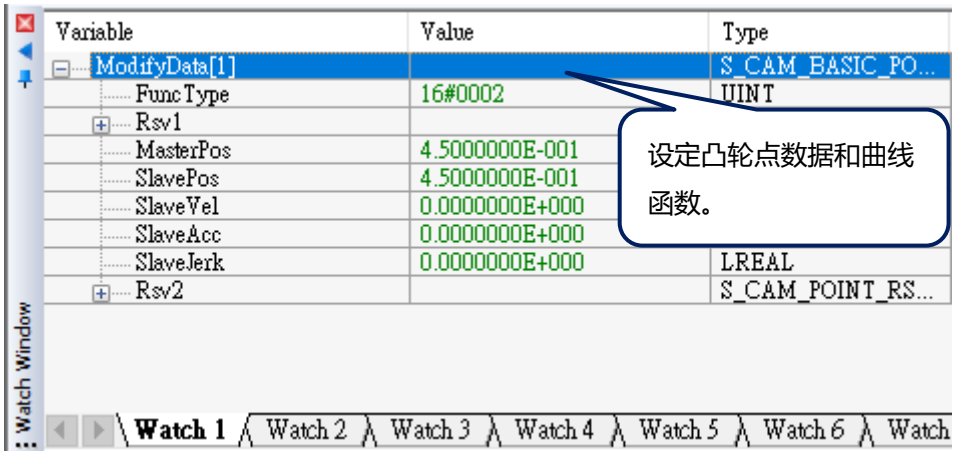
6

4. 在在线监控模式  中于变量 ModifyData 按鼠标右键，可以将该变量新增至 Watch Window。

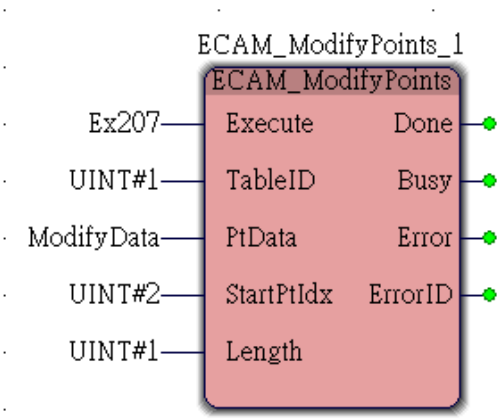




5. 在 Watch Window 中可以对变量 ModifyData[1]填写欲修改的凸轮点信息；FuncType 填入 2 (抛物线)，MasterPos 填入 0.45，SlavePos 填入 0.45。



6. 将 Ex207 脚位由 False 设定为 True。
7. ECAM_ModifyPoints 功能块将 ModifyData[1]该凸轮点信息写入凸轮表中起始地址为 2、长度为 1 的位置。
8. 此功能块执行成功，Done 脚位变为 True，凸轮表中第二点数据被更新。(欲查看变更后的凸轮表内容，可以使用 ECAM_ReadTableByID 或 ECAM_ReadPointsByID)
9. 新的凸轮表生效时机会参考 ECAM_SetOnlineChgMode 功能块的 ActiveMode 脚位，详细请参考章节 6.5.8.17。
10. 先使用其他功能块来建立凸轮表，该凸轮表包含三点信息，详细请参考章节 6.5.8.1。
11. ECAM_ModifylPoints 功能块设定如下图：
    凸轮表编号为 1；删除点起始地址为 2；删除长度为 1。



12. 将 Ex207 脚位由 False 设定为 True。
13. ECAM_ModifyPoints 功能块将修改凸轮表中第二点信息为上述所设定数据。
    FuncType 为 2、MasterPos 为 0.45、SlavePos 为 0.45。
14. 此功能块执行成功时，Done 脚位变为 True，凸轮表总点数变更为两点。(欲查看变更后的凸轮表内容，可以使用 ECAM_ReadTableByID 或 ECAM_ReadPointsByID)
15. 新的凸轮表生效时机会参考 DMC_CamIn 功能块的 ActiveMode 脚位，详细请参考章节 6.5.3.3。

6

6.5.8.7 ECAM_ReadPointsByID

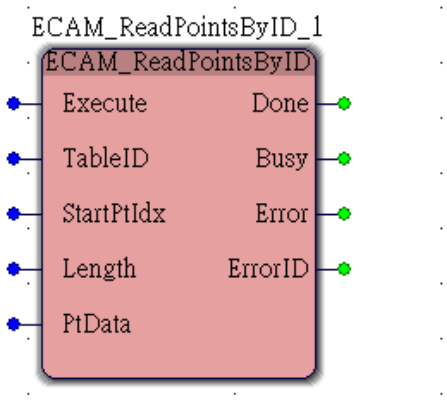
■ 类型

Function Block

■ 功能描述

藉由输入凸轮表的 TableID 和参数来读取凸轮表中关键点信息。(使用此功能块之前，必须设有在线凸轮表，若无凸轮表则需使用 ECAM_GenTableByData 或者 ECAM_GenTableByVel 等可产生凸轮表之功能块产生在线凸轮表。)

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	当Execute为上升缘时，此功能即开始执行
TableID	UINT	0 ~ 65535 (0)	欲使用的凸轮表编号
StartPtIdx	UINT	2 ~ 65535 (2)	设定凸轮表读取起始点
Length	UINT	0 ~ 65535 (0)	设定读取点数长度

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	绝对寻址完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	错误发生时记录的错误码 详细说明请参考第9章
PtData	S_CAM_POINT_ARR	-	读取凸轮点数据[注]

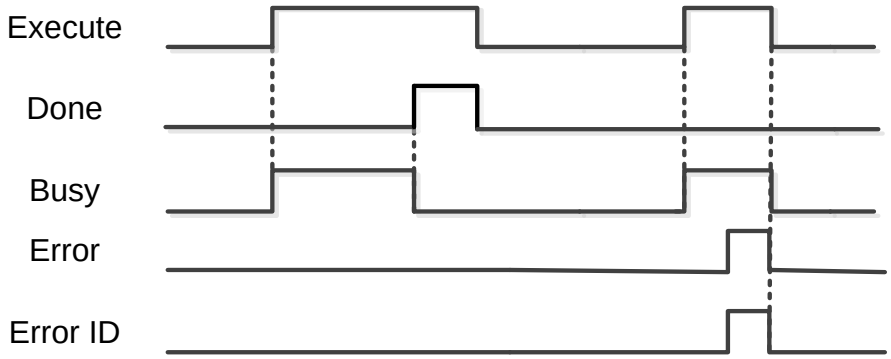
注:

1. S_CAM_POINT_ARR 为自定义数据型态，该数组包含 10,000 个 S_CAM_BASIC_POINT 结构，详细请参考第 6.5.9 章节。
2. PtData 虽然放在功能块的左边，但是此脚位为输出脚位。

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	新增点数据传入系统时	■ Execute转为False时 ■ 若Execute为False而Done转为True，此时Done 维持一个扫描周期的True状态后，立刻转为 False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

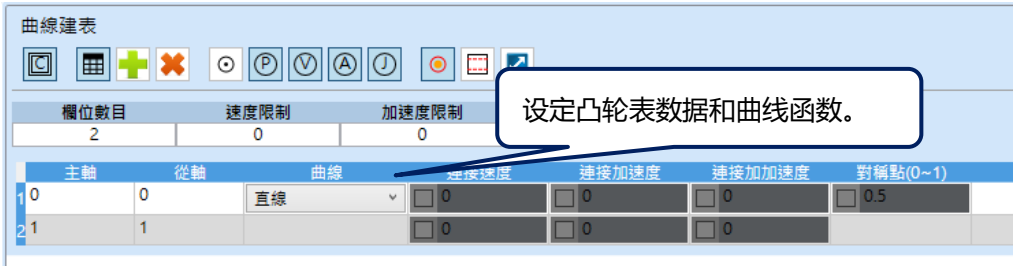
■ 输出脚位的变化时序



■ 参考范例

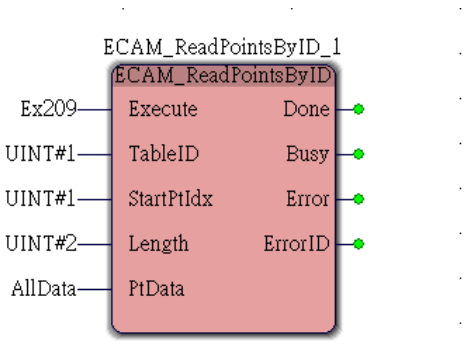
此范例说明如何透过 ECAM_ReadPointsByID 来读取现有的凸轮表数据。

1. 先使用其他功能块来建立凸轮表，该凸轮表包含两点信息，详细请参考章节 6.5.8.1。

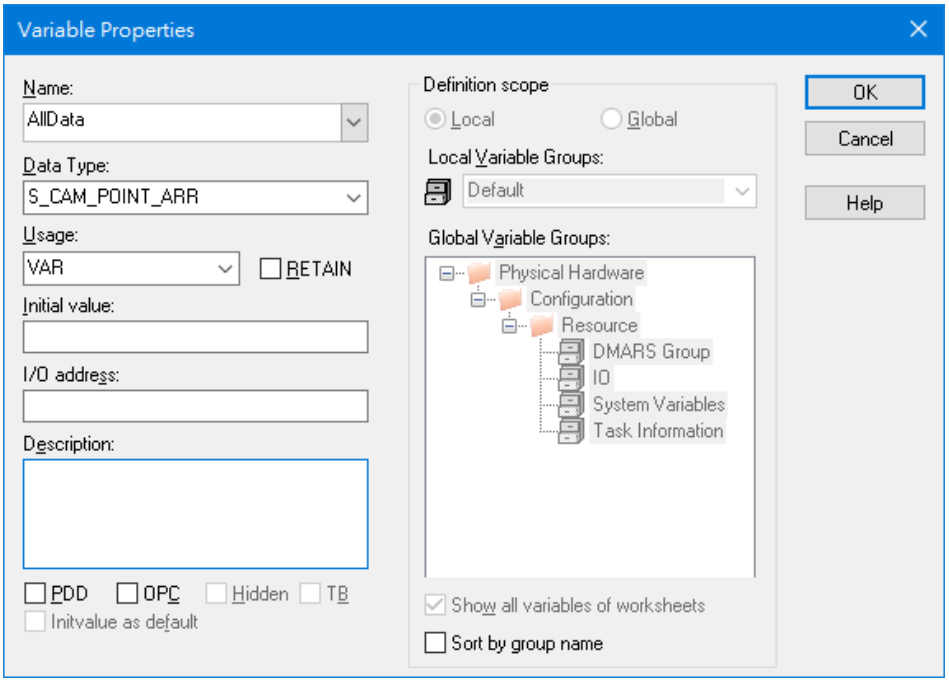


6

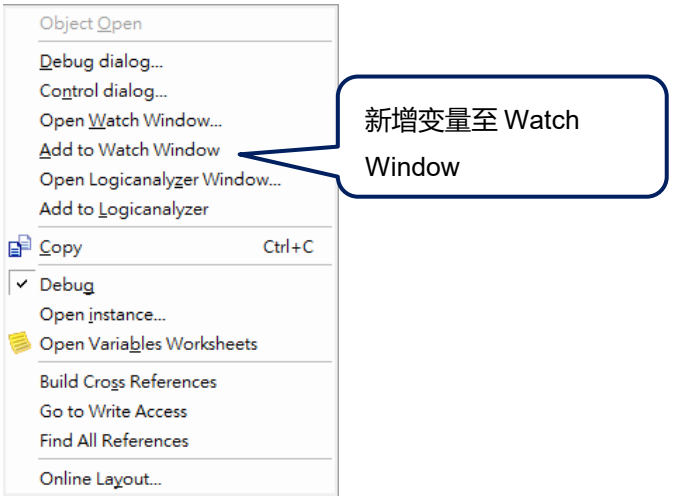
2. ECAM_ReadPointsByID 功能块设定如下图：

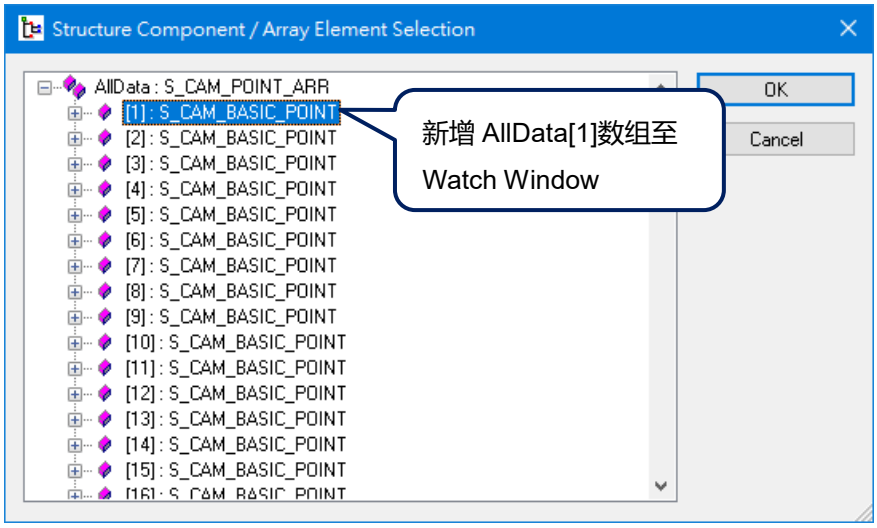


3. AllData 变量数据型态为 S_CAM_POINT_ARR。

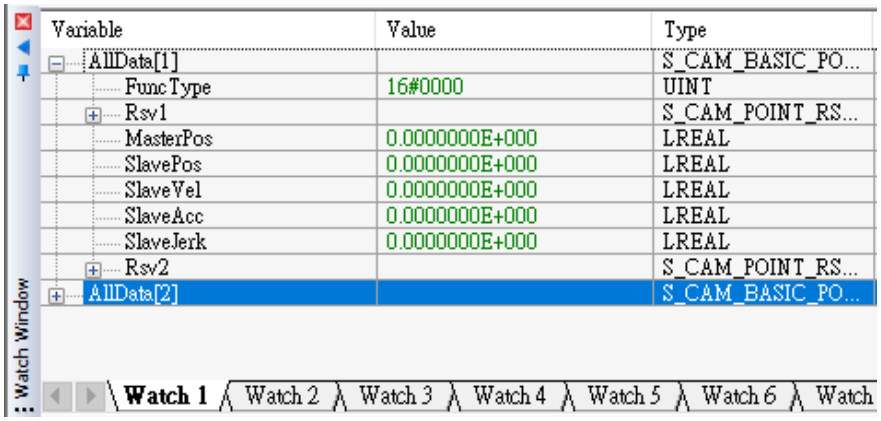


4. 在在线监控模式  中于变量 AllData 按鼠标右键，可以将该变量新增至 Watch Window。

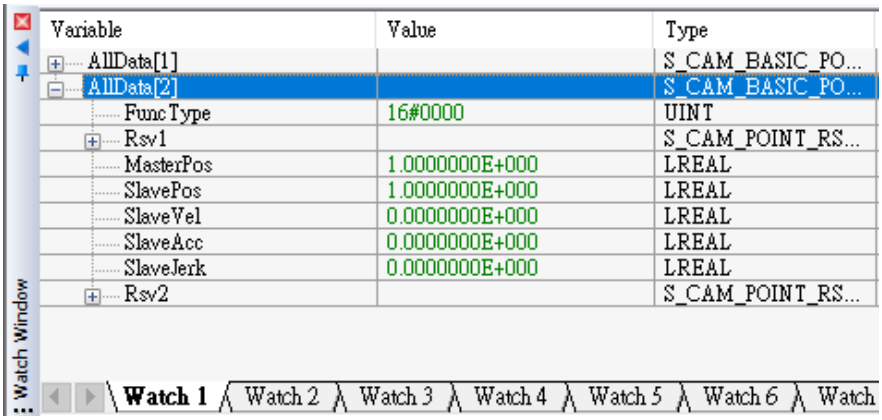




5. 重复步骤 4，以新增 AllData[2]数组至 Watch Window。
6. 此时在 Watch Window 中变量 AllData[1]和 AllData[2]填写数据内容皆为 0。



7. 将 Ex209 脚位由 Fasle 设定为 True。
8. ECAM_ReadPointsByID 功能块将读取凸轮表数据，其起始地址为 1、读取长度为 2 的数据至变量 AllData。
9. 此功能块执行成功时，Done 脚位变为 True，可以透过 Watch Window 中变量 AllData[1]和 AllData[2]来查读取的凸轮表内容。AllData[2]资料如下图所示：



注：凸轮表的最后一点，并不需要曲线函数(FuncType)参数，因此预设为 0。

6

6.5.8.8 ECAM_ReadTableByID

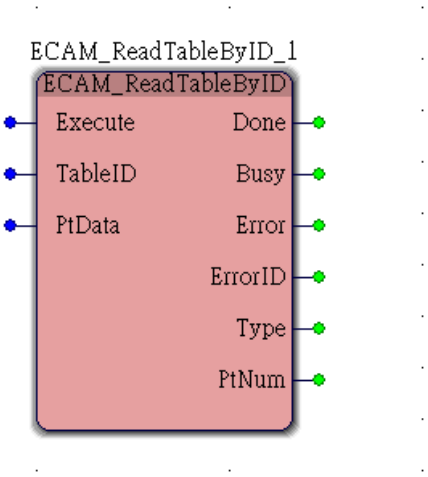
■ 类型

Function Block

■ 功能描述

藉由输入凸轮表的 TableID 来读取整个凸轮表的数据。(使用此功能块之前，必须设有在线凸轮表，若无凸轮表则需使用 ECAM_GenTableByData 或者 ECAM_GenTableByVel 等可产生凸轮表之功能块产生在线凸轮表。)

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	当Execute为上升缘时，此功能即开始执行
TableID	UINT	0 ~ 65535 (0)	欲使用的凸轮表编号

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	绝对寻址完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 错误码详细说明请参考第9章
PtData	S_CAM_POINT_ARR	-	读取凸轮点数据[注]

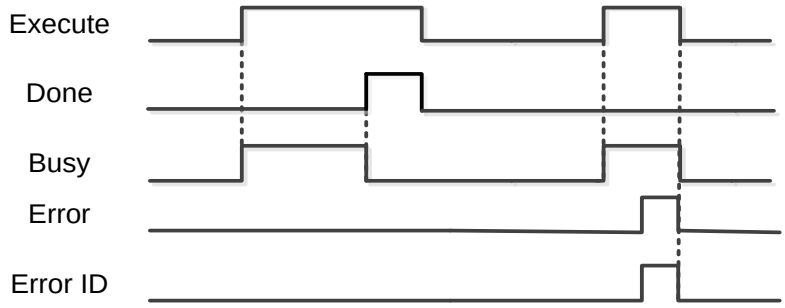
注:

1. S_CAM_POINT_ARR 为自定义数据型态，该数组包含 10,000 个 S_CAM_BASIC_POINT 结构，详细请参考第 6.5.9 章节。
2. PtData 虽然放在功能块的左边，但是此脚位为输出脚位。

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	新增点数据传入系统时	■ Execute转为False时 ■ 若Execute为False而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

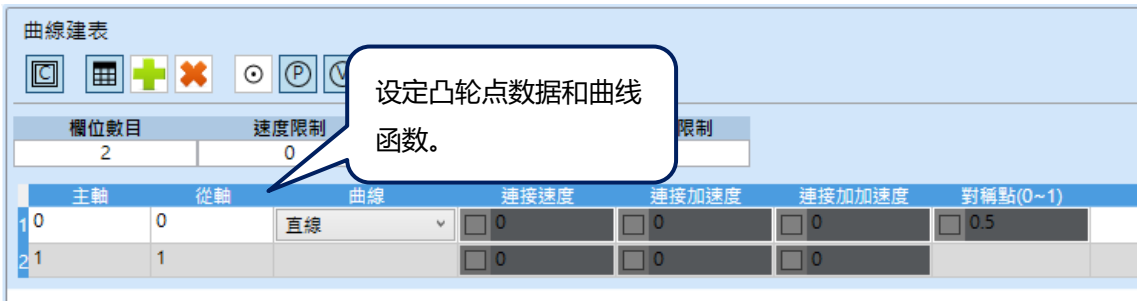
■ 输出脚位的变化时序



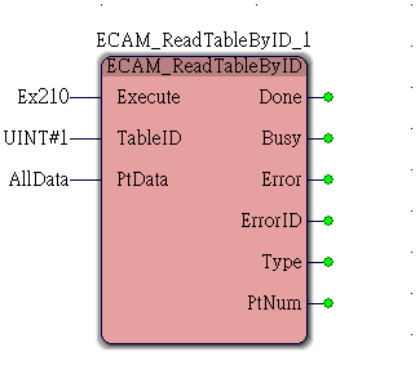
■ 参考范例

此范例说明如何透过 ECAM_ReadTableByID 来读取凸轮表的数据。

1. 先使用其他功能块来建立凸轮表，该凸轮表包含两点信息，详细请参考章节6.5.8.1。

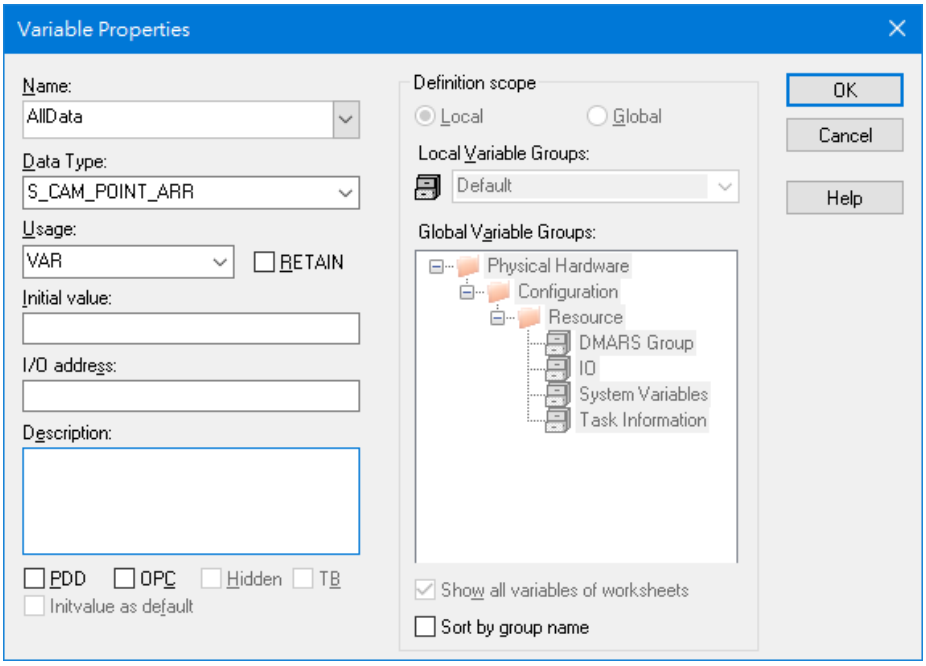


2. ECAM_ReadTableByID 功能块设定如下图：

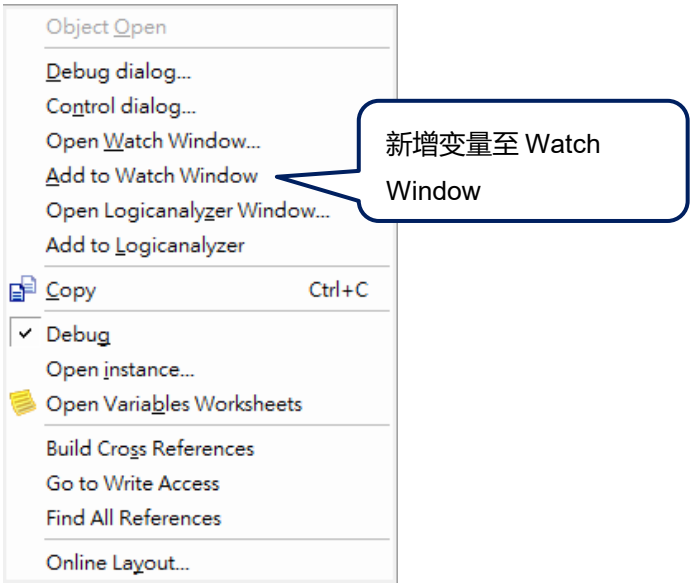


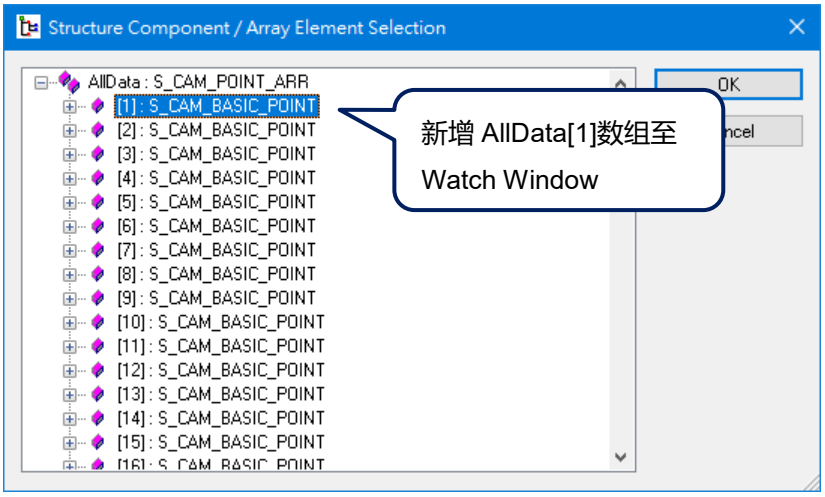
6

3. AllData 变量数据型态为 S_CAM_POINT_ARR

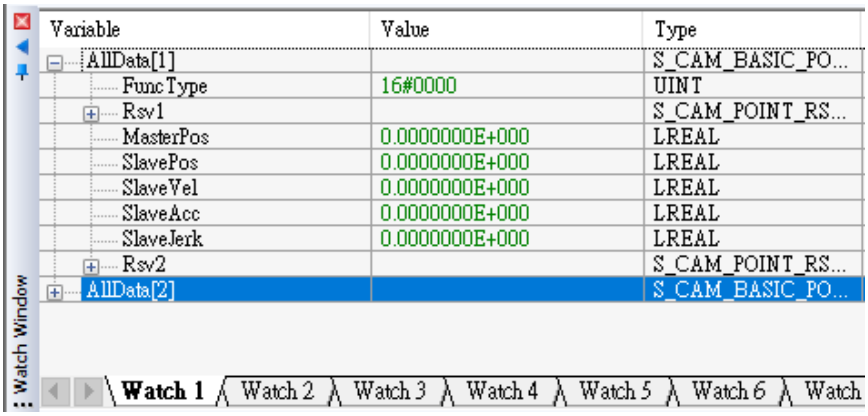


4. 在在线监控模式  中对变量 AllData 按鼠标右键，可以将该变量新增至 Watch Window。

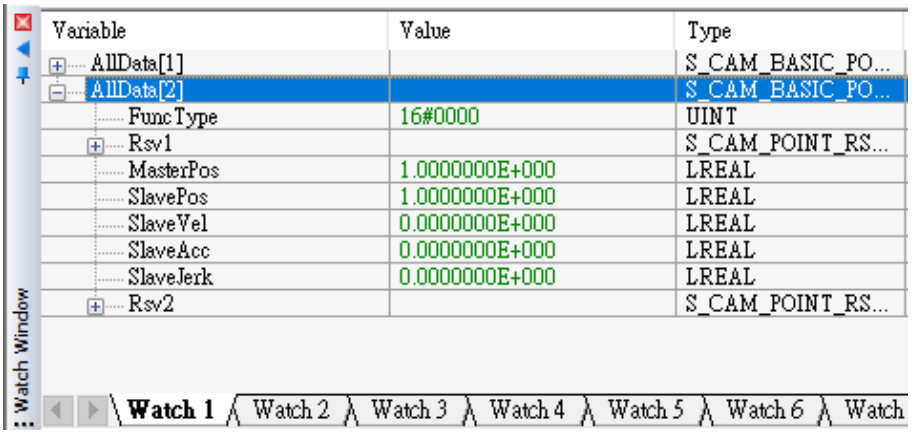




5. 重复步骤 4，以新增 AllData[2]数组至 Watch Window
6. 此时在 Watch Window 中变量 AllData[1]和 AllData[2]填写数据内容皆为 0。



7. 将 Ex210 脚位由 Fasle 设定为 True。
8. ECAM_ReadTableByID 功能块将读取整个凸轮表数据料至变量 AllData。
9. 此功能块执行成功，Done 脚位变为 True，可以透过 Watch Window 中变量 AllData[1]和 AllData[2]来查读取的凸轮表内容。AllData[2]资料如下图所示:



注：凸轮表的最后一点，并不需要曲线函数(FuncType)参数，因此预设为 0。

6.5.8.9 ECAM_SaveTable

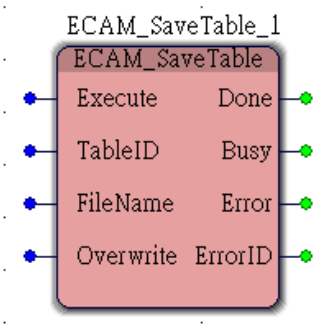
■ 类型

Function Block

■ 功能描述

根据输入凸轮表的 TableID，将编辑过后的凸轮表储存至控制器的断电保持区。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，此功能即开始执行
TableID	UINT	0 ~ 65535 (0)	欲产生的凸轮表编号注
FileName	STRING	字符串	欲储存.bin文件名
Overwrite	BOOL	True / False (False)	TableID存在时，是否覆写 False：不覆写 True：覆写

注：用户如果未输入 TableID，则系统会自动将 TableID 设为 0。

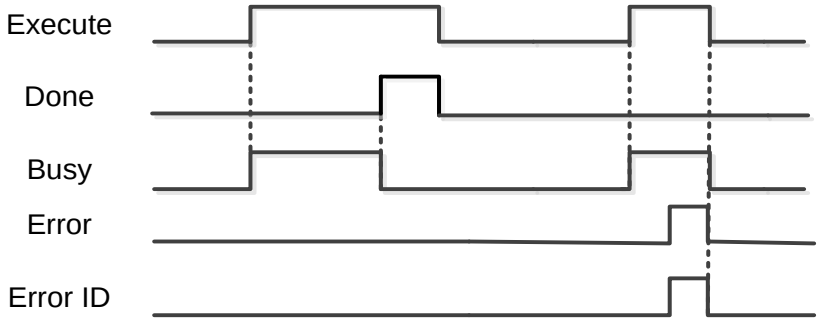
■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	绝对寻址完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 详细说明请参考第9章

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	新增点数据传入系统时	■ Execute转为False时 ■ 若Execute为False而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

■ 输出脚位的变化时序

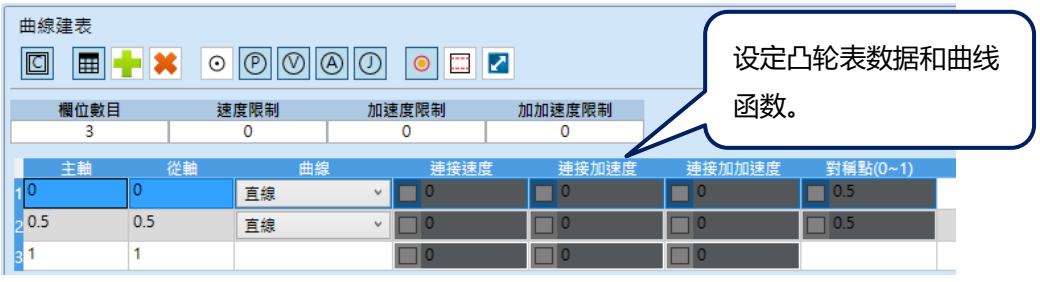


■ 参考范例

此范例说明如何透过 ECAM_ SaveTable 来储存凸轮表至控制器的断电保持区。

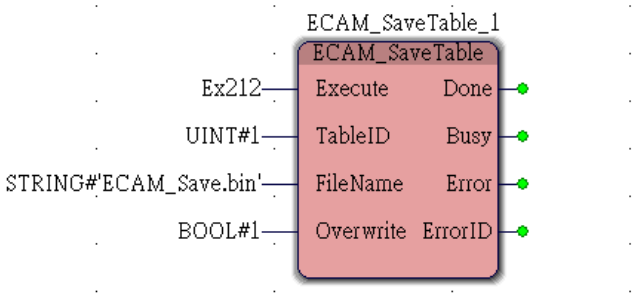
由于凸轮表可以支持在线编辑，若不使用 ECAM_ SaveTable 来储存，则在线编辑后的数据并不会储存于控制器的断电保持区。

1. 先使用其他功能块来建立凸轮表，该凸轮表包含三点信息，详细请参考章节 6.5.8.1。

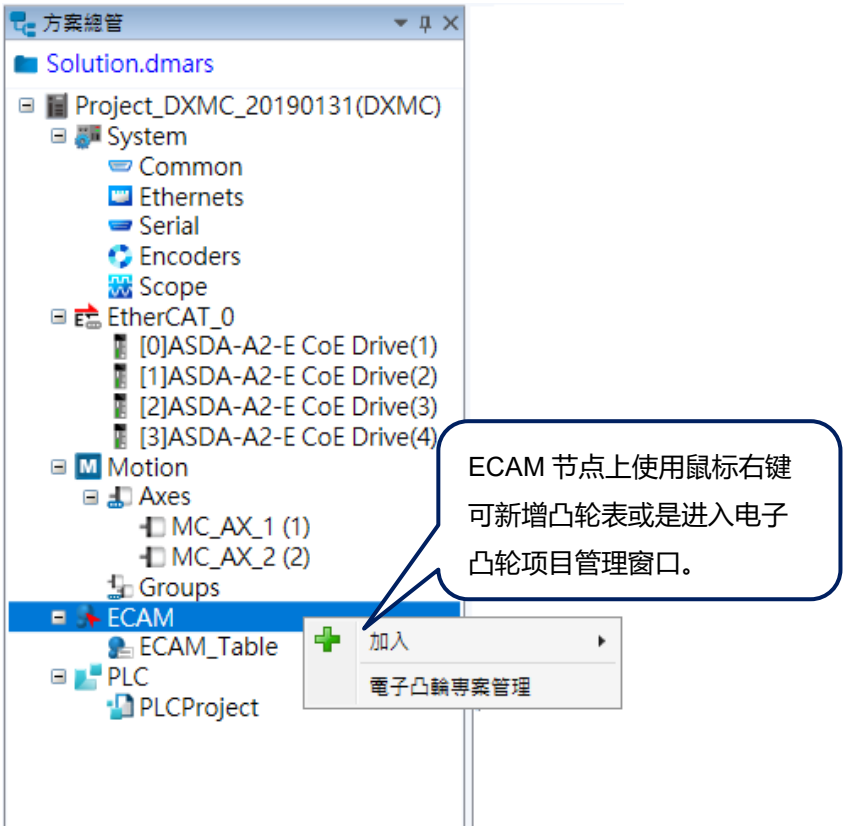


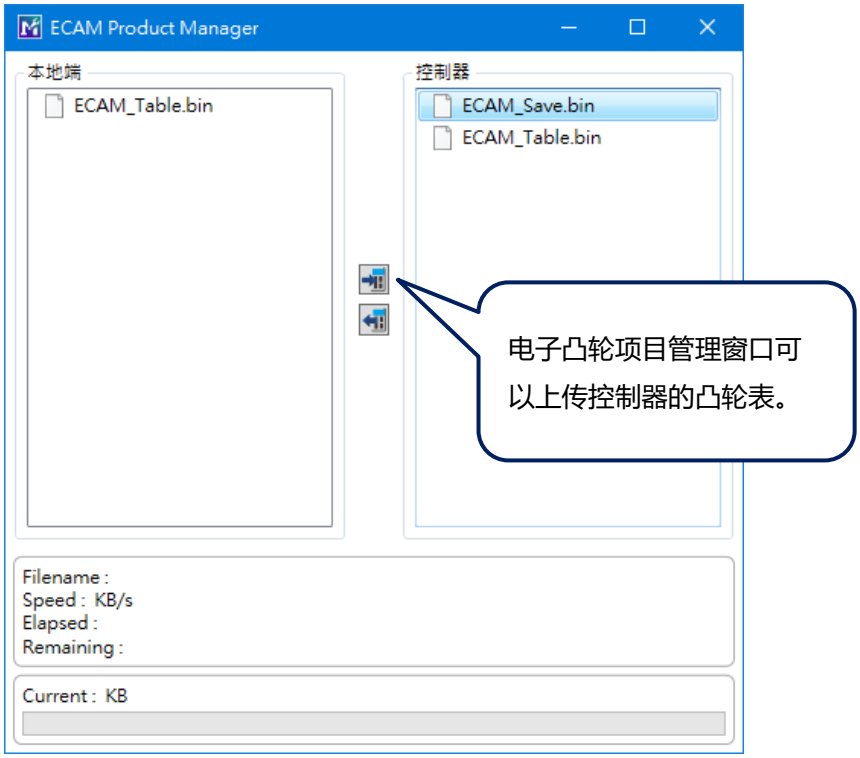
6

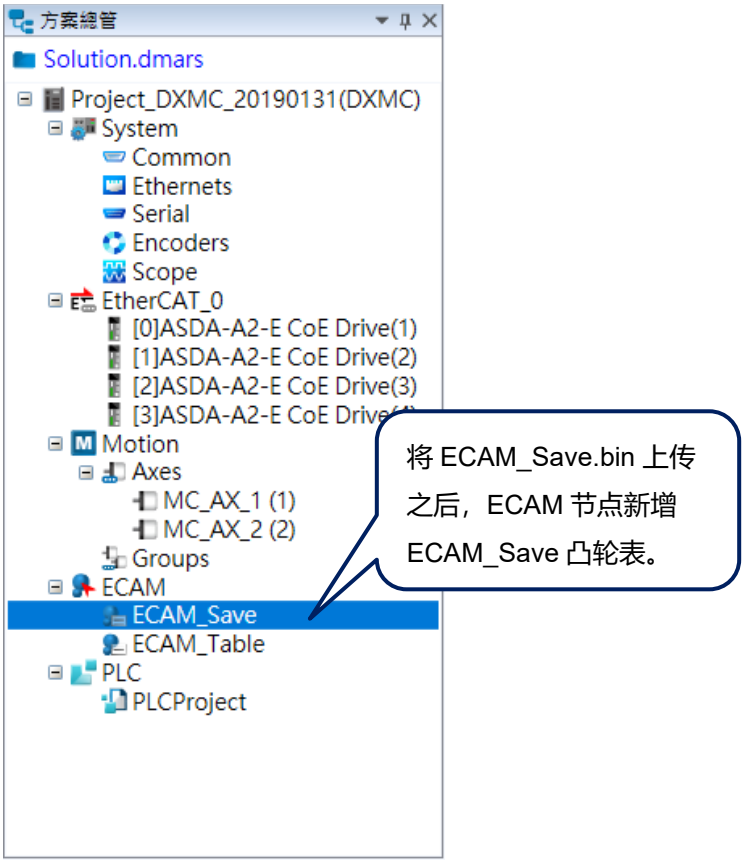
2. ECAM_ SaveTable 功能块设定如下图：凸轮表编号为 1，档名为 ECAM_Save.bin，覆写参数为 1。



3. 将 Ex212 脚位由 Fasle 设定为 True。
4. ECAM_SaveTable 功能块将凸轮表储存至控制器的断电保持区，档名为 ECAM_Save.bin。
5. 此功能块执行成功，Done 脚位变为 True。
6. 欲查看储存后的凸轮表曲线可以在 ECAM 节点右键单击进入「电子凸轮项目管理」窗口，然后将控制器内的凸轮表（ECAM_Save.bin）上传至 DMARS。




ECAM Product Manager
本地端
ECAM_Table.bin
控制器
ECAM_Save.bin
ECAM_Table.bin
电子凸轮项目管理窗口可以上传控制器的凸轮表。
Filename :
Speed : KB/s
Elapsed :
Remaining :
Current : KB


方案總管
Solution.dmars
Project_DXMC_20190131(DXMC)
System
Common
Ethernets
Serial
Encoders
Scope
EtherCAT_0
[0]ASDA-A2-E CoE Drive(1)
[1]ASDA-A2-E CoE Drive(2)
[2]ASDA-A2-E CoE Drive(3)
[3]ASDA-A2-E CoE Drive
Motion
Axes
MC_AX_1 (1)
MC_AX_2 (2)
Groups
ECAM
ECAM_Save
ECAM_Table
PLC
PLCProject
将 ECAM_Save.bin 上传之后，ECAM 节点新增 ECAM_Save 凸轮表。

6.5.8.10 ECAM_GenRotaryCutTable

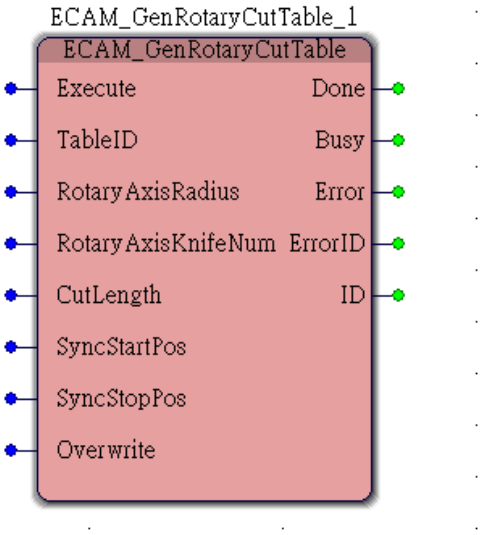
■ 类型

Function Block

■ 功能描述

由使用者输入的参数生成一张飞剪凸轮表。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，此功能即开始执行
TableID	UINT	0 ~ 65535 (0)	要储存的凸轮表编号
RotaryAxis Radius	LREAL	正数或0.0 (0.0)	旋切轴半径
RotaryAxis KnifeNum	UINT	0 ~ 65535 (0)	旋切刀片数目
CutLength	LREAL	正数或0.0 (0.0)	切割长度
SyncStartPos	LREAL	正数或0.0 (0.0)	同步开始位置
Overwrite	LREAL	True / False (False)	TableID存在时，是否覆写 False：不覆写 True：覆写

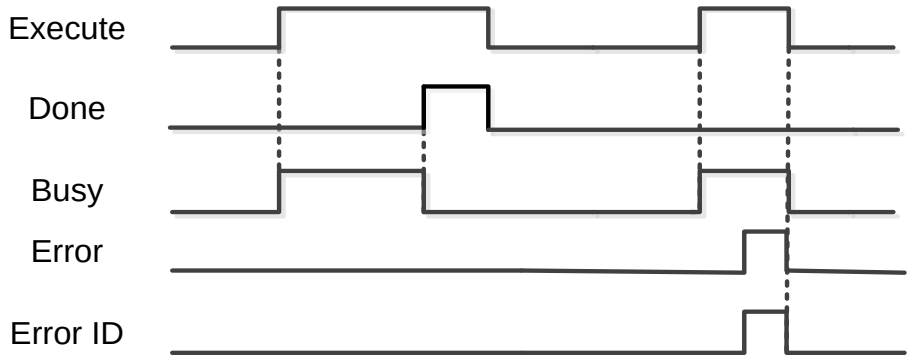
■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	绝对寻址完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF(0)	错误发生时记录的错误码 详细说明请参考第9章
ID	STRING	字符串	生成的凸轮表编号

■ 输出脚位的变化时序

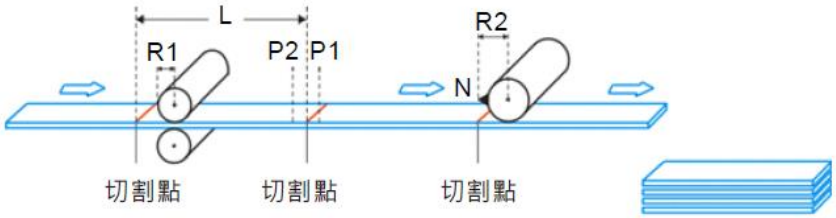
脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	飞剪表完成时	■ Execute转为False时 ■ 若Execute为False而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

■ 输出脚位的变化时序



6

■ 参数说明

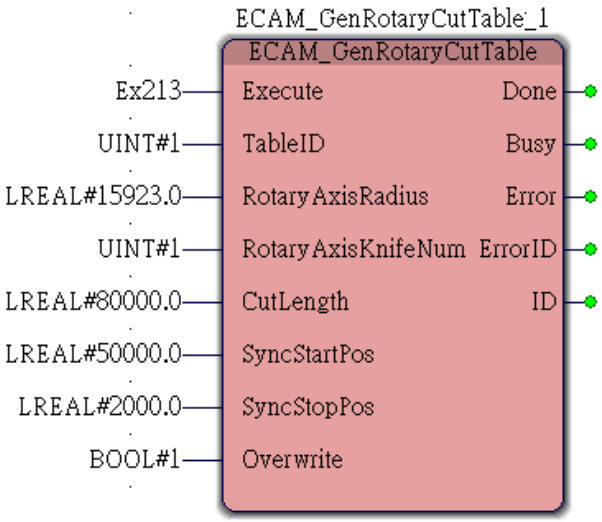


图中参数	说明	脚位名称
N	旋切轴的刀头数，图中的刀头数为 1	RotaryAxisKnifeNum
R2	旋切轴半径，为旋切轴中心到刀尖的距离。(unit：mm)	RotaryAxisRadius
L	材料的切割长度。(unit：mm)	CutLength
P1	同步区开始位置。(unit：mm)	SyncStartPos
P2	同步区结束位置。(unit：mm)	SyncStopPos

■ 参考范例

此范例说明如何透过 ECAM_GenRotaryCutTable 来产生一个飞剪凸轮表。

1. ECAM_GenRotaryCutTable 功能块设定如下图：凸轮表编号为 1、切刀轴半径为 15923 切刀数为 1、材料切割长度为 80000.0、同步区开始位置 50000.0，结束位置为 2000.0。



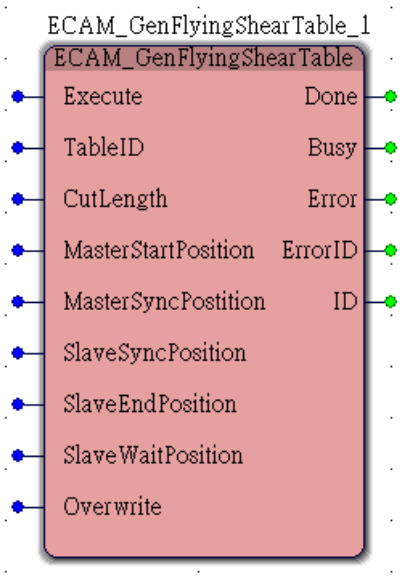
2. 将 Ex213 脚位由 Fasle 设定为 True。
3. 此功能块执行成功时，Done 脚位变为 True，ID 脚位输出 1，ECAM_GenRotaryCutTable 功能块根据参数来产生飞剪凸轮表。

6.5.8.11 ECAM_GenFlyingShearTable

■ 功能描述
Function Block

■ 功能描述
根据输入的参数生成一张追剪凸轮表。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	当Execute为上升缘时，此功能即开始执行
TableID	UINT	0 ~ 65535 (0)	要储存的凸轮表编号
CutLength	LREAL	正数或0.0 (0.0)	切割长度
MasterStart Position	LREAL	正数或0.0 (0.0)	从轴开始追赶主轴的位置
MasterSync Position	LREAL	正数或0.0 (0.0)	主轴同步位置
SlaveSync Position	LREAL	正数或0.0 (0.0)	从轴同步位置
SlaveEnd Position	LREAL	正数或0.0 (0.0)	从轴同步结束位置
SlaveWait Position	LREAL	正数或0.0 (0.0)	从轴周期起始等待位置
Overwrite	BOOL	True / False (False)	TableID存在时，是否覆写 False：不覆写 True：覆写

6

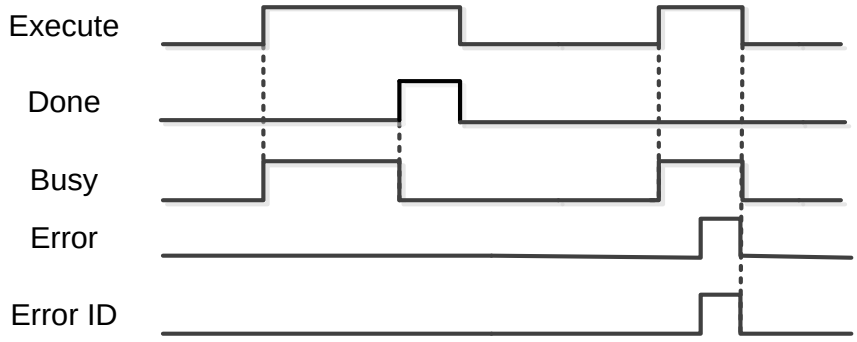
■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	绝对寻址完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指错误发生时记录的错误码 详细说明请参考第9章
ID	STRING	字符串	凸轮表编号

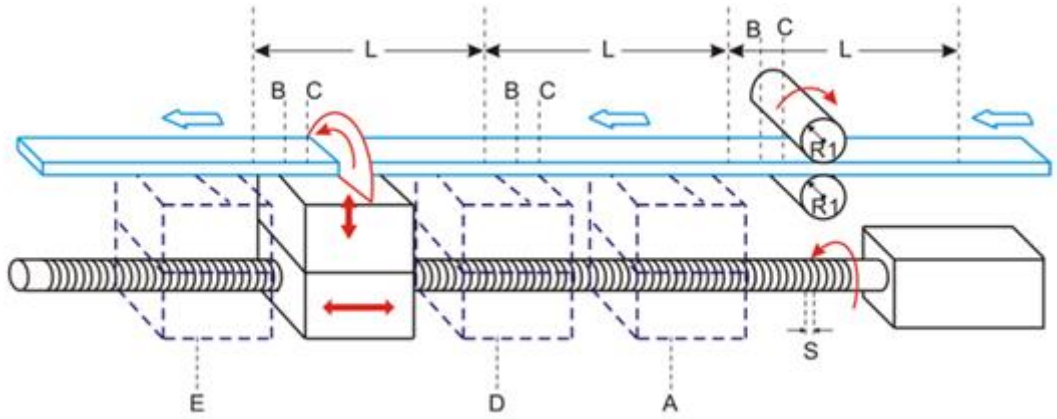
■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	追剪表完成时	■ Execute转为False时 ■ 若Execute为False而Done 转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时 ■ 在CommandAborted上升沿时
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

■ 输出脚位的变化时序



■ 参数说明



图中参数	说明	脚位名称
A	从轴的等待位置。当追剪功能启动后，从轴自动运行到此位置。	SlaveWaitPosition
B	开始位置。当主轴到达此位置时，从轴由等待位置开始开始追赶主轴，以实现速度同步。	MasterStartPosition
C	同步区开始时，对应的主轴位置。	MasterSyncPosition
D	同步区开始时，对应的从轴位置。	SlaveSyncPosition
E	同步区结束时，对应的从轴位置。	SlaveEndPosition
L	材料的切割长度。	CutLength

注:

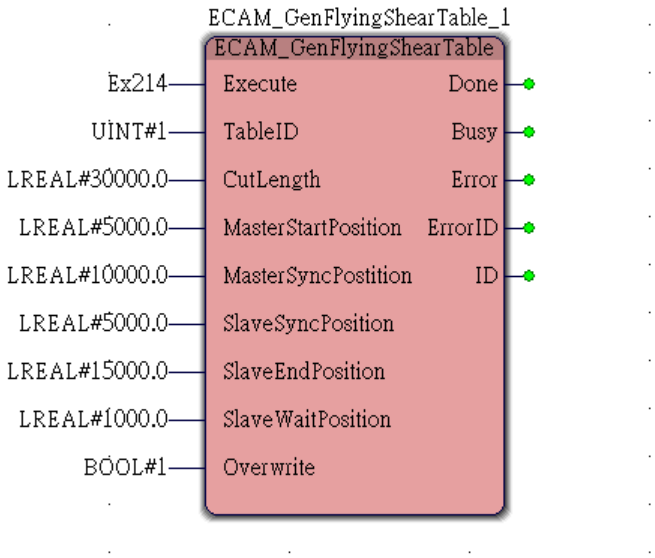
1. R1：主轴半径，即送料轴半径，此数值将从轴参数设定中获得，使用者只需在轴参数中设定各相关机构尺寸即可。
2. R2：从轴半径。采用螺杆时，R2 = 螺杆导程/2π = S/2π，此数值将从轴参数设定中获得，使用者只需在轴参数中设定各相关机构尺寸即可。

6

■ 参考范例

此范例说明如何透过 ECAM_ GenFlyingShearTable 来产生一个追剪凸轮表。

1. ECAM_ GenFlyingShearTable 功能块设定如下图：凸轮表编号为 1，材料长度为 30000.0、开始位置 5000.0、同步区开始时主轴位置 10000.0、同步区开始时主从位置 5000.0，同步结束时从轴位置为 15000.0、从轴等待位置 1000.0。



2. 将 Ex214 脚位由 Fasle 设定为 True。
3. 此功能块执行成功，Done 脚位变为 True，ID 脚位输出 1，ECAM_GenFlyingShearTable 功能块根据参数来产生追剪凸轮表。

6.5.8.12 ECAM_SetCamTappetData

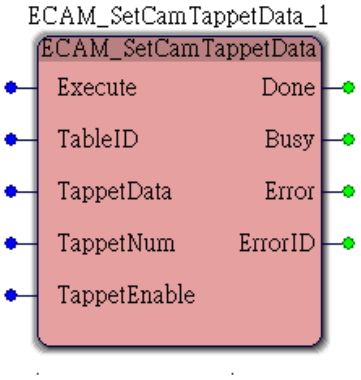
■ 类型

Function Block

■ 功能描述

设定挺杆点数据。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，此功能即开始执行
TableID	UINT	0 ~ 65535 (0)	设定凸轮表编号
TappetData	S_CAM_TAPPET_ARR	-	挺杆点资料注
TappetNum	UINT	0 ~ 65535 (0)	挺杆点数量
TappeEnable	BOOL	True / False (False)	是否生效此挺杆点资料 False：未生效 True：生效

注：S_CAM_TAPPET_ARR 为自定义数据型态，该数组包含 64 个 S_CAM_TAPPET_DATA 结构，详细请参考第 6.5.9 节。

■ 输出脚位

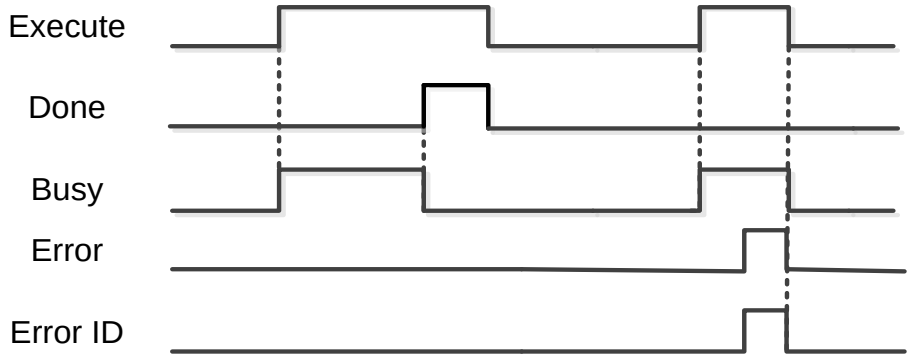
脚位名称	数据型态	设定值(默认值)	功能
Done	BOOL	True / False (False)	绝对寻址完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 详细说明请参考第9章

6

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	当新增点数据传入系统时	■ Execute转为False时 ■ 若Execute为False而Done 转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

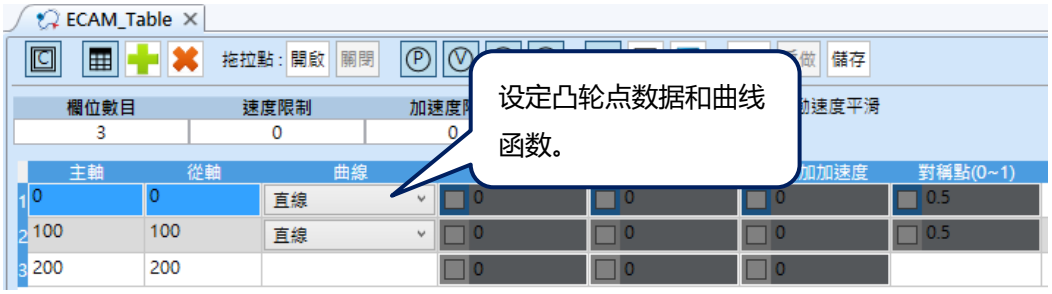
■ 输出脚位的变化时序



■ 参考范例

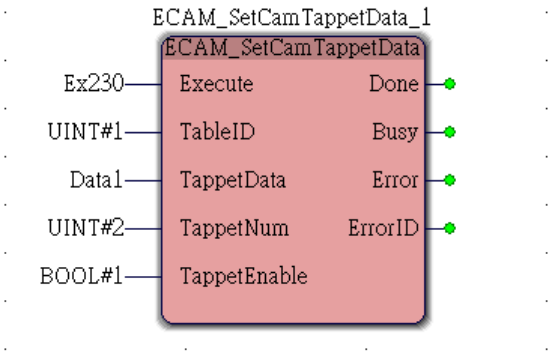
此范例说明如何透过 ECAM_SetCamTappetData 来产生挺杆点，当主轴在正向通过 120 位置时 True，正向通过 150 位置时 False。

1. 输入要设定的凸轮表 ID，宣告挺杆点资料、数量、是否启用。

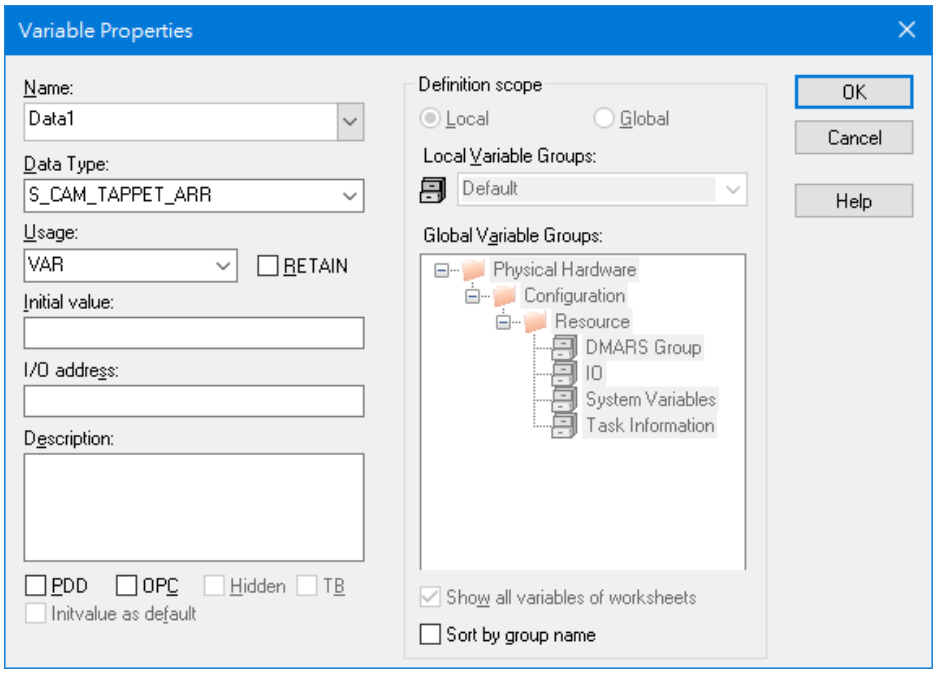


2. 先使用其他功能块来建立凸轮表，该凸轮表包含三点信息，详细请参考章节 6.5.8.1。

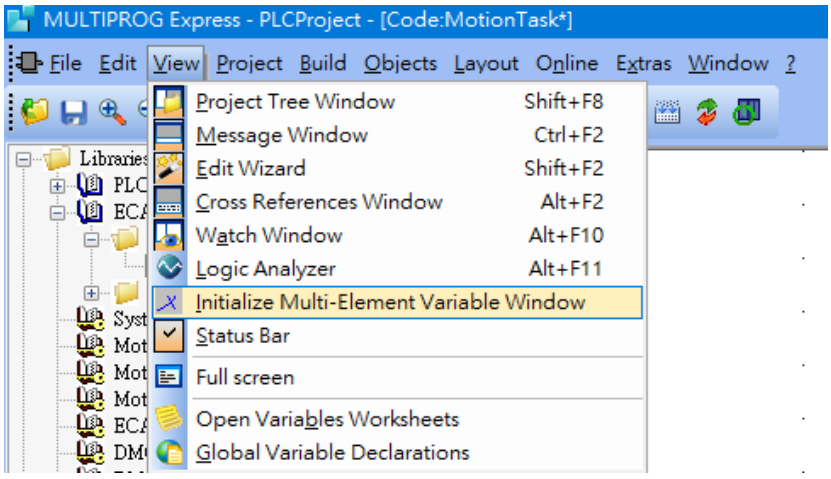
3. ECAM_SetCamTappetData 功能块设定如下图：
   凸轮表编号为 1；挺杆点资料为 Data1；挺杆点数量为 2；挺杆点功能设定为启用。



4. 设定 Data1 挺杆点数据表，Data Type 选择 S_CAM_TAPPET_ARR。

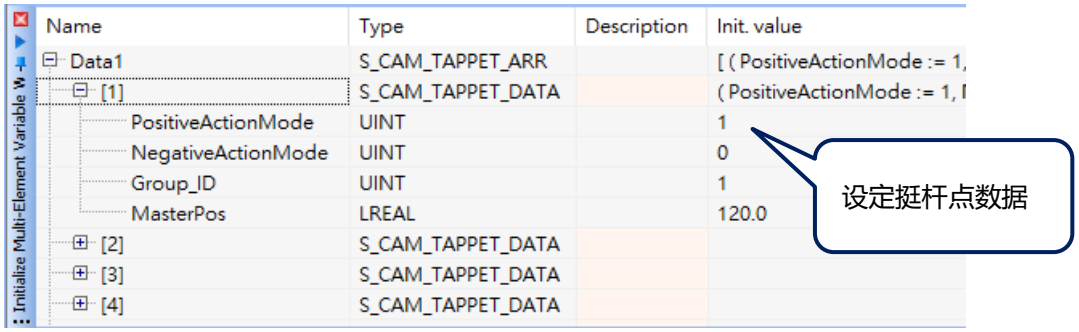


5. 开启 Initialize Muli-Element Variabl Window。

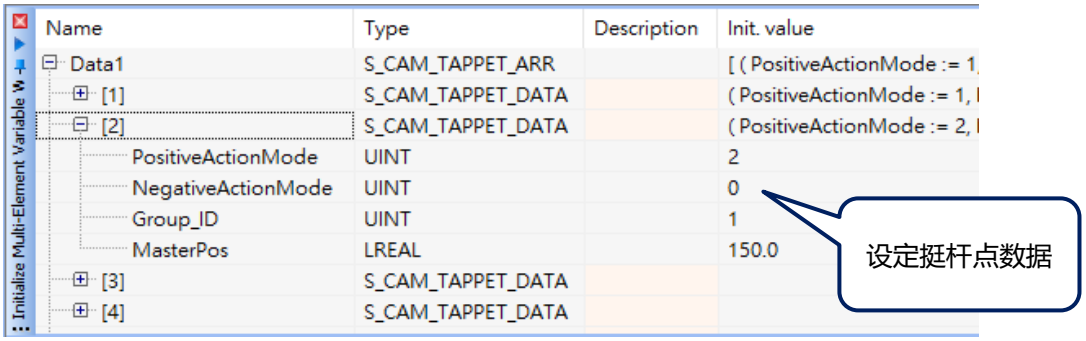


6

6. 于工作区选择 Data1 变量，则可在输入窗口中将 Data1[1] 的挺杆点设定 PositiveActionMode 为 1、NegativeActionMode 为 0、Group_ID 为 1 和 MasterPos 为 120.0。



7. 同上，于输入窗口中将 Data1[2] 的挺杆点设定 PositiveActionMode 为 2、NegativeActionMode 为 0、Group_ID 为 1 和 MasterPos 为 150.0。



8. 触发 Ex230 后，即可将挺杆点数据输入至 DXMC 控制器中。
9. 由于已经启用挺杆点（TappetEnable 设为 1），如欲读取挺杆点状态，请参考 ECAM_GetCamTappetStatus 功能块。

6.5.8.13 ECAM_GetCamTappetStatus

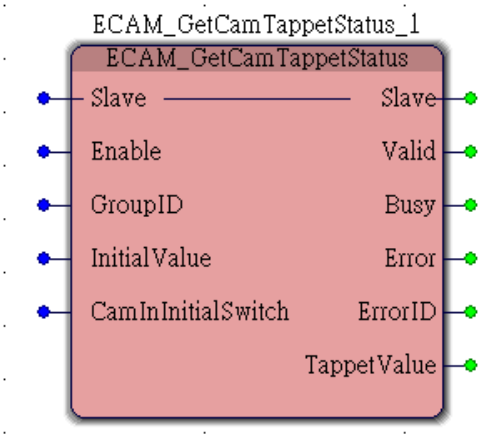
■ 类型

Function Block

■ 功能描述

取得挺杆点状态。(在使用此功能块之前，必须先使用 ECAM_SetCamTappetData 功能块来设定挺杆点数据)

■ 图形表示



■ 输入输出共享脚位

脚位名称	数据型态	设定值 (默认值)	功能
Slave	AXIS_REF	-	指定轴编号

■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Enable	BOOL	True / False (False)	Enable为True时，启用此功能
GroupID	UINT	0 ~ 7 (0)	欲读取之挺杆点群组ID
InitialValue	BOOL	True / False (False)	挺杆点初始值
CamInInitial Switch	BOOL	True / False (False)	下次啮合时是否将目前的状态设为InitialValue False：不设定为Initial Value True：设定Initial Value

6

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Valid	BOOL	True / False (False)	命令完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 错误码详细说明请参考第9章
TappetValue	BOOL	True / False (False)	挺杆点输出状态

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	新增点数据传入系统时	■ Execute转为False时 ■ 若Execute为False而Done 转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

■ 输出脚位的变化时序

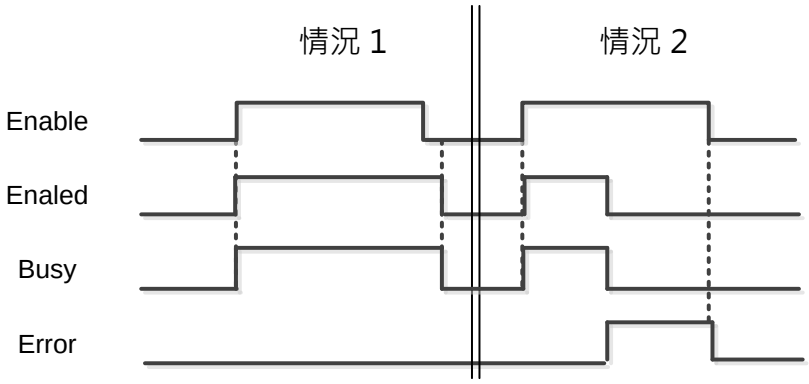


图 6.5.8.12.1 ECAM_GetCamTappetStatus 脚位时序图 (时序详细说明请参考 6.4.3 节)

情况 1：当此指令第一次执行(Enable 由 FALSE 变为 TRUE)时，Busy 与 Enabled 脚位在同一时间变为 TRUE，表示指令已被执行。当 Enable 由 TRUE 变为 FALSE 时，Done 维持一个周期后变为 FALSE。

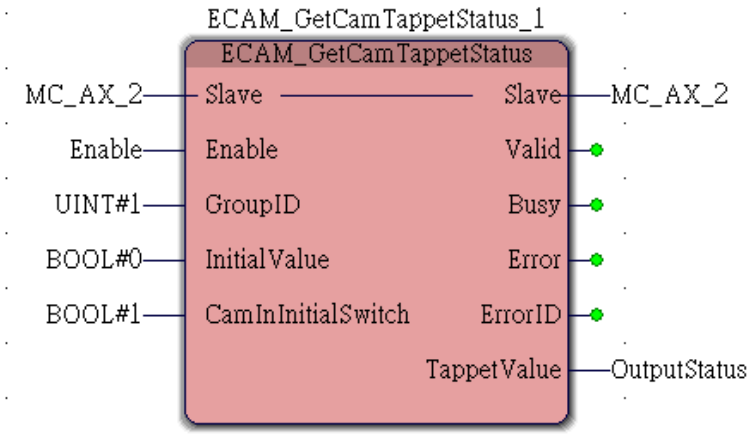
情况 2：当此指令执行的过程中发生错误，则 Error 变为 TRUE，同时 Busy 变为 FALSE，此时请观察 ErrorID 脚位所显示的错误码，了解详细错误讯息。Error 与 ErrorID 脚位将一直维持 TRUE 直到 Enable 脚位变为 FALSE。

■ 参考范例

此范例说明如何透过 ECAM_GetCamTappetStatus 来读取挺杆点状态。

接续章节 6.5.8.11 的范例说明，挺杆点的数据内容为当主轴在正向通过 120 位置时为 True，正向通过 150 位置时为 False。

1. 输入要读取的凸轮表，其为 GroupID 为 1，初始值设定为 0，CamInInitialSwitch 为 1。



2. 当执行 CamIn 之后且 Enable 脚位为 True，则 OutputStatus 状态会变成 InitialValue 的设定值，也就是 0。
3. 假设 CamIn 之后，主轴位置为 0，当主轴往正方向持续运动，经过 120 位置时，则 OutputStatus 会变成 True，经过 150 位置时，则状态会变成 False。

6.5.8.14 ECAM_SetConnectVelocity

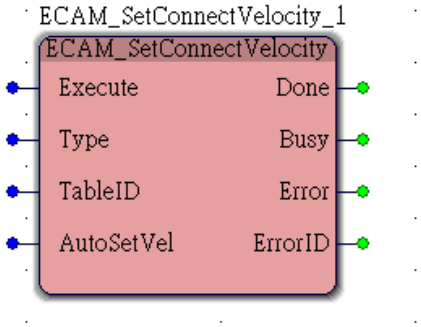
■ 类型

Function Block

■ 功能描述

根据输入凸轮表的 TableID，设定是否自动速度平滑凸轮表，开启 AutoSetVel 功能之后，凸轮表会自动填入连接速度，让凸轮在移动的过程中，速度曲线不会减速至 0。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，此功能即开始执行
Type	UINT	0 ~ 2 (0)	0：使用三次曲线平滑 1：使用五次曲线平滑 2：使用七次曲线平滑
TableID	UINT	0 ~ 65535 (0)	欲设定的凸轮表ID
AutoSetVel	BOOL	True / False (False)	设定是否执行自动平滑功能 False：不设定自动平滑功能 True：设定自动平滑功能

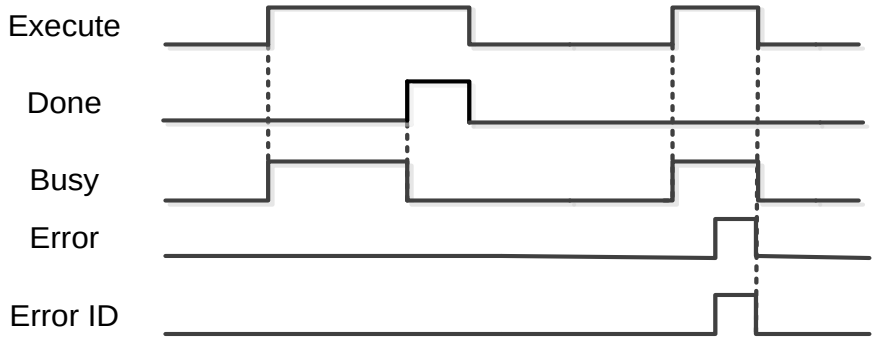
■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	设定完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF(0)	指令错误发生时记录的错误码 详细说明请参考第9章

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	参数设定完成时	■ 当Execute转为False时 ■ 若Execute为False而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

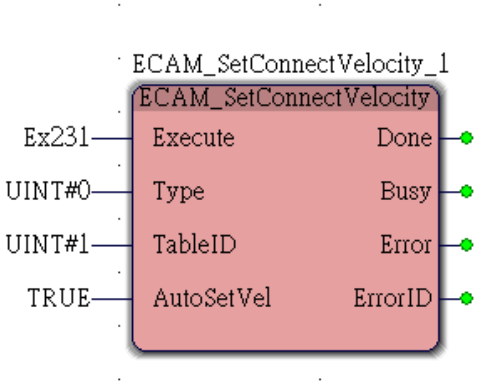
■ 输出脚位的变化时序



■ 参考范例

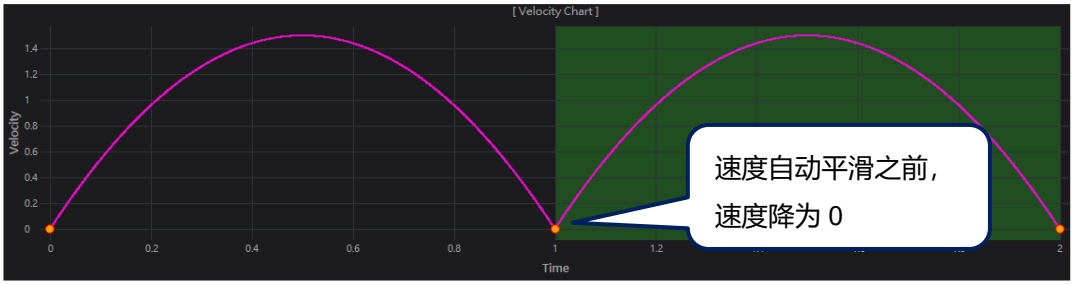
此范例说明如何透过 ECAM_SetConnectVelocity 来开启自动速度平滑功能。

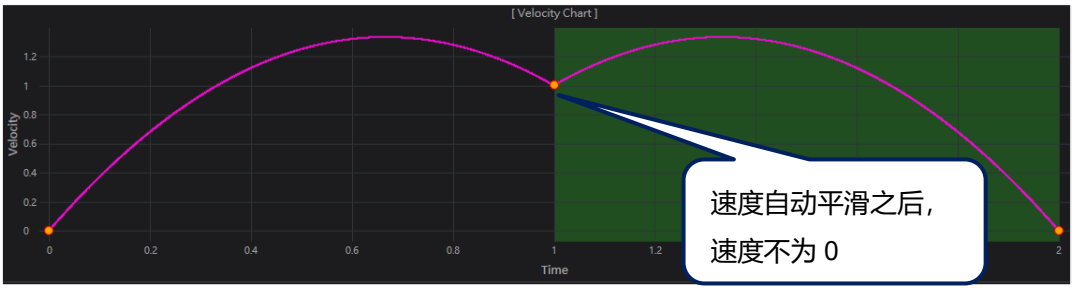
1. 先使用其他功能块例如:ECAM_GenTableByData 等方式，将已经产生的表加载至控制器 RAM 中，并设 TableID 为 1。
2. 设定 ECAM_SetConnectVelocity，使用三次曲线平滑（Type 设为 0），并将 AutoSetVel 设为 TRUE，并触发 Ex231 之后，即可将此凸轮表速度平滑化。



6

3. 下自动速度平滑之前和之后的速度曲线图，如图所示：





4. 之后使用 ECAM_AddPoints、ECAM_DelPoints、ECAM_ModifyPoints 等功能会自动补上连接速度，并且根据用户设定的 Type 来计算。
5. 如欲取消自动计算连接速度，只需将 AutoSetVel 设为 FALSE，并再次 Execute 此功能块。

6.5.8.15 ECAM_PreviewTable

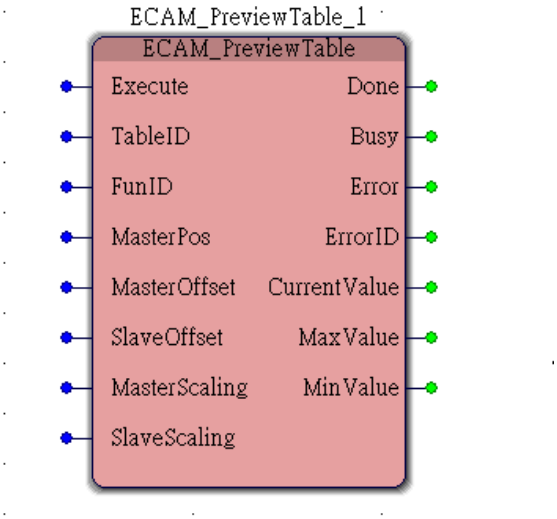
■ 类型

Function Block

■ 功能描述

预览凸轮表的各项信息。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	当Execute为上升缘时，此功能即开始执行
TableID	UINT	0 ~ 65535 (0)	欲设定的凸轮表ID
FunID	UINT	0 ~ 3 (0)	0：预览位置 1：预览速度 2：预览加速度 3：预览凸轮曲线长度
MasterPos	LREAL	正数或0.0 (0.0)	指定预览的主轴位置
MasterOffset	LREAL	正数或0.0 (0.0)	指定预览凸轮表格中主轴的偏移量
SlaveOffset	LREAL	正数或0.0 (0.0)	指定预览凸轮表格中从轴的偏移量
MasterScaling	LREAL	正数或0.0 (1.0)	指定预览凸轮表格中主轴的缩放比例
SlaveScaling	LREAL	正数0.0 (1.0)	指定预览凸轮表格中从轴的缩放比例

6

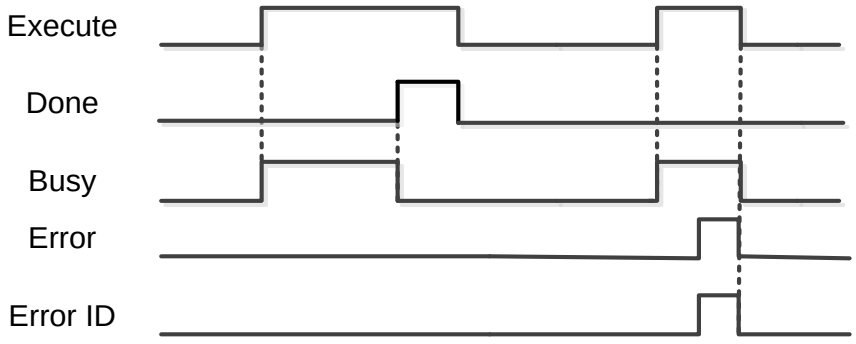
■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	设定完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 错误码详细说明请参考第9章
CurrentValue	LREAL	正数或0.0 (0.0)	当主轴位置在MasterPos的位置时的从轴信息
MaxValue	LREAL	正数或0.0 (0.0)	凸轮表格中从轴信息的最大值
MintValue	LREAL	正数或0.0 (0.0)	凸轮表格中从轴信息的最小值

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	参数设定完成时	■ Execute转为False时 ■ 若Execute为False而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

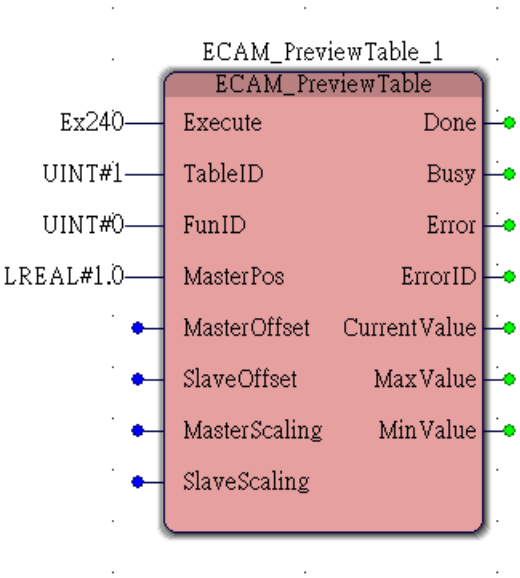
■ 输出脚位的变化时序



■ 参考范例

此范例说明如何透过 ECAM_PreviewTable 来预览凸轮表的特定信息。

1. 先使用其他功能块例如：ECAM_GenTableByData 等方式，将已经产生的表加载至控制器 RAM 中，并设 TableID 为 1。
2. 设定 ECAM_PreviewTable，读取 TableID 为 1，预览当主轴为位置 1.0 的时候 (MasterPos 设为 1.0)，从轴的位置 (FunID 设为 0)，触发 Ex240 之后，即可预览此凸轮表信息。



3. CurrentValue 代表当主轴位置在 1.0 时从轴的相对应位置。
4. MaxValue 代表当主从轴啮合之后，从轴的最大位置。
5. MinValue 代表当主从轴啮合之后，从轴的最小位置。

6.5.8.16 ECAM_ModRotCutVelRatio

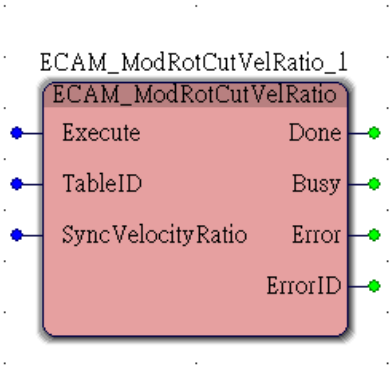
■ 类型

Function Block

■ 功能描述

调整飞剪凸轮表中的同步速度比例。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False(False)	Execute为上升缘时，此功能即开始执行
TableID	UINT	0 ~ 65535 (0)	欲设定的凸轮表ID
SyncVelocity Ratio	LREAL	正数或0.0 (0.0)	指定同步区速度比例

■ 输出脚位

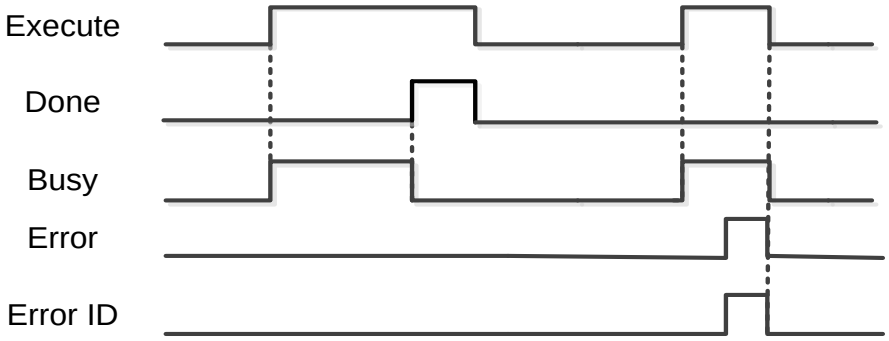
脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	当设定完成时为True
Busy	BOOL	True / False (False)	当指令被触发执行时为True
Error	BOOL	True / False(False)	当指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF(0)	指令错误发生时记录的错误码详细说明请参考第9章

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	参数设定完成时	■ Execute转为False时 ■ 若Execute为False而Done 转为True，此时Done 维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ 在Done上升沿时 ■ 在Error上升沿时

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

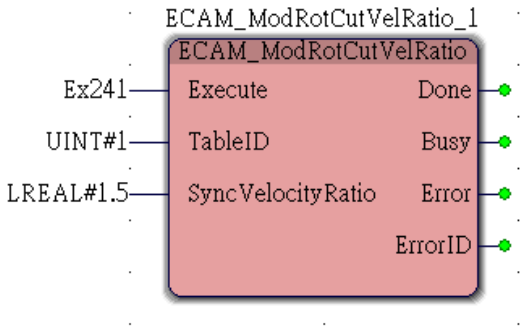
■ 输出脚位的变化时序



■ 参考范例

此范例说明如何透过 ECAM_ ModRotCutVelRatio 来将凸轮表中删除特定点信息。

1. 先使用其他功能块例如：ECAM_GenTableByData 等方式，将已经产生的表加载至控制器 RAM 中，并设 TableID 为 1。
2. 设定 ECAM_ ModRotCutVelRatio，读取 TableID 为 1，速度比例设为 1.5 倍 (SyncVelocityRatio 设为 1.5)。



3. 触发 Ex241 之后，从轴在同步区时的速度比例为主轴速度的 1.5 倍。

6.5.8.17 ECAM_SetOnlineChgMode

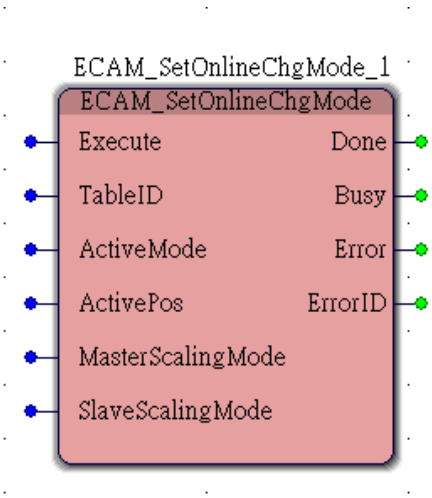
■ 类型

Function Block

■ 功能描述

设定变更凸轮表生效时机。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
Execute	BOOL	True / False (False)	Execute为上升缘时，此功能即开始执行
TableID	UINT	0 ~ 65535 (0)	欲设定的凸轮表ID
ActiveMode	UINT	0 ~ 4 (0)	设定更换凸轮表的生效时机 0：立即运行 1：主轴绝对位置。 当主轴位置到达ActivePos指定位置时开始运行。 2：主轴绝对相位。 将ActivePos对应到主轴CAM表格中的位置，当主轴运行到此位置时开始运行。 3：当凸轮走到下个周期开始运行。 4：不运行。
ActivePos	LREAL	正数或0.0 (0.0)	调整主从轴啮合时的主轴位置
MasterScaling Mode	UINT	-	保留
SlaveScaling Mode	UINT	-	保留

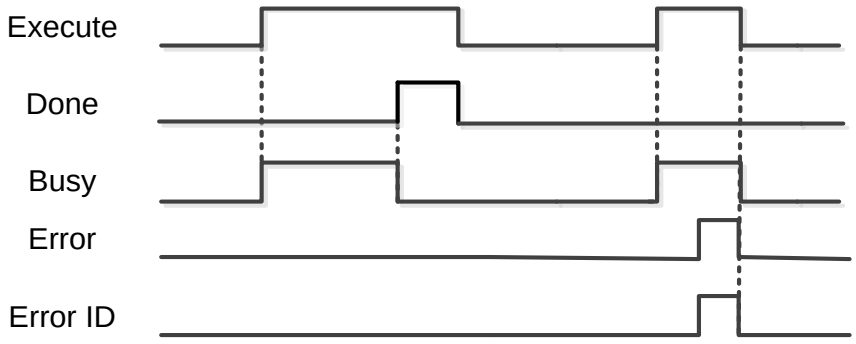
■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
Done	BOOL	True / False (False)	设定完成时为True
Busy	BOOL	True / False (False)	指令被触发执行时为True
Error	BOOL	True / False (False)	指令错误发生时为True
ErrorID	WORD	0x0000 ~ 0xFFFF (0)	指令错误发生时记录的错误码 详细说明请参考第9章

■ 输出脚位的变化时序

脚位名称	输出脚位上升沿时机	输出脚位下降沿时机
Done	参数设定完成时	■ Execute转为False时 ■ 若Execute为False而Done转为True，此时Done维持一个扫描周期的True状态后，立刻转为False
Busy	Execute上升沿触发时	■ Done上升沿时 ■ Error上升沿时
Error / ErrorID	指令的执行条件或输入值发生错误时 (错误码记录在ErrorID)	在Execute下降沿时 (清除ErrorID记录之错误码)

■ 输出脚位的变化时序

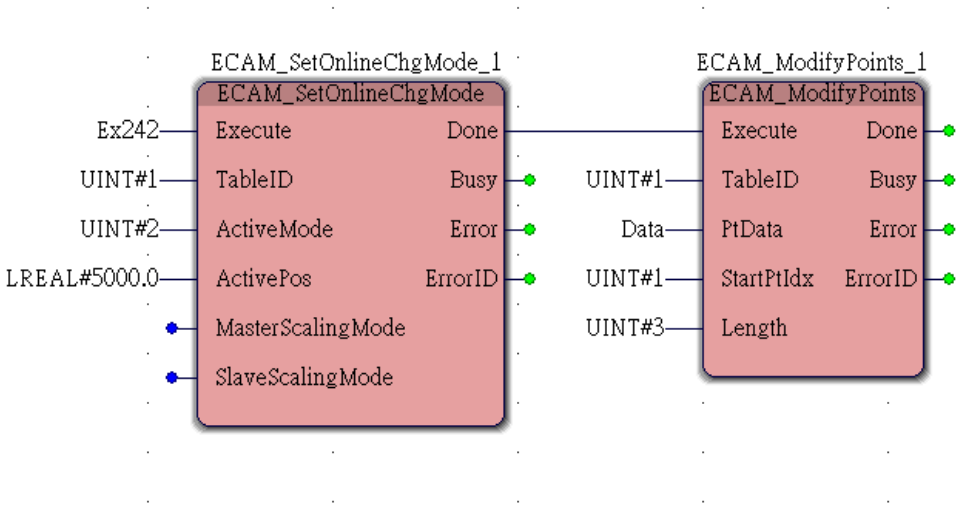


6

■ 参考范例

此范例说明如何透过 ECAM_SetOnlineChgMode 来指定变更凸轮表的生效时机。

1. 先使用其他功能块例如:ECAM_GenTableByData 等方式，将已经产生的表加载至控制器 RAM 中，并设 TableID 为 1。
2. 设定 ECAM_SetOnlineChgMode，读取 TableID 为 1，切换模式设为 2 (ActiveMode = 2 主轴绝对位置模式)，ActivePos 设为 5000.0。



3. 触发 Ex242 之后，从轴并不会立刻执行换表动作，直到主轴绝对位置到达 5000.0 时才会执行换表。

6.5.9 自定义数据型态

6.5.9.1 自定义数据型态总览

在 MULTIPROG 中，某些功能块的脚位为 ANY 型态，此为必填脚位，用户须输入对应的数据结构，不填写则会造成编译错误。本小节将介绍台达所提供的自定义数据型态。

类型	数据结构名称	说明
Task Information	Task_Info_eCLR	PLC Task信息
	Extended_Task_Info	PLC Task延伸信息
PLCOpenDataType	AXIS_REF	轴信息
	MC_TRIGGER_REF	设定触发事件
	GROUP_REF	群组信息
	S_GRP_POS	群组位置、姿态与机械臂组态数据结构
ECAMEditorDataType	S_CAM_BASIC_POINT	电子凸轮表格点位
	S_CAM_POINT_ARR	电子凸轮表格点位数组

6.5.9.2 任务信息数据型态（Task Information)

■ Task_Info_eCLR 数据结构成员说明：

成员名称	数据型态	说明
TaskStack	INT	显示任务大小
TaskPrio	INT	任务的优先级
TaskPeriod_us	DINT	任务周期 (单位：微秒)
TaskWatchdog_us	DINT	Watchdog 周期 (单位：微秒)
TaskPeriod	INT	任务周期 (单位：毫秒)
TaskWatchdog	INT	Watchdog 周期 (单位：毫秒)
MinDuration_us	DINT	任务最短持续时间 (单位：微秒)
MaxDuration_us	DINT	任务最长持续时间 (单位：微秒)
CurDuration_us	DINT	任务目前持续时间 (单位：微秒)
MinDelay_us	DINT	任务最短延迟时间 (单位：微秒)
MaxDelay_us	DINT	任务最长延迟时间 (单位：微秒)
CurDelay_us	DINT	任务目前延迟时间 (单位：微秒)
MinDuration	INT	任务最短持续时间 (单位：毫秒)
MaxDuration	INT	任务最长持续时间 (单位：毫秒)
CurDuration	INT	任务目前持续时间 (单位：毫秒)
MinDelay	INT	任务最短延迟时间 (单位：毫秒)
MaxDelay	INT	任务最长延迟时间 (单位：毫秒)
CurDelay	INT	任务目前延迟时间 (单位：毫秒)
unused_1	DINT	-
unused_2	DINT	-

6

成员名称	数据型态	说明
unused_3	DINT	-
unused_4	DINT	-
unused_5	DINT	-
unused_6	DINT	-
unused_7	DINT	-
unused_8	DINT	-
unused_9	DINT	-
TaskName	STRING (35)	任务名称

6.5.9.3 PLCOpen 数据型态 (PLCOpenDataType)

■ AXIS_REF 数据结构成员说明:

成员名称	数据型态	说明
Enable	BOOL	轴是否配置成功
Ready	BOOL	轴是否处于 StandStill 状态
Number	UINT	轴编号
StServoOn	BOOL	轴是否已使能
Fault	BOOL	是否有错误发生
StQuickStop	BOOL	是否处于紧急停止状态
Warning	BOOL	是否有警告发生
TargetReach	BOOL	是否到达目标位置
SwitchHome	BOOL	回原开关 (*暂无作用)
PLimit	BOOL	是否到达硬件正极限 (*暂无作用)
NLimit	BOOL	是否到达硬件负极限 (*暂无作用)
SPLimit	BOOL	是否到达软件正极限
SNLimit	BOOL	是否到达软件负极限
VelLimit	BOOL	是否到达速度极限
TrqLimit	BOOL	是否到达扭矩极限
DirPositive	BOOL	是否正向移动
DirNegative	BOOL	是否反向移动
HomeDefine	BOOL	是否已完成回原动作
InHomePosition	BOOL	是否到达原点
Capture1	BOOL	Capture1 状态 (*暂无功能)
Capture2	BOOL	Capture2 状态 (*暂无功能)
StHomeProcess	BOOL	是否处于回原状态
StModeSelect	SINT	轴运行状态
PosTarget	LREAL	目标位置 (单位：用户单位)
PosCmd	LREAL	命令位置 (单位：用户单位)
VelCmd	LREAL	命令速度 (单位：用户单位/秒)
AccDecCmd	LREAL	命令加、减速

成员名称	数据型态	说明
		(单位：用户单位/秒 ^{2})
TorqueCmd	LREAL	命令扭矩 (单位：额定扭矩%)
PosFb	LREAL	回授位置 (单位：用户单位)
PosErr	LREAL	位置误差 (单位：用户单位)
VelFb	LREAL	回授速度 (单位：用户单位/秒)
AccDecFb	LREAL	回授加、减速 (单位：用户单位/秒 ^{2})
TorqueFb	LREAL	回授扭矩 (单位：额定扭矩%)
ErrorID	DWORD	错误码；错误码列表请参考第 9 章
WarningID	DWORD	警告码；警告码列表请参考第 9 章
Coordinated	BOOL	是否经过坐标系转换
State	USINT	显示轴 PLCopen 定义状态 0：DISABLED 1：STANDSTILL 2：ERROR_STOP 3：STOPPING 4：HOMING 5：DISCRETE_MOTION 6：CONTINUOUS_MOTION 7：SYNCHRONIZED_MOTION
RSVD	ARRAY OF SINT[14]	保留

■ MC_TRIGGER_REF 数据结构成员说明：

成员名称	数据型态	说明
TriggerMode	USINT	设定触发模式 0：驱动器模式 (Drive Mode) 1：控制器模式 (Controller Mode)
TriggerID	USINT	指定触发事件编号 0 ~ 1：驱动器模式 (Drive Mode) 2 ~ 5：控制器模式 (Controller Mode)
TriggerType	USINT	设定触发型态 0：Built-in DI (控制器/伺服本体 DI) 1：Local DI (右侧扩充模块 DI) 2：Remote DI (EtherCAT 模块 DI) 3：PLC variable
DINumber	UINT	指定触发事件所使用的 DI 编号 格式：AUZYX (十进制) AUZ：编号 (DI 模块 Node ID) YX：次编号 (DI 端口编号)

6

成员名称	数据型态	说明
RisingTrigger	BOOL	若为 True，触发事件为上升沿触发，反之为下降沿触发
RecordedData	USINT	指定纪录数据源 00：Drive Capture Pos Cmd 01：Drive Capture Pos Fb 02：Drive Capture AUX ENC 03：Controller Capture Pos Fb 10：Pos Cmd 11：Pos Fb PUU 12：PosTarget PUU 13：Vel Cmd PUU 14：Vel Fb PUU 15：AccDec Cmd PUU 16：AccDec Fb PUU
RSVD	ARRAY OF SINT[14]	保留

■ GROUP_REF 数据结构成员说明：

成员名称	数据型态	说明
Enable	BOOL	群组是否配置成功
Ready	BOOL	群组是否处于 GroupStandby 状态
InPosition	BOOL	群组是否到达目标位置
State	USINT	显示群组 PLCopen 定义状态 0：GRPDISABLED 1：GRPSTANDBY 2：GRPERROR_STOP 3：GRPSTOPPING 4：GRPHOMING 5：GRPMOVING
Number	UINT	群组编号
RobotModel	WORD	显示群组机器手臂型态
ErrorID	DWORD	错误码；错误码列表请参考第 9 章
WarningID	DWORD	警告码，警告码列表请参考第 9 章
PosTarget	S_GRP_POS	群组目标位置、姿态及机械臂组态信息，详细内容请参考 S_GRP_POS 数据结构成员说明表
PosCmd	S_GRP_POS	群组命令位置、姿态及机械臂组态信息，详细内容请参考 S_GRP_POS 数据结构成员说明表

成员名称	数据型态	说明
PosFb	S_GRP_POS	群组回授位置、姿态及机械臂组态信息，详细内容请参考 S_GRP_POS 数据结构成员说明表
VelCmd	LREAL	群组命令速度
VelFb	LREAL	群组回授速度
AccDecCmd	LREAL	群组命令加、减速
AccDecFb	LREAL	群组回授加、减速
RSVD	ARRAY [1..8] OF SINT	保留

■ S_GRP_POS 数据结构成员说明:

成员名称	数据型态	说明
X/J1	LREAL	机器手臂末端点 X 轴坐标 / 群组 J1 位置
Y/J2	LREAL	机器手臂末端点 Y 轴坐标 / 群组 J2 位置
Z/J3	LREAL	机器手臂末端点 Z 轴坐标 / 群组 J3 位置
A/J4	LREAL	机器手臂末端点 X 轴旋转坐标 / 群组 J4 位置
B/J5	LREAL	机器手臂末端点 Y 轴旋转坐标 / 群组 J5 位置
C/J6	LREAL	机器手臂末端点 Z 轴旋转坐标 / 群组 J6 位置
E1/J7	LREAL	机器手臂末端点 E1 轴坐标 / 群组 J7 位置
E2/J8	LREAL	机器手臂末端点 E2 轴坐标 / 群组 J8 位置
E3/J9	LREAL	机器手臂末端点 E3 轴坐标 / 群组 J9 位置
E4/J10	LREAL	机器手臂末端点 E4 轴坐标/ 群组 J10 位置
E5/J11	LREAL	机器手臂末端点 E5 轴坐标 / 群组 J11 位置
E6/J12	LREAL	机器手臂末端点 E6 轴坐标/ 群组 J12 位置
M1	SINT	-
M2	SINT	-
M3	SINT	-
M4	SINT	-
M5	SINT	-
M6	SINT	-
M7	SINT	-
M8	SINT	-
M9	SINT	-
M10	SINT	-
M11	SINT	-
M12	SINT	-
Shoulder	BOOL	机器手臂组态信息
Elbow	BOOL	机器手臂组态信息
Flip	BOOL	机器手臂组态信息
Pose	BOOL	机器手臂组态信息
UF	USINT	使用者坐标系编号
TF	USINT	工具坐标系编号
Coord	USINT	群组坐标系种类

成员名称	数据型态	说明
RSVD	ARRAY OF SINT[5]	保留

6

■ S_MC_TRANSITION_PARAMETER_REF 数据结构成员说明：

重迭模式	成员名称	数据型态	说明
None	数组成员 1	LREAL	无意义
	数组成员 2	LREAL	无意义
	数组成员 3	LREAL	无意义
	数组成员 4	LREAL	无意义
CornerDistance	数组成员 1	LREAL	转角距离（单位：pulse)
	数组成员 2	LREAL	无意义
	数组成员 3	LREAL	无意义
	数组成员 4	LREAL	无意义
Immediate	数组成员 1	LREAL	无意义
	数组成员 2	LREAL	无意义
	数组成员 3	LREAL	无意义
	数组成员 4	LREAL	无意义

6.5.9.4 ECAM 数据型态 (ECAMEditorDataType)

■ S_CAM_BASIC_POINT 数据结构成员说明：

成员名称	数据型态	说明
FuncType	UINT	曲线函数
Rsv1	ARRAY [1..3] OF UINT	保留
MasterPos	LREAL	主轴位置
SlavePos	LREAL	从轴位置
SlaveVel	LREAL	从轴速度
SlaveAcc	LREAL	从轴加速度
SlaveJerk	LREAL	从轴加加速度
Rsv2	ARRAY [1..4] OF REAL	保留

■ S_CAM_POINT_ARR 数据结构成员说明：

成员名称	数据型态	说明
S_CAM_POINT_ARR	ARRAY [1..10000] OF S_CAM_BASIC_POINT	包含 10,000 个 S_CAM_BASIC_POINT 数据结构

■ S_CAM_TAPPET_DATA 数据结构成员说明：

成员名称	数据型态	说明
PositiveActionMode	UINT	凸轮主轴正向经过指定位置时的状态模式。 0：不动作 1：正向经过 MasterPos 位置，将状态设为 True 2：正向经过 MasterPos位置，将状态设为False 3：正向经过 MasterPos 位置，将状态设为反向 4：正向经过 MasterPos位置将状态设为True，且维持一个PLC Task 周期

成员名称	数据型态	说明
NegativeActionMode	UINT	凸轮主轴反向经过指定位置时的状态模式。 0：不动作 1：反向经过 MasterPos 位置，将状态设为 True 2：反向经过 MasterPos 位置，将状态设为 False 3：反向经过 MasterPos 位置，将状态设为反向 4：反向经过 MasterPos 位置，将状态设为 True 且维持一个 PLC Task 周期
Group_ID	UINT	欲设定的群组 ID，输入范围：0 ~ 7
MasterPos	LREAL	当凸轮主轴经过此凸轮位置时，根据 PositiveActionMode、NegativeActionMode 触发状态做变化。 同一个群组 ID 最多输入 8 个挺杆点。

■ S_CAM_TAPPET_ARR 数据结构成员说明：

成员名称	数据型态	说明
S_CAM_TAPPET_ARR	ARRAY [1..64] OF S_CAM_TAPPET_DATA	包含 64 个 S_CAM_TAPPET_DATA 数据结构

6.6 标准功能

6.6.1 标准功能总览

分类	指令名称	说明
数学运算功能	ABS	取整数或浮点数的绝对值。
	ACOS	计算反余弦值。
	ASIN	计算反正弦值。
	ATAN	计算反正切值。
	COS	计算余弦值。
	EXP	计算自然指数(natural exponential)。
	LN	计算自然对数。
	LOG	计算以10为底的对数。
	SIN	计算正弦值。
	SQRT	计算平方根。
	TAN	计算正切值。
	ADD	求两数值之和。
	ADD_T_T	求两个时间数据之和。
	DIV	除法运算。
	DIV_T_AI	时间除以整数的运算。
	DIV_T_AN	时间除以整数或实数的运算。
	DIV_T_R	时间除以实数的运算。
	EXPT	求幂运算。
	MOD	取余运算。
	MOVE	数据搬移。
	MUL	乘法运算。
	MUL_T_AI	时间数值乘以一个整数数值的运算。
	MUL_T_AN	时间乘以整数或实数的运算。
	MUL_T_R	时间乘以实数运算。
	NEG	求输入值之负数。
	SUB	减法运算。
	SUB_T_T	时间减法运算。
位串功能	AND	位AND运算。
	NOT	位NOT运算。
	OR	位OR运算。
	XOR	位XOR运算。

位移功能	ROL	位向左旋转位移。
	ROR	位向右旋转位移。
	SHL	位向左位移。
	SHR	位向右位移。
选择运算功能	LIMIT	输出限制。
	MAX	求最大值。
	MIN	求最小值。
	SEL	二选一功能。
比较运算功能	EQ	等于判断。
	GE	大于等于判断。
	GT	大于判断
	LE	小于等于判断。
	LT	小于判断。
	NE	不等于判断。
字符串操作功能	CONCAT	字符串连接。
	DELETE	删除指定长度的字符串。
	EQ_STRING	字符串相等判断。
	FIND	字符串查找。
	GE_STRING	英文字符顺序大于等于判断。
	GT_STRING	英文字符顺序大于判断。
	INSERT	字符串插入。
	LE_STRING	英文字符顺序小于等于判断。
	LEFT	取字符串中由左往右连续个数字符。
	LEN	计算字符串的字符个数。
	LT_STRING	英文字符顺序小于判断。
	MID	撷取自串中一指定长度的字符串。
	NE_STRING	英文字符顺序不等于判断。
	REPLACE	字符串取代。
	RIGHT	取字符串中由右往左连续个数字符。

6.6.2 指令启用说明 (EN 和 ENO 说明)

当用户使用的指令带有 EN 和 ENO 时，其运作如下：

- 若 EN 接脚的参数值为 FALSE (0)，则指令定义的功能将不会被执行，且指令的输出接脚输出值不会被刷新。
- 若指令定义的 EN 接脚参数值为 TRUE (1)，则指令定义的功能将会被执行，且指令输出接脚的值也会被刷新。
- ENO 接脚输出和 EN 接脚的输入保持一致，EN 接脚为 TRUE，ENO 接脚同时变为 TRUE；EN 接脚为 FALSE，ENO 接脚同时变为 FALSE。

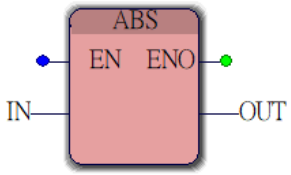
当指令为功能块(FB)时，如果功能块(FB)执行后，EN 由 TRUE 变为 FALSE，该功能块(FB)仍继续执行，只是该功能块(FB)输出接脚的值不会被刷新。

6

6.6.3 数学运算功能

6.6.3.1 ABS

- 类型
  Function
- 功能描述
  取整数或浮点数的绝对值。
- 图形表示



- 输入脚位

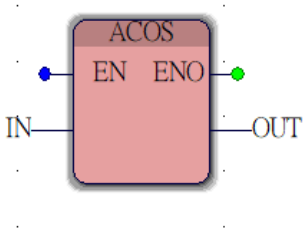
脚位名称	数据型态	设定值 (默认值)	功能
IN	ANY_NUM	正数、负数或0.0 (0.0)	任意型态数值输入脚位

- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	ANY_NUM	正数、负数或0.0 (0.0)	取得计算后绝对数值输出脚位

6.6.3.2 ACOS

- 类型
  Function
- 功能描述
  计算反余弦值。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN	ANY_REAL	正数、负数或0.0 (0.0)	任意数值输入脚位

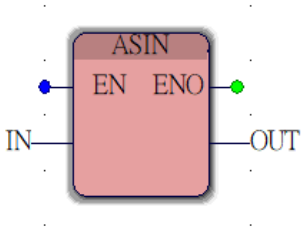
- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	ANY_REAL	正数、负数或0.0 (0.0)	取得计算后之数值输出脚位 (单位:弧度)

6

6.6.3.3 ASIN

- 类型
  Function
- 功能描述
  计算反正弦值。
- 图形表示



- 输入脚位

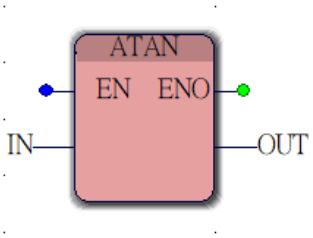
脚位名称	数据型态	设定值 (默认值)	功能
IN	ANY_REAL	正数、负数或0.0 (0.0)	任意数值输入脚位

- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	ANY_REAL	正数、负数或0.0 (0.0)	取得计算后之数值输出脚位 (单位：弧度)

6.6.3.4 ATAN

- 类型
  Function
- 功能描述
  计算反正切值。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN	ANY_REAL	正数、负数或0.0 (0.0)	任意数值输入脚位

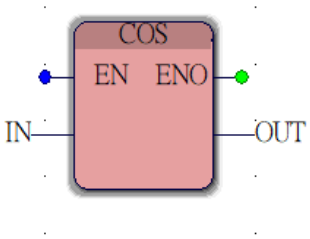
- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	ANY_REAL	正数、负数或0.0 (0.0)	取得计算后之数值输出脚位 (单位：弧度)

6

6.6.3.5 COS

- 类型
  Function
- 功能描述
  计算余弦值。
- 图形表示



- 输入脚位

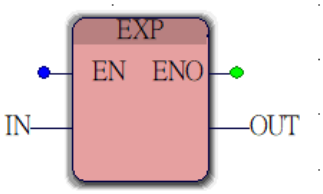
脚位名称	数据型态	设定值 (默认值)	功能
IN	ANY_REAL	正数、负数或0.0 (0.0)	任意数值输入脚位(单位:弧度)

- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	ANY_REAL	正数、负数或0.0 (0.0)	取得计算后之数值输出脚位

6.6.3.6 EXP

- 类型
  Function
- 功能描述
  计算自然指数(natural exponential)。
- 图形表示



- 输入脚位

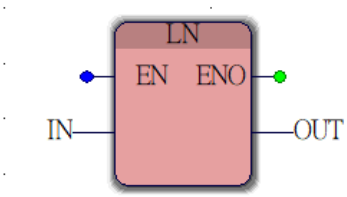
脚位名称	数据型态	设定值 (默认值)	功能
IN	ANY_REAL	正数、负数或0.0 (0.0)	以e为底的指数

- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	ANY_REAL	正数、负数或0.0 (0.0)	取得EXP计算后之结果数值

6.6.3.7 LN

- 类型
  Function
- 功能描述
  计算自然对数(Natural logarithm)。
- 图形表示



- 输入脚位

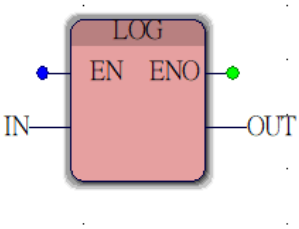
脚位名称	数据型态	设定值 (默认值)	功能
IN	ANY_REAL	正数、负数或0.0 (0.0)	欲计算之数值

- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	ANY_REAL	正数、负数或0.0 (0.0)	经LN功能块计算后之结果

6.6.3.8 LOG

- 类型
  Function
- 功能描述
  计算以 10 为底的对数。
- 图形表示



- 输入脚位

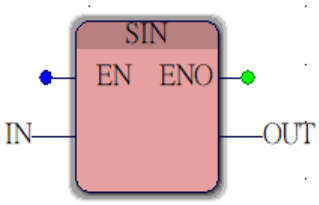
脚位名称	数据型态	设定值 (默认值)	功能
IN	ANY_REAL	正数、负数或 0.0 (0.0)	欲计算之数值

- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	ANY_REAL	正数、负数或 0.0 (0.0)	经LOG功能块计算后之结果

6.6.3.9 SIN

- 类型
  Function
- 功能描述
  计算正弦值。
- 图形表示



- 输入脚位

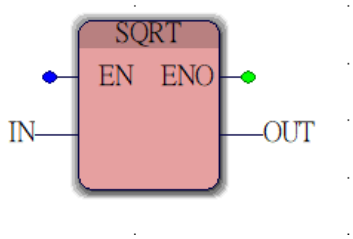
脚位名称	数据型态	设定值 (默认值)	功能
IN	ANY_REAL	正数、负数或0.0 (0.0)	欲计算之数值 (单位：弧度)

- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	ANY_REAL	正数、负数或0.0 (0.0)	经SIN功能块计算后之结果

6.6.3.10 SQRT

- 类型
  Function
- 功能描述
  计算平方根。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN	ANY_REAL	正数、负数或0.0 (0.0)	欲计算之数值

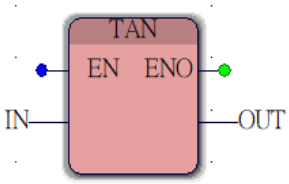
- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	ANY_REAL	正数、负数或0.0 (0.0)	经SQRT功能块计算后之结果

6

6.6.3.11 TAN

- 类型
  Function
- 功能描述
  计算正切值。
- 图形表示



- 输入脚位

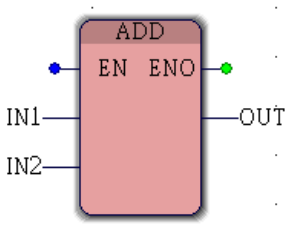
脚位名称	数据型态	设定值 (默认值)	功能
IN	ANY_REAL	正数、负数或0.0 (0.0)	欲计算之数值 (单位：弧度)

- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	ANY_REAL	正数、负数或0.0 (0.0)	经TAN功能块计算后之结果

6.6.3.12 ADD

- 类型
  Function
- 功能描述
  求两数值之和。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN1	ANY_NUM	正数、负数或0.0 (0.0)	被加数
IN2	ANY_NUM	正数、负数或0.0 (0.0)	被加数

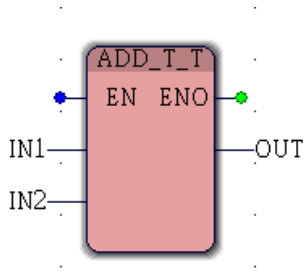
- 输出脚位

脚位名称	数据型态	设定值(默认值)	功能
OUT	ANY_NUM	正数、负数或0.0 (0.0)	IN1加IN2之结果

6.6.3.13 ADD_T_T

6

- 类型
  Function
- 功能描述
  求两个时间数据之和。
- 图形表示



- 输入脚位

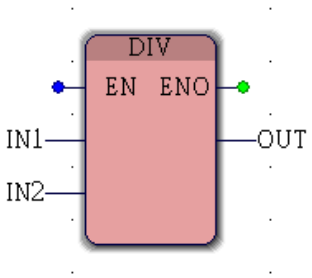
脚位名称	数据型态	设定值 (默认值)	功能
IN1	TIME	正数或0.0 (0.0)	时间数据
IN2	TIME	正数或0.0 (0.0)	时间数据

- 输出脚位

脚位名称	数据型态	设定值(默认值)	功能
OUT	TIME	正数或0.0 (0.0)	IN1与IN2时间之和

6.6.3.14 DIV

- 类型
  Function
- 功能描述
  除法运算。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN1	ANY_NUM	正数、负数或0.0 (0.0)	被除数
IN2	ANY_NUM	正数、负数或0.0 (0.0)	除数

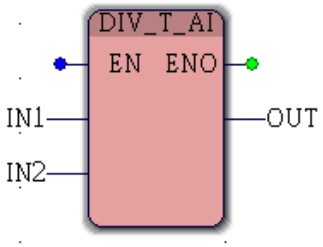
- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	ANY_NUM	正数、负数或0.0 (0.0)	IN1除以IN2之结果

6.6.3.15 DIV_T_AI

6

- 类型
  Function
- 功能描述
  时间除以整数的运算。
- 图形表示



- 输入脚位

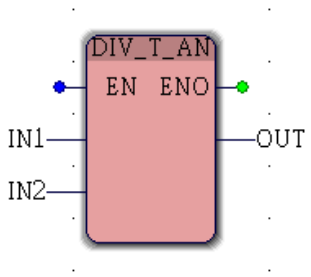
脚位名称	数据型态	设定值(默认值)	功能
IN1	TIME	正数或0.0 (0.0)	被除数 (时间)
IN2	ANY_INT	正数、负数或0.0 (0.0)	除数 (整数)

- 输出脚位

脚位名称	数据型态	设定值(默认值)	功能
OUT	TIME	正数或0.0 (0.0)	IN1除以IN2之结果

6.6.3.16 DIV_T_AN

- 类型
  Function
- 功能描述
  时间除以整数或实数的运算。
- 图形表示



- 输入脚位

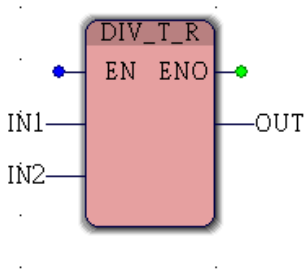
脚位名称	数据型态	设定值 (默认值)	功能
IN1	TIME	正数或0.0 (0.0)	被除数 (时间)
IN2	ANY_NUM	正数、负数或0.0 (0.0)	除数 (整数或实数)

- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	TIME	正数或0.0 (0.0)	IN1除以IN2之结果

6.6.3.17 DIV_T_R

- 类型
  Function
- 功能描述
  时间除以实数的运算。
- 图形表示



- 输入脚位

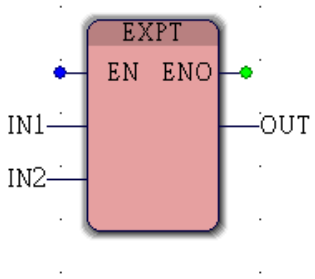
脚位名称	数据型态	设定值(默认值)	功能
IN1	TIME	正数、负数或0.0 (0.0)	被除数 (时间)
IN2	REAL	正数、负数或0.0 (0.0)	除数 (实数)

- 输出脚位

脚位名称	数据型态	设定值(默认值)	功能
OUT	TIME	正数或0.0 (0.0)	IN1除以IN2之结果

6.6.3.18 EXPT

- 类型
  Function
- 功能描述
  求幂运算。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN1	ANY_REAL	正数、负数或0.0 (0.0)	基底数
IN2	ANY_NUM	正数、负数或0.0 (0.0)	指数

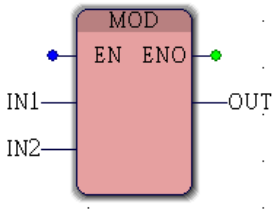
- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	ANY_REAL	正数、负数或0.0 (0.0)	以IN1为基底、IN2为指数之求幂结果

6.6.3.19 MOD

6

- 类型
  Function
- 功能描述
  取余运算。
- 图形表示



- 输入脚位

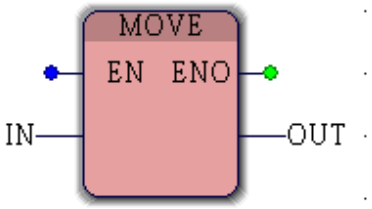
脚位名称	数据型态	设定值 (默认值)	功能
IN1	ANY_INT	正数、负数或0.0 (0.0)	被除数
IN2	ANY_INT	正数、负数或0.0 (0.0)	除数

- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	ANY_INT	正数、负数或0.0 (0.0)	余数

6.6.3.20 MOVE

- 类型
  Function
- 功能描述
  数据搬移。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN	ANY	正数、负数或0.0 (0.0)	欲搬移之数值

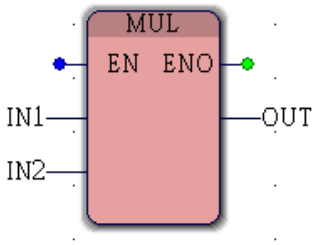
- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	ANY	正数、负数或0.0 (0.0)	被覆盖后之数值

6.6.3.21 MUL

6

- 类型
  Function
- 功能描述
  乘法运算。
- 图形表示



- 输入脚位

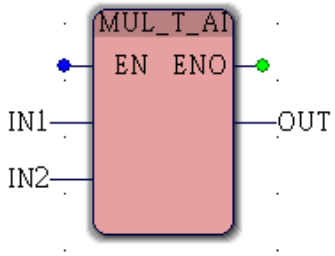
脚位名称	数据型态	设定值 (默认值)	功能
IN1	ANY_NUM	正数、负数或0.0 (0.0)	输入数值
IN2	ANY_NUM	正数、负数或0.0 (0.0)	输入数值

- 输出脚位

脚位名称	数据型态	设定值(默认值)	功能
OUT	ANY_NUM	正数、负数或0.0 (0.0)	IN1乘以IN2之结果

6.6.3.22 MUL_T_AI

- 类型
  Function
- 功能描述
  时间数值乘以一个整数数值的运算。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN1	TIME	正数或0.0 (0.0)	输入数值 (时间)
IN2	ANY_INT	正数、负数或0.0 (0.0)	输入数值 (整数)

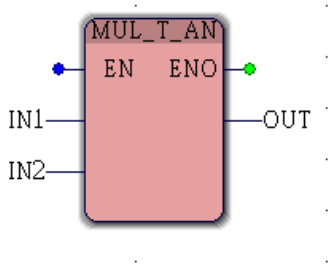
- 输出脚位

脚位名称	数据型态	设定值(默认值)	功能
OUT	TIME	正数或0.0 (0.0)	IN1乘以IN2之结果

6.6.3.23 MUL_T_AN

6

- 类型
  Function
- 功能描述
  时间乘以整数或实数的运算。
- 图形表示



- 输入脚位

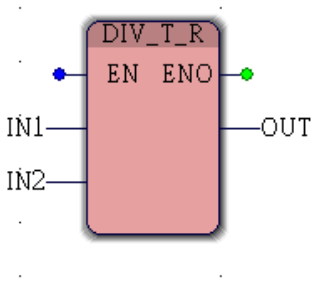
脚位名称	数据型态	设定值 (默认值)	功能
IN1	TIME	正数或0.0 (0.0)	输入数值 (时间)
IN2	ANY_NUM	正数、负数或0.0 (0.0)	输入数值 (整数或实数)

- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	TIME	正数或0.0 (0.0)	IN1乘以IN2之结果

6.6.3.24 MUL_T_R

- 类型
  Function
- 功能描述
  时间乘以实数运算。
- 图形表示



- 输入脚位

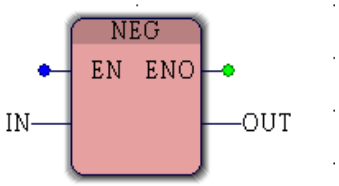
脚位名称	数据型态	设定值 (默认值)	功能
IN1	TIME	正数或0.0 (0.0)	输入数值 (时间)
IN2	REAL	正数、负数或0.0 (0.0)	输入数值 (实数)

- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	TIME	正数或0.0 (0.0)	IN1乘以IN2之结果

6.6.3.25 NEG

- 类型
  Function
- 功能描述
  求输入值之负数。
- 图形表示



- 输入脚位

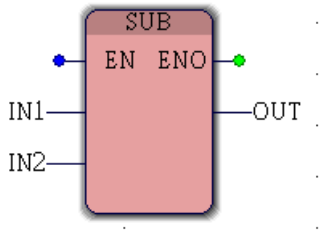
脚位名称	数据型态	设定值 (默认值)	功能
IN	ANY_NUM	正数、负数或0.0 (0.0)	欲转换之输入值

- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	ANY_NUM	负数或0.0 (0.0)	经NEG功能块转换后之数值

6.6.3.26 SUB

- 类型
  Function
- 功能描述
  减法运算。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN1	ANY_NUM	正数、负数或0.0 (0.0)	被减数
IN2	ANY_NUM	正数、负数或0.0 (0.0)	减数

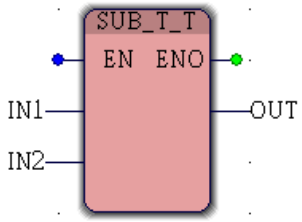
- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	ANY_NUM	正数、负数或0.0 (0.0)	IN1与IN2之差值

6

6.6.3.27 SUB_T_T

- 类型
  Function
- 功能描述
  时间减法运算。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN1	TIME	正数或0.0 (0.0)	被减数 (时间)
IN2	TIME	正数或0.0 (0.0)	减数 (时间)

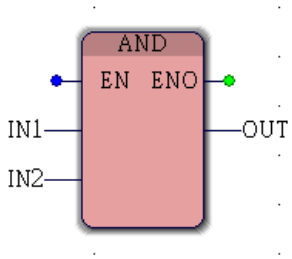
- 输出脚位

脚位名称	数据型态	设定值(默认值)	功能
OUT	TIME	正数或0.0 (0.0)	IN1与IN2之差值

6.6.4 位串功能

6.6.4.1 AND

- 类型
  Function
- 功能描述
  位 AND 运算。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN1	ANY_BIT	正整数或0 (0)	第一个输入值
IN2	ANY_BIT	正整数或0 (0)	第二个输入值

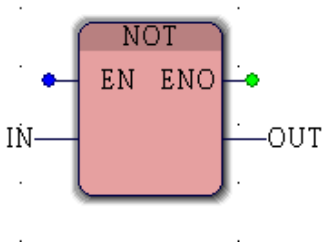
- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	ANY_BIT	正整数或0 (0)	IN1与IN2进行位AND运算之结果

6

6.6.4.2 NOT

- 类型
  Function
- 功能描述
  位 NOT 运算。
- 图形表示



- 输入脚位

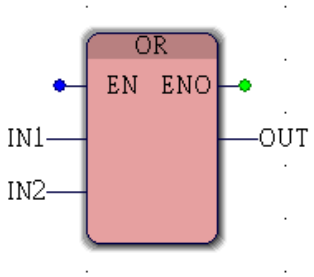
脚位名称	数据型态	设定值 (默认值)	功能
IN1	ANY_BIT	正整数或0 (0)	欲做反运算的输入数值

- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	ANY_BIT	正整数或0 (0)	反运算之结果

6.6.4.3 OR

- 类型
  Function
- 功能描述
  位 OR 运算。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN1	ANY_BIT	正整数或0 (0)	第一个输入数值
IN2	ANY_BIT	正整数或0 (0)	第二个输入数值

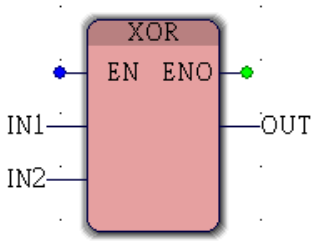
- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	ANY_BIT	正整数或0 (0)	IN1与IN2的逻辑OR运算之结果

6

6.6.4.4 XOR

- 类型
  Function
- 功能描述
  位 XOR 运算。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN1	ANY_BIT	正整数或0 (0)	输入的数值
IN2	ANY_BIT	正整数或0 (0)	输入的数值

- 输出脚位

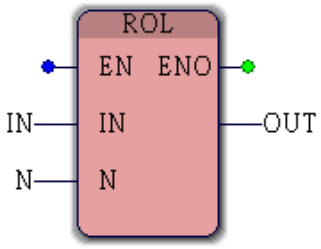
脚位名称	数据型态	设定值 (默认值)	功能
OUT	ANY_BIT	正整数或0 (0)	IN1与IN2的逻辑XOR运算之结果

6.6.5 位移功能

6.6.5.1 ROL

6

- 类型
  Function
- 功能描述
  位向左旋转位移。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN	ANY_BIT	正整数或0 (0)	欲被位移之数值
N	ANY_INT	正整数或0 (0)	向左旋转位移的位数

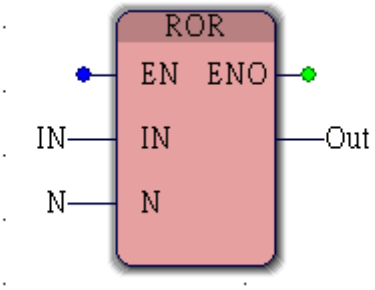
- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	ANY_BIT	正整数或0 (0)	将IN数值的所有位向左旋转移动N位后之结果

6.6.5.2 ROR

6

- 类型
  Function
- 功能描述
  位向右旋转位移。
- 图形表示



- 输入脚位

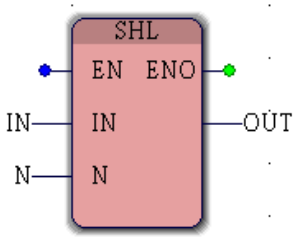
脚位名称	数据型态	设定值 (默认值)	功能
IN	ANY_BIT	正整数或0 (0)	欲被位移之数值
N	ANY_INT	正整数或0 (0)	向右旋转位移的位数

- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	ANY_BIT	正整数或0 (0)	将IN数值的所有位向右旋转移动N位后之结果

6.6.5.3 SHL

- 类型
  Function
- 功能描述
  位向左位移。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN	ANY_BIT	正整数或0 (0)	欲被位移之数值
N	ANY_INT	正整数或0 (0)	向左位移的位数

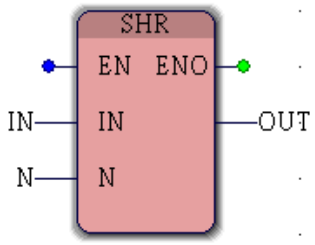
- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	ANY_BIT	正整数或0 (0)	将IN数值的所有位向左移动N位后之结果

6.6.5.4 SHR

6

- 类型
  Function
- 功能描述
  位向右位移。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN	ANY_BIT	正整数或0 (0)	欲被位移之数值
N	ANY_INT	正整数或0 (0)	向右位移的位数

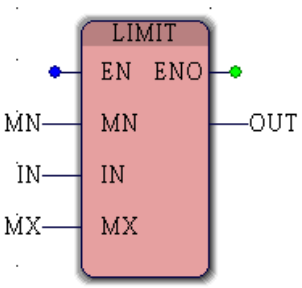
- 输出脚位

脚位名称	数据型态	设定值(默认值)	功能
OUT	ANY_BIT	正整数或0 (0)	将IN数值的所有位向右移动N位后之结果

6.6.6 选择运算功能

6.6.6.1 LIMIT

- 类型
  Function
- 功能描述
  输出限制。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
MN	ELEMENTARY	取决于输入参数所连变量的数据型态	最小限制值
IN	ELEMENTARY	取决于输入参数所连变量的数据型态	被限制的输入数值
MX	ELEMENTARY	取决于输入参数所连变量的数据型态	最大限制值

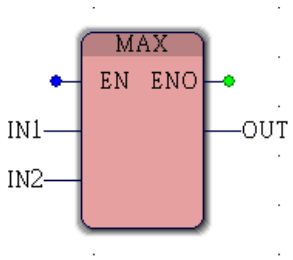
- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	ELEMENTARY	取决于输出参数所连变量的数据型态	执行限制后的结果

6

6.6.6.2 MAX

- 类型
  Function
- 功能描述
  求最大值。
- 图形表示



- 输入脚位

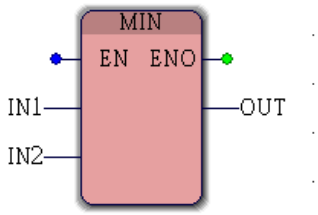
脚位名称	数据型态	设定值 (默认值)	功能
IN1	ELEMENTARY	取决于输入参数所连变量的数据型态	输入的数值
IN2	ELEMENTARY	取决于输入参数所连变量的数据型态	输入的数值

- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	ELEMENTARY	取决于输出参数所连变量的数据型态	输出IN1及IN2中之最大值

6.6.6.3 MIN

- 类型
  Function
- 功能描述
  求最小值。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN1	ELEMENTARY	取决于输入参数所连变量的数据型态	输入的数值
IN2	ELEMENTARY	取决于输入参数所连变量的数据型态	输入的数值

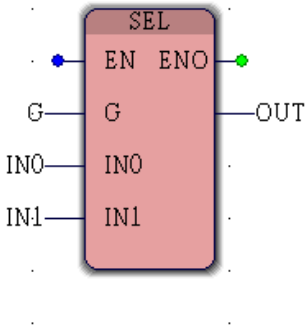
- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	ELEMENTARY	取决于输出参数所连变量的数据型态	输出IN1及IN2中之较小值

6.6.6.4 SEL

6

- 类型
  Function
- 功能描述
  二选一功能。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
G	BOOL	True / False (False)	当G为FALSE，选择IN0 当G为TRUE，选择IN1
IN0	ANY	取决于输入参数所连变量的数据型态	输入的数值
IN1	ANY	取决于入参数所连变量的数据型态	输入的数值

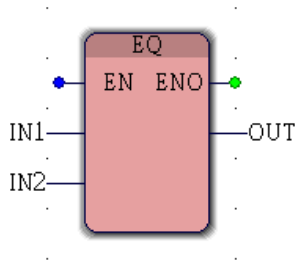
- 输出脚位

脚位名称	数据型态	设定值(默认值)	功能
OUT	ANY	取决于输出参数所连变量的数据型态	输出选择的结果

6.6.7 比较运算功能

6.6.7.1 EQ

- 类型
  Function
- 功能描述
  等于判断。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN1	ELEMENTARY	正数、负数或0.0 (0.0)	输入的数值
IN2	ELEMENTARY	正数、负数或0.0 (0.0)	输入的数值

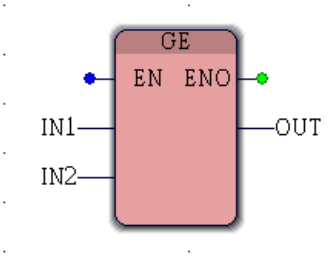
- 输出脚位

脚位名称	数据型态	设定值(默认值)	功能
OUT	BOOL	True / False (False)	IN1 = IN2时，输出为1，否则输出0

6.6.7.2 GE

6

- 类型
  Function
- 功能描述
  大于等于判断。
- 图形表示



- 输入脚位

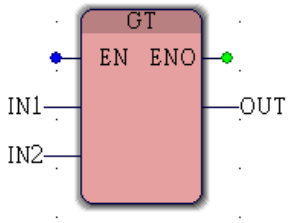
脚位名称	数据型态	设定值 (默认值)	功能
IN1	ELEMENTARY	正数、负数或0.0 (0.0)	输入的数值
IN2	ELEMENTARY	正数、负数或0.0 (0.0)	输入的数值

- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	BOOL	True / False (False)	IN1 >= IN2时，输出为1，否则输出为0

6.6.7.3 GT

- 类型
  Function
- 功能描述
  大于判断。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN1	ELEMENTARY	正数、负数或0.0 (0.0)	输入的数值
IN2	ELEMENTARY	正数、负数或0.0 (0.0)	输入的数值

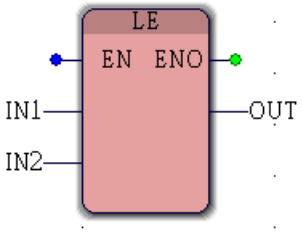
- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	BOOL	True / False (False)	IN1 > IN2时，输出为1，否则输出为0

6

6.6.7.4 LE

- 类型
  Function
- 功能描述
  小于等于判断。
- 图形表示



- 输入脚位

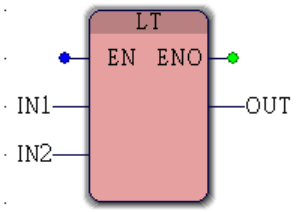
脚位名称	数据型态	设定值 (默认值)	功能
IN1	ELEMENTARY	正数、负数或0.0 (0.0)	输入的数值
IN2	ELEMENTARY	正数、负数或0.0 (0.0)	输入的数值

- 输出脚位

脚位名称	数据型态	设定值(默认值)	功能
OUT	BOOL	True / False (False)	IN1 <= IN2时，输出为1，否则输出为0

6.6.7.5 LT

- 类型
  Function
- 功能描述
  小于判断。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN1	ELEMENTARY	正数、负数或0.0 (0.0)	输入的数值
IN2	ELEMENTARY	正数、负数或0.0 (0.0)	输入的数值

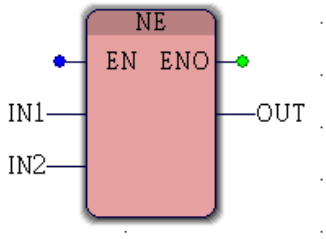
- 输出脚位

脚位名称	数据型态	设定值(默认值)	功能
OUT	BOOL	True / False (False)	IN1 < IN2时，输出为1，否则输出为0

6

6.6.7.6 NE

- 类型
  Function
- 功能描述
  不等于判断。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN1	ELEMENTARY	正数、负数或0.0 (0.0)	输入的数值
IN2	ELEMENTARY	正数、负数或0.0 (0.0)	输入的数值

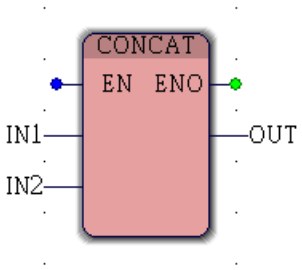
- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	BOOL	True / False (False)	IN1≠IN2时，输出为1，否则输出为0

6.6.8 字符串操作功能

6.6.8.1 CONCAT

- 类型
  Function
- 功能描述
  字符串连接。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN1	STRING	字符串	输入的字符串
IN2	STRING	字符串	输入连接的字符串

- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	STRING	字符串	输出连接后的字符串

6

6.6.8.2 DELETE

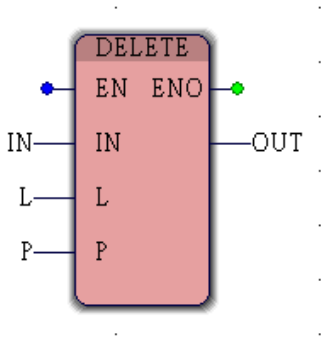
■ 类型

Function

■ 功能描述

删除指定长度的字符串。

■ 图形表示



■ 输入脚位

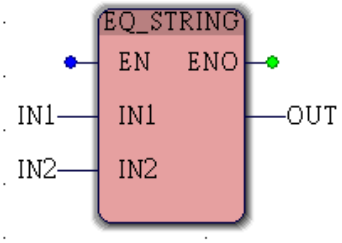
脚位名称	数据型态	设定值 (默认值)	功能
IN	STRING	正数、负数或0.0 (0.0)	输入的字符串
L	ANY_INT	0 ~ 字符串最大长度	删减字符的个数
P	ANY_INT	1 ~ 字符串最大长度	删减字符的起始位置

■ 输出脚位

脚位名称	数据型态	设定值(默认值)	功能
OUT	STRING	字符串	删减后的字符串

6.6.8.3 EQ_STRING

- 类型
  Function
- 功能描述
  字符串相等判断。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN1	STRING	字符串	第一个输入的字符串
IN2	STRING	字符串	第二个输入的字符串

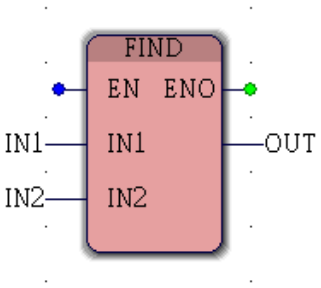
- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	BOOL	True / False (False)	IN1 = IN2时，输出为1，反之输出为0

6

6.6.8.4 FIND

- 类型
  Function
- 功能描述
  字符串查找。
- 图形表示



- 输入脚位

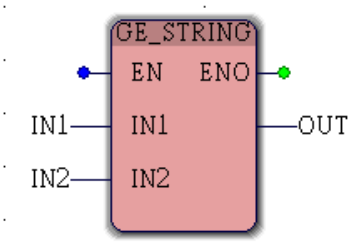
脚位名称	数据型态	设定值 (默认值)	功能
IN1	STRING	字符串	被查找之串
IN2	STRING	字符串	寻找的关键词串

- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	INT	正数或0	输出IN2字符串中的第一个字符出现在IN1字符串中的位置

6.6.8.5 GE_STRING

- 类型
  Function
- 功能描述
  英文字符顺序大于等于判断。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN1	STRING	字符串	第一个输入的字符串
IN2	STRING	字符串	第二个输入的字符串

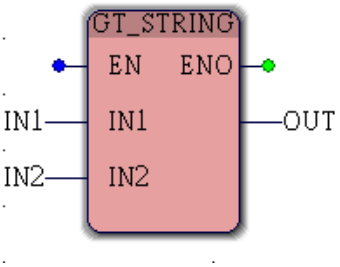
- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	BOOL	True / False (False)	IN1 >= IN2时，输出为1，否则输出为0

6.6.8.6 GT_STRING

6

- 类型
  Function
- 功能描述
  英文字符顺序大于判断。
- 图形表示



- 输入脚位

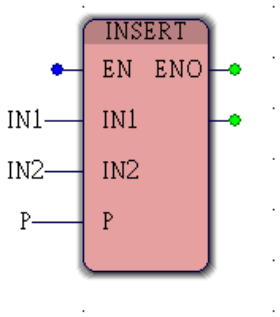
脚位名称	数据型态	设定值 (默认值)	功能
IN1	STRING	字符串	第一个输入的字符串
IN2	STRING	字符串	第二个输入的字符串

- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	BOOL	True / False (False)	IN1 > N2时，输出为1，否则输出为0

6.6.8.7 INSERT

- 类型
  Function
- 功能描述
  字符串插入。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN1	STRING	字符串	被插入之字符串
IN2	STRING	字符串	欲插入之字符串
P	ANY_INT	1 ~ 字符串最大长度	插入字符之起始位置

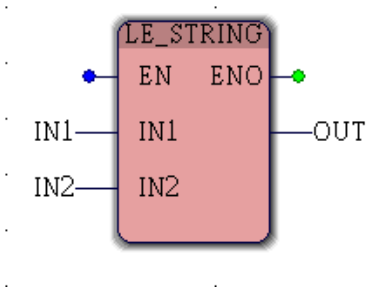
- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	STRING	字符串	插入后的字符串

6.6.8.8 LE_STRING

6

- 类型
  Function
- 功能描述
  英文字符顺序小于等于判断。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN1	STRING	字符串	第一个字符串
IN2	STRING	字符串	第二个字符串

- 输出脚位

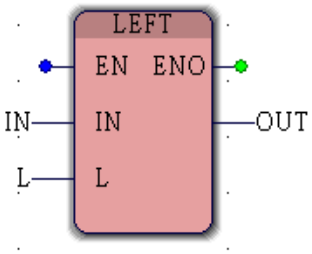
脚位名称	数据型态	设定值 (默认值)	功能
OUT	BOOL	True / False (False)	IN1 <= IN2时，输出为1，否则输出为0

6.6.8.9 LEFT

■ 类型
Function

■ 功能描述
取字符串中由最左往右边连续个数字符。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN	STRING	字符串	输入的字符串
L	ANY_INT	0 ~ 截取字符串的长度 (0)	撷取字符串的长度

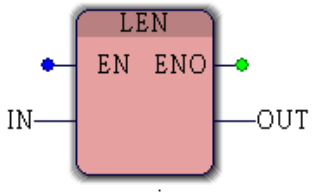
■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	STRING	字符串	撷取的字符串

6.6.8.10 LEN

6

- 类型
  Function
- 功能描述
  计算字符串的字符个数。
- 图形表示



- 输入脚位

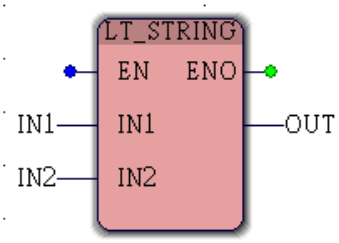
脚位名称	数据型态	设定值 (默认值)	功能
IN	STRING	字符串	输入的字符串

- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	INT	正整数或0 (0)	字符串之长度

6.6.8.11 LT_STRING

- 类型
  Function
- 功能描述
  英文字符顺序小于判断。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN1	STRING	字符串	第一个输入的字符串
IN2	STRING	字符串	第二个输入的字符串

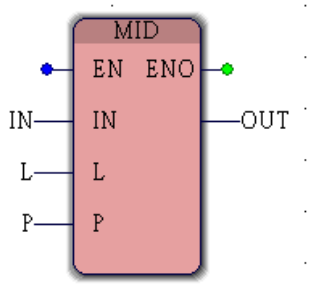
- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	BOOL	True / False (False)	IN1 < IN2 时，输出为 1，否则输出为 0。

6.6.8.12 MID

6

- 类型
  Function
- 功能描述
  撷取字符串中一指定长度的字符串。
- 图形表示



- 输入脚位

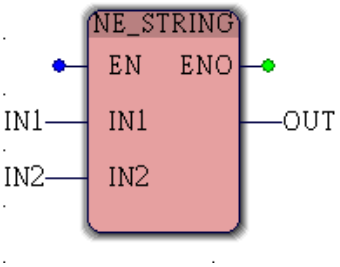
脚位名称	数据型态	设定值（默认值）	功能
IN	STRING	字符串	输入的字符串
L	ANY_INT	0 ~ 字符串最大长度（0）	撷取字符串的长度，最多为80字符
P	ANY_INT	1 ~ 字符串最大长度（1）	撷取字符串的位置，不可为0

- 输出脚位

脚位名称	数据型态	设定值（默认值）	功能
OUT	STRING	字符串	截取的字符串

6.6.8.13 NE_STRING

- 类型
  Function
- 功能描述
  英文字符顺序不等于判断。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN1	STRING	字符串	第一个字符串
IN2	STRING	字符串	第二个字符串

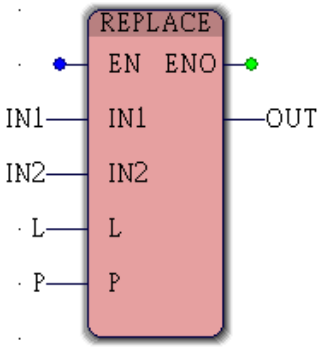
- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	BOOL	True / False (False)	IN1 ≠ N2时，输出为1，反之输出为0

6

6.6.8.14 REPLACE

- 类型
  Function
- 功能描述
  字符串取代。
- 图形表示



- 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN1	STRING	字符串	原始字符串
IN2	STRING	字符串	欲取代之字符串
L	ANY_INT	0 ~ 字符串最大长度	欲取代的字符个数
P	ANY_INT	1 ~ 字符串最大长度	欲取代的字符起始位置

- 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	STRING	字符串	取代后的字符串

6.6.8.15 RIGHT

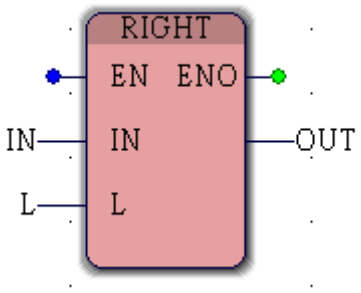
■ 类型

Function

■ 功能描述

取字符串中由右往左连续个数字符。

■ 图形表示



■ 输入脚位

脚位名称	数据型态	设定值 (默认值)	功能
IN	STRING	字符串	输入的字符串
L	ANY_INT	0 ~ 撷取字符串的长度 (0)	撷取字符串的长度

■ 输出脚位

脚位名称	数据型态	设定值 (默认值)	功能
OUT	STRING	字符串	撷取的字符串

(此页有意留为空白)

6

7 运动控制应用

本章将对 DXMC 所提供的抓取 (Capture) 功能与电子凸轮功能进行说明，包括高速抓取、低速抓取的范例与飞剪、追剪的范例说明。

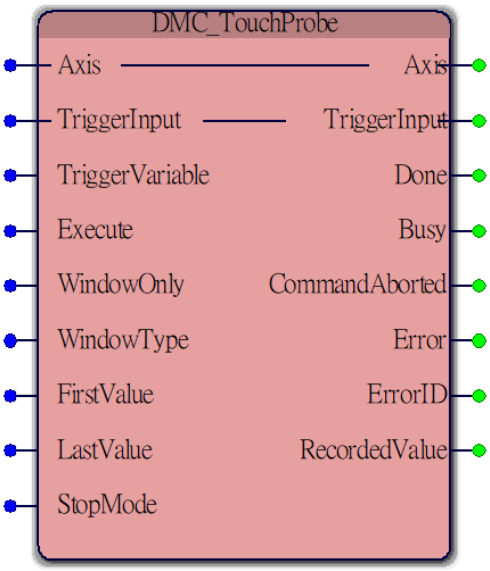
7.1 抓取 (Capture) 功能说明 …… 7-2
7.1.1 DMC_TouchProbe 功能块说明 …… 7-2
7.1.2 高速抓取范例 (Driver Mode) …… 7-6
7.1.3 高速抓取范例 (Controller Mode) …… 7-7
7.1.4 低速抓取范例 …… 7-8
7.2 电子凸轮 (E-CAM) 功能说明 …… 7-9
7.2.1 飞剪功能 …… 7-11
7.2.2 追剪功能 …… 7-13

7.1 抓取 (Capture) 功能说明

7

DMC_TouchProbe 此功能块的概念是依据使用者之设定，在触发事件中抓取并记录所需要的数据作为后续运动控制使用。数据抓取的动作分为高速抓取及低速抓取功能。高速抓取的优点在于由硬件完成，没有软件延迟的问题，对于高速运转的运动轴也可以准确地抓取，但仅能抓取轴位置为其缺点。相对于高速抓取而言，虽然低速抓取受限于控制器通讯周期的影响，无法做到高精度的数据抓取，但其优点在于多种类的数据抓取 (轴位置、速度、加减速度、扭矩…等)，为后续的运动控制提供了多样性。

7.1.1 DMC_TouchProbe 功能块说明



■ 触发指定 (TriggerInput)

通过触发指定 (TriggerInput) 的数据来抓取来源轴，该来源轴最多可同时执行 6 个 DMC_TouchProbe。触发指定 (TriggerInput) 脚位的数据型态为 MC_TRIGGER_REF。

注：MC_TRIGGER_REF 为自定义数据型态，详细请参考章节 6.5.9。

MC_TRIGGER_REF 数据结构成员说明

成员名称	数据型态	说明
TriggerMode	USINT	设定触发模式。 0：驱动器模式 (Drive Mode) 1：控制器模式 (Controller Mode)
TriggerID	USINT	指定触发事件编号。 0 ~ 1：驱动器模式 (Drive Mode) 2 ~ 5：控制器模式 (Controller Mode)
TriggerType	USINT	设定触发型态。 0：Build-in DI (控制器本体 DI) 1：Local DI (右侧扩充模块 DI) 2：Remote DI (EtherCAT 模块 DI) 3：PLC Variable
DINumber	UINT	指定触发事件所使用的 DI 编号。 格式：AUZYX (十进制) AUZ：编号 (DI 模块 Node ID) YX：次编号 (DI 端口编号)
RisingTrigger	BOOL	若为 True，触发事件为上升沿触发；反之，为下降沿触发。
RecordedData	USINT	指定纪录数据源。 00：Drive Capture Pos Cmd 01：Drive Capture Pos Fb 02：Drive Capture 外扩编码器 03：Controller Capture Pos Fb 10：Pos Cmd 11：Pos Fb PUU 12：PosTarget PUU 13：Vel Cmd PUU 14：Vel Fb PUU 15：AccDec Cmd PUU 16：AccDec Fb PUU
RSVD	ARRAY OF SINT[14]	保留

7

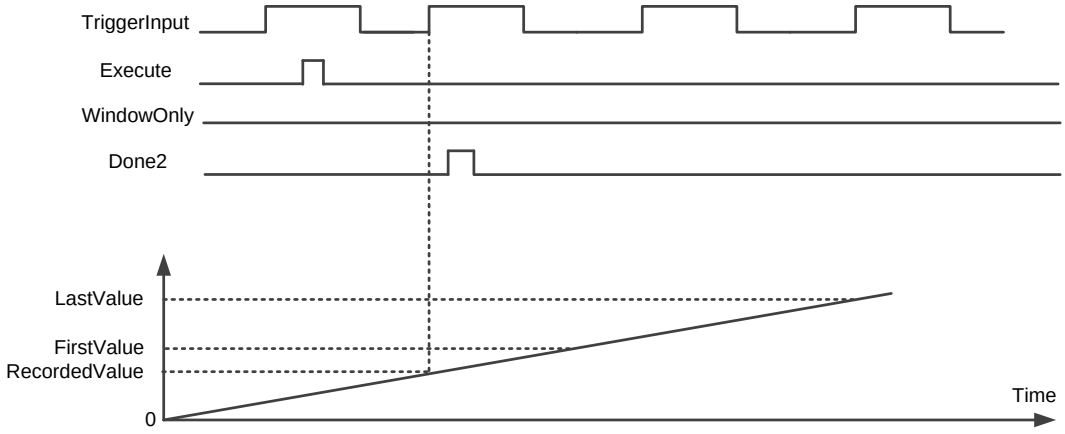
■ 触发变数 (TriggerVariable)

当触发型态 (TriggerType) 为 PLC Variable 的时候 (TriggerType = 3)，此脚位为触发信号来源。当此脚位为上升沿时，完成数据抓取的动作。

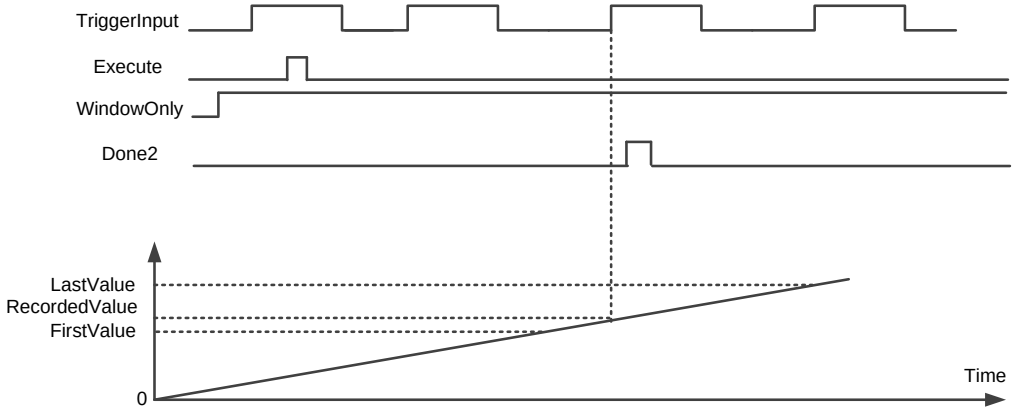
■ Window 功能说明 (WindowOnly 和 WindowType)

在 WindowOnly 中，可以指定 Window 功能是否开启。

情况 1：当 WindowOnly 设为 False 时，在所有轴位置皆可接收触发信号，如下图所示。



情况 2：当 WindowOnly 设为 True 时，根据 WindowType 当作参考来源。当选定的参考来源数值落于 FirstValue 与 LastValue 的范围内时，接受触发信号。



当 WindowType = 0 时，Window 范围参考来源为 RecordedData 所设定的数据种类。
当 WindowType = 1 时，Window 范围参考来源为轴命令位置。
当 WindowType = 2 时，Window 范围参考来源为轴回授位置。
当 WindowType = 3 时，Window 范围参考来源为轴速度命令。
当 WindowType = 4 时，Window 范围参考来源为轴速度回授。
当 WindowType = 5 时，Window 范围参考来源为轴加速度命令。
当 WindowType = 6 时，Window 范围参考来源为轴加速度回授。

例如：若 WindowType = 2，当轴回授位置在 FirstValue 与 LastValue 的范围内时，则完成数据抓取的动作。

■ 起始值与末端值说明 (FirstValue 和 LastValue)

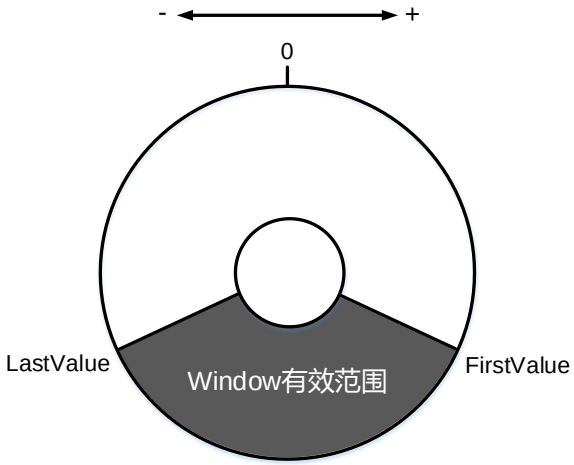
FirstValue 与 LastValue 分别为 Window 有效范围的起始值与末端值。不同计数模式的 FirstValue 和 LastValue 的范围如下所示。

线性模式：FirstValue (起始值) ≤ Window 有效范围 ≤ LastValue (末端值)

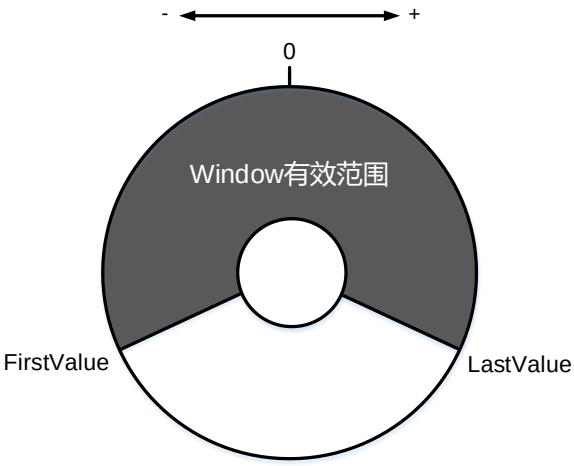
注：若 LastValue (末端值) ≤ FirstValue (起始值)，则会发生报警 0x5012。

循环模式：

1. 当 FirstValue (起始值) ≤ LastValue (末端值)，Window 有效范围如下图所示。



2. 当 LastValue (末端值) ≤ FirstValue (起始值)，Window 有效范围如下图所示。



■ StopMode

目前不支持此功能。

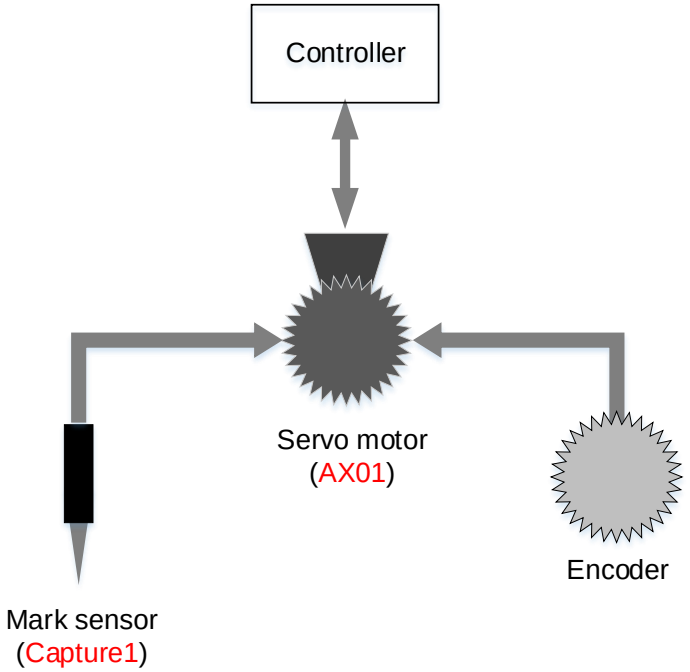
■ 中断说明

当触发 DMC_TouchProbe 功能块后，若要中断资料抓取，可以通过 MC_AbortTrigger 功能块来实现，详细说明请参考章节 6.5.2.18。

7.1.2 高速抓取范例 (Driver Mode)

7

高速抓取可以使用伺服驱动器提供的 Capture 功能进行数据抓取，其硬件配置如下图所示。



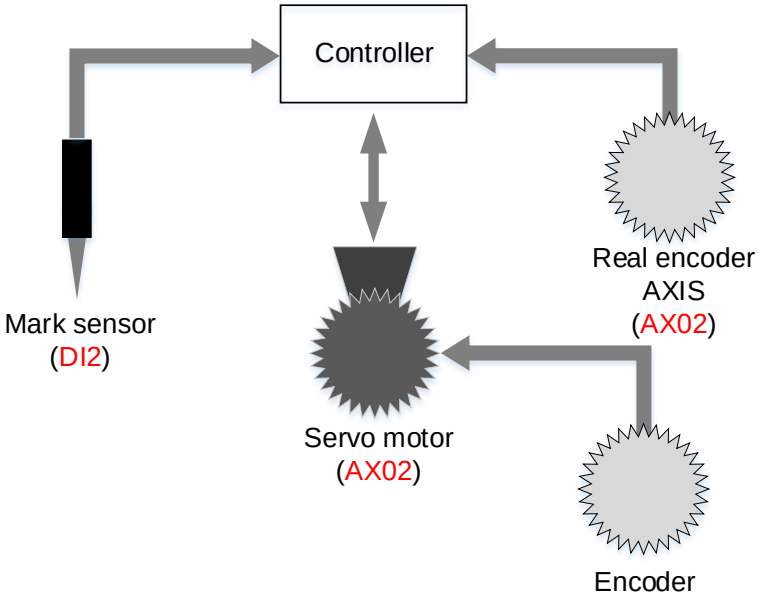
PLC 功能块的设定如下：

- MC_TRIGGER_REF 设定
  1. 将触发模式 (TriggerMode) 指定为驱动器模式 (Drive Mode)：TriggerMode = 0。
  2. 将指定触发事件编号 (TriggerID) 设为 0：TriggerID = 0。
  3. 将触发型态 (TriggerType) 指定为伺服本体 DI：TriggerType = 0。
  4. 指定 DI 编号。若使用 Capture1 功能，DINumber = 00001；若使用 Capture2 功能，DINumber = 00002 (是否具备 Capture2 功能视伺服机种而定)。
  5. 指定上升沿触发：RisingTrigger = True。
  6. 指定伺服撷取的数据种类：RecordedData = 01。
- WindowOnly 设定

  开启 Window 功能：WindowOnly = True。
- Window 起始和结束位置
  1. 将 Window 起始位置指定为变量 FirstValue。
  2. 将 Window 末端位置指定为变量 LastValue。

7.1.3 高速抓取范例 (Controller Mode)

高速抓取亦可以使用控制器本身的 Capture 功能进行数据抓取，其硬件配置如下图所示。



PLC 功能块的设定如下：

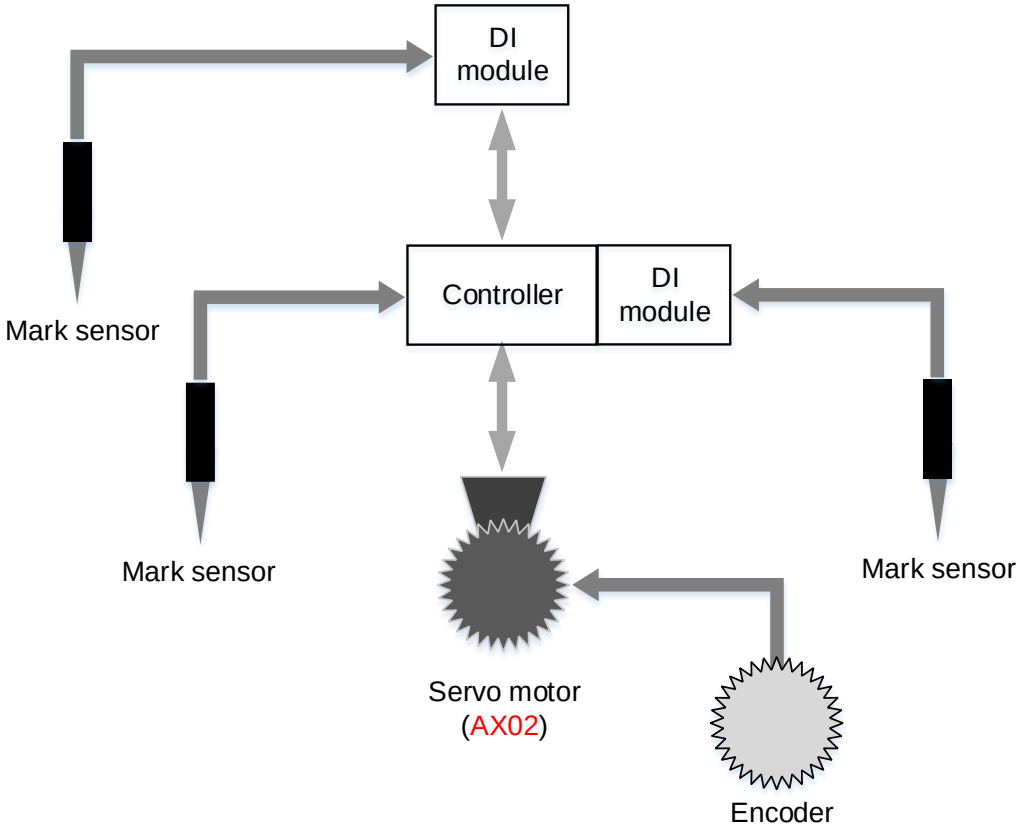
- MC_TRIGGER_REF 设定
  1. 将触发模式 (TriggerMode) 指定为控制器模式 (Controller Mode)：TriggerMode = 1。
  2. 将指定触发事件编号 (TriggerID) 设为 2：TriggerID = 2。
  3. 将触发型态 (TriggerType) 指定为控制器本体 DI：TriggerType = 0。
  4. 指定 DI 编号。若欲指定控制器本体 DI 1 作为触发信号，DINumber = 00001；欲指定控制器本体 DI 2 作为触发信号，DINumber = 00002，依此类推。
  5. 指定上升沿触发：RisingTrigger = True。
  6. 指定伺服撷取的数据种类：RecordedData = 03。
- WindowOnly 设定

  开启 Window 功能：WindowOnly = True。
- Window 起始和结束位置
  1. 将 Window 起始位置指定为变量 FirstValue。
  2. 将 Window 末端位置指定为变量 LastValue。

7.1.4 低速抓取范例

7

低速抓取可以使用扩充 DI 模块为触发信号来源并进行数据抓取，其硬件配置如下图所示。



PLC 功能块的设定如下：

- MC_TRIGGER_REF 设定
  1. 将触发模式 (TriggerMode) 指定为控制器模式 (Controller Mode)：TriggerMode = 1。
  2. 将指定触发事件编号 (TriggerID) 设为 2：TriggerID = 2。
  3. 将触发型态 (TriggerType) 指定为扩充 DI 模块：TriggerType = 1。
  4. 指定 DI 编号。欲使用扩充 DI 模块 (Node ID 为 3) 的 DI 4 作为触发信号，则 DINumber = 00304，依此类推。
  5. 指定上升沿触发：RisingTrigger = True。
  6. 指定伺服撷取的数据种类：RecordedData = 10 ~ 16。
- WindowOnly 设定

  开启 Window 功能：WindowOnly = True。
- Window 起始和结束位置
  1. 将 Window 起始位置指定为变量 FirstValue。
  2. 将 Window 末端位置指定为变量 LastValue。

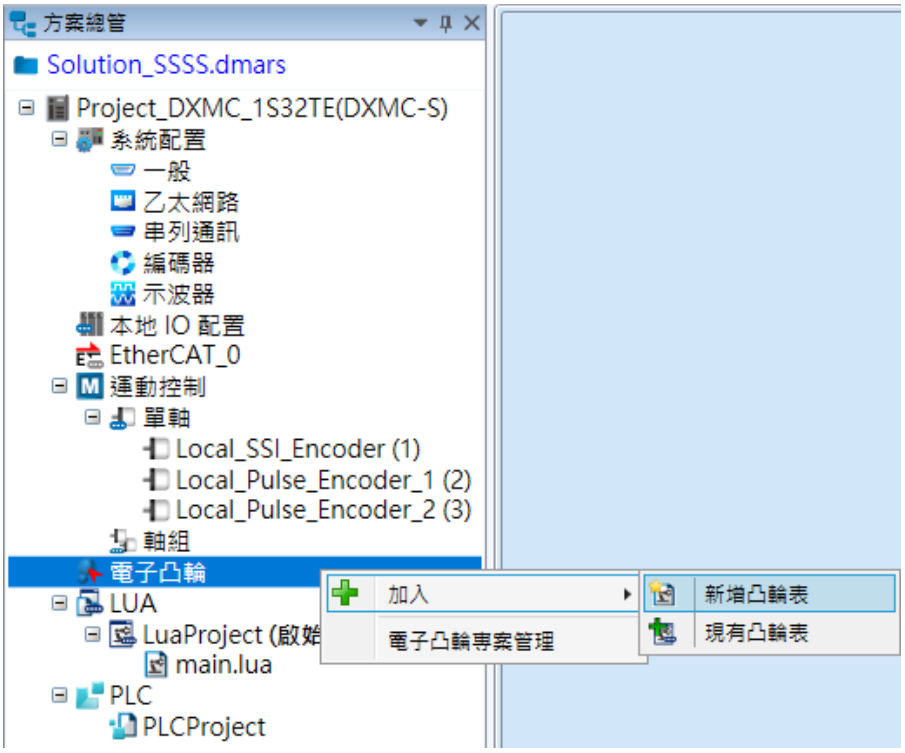
7.2 电子凸轮 (E-CAM) 功能说明

电子凸轮为机械凸轮的延伸，当主轴开始旋转，机械凸轮将带动凸轮的旋转，透过机械连杆，使从轴随着凸轮上的曲线上下滑动。电子凸轮则是透过运算的方式使主从轴呈现机械凸轮的运动关系，主轴与从轴间的周期运动可以以一张凸轮表呈现。目前可以至 DMARS 的 E-CAM 功能中绘制凸轮表，或是至 PLC 项目中，使用 ECAM_GenTableByData 来建立凸轮表。

■ DMARS 建表

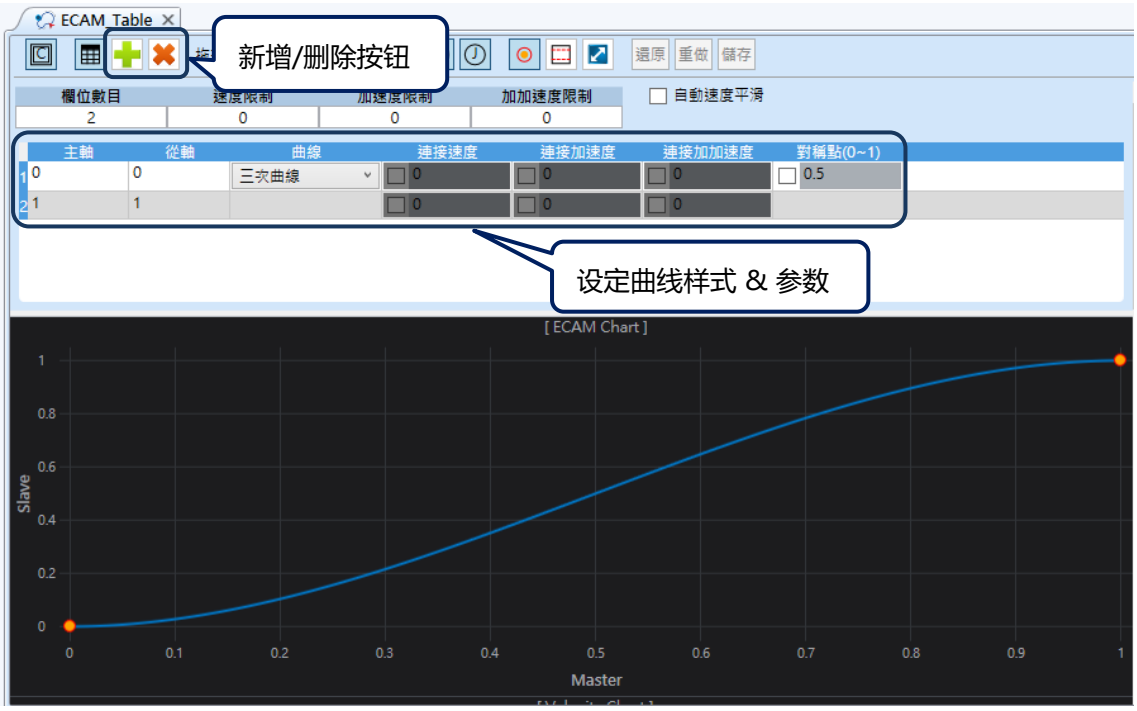
透过 DMARS 可以生成凸轮表，首先可以在左方方案总管中的**电子凸轮**点选右键加入新的凸轮表。输入完文件名后，系统会自动建立一张曲线表。

注：DMARS 目前仅提供曲线建表功能。

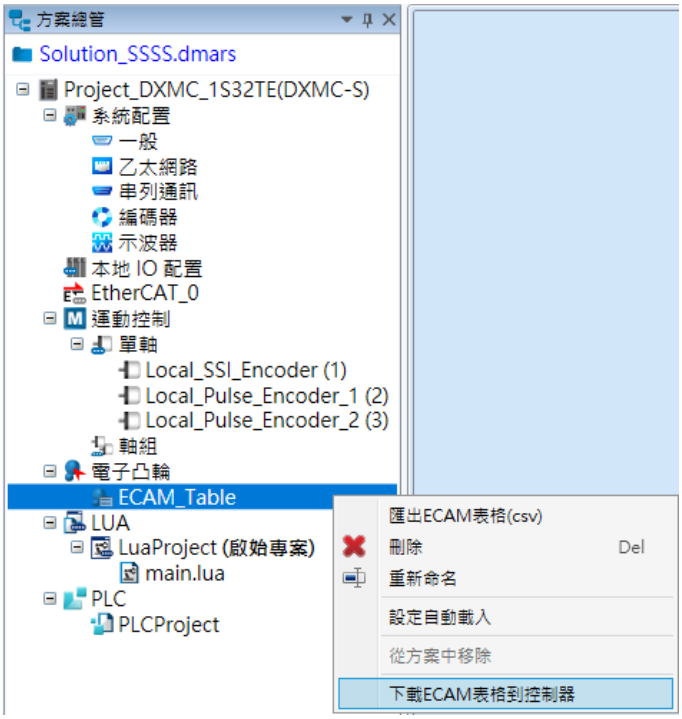


7

在建表页面中，可以透过左上角的  按钮来新增及删除点数。更改主从轴每一点的位置和设定连接的曲线样式，即可建立想要的凸轮表。特定的曲线样式还可以设定连接速度、连接加速度和连接加加速度等参数。



完成建表之后，可以将设计好的凸轮表下载到控制器的断电保持区 (FLASH)，副文件格式为.bin 檔。当需要存取凸轮表时，可透过 ECAM 功能块，来将数据加载 RAM。

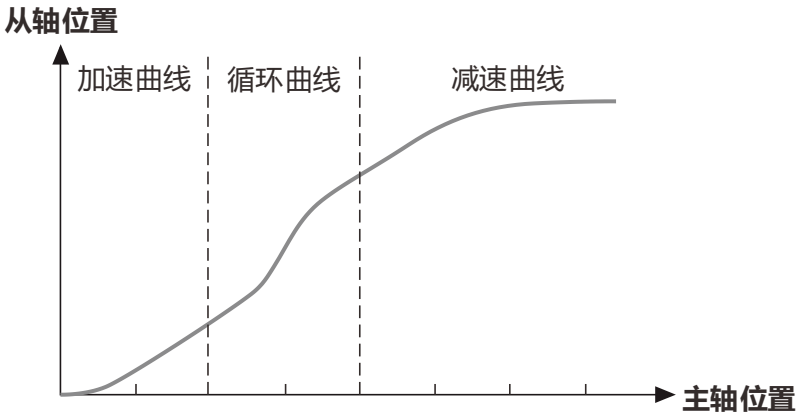


■ ECAM_GenTableByData 建表

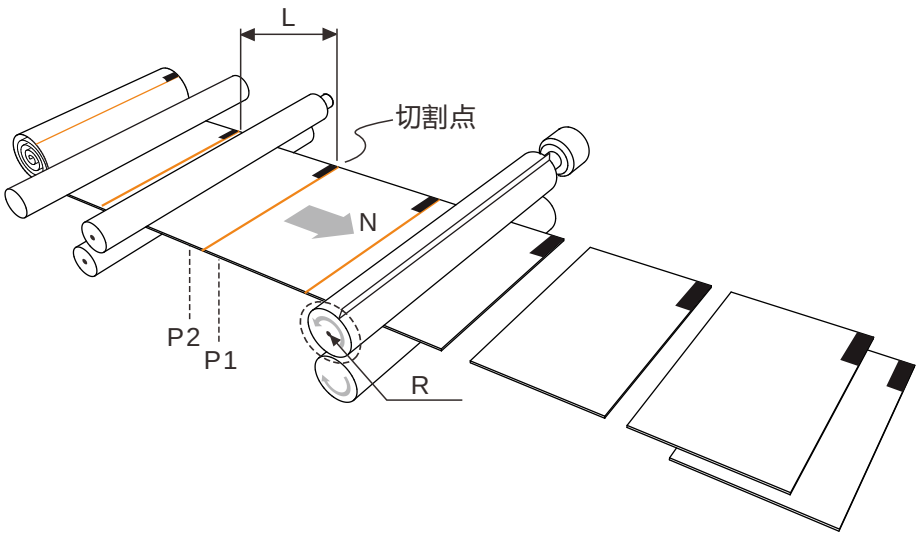
除了透过 DMARS 软件建表之外，也可以透过 PLC 提供的 ECAM 功能块来建立凸轮表。ECAM 功能块使用方式请参考章节 6.5.8。

7.2.1 飞剪功能

飞剪功能为电子凸轮中常见的应用，泛用于许多生产在线。例如常应用于枕式包装机的切刀轴，其重点在于切下物品时必须与进给轴做等速运动，才不会使包装纸发生挤压。飞剪曲线可以分为加速曲线、循环曲线和减速曲线。飞剪功能启动时，会先执行加速曲线，接着重复执行循环曲线，直到关闭飞剪功能时，会执行减速曲线并停止在起始位置(即终点位置)。DXMC 提供 ECAM_GenRotaryCutTable 来设置飞剪表的各个参数，包含切刀轴的半径、切刀数和切刀长度等，其参数设定可以参考章节 6.5.8.10。



如下图所示，使用者可以根据工艺要求设置切割长度 (L)，切割长度可以小于或大于切刀周长。



7

图中参数	说明	脚位名称
N	旋切轴的刀头数，图中的刀头数为1。	RotaryAxisKnifeNum
R	旋切轴半径，为旋切轴中心到刀尖的距离 (单位：mm)。	RotaryAxisRadius
L	材料的切割长度 (单位：mm)。	CutLength
P1	同步区开始位置 (单位：mm)。	SyncStartPos
P2	同步区结束位置 (单位：mm)。	SyncStopPos

另外，旋切功能支持多刀头的旋切棍 (如下图)，使用者可以根据机构设计来设置此参数。

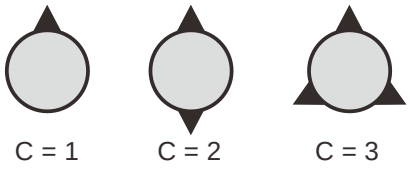
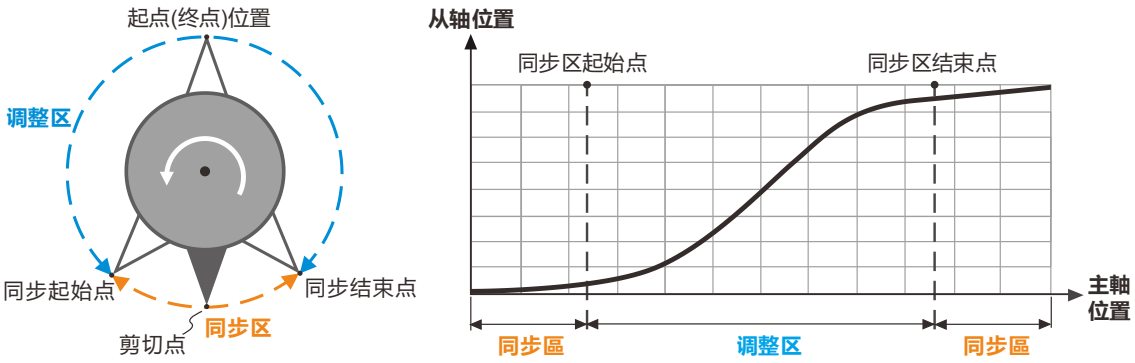


图 7.2.1.1 刀头数示意图

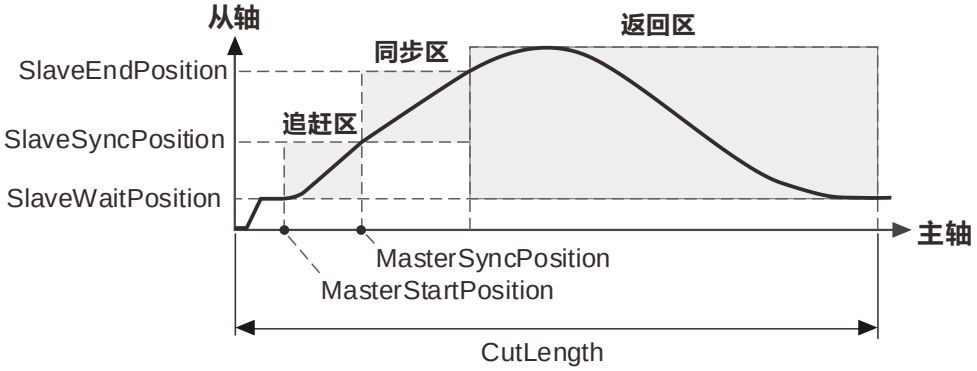
用户可以设定循环曲线中的同步区起始位置及同步区停止位置。如果同步区设定太小会出现扯膜现象；如果设定太大会压缩到其他区域的运行速度，造成系统震荡。在同步区时，进给轴与切刀轴会按照固定的速度比例运转 (切刀的切线速度与进给轴速度相等)，切割发生在同步区内。有时因为包装膜受热或物品高度的关系，使同步区速度必须做微调，可以使用 ECAM_ModRotCutVelRatio 功能块来进行同步区速度的微调。



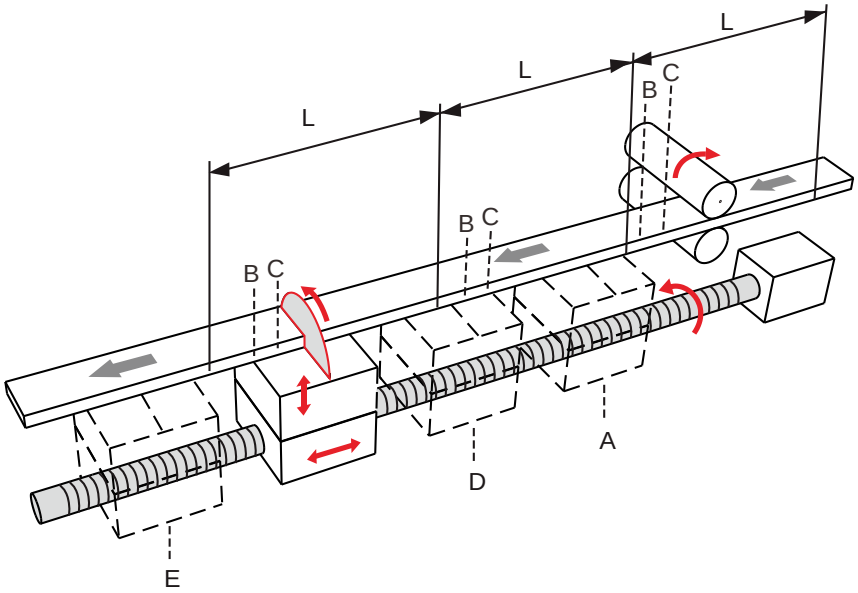
7.2.2 追剪功能

追剪的切刀机构会平行于目标物体，切刀会做与运行平面垂直的切剪，并且在切下物品的瞬间与送料轴做等速运动，主要是避免送料物的挤压现象。DXMC 提供 ECAM_GenFlyingShearTable 来设置追剪表的各个参数，包含追赶区和同步区参数，其参数设定可以参考章节 6.5.8.11。

追剪主要分成追赶区、同步区和返回区三个部分做循环运动。追剪功能启动后，从轴运动至等待位置并停住。当主轴运动至开始位置时，从轴开始追赶主轴，追剪功能进入追赶区。同步区开始时，主从轴以等速度比例进行运动。从轴到达时，同步区结束，从轴开始减速，追剪功能进入返回区。最终，从轴反转到等待位置，等待下一周期。



使用者可以根据需求自由设置切割长度，即下图的 L，也可以根据工艺要求设置同步区位置与长度。下图中，A 为从轴等待位置，当追剪功能启动后，从轴会运行到此位置；B 为主轴开始位置，当主轴通过此点，从轴由等待位置开始追赶主轴；C 为主轴同步位置；D 为同步区开始时，对应的从轴位置；E 为同步区结束时，对应的从轴位置。



图中参数	说明	脚位名称
A	从轴的等待位置。当追剪功能启动后，从轴自动运行到此位置。	SlaveWaitPosition
B	开始位置。当主轴到达此位置时，从轴由等待位置开始追赶主轴，以实现速度同步。	MasterStartPosition
C	同步区开始时，对应的主轴位置。	MasterSyncPosition
D	同步区开始时，对应的从轴位置。	SlaveSyncPosition
E	同步区结束时，对应的从轴位置。	SlaveEndPosition
L	材料的切割长度。	CutLength

在同步区内时，主轴与从轴会按照一定的速度比例运转，通常两者速度会设定相等，并且切割发生在同步区内。当追剪功能结束后，从轴会回到等待位置。

8 通讯功能

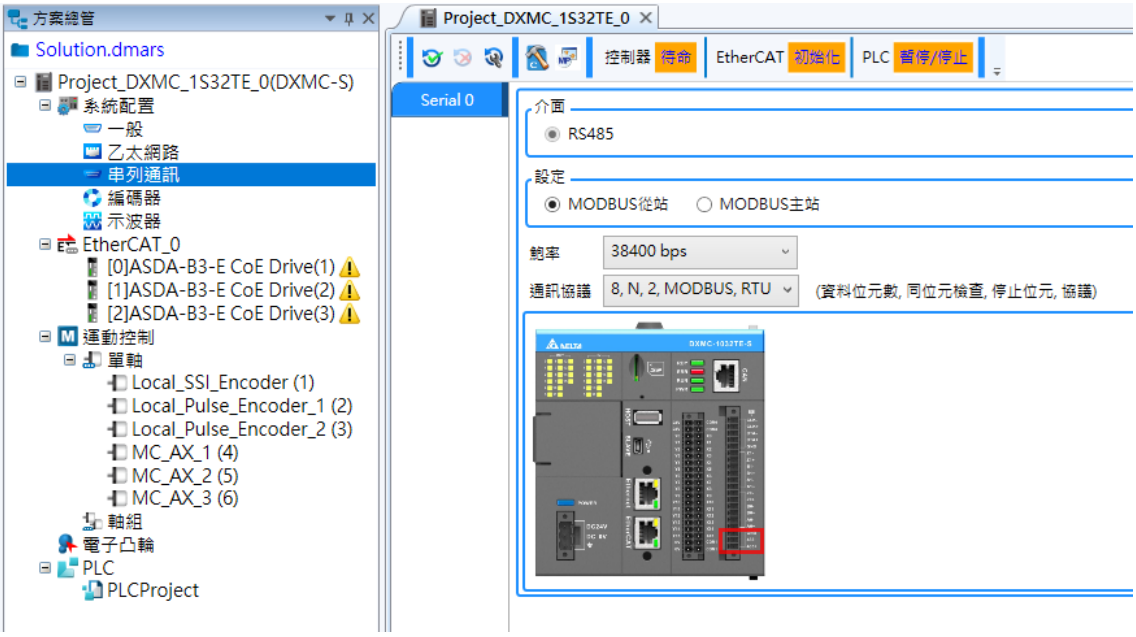
本章节介绍 DXMC 之 MODBUS 通讯操作。MODBUS 通讯主要用于一般参数的通讯读写，若要使用运动总线控制则请参考 DMCNET 的相关说明文件。此章节也会提到三种通讯格式：ASCII、RTU 和 TCP，及各模式的编码意义与通讯数据结构。

8.1 通讯参数设定 …… 8-2
8.2 MODBUS 通讯协议 …… 8-4
8.3 自由口通讯协议 (Freeport protocol) …… 8-18

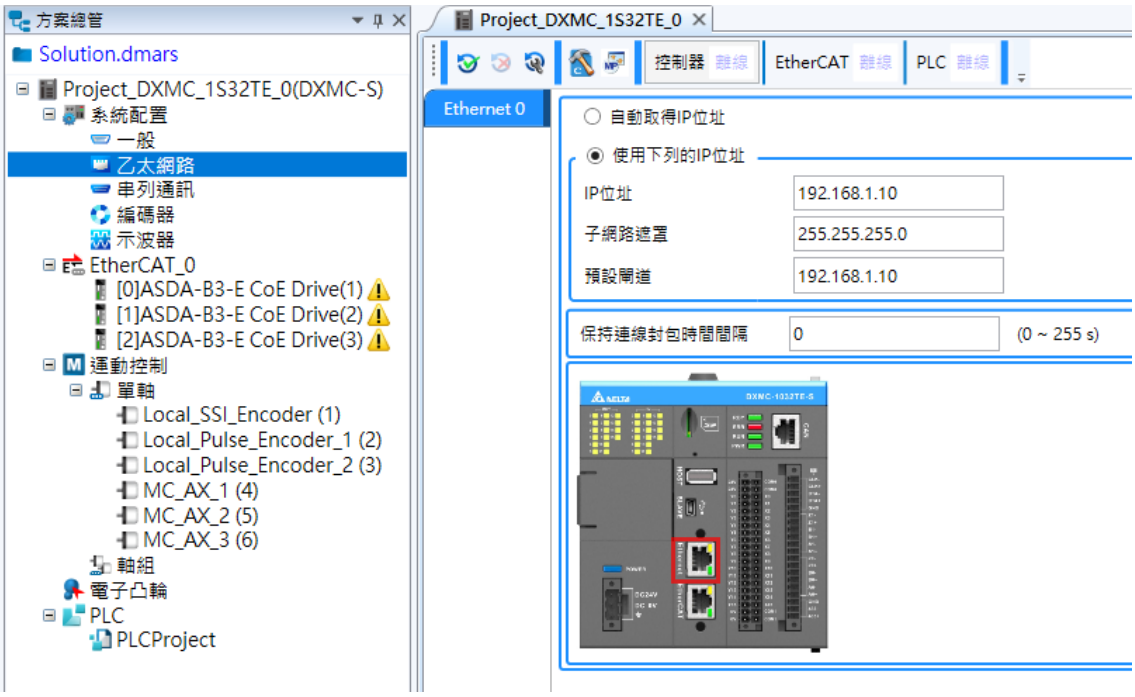
8.1 通讯参数设定

8

用户可以根据通讯方式的不同，选择设定相关的通讯参数。若使用串行通讯 RS-485，可在 DMARS 软件中左侧的项目树选择**串行通讯**，而右方工作区可以设定相关通讯参数，包括 MODBUS 通讯主站或是从站模式、波特率、通讯协议等参数。

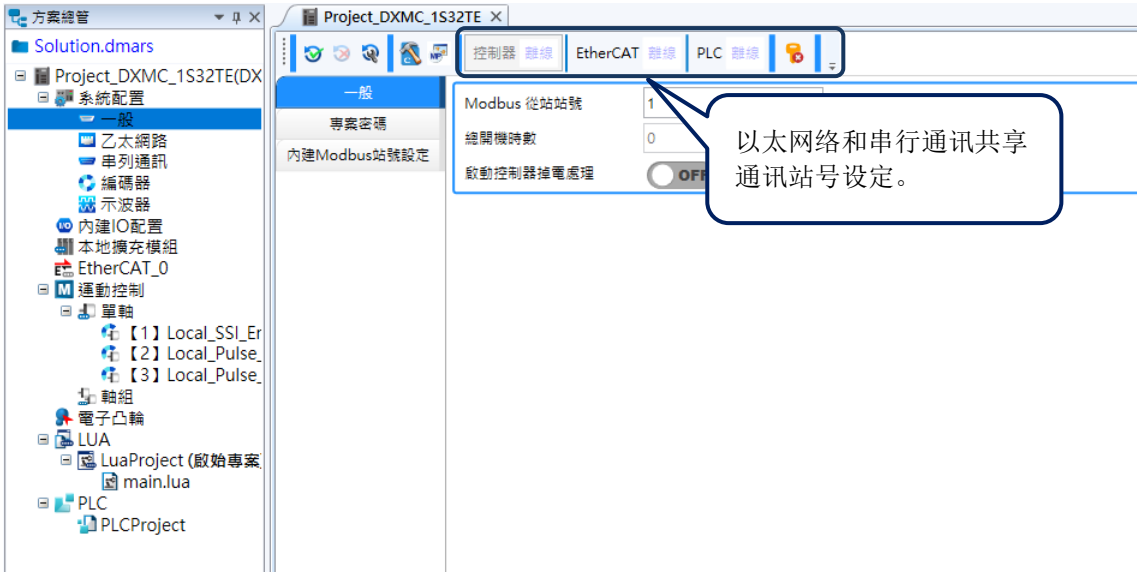


同样地，若需要使用以太网络，可以在项目树中选择**以太网络**，并设定为自动取得 IP 地址模式或使用固定 IP 地址模式。若欲使用固定 IP 地址模式，用户必须点选**使用下列的 IP 地址**，并设定 Ethernet 的 **IP 地址**、**子网掩码**，以及**默认网关**等必要参数。



DXMC控制器的所有通讯参数，除了透过DMARS软件设定之外，亦可透过PLC功能块来进行通讯参数设定，详情请参照章节6.5.5系统功能块（System Function)。

由于以太网络和串行通讯均可以设定通讯站号，因此在 DMARS 项目树的**一般**中可以进行站号设定，站号的范围为 1 ~ 247。



设定完通讯参数并下载设定至 DXMC 控制器后，便可透过 PLC 功能块来存取联机的装置，详细说明请参照章节 6.5.7 自定义通讯功能块（DMC FieldBus)。

8.2 MODBUS 通讯协议

8

MODBUS networks 通讯有三种模式：ASCII (American Standard Code for Information Interchange) 模式、RTU (Remote Terminal Unit) 模式与 TCP (Transmission Control Protocol)。用户可于 DMARS 项目树的**串行通讯**中设定 RS-485 所需之通讯协议 (ASCII 与 RTU 通讯模式)。而 Ethernet 仅能使用 TCP、USB-Serial 仅能使用 RTU，不能切换成 ASCII。DXMC 控制器支持功能 (Function) 03H 读取多笔数据、06H 写入单笔字符、10H 写入多笔字符，请参考以下说明。

编码意义

ASCII 模式：

所谓的 ASCII 模式，即在两个站 (主站与从站) 之间传输数据时，使用美国标准通讯交换码 (ASCII)。例如，主站与从站之间若要传输数值 64H，则会送出 ASCII 码的 36H 信号代表传输的数值‘6’，送出 ASCII 码的 34H 信号代表传输的数值‘4’。

数字 0 至 9 与字母 A 至 F 的 ASCII 码，如下表：

字符符号	‘0’	‘1’	‘2’	‘3’	‘4’	‘5’	‘6’	‘7’
对应 ASCII 码	30H	31H	32H	33H	34H	35H	36H	37H
字符符号	‘8’	‘9’	‘A’	‘B’	‘C’	‘D’	‘E’	‘F’
对应 ASCII 码	38H	39H	41H	42H	43H	44H	45H	46H

RTU 模式：

每个 8-bit 数据由两个 4-bit 之十六进制字符所组成。若两站之间要交换数值 64H，则直接传送数据 64H。此方式的传输效率比 ASCII 模式好。

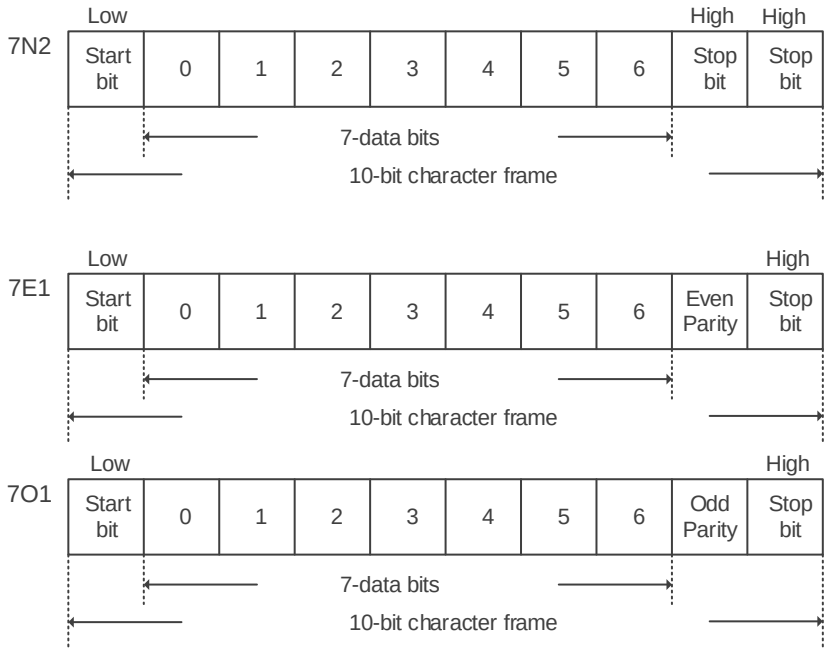
TCP 模式：

同 RTU 模式。

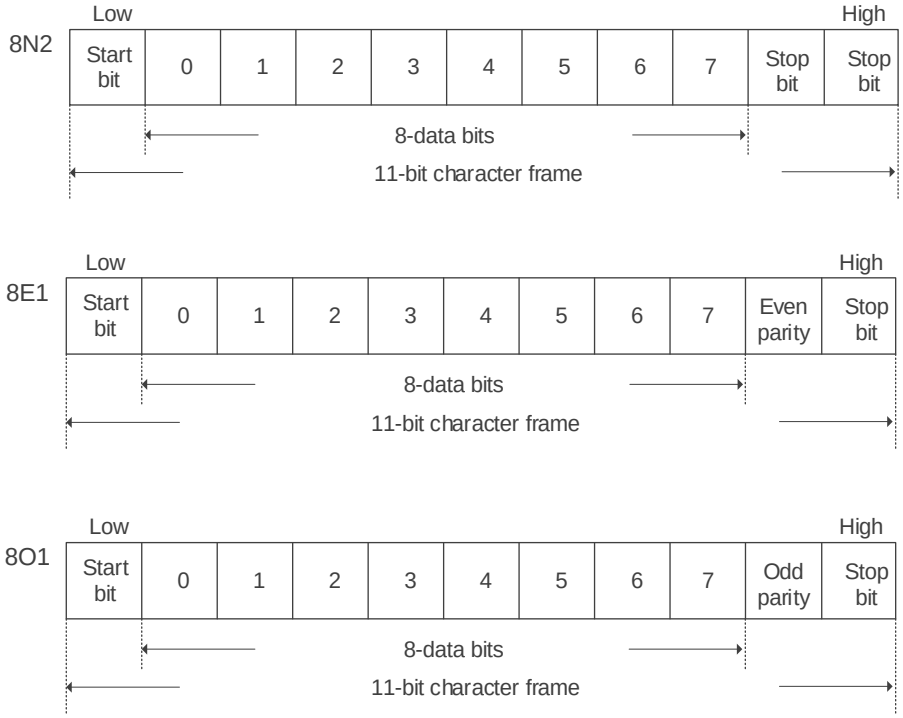
字符结构

字符将被编码成以下的框架（framing）并以串行方式传输，不同位的检查方法如下：

10 bits 字符框 (用于 7-bit 字符)



11 bits 字符框 (用于 8-bit 字符)



8

通讯数据结构

三种不同通讯模式的数据框（Data Frame）定义如下：

ASCII 模式：

Start	起始字符‘：’(3AH)
Slave Address	通讯地址：1-byte 包含了 2 个 ASCII 码
Function	功能码：1-byte 包含了 2 个 ASCII 码
Data (n-1)	数据内容：n-word = 2n-byte 包含了 4n 个 ASCII 码，n ≤ 10
…….	
Data (0)	
LRC	错误查核：1-byte 包含了 2 个 ASCII 码
End 1	结束码 1：(0DH)(CR)
End 0	结束码 0：(0AH)(LF)

ASCII 模式通讯的开头由冒号开始‘：’(ASCII 为 3AH)，ADR 为两个字符的 ASCII 码，结尾则为 CR (Carriage Return) 及 LF (Line Feed)。在开头与结尾之间，则为通讯位置、功能码、数据内容和错误查核 LRC (Longitudinal Redundancy Check) 等。

RTU 模式：

Start	超过 10 ms 的静止时段
Slave Address	通讯地址：1-byte
Function	功能码：1-byte
Data (n-1)	数据内容：n-word = 2n-byte，n ≤ 10
…….	
Data (0)	
CRC	错误查核：1-byte
End 1	超过 10 ms 的静止时段

RTU (Remote Terminal Unit) 模式通讯的开头由一静止信号开始，结束为另一静止信号。在开头与结尾之间，则为通讯位置、功能码、数据内容和错误查核 CRC (Cyclical Redundancy Check) 等。

TCP 模式：

Start	TCP 封包开头
Transaction ID	传输顺序标识符：2-byte
Protocol ID	协定标识符：2-byte
Length	字段长度：2-byte
Unit ID	通讯地址：1-byte
Function	功能码：1-byte
Data (n-1)	数据内容：n-word = 2n-byte，n ≤ 10
…….	
Data (0)	
End 1	TCP 封包结尾

TCP (Transmission Control Protocol) 模式传输在一个完整的 TCP 封包内，开头为一个 TCP 封包开始，结束则为同一个封包结尾。在开头与结尾之间，则为传输顺序标识符号、协议标识符号、字段长度、通讯地址、功能码和数据内容等。

8

范例 1：功能码 03H，读取多个字组 (word)

以下的范例为主站下命令给 1 号从站：

读取由起始地址 0200H 开始的连续 2 个字组 (word) 资料。从站回复的数据中，地址 0200H 的内容为 00B1H 及地址 0201H 的内容为 1F40H，其中最大允许单次读取的笔数为 10 笔。LRC 与 CRC 的产生，将于以下章节说明。

ASCII 模式：

主站命令讯息：

Start	‘:’
Slave Address	‘0’
	‘1’
Function	‘0’
	‘3’
起始数据位置	‘0’
	‘2’
	‘0’
	‘0’
资料数目 (以 word 计算)	‘0’
	‘0’
	‘0’
	‘2’
LRC Check	‘F’
	‘8’
End 1	(0DH)(CR)
End 0	(0AH)(LF)

从站响应消息：

Start	‘:’
Slave Address	‘0’
	‘1’
Function	‘0’
	‘3’
资料数目 (以 byte 计算)	‘0’
	‘4'
起始数据地址 0200H 的内容	‘0’
	‘0’
	‘B’
	‘1’
第二笔数据地址 0201H 的内容	‘1’
	‘F’
	‘4’
	‘0’
LRC Check	‘E’
	‘8’
End 1	(0DH)(CR)
End 0	(0AH)(LF)

RTU 模式：

主站命令讯息：

Slave Address	01H
Function	03H
起始数据位置	02H (高字节)
	00H (低字节)
资料数目 (以 word 计算)	00H
	02H
CRC Check Low	C5H (低字节)
CRC Check High	B3H (高字节)

从站响应消息：

Slave Address	01H
Function	03H
资料数目 (以 byte 计算)	04H
起始数据地址 0200H 的内容	00H (高字节)
	B1H (低字节)
第二笔数据地址 0201H 的内容	1FH (高字节)
	40H (低字节)
CRC Check Low	A3H (低字节)
CRC Check High	D4H (高字节)

注：RTU 模式下的传输前与传输完成后，需有 10 ms 的静止时段。

TCP 模式：

主站命令讯息：

Transaction ID	00H (高字节)
	01H (低字节)
Protocol ID	00H (高字节)
	00H (低字节)
Length	00H (高字节)
	06H (低字节)
Unit ID	01H
Function	03H
起始数据位置	02H (高字节)
	00H (低字节)
资料数目 (以 word 计算)	00H
	02H

从站响应消息：

Transaction ID	00H (高字节)
	01H (低字节)
Protocol ID	00H (高字节)
	00H (低字节)
Length	00H (高字节)
	07H (低字节)
Unit ID	01H
Function	03H
资料数目 (以 byte 计算)	04H
起始数据地址 0200H 的内容	00H (高字节)
	B1H (低字节)
第二笔数据地址 0201H 的内容	1FH (高字节)
	40H (低字节)

注：TCP 模式下的 Length 指的是后面的字段长度。

8

范例 2：功能码 06H，写入单笔字组 (word)

以下的范例为主站下达写入命令给 1 号从站：

写入数据 0064H 到地址 0200H。从站在写入完成后则回复主站。LRC 与 CRC 的产生，将于以下章节说明。

ASCII 模式：

主站命令讯息：

Start	':'
Slave Address	'0'
	'1'
Function	'0'
	'6'
起始数据地址	'0'
	'2'
	'0'
	'0'
数据内容	'0'
	'0'
	'6'
	'4'
LRC Check	'9'
	'3'
End 1	(0DH)(CR)
End 0	(0AH)(LF)

从站响应消息：

Start	':'
Slave Address	'0'
	'1'
Function	'0'
	'6'
起始数据地址	'0'
	'2'
	'0'
	'0'
数据内容	'0'
	'0'
	'6'
	'4'
LRC Check	'9'
	'3'
End 1	(0DH)(CR)
End 0	(0AH)(LF)

RTU 模式：

主站命令讯息：

Address	01H
Slave Function	06H
起始数据地址	02H (高字节)
	00H (低字节)
数据内容	00H (高字节)
	64H (低字节)
CRC Check Low	89H (低字节)
CRC Check High	99H (高字节)

从站响应消息：

Address	01H
Slave Function	06H
起始数据地址	02H (高字节)
	00H (低字节)
数据内容	00H (高字节)
	64H (低字节)
CRC Check Low	89H (低字节)
CRC Check High	99H (高字节)

注：RTU 模式下的传输前与传输完成后，需有 10 ms 的静止时段。

TCP 模式：

主站命令讯息：

Transaction ID	00H (高字节)
	01H (低字节)
Protocol ID	00H (高字节)
	00H (低字节)
Length	00H (高字节)
	06H (低字节)
Unit ID	01H
Slave Function	06H
起始数据地址	02H (高字节)
	00H (低字节)
数据内容	00H (高字节)
	64H (低字节)

从站响应消息：

Transaction ID	00H (高字节)
	01H (低字节)
Protocol ID	00H (高字节)
	00H (低字节)
Length	00H (高字节)
	06H (低字节)
Unit ID	01H
Slave Function	06H
起始数据地址	02H (高字节)
	00H (低字节)
数据内容	00H (高字节)
	64H (低字节)

注：TCP 模式下的 Length 指的是后面的字段长度。

8

范例 3：功能码 10H，写入多个字组 (multiple words)

以下的范例为主站下达写入命令给 1 号从站：

写入 2 个字组 0BB8H 与 0000H 的数据到起始地址 0112H。即于位置 0112H 写入 0BB8H，于位置 0113H 写入 0000H，最大允许单次写入的笔数为 10 笔，从站在写入完成后则回复主站。LRC 与 CRC 的产生，将于以下章节说明。

ASCII 模式：

主站命令讯息：

Start	‘:’
Slave Address	‘0’
	‘1’
Function	‘1’
	‘0’
起始数据地址	‘0’
	‘1’
	‘1’
	‘2’
资料数目 (以 word 计算)	‘0’
	‘0’
	‘0’
	‘2’
资料数目 (以 byte 计算)	‘0’
	‘4’
第一笔数据内容	‘0’
	‘B’
	‘B’
	‘8’
第二笔数据内容	‘0’
	‘0’
	‘0’
	‘0’
LRC Check	‘1’
	‘3’
End 1	(0DH)(CR)
End 0	(0AH)(LF)

从站响应消息：

Start	‘:’
Slave Address	‘0’
	‘1’
Function	‘1’
	‘0’
起始数据地址	‘0’
	‘1'
	‘1’
	‘2’
资料数目	‘0’
	‘0’
	‘0’
	‘2’
LRC Check	‘D’
	‘A’
End 1	(0DH)(CR)
End 0	(0AH)(LF)

RTU 模式：

主站命令讯息：

Slave Address	01H
Function	10H
起始数据地址	01H (高字节)
	12H (低字节)
资料数目 (以 word 计算)	00H (高字节)
	02H (低字节)
资料数目 (以 byte 计算)	04H
第一笔数据内容	0BH (高字节)
	B8H (低字节)
第二笔数据内容	00H (高字节)
	00H (低字节)
CRC Check Low	FCH (低字节)
CRC Check High	EBH (高字节)

从站响应消息：

Slave Address	01H
Function	10H
起始数据地址	01H (高字节)
	12H (低字节)
资料数目 (以 word 计算)	00H (高字节)
	02H (低字节)
CRC Check Low	E0H (低字节)
CRC Check High	31H (高字节)

注：RTU 模式下的传输前与传输完成后，需有 10 ms 的静止时段。

TCP 模式：

主站命令讯息：

Transaction ID	00H (高字节)
	01H (低字节)
Protocol ID	00H (高字节)
	00H (低字节)
Length	00H (高字节)
	0BH (低字节)
Unit ID	01H
Function	10H
起始数据地址	01H (高字节)
	12H (低字节)
资料数目 (以 word 计算)	00H (高字节)
	02H (低字节)
资料数目	04H
第一笔数据内容	0BH (高字节)
	B8H (低字节)
第二笔数据内容	00H (高字节)
	00H (低字节)

从站响应消息：

Transaction ID	00H (高字节)
	01H (低字节)
Protocol ID	00H (高字节)
	00H (低字节)
Length	00H (高字节)
	06H (低字节)
Unit ID	01H
Function	10H
起始数据地址	01H (高字节)
	12H (低字节)
资料数目 (以 word 计算)	00H (高字节)
	02H (低字节)

注：TCP 模式下的 Length 指的是后面的字段长度。

8

LRC 与 CRC 传输错误检核

ASCII 通讯模式的错误检核使用 LRC (Longitudinal Redundancy Check)；RTU 通讯模式的错误检核使用 CRC (Cyclical Redundancy Check)；TCP 通讯模式由底层负责检核错误，不需额外加入 LRC 或 CRC 等传输错误检核。其算法说明如下。

LRC (ASCII 模式)：

Start	‘:’
Slave Address	‘7’
	‘F’
Function	‘0’
	‘3’
起始数据地址	‘0’
	‘5’
	‘C’
	‘4’
资料数目	‘0’
	‘0’
	‘0’
	‘1’
LRC Check	‘B’
	‘4’
End 1	(0DH)(CR)
End 0	(0AH)(LF)

将所有字节相加，舍去进位，然后取 2 的补码，即为 LRC 的算法。
以上例而言：
7FH + 03H + 05H + C4H + 00H + 01H = 14CH，舍去进位 1，只取 4CH。
4CH 取 2 的补码为：B4H。

CRC (RTU 模式):

CRC 侦误值计算以下列步骤说明:

1. 加载一个内容为 FFFFH 之 16-bit 缓存器，称之为「CRC」缓存器。
2. 将命令讯息的第一个字节与 16-bit CRC 缓存器的低字节进行 Exclusive OR 运算，并将结果存回 CRC 缓存器。
3. 检查 CRC 缓存器的最低位 (LSB)，若此位为 0，则 CRC 缓存器值右移一位；若此位为 1，则 CRC 缓存器值右移一位后，再与 A001H 进行 Exclusive OR 运算。此步骤需执行 8 次。
4. 请重复步骤 2 到步骤 3，直到所有字节皆被完全处理过，此时 CRC 缓存器的内容即是 CRC 侦误值。

说明：计算出 CRC 侦误值之后，在命令讯息中，须先填上 CRC 的低位，再填上 CRC 的高位。如 CRC 算法所算出的值为 3794H，则先将 94H 填入，再将 37H 填入，
如下表所示。

ARD	01H
CMD	03H
起始数据位置	01H (高字节)
	01H (低字节)
资料数目 (以 word 计算)	00H (高字节)
	02H (低字节)
CRC Check Low	94H (低字节)
CRC Check High	37H (高字节)

8

CRC 程序范例

以下范例以 C 语言产生 CRC 值。

```
unsigned char* data;
unsigned char length
/*此函数将回传 unsigned integer 型态之 CRC 值*/
unsigned int crc_chk(unsigned char* data, unsigned char length) {
        int j;
        unsigned int reg_crc=0xFFFF;

        while( length-- ) {
                reg_crc^= *data++;
                for (j=0; j<8; j++ ) {
                        if( reg_crc & 0x01 ) { /*LSB(bit 0 ) = 1 */
                                reg_crc = (reg_crc >> 1)^0xA001;
                        } else {
                                reg_crc = (reg_crc>>1);
                        }
                }
        }
        return reg_crc;
}
```

个人计算器通讯程序范例

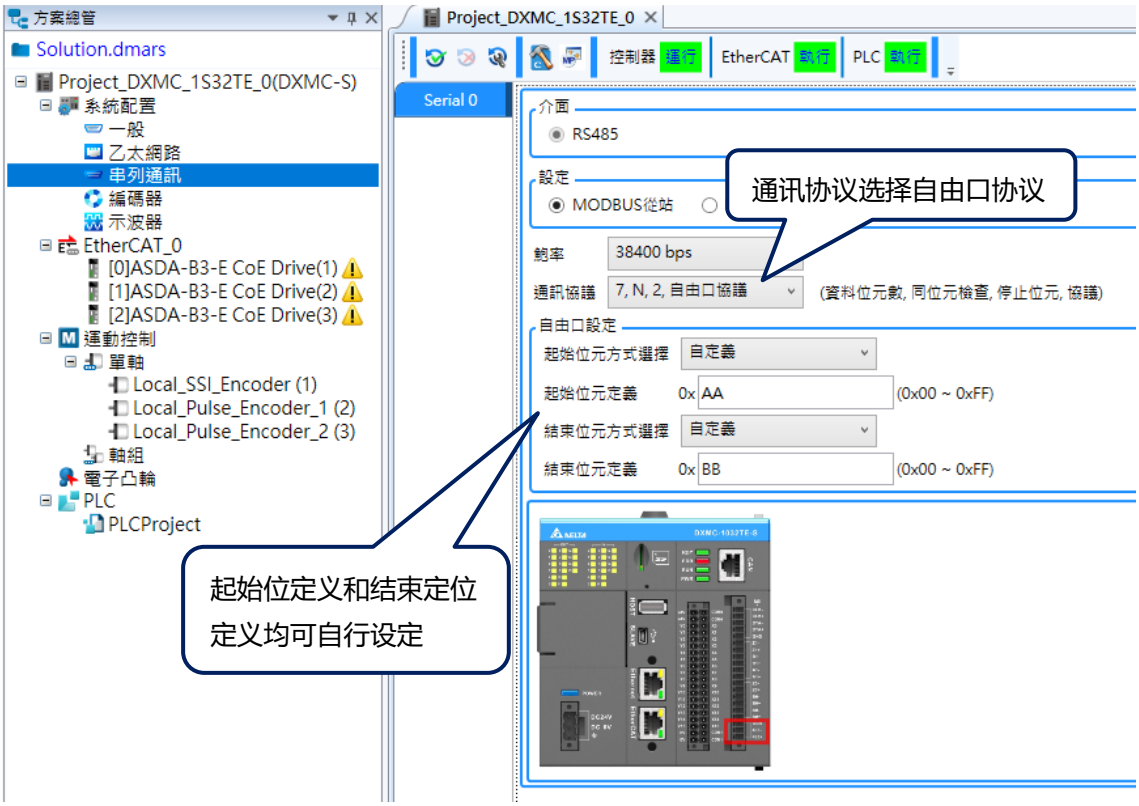
```
#include<stdio.h>
#include<dos.h>
#include<conio.h>
#include<process.h>
#define PORT 0x03F8    /*    the address of COM 1    */
#define THR 0x0000
#define RDR 0x0000
#define BRDL 0x0000
#define IER 0x0001
#define BRDH 0x0001
#define LCR 0x0003
#define MCR 0x0004
#define LSR 0x0005
#define MSR 0x0006
unsigned char rdat[60];
/* read 2 data from address 0200H of ASD with address 1 */
unsigned char tdat[60]={':','0','1','0','3','0','2','0','0','0','0','0','2','F','8','\r','\n'};
void main() {
int I;
outportb(PORT+MCR,0x08);          /*    interrupt enable    */
outportb(PORT+IER,0x01);                  /*    interrupt as data in    */
outportb(PORT+LCR,( inportb(PORT+LCR) | 0x80 ) );
/*    the BRDL/BRDH can be access as LCR.b7 == 1    */
outportb(PORT+BRDL,12);
outportb(PORT+BRDH,0x00);
outportb(PORT+LCR,0x06);          /* set protocol
```

```
                                        <7,E,1> = 1AH,          <7,O,1> = 0AH
                                        <8,N,2> = 07H       <8,E,1> = 1BH
                                        <8,O,1> = 0BH                          */
for( I = 0; I<=16; I++ ) {
      while( !(inportb(PORT+LSR) & 0x20) ); /*   wait until THR empty   */
      outportb(PORT+THR,tdat[I]);           /*   send data to THR   */
}
I = 0;
while( !kbhit() ) {
      if( inportb(PORT+LSR)&0x01 ) {   /*   b0==1, read data ready   */
            rdat[I++] = inportb(PORT+RDR); /*     read data from RDR   */
      }
}
}
```

8

8.3 自由口通讯协议（Freeport protocol)

DXMC控制器支持自由口通讯协议。若要设定自由口通讯协议，可于DMARS软件的左侧项目树中，选择**串行通讯**，于右方工作区设定**通讯协议**为**自由口协议**和**起始位**等相关参数。



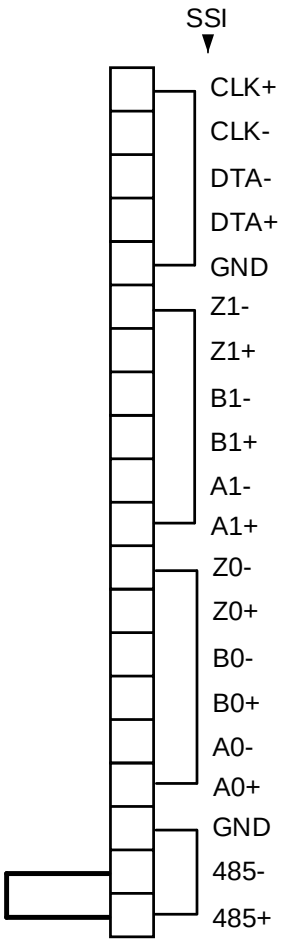
此通讯模式下，其协议是由用户自由定义，包含常见的指令、数据与校验码，甚至纯数据传输皆由用户自行规划。在此协议下，只提供接收（RX)、传送（TX）指令进行通讯分组交换，可用于Barcode reader、Printer、PLC device和Vision sensor等装置。

所允许的封包起始位与结束位以及最大封包长度的规格表如下：

	命令型式	RL 对应函式	封包格式			最大封包长度
			起始位	数据字节	结束位	
串行通讯 RS-485	传送 控制器→外部装置	FreePort.Serial.Tx	none or 00h ~ FFh (最大 1 byte)	最大 256 byte	none or 00h ~ FFh or CR+LF (最大 2 byte)	259 byte
	接收 控制器←外部装置	FreePort.Serial.Rx				
以太网络 Ethernet	TCP Server	FreePort.Eth.Rx		最大 1024 byte		1027 byte
	TCP Client	FreePort.Eth.Tx				

自发自收测试范例

硬件设置：将控制器上的外部通信端口 485-和 485+短路（Short circuit)。



Lua 脚本撰写：

```
--1.===设定开始/终止符，并清除接收缓冲区 ...
P309,Err1,Err2 = FreePort.Serial.Init(1,":",";")
--2.===自发自收测试 ...
FreePort.Serial.Tx(1,"3.1415")
DELAY(0.1)
local RxValid, RxData = FreePort.Serial.Rx(1)
my_print(string.format (">Rx: Rtn:0x%x Data:%s", RxValid,RxData))
--可收到 3.1415 字符串
```

8

DRL 相关函式

指令：FreePort.Eth.OpenAsServer

FreePort.Eth.OpenAsServer(SessionID,Port,SByte,EByte)

- 自由口 - 以太网络伺服端联机。
- SessionID：联机标识符 (型态：数字)。
- Port：网络端口号 (型态：数字)。
- SByte：起始位 (型态：数字或字符串)。nil：空字符；0 ~ 0xFF：自定义字符。
- EByte：结束位 (型态：数字或字符串)。nil：空字符；0 ~ 0xFF：自定义字符；CR_LF：换行符号。
- 回传自变量：#1：有效值 (0：成功；其他：错误)。

范例：

```
ret = FreePort.Eth.OpenAsServer(Dev1,9000,nil,";")
--以变量 Dev1 作为标识符，网络端口号 9000，起始位无，";"作为结束位，
  来建立伺服端联机
```

指令：FreePort.Eth.OpenAsClient

FreePort.Eth.OpenAsClient(SessionID,IPAddr,Port,SByte,EByte)

- 自由口 - 以太网络客户端联机。
- SessionID：联机标识符 (型态：数字)。
- 网络 IP 地址：字符串表示如"192.168.1.1" (型态：字符串)。
- Port：网络端口号，范围 3000 ~ 10000 (型态：数字)。
- SByte：起始位 (型态：数字或字符串)。nil：空字符；0 ~ 0xFF：自定义字符。
- EByte：结束位 (型态：数字或字符串)。nil：空字符；0 ~ 0xFF：自定义字符；CR_LF：换行符号。
- 回传自变量：#1：有效值 (0：成功；其他：错误)。

范例：

```
ret = FreePort.Eth.OpenAsClient(Dev2,"192.168.1.1",9003,nil,";")
--以变量 Dev2 作为标识符，IP 地址"192.168.1.1"，网络端口号 9003，起始位
  无，结束位";"，来建立伺服端联机
```

指令：FreePort.Eth.Tx

FreePort.Eth.Tx (SessionID,Msg)

- 自由口 - 以太网络传送。
- SessionID：联机标识符 (型态：数字)。
- Msg：传送数据 (型态：任意)。
- 回传自变量：#1：错误码 (0：成功；其他：错误)；#2 数据副本 (型态：任意)。

范例：

```
ret,Msg = FreePort.Eth.Tx (Dev1,EchoMsg)
--联机标识符 Dev1 的装置传送 EchoMsg 的数据
```

指令：FreePort.Eth.Rx

FreePort.Eth.Rx(SessionID)

- 自由口 - 以太网络接收。
- SessionID：联机标识符 (型态：数字)。
- 回传自变量：#1：错误码 (0：成功；其他：错误)；#2 数据 (型态：任意)；#3 资料笔数 (型态：数字)。

范例：

```
ret,Data,Sz = FreePort.Eth.Rx (Dev2)
--联机标识符 Dev2 的装置接收数据
```

指令：FreePort.Eth.Close

FreePort.Eth.Close(SessionID)

- 自由口 - 以太网络关闭指定联机。
- SessionID：联机标识符 (型态：数字)。
- 回传自变量：#1：错误码 (0：成功；其他：错误)。

范例：

```
ret = FreePort.ECM.Close(Dev1)   --关闭 Dev1 的联机
```

8

指令：FreePort.Eth.CloseAll

FreePort.Eth.CloseAll()

- 自由口 - 以太网络关闭全部联机。
- 回传自变量：#1：错误码 (0：成功；其他：错误)。

范例：

```
FreePort.Eth.CloseAll()  --关闭所有自由口联机
```

指令：FreePort.Eth.BuffClear

FreePort.Eth.BuffClear (SessionID)

- 自由口 - 以太网络清除传送和接收缓冲区。
- SessionID：联机标识符 (型态：数字)。
- 回传自变量：#1：错误码 (0：成功；其他：错误)。

范例：

```
ret = FreePort.Eth.BuffClear(Dev1)  --清除 Dev1 的缓冲区
```

指令：FreePort.Eth.GetSessions

FreePort.Eth.GetSessions()

- 自由口 - 以太网络取得目前联机标识符。
- 回传自变量：#1：错误码 (0：成功；其他：错误)；#2：已建立联机标识符 (型态：数组)。

范例：

```
ret,Data = FreePort.Eth.GetSessions()  --取得目前联机标识符
```

指令：FreePort.Eth.ConnectCheck

FreePort.Eth.ConnectCheck(SessionID)

- 自由口 - 以太网络确认联机是否建立。
- SessionID：联机标识符 (型态：数字)。
- 回传自变量：#1：错误码 (0：成功；其他：错误)。

范例：

```
ret = FreePort.Eth.ConnectCheck(Dev1)
--确认标识符 Dev1 的联机是否建立
```

指令：FreePort.Serial.Init

FreePort.Serial.Init(port, SrtByte, EndByte, Protocol)

- 自由口 – 串行埠初始化。
- port：埠号 (目前只支持埠号 1)。
- SByte：起始位 (型态：数字或字符串)。nil：空字符；0 ~ 0xFF：自定义字符。
- EByte：结束位 (型态：数字或字符串)。nil：空字符；0 ~ 0xFF：自定义字符；CR_LF：换行符号。
- Protocol：通讯协议 (型态：数字)。

  0x09：7, N, 2　　0x0A：7, E, 1　　0x0B：7, O, 1

  0x0C：8, N, 2　　0x0D：8, E, 1　　0x0E：8, O, 1

  0x10：7, N, 1　　0x12：7, E, 2　　0x14：7, O, 2

  0x17：8, N, 1　　0x1A：8, E, 2　　0x1D：8, O, 2
- 回传自变量：#1：错误码 (0：成功；其他：错误)；#2：写入控制器参数 P3-09 数值 (型态：数字)。

范例：

```
P309 = FreePort.Serial.Init(1, nil, CR_LF ,0xC)
--建立端口号 1，起始位 nil，结束位为换行符号，通讯协议为 8, n, 2 的联机
```

指令：FreePort.Serial.Tx

FreePort.Serial.Tx(port,Msg)

- 自由口 – 串行埠传送。
- port：埠号 (目前只支持埠号 1)。
- Msg：数据 (笔数单位：Byte，最大 256 笔)。
- 回传自变量：#1：错误码 (0：成功；其他：错误)；#2：资料副本。

范例：

```
valid, Msg = FreePort.Serial.Tx(1,"TEST")  --埠号 1 传送“Test”字符串
```

指令：FreePort.Serial.Rx

FreePort.Serial.Rx(port)

- 自由口 – 串行埠传送。
- port：埠号 (目前只支持埠号 1)。
- Msg：数据 (笔数单位：Byte，最大 256 笔)。
- 回传自变量：#1：错误码 (0：成功；其他：错误)；#2：资料；#3：资料笔数。

范例：

```
RxValid, RxData = FreePort.Serial.Rx(1)  --接收端口号 1 号的数据
```

(此页有意留为空白)

8

9 报警排除

本章节介绍各报警及其排除方式，使用者可利用此章节搜寻报警发生的原因和报警处置方法。

9.1 报警一览表……9-3
9.2 报警原因与处置……9-24
9.3 SDO 终止传输代码……9-104

9

报警代码长度为 32 bits，低位的 16 bits 作为事件码的索引，而高位的 16 bits 作为扩充码协助进一步的报警查询。下表为报警代码的详细格式说明。

32 bits 报警代码：0xXYZUABCD			
X：(代码等级)(来源类别)		**YZU**：扩充码	**ABCD**：事件码
(b1)：代码等级	(b2)(b3)(b4)：来源类别		
0：错误 1：警告	0：通用 1：运动核心 2：PLC 核心 3：RL 核心 4：NC 核心 5：保留	扩充码用于扩充事件码的描述，因此每个事件码的扩充码定义皆不同，详细定义请见报警一览表。	在来源 0 ~ 5 的区间内，事件码为一组唯一的代码，用于索引事件。详细定义请见报警一览表。
	6：运动总线	总线装置站号	来自总线装置的事件码
	7：使用者自定义	使用者自定义	使用者自定义的事件码

最高位的 bit 用来区别报警代码为「错误」或「警告」，「错误」为 0，「警告」为 1。
发生错误时，运动命令将立即停止，且在排除错误前，无法再下达新的命令；发生警告时，运动命令不会立即停止，但在排除警告前，无法再下达新的命令。

以下举几个报警代码的范例：

1. 0x10010815：$(0001)_2(0010815)_{16}$
   代码等级为 0：「错误」等级的代码。
   来源类别为 1：来源为运动核心。
   事件码为 0815：超出轴位置软正极限。
   扩充码为 001：经事件码查表后得知，代表为第 1 轴。
   此报警代码为来源为运动核心的错误，第 1 轴超出位置软正极限。

2. 0x90010815：$(1001)_2(0010815)_{16}$
   代码等级为 1：「警告」等级的代码。
   来源类别为 1：来源为运动核心。
   事件码为 0815：超出轴位置软正极限。
   扩充码为 001：经事件码查表后得知，代表为第 1 轴。
   此报警代码为来源为运动核心的警告，第 1 轴超出位置软正极限。

3. 0x60010185：$(0110)_2(0010185)_{16}$
   代码等级为 0：「错误」等级的代码。
   来源类别为 6：来源为运动总线。
   事件码为 0185：详细定义需参考运动总线所连接装置的使用手册。
   扩充码为 001：运动总线的站号号码，代表第 1 站。
   此报警代码为运动总线站号 1 装置发出报警，造成控制器发生错误，详细事件码定义请参考运动总线所连接装置的使用手册。

9.1 报警一览表

事件码	事件名称	扩充码	可能原因
0x0001	运动命令空操作	N/A	运动命令执行过程中，控制器发生非法操作。
0x0002	运动命令轴类型错误	N/A	运动命令的轴类型不正确。
0x0003	运动命令群组类型错误	N/A	运动命令的群组类型不正确。
0x0004	运动命令轴编号错误	N/A	运动命令的轴编号不正确。
0x0005	运动命令群组编号错误	N/A	运动命令的群组编号不正确。
0x0006	运动命令指令类型错误	N/A	运动命令类型不正确。
0x0007	轴运动命令缓存已满	N/A	运动命令缓存已满。
0x0008	群组运动命令缓存已满	N/A	运动命令缓存已满。
0x000B	运动命令生命周期错误	N/A	运动命令生命周期不正确。
0x000C	运动命令方向参数错误	N/A	运动命令内的方向参数不正确。
0x000D	运动命令坐标系参数错误	N/A	运动命令内的坐标系参数不正确。
0x000E	运动命令缓存模式错误	N/A	运动命令内的缓存模式不正确。
0x000F	运动命令重迭模式错误	N/A	运动命令内的重迭模式不正确。
0x0010	运动命令重迭缓存组合错误	N/A	运动命令内的缓存模式与重迭模式的组合不正确。
0x0011	运动命令主轴编号错误	N/A	主轴编号不正确。
0x0012	读写参数 index 错误	N/A	使用 MC_ReadParameter、MC_ReadBoolParameter、MC_WriteParameter、MC_WriteBoolParameter 时，运动参数编号（ParameterNumber）设定错误。
0x0013	运动命令分母为零	N/A	运动命令参数内作为分母使用的参数为零。
0x0014	功能块编号错误	N/A	使用的功能块数量超过最大限制。
0x0015	速度脚位小于最小值	YXU：轴编号	速度脚位输入值必须大于零。
0x0016	加速度脚位小于最小值	YXU：轴编号	加速度脚位输入值必须大于零。
0x0017	减速度脚位小于最小值	YXU：轴编号	减速度脚位输入值必须大于零。
0x0018	加加速度脚位小于最小值	YXU：轴编号	加加速度脚位输入值必须大于等于零。

9

事件码	事件名称	扩充码	可能原因
0x0019	功能块的 Any 脚位数据型态错误	N/A	Any 型态脚位（Value）所连接的变量长度小于欲读写的长度(Length)。
0x001A	速度倍率脚位输入错误	YXU：轴编号	速度倍率（VelFactor）必须大于等于零。
0x001B	加速倍率脚位输入错误	YXU：轴编号	加速度倍率（AccFactor）及加加速度倍率（JerkFactor）必须大于零。
0x001C	Operation Mode 不支援此运动命令	YXU：轴编号	轴目前处于的 Operation Mode 不支持此运动命令。
0x001D	轴正在执行群组命令	YXU：轴编号	指定运动轴正在执行群组运动命令。
0x001E	群组成员正在执行单轴命令	YXU：群组编号	运动群组的成员正在执行单轴运动命令。
0x001F	命令缓存处于忙碌状态	N/A	同时有多个 PLC task 或 Lua 项目对同一轴执行命令。
0x0020	执行运动命令时控制器处于错误的模式	N/A	当前控制器模式不能执行命令。
0x0040	凸轮应用功能码错误	N/A	使用了错误的凸轮应用功能码。
0x0041	无可用的凸轮 ID	N/A	已经没有剩余的凸轮 ID 可用，最大可用 ID 数量为 256 个。
0x0042	凸轮 ID 已存在	N/A	指定的凸轮 ID 已经存在。
0x0043	凸轮 ID 不存在	N/A	指定的凸轮 ID 不存在。
0x0044	凸轮 CamIn scaling 参数错误	N/A	CamIn 使用的 scaling 参数错误。
0x0045	建立凸轮表时无法覆写	N/A	建立凸轮表时，没有开启覆写功能。
0x0046	凸轮表类型参数错误	N/A	在建立凸轮的操作中，表类型参数错误。
0x0047	凸轮表主轴位置参数错误	N/A	在建立凸轮的操作中或是新增、修改凸轮表点位时，主轴位置错误。
0x0048	凸轮表起始点位编号错误	N/A	新增、修改、删除凸轮表点位的操作中，起始点位编号错误。
0x0049	凸轮表超出范围	N/A	新增凸轮表点位的操作超出一个凸轮表预设最大点位。

事件码	事件名称	扩充码	可能原因
0x004A	欲使用的凸轮表是空的	N/A	于 CamIn 新增、修改、删除凸轮表点位的操作中，欲使用的凸轮表是空的。
0x004B	凸轮表已满	N/A	于建立凸轮表的操作中，无剩余空间可以使用。
0x004C	凸轮表在线切换功能，ActiveMode 参数错误	N/A	设置在线切换功能操作中，ActiveMode 参数不正确。
0x004E	凸轮表使用中	N/A	删除凸轮表操作中，欲操作的表格正在使用中。
0x004F	凸轮表档案不存在	N/A	指定路径内并不存在用户输入的 ECAM 文件名。
0x0050	凸轮表档案数据错误	N/A	ECAM 档案内容长度有问题，系统无法读取内容。
0x0051	凸轮表的 ID 为空白	N/A	没有输入 TableID。
0x0052	凸轮表档名为空白	N/A	没有输入 FileName。
0x0053	追剪功能返回距离不足	N/A	建立追剪表格操作中，因同步区太长或切长太短导致追剪功能返回距离不足。
0x0054	旋切调整区域小于零	N/A	建立旋切表格操作中，因切长太短或是同步区太长导致旋切调整区域小于零。
0x0055	凸轮表档案路径错误	N/A	用户输入的路径有误。
0x0056	凸轮表档案无法覆写	N/A	档案无法覆写。
0x0057	挺杆点主轴位置错误	N/A	挺杆点设置操作中，主轴位置参数不正确。
0x0058	挺杆点方向错误	N/A	挺杆点设置操作中，挺杆点方向参数不正确。
0x005A	旋切同步区起始位置错误	N/A	建立旋切表格操作中，同步区起始位置参数不正确。
0x005B	旋切同步区结束位置错误	N/A	建立旋切表格操作中，同步区结束位置参数不正确。
0x005C	旋切主轴或从轴为旋转轴	N/A	CamIn 操作中，旋切主轴或从轴为旋转轴。
0x005D	凸轮比例尺参数为零	N/A	凸轮比例尺功能操作中，比例尺参数为零。

9

事件码	事件名称	扩充码	可能原因
0x005E	凸轮比例尺参数小于零	N/A	凸轮比例尺功能操作中，比例尺参数小于零。
0x005F	凸轮比例尺功能 MasterScalingMode 错误	N/A	凸轮比例尺功能操作中，MasterScalingMode 参数不正确。
0x0060	凸轮比例尺模式错误	N/A	凸轮比例尺功能操作中或在线切换功能操作中，比例尺模式参数不正确。
0x0061	旋切同步区速度比例为零	N/A	建立旋切凸轮表操作中或修改旋切表格速度比例操作中，同步区速度比例参数为零。
0x0062	凸轮表不是旋切凸轮表	N/A	修改旋切表格速度比例操作中，欲修改的表格非旋切凸轮表。
0x0063	凸轮表类型错误	N/A	建立凸轮表操作中，类型参数不正确。
0x0064	凸轮挺杆点数量超出范围	N/A	凸轮挺杆点设置操作中，挺杆点数量参数超出范围。
0x0065	CamIn 使用的 ID 参数错误	N/A	凸轮 CamIn 操作中，CamIn 使用的 TableID 参数不正确。
0x0066	主轴来源参数错误	N/A	凸轮 CamIn 操作中，CamIn 使用的主轴来源参数不正确。
0x0067	CamIn ActiveMode 错误	N/A	凸轮 CamIn 操作中，ActiveMode 参数不正确。
0x0068	CamIn BufferMode 错误	N/A	凸轮 CamIn 操作中，BufferMode 参数不正确。
0x0069	CamIn StartMode 错误	N/A	凸轮 CamIn 操作中，StartMode 参数不正确。
0x006A	凸轮表数据数量错误	N/A	建立凸轮表操作中，凸轮表数据数量参数不正确。
0x006B	追剪主轴位置参数错误	N/A	建立追剪表格操作中，主轴位置参数不正确。
0x006C	追剪从轴等待位置参数错误	N/A	建立追剪表格操作中，从轴等待位置参数不正确。
0x006D	追剪从轴同步位置参数错误	N/A	建立追剪表格操作中，从轴同步位置参数不正确。
0x006E	ECAM 点位曲线型态错误	N/A	ECAM 点位曲线型态不支持。

事件码	事件名称	扩充码	可能原因
0x006F	飞剪曲线刀数为零	N/A	飞剪曲线刀数为零。
0x0070	ECAM 表格点位数量错误	N/A	ECAM 造表功能块造表长度设定大于输入的点位数据数组。
0x0071	挺杆点群组 ID 错误	N/A	挺杆点群组 ID 大于 8。
0x0072	挺杆点主轴位置重复	N/A	挺杆点的主轴位置有两点相同。
0x0073	凸轮表速度建表的速度参数错误	N/A	速度百分比加总不等于 100%。
0x0074	建立伺服追飞剪凸轮表错误	N/A	追飞剪造表参数不符合伺服规则。
0x0080	未啮合齿轮便脱离齿轮	N/A	目标轴尚未 InGear 时使用 MC_GearOut。
0x0081	啮合齿轮没有主轴	N/A	使用 MC_GearIn 时并没有设定主轴。
0x0082	GearInPos 目标速度为零	YXU：轴编号	MC_GearInPos 功能块脚位设定造成从轴目标速度为零。
0x0083	GearInPos 最终速度为零	YXU：轴编号	MC_GearInPos 功能块脚位设定造成从轴最终速度为零。
0x0084	GearInPos 设定错误	YXU：轴编号	GearInPos 参数设定错误，造成从轴无法完成动作。
0x0100	轴在错误停止（ErrorStop）状态	YXU：轴编号	当轴发生错误后，轴会进入 ErrorStop 的状态。
0x0101	轴没有使能	YXU：轴编号	该轴不在使能状态。
0x0102	轴正在停止	YXU：轴编号	轴正在进行停止命令。
0x0103	轴正在执行回原	YXU：轴编号	轴正在执行回原命令。
0x0104	轴处于同步状态	YXU：轴编号	轴正处于同步状态。
0x0106	轴不在静止状态	YXU：轴编号	轴不处于静止状态。
0x0107	轴回原失败	YXU：轴编号	轴执行回原命令失败。
0x0108	多个 MC_Power 操作同一轴	YXU：轴编号	PLC 程序内，使用了多个 MC_Power 控制同一个轴。
0x0109	轴不在同步状态	YXU：轴编号	轴不在同步状态。
0x010A	轴不在 EtherCAT 模式	YXU：轴编号	伺服轴控制模式不为 EtherCAT。
0x010B	回原命令超时	YXU：轴编号	回原命令下达后超过 5 秒未被执行。
0x0801	轴没有回原	YXU：轴编号	轴的回原点定义失效。
0x0815	超出轴位置软正极限	YXU：轴编号	轴位置超出软件正极限。

9

事件码	事件名称	扩充码	可能原因
0x0816	超出轴位置软负极限	YXU：轴编号	轴位置超出软件负极限。
0x0817	超出轴速度限制	YXU：轴编号	轴速度超出软件限制。
0x0818	超出轴加速度限制	YXU：轴编号	轴加速度超出软件限制。
0x0819	超出轴减速度限制	YXU：轴编号	轴减速度超出软件限制。
0x081A	超出轴加加速度限制	YXU：轴编号	轴加加速度超出软件限制。
0x081B	超出轴位置误差限制	YXU：轴编号	轴位置误差超出软件限制。
0x0917	超出轴速度警告	YXU：轴编号	轴速度超出软件警告值。
0x0918	超出轴加速度警告	YXU：轴编号	轴加速度超出软件警告值。
0x0919	超出轴减速度警告	YXU：轴编号	轴减速度超出软件警告值。
0x091A	超出轴加加速度警告	YXU：轴编号	轴加加速度超出软件警告值。
0x091B	超出轴位置误差警告数值	YXU：轴编号	轴位置误差超出软件警告值。
0x091C	超出轴扭矩警告正极限	YXU：轴编号	轴扭矩超出软件警告值。
0x091D	超出轴扭矩警告负极限	YXU：轴编号	轴扭矩超出软件警告值。
0x091E	利用 SDO 切换轴运行模式失败	YXU：轴编号	利用 SDO 切换轴运行模式失败。
0x091F	上一笔切换轴运行模式的 SDO 命令尚在执行	YXU：轴编号	上一笔切换轴运行模式的 SDO 命令尚在执行。
0x0920	回原参数超界	YXU：轴编号	回原参数设定超出范围。
0x0921	指定回原模式不支持	YXU：轴编号	回原模式（HomeMode）不支持。
0x0922	回原第一段速超界	YXU：轴编号	回原第一段速（VelocityFast）超出极限。
0x0923	回原第二段速超界	YXU：轴编号	回原第二段速（VelocitySlow）超出极限。
0x0924	回原加速度超界	YXU：轴编号	回原加速度（Acceleration）超出极限。
0x0925	单轴命令 BlendingMode 加（减）速距离不足	YXU：轴编号	单轴命令以 BlendingMode 衔接时，加（减）速距离不足。
0x0926	多轴命令 BlendingMode 加（减）速距离不足	YXU：轴编号	多轴命令以 BlendingMode 衔接时，加（减）速距离不足。
0x0927	位置命令溢位	YXU：轴编号	位置命令已超过双精度浮点数之最大值，因此无法取得轴实际位置。
0x1000	群组没有启用	Y：保留 XU：群组编号	对象群组没有启用。

事件码	事件名称	扩充码	可能原因
0x1001	群组处于错误停止 (ErrorStop) 状态	Y：保留 XU：群组编号	群组处于 ErrorStop 状态。
0x1002	群组停止中	Y：保留 XU：群组编号	群组正在进行停止命令中。
0x1003	群组回原中	Y：保留 XU：群组编号	该群组正在进行回原命令中。
0x1004	该群组成员不存在	Y：保留 XU：群组编号	群组选择成员轴时，发现该成员轴不存在。
0x1005	群组没有回原	Y：保留 XU：群组编号	群组还没有成功执行过回原命令。
0x1006	群组命令 BlendingMode 加（减）速距离不足	YXU：群组编号	群组命令以 BlendingMode 衔接时，加（减）速距离不足。
0x1007	群组 PLC 状态并未处于 Standby	YXU：群组编号	执行群组回原时，群组成员并未全部使能。
0x1010	群组速度超界	YXU：群组编号	群组命令或回授速度超过最大限制。
0x1011	群组加速度超界	YXU：群组编号	群组命令或回授加速度超过最大限制。
0x1012	群组减速度超界	YXU：群组编号	群组命令或回授减速度超过最大限制。
0x1013	群组加加速度超界	YXU：群组编号	群组命令或回授加加速度超过最大限制。
0x1020	群组速度超过警告值	YXU：群组编号	群组命令或回授速度超过警告。
0x1021	群组加速度超过警告值	YXU：群组编号	群组命令或回授加速度超过警告。
0x1022	群组减速度超过警告值	YXU：群组编号	群组命令或回授减速度超过警告。
0x1023	群组加加速度超过警告值	YXU：群组编号	群组命令或回授加加速度超过警告。
0x1024	群组点对点运动速度超界	YXU：群组编号	群组执行点对点运动时超出成员轴最大速度。
0x1025	群组点对点运动加速度超界	YXU：群组编号	群组执行点对点运动时超出成员轴最大加速度。
0x1026	群组点对点运动减速度超界	YXU：群组编号	群组执行点对点运动时超出成员轴最大减速度。
0x1027	群组点对点运动加加速度超界	YXU：群组编号	群组执行点对点运动时超出成员轴最大加加速度。

9

事件码	事件名称	扩充码	可能原因
0x1028	群组点对点运动速度超出警告值	YXU：群组编号	群组执行点对点运动时超出成员轴速度警告值。
0x1029	群组点对点运动加速度超出警告值	YXU：群组编号	群组执行点对点运动时超出成员轴加速度警告值。
0x102A	群组点对点运动减速度超出警告值	YXU：群组编号	群组执行点对点运动时超出成员轴减速度警告值。
0x102B	群组点对点运动加加速度超出警告值	YXU：群组编号	群组执行点对点运动时超出成员轴加加速度警告值。
0x1120	Robot 三角运算超出范围	Y：保留 XU：群组编号	计算逆向运动学时，发生三角运算超出范围。
0x1121	Robot 三角运算未定义	Y：保留 XU：群组编号	计算逆向运动学时，发生三角运算结果为值域未定义。
0x1122	Robot 手系错误	Y：保留 XU：群组编号	Robot 手系错误。
0x1123	Robot 肩部奇异点	Y：保留 XU：群组编号	插值过程中产生奇异点。
0x1124	Robot 肘部奇异点	Y：保留 XU：群组编号	插值过程中产生奇异点。
0x1125	Robot 腕部奇异点	Y：保留 XU：群组编号	插值过程中产生奇异点。
0x1126	Robot 成员轴 1 超出极限	Y：保留 XU：群组编号	成员轴超出位置极限。
0x1127	Robot 成员轴 2 超出极限	Y：保留 XU：群组编号	成员轴超出位置极限。
0x1128	Robot 成员轴 3 超出极限	Y：保留 XU：群组编号	成员轴超出位置极限。
0x1129	Robot 成员轴 4 超出极限	Y：保留 XU：群组编号	成员轴超出位置极限。
0x112A	Robot 成员轴 5 超出极限	Y：保留 XU：群组编号	成员轴超出位置极限。
0x112B	Robot 成员轴 6 超出极限	Y：保留 XU：群组编号	成员轴超出位置极限。
0x112C	Robot 成员轴 7 超出极限	Y：保留 XU：群组编号	成员轴超出位置极限。

事件码	事件名称	扩充码	可能原因
0x112D	Robot 成员轴 8 超出极限	Y：保留 XU：群组编号	成员轴超出位置极限。
0x112E	Robot 成员轴 9 超出极限	Y：保留 XU：群组编号	成员轴超出位置极限。
0x112F	Robot 成员轴 10 超出极限	Y：保留 XU：群组编号	成员轴超出位置极限。
0x1130	Robot 成员轴 11 超出极限	Y：保留 XU：群组编号	成员轴超出位置极限。
0x1131	Robot 成员轴 12 超出极限	Y：保留 XU：群组编号	成员轴超出位置极限。
0x1140	Robot UF 教导法错误	Y：保留 XU：群组编号	使用了不支持的 UF 教导法。
0x1141	Robot UF 教导共线错误	Y：保留 XU：群组编号	UF 教导点位存在共线状况。
0x1142	Robot UF 出现零向量	Y：保留 XU：群组编号	UF 其中某一轴向量为零向量。
0x1143	Robot UF 比例尺为 0	Y：保留 XU：群组编号	UF 的比例尺被填为 0。
0x1144	Robot UF 编号错误	Y：保留 XU：群组编号	使用了不合法的 UF 编号。
0x1145	Robot UF 未教导	Y：保留 XU：群组编号	使用了尚未教导的 UF。
0x1146	Robot UF 旋转角度错误	Y：保留 XU：群组编号	UF 教导过程中，产生不合法的角度设定。
0x1160	Robot TF 教导法错误	Y：保留 XU：群组编号	使用了不支持的 TF 教导法。
0x1161	Robot TF 教导共平面错误	Y：保留 XU：群组编号	TF 教导点位存在共平面状况。
0x1162	Robot TF 编号错误	Y：保留 XU：群组编号	使用了不合法的 TF 编号。
0x1163	Robot TF 未教导	Y：保留 XU：群组编号	使用了尚未教导的 TF。
0x1164	Robot 命令数据长度不正确	Y：保留 XU：群组编号	命令的数据长度太短。

事件码	事件名称	扩充码	可能原因
0x1165	Robot 重迭参数错误	Y：保留 XU：群组编号	使用了不合法的 Robot 重迭参数。
0x1166	Robot 超出工作范围	YXU：群组编号	Robot 超出工作范围。
0x1167	Robot 进入工作禁区	YXU：群组编号	Robot 进入工作禁区。
0x1168	工作范围设定失败	YXU：群组编号	工作范围设定有误造成设定失败。
0x1169	工作禁区设定失败	YXU：群组编号	工作禁区设定有误造成设定失败。
0x3000	参数空操作	N/A	参数存取命令执行过程中， 控制器发生非法操作。
0x3001	参数操作对象错误	N/A	参数存取命令对象类型错误。
0x3004	参数操作对象编号错误	N/A	参数存取命令对象编号错误。
0x3005	参数操作轴编号错误	N/A	参数存取命令轴编号错误。
0x3006	参数操作群组编号错误	N/A	参数存取命令群组编号错误。
0x3007	参数操作功能码错误	N/A	参数操作功能码错误。
0x3008	参数命令接收缓存已满	N/A	短时间内下达太多参数操作命令。
0x3009	参数命令接收缓存为空	N/A	参数命令接收缓存为空。
0x300B	参数命令传送缓存为空	N/A	参数命令传送缓存为空。
0x300C	参数数值超出最大值	N/A	参数输入数值超出最大值。
0x300D	参数数值超出最小值	N/A	参数输入数值超出最小值。
0x300E	参数编号错误	N/A	参数编号错误。
0x300F	参数不可写入	N/A	操作对象参数不可进行写入操作。
0x3010	参数不可在伺服使能时修改	N/A	操作对象参数不可在伺服使能时进行修改。
0x3011	SDO pReadData 错误	N/A	使用 SDO 功能块时，pReadData 脚位没有接变量。
0x3012	参数操作 pReadData 错误	N/A	使用系统参数功能块时， pReadData 脚位没有接变量。
0x3013	参数数值错误	N/A	参数输入数值不在合法范围内。
0x3014	参数不可读取	N/A	操作对象参数不可进行读取操作。
0x3015	PDO pReadData 错误	N/A	使用 PDO 功能块时，pReadData 脚位没有接变量。
0x3016	参数读写超时	N/A	参数读写超时。
0x3030	使用配置文件发生空操作	N/A	使用配置文件过程中，控制器发生非法操作。

事件码	事件名称	扩充码	可能原因
0x3039	配置文件档案读取失败	N/A	使用配置文件时，读取失败。
0x303A	配置文件内容与实际硬件不符	YXU：装置 ID	无法在目前控制器连接的装置内，找到符合配置文件内设定的装置。
0x303B	配置文件使用了备份档	N/A	控制器使用配置文件时，由于找不到正确的配置文件，因此使用了 备份的配置文件。
0x303C	控制器配置初始化失败	N/A	由于配置文件内容不正确，控制器使用配置文件进行初始化时，初始化失败。
0x303D	Motion 核心初始化失败	N/A	建立 Motion 对象时，由于配置文件的内容有误而发生错误。
0x303E	建立轴对象失败	YXU：轴编号	建立轴对象时，由于配置文件的内容有误而发生错误。
0x303F	建立群组对象失败	YXU：群组编号	建立群组对象时，由于配置文件的内容有误而发生错误。
0x3040	取得配置文件失败	N/A	取得配置文件失败。
0x3041	配置文件缺少主要参数	N/A	配置文件缺少主要参数。
0x3042	装置初始化失败	N/A	装置配置文件错误造成装置初始化失败。
0x3043	配置文件版本错误	N/A	韧体与配置文件版本不匹配。
0x3044	读取编码器形态失败	N/A	透过 SDO 通讯读取伺服编码器形态时失败。
0x3045	伺服电子齿轮比写入失败	N/A	透过 SDO 通讯写入伺服电子齿轮比时失败。
0x3046	读取伺服速度回授单位失败	N/A	透过 SDO 通讯读取伺服速度回授单位时失败。
0x3047	伺服软极限写入失败	N/A	透过 SDO 通讯写入伺服软极限时失败。
0x3048	伺服速度限制写入失败	N/A	透过 SDO 通讯写入伺服速度限制时失败。
0x3049	错误的内建 MODBUS 站号配置	N/A	内建 MODBUS 站号有重复的站号设定。
0x304A	配置文件使用了错误的机型	N/A	配置文件内的机型与当前控制器的机型不一致。

9

事件码	事件名称	扩充码	可能原因
0x3064	监控轴编号错误	YXU：轴编号	进行轴监控操作时，轴编号设定错误。
0x3065	监控群组编号错误	YZU：群组编号	进行群组监控操作时，群组编号设定错误。
0x3081	控制器状态切换失败	N/A	控制器进行状态切换时，由于硬件配置不符合等因素，而造成切换失败。
0x3082	无法在非 Edit Mode 重设 EtherCAT	N/A	只有在 Edit Mode 才能进行 EtherCAT 联机操作。
0x3083	无法在非 Edit Mode 进行 EtherCAT 重新扫站	N/A	只有在 Edit Mode 才能进行 EtherCAT 联机操作。
0x3084	无法在非 Edit Mode 设定 EtherCAT 从站别名	N/A	只有在 Edit Mode 才能进行 EtherCAT 联机操作。
0x3085	控制器共享内存初始失败	N/A	控制器建立共享内存时发生错误。
0x3086	EtherCAT 共享内存初始失败	N/A	EtherCAT 建立共享内存时发生错误。
0x3087	在非 Edit Mode 重新启动 EtherCAT	N/A	只有在 Edit Mode 才能进行 EtherCAT 联机操作。
0x3088	PLC 程序使用了版本不匹配的 FwLib	N/A	控制器韧体版本与 PLC FwLib 不匹配。
0x3089	控制器无法执行运动命令	N/A	控制器目前的模式无法执行运动命令。
0x308A	报警码重置超时	N/A	报警码无法清除。
0x308C	控制器目标状态错误	N/A	控制器目标状态不支持。
0x308D	RTC 电池低电压提醒	N/A	RTC 电池达到低电压警示范围。
0x308E	控制器低电压提醒	N/A	系统侦测到低电压，但目前电压仍是正常。
0x308F	控制器掉电储存失败	N/A	控制器掉电处理流程未完成。
0x3090	控制器异常	N/A	控制器遭遇异常情况。
0x3102	反矩阵运算失败	N/A	执行反矩阵运算时失败。
0x31A0	错误的圆弧模式	YXU：群组编号	使用圆弧命令时，使用了错误的圆弧模式。
0x31A1	圆弧起始点与中间点相同	YXU：群组编号	圆弧起始点与中间点为同一点，无法成圆弧。

事件码	事件名称	扩充码	可能原因
0x31A2	圆弧起始点与结束点为同一点	YXU：群组编号	圆弧起始点与结束点为同一点，无法成圆弧。
0x31A3	圆弧中间点与结束点为同一点	YXU：群组编号	圆弧中间点与结束点为同一点，无法成圆弧。
0x31A4	圆面积过小	YXU：群组编号	圆面积过小，无法成圆。
0x31A5	圆弧起始点、中间点与结束点共线	YXU：群组编号	圆弧起始点、中间点与结束点共线，无法成圆弧。
0x31A6	圆弧平面没有法向量	YXU：群组编号	无法从计算出的圆弧平面找到法向量。
0x31A7	圆弧两点一圆心共线	YXU：群组编号	计算圆弧使用的两点与圆心共线，造成无法算出圆弧。
0x31A8	圆半径太短	YXU：群组编号	圆半径太短，无法成圆。
0x31A9	圆弧起始点与圆心相同	YXU：群组编号	圆弧命令的起始点与圆心相同，无法成圆。
0x31AA	圆弧结束点与圆心相同	YXU：群组编号	圆弧命令的结束点与圆心相同，无法成圆。
0x31AB	圆平面没有 X 向量	YXU：群组编号	无法从计算出的圆弧平面找到 X 向量。
0x4000	PLC 发生空操作	N/A	PLC 核心运行中，发生非法操作。
0x4001	PLC system function lib 初始失败	N/A	PLC 核心初始化 system function lib 失败。
0x4002	PLC shared memory lib 初始失败	N/A	PLC 核心初始化 shared memory lib 失败。
0x4003	PLC IO lib 初始失败	N/A	PLC 核心初始化 IO lib 失败。
0x4004	在错误的 PLC task 使用运动命令	N/A	在无法使用运动命令的 PLC task 使用了运动命令。
0x4006	PLC task 运动命令缓存已满	N/A	在与 Motion 不同步的 PLC task 一个周期内填入太多运动命令。
0x4007	PLC Fieldbus lib 初始失败	N/A	PLC 核心初始化 Fieldbus lib 失败。
0x4008	PLC 建立 eclr 控件失败	N/A	PLC 核心建立 eclr 控件失败。
0x4009	PLC 建立 eclr 应用领域失败	N/A	PLC 核心建立 eclr 应用领域失败。
0x400A	PLC 建立 eclr 系统领域失败	N/A	PLC 核心建立 eclr 系统领域失败。
0x400B	PLC 建立 eclr 数据 heap 失败	N/A	PLC 核心建立 eclr 数据 heap 失败。

9

事件码	事件名称	扩充码	可能原因
0x400C	PLC 建立 eclr 程序代码 heap 失败	N/A	PLC 核心建立 eclr 程序代码 heap 失败。
0x400D	PLC 冷启加载失败	N/A	PLC 核心以冷启动的方式加载程序时失败。
0x400E	PLC 暖启加载失败	N/A	PLC 核心以热启动的方式加载程序时失败。
0x400F	PLC 静止加载失败	N/A	PLC 核心以静止启动的方式加载程序时失败。
0x4010	未知的 PLC 启动加载模式	N/A	PLC 核心使用了未知的 PLC 启动加载模式。
0x4011	未知的 PLC 启动法则	N/A	PLC 核心使用了未知的 PLC 启动法则。
0x4012	PLC 任务超出最大数量	N/A	PLC 任务配置超过 10 组。
0x4013	PLC 事件任务超出最大数量	N/A	PLC 事件任务超出 6 组 (包含 Event0)。
0x4014	PLC 事件 ID 重复定义	N/A	PLC 事件 ID 重复定义。
0x4015	PLC 使用了错误的事件 ID	N/A	PLC 使用了错误的事件 ID。
0x4016	PLC 事件任务配置文件不存在	N/A	PLC 事件任务配置文件不存在。
0x4017	PLC 事件任务配置内容错误	N/A	PLC 事件任务配置内容错误。
0x4018	PLC 事件任务未进行配置	N/A	PLC 事件任务未进行配置。
0x4019	PLC 核心冷启动失败	N/A	PLC 核心冷启动失败。
0x401A	PLC 核心停止失败	N/A	PLC 核心停止失败。
0x401B	触发 PLC 的 DI 编号不正确	N/A	触发 PLC 的 DI 编号不正确。
0x401C	PLC DMC function lib 初始失败	N/A	函式库版本不匹配。
0x4100	未知的 PLC 例外	N/A	PLC 发生了未知的例外。
0x4101	PLC 程序发生字符串错误	N/A	PLC 程序中存在错误的字符串操作，例如存取超出范围等。
0x4102	PLC 任务运行时间超出 watch dog 设定值	N/A	PLC 任务所需运行时间过长。
0x4103	执行 PLC 任务超出 CPU 负荷	N/A	PLC 任务所需运行时间过长。
0x4104	PLC 核心发生系统例外	N/A	PLC 核心发生系统例外。

事件码	事件名称	扩充码	可能原因
0x4105	PLC 控制命令不支持	N/A	透过 SDV 控制 PLC 时，命令功能码不支持。
0x4106	停止 PLC IO 更新流程失败	N/A	PLC IO 更新配置不正确。
0x5000	所有的探针都在使用中	N/A	所有的探针都在使用中。
0x5001	触发 ID 已被使用	N/A	DMC_TouchProbe 功能块指定的探针正在使用中，使用中的探针无法重复使用。
0x5002	触发 ID 没有被使用	N/A	MC_AbortTrigger 欲中断的探针为空闲状态。
0x5003	探针触发事件轴 (Axis) 错误	N/A	设定的触发事件轴 (Axis) 错误。
0x5004	触发模式 (TriggerType) 超出范围	N/A	设定的触发模式 (TriggerType) 超出范围。
0x5005	探针驱动器模式 (Drive Mode) 只能使用实体轴	N/A	使用 DMC_TouchProbe 功能块时，虚拟轴和实体编码器轴不支持驱动器模式 (Drive Mode)。
0x5006	触发 ID (TriggerID) 超出范围	N/A	触发 ID (TriggerID) 超过各轴探针数量。
0x5007	驱动器模式 (Drive Mode) 触发 ID (TriggerID) 超出范围	N/A	驱动器模式 (Drive Mode) 下，触发 ID (TriggerID) 超出范围。
0x5008	控制器模式 (Controller Mode) 触发 ID (TriggerID) 超出范围	N/A	控制器模式 (Controller Mode) 下，触发 ID (TriggerID) 超出范围。
0x5009	触发类型 (TriggerType) 超出范围	N/A	所设定的触发类型 (TriggerType) 不支持。
0x500A	驱动器模式 (Drive Mode) 只能使用本地 DI 触发	N/A	在驱动器模式 (Drive Mode) 下，仅支持本地 DI 触发。
0x500B	探针 DI 编号 (DINumber) 超出范围	N/A	探针 DI 编号 (DINumber) 超出支持的范围。
0x500C	PLC 变量触发 DI 编号 (DINumber) 必须为零	N/A	在 PLC 变量触发模式下，探针 DI 编号 (DINumber) 必须为零。
0x500D	驱动器模式 (Drive Mode) 下 DI 编号 (DINumber) 设定错误	N/A	在驱动器模式 (Drive Mode) 下，探针 DI 编号 (DINumber) 必须在 1 ~ 2 之间。
0x500E	探针数据记录总类 (RecordedDate) 设定超出范围	N/A	设定的数据记录总类 (RecordedDate) 不支持。

事件码	事件名称	扩充码	可能原因
0x500F	驱动器模式 (Drive Mode) 下探针数据记录总类 (RecordedDate) 设定错误	N/A	在驱动器模式 (Drive Mode) 下，探针数据记录总类 (RecordedDate) 必须在 0 ~ 2 之间。
0x5010	控制器模式 (Controller Mode) 使用本地 DI 触发型态时，探针数据记录总类 (RecordedDate) 设定错误	N/A	在控制器模式 (Controller Mode) 下，使用本地 DI 触发型态时，探针数据记录总类 (RecordedDate) 必须大于 3。
0x5011	探针数据纪录设定错误	N/A	使用非本地 DI 触发型态时，探针数据记录总类 (RecordedDate) 必须大于等于 10。
0x5012	探针 window 设定错误	N/A	在开启 window 功能的情况下，起始值 (FirstValue) 与末端值 (LastValue) 设定错误。
0x5013	PDO 配置错误	YXU：轴编号	使用伺服 capture 功能，但未将 0x60BA 与 0x60BB 放入 PDO 配置。
0x5014	window 数据型态不支持	YXU：轴编号	window 数据型态 (WindowType) 超出设定范围。
0x5015	本地触发 DI 编号设定错误	YXU：轴编号	使用本地 DI 触发型态 (Local DI) 时，触发 DI 编号 (DINumber) 设定错误。
0x5016	无效的扩充 DI 编号	YXU：轴编号	使用扩充 DI 触发型态 (Extension DI) 时，触发 DI 编号 (DINumber) 设定错误。
0x5017	无效的远程 DI 编号	YXU：轴编号	使用远程 DI 触发型态 (Remote DI) 时，触发 DI 编号 (DINumber) 设定错误。
0x5018	PLC 变量触发 DI 编号设定错误	YXU：轴编号	使用 PLC 变量触发型态 (PLC Variable) 时，触发 DI 编号 (DINumber) 设定错误。
0x5019	无效的本地 DI 编号	YXU：轴编号	使用本地 DI 触发型态 (Local DI) 时，触发 DI 编号 (DINumber) 设定错误。
0x6000	总线装置存在错误	YXU：总线装置 ID	使用功能块时，对象装置存在错误。

事件码	事件名称	扩充码	可能原因
0x6001	错误的总线类型	YXU：总线装置 ID	使用总线操作的功能块时，选择了错误的总线类型，详细的参数说明请参考章节 5.5 轴参数说明。
0x6002	PLC 功能块的输入变量长度不合法	YXU：总线装置 ID	PLC 功能块的输入变量长度不符合功能块定义。
0x6003	PLC 功能块的输入变量编号不合法	YXU：总线装置 ID	PLC 功能块的输入变量编号不符合功能块定义。
0x6100	EtherCAT 扫站失败	N/A	EtherCAT 执行扫站时发生错误。
0x6101	EtherCAT 从站信息取得失败	N/A	EtherCAT 取得从站信息时发生错误。
0x6102	切换 EtherCAT 至 OP 模式失败	N/A	EtherCAT 切换至 OP 模式时发生错误。
0x6103	EtherCAT 断线	N/A	EtherCAT 运作时，发生断线。
0x6104	切换 EtherCAT 至 Init.模式失败	N/A	EtherCAT 切换至 Init.模式时发生错误。
0x6105	设定 EtherCAT 从站别名失败	N/A	设定指定站号时发生错误。
0x6106	储存 EtherCAT 从站别名失败	N/A	储存设定的指定站号时发生错误。
0x6107	EtherCAT 网络程序错误	N/A	韧体版本不匹配。
0x6108	网络联机有问题	N/A	网络卡故障或是网络线没有接好。
0x6109	硬件没有正确初始化	N/A	硬件标识符错误。
0x610A	没有找到任何 EtherCAT 模块	N/A	扫描不到任何模块。
0x610B	找不到模块对应的 dat 档案	N/A	模块没有对应的 dat。
0x610C	OD数量太多	N/A	PDO 封包长度超过限制。
0x610D	超过支持的站数	N/A	超过支持的站数。
0x610E	OD 映射错误	N/A	PDO 配置错误。
0x610F	切换 op mode 失败	N/A	等待切换 op 超过时间。
0x6110	EtherCAT 模块设定失败	N/A	EtherCAT 模块设定失败。
0x6111	EtherCAT 状态不在 OP 模式	N/A	未将 EtherCAT 状态切换至 OP 模式。
0x6112	EtherCAT PRE-OP 模式切换失败	N/A	EtherCAT 无法切换至 PRE-OP 模式。
0x6C01	DMCNET 操作指定站号错误	N/A	DMCNET 指令指定的站号未联机或超过范围。

9

事件码	事件名称	扩充码	可能原因
0x6C02	DMCNET 网络联机未建立	N/A	DMCNET 网络未完成初始化或联机状态异常。
0x6C03	DMCNET SDO 存取指令数据类型错误	N/A	利用 SDO 读写数据的指令，其指定的数据类型错误。
0x6C04	DMCNET 传送 SDO 封包错误	N/A	系统无法处理 DMCNET SDO 封包发送请求。
0x6C05	DMCNET 接收 SDO 封包错误	N/A	系统无法处理 DMCNET SDO 封包接收请求。
0x6C06	DMCNET 接收 SDO 回应封包尚未完成	N/A	DMCNET SDO 通讯流程进行中，尚在等待响应封包。
0x6C08	DMCNET SDO 读写数据的指令，响应内容的数据长度超过所要求的	N/A	利用 SDO 读写数据的指令，其指定的数据长度小于存取对象的实际长度。
0x6C09	DMCNET 网络切换操作模式进行中	N/A	切换 DMCNET 操作模式已经启动，在进行中又重复下达切换操作模式。
0x6C0B	DMCNET 网络执行状态未知	N/A	DMCNET 网络当前状态未明，无法进行切换模式。
0x6C0C	检查 DMCNET 切换操作模式进行状态：进行中	N/A	DMCNET 切换操作模式仍在进行中。
0x6C0D	检查 DMCNET 切换操作模式进行状态：失败	N/A	DMCNET 网络无法切换操作模式。
0x6C0E	检查 DMCNET 切换操作模式进行状态：未启动	N/A	尚未启动切换 DMCNET 操作模式。
0x6C0F	DMCNET 网络尚未切换到可操作模式	N/A	DMCNET 没有切换到可操作模式。
0x6C11	系统无法取得响应的 DMCNET PDO 封包	N/A	系统无法取得更新的 DMCNET PDO 响应，可能是通讯干扰造成数据被芯片丢弃或断线造成收不到数据。
0x6C12	DMCNET 设定 DO 模块数据长度错误	N/A	DMCNET 设定 DO 模块数据量过大。
0x6C13	DMCNET 设定 DO 数据模块信息错误	N/A	下达设定 DO 数据指令于没有 DO 输出的 DMCNET 模块。
0x6C14	DMCNET 网络中无任何可识别装置	N/A	DMCNET 网络初始化完成，但没有任何装置切换到可操作模式。

事件码	事件名称	扩充码	可能原因
0x6C15	DMCNET 切换操作模式逾时	N/A	切换 DMCNET 操作模式超过 30 秒未完成。
0x6C16	DMCNET SDO 通讯响应封包格式不符	N/A	DMCNET SDO 回应封包的标头异常。
0x6C17	DMCNET SDO 通讯响应封包站号不符	N/A	DMCNET SDO 响应封包的站号信息与发送记录不符。
0x6C18	DMCNET SDO 通讯响应封包记录数据长度为零	N/A	DMCNET SDO 响应封包显示数据长度为零。
0x6C19	DMCNET SDO 读写指令内容不合法	N/A	SDO 读写指令欲存取的数据不被 DMCNET 联机装置接受。
0x6C1A	DMCNET monitor id 设定错误	N/A	monitor id 设定超出范围。
0x6C1B	DMCNET 通讯硬件初始化失败	N/A	无法正确初始化硬件。
0x6D01	MODBUS 命令错误	N/A	MODBUS master 通讯异常，回传值显示 slave 端不支持该指令。
0x6D02	MODBUS 存取地址错误	N/A	MODBUS master 通讯异常，回传值显示存取的地址不合法。
0x6D03	MODBUS 写入值错误	N/A	MODBUS master 通讯异常，回传值显示要写入 slave 的值不合法。
0x6D04	MODBUS 通讯地址不合法	N/A	输入的内存起始位置错误。
0x6D05	MODBUS 通讯数据长度不合法	N/A	通讯数据长度不正确。
0x6D06	MODBUS 通讯不支持存取模拟输入/输出	N/A	通讯地址存取到了模拟输入/输出的范围。
0x6D20	MODBUS 功能块传入不支持的指令代码	N/A	MODBUS master 通讯异常，功能块传入不支持的指令代码。
0x6D21	MODBUS master connect 脚位设定错误	N/A	MODBUS master connect 功能块传入错误的网络设定值 (IP/Port)；或者 disconnect、serial_set 或 read/write 的功能块传入错误的联机代号。
0x6D22	MODBUS 通讯未完成	N/A	还在等待 slave 回传响应封包的阶段，此次通讯仍在进行中。

9

事件码	事件名称	扩充码	可能原因
0x6D23	MODBUS 所有联机资源都在使用中	N/A	MODBUS master connect 功能块显示所有联机资源都在使用中，无法取得新的联机。
0x6D24	MODBUS 建立联机失败	N/A	MODBUS master connect 建立联机失败，可能是 slave 不支持指定的 port 或联机操作时间超过逾时设定值。
0x6D25	MODBUS 通讯逾时	N/A	MODBUS master 送出指令封包后，超过逾时设定值的时间仍未收到 slave 的响应。
0x6D26	MODBUS master 无法开启对应的 UART port	N/A	透过 serial 的 MODBUS master 无法开启对应的 UART port。
0x6D27	MODBUS 设定网络 socket 失败	N/A	MODBUS master connect 过程中，设定网络 socket 失败。
0x6D28	MODBUS 无法与系统的网络接口正确链接	N/A	MODBUS master connect 过程中，无法与系统的网络接口正确链接。
0x6D29	MODBUS 被 slave 端拒绝	N/A	MODBUS master connect 过程中，被 slave 端拒绝。
0x6D2A	MODBUS 读写功能块发送指令封包发生错误	N/A	MODBUS master 读写功能块发送指令封包发生错误。
0x6D2B	MODBUS 读写功能块读取响应封包发生错误	N/A	MODBUS master 读写功能块读取回应封包发生错误。
0x6D2C	MODBUS 功能块尝试关闭未使用的联机代号	N/A	MODBUS master disconnect 功能块尝试关闭未使用的联机代号。
0x6D2D	MODBUS 读写的功能块指定了尚未建立联机的联机代号	N/A	MODBUS master 读写的功能块指定了尚未建立联机的联机代号。
0x6D2E	MODBUS 读写功能块指定的联机代号尚未完成上次的通讯要求	N/A	MODBUS master 读写功能块指定的联机代号尚未完成上次的通讯要求。
0x6D2F	MODBUS master disconnect 功能块在设定的逾时时间内无法完成关闭联机的动作	N/A	MODBUS master disconnect 功能块在设定的逾时时间内无法完成关闭联机的动作。
0x6D30	MODBUS 通讯命令缓存已满	N/A	缓存内的命令来不及处理。

事件码	事件名称	扩充码	可能原因
0x7000	DRL 控制命令错误	N/A	透过 SDV 控制 DRL 时发生错误。

9.2 报警原因与处置

9

事件码	0x0001 运动命令空操作
扩充码	N/A
原因	运动命令执行过程中，控制器发生非法操作。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

事件码	0x0002 运动命令轴类型错误
扩充码	N/A
原因	运动命令的轴类型不正确。
检查及处置	确认运动命令使用的轴类型是否在合法范围。
排除方法	修正运动命令所使用的轴类型。

事件码	0x0003 运动命令群组类型错误
扩充码	N/A
原因	运动命令的群组类型不正确。
检查及处置	确认运动命令使用的群组类型是否在合法范围。
排除方法	修正运动命令所使用的群组类型。

事件码	0x0004 运动命令轴编号错误
扩充码	N/A
原因	运动命令的轴编号不正确。
检查及处置	确认运动命令使用的轴编号是否在合法范围。
排除方法	修正运动命令所使用的轴编号。

事件码	0x0005 运动命令群组编号错误
扩充码	N/A
原因	运动命令的群组编号不正确。
检查及处置	确认运动命令使用的群组编号是否在合法范围。
排除方法	修正运动命令所使用的群组编号。

事件码	0x0006 运动命令指令类型错误
扩充码	N/A
原因	运动命令类型不正确。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

事件码	0x0007 轴运动命令缓存已满
扩充码	N/A
原因	运动命令缓存已满。
检查及处置	确认是否在短时间内填入太多等待模式的命令，最大缓存深度为 3 笔命令。
排除方法	修改程序逻辑判断，等待命令缓存有空间之后，再触发运动命令。

事件码	0x0008 群组运动命令缓存已满
扩充码	N/A
原因	运动命令缓存已满。
检查及处置	确认是否在短时间内填入太多等待模式的命令，最大缓存深度为 3 笔命令。
排除方法	修改程序逻辑判断，等待命令缓存有空间之后，再触发运动命令。

事件码	0x000B 运动命令生命周期错误
扩充码	N/A
原因	运动命令生命周期不正确。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

事件码	0x000C 运动命令方向参数错误
扩充码	N/A
原因	运动命令内的方向参数不正确。
检查及处置	检查运动命令内是否填入了不正确的方向参数。
排除方法	修正方向参数输入。

9

事件码	0x000D 运动命令坐标系参数错误
扩充码	N/A
原因	运动命令内的坐标系参数不正确。
检查及处置	检查运动命令内是否填入了不正确的坐标系参数。
排除方法	修正坐标系参数输入。

事件码	0x000E 运动命令缓存模式错误
扩充码	N/A
原因	运动命令内的缓存模式不正确。
检查及处置	检查运动命令内是否填入了不正确的缓存模式。
排除方法	修正缓存模式输入。

事件码	0x000F 运动命令重迭模式错误
扩充码	N/A
原因	运动命令内的重迭模式不正确。
检查及处置	检查运动命令内是否填入了不正确的重迭模式。
排除方法	修正重迭模式输入。

事件码	0x0010 运动命令重迭缓存组合错误
扩充码	N/A
原因	运动命令内的缓存模式与重迭模式的组合不正确。
检查及处置	检查运动命令内是否填入了不正确的缓存模式与重迭模式组合。
排除方法	修正缓存模式与重迭模式输入。

事件码	0x0011 运动命令主轴编号错误
扩充码	N/A
原因	主轴编号不正确。
检查及处置	确认运动命令 (MC_GearIn、DMC_CamIn 等) 使用的主轴编号是否不正确或不存在。
排除方法	修正主轴编号。

事件码	0x0012 读写参数 index 错误
扩充码	N/A
原因	使用 MC_ReadParameter、MC_ReadBoolParameter、MC_WriteParameter、MC_WriteBoolParameter 时，运动参数编号 (ParameterNumber) 设定错误。
检查及处置	检查运动参数编号 (ParameterNumber) 设定。
排除方法	设定正确的运动参数编号 (ParameterNumber)。

事件码	0x0013 运动命令分母为零
扩充码	N/A
原因	运动命令参数内作为分母使用的参数为零。
检查及处置	确认运动命令内是否有参数为零。
排除方法	修正该项参数不为零。

事件码	0x0014 功能块编号错误
扩充码	N/A
原因	使用的功能块数量超过最大限制。
检查及处置	检查是否有不必要的功能块。
排除方法	删除不必要的功能块。

事件码	0x0015 速度脚位小于最小值
扩充码	YXU：轴编号
原因	速度脚位输入值必须大于零。
检查及处置	检查速度脚位输入值。
排除方法	将速度脚位填入正数。

事件码	0x0016 加速度脚位小于最小值
扩充码	YXU：轴编号
原因	加速度脚位输入值必须大于零。
检查及处置	检查加速度脚位输入值。
排除方法	将加速度脚位填入正数。

9

事件码	0x0017 减速度脚位小于最小值
扩充码	YXU：轴编号
原因	减速度脚位输入值必须大于零。
检查及处置	检查减速度脚位输入值。
排除方法	将减速度脚位填入正数。

事件码	0x0018 加加速度脚位小于最小值
扩充码	YXU：轴编号
原因	加加速度脚位输入值必须大于等于零。
检查及处置	检查加加速度脚位输入值。
排除方法	将加加速度脚位填入正数或零。

事件码	0x0019 功能块的 Any 脚位数据型态错误
扩充码	N/A
原因	Any 型态脚位 (Value) 所连接的变量长度小于欲读写的长度 (Length)。
检查及处置	检查 Any 型态脚位 (Value) 所连接的变量长度。
排除方法	使 Any 型态脚位 (Value) 所连接的变量长度与欲读写的长度 (Length) 一致。

事件码	0x001A 速度倍率脚位输入错误
扩充码	YXU：轴编号
原因	速度倍率 (VelFactor) 必须大于等于零。
检查及处置	检查速度倍率 (VelFactor)。
排除方法	将速度倍率 (VelFactor) 填入正数或零。

事件码	0x001B 加速倍率脚位输入错误
扩充码	YXU：轴编号
原因	加速度倍率 (AccFactor) 及加加速度倍率 (JerkFactor) 必须大于零。
检查及处置	检查加速度倍率 (AccFactor) 及加加速度倍率 (JerkFactor)。
排除方法	将加速度倍率 (AccFactor) 及加加速度倍率 (JerkFactor) 填入正数或零。

事件码	0x001C Operation Mode 不支持此运动命令
扩充码	YXU：轴编号
原因	轴目前处于的 Operation Mode 不支持此运动命令。
检查及处置	检查此运动命令是否能在此模式下进行。
排除方法	1. 更换适用的运动命令。 2. 将控制器切换到正确的模式。

事件码	0x001D 轴正在执行群组命令
扩充码	YXU：轴编号
原因	指定运动轴正在执行群组运动命令。
检查及处置	检查指定运动轴的状态是否处于 SynchronizedMotion。
排除方法	中断群组命令或等待其完成，并重新执行单轴运动命令。

事件码	0x001E 群组成员正在执行单轴命令
扩充码	YXU：群组编号
原因	运动群组的成员正在执行单轴运动命令。
检查及处置	是否有成员轴正在执行单轴运动命令。
排除方法	中断单轴命令或等待其完成，并重新执行群组运动命令。

事件码	0x001F 命令缓存处于忙碌状态
扩充码	N/A
原因	同时有多个 PLC task 或 Lua 项目对同一轴执行命令。
检查及处置	检查是否同时有多个 PLC task 或 Lua 项目对同一轴执行命令。
排除方法	修改命令执行时序。

事件码	0x0020 执行运动命令时控制器处于错误的模式
扩充码	N/A
原因	当前控制器模式不能执行命令。
检查及处置	检查控制器是否处于待命模式。
排除方法	将控制器切换至联机操作或联机诊断模式。

9

事件码	0x0040 凸轮应用功能码错误
扩充码	N/A
原因	使用了错误的凸轮应用功能码。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

事件码	0x0041 无可用的凸轮 ID
扩充码	N/A
原因	已经没有剩余的凸轮 ID 可用，最大可用 ID 数量为 256 个。
检查及处置	检查使用的凸轮 ID 个数是否已经超出最大数量。
排除方法	1. 移除已经不使用的凸轮 ID。 2. 覆写既有凸轮的资料。

事件码	0x0042 凸轮 ID 已存在
扩充码	N/A
原因	指定的凸轮 ID 已经存在。
检查及处置	在建立凸轮的操作中，检查该 ID 是否已经使用过。
排除方法	1. 更换指定的 ID。 2. 开启覆写功能，覆盖原先凸轮的内容。

事件码	0x0043 凸轮 ID 不存在
扩充码	N/A
原因	指定的凸轮 ID 不存在。
检查及处置	在使用凸轮的操作中，检查该 ID 是否已经建立过。
排除方法	更换 ID 为建立过的 ID。

事件码	0x0044 凸轮 CamIn scaling 参数错误
扩充码	N/A
原因	CamIn 使用的 scaling 参数错误。
检查及处置	检查 CamIn 参数 MasterScaling 或 SlaveScaling 是否为零。
排除方法	修正该参数。

事件码	0x0045 建立凸轮表时无法覆写
扩充码	N/A
原因	建立凸轮表时，没有开启覆写功能。
检查及处置	检查建立凸轮表时，是否有开启覆写功能。
排除方法	开启覆写功能。

事件码	0x0046 凸轮表类型参数错误
扩充码	N/A
原因	在建立凸轮的操作中，表类型参数错误。
检查及处置	确认类型是否正确。
排除方法	修正该参数。

事件码	0x0047 凸轮表主轴位置参数错误
扩充码	N/A
原因	在建立凸轮的操作中或是新增、修改凸轮表点位时，主轴位置错误。
检查及处置	1. 确认凸轮表主轴位置第一点是否为零。 2. 主轴位置是否为递增。
排除方法	修正该参数。

事件码	0x0048 凸轮表起始点位编号错误
扩充码	N/A
原因	新增、修改、删除凸轮表点位的操作中，起始点位编号错误。
检查及处置	1. 确认起始点位编号是否大于等于 1。 2. 如为新增凸轮表点位的操作中，确认起始点位编号是否超过原凸轮表的点位大小。 3. 如为修改、删除凸轮表点位的操作中，确认起始点位编号加上修改数量是否超过原凸轮表的点位大小。
排除方法	修正该参数。

9

事件码	0x0049 凸轮表超出范围
扩充码	N/A
原因	新增凸轮表点位的操作超出一个凸轮表预设最大点位。
检查及处置	确认起始点位编号加上修改数量是否超过单一凸轮表格最大点数上限。
排除方法	修正起始点位编号参数或修正数量参数。

事件码	0x004A 欲使用的凸轮表是空的
扩充码	N/A
原因	于 CamIn 新增、修改、删除凸轮表点位的操作中，欲使用的凸轮表是空的。
检查及处置	确认是否已建立欲使用的凸轮表。
排除方法	建立欲使用的凸轮表。

事件码	0x004B 凸轮表已满
扩充码	N/A
原因	于建立凸轮表的操作中，无剩余空间可以使用。
检查及处置	删除已经不使用的凸轮表，释放出空间。
排除方法	删除已经不使用的凸轮表，释放出空间。

事件码	0x004C 凸轮表在线切换功能，ActiveMode 参数错误
扩充码	N/A
原因	设置在线切换功能操作中，ActiveMode 参数不正确。
检查及处置	确认 ActiveMode 参数是否正确，详细的参数说明请参考章节 6.5.8.17 ECAM_SetOnlineChgMode。
排除方法	修正该参数。

事件码	0x004E 凸轮表使用中
扩充码	N/A
原因	删除凸轮表操作中，欲操作的表格正在使用中。
检查及处置	检查是否有其他轴正在使用该凸轮表。
排除方法	1. 停止使用该凸轮表的轴的运动 (MC_Stop 或 MC_CamOut)。 2. 可开启复写参数强制删除。

事件码	0x004F 凸轮表档案不存在
扩充码	N/A
原因	指定路径内并不存在用户输入的 ECAM 文件名。
检查及处置	检查输入的文件名是否存在。
排除方法	重新输入路径中存在的 ECAM 档案。

事件码	0x0050 凸轮表档案数据错误
扩充码	N/A
原因	ECAM 档案内容长度有问题，系统无法读取内容。
检查及处置	检查输入文件名是否为凸轮表格式。
排除方法	1. 确认文件格式。 2. 更换凸轮表。

事件码	0x0051 凸轮表的 ID 为空白
扩充码	N/A
原因	没有输入 TableID。
检查及处置	检查 TableID 是否空白。
排除方法	输入 TableID。

事件码	0x0052 凸轮表档名为空白
扩充码	N/A
原因	没有输入 FileName。
检查及处置	检查 FileName 是否空白。
排除方法	输入 FileName。

事件码	0x0053 追剪功能返回距离不足
扩充码	N/A
原因	建立追剪表格操作中，因同步区太长或切长太短导致追剪功能返回距离不足。
检查及处置	检查以下返回距离是否不足：返回距离 = 切长 - 主轴同步位置 - (从轴结束位置 - 从轴同步位置)。
排除方法	修正上述相关参数使返回距离大于 0。

9

事件码	0x0054 旋切调整区域小于零
扩充码	N/A
原因	建立旋切表格操作中，因切长太短或是同步区太长导致旋切调整区域小于零。
检查及处置	确认切长与同步区的范围是否合理。
排除方法	1. 缩短同步区长度 (SyncStartPos 或 SyncStopPos)。 2. 缩小同步区速度比例。

事件码	0x0055 凸轮表档案路径错误
扩充码	N/A
原因	用户输入的路径有误。
检查及处置	确认输入的路径为系统提供的路径。
排除方法	更换输入的档案路径。

事件码	0x0056 凸轮表档案无法覆写
扩充码	N/A
原因	档案无法覆写。
检查及处置	确认 overwrite 是否为 true。
排除方法	将 overwrite 脚位更改为 true。

事件码	0x0057 挺杆点主轴位置错误
扩充码	N/A
原因	挺杆点设置操作中，主轴位置参数不正确。
检查及处置	确认主轴位置是否为递增。
排除方法	修正该参数。

事件码	0x0058 挺杆点方向错误
扩充码	N/A
原因	挺杆点设置操作中，挺杆点方向参数不正确。
检查及处置	确认挺杆点方向参数是否正确。
排除方法	修正该参数。

事件码	0x005A 旋切同步区起始位置错误
扩充码	N/A
原因	建立旋切表格操作中，同步区起始位置参数不正确。
检查及处置	1. 确认旋切同步区起始位置是否小于同步区停止位置。 2. 确认旋切同步区起始位置是否小于零。 3. 确认旋切同步区起始位置是否超过切长。
排除方法	修正该参数。

事件码	0x005B 旋切同步区结束位置错误
扩充码	N/A
原因	建立旋切表格操作中，同步区结束位置参数不正确。
检查及处置	确认旋切同步区结束位置是否超过切长。
排除方法	修正该参数。

事件码	0x005C 旋切主轴或从轴为旋转轴
扩充码	N/A
原因	CamIn 操作中，旋切主轴或从轴为旋转轴。
检查及处置	确认主轴或从轴是否为非旋转轴。
排除方法	更换 CamIn 操作中的主轴或从轴，使主轴和从轴皆非旋转轴。

事件码	0x005D 凸轮比例尺参数为零
扩充码	N/A
原因	凸轮比例尺功能操作中，比例尺参数为零。
检查及处置	确认主轴比例尺参数或从轴比例尺参数是否为零。
排除方法	修正该参数。

事件码	0x005E 凸轮比例尺参数小于零
扩充码	N/A
原因	凸轮比例尺功能操作中，比例尺参数小于零。
检查及处置	确认主轴比例尺参数或从轴比例尺参数是否小于零。
排除方法	修正该参数。

9

事件码	0x005F 凸轮比例尺功能 MasterScalingMode 错误
扩充码	N/A
原因	凸轮比例尺功能操作中，MasterScalingMode 参数不正确。
检查及处置	确认 MasterScalingMode 参数是否正确。
排除方法	修正该参数。

事件码	0x0060 凸轮比例尺模式错误
扩充码	N/A
原因	凸轮比例尺功能操作中或在线切换功能操作中，比例尺模式参数不正确。
检查及处置	确认主（从）轴比例尺模式参数是否正确。
排除方法	修正该参数。

事件码	0x0061 旋切同步区速度比例为零
扩充码	N/A
原因	建立旋切凸轮表操作中或修改旋切表格速度比例操作中，同步区速度比例参数为零。
检查及处置	确认同步区速度比例参数是否为零。
排除方法	修正该参数。

事件码	0x0062 凸轮表不是旋切凸轮表
扩充码	N/A
原因	修改旋切表格速度比例操作中，欲修改的表格非旋切凸轮表。
检查及处置	检查功能块 TableID 脚位所对应的凸轮表格是否为旋切凸轮表，详细的参数说明请参考章节 6.5.8.16 ECAM_ModRotCutVelRatio。
排除方法	修正 TableID 脚位为旋切凸轮表。

事件码	0x0063 凸轮表类型错误
扩充码	N/A
原因	建立凸轮表操作中，类型参数不正确。
检查及处置	确认类型参数是否正确，详细的参数说明请参考章节 6.5.8.2 ECAM_GenTableByData。
排除方法	修正该参数。

事件码	0x0064 凸轮挺杆点数量超出范围
扩充码	N/A
原因	凸轮挺杆点设置操作中，挺杆点数量参数超出范围。
检查及处置	确认挺杆点数量参数是否超出范围。
排除方法	修正该参数。

事件码	0x0065 CamIn 使用的 ID 参数错误
扩充码	N/A
原因	凸轮 CamIn 操作中，CamIn 使用的 TableID 参数不正确。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

事件码	0x0066 主轴来源参数错误
扩充码	N/A
原因	凸轮 CamIn 操作中，CamIn 使用的主轴来源参数不正确。
检查及处置	确认主轴来源参数是否正确，详细的参数说明请参考章节 6.5.3.3 DMC_CamIn。
排除方法	修正该参数。

事件码	0x0067 CamIn ActiveMode 错误
扩充码	N/A
原因	凸轮 CamIn 操作中，ActiveMode 参数不正确。
检查及处置	确认 ActiveMode 参数是否正确，详细的参数说明请参考章节 6.5.3.3 DMC_CamIn。
排除方法	修正该参数。

事件码	0x0068 CamIn BufferMode 错误
扩充码	N/A
原因	凸轮 CamIn 操作中，BufferMode 参数不正确。
检查及处置	确认 BufferMode 参数是否正确，详细的参数说明请参考章节 6.5.3.3 DMC_CamIn。
排除方法	修正该参数。

9

事件码	0x0069 CamIn StartMode 错误
扩充码	N/A
原因	凸轮 CamIn 操作中，StartMode 参数不正确。
检查及处置	确认 StartMode 参数是否正确，详细的参数说明请参考章节 6.5.3.3 DMC_CamIn。
排除方法	修正该参数。

事件码	0x006A 凸轮表数据数量错误
扩充码	N/A
原因	建立凸轮表操作中，凸轮表数据数量参数不正确。
检查及处置	确认资料数量参数是否正确，详细的参数说明请参考章节 6.5.8.2 ECAM_GenTableByData。
排除方法	修正该参数。

事件码	0x006B 追剪主轴位置参数错误
扩充码	N/A
原因	建立追剪表格操作中，主轴位置参数不正确。
检查及处置	确认主轴起始位置是否小于主轴同步位置。
排除方法	修正该参数。

事件码	0x006C 追剪从轴等待位置参数错误
扩充码	N/A
原因	建立追剪表格操作中，从轴等待位置参数不正确。
检查及处置	确认从轴等待位置是否小于从轴同步位置。
排除方法	修正该参数。

事件码	0x006D 追剪从轴同步位置参数错误
扩充码	N/A
原因	建立追剪表格操作中，从轴同步位置参数不正确。
检查及处置	确认从轴同步位置是否小于从轴结束位置。
排除方法	修正该参数。

事件码	0x006E ECAM 点位曲线型态错误
扩充码	N/A
原因	ECAM 点位曲线型态不支持。
检查及处置	检查 ECAM 点位曲线型态设定是否错误。
排除方法	设定正确 ECAM 点位曲线。

事件码	0x006F 飞剪曲线刀数为零
扩充码	N/A
原因	飞剪曲线刀数为零。
检查及处置	检查飞剪造表功能块刀数脚位设定是否大于零。
排除方法	设定正确刀数，并重新执行飞剪造表功能块。

事件码	0x0070 ECAM 表格点位数量错误
扩充码	N/A
原因	ECAM 造表功能块造表长度设定大于输入的点位数据数组。
检查及处置	检查 ECAM 造表功能块造表长度及点位数据数组设定是否正确。
排除方法	设定正确 ECAM 造表功能块造表长度及点位数据数组，并重新执行造表功能块。

事件码	0x0071 挺杆点群组 ID 错误
扩充码	N/A
原因	挺杆点群组 ID 大于 8。
检查及处置	检查挺杆点群组 ID 设定。
排除方法	设定正确挺杆点群组 ID 并重新触发功能块。

事件码	0x0072 挺杆点主轴位置重复
扩充码	N/A
原因	挺杆点的主轴位置有两点相同。
检查及处置	检查挺杆点的主轴位置是否两点相同。
排除方法	确认所有挺杆点的主轴位置不同。

9

事件码	0x0073 凸轮表速度建表的速度参数错误
扩充码	N/A
原因	速度百分比加总不等于 100%。
检查及处置	检查速度造表使用的速度百分比加总是否等于 100%。
排除方法	调整造表参数。

事件码	0x0074 建立伺服追飞剪凸轮表错误
扩充码	N/A
原因	追飞剪造表参数不符合伺服规则。
检查及处置	确认输入参数是否符合伺服规则。
排除方法	修改输入参数。

事件码	0x0080 未啮合齿轮便脱离齿轮
扩充码	N/A
原因	目标轴尚未 InGear 时使用 MC_GearOut。
检查及处置	确认目标轴是否 InGear。
排除方法	将目标轴使用 MC_GearIn。

事件码	0x0081 啮合齿轮没有主轴
扩充码	N/A
原因	使用 MC_GearIn 时并没有设定主轴。
检查及处置	确认是否设置主轴。
排除方法	设置 MC_GearIn 功能块的 Master 脚位。

事件码	0x0082 GearInPos 目标速度为零
扩充码	YXU：轴编号
原因	MC_GearInPos 功能块脚位设定造成从轴目标速度为零。
检查及处置	检查 MC_GearInPos 功能块速度脚位。
排除方法	重新设定 MC_GearInPos 功能块速度脚位，并再次执行功能块。

事件码	0x0083 GearInPos 最终速度为零
扩充码	YXU：轴编号
原因	MC_GearInPos 功能块脚位设定造成从轴最终速度为零。
检查及处置	检查 MC_GearInPos 功能块 RatioNumerator 脚位。
排除方法	重新设定 MC_GearInPos 功能块 RatioNumerator 脚位，并再次执行功能块。

事件码	0x0084 GearInPos 设定错误
扩充码	YXU：轴编号
原因	GearInPos 参数设定错误，造成从轴无法完成动作。
检查及处置	检查 MC_GearInPos 功能块是否正确。
排除方法	重新设定 MC_GearInPos 功能块，并再次执行功能块。

事件码	0x0100 轴在错误停止（ErrorStop）状态
扩充码	YXU：轴编号
原因	当轴发生错误后，轴会进入 ErrorStop 的状态。
检查及处置	检查轴状态是否为 ErrorStop。
排除方法	造成轴错误的状况排除后，使用重置命令。

事件码	0x0101 轴没有使能
扩充码	YXU：轴编号
原因	该轴不在使能状态。
检查及处置	检查该轴是否已经使能。
排除方法	使能该轴。

事件码	0x0102 轴正在停止
扩充码	YXU：轴编号
原因	轴正在进行停止命令。
检查及处置	1. 检查该轴是否已经完成停止命令。 2. 确认该群组的停止命令的 Execute 脚位是否为 False。
排除方法	1. 等待该轴完成停止命令。 2. 将停止命令的 Execute 脚位改为 False。

9

事件码	0x0103 轴正在执行回原
扩充码	YXU：轴编号
原因	轴正在执行回原命令。
检查及处置	检查该轴是否正在进行回原。
排除方法	等待该轴执行完回原命令。

事件码	0x0104 轴处于同步状态
扩充码	YXU：轴编号
原因	轴正处于同步状态。
检查及处置	检查该轴是否处于同步状态。
排除方法	先下达停止或脱离命令，使该轴离开同步状态。

事件码	0x0106 轴不在静止状态
扩充码	YXU：轴编号
原因	轴不处于静止状态。
检查及处置	检查该轴是否处于静止状态。
排除方法	1. 等待该轴执行完命令。 2. 下达停止命令来中断当前命令，并等待停止命令结束。

事件码	0x0107 轴回原失败
扩充码	YXU：轴编号
原因	轴执行回原命令失败。
检查及处置	1. 检查总线是否正常。 2. 检查伺服设定是否正常。
排除方法	1. 排除总线异常。 2. 设定正确伺服参数。

事件码	0x0108 多个 MC_Power 操作同一轴
扩充码	YXU：轴编号
原因	PLC 程序内，使用了多个 MC_Power 控制同一个轴。
检查及处置	检查 PLC 程序内是否使用了多个 MC_Power 控制同一个轴。
排除方法	一个轴只能由一个 MC_Power 进行控制，移除多余的 MC_Power。

事件码	0x0109 轴不在同步状态
扩充码	YXU：轴编号
原因	轴不在同步状态。
检查及处置	确认运动命令的对象轴是否有处于同步状态。
排除方法	先让对象轴进入同步状态之后再使用运动命令。

事件码	0x010A 轴不在 EtherCAT 模式
扩充码	YXU：轴编号
原因	伺服轴控制模式不为 EtherCAT。
检查及处置	检查伺服轴控制模式参数。
排除方法	设定正确伺服参数并重新将控制器切换至"联机操作"模式。

事件码	0x010B 回原命令超时
扩充码	YXU：轴编号
原因	回原命令下达后超过 5 秒未被执行。
检查及处置	检查 MC_Power 功能块是否正确完成。
排除方法	MC_Power 功能块正确完成后，清除报警并重新触发回原功能块。

事件码	0x0801 轴没有回原
扩充码	YXU：轴编号
原因	轴的回原点定义失效。
检查及处置	确认轴是否有成功执行过回原命令。
排除方法	重新进行回原命令。

事件码	0x0815 超出轴位置软正极限
扩充码	YXU：轴编号
原因	轴位置超出软件正极限。
检查及处置	确认轴当前位置是否有超出软件极限。
排除方法	将轴位置修正到极限内。

9

事件码	0x0816 超出轴位置软负极限
扩充码	YXU：轴编号
原因	轴位置超出软件负极限。
检查及处置	确认轴当前位置是否有超出软件极限。
排除方法	将轴位置修正到极限内。

事件码	0x0817 超出轴速度限制
扩充码	YXU：轴编号
原因	轴速度超出软件限制。
检查及处置	确认轴速度命令是否超出限制。
排除方法	1. 降低速度命令。 2. 提高速度软件限制。

事件码	0x0818 超出轴加速度限制
扩充码	YXU：轴编号
原因	轴加速度超出软件限制。
检查及处置	确认轴加速度命令是否超出限制。
排除方法	1. 降低加速度命令。 2. 提高加速度软件限制。

事件码	0x0819 超出轴减速度限制
扩充码	YXU：轴编号
原因	轴减速度超出软件限制。
检查及处置	确认轴减速度命令是否超出限制。
排除方法	1. 降低减速度命令。 2. 提高减速度软件限制。

事件码	0x081A 超出轴加加速度限制
扩充码	YXU：轴编号
原因	轴加加速度超出软件限制。
检查及处置	确认轴加加速度命令是否超出限制。
排除方法	1. 降低加加速度命令。 2. 提高加加速度软件限制。

事件码	0x081B 超出轴位置误差限制
扩充码	YXU：轴编号
原因	轴位置误差超出软件限制。
检查及处置	确认轴位置误差是否超出限制。
排除方法	1. 降低速度或加减速来减少位置误差。 2. 调整伺服增益。 3. 提高位置误差软件限制。

事件码	0x0917 超出轴速度警告
扩充码	YXU：轴编号
原因	轴速度超出软件警告值。
检查及处置	确认轴速度命令是否超出警告值。
排除方法	1. 降低速度命令。 2. 提高速度软件警告值。

事件码	0x0918 超出轴加速度警告
扩充码	YXU：轴编号
原因	轴加速度超出软件警告值。
检查及处置	确认轴加速度命令是否超出警告值。
排除方法	1. 降低加速度命令。 2. 提高加速度软件警告值。

9

事件码	0x0919 超出轴减速度警告
扩充码	YXU：轴编号
原因	轴减速度超出软件警告值。
检查及处置	确认轴减速度命令是否超出警告值。
排除方法	1. 降低减速度命令。 2. 提高减速度软件警告值。

事件码	0x091A 超出轴加加速度警告
扩充码	YXU：轴编号
原因	轴加加速度超出软件警告值。
检查及处置	确认轴加加速度命令是否超出警告值。
排除方法	1. 降低加加速度命令。 2. 提高加加速度软件警告值。

事件码	0x091B 超出轴位置误差警告数值
扩充码	YXU：轴编号
原因	轴位置误差超出软件警告值。
检查及处置	确认轴位置误差是否超出警告值。
排除方法	1. 降低速度或加减速来减少位置误差。 2. 调整伺服增益。 3. 提高位置误差软件警告值。

事件码	0x091C 超出轴扭矩警告正极限
扩充码	YXU：轴编号
原因	轴扭矩超出软件警告值。
检查及处置	确认轴扭矩是否超出警告正极限。
排除方法	1. 降低加减速命令。 2. 提高扭矩警告正极限。

事件码	0x091D 超出轴扭矩警告负极限
扩充码	YXU：轴编号
原因	轴扭矩超出软件警告值。
检查及处置	确认轴扭矩是否超出警告负极限。
排除方法	1. 降低加减速命令。 2. 提高扭矩警告负极限。

事件码	0x091E 利用 SDO 切换轴运行模式失败
扩充码	YXU：轴编号
原因	利用 SDO 切换轴运行模式失败。
检查及处置	1. 确认轴 EtherCAT 状态是否切换到 OP 模式。 2. 确认轴是否能利用 SDO 切换运行模式。
排除方法	1. 重新下载配置文件并将 DXMC 切换到联机操作模式。 2. 确认轴是否能利用 SDO 切换运行模式。

事件码	0x091F 上一笔切换轴运行模式的 SDO 命令尚在执行
扩充码	YXU：轴编号
原因	上一笔切换轴运行模式的 SDO 命令尚在执行。
检查及处置	等待上一笔切换轴运行模式的 SDO 命令执行完毕。
排除方法	等待上一笔切换轴运行模式的 SDO 命令执行完毕。

事件码	0x0920 回原参数超界
扩充码	YXU：轴编号
原因	回原参数设定超出范围。
检查及处置	检查回原参数是否超出范围。
排除方法	设定正确的回原参数，参数范围请详阅伺服应用技术手册。

9

事件码	0x0921 指定回原模式不支持
扩充码	YXU：轴编号
原因	回原模式 (HomeMode) 不支持。
检查及处置	确认回原模式 (HomeMode) 是否支持，回原模式 (HomeMode) 设定范围请详阅伺服应用技术手册。
排除方法	设定正确的回原模式 (HomeMode)，回原模式 (HomeMode) 设定范围请详阅伺服应用技术手册。

事件码	0x0922 回原第一段速超界
扩充码	YXU：轴编号
原因	回原第一段速 (VelocityFast) 超出极限。
检查及处置	确认回原第一段速 (VelocityFast) 是否超出极限，回原第一段速 (VelocityFast) 设定范围请详阅伺服应用技术手册。
排除方法	设定正确的回原第一段速 (VelocityFast)，回原第一段速 (VelocityFast) 设定范围请详阅伺服应用技术手册。

事件码	0x0923 回原第二段速超界
扩充码	YXU：轴编号
原因	回原第二段速 (VelocitySlow) 超出极限。
检查及处置	确认回原第二段速 (VelocitySlow) 是否超出极限，回原第二段速 (VelocitySlow) 设定范围请详阅伺服应用技术手册。
排除方法	设定正确的回原第二段速 (VelocitySlow)，回原第二段速 (VelocitySlow) 设定范围请详阅伺服应用技术手册。

事件码	0x0924 回原加速度超界
扩充码	YXU：轴编号
原因	回原加速度 (Acceleration) 超出极限。
检查及处置	确认回原加速度 (Acceleration) 是否超出极限，回原加速度 (Acceleration) 设定范围请详阅伺服应用技术手册。
排除方法	设定正确的回原加速度 (Acceleration)，回原加速度 (Acceleration) 设定范围请详阅伺服应用技术手册。

事件码	0x0925 单轴命令 BlendingMode 加（减）速距离不足
扩充码	YXU：轴编号
原因	单轴命令以 BlendingMode 衔接时，加（减）速距离不足。
检查及处置	检查单轴命令衔接时的加（减）速度或加加速度是否过小，造成加（减）速距离不足。
排除方法	调整单轴命令的加（减）速度和加加速度。

事件码	0x0926 多轴命令 BlendingMode 加（减）速距离不足
扩充码	YXU：轴编号
原因	多轴命令以 BlendingMode 衔接时，加（减）速距离不足。
检查及处置	检查多轴命令衔接时的加（减）速度或加加速度是否过小，造成加（减）速距离不足。
排除方法	调整多轴命令的加（减）速度和加加速度。

事件码	0x0927 位置命令溢位
扩充码	YXU：轴编号
原因	位置命令已超过双精度浮点数之最大值，因此无法取得轴实际位置。
检查及处置	检查位置命令是否超过双精度浮点数之最大值。
排除方法	重新寻找轴原点。

事件码	0x1000 群组没有启用
扩充码	Y：保留、XU：群组编号
原因	对象群组没有启用。
检查及处置	检查该群组是否已经启用。
排除方法	启用该群组。

事件码	0x1001 群组处于错误停止（ErrorStop）状态
扩充码	Y：保留、XU：群组编号
原因	群组处于 ErrorStop 状态。
检查及处置	确认群组是否处于 ErrorStop 状态。
排除方法	排除造成错误的状况后，使用重置命令。

9

事件码	0x1002 群组停止中
扩充码	Y：保留、XU：群组编号
原因	群组正在进行停止命令中。
检查及处置	1. 确认该群组是否还没完成停止命令。 2. 确认该群组的停止命令的 Execute 脚位是否为 False。
排除方法	1. 等待该群组完成停止命令。 2. 将停止命令的 Execute 脚位改为 False。

事件码	0x1003 群组回原中
扩充码	Y：保留、XU：群组编号
原因	该群组正在进行回原命令中。
检查及处置	确认群组是否正在执行回原。
排除方法	1. 等待该群组完成回原。 2. 下达停止命令中断回原。

事件码	0x1004 该群组成员不存在
扩充码	Y：保留、XU：群组编号
原因	群组选择成员轴时，发现该成员轴不存在。
检查及处置	确认群组的成员轴设定是否正确并存在。
排除方法	修正群组的成员轴设定。

事件码	0x1005 群组没有回原
扩充码	Y：保留、XU：群组编号
原因	群组还没有成功执行过回原命令。
检查及处置	确认群组的成员轴关于回原的设置是否正确。
排除方法	将群组的成员轴回原设定设置正确。

事件码	0x1006 群组命令 BlendingMode 加（减）速距离不足
扩充码	YXU：群组编号
原因	群组命令以 BlendingMode 衔接时，加（减）速距离不足。
检查及处置	检查群组命令衔接时的加（减）速度或加加速度是否过小，造成加(减）速距离不足。
排除方法	调整群组命令的加（减）速度和加加速度。

事件码	0x1007 群组 PLC 状态并未处于 Standby
扩充码	YXU：群组编号
原因	执行群组回原时，群组成员并未全部使能。
检查及处置	检查各群组成员是否都已使能。
排除方法	确认成员都已使能并再次触发群组回原功能。

事件码	0x1010 群组速度超界
扩充码	YXU：群组编号
原因	群组命令或回授速度超过最大限制。
检查及处置	检查群组命令或回授速度是否超过配置文件最大合成速度设定。
排除方法	1. 更改最大合成速度设定。 2. 更改速度命令。

事件码	0x1011 群组加速度超界
扩充码	YXU：群组编号
原因	群组命令或回授加速度超过最大限制。
检查及处置	检查群组命令或回授加速度是否超过配置文件最大合成加速度设定。
排除方法	1. 更改最大合成加速度设定。 2. 更改加速度命令。

事件码	0x1012 群组减速度超界
扩充码	YXU：群组编号
原因	群组命令或回授减速度超过最大限制。
检查及处置	检查群组命令或回授减速度是否超过配置文件最大合成减速度设定。
排除方法	1. 更改最大合成减速度设定。 2. 更改减速度命令。

事件码	0x1013 群组加加速度超界
扩充码	YXU：群组编号
原因	群组命令或回授加加速度超过最大限制。
检查及处置	检查群组命令或回授加加速度是否超过配置文件最大合成加加速度设定。
排除方法	1. 更改最大合成加加速度设定。 2. 更改加加速度命令。

9

事件码	0x1020 群组速度超过警告值
扩充码	YXU：群组编号
原因	群组命令或回授速度超过警告。
检查及处置	检查群组命令或回授速度是否超过配置文件合成速度警告值设定。
排除方法	1. 更改合成速度警告值设定。 2. 更改速度命令。

事件码	0x1021 群组加速度超过警告值
扩充码	YXU：群组编号
原因	群组命令或回授加速度超过警告。
检查及处置	检查群组命令或回授加速度是否超过配置文件合成加速度警告值设定。
排除方法	1. 更改合成加速度警告值设定。 2. 更改加速度命令。

事件码	0x1022 群组减速度超过警告值
扩充码	YXU：群组编号
原因	群组命令或回授减速度超过警告。
检查及处置	检查群组命令或回授减速度是否超过配置文件合成减速度警告值设定。
排除方法	1. 更改合成减速度警告值设定。 2. 更改减速度命令。

事件码	0x1023 群组加加速度超过警告值
扩充码	YXU：群组编号
原因	群组命令或回授加加速度超过警告。
检查及处置	检查群组命令或回授加加速度是否超过配置文件合成加加速度警告值设定。
排除方法	1. 更改合成加加速度警告值设定。 2. 更改加加速度命令。

事件码	0x1024 群组点对点运动速度超界
扩充码	YXU：群组编号
原因	群组执行点对点运动时超出成员轴最大速度。
检查及处置	检查点对点运动目标线速度是否超出成员轴最大速度。
排除方法	1. 重新设点对点运动目标线速度，并重新执行功能块。 2. 调整成员轴最大速度，并重新执行功能块。

事件码	0x1025 群组点对点运动加速度超界
扩充码	YXU：群组编号
原因	群组执行点对点运动时超出成员轴最大加速度。
检查及处置	检查点对点运动目标线加速度是否超出成员轴最大加速度。
排除方法	1. 重新设点对点运动目标线加速度，并重新执行功能块。 2. 调整成员轴最大加速度，并重新执行功能块。

事件码	0x1026 群组点对点运动减速度超界
扩充码	YXU：群组编号
原因	群组执行点对点运动时超出成员轴最大减速度。
检查及处置	检查点对点运动目标线减速度是否超出成员轴最大减速度。
排除方法	1. 重新设点对点运动目标线减速度，并重新执行功能块。 2. 调整成员轴最大减速度，并重新执行功能块。

事件码	0x1027 群组点对点运动加加速度超界
扩充码	YXU：群组编号
原因	群组执行点对点运动时超出成员轴最大加加速度。
检查及处置	检查点对点运动目标线速度是否超出成员轴最大加加速度。
排除方法	1. 重新设点对点运动目标线加加速度，并重新执行功能块。 2. 调整成员轴最大加加速度，并重新执行功能块。

9

事件码	0x1028 群组点对点运动速度超出警告值
扩充码	YXU：群组编号
原因	群组执行点对点运动时超出成员轴速度警告值。
检查及处置	检查点对点运动目标线速度是否超出成员轴速度警告值。
排除方法	1. 重新设点对点运动目标线速度，并重新执行功能块。 2. 调整成员轴速度警告值，并重新执行功能块。

事件码	0x1029 群组点对点运动加速度超出警告值
扩充码	YXU：群组编号
原因	群组执行点对点运动时超出成员轴加速度警告值。
检查及处置	检查点对点运动目标线加速度是否超出成员轴加速度警告值。
排除方法	1. 重新设点对点运动目标线加速度，并重新执行功能块。 2. 调整成员轴加速度警告值，并重新执行功能块。

事件码	0x102A 群组点对点运动减速度超出警告值
扩充码	YXU：群组编号
原因	群组执行点对点运动时超出成员轴减速度警告值。
检查及处置	检查点对点运动目标线减速度是否超出成员轴减速度警告值。
排除方法	1. 重新设点对点运动目标线减速度，并重新执行功能块。 2. 调整成员轴减速度警告值，并重新执行功能块。

事件码	0x102B 群组点对点运动加加速度超出警告值
扩充码	YXU：群组编号
原因	群组执行点对点运动时超出成员轴加加速度警告值。
检查及处置	检查点对点运动目标线加加速度是否超出成员轴加加速度警告值。
排除方法	1. 重新设点对点运动目标线加加速度，并重新执行功能块。 2. 调整成员轴加加速度警告值，并重新执行功能块。

事件码	0x1120 Robot 三角运算超出范围
扩充码	Y：保留、XU：群组编号
原因	计算逆向运动学时，发生三角运算超出范围。
检查及处置	确认命令点位在工作范围内。
排除方法	将命令点位教导在工作范围内。

事件码	0x1121 Robot 三角运算未定义
扩充码	Y：保留、XU：群组编号
原因	计算逆向运动学时，发生三角运算结果为值域未定义。
检查及处置	确认命令点位在工作范围内。
排除方法	将命令点位教导在工作范围内。

事件码	0x1122 Robot 手系错误
扩充码	Y：保留、XU：群组编号
原因	Robot 手系错误。
检查及处置	确认当前命令是否符合手系规则。
排除方法	修改当前命令符合手系规则。

事件码	0x1123 Robot 肩部奇异点
扩充码	Y：保留、XU：群组编号
原因	插值过程中产生奇异点。
检查及处置	检查路径上是否含有奇异点。
排除方法	修改教导路径，请勿经过奇异点。

事件码	0x1124 Robot 肘部奇异点
扩充码	Y：保留、XU：群组编号
原因	插值过程中产生奇异点。
检查及处置	检查路径上是否含有奇异点。
排除方法	修改教导路径，请勿经过奇异点。

9

事件码	0x1125 Robot 腕部奇异点
扩充码	Y：保留、XU：群组编号
原因	插值过程中产生奇异点。
检查及处置	检查路径上是否含有奇异点。
排除方法	修改教导路径，请勿经过奇异点。

事件码	0x1126 Robot 成员轴 1 超出极限
扩充码	Y：保留、XU：群组编号
原因	成员轴超出位置极限。
检查及处置	确认位置命令是否会造成该成员轴超出位置极限。
排除方法	修改位置命令。

事件码	0x1127 Robot 成员轴 2 超出极限
扩充码	Y：保留、XU：群组编号
原因	成员轴超出位置极限。
检查及处置	确认位置命令是否会造成该成员轴超出位置极限。
排除方法	修改位置命令。

事件码	0x1128 Robot 成员轴 3 超出极限
扩充码	Y：保留、XU：群组编号
原因	成员轴超出位置极限。
检查及处置	确认位置命令是否会造成该成员轴超出位置极限。
排除方法	修改位置命令。

事件码	0x1129 Robot 成员轴 4 超出极限
扩充码	Y：保留、XU：群组编号
原因	成员轴超出位置极限。
检查及处置	确认位置命令是否会造成该成员轴超出位置极限。
排除方法	修改位置命令。

事件码	0x112A Robot 成员轴 5 超出极限
扩充码	Y：保留、XU：群组编号
原因	成员轴超出位置极限。
检查及处置	确认位置命令是否会造成该成员轴超出位置极限。
排除方法	修改位置命令。

事件码	0x112B Robot 成员轴 6 超出极限
扩充码	Y：保留、XU：群组编号
原因	成员轴超出位置极限。
检查及处置	确认位置命令是否会造成该成员轴超出位置极限。
排除方法	修改位置命令。

事件码	0x112C Robot 成员轴 7 超出极限
扩充码	Y：保留、XU：群组编号
原因	成员轴超出位置极限。
检查及处置	确认位置命令是否会造成该成员轴超出位置极限。
排除方法	修改位置命令。

事件码	0x112D Robot 成员轴 8 超出极限
扩充码	Y：保留、XU：群组编号
原因	成员轴超出位置极限。
检查及处置	确认位置命令是否会造成该成员轴超出位置极限。
排除方法	修改位置命令。

事件码	0x112E Robot 成员轴 9 超出极限
扩充码	Y：保留、XU：群组编号
原因	成员轴超出位置极限。
检查及处置	确认位置命令是否会造成该成员轴超出位置极限。
排除方法	修改位置命令。

9

事件码	0x112F Robot 成员轴 10 超出极限
扩充码	Y：保留、XU：群组编号
原因	成员轴超出位置极限。
检查及处置	确认位置命令是否会造成该成员轴超出位置极限。
排除方法	修改位置命令。

事件码	0x1130 Robot 成员轴 11 超出极限
扩充码	Y：保留、XU：群组编号
原因	成员轴超出位置极限。
检查及处置	确认位置命令是否会造成该成员轴超出位置极限。
排除方法	修改位置命令。

事件码	0x1131 Robot 成员轴 12 超出极限
扩充码	Y：保留、XU：群组编号
原因	成员轴超出位置极限。
检查及处置	确认位置命令是否会造成该成员轴超出位置极限。
排除方法	修改位置命令。

事件码	0x1140 Robot UF 教导法错误
扩充码	Y：保留、XU：群组编号
原因	使用了不支持的 UF 教导法。
检查及处置	确认使用的 UF 教导法是否支持。
排除方法	使用支持的 UF 教导法。

事件码	0x1141 Robot UF 教导共线错误
扩充码	Y：保留、XU：群组编号
原因	UF 教导点位存在共线状况。
检查及处置	确认 UF 教导点位是否共线。
排除方法	修正 UF 教导点位存在共线状况。

事件码	0x1142 Robot UF 出现零向量
扩充码	Y：保留、XU：群组编号
原因	UF 其中某一轴向量为零向量。
检查及处置	检查是否 UF 教导点位有重复的点位。
排除方法	修正重复的点位。

事件码	0x1143 Robot UF 比例尺为 0
扩充码	Y：保留、XU：群组编号
原因	UF 的比例尺被填为 0。
检查及处置	检查 UF 教导过程中是否有将比例尺填为 0。
排除方法	请勿将比例尺填为 0。

事件码	0x1144 Robot UF 编号错误
扩充码	Y：保留、XU：群组编号
原因	使用了不合法的 UF 编号。
检查及处置	确认使用的编号是否在范围内。
排除方法	修正使用的编号。

事件码	0x1145 Robot UF 未教导
扩充码	Y：保留、XU：群组编号
原因	使用了尚未教导的 UF。
检查及处置	确认欲使用的 UF 是否已经教导。
排除方法	先教导该 UF 再使用该 UF。

事件码	0x1146 Robot UF 旋转角度错误
扩充码	Y：保留、XU：群组编号
原因	UF 教导过程中，产生不合法的角度设定。
检查及处置	确认 UF 教导点位是否合法。
排除方法	修正 UF 教导点位。

9

事件码	0x1160 Robot TF 教导法错误
扩充码	Y：保留、XU：群组编号
原因	使用了不支持的 TF 教导法。
检查及处置	确认使用的 TF 教导法是否支持。
排除方法	使用支持的 TF 教导法。

事件码	0x1161 Robot TF 教导共平面错误
扩充码	Y：保留、XU：群组编号
原因	TF 教导点位存在共平面状况。
检查及处置	检查教导点位是否为同一平面上的点。
排除方法	修正教导点位。

事件码	0x1162 Robot TF 编号错误
扩充码	Y：保留、XU：群组编号
原因	使用了不合法的 TF 编号。
检查及处置	确认使用的编号是否在范围内。
排除方法	修正使用的编号。

事件码	0x1163 Robot TF 未教导
扩充码	Y：保留、XU：群组编号
原因	使用了尚未教导的 TF。
检查及处置	确认欲使用的 TF 是否已经教导。
排除方法	先教导该 TF 再使用该 TF。

事件码	0x1164 Robot 命令数据长度不正确
扩充码	Y：保留、XU：群组编号
原因	命令的数据长度太短。
检查及处置	确认命令的数据长度是否符合 Robot 命令的数据长度。
排除方法	修改命令的数据长度。

事件码	0x1165 Robot 重迭参数错误
扩充码	Y：保留、XU：群组编号
原因	使用了不合法的 Robot 重迭参数。
检查及处置	确认使用的重迭参数是否符合规则。
排除方法	修正重迭参数。

事件码	0x1166 Robot 超出工作范围
扩充码	YXU：群组编号
原因	Robot 超出工作范围。
检查及处置	检查 Robot 目标位置或命令轨迹是否超出工作范围。
排除方法	1. 重新设定 Robot 目标位置或命令轨迹。 2. 重新设定工作范围。

事件码	0x1167 Robot 进入工作禁区
扩充码	YXU：群组编号
原因	Robot 进入工作禁区。
检查及处置	检查 Robot 目标位置或命令轨迹是否经过工作禁区。
排除方法	1. 重新设定 Robot 目标位置或命令轨迹。 2. 重新设定工作禁区。

事件码	0x1168 工作范围设定失败
扩充码	YXU：群组编号
原因	工作范围设定有误造成设定失败。
检查及处置	检查工作范围设定是否合理。
排除方法	调整工作范围设定参数，并重新执行设定工作范围功能。

事件码	0x1169 工作禁区设定失败
扩充码	YXU：群组编号
原因	工作禁区设定有误造成设定失败。
检查及处置	检查工作禁区设定是否合理。
排除方法	调整工作禁区设定参数，并重新执行设定工作禁区功能。

9

事件码	0x3000 参数空操作
扩充码	N/A
原因	参数存取命令执行过程中，控制器发生非法操作。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

事件码	0x3001 参数操作对象错误
扩充码	N/A
原因	参数存取命令对象类型错误。
检查及处置	确认参数存取对象的类型是否有误，详细的参数说明请参考第 5 章。
排除方法	修正为正确的对象类型。

事件码	0x3004 参数操作对象编号错误
扩充码	N/A
原因	参数存取命令对象编号错误。
检查及处置	确认参数存取对象的编号是否有误，详细的参数说明请参考第 5 章。
排除方法	修正为正确的对象编号。

事件码	0x3005 参数操作轴编号错误
扩充码	N/A
原因	参数存取命令轴编号错误。
检查及处置	确认参数存取轴编号是否有误，详细的参数说明请参考第 5 章。
排除方法	修正为正确轴编号。

事件码	0x3006 参数操作群组编号错误
扩充码	N/A
原因	参数存取命令群组编号错误。
检查及处置	确认参数存取群组编号是否有误，详细的参数说明请参考第 5 章。
排除方法	修正为正确群组编号。

事件码	0x3007 参数操作功能码错误
扩充码	N/A
原因	参数操作功能码错误。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

事件码	0x3008 参数命令接收缓存已满
扩充码	N/A
原因	短时间内下达太多参数操作命令。
检查及处置	确认是否一次送出太多参数操作命令需求。
排除方法	等待前一笔参数命令完成之后再进行下一笔参数命令。

事件码	0x3009 参数命令接收缓存为空
扩充码	N/A
原因	参数命令接收缓存为空。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

事件码	0x300B 参数命令传送缓存为空
扩充码	N/A
原因	参数命令传送缓存为空。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

事件码	0x300C 参数数值超出最大值
扩充码	N/A
原因	参数输入数值超出最大值。
检查及处置	检查参数输入数值是否超出最大值。
排除方法	使用范围内的数值。

9

事件码	0x300D 参数数值超出最小值
扩充码	N/A
原因	参数输入数值超出最小值。
检查及处置	检查参数输入数值是否超出最小值。
排除方法	使用范围内的数值。

事件码	0x300E 参数编号错误
扩充码	N/A
原因	参数编号错误。
检查及处置	检查参数存取操作中，使用的参数编号是否在范围内，详细的参数说明请参考第 5 章。
排除方法	使用范围内的数值。

事件码	0x300F 参数不可写入
扩充码	N/A
原因	操作对象参数不可进行写入操作。
检查及处置	检查操作对象的读写属性是否为可写。
排除方法	取消对该参数的写入操作。

事件码	0x3010 参数不可在伺服使能时修改
扩充码	N/A
原因	操作对象参数不可在伺服使能时进行修改。
检查及处置	检查操作对象的读写属性是否为在伺服使能时能进行修改。
排除方法	1. 取消对该参数的修改。 2. 将伺服取消使能之后，再对该参数进行修改。

事件码	0x3011 SDO pReadData 错误
扩充码	N/A
原因	使用 SDO 功能块时，pReadData 脚位没有接变量。
检查及处置	检查该功能块的 pReadData 脚位是否有接变量。
排除方法	连接变量至该功能块的 pReadData 脚位。

事件码	0x3012 参数操作 pReadData 错误
扩充码	N/A
原因	使用系统参数功能块时，pReadData 脚位没有接变量。
检查及处置	检查该功能块的 pReadData 脚位是否有接变量。
排除方法	连接变量至该功能块的 pReadData 脚位。

事件码	0x3013 参数数值错误
扩充码	N/A
原因	参数输入数值不在合法范围内。
检查及处置	检查参数输入数值是否在合法范围内，详细的参数说明请参考第 5 章。
排除方法	修正输入数值至合法范围内。

事件码	0x3014 参数不可读取
扩充码	N/A
原因	操作对象参数不可进行读取操作。
检查及处置	检查操作对象的读写属性是否为可读。
排除方法	取消对该参数的读取操作。

事件码	0x3015 PDO pReadData 错误
扩充码	N/A
原因	使用 PDO 功能块时，pReadData 脚位没有接变量。
检查及处置	检查该功能块的 pReadData 脚位是否有接变量。
排除方法	连接变量至该功能块的 pReadData 脚位。

事件码	0x3016 参数读写超时
扩充码	N/A
原因	参数读写超时。
检查及处置	1. 检查参数读写从站是否正常响应。 2. 读写控制器配置文件参数时，控制器处于"待命模式"。
排除方法	1. 确认从站状况后，重新执行参数读写功能。 2. 将控制器切换到"联机操作"或"联机诊断"模式，并重新执行参数读写。

9

事件码	0x3030 使用配置文件发生空操作
扩充码	N/A
原因	使用配置文件过程中，控制器发生非法操作。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

事件码	0x3039 配置文件档案读取失败
扩充码	N/A
原因	使用配置文件时，读取失败。
检查及处置	1. 检查配置文件是否存在。 2. 确认配置文件是否没有损毁。
排除方法	1. 下载新的配置文件。 2. 将损毁的配置文件送回原厂进行检测。

事件码	0x303A 配置文件内容与实际硬件不符
扩充码	YXU：装置 ID
原因	无法在目前控制器连接的装置内，找到符合配置文件内设定的装置。
检查及处置	检查控制器是否有链接配置文件内设定的装置。
排除方法	1. 链接该装置。 2. 移除配置文件该装置的设定。

事件码	0x303B 配置文件使用了备份档
扩充码	N/A
原因	控制器使用配置文件时，由于找不到正确的配置文件，因此使用了备份的配置文件。
检查及处置	1. 确认控制器内有配置文件。 2. 确认配置文件是否有损毁。
排除方法	1. 下载新的配置文件。 2. 将损毁的配置文件送回原厂进行检测。

事件码	0x303C 控制器配置初始化失败
扩充码	N/A
原因	由于配置文件内容不正确，控制器使用配置文件进行初始化时，初始化失败。
检查及处置	确认配置文件是否为软件产生。
排除方法	将配置文件送回原厂进行分析。

事件码	0x303D Motion 核心初始化失败
扩充码	N/A
原因	建立 Motion 对象时，由于配置文件的内容有误而发生错误。
检查及处置	确认配置文件内关于轴、群组等 Motion 对象的配置是否有误。
排除方法	修正配置文件内 Motion 对象的内容。

事件码	0x303E 建立轴对象失败
扩充码	YXU：轴编号
原因	建立轴对象时，由于配置文件的内容有误而发生错误。
检查及处置	确认配置文件内关于轴对象的配置是否有误。
排除方法	修正配置文件内轴对象的内容。

事件码	0x303F 建立群组对象失败
扩充码	YXU：群组编号
原因	建立群组对象时，由于配置文件的内容有误而发生错误。
检查及处置	确认配置文件内关于群组对象的配置是否有误。
排除方法	修正配置文件内群组对象的内容。

事件码	0x3040 取得配置文件失败
扩充码	N/A
原因	取得配置文件失败。
检查及处置	检查配置文件是否意外损毁。
排除方法	利用 DMARS 软件重新下载配置文件。

9

事件码	0x3041 配置文件缺少主要参数
扩充码	N/A
原因	配置文件缺少主要参数。
检查及处置	检查控制器内部配置文件是否损毁。
排除方法	透过 DMARS 软件重新下载配置文件。

事件码	0x3042 装置初始化失败
扩充码	N/A
原因	装置配置文件错误造成装置初始化失败。
检查及处置	请连络原厂。
排除方法	请连络原厂。

事件码	0x3043 配置文件版本错误
扩充码	N/A
原因	韧体与配置文件版本不匹配。
检查及处置	检查控制器内部配置文件是否由 v2.1.1.10100 以前版本的 DMARS 软件生成。
排除方法	更新 DMARS 软件至 v2.1.1.10100 以上版本，并重新下载配置文件。

事件码	0x3044 读取编码器形态失败
扩充码	N/A
原因	透过 SDO 通讯读取伺服编码器形态时失败。
检查及处置	检查伺服是否在可接受 SDO 通讯存取的状态。
排除方法	将伺服切换至通讯模式，清除报警并重新将控制器切换至"联机操作模式"。

事件码	0x3045 伺服电子齿轮比写入失败
扩充码	N/A
原因	透过 SDO 通讯写入伺服电子齿轮比时失败。
检查及处置	1. 检查伺服是否在可接受 SDO 通讯存取的状态。 2. 伺服是否处于始能状态。
排除方法	1. 将伺服切换至通讯模式，清除报警并重新将控制器切换至"联机操作模式"。 2. 将伺服脱离始能状态，清除报警并重新将控制器切换至"联机操作模式"。

事件码	0x3046 读取伺服速度回授单位失败
扩充码	N/A
原因	透过 SDO 通讯读取伺服速度回授单位时失败。
检查及处置	检查伺服是否在可接受 SDO 通讯存取的状态。
排除方法	将伺服切换至通讯模式，清除报警并重新将控制器切换至"联机操作模式"。

事件码	0x3047 伺服软极限写入失败
扩充码	N/A
原因	透过 SDO 通讯写入伺服软极限时失败。
检查及处置	检查伺服是否在可接受 SDO 通讯存取的状态。
排除方法	将伺服切换至通讯模式，清除报警并重新将控制器切换至"联机操作模式"。

事件码	0x3048 伺服速度限制写入失败
扩充码	N/A
原因	透过 SDO 通讯写入伺服速度限制时失败。
检查及处置	1. 检查伺服是否在可接受 SDO 通讯存取的状态。 2. 检查配置文件上最大速度是否在马达速度限制范围内，马达型号及其速度限制请参考伺服应用技术手册。
排除方法	1. 将伺服切换至通讯模式，清除报警并重新将控制器切换至"联机操作模式"。 2. 修改配置文件最大速度，清除报警并重新将控制器切换至"联机操作模式"。 3. 更换马达，清除报警并重新将控制器切换至"联机操作模式"。

事件码	0x3049 错误的内建 MODBUS 站号配置
扩充码	N/A
原因	内建 MODBUS 站号有重复的站号设定。
检查及处置	确认内建 MODBUS 站号是否有重复。
排除方法	请勿重复设定内建 MODBUS 站号。

9

事件码	0x304A 配置文件使用了错误的机型
扩充码	N/A
原因	配置文件内的机型与当前控制器的机型不一致。
检查及处置	检查配置文件的机型是否与当前控制器的机型相同。
排除方法	请勿使用不同机型的配置文件。

事件码	0x3064 监控轴编号错误
扩充码	YXU：轴编号
原因	进行轴监控操作时，轴编号设定错误。
检查及处置	确认进行监控操作时，轴编号是否超出范围或该轴不存在。
排除方法	使用范围内的轴编号。

事件码	0x3065 监控群组编号错误
扩充码	YZU：群组编号
原因	进行群组监控操作时，群组编号设定错误。
检查及处置	确认进行监控操作时，群组编号是否超出范围或该轴不存在。
排除方法	使用范围内的群组编号。

事件码	0x3081 控制器状态切换失败
扩充码	N/A
原因	控制器进行状态切换时，由于硬件配置不符合等因素，而造成切换失败。
检查及处置	1. 确认配置文件的装置与硬件配置是否符合。 2. 确认总线是否正确连结。
排除方法	1. 链接缺少的硬件装置。 2. 从配置文件移除没有使用的硬件装置。

事件码	0x3082 无法在非 Edit Mode 重设 EtherCAT
扩充码	N/A
原因	只有在 Edit Mode 才能进行 EtherCAT 联机操作。
检查及处置	确认控制器模式是否为 Edit Mode。
排除方法	开关器模式至 Edit Mode，再进行 EtherCAT 的联机操作。

事件码	0x3083 无法在非 Edit Mode 进行 EtherCAT 重新扫站
扩充码	N/A
原因	只有在 Edit Mode 才能进行 EtherCAT 联机操作。
检查及处置	确认控制器模式是否为 Edit Mode。
排除方法	开关器模式至 Edit Mode，再进行 EtherCAT 的联机操作。

事件码	0x3084 无法在非 Edit Mode 设定 EtherCAT 从站别名
扩充码	N/A
原因	只有在 Edit Mode 才能进行 EtherCAT 联机操作。
检查及处置	确认控制器模式是否为 Edit Mode。
排除方法	开关器模式至 Edit Mode，再进行 EtherCAT 的联机操作。

事件码	0x3085 控制器共享内存初始失败
扩充码	N/A
原因	控制器建立共享内存时发生错误。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

事件码	0x3086 EtherCAT 共享内存初始失败
扩充码	N/A
原因	EtherCAT 建立共享内存时发生错误。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

事件码	0x3087 在非 Edit Mode 重新启动 EtherCAT
扩充码	N/A
原因	只有在 Edit Mode 才能进行 EtherCAT 联机操作。
检查及处置	确认控制器模式是否为 Edit Mode。
排除方法	开关器模式至 Edit Mode，再进行 EtherCAT 的联机操作。

9

事件码	0x3088 PLC 程序使用了版本不匹配的 FwLib
扩充码	N/A
原因	控制器韧体版本与 PLC FwLib 不匹配。
检查及处置	检查控制器韧体版本与 PLC FwLib 是否为兼容版本。
排除方法	1. 更新控制器韧体至匹配版本。 2. 更新 PLC FwLib 至匹配版本。

事件码	0x3089 控制器无法执行运动命令
扩充码	N/A
原因	控制器目前的模式无法执行运动命令。
检查及处置	1. 使用软件调机页面时，检查控制器目前是否处于联机诊断模式。 2. 使用 MODBUS 或 AH protocol 执行运动命令时，检查控制器是否处于联机操作或联机诊断模式。
排除方法	将控制器切换到正确模式。

事件码	0x308A 报警码重置超时
扩充码	N/A
原因	报警码无法清除。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

事件码	0x308C 控制器目标状态错误
扩充码	N/A
原因	控制器目标状态不支持。
检查及处置	检查设定的控制器目标状态是否为"待命"、"联机操作"或"联机诊断"。
排除方法	重新设定控制器目标模式，并重新执行功能。

事件码	0x308D RTC 电池低电压提醒
扩充码	N/A
原因	RTC 电池达到低电压警示范围。
检查及处置	检查 RTC 电池是否有安装正确，若问题仍存在请尽速更换电池。
排除方法	清除报警可继续使用，但需尽速更换电池。

事件码	0x308E 控制器低电压提醒
扩充码	N/A
原因	系统侦测到低电压，但目前电压仍是正常。
检查及处置	检查电源供应器或市电，确认电源稳定供应。
排除方法	检查电源供应器或市电，确认电源稳定供应。

事件码	0x308F 控制器掉电储存失败
扩充码	N/A
原因	控制器掉电处理流程未完成。
检查及处置	检查掉电前变更的数据是否正确，可重写再重开机确认。
排除方法	清除报警可继续操作控制器。若此报警持续发生，建议不启用控制器掉电处理。

事件码	0x3090 控制器异常
扩充码	N/A
原因	控制器遭遇异常情况。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

事件码	0x3102 反矩阵运算失败
扩充码	N/A
原因	执行反矩阵运算时失败。
检查及处置	确认执行反矩阵运算的矩阵能计算出反矩阵。
排除方法	输入有反矩阵的矩阵。

事件码	0x31A0 错误的圆弧模式
扩充码	YXU：群组编号
原因	使用圆弧命令时，使用了错误的圆弧模式。
检查及处置	确认圆弧模式设定值是否正确，详细的参数说明请参考章节 6.5.4 群轴功能块 (Motion Group)。
排除方法	使用正确的圆弧模式。

9

事件码	0x31A1 圆弧起始点与中间点相同
扩充码	YXU：群组编号
原因	圆弧起始点与中间点为同一点，无法成圆弧。
检查及处置	确认圆弧命令的起始点与中间点是否为同一点。
排除方法	请将圆弧的起始点与中间点设置为不同点。

事件码	0x31A2 圆弧起始点与结束点为同一点
扩充码	YXU：群组编号
原因	圆弧起始点与结束点为同一点，无法成圆弧。
检查及处置	确认圆弧命令的起始点与结束点是否为同一点。
排除方法	请将圆弧的起始点与结束点设置为不同点。

事件码	0x31A3 圆弧中间点与结束点为同一点
扩充码	YXU：群组编号
原因	圆弧中间点与结束点为同一点，无法成圆弧。
检查及处置	确认圆弧命令的中间点与结束点是否为同一点。
排除方法	请将圆弧的中间点与结束点设置为不同点。

事件码	0x31A4 圆面积过小
扩充码	YXU：群组编号
原因	圆面积过小，无法成圆。
检查及处置	确认圆命令形成的圆面积是否过小，详细的参数说明请参考章节 6.5.4 群轴功能块 (Motion Group)。
排除方法	更换圆命令，输入较大圆面积的命令。

事件码	0x31A5 圆弧起始点、中间点与结束点共线
扩充码	YXU：群组编号
原因	圆弧起始点、中间点与结束点共线，无法成圆弧。
检查及处置	确认圆弧命令的起始点、中间点与结束点是否在同一条在线。
排除方法	圆弧的起始点、中间点与结束点需设置在不同在线。

事件码	0x31A6 圆弧平面没有法向量

扩充码	YXU：群组编号
原因	无法从计算出的圆弧平面找到法向量。
检查及处置	确认圆弧命令设定值是否正确，详细的参数说明请参考章节 6.5.4 群轴功能块 (Motion Group)。
排除方法	使用正确的圆弧命令。

事件码	0x31A7 圆弧两点一圆心共线
扩充码	YXU：群组编号
原因	计算圆弧使用的两点与圆心共线，造成无法算出圆弧。
检查及处置	确认圆弧命令使用的两点与圆心不在同一条在线。
排除方法	使用不在同一条在线的两点与圆心。

事件码	0x31A8 圆半径太短
扩充码	YXU：群组编号
原因	圆半径太短，无法成圆。
检查及处置	确认圆弧命令造出的圆半径符合范围，详细的参数说明请参考章节 6.5.4 群轴功能块 (Motion Group)。
排除方法	修改圆弧命令，让圆半径不会太短。

事件码	0x31A9 圆弧起始点与圆心相同
扩充码	YXU：群组编号
原因	圆弧命令的起始点与圆心相同，无法成圆。
检查及处置	确认圆弧命令的起始点是否与圆心相同。
排除方法	设定圆弧命令时，请勿让起始点与圆心相同。

事件码	0x31AA 圆弧结束点与圆心相同
扩充码	YXU：群组编号
原因	圆弧命令的结束点与圆心相同，无法成圆。
检查及处置	确认圆弧命令的结束点是否与圆心相同。
排除方法	设定圆弧命令时，请勿让结束点与圆心相同。

9

事件码	0x31AB 圆平面没有 X 向量
扩充码	YXU：群组编号
原因	无法从计算出的圆弧平面找到 X 向量。
检查及处置	确认圆弧命令设定值是否正确。
排除方法	使用正确的圆弧命令。

事件码	0x4000 PLC 发生空操作
扩充码	N/A
原因	PLC 核心运行中，发生非法操作。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

事件码	0x4001 PLC system function lib 初始失败
扩充码	N/A
原因	PLC 核心初始化 system function lib 失败。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

事件码	0x4002 PLC shared memory lib 初始失败
扩充码	N/A
原因	PLC 核心初始化 shared memory lib 失败。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

事件码	0x4003 PLC IO lib 初始失败
扩充码	N/A
原因	PLC 核心初始化 IO lib 失败。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

事件码	0x4004 在错误的 PLC task 使用运动命令
扩充码	N/A
原因	在无法使用运动命令的 PLC task 使用了运动命令。
检查及处置	确认无法使用运动命令的 PLC task 是否有使用运动命令。
排除方法	将该运动命令移至可以使用运动命令的 PLC task。

事件码	0x4006 PLC task 运动命令缓存已满
扩充码	N/A
原因	在与 Motion 不同步的 PLC task 一个周期内填入太多运动命令。
检查及处置	检查是否在与 Motion 不同步的 PLC task 一个周期内填入太多运动命令。
排除方法	1. 减少一个周期内填入的运动命令。 2. 将部分运动命令延迟一个周期填入。

事件码	0x4007 PLC Fieldbus lib 初始失败
扩充码	N/A
原因	PLC 核心初始化 Fieldbus lib 失败。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

事件码	0x4008 PLC 建立 eclr 控件失败
扩充码	N/A
原因	PLC 核心建立 eclr 控件失败。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

事件码	0x4009 PLC 建立 eclr 应用领域失败
扩充码	N/A
原因	PLC 核心建立 eclr 应用领域失败。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

9

事件码	0x400A PLC 建立 eclr 系统领域失败
扩充码	N/A
原因	PLC 核心建立 eclr 系统领域失败。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

事件码	0x400B PLC 建立 eclr 数据 heap 失败
扩充码	N/A
原因	PLC 核心建立 eclr 数据 heap 失败。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

事件码	0x400C PLC 建立 eclr 程序代码 heap 失败
扩充码	N/A
原因	PLC 核心建立 eclr 程序代码 heap 失败。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

事件码	0x400D PLC 冷启加载失败
扩充码	N/A
原因	PLC 核心以冷启动的方式加载程序时失败。
检查及处置	1. 检查控制器内是否存有 PLC 程序文件。 2. 确认 PLC 程序文件使用的 FwLib 符合控制器支持的版本。
排除方法	1. 下载新的 PLC 程序文件至控制器。 2. 使用控制器支持的 FwLib 版本。

事件码	0x400E PLC 暖启加载失败
扩充码	N/A
原因	PLC 核心以热启动的方式加载程序时失败。
检查及处置	1. 检查控制器内是否存有 PLC 程序文件。 2. 确认 PLC 程序文件使用的 FwLib 符合控制器支持的版本。
排除方法	1. 下载新的 PLC 程序文件至控制器。 2. 使用控制器支持的 FwLib 版本。

事件码	0x400F PLC 静止加载失败
扩充码	N/A
原因	PLC 核心以静止启动的方式加载程序时失败。
检查及处置	1. 检查控制器内是否存有 PLC 程序文件。 2. 确认 PLC 程序文件使用的 FwLib 符合控制器支持的版本。
排除方法	1. 下载新的 PLC 程序文件至控制器。 2. 使用控制器支持的 FwLib 版本。

事件码	0x4010 未知的 PLC 启动加载模式
扩充码	N/A
原因	PLC 核心使用了未知的 PLC 启动加载模式。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

事件码	0x4011 未知的 PLC 启动法则
扩充码	N/A
原因	PLC 核心使用了未知的 PLC 启动法则。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

事件码	0x4012 PLC 任务超出最大数量
扩充码	N/A
原因	PLC 任务配置超过 10 组。
检查及处置	检查 PLC 任务配置数量。
排除方法	重新配置 PLC 任务。

事件码	0x4013 PLC 事件任务超出最大数量
扩充码	N/A
原因	PLC 事件任务超出 6 组 (包含 Event0)。
检查及处置	检查 PLC 事件任务配置数量。
排除方法	重新配置 PLC 事件任务。

9

事件码	0x4014 PLC 事件 ID 重复定义
扩充码	N/A
原因	PLC 事件 ID 重复定义。
检查及处置	检查 PLC 事件任务配置设定。
排除方法	重置并重新下载 PLC 程序。

事件码	0x4015 PLC 使用了错误的事件 ID
扩充码	N/A
原因	PLC 使用了错误的事件 ID。
检查及处置	检查 PLC 事件任务配置设定。
排除方法	重置并重新下载 PLC 程序。

事件码	0x4016 PLC 事件任务配置文件不存在
扩充码	N/A
原因	PLC 事件任务配置文件不存在。
检查及处置	检查 PLC 事件任务配置设定是否错误。
排除方法	重置并重新下载控制器配置文件。

事件码	0x4017 PLC 事件任务配置内容错误
扩充码	N/A
原因	PLC 事件任务配置内容错误。
检查及处置	检查 PLC 事件任务配置设定是否错误。
排除方法	重置并重新下载控制器配置文件。

事件码	0x4018 PLC 事件任务未进行配置
扩充码	N/A
原因	PLC 事件任务未进行配置。
检查及处置	检查 PLC 事件任务配置是否已设定。
排除方法	重置并重新下载控制器配置文件。

事件码	0x4019 PLC 核心冷启动失败
扩充码	N/A
原因	PLC 核心冷启动失败。
检查及处置	1. 检查控制器内是否存有 PLC 程序文件。 2. 确认 PLC 程序文件使用的 FwLib 符合控制器支持的版本。
排除方法	1. 下载新的 PLC 程序文件至控制器。 2. 使用控制器支持的 FwLib 版本。

事件码	0x401A PLC 核心停止失败
扩充码	N/A
原因	PLC 核心停止失败。
检查及处置	1. 检查控制器内是否存有 PLC 程序文件。 2. 确认 PLC 程序文件使用的 FwLib 符合控制器支持的版本。
排除方法	1. 下载新的 PLC 程序文件至控制器。 2. 使用控制器支持的 FwLib 版本。

事件码	0x401B 触发 PLC 的 DI 编号不正确
扩充码	N/A
原因	触发 PLC 的 DI 编号不正确。
检查及处置	检查触发 PLC 的 DI 编号是否正确。
排除方法	将触发 PLC 的 DI 编号设定在 0 ~ 15 之间。

事件码	0x401C PLC DMC function lib 初始失败
扩充码	N/A
原因	函式库版本不匹配。
检查及处置	确认函式库版本是否与韧体版本匹配。
排除方法	更换函式库或韧体。

9

事件码	0x4100 未知的 PLC 例外
扩充码	N/A
原因	PLC 发生了未知的例外。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

事件码	0x4101 PLC 程序发生字符串错误
扩充码	N/A
原因	PLC 程序中存在错误的字符串操作，例如存取超出范围等。
检查及处置	检查 PLC 程序中是否存在错误的字符串操作。
排除方法	修正错误的字符串操作。

事件码	0x4102 PLC 任务运行时间超出 watch dog 设定值
扩充码	N/A
原因	PLC 任务所需运行时间过长。
检查及处置	1. 确认 PLC 任务是否存在运行时间过长的程序。 2. 确认 watch dog 设定值是否正确。
排除方法	1. 移除运行时间过长的程序。 2. 修正 watch dog 设定值。

事件码	0x4103 执行 PLC 任务超出 CPU 负荷
扩充码	N/A
原因	PLC 任务所需运行时间过长。
检查及处置	确认 PLC 任务是否存在无穷循环或所需时间太长的程序。
排除方法	修正 PLC 程序。

事件码	0x4104 PLC 核心发生系统例外
扩充码	N/A
原因	PLC 核心发生系统例外。
检查及处置	此错误为内部错误，请联系原厂。
排除方法	此错误为内部错误，请联系原厂。

事件码	0x4105 PLC 控制命令不支持
扩充码	N/A
原因	透过 SDV 控制 PLC 时，命令功能码不支持。
检查及处置	检查命令功能码，详细的参数说明请参考附录 A 的 MODBUS 地址表。
排除方法	清除报警，填入正确功能码并重新触发 PLC 控制功能。

事件码	0x4106 停止 PLC IO 更新流程失败
扩充码	N/A
原因	PLC IO 更新配置不正确。
检查及处置	确认 IO 更新配置是否正确。
排除方法	删除原先配置，由软件重新配置。

事件码	0x5000 所有的探针都在使用中
扩充码	N/A
原因	所有的探针都在使用中。
检查及处置	1. 检查是否有一直处于 Busy 状态无法完成的 DMC_TouchProbe 功能块。 2. 检查执行中的 DMC_TouchProbe 数量是否已达到最大上限。
排除方法	1. 使用 MC_AbortTrigger 来中断指定触发事件之功能块。 2. 等待执行中的探针完成动作再执行 DMC_TouchProbe。

事件码	0x5001 触发 ID 已被使用
扩充码	N/A
原因	DMC_TouchProbe 功能块指定的探针正在使用中，使用中的探针无法重复使用。
检查及处置	检查是否有 DMC_TouchProbe 功能块针对同一探针执行。
排除方法	将 DMC_TouchProbe 设定为使用空闲的探针。

事件码	0x5002 触发 ID 没有被使用
扩充码	N/A
原因	MC_AbortTrigger 欲中断的探针为空闲状态。
检查及处置	检查欲中断的探针是否在使用中。
排除方法	设定正确的探针并重新触发功能块。

9

事件码	0x5003 探针触发事件轴 (Axis) 错误
扩充码	N/A
原因	设定的触发事件轴 (Axis) 错误。
检查及处置	1. 检查功能块设定的触发事件轴是否已配置在配置文件内。 2. 检查配置文件是否已成功下载。
排除方法	1. 设定正确的触发事件轴。 2. 下载正确的配置文件。

事件码	0x5004 触发模式 (TriggerType) 超出范围
扩充码	N/A
原因	设定的触发模式 (TriggerType) 超出范围。
检查及处置	检查设定的触发模式 (TriggerType) 是否超出范围。 0：驱动器模式 1：控制器模式
排除方法	将触发模式 (TriggerType) 设定为驱动器模式或控制器模式。

事件码	0x5005 探针驱动器模式 (Drive Mode) 只能使用实体轴
扩充码	N/A
原因	使用 DMC_TouchProbe 功能块时，虚拟轴和实体编码器轴不支持驱动器模式 (Drive Mode)。
检查及处置	检查是否在虚拟轴和实体编码器轴的装置使用驱动器模式 (Drive Mode)。
排除方法	1. 将装置更换为实体轴。 2. 将 TriggerMode 设定为控制器模式 (Controller Mode)。

事件码	0x5006 触发 ID (TriggerID) 超出范围
扩充码	N/A
原因	触发 ID (TriggerID) 超过各轴探针数量。
检查及处置	检查触发 ID (TriggerID) 是否大于 5。
排除方法	将触发 ID (TriggerID) 设定在 0 ~ 5 之间。

事件码	0x5007 驱动器模式 (Drive Mode) 触发 ID (TriggerID) 超出范围
扩充码	N/A
原因	驱动器模式 (Drive Mode) 下，触发 ID (TriggerID) 超出范围。
检查及处置	检查触发 ID (TriggerID) 是否大于 1。
排除方法	将触发 ID (TriggerID) 设定在 0 ~ 1 之间。

事件码	0x5008 控制器模式 (Controller Mode) 触发 ID (TriggerID) 超出范围
扩充码	N/A
原因	控制器模式 (Controller Mode) 下，触发 ID (TriggerID) 超出范围。
检查及处置	检查触发 ID (TriggerID) 是否小于 2。
排除方法	将触发 ID (TriggerID) 设定在 2 ~ 5 之间。

事件码	0x5009 触发类型 (TriggerType) 超出范围
扩充码	N/A
原因	所设定的触发类型 (TriggerType) 不支持。
检查及处置	检查设定的触发类型 (TriggerType) 是否支持。 0：本地 DI 触发 1：扩充 DI 触发 2：远程 DI 触发 3：PLC 变数触发
排除方法	将触发类型 (TriggerType) 设定为本地 DI 触发、扩充 DI 触发、远程 DI 触发或 PLC 变数触发。

事件码	0x500A 驱动器模式 (Drive Mode) 只能使用本地 DI 触发
扩充码	N/A
原因	在驱动器模式 (Drive Mode) 下，仅支持本地 DI 触发。
检查及处置	检查是否在驱动器模式 (Drive Mode) 下，使用本地 DI 触发模式以外的触发类型。
排除方法	将触发模式设定为本地 DI 触发。

9

事件码	0x500B 探针 DI 编号 (DINumber) 超出范围
扩充码	N/A
原因	探针 DI 编号 (DINumber) 超出支持的范围。
检查及处置	检查探针 DI 编号 (DINumber) 是否错误。
排除方法	设定正确探针 DI 编号 (DINumber)。

事件码	0x500C PLC 变量触发 DI 编号 (DINumber) 必须为零
扩充码	N/A
原因	在 PLC 变量触发模式下，探针 DI 编号 (DINumber) 必须为零。
检查及处置	检查探针 DI 编号 (DINumber)。
排除方法	将探针 DI 编号 (DINumber) 设定为零。

事件码	0x500D 驱动器模式 (Drive Mode) 下 DI 编号 (DINumber) 设定错误
扩充码	N/A
原因	在驱动器模式 (Drive Mode) 下，探针 DI 编号 (DINumber) 必须在 1 ~ 2 之间。
检查及处置	检查探针 DI 编号 (DINumber)。
排除方法	将探针 DI 编号 (DINumber) 设定在 1 ~ 2 之间。

事件码	0x500E 探针数据记录总类 (RecordedDate) 设定超出范围
扩充码	N/A
原因	设定的数据记录总类 (RecordedDate) 不支持。
检查及处置	检查是否正确设定数据记录总类 (RecordedDate)。
排除方法	设定正确数据记录总类 (RecordedDate)。

事件码	0x500F 驱动器模式 (Drive Mode) 下探针数据记录总类 (RecordedDate) 设定错误
扩充码	N/A
原因	在驱动器模式 (Drive Mode) 下，探针数据记录总类 (RecordedDate) 必须在 0 ~ 2 之间。
检查及处置	检查探针数据记录总类 (RecordedDate)。
排除方法	将探针数据记录总类 (RecordedDate) 设定在 0 ~ 2 之间。

事件码	0x5010 控制器模式 (Controller Mode) 使用本地 DI 触发型态时，探针数据记录总类 (RecordedDate) 设定错误
扩充码	N/A
原因	在控制器模式 (Controller Mode) 下，使用本地 DI 触发型态时，探针数据记录总类 (RecordedDate) 必须大于 3。
检查及处置	检查探针数据记录总类 (RecordedDate)。
排除方法	将探针数据记录总类 (RecordedDate) 设定大于 3。

事件码	0x5011 探针数据纪录设定错误
扩充码	N/A
原因	使用非本地 DI 触发型态时，探针数据记录总类 (RecordedDate) 必须大于等于 10。
检查及处置	检查探针数据记录总类 (RecordedDate)。
排除方法	将探针数据记录总类 (RecordedDate) 设定大于等于 10。

事件码	0x5012 探针 window 设定错误
扩充码	N/A
原因	在开启 window 功能的情况下，起始值 (FirstValue) 与末端值 (LastValue) 设定错误。
检查及处置	检查起始值 (FirstValue) 与末端值 (LastValue) 设定。
排除方法	设定正确起始值 (FirstValue) 与末端值 (LastValue)。

事件码	0x5013 PDO 配置错误
扩充码	YXU：轴编号
原因	使用伺服 capture 功能，但未将 0x60BA 与 0x60BB 放入 PDO 配置。
检查及处置	检查 PDO 配置内是否有 0x60BA 与 0x60BB。
排除方法	将 0x60BA 与 0x60BB 放入 PDO 配置。

事件码	0x5014 window 数据型态不支持
扩充码	YXU：轴编号
原因	window 数据型态 (WindowType) 超出设定范围。
检查及处置	检查 window 数据型态 (WindowType) 脚位设定是否超出范围。
排除方法	将 window 数据型态 (WindowType) 脚位设定在合理范围内。

9

事件码	0x5015 本地触发 DI 编号设定错误
扩充码	YXU：轴编号
原因	使用本地 DI 触发型态 (Local DI) 时，触发 DI 编号 (DINumber) 设定错误。
检查及处置	检查触发 DI 编号 (DINumber) 是否设定正确。
排除方法	设定合理的触发 DI 编号 (DINumber)，并重新执行功能块。

事件码	0x5016 无效的扩充 DI 编号
扩充码	YXU：轴编号
原因	使用扩充 DI 触发型态 (Extension DI) 时，触发 DI 编号 (DINumber) 设定错误。
检查及处置	检查触发 DI 编号 (DINumber) 是否设定正确。
排除方法	设定合理的触发 DI 编号 (DINumber)，并重新执行功能块。

事件码	0x5017 无效的远程 DI 编号
扩充码	YXU：轴编号
原因	使用远程 DI 触发型态 (Remote DI) 时，触发 DI 编号 (DINumber) 设定错误。
检查及处置	检查触发 DI 编号 (DINumber) 是否设定正确。
排除方法	设定合理的触发 DI 编号 (DINumber)，并重新执行功能块。

事件码	0x5018 PLC 变量触发 DI 编号设定错误
扩充码	YXU：轴编号
原因	使用 PLC 变量触发型态 (PLC Variable) 时，触发 DI 编号 (DINumber) 设定错误。
检查及处置	检查触发 DI 编号 (DINumber) 是否设定正确。
排除方法	设定合理的触发 DI 编号 (DINumber)，并重新执行功能块。

事件码	0x5019 无效的本地 DI 编号
扩充码	YXU：轴编号
原因	使用本地 DI 触发型态 (Local DI) 时，触发 DI 编号 (DINumber) 设定错误。
检查及处置	检查触发 DI 编号 (DINumber) 是否设定正确。
排除方法	设定合理的触发 DI 编号 (DINumber)，并重新执行功能块。

事件码	0x6000 总线装置存在错误
扩充码	YXU：总线装置 ID
原因	使用功能块时，对象装置存在错误。
检查及处置	确认轴错误码，以便确认装置上的错误码。
排除方法	排除造成装置回报错误的原因。

事件码	0x6001 错误的总线类型
扩充码	YXU：总线装置 ID
原因	使用总线操作的功能块时，选择了错误的总线类型，详细的参数说明请参考章节 5.5 轴参数说明。
检查及处置	确认功能块是否选择了正确的总线类型。
排除方法	使用正确的总线类型。

事件码	0x6002 PLC 功能块的输入变量长度不合法
扩充码	YXU：总线装置 ID
原因	PLC 功能块的输入变量长度不符合功能块定义。
检查及处置	确认该功能块的输入变量数据长度。
排除方法	修正该变量的数据长度。

事件码	0x6003 PLC 功能块的输入变量编号不合法
扩充码	YXU：总线装置 ID
原因	PLC 功能块的输入变量编号不符合功能块定义。
检查及处置	确认该功能块的输入变量数值。
排除方法	修正该变数的数值。

事件码	0x6100 EtherCAT 扫站失败
扩充码	N/A
原因	EtherCAT 执行扫站时发生错误。
检查及处置	确认总线连接正常。
排除方法	将总线连接正确。

9

事件码	0x6101 EtherCAT 从站信息取得失败
扩充码	N/A
原因	EtherCAT 取得从站信息时发生错误。
检查及处置	确认总线连接正常。
排除方法	将总线连接正确。

事件码	0x6102 切换 EtherCAT 至 OP 模式失败
扩充码	N/A
原因	EtherCAT 切换至 OP 模式时发生错误。
检查及处置	确认总线连接正常。
排除方法	将总线连接正确。

事件码	0x6103 EtherCAT 断线
扩充码	N/A
原因	EtherCAT 运作时，发生断线。
检查及处置	确认总线连接正常。
排除方法	重新启动 EtherCAT 联机。

事件码	0x6104 切换 EtherCAT 至 Init.模式失败
扩充码	N/A
原因	EtherCAT 切换至 Init.模式时发生错误。
检查及处置	确认 EtherCAT 模式目前是否已经在 Init.模式。
排除方法	在 Init.模式下无法再执行切换至 Init.模式。

事件码	0x6105 设定 EtherCAT 从站别名失败
扩充码	N/A
原因	设定指定站号时发生错误。
检查及处置	检查站号是不是没有依照设定的站号更新。
排除方法	确认这个站号是不是在合法范围内。

事件码	0x6106 储存 EtherCAT 从站别名失败
扩充码	N/A
原因	储存设定的指定站号时发生错误。
检查及处置	检查站号是不是没有依照设定的站号更新。
排除方法	确认这个站号是不是在合法范围内。

事件码	0x6107 EtherCAT 网络程序错误
扩充码	N/A
原因	韧体版本不匹配。
检查及处置	韧体版本不匹配，请更新正确的韧体版本。
排除方法	请更新正确的韧体版本。

事件码	0x6108 网络联机有问题
扩充码	N/A
原因	网络卡故障或是网络线没有接好。
检查及处置	查看网络卡或是网络线状况。
排除方法	确认网络卡是否正常，网络线是否接好。

事件码	0x6109 硬件没有正确初始化
扩充码	N/A
原因	硬件标识符错误。
检查及处置	更新正确的韧体版本。
排除方法	更新正确的韧体版本。

事件码	0x610A 没有找到任何 EtherCAT 模块
扩充码	N/A
原因	扫描不到任何模块。
检查及处置	确认有正常连接网络相关设备。
排除方法	确认网络卡是否正常，网络线是否接好。

9

事件码	0x610B 找不到模块对应的 dat 档案
扩充码	N/A
原因	模块没有对应的 dat。
检查及处置	检查是否缺少相关的 dat 档案。
排除方法	补上缺少的 dat 档案。

事件码	0x610C OD 数量太多
扩充码	N/A
原因	PDO 封包长度超过限制。
检查及处置	查看 OD 数量是否过多。
排除方法	减少 OD 数量或站数。

事件码	0x610D 超过支持的站数
扩充码	N/A
原因	超过支持的站数。
检查及处置	检查是否超过支持的站数。
排除方法	减少站数。

事件码	0x610E OD 映射错误
扩充码	N/A
原因	PDO 配置错误。
检查及处置	模块是否没有对应的 dat 档案。
排除方法	增加对应的 dat 档案。

事件码	0x610F 切换 op mode 失败
扩充码	N/A
原因	等待切换 op 超过时间。
检查及处置	模块对应的 dat 参数设定错误。
排除方法	1. 检查模块 DC 时间设置是否符合。 2. 检查模块 PDO 组成设定是否符合。 3. 检查模块错误码并排除该错误。

事件码	0x6110 EtherCAT 模块设定失败
扩充码	N/A
原因	EtherCAT 模块设定失败。
检查及处置	模块是否没有对应的 dat 档案。
排除方法	增加对应的 dat 档案。

事件码	0x6111 EtherCAT 状态不在 OP 模式
扩充码	N/A
原因	未将 EtherCAT 状态切换至 OP 模式。
检查及处置	检查软件 EtherCAT 灯号是否为"执行"。
排除方法	重新将控制器切换到"联机操作"模式。

事件码	0x6112 EtherCAT PRE-OP 模式切换失败
扩充码	N/A
原因	EtherCAT 无法切换至 PRE-OP 模式。
检查及处置	检查 EtherCAT 总线是否掉线或接触不良。
排除方法	确认线路无误后，重新执行扫站。

事件码	0x6C01 DMCNET 操作指定站号错误
扩充码	N/A
原因	DMCNET 指令指定的站号未联机或超过范围。
检查及处置	检查指令的设定站号是否符合联机状态与通讯规范。
排除方法	设定正确站号。

事件码	0x6C02 DMCNET 网络联机未建立
扩充码	N/A
原因	DMCNET 网络未完成初始化或联机状态异常。
检查及处置	检查各装置的网络连接是否正常且网络初始化动作是否已完成。
排除方法	确定 DMCNET 联机正常。

9

事件码	0x6C03 DMCNET SDO 存取指令数据类型错误
扩充码	N/A
原因	利用 SDO 读写数据的指令，其指定的数据类型错误。
检查及处置	检查指令设定的数据类型是否符合规范。
排除方法	针对读写数据的对象，设定正确数据类型。

事件码	0x6C04 DMCNET 传送 SDO 封包错误
扩充码	N/A
原因	系统无法处理 DMCNET SDO 封包发送请求。
检查及处置	检查 DMCNET 线路是否正常。
排除方法	重新初始化 DMCNET 联机。

事件码	0x6C05 DMCNET 接收 SDO 封包错误
扩充码	N/A
原因	系统无法处理 DMCNET SDO 封包接收请求。
检查及处置	检查 DMCNET 线路是否正常。
排除方法	重新初始化 DMCNET 联机。

事件码	0x6C06 DMCNET 接收 SDO 回应封包尚未完成
扩充码	N/A
原因	DMCNET SDO 通讯流程进行中，尚在等待响应封包。
检查及处置	SDO 通讯响应需要时间处理，检查处理时间设定是否合理。
排除方法	设定合理的 SDO 通讯指令处理时间。

事件码	0x6C08 DMCNET SDO 读写数据的指令，响应内容的数据长度超过所要求的
扩充码	N/A
原因	利用 SDO 读写数据的指令，其指定的数据长度小于存取对象的实际长度。
检查及处置	检查指令设定的数据长度是否匹配存取对象的实际长度。
排除方法	针对读写数据的对象，设定正确数据长度。

事件码	0x6C09 DMCNET 网络切换操作模式进行中
扩充码	N/A
原因	切换 DMCNET 操作模式已经启动，在进行中又重复下达切换操作模式。
检查及处置	切换 DMCNET 的操作模式需要时间，检查处理时间设定是否合理。
排除方法	设定合理的切换 DMCNET 操作模式处理时间。

事件码	0x6C0B DMCNET 网络执行状态未知
扩充码	N/A
原因	DMCNET 网络当前状态未明，无法进行切换模式。
检查及处置	DMCNET 网络执行状态出现异常，需要重置 DMCNET 联机。
排除方法	重置 DMCNET 联机，再执行切换 DMCNET 操作模式。

事件码	0x6C0C 检查 DMCNET 切换操作模式进行状态：进行中
扩充码	N/A
原因	DMCNET 切换操作模式仍在进行中。
检查及处置	切换 DMCNET 的操作模式需要时间，检查处理时间设定是否合理。
排除方法	设定合理的切换 DMCNET 操作模式处理时间。

事件码	0x6C0D 检查 DMCNET 切换操作模式进行状态：失败
扩充码	N/A
原因	DMCNET 网络无法切换操作模式。
检查及处置	检查 DMCNET 联机装置的 DMCNET 通信设置是否正确。
排除方法	设定正确的 DMCNET 通讯选项，重置 DMCNET 后再重新切换其操作模式。

事件码	0x6C0E 检查 DMCNET 切换操作模式进行状态：未启动
扩充码	N/A
原因	尚未启动切换 DMCNET 操作模式。
检查及处置	检查切换 DMCNET 操作模式流程，确定启动指令有执行。
排除方法	安排正确的 DMCNET 启动流程。

9

事件码	0x6C0F DMCNET 网络尚未切换到可操作模式
扩充码	N/A
原因	DMCNET 没有切换到可操作模式。
检查及处置	检查 DMCNET 是否有正确下达切换操作模式并已完成无误。
排除方法	确定 DMCNET 网络已切换至可操作模式，再重新执行相关操作。

事件码	0x6C11 系统无法取得响应的 DMCNET PDO 封包
扩充码	N/A
原因	系统无法取得更新的 DMCNET PDO 响应，可能是通讯干扰造成数据被芯片丢弃或断线造成收不到数据。
检查及处置	检查 DMCNET 网络联机状态。
排除方法	确认 DMCNET 网络联机正常。

事件码	0x6C12 DMCNET 设定 DO 模块数据长度错误
扩充码	N/A
原因	DMCNET 设定 DO 模块数据量过大。
检查及处置	检查 DO 数据设定是否正确。
排除方法	设定正确 DO 模块数据长度，重新下达设定 DO 数据指令。

事件码	0x6C13 DMCNET 设定 DO 数据模块信息错误
扩充码	N/A
原因	下达设定 DO 数据指令于没有 DO 输出的 DMCNET 模块。
检查及处置	检查指令设定的 DMCNET 装置是否适当。
排除方法	确认设定 DO 数据指令于正确的 DMCNET 装置。

事件码	0x6C14 DMCNET 网络中无任何可识别装置
扩充码	N/A
原因	DMCNET 网络初始化完成，但没有任何装置切换到可操作模式。
检查及处置	检查 DMCNET 联机装置的通讯格式设定是否正确。
排除方法	确认 DMCNET 联机装置的通信设置正确，再重置 DMCNET 网络并切换操作模式。

事件码	0x6C15 DMCNET 切换操作模式逾时
扩充码	N/A
原因	切换 DMCNET 操作模式超过 30 秒未完成。
检查及处置	检查 DMCNET 联机装置的通信设置是否正确。
排除方法	确认 DMCNET 联机装置的通信设置正确，再重置 DMCNET 网络并切换操作模式。

事件码	0x6C16 DMCNET SDO 通讯响应封包格式不符
扩充码	N/A
原因	DMCNET SDO 回应封包的标头异常。
检查及处置	检查 SDO 通讯对象执行状态是否正常。
排除方法	确认 SDO 通讯对象可以正常操作，重新下达 SDO 读写指令。

事件码	0x6C17 DMCNET SDO 通讯响应封包站号不符
扩充码	N/A
原因	DMCNET SDO 响应封包的站号信息与发送记录不符。
检查及处置	此为内部错误，请联系原厂。
排除方法	此为内部错误，请联系原厂。

事件码	0x6C18 DMCNET SDO 通讯响应封包记录数据长度为零
扩充码	N/A
原因	DMCNET SDO 响应封包显示数据长度为零。
检查及处置	此为内部错误，请联系原厂。
排除方法	此为内部错误，请联系原厂。

事件码	0x6C19 DMCNET SDO 读写指令内容不合法
扩充码	N/A
原因	SDO 读写指令欲存取的数据不被 DMCNET 联机装置接受。
检查及处置	检查要读取的地址是否合法、要写入的数值是否符合规范。
排除方法	确认要读写的数据都符合规范，再重新下达读写指令。

9

事件码	0x6C1A DMCNET monitor id 设定错误
扩充码	N/A
原因	monitor id 设定超出范围。
检查及处置	确认 monitor id 是否设定超出范围。
排除方法	修正 monitor id。

事件码	0x6C1B DMCNET 通讯硬件初始化失败
扩充码	N/A
原因	无法正确初始化硬件。
检查及处置	此为内部错误，请联系原厂。
排除方法	此为内部错误，请联系原厂。

事件码	0x6D01 MODBUS 命令错误
扩充码	N/A
原因	MODBUS master 通讯异常，回传值显示 slave 端不支持该指令。
检查及处置	检查 slave 端支持的 MODBUS 通讯指令是否与 MODBUS 功能块相符。
排除方法	选择适合的 MODBUS 功能块。

事件码	0x6D02 MODBUS 存取地址错误
扩充码	N/A
原因	MODBUS master 通讯异常，回传值显示存取的地址不合法。
检查及处置	检查 slave 端可用的 MODBUS 通讯可存取址址范围。
排除方法	设定合理的存取地址，并重新执行功能块。

事件码	0x6D03 MODBUS 写入值错误
扩充码	N/A
原因	MODBUS master 通讯异常，回传值显示要写入 slave 的值不合法。
检查及处置	检查写入地址是否为 slave 端的有效值。
排除方法	设定合理的写入值，并重新执行功能块。

事件码	0x6D04 MODBUS 通讯地址不合法
扩充码	N/A
原因	输入的内存起始位置错误。
检查及处置	确认地址是否正确。
排除方法	修正地址。

事件码	0x6D05 MODBUS 通讯数据长度不合法
扩充码	N/A
原因	通讯数据长度不正确。
检查及处置	确认通讯数据长度。
排除方法	修正数据长度。

事件码	0x6D06 MODBUS 通讯不支持存取模拟输入/输出
扩充码	N/A
原因	通讯地址存取到了模拟输入/输出的范围。
检查及处置	确认通讯地址。
排除方法	修正通讯地址。

事件码	0x6D20 MODBUS 功能块传入不支持的指令代码
扩充码	N/A
原因	MODBUS master 通讯异常，功能块传入不支持的指令代码。
检查及处置	检查功能块指定的指令代码是否正确。
排除方法	设定正确的指令代码，并重新执行功能块。

事件码	0x6D21 MODBUS master connect 脚位设定错误
扩充码	N/A
原因	MODBUS master connect 功能块传入错误的网络设定值（IP/Port)；或者 disconnect、serial_set 或 read/write 的功能块传入错误的联机代号。
检查及处置	检查功能块的相关参数设定是否正确。
排除方法	修正功能块的相关参数设定。

9

事件码	0x6D22 MODBUS 通讯未完成
扩充码	N/A
原因	还在等待 slave 回传响应封包的阶段，此次通讯仍在进行中。
检查及处置	更新韧体或对应功能块的 FwLib。
排除方法	更新韧体或对应功能块的 FwLib。

事件码	0x6D23 MODBUS 所有联机资源都在使用中
扩充码	N/A
原因	MODBUS master connect 功能块显示所有联机资源都在使用中，无法取得新的联机。
检查及处置	检查已启动的联机数是否已达到最大可用量，最大联机数为 6。
排除方法	关闭用不到的联机，再重新执行此功能块。

事件码	0x6D24 MODBUS 建立联机失败
扩充码	N/A
原因	MODBUS master connect 建立联机失败，可能是 slave 不支持指定的 port 或联机操作时间超过逾时设定值。
检查及处置	检查 slave 端的联机设定是否与 connect 功能块的参数相符。
排除方法	设定合理的联机参数，并重新执行功能块。

事件码	0x6D25 MODBUS 通讯逾时
扩充码	N/A
原因	MODBUS master 送出指令封包后，超过逾时设定值的时间仍未收到 slave 的响应。
检查及处置	检查 slave 端的通讯效能规格，是否需要更多时间才能处理通讯指令。
排除方法	设定合理的通讯逾时时间，并重新执行功能块。

事件码	0x6D26 MODBUS master 无法开启对应的 UART port
扩充码	N/A
原因	透过 serial 的 MODBUS master 无法开启对应的 UART port。
检查及处置	检查 serial_set 功能块的参数设定是否正确。
排除方法	设定正确的联机参数，并重新执行功能块。

事件码	0x6D27 MODBUS 设定网络 socket 失败
扩充码	N/A
原因	MODBUS master connect 过程中，设定网络 socket 失败。
检查及处置	内部错误，请更新韧体。
排除方法	内部错误，请更新韧体。

事件码	0x6D28 MODBUS 无法与系统的网络接口正确链接
扩充码	N/A
原因	MODBUS master connect 过程中，无法与系统的网络接口正确链接。
检查及处置	内部错误，请更新韧体。
排除方法	内部错误，请更新韧体。

事件码	0x6D29 MODBUS 被 slave 端拒绝
扩充码	N/A
原因	MODBUS master connect 过程中，被 slave 端拒绝。
检查及处置	检查联机设定是否对应正确的 slave，或者 slave 已有其他联机占用。
排除方法	确认参数设定的联机对象可被使用，再重新执行功能块。

事件码	0x6D2A MODBUS 读写功能块发送指令封包发生错误
扩充码	N/A
原因	MODBUS master 读写功能块发送指令封包发生错误。
检查及处置	检查联机是否已被 slave 断线。
排除方法	开启 Ethernet「保持联机」功能，请于 DMARS 软件中设定合理的「保持联机封包时间间隔」，并重新执行功能块。

事件码	0x6D2B MODBUS 读写功能块读取响应封包发生错误
扩充码	N/A
原因	MODBUS master 读写功能块读取响应封包发生错误。
检查及处置	检查通讯是否受到干扰或联机已被 slave 断线。
排除方法	开启 Ethernet「保持联机」功能，请于 DMARS 软件中设定合理的「保持联机封包时间间隔」，并重新执行功能块。

9

事件码	0x6D2C MODBUS 功能块尝试关闭未使用的联机代号
扩充码	N/A
原因	MODBUS master disconnect 功能块尝试关闭未使用的联机代号。
检查及处置	检查输入的联机代号是否正确。
排除方法	设定正确的联机代号，并重新执行功能块。

事件码	0x6D2D MODBUS 读写的功能块指定了尚未建立联机的联机代号
扩充码	N/A
原因	MODBUS master 读写的功能块指定了尚未建立联机的联机代号。
检查及处置	检查输入的联机代号是否已建立联机，如何建立联机并取得联机代号请参考章节 6.5.7 自定义通讯功能块 (DMC FieldBus)。
排除方法	设定合理的联机代号，并重新执行功能块。

事件码	0x6D2E MODBUS 读写功能块指定的联机代号尚未完成上次的通讯要求
扩充码	N/A
原因	MODBUS master 读写功能块指定的联机代号尚未完成上次的通讯要求。
检查及处置	检查是否有其他读写功能块指定相同的联机代号而尚未完成。
排除方法	确定设定的联机代号可被使用，并重新执行功能块。

事件码	0x6D2F MODBUS master disconnect 功能块在设定的逾时时间内无法完成关闭联机的动作
扩充码	N/A
原因	MODBUS master disconnect 功能块在设定的逾时时间内无法完成关闭联机的动作。
检查及处置	检查逾时参数是否设定太小。
排除方法	设定合理的逾时时间，并重新执行功能块。

事件码	0x6D30 MODBUS 通讯命令缓存已满
扩充码	N/A
原因	缓存内的命令来不及处理。
检查及处置	确认使用通讯命令的频率或从站回复时间。
排除方法	等待缓存有空间再填入新命令。

事件码	0x7000 DRL 控制命令错误
扩充码	N/A
原因	透过 SDV 控制 DRL 时发生错误。
检查及处置	检查 DRL 控制命令功能码是否正确，详细的参数说明请参考附录 A 的 MODBUS 地址表。
排除方法	清除报警并重新触发 DRL 控制功能。

9.3 SDO 终止传输代码

9

下表列出了 SDO 通讯错误的中止代码：

代码	说明
0x0503 0000	分段传输时，转换位没有变化。
0x0504 0000	SDO 传输超时。
0x0504 0001	命令码无效或未知。
0x0504 0002	无效的 block 大小 (仅 block 模式)。
0x0504 0003	无效的序列号 (仅 block 模式)。
0x0504 0004	CRC 错误。
0x0504 0005	内存溢出。
0x0601 0000	不支持对此对象的操作。
0x0601 0001	尝试读取一个唯写对象。
0x0601 0002	尝试写入一个只读对象。
0x0602 0000	对象在对象字典中不存在。
0x0604 0041	物件不能映像到 PDO 中。
0x0604 0042	要映像的对象数量和长度超过了 PDO 数据长度。
0x0604 0043	常规的参数不相容。
0x0604 0047	设备中通用的内部数值不兼容。
0x0606 0000	由于硬件错误导致操作失败。
0x0607 0010	数据类型不匹配，服务参数长度不匹配。
0x0607 0012	数据类型不匹配，服务参数长度过长。
0x0607 0013	数据类型不匹配，服务参数长度过短。
0x0609 0011	子索引不存在。
0x0609 0030	写入操作时，写入数据值超出范围。
0x0609 0031	写入数据值太大。
0x0609 0032	写入数据值太小。
0x0609 0036	最大值小于最小值。
0x0800 0000	控制器端的一般错误。
0x0800 0020	数据不可以被传输或储存到应用程序。
0x0800 0021	由于本地控制原因，数据不可以被传输或储存到应用程序。
0x0800 0022	由于当前设备状态原因，数据不可以被传输或储存到应用程序。
0x0800 0023	对象字典动态生成错误，或没有找到对象字典。

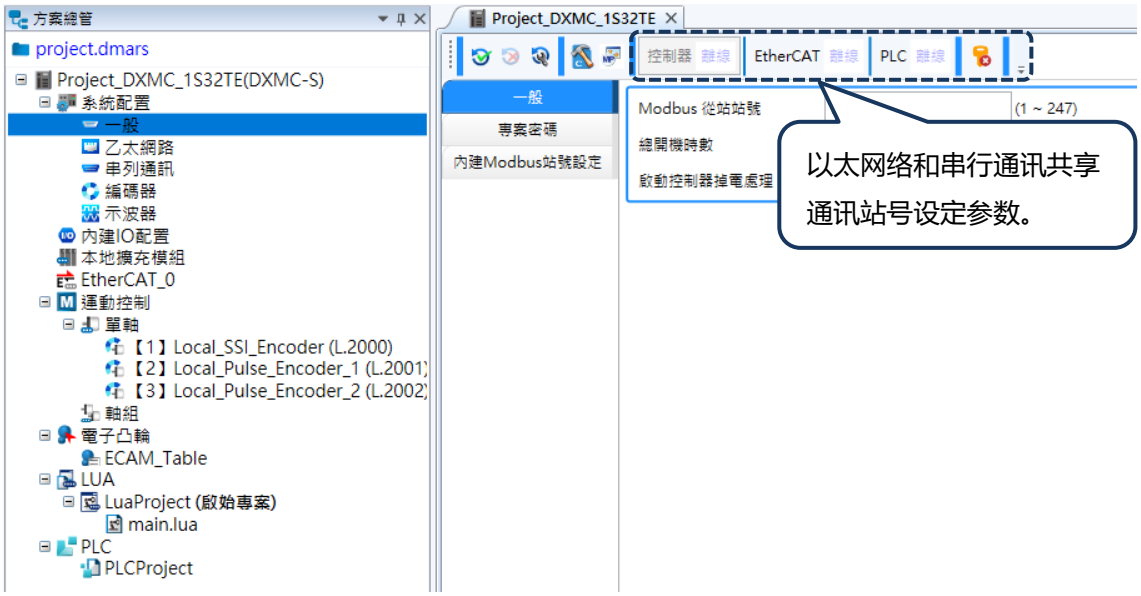
MODBUS 地址表　　附录 A

A.1　MODBUS 站号说明……A-2
A.2　分区说明 ……A-2
A.3　DV、SDV 区说明 ……A-3
A.4　DH、SDH 区说明……A-7
A.5　MODBUS 设定范例……A-8

A.1 MODBUS 站号说明

A

无论是使用以太网络或是串行通讯的 MODBUS，一台 DXMC 控制器占据一个站号。站号的默认值为 1。使用者可以在 DMARS 项目树的**一般**中进行站号设定。



A.2 分区说明

DXMC 的内存分为四个区域：DV 区、SDV 区、DH 区和 SDH 区。除了在 MULTIPROG 使用 PLC 地址来存取之外，这些区域同时也映像到 MODBUS 地址，区域对应分布如下表。由于 PLC MB3 的寻址模式为 Byte，而 MODBUS 寻址模式为 Word，因此 PLC MB3 地址是 MODBUS 地址的两倍。

分区名称	大小 (Byte)	属性	功能说明	PLC MB3 区域 (十进制)(Byte)	MODBUS 区域 (十六进制)(Word)
DV	24K	-	未使用区域	0 ~ 24575	0x0000 ~ 0x2FFF
SDV	40K	-	系统保留区域	3000000 ~ 3029951	0x0300 ~ 0x6A7F
			联机机制、程序流程管理	3029952 ~ 3030463	0x6A80 ~ 0x6B7F
			寸动功能、增益调整功能	3030464 ~ 3030591	0x6B80 ~ 0x6BBF
			系统保留区域	3030592 ~ 3040959	0x6BC0 ~ 0x7FFF
DH	24K	断电保持	未使用区域	1000000 ~ 1024575	0x8000 ~ 0xAFFF
SDH	40K	断电保持	系统保留区域	6000000 ~ 6040959	0xB000 ~ 0xFFFF

A.3 DV、SDV 区说明

■ DV 区

DV 区为非断电保持区，DXMC 提供的 MODBUS 地址 0x0000 至 0x2FFF 为未使用区域，用户可以自行定义功能。

群组功能	PLC MB3 地址 (十进制)	MODBUS 地址 (十进制)	MODBUS 地址 (十六进制)	定义
未定义	0	0	0x0000	未使用区域
	…			
	24575	24575	0x2FFF	未使用区域

■ SDV 区

SDV 区为特殊功能之非断电保持区，DXMC 保留的 MODBUS 特殊功能地址，内容包含联机机制、程序流程管理、寸动等功能。

控制器联机机制功能如下表所示。

群组功能	PLC MB3 地址 (十进制)	MODBUS 地址 (十进制)	MODBUS 地址 (十六进制)	定义
联机机制	3029952	27264	0x6A80	控制器当前运行状态: 0：Initial 1：Fail 2：Edit 4：Run 5：Diagnosis
	3029954	27265	0x6A81	控制器目标运行状态: 2：Edit 4：Run 5：Diagnosis
	3029956	27266	0x6A82	保留
	3029958	27267	0x6A83	保留
	3029960	27268	0x6A84	保留
	3029962	27269	0x6A85	保留
	3029966	27271	0x6A87	保留
	3029970	27273	0x6A89	保留
	3029974	27275	0x6A8B	保留
	3029978	27277	0x6A8D	保留

A

群组功能	PLC MB3 地址 (十进制)	MODBUS 地址 (十进制)	MODBUS 地址 (十六进制)	定义
	3029982	27279	0x6A8F	保留
	3029986	27281	0x6A91	保留
	3029990	27283	0x6A93	保留
	3029994	27285	0x6A95	保留
	3029998	27287	0x6A97	保留
	3030002	27289	0x6A99	保留
	3030006	27291	0x6A9B	保留
	3030010	27293	0x6A9D	保留
	3030014	27295	0x6A9F	保留
	3030018	27297	0x6AA1	保留
	3030022	27299	0x6AA3	保留
	3030026	27301	0x6AA5	保留
	3030030	27303	0x6AA7	保留
	3030034	27305	0x6AA9	保留
	3030038	27307	0x6AAB	保留
	3030042	27309	0x6AAD	保留

控制器联机机制操作范例：

1. 设定控制器切换至运行状态：对 0x6A81 写入 0x0004。
2. 查看控制器当前运行状态：读取 0x6A80。如果控制器成功切换至运行模式，则 0x6A80 数据内容为 4；如果控制器切换失败，则 0x6A80 资料内容为 1。

程序流程管理与 PLC 控制功能如下表所示。

群组功能	PLC MB3 地址 (十进制)	MODBUS 地址 (十进制)	MODBUS 地址 (十六进制)	定义
程序流程	3030208	27392	0x6B00	目前 DRL ID 显示 (0 ~ 9999)
	3030210	27393	0x6B01	目前 DRL 执行状态显示: 0：Close 1：Running 2：Break Point 3：Pause 4：Pre-Run
	3030212	27394	0x6B02	目前 DRL 执行行号显示 (main.lua only)

群组功能	PLC MB3 地址 (十进制)	MODBUS 地址 (十进制)	MODBUS 地址 (十六进制)	定义
	3030214	27395	0x6B03	DRL 运行速度 (%)
	3030216	27396	0x6B04	DRL ID 选择设定 (0 ~ 9999)
	3030218	27397	0x6B05	■ DRL 控制 1：Open 2：Release Run 3：Stop 4：Pause 6：Open & Release Run ■ Debug 11：Step Into 12：Step Over 13：Step Out 21：Block Run 22：Open & Block Run 44：Debug Run 45：Open & Debug Run ■ 专案汇入/汇出 31：Export Project (FTP -> USB) 32：Import Project (USB -> FTP) ■ 断点 41：Set BP 42：Add BP 43：Delete BP 注：在程序执行之前，利用 Set BP 最多可设定八组断点，执行中只能透过 Add BP 每次添加一个断点。
	3030220	27398	0x6B06	■ DRL 状态显示 0：BUSY 1：CHECKING 2：FAILED 10：CANNOT_OPEN 11：OPENING 12：OPENED 20：CANNOT_RUN 21：RUNNING 30：CANNOT_STOP 31：STOPPING 32：STOPPED

A

A

群组功能	PLC MB3 地址 (十进制)	MODBUS 地址 (十进制)	MODBUS 地址 (十六进制)	定义
				40：CANNOT_PAUSE 41：PAUSED 50：CANNOT_STEP 51：STEPPED ■ 项目汇入/导出讯息 60：PASSWORD_ERROR 61：FTP_ID_ERROR 62：USB_MOUNT_ERROR 63：EXPORTING 64：USB_UNMOUNT_ERROR 65：EXPORT_DONE 66：USB_ID_CHECKING 67：USB_ID_ERROR 68：IMPORTING 69：IMPORT_DONE ■ 断点讯息 70：CANNOT_SETBP 71：BP_SETTING 72：BP_SET 73：CANNOT_ADDBP 74：BP_ADDED 75：CANNOT_DELETEBP 76：BP_DELETED
	3030222	27399	0x6B07	保留
	3030224	27400	0x6B08	保留
	3030226	27401	0x6B09	保留
	3030228	27402	0x6B0A	保留
	3030230	27403	0x6B0B	保留
	3030232	27404	0x6B0C	保留
	3030234	27405	0x6B0D	保留
	3030236	27406	0x6B0E	保留
PLC 控制	3030240	27408	0x6B10	PLC 执行控制设定: 0：ColdStart 1：WarmStart 2：HotStart 3：Stop

群组功能	PLC MB3 地址 (十进制)	MODBUS 地址 (十进制)	MODBUS 地址 (十六进制)	定义
	3030242	27409	0x6B11	PLC 执行控制状态显示: 0：PlcOn 1：PlcLoading 2：PlcStarting 3：PlcRunning 4：PlcHaltRequested 5：PlcHalt 6：PlcStopping 7：PlcStop 8：PlcResetting

A.4 DH、SDH 区说明

■ DH 区

DH 区为断电保持区，0x8000 至 0xAFFF 为未使用区域。

群组功能	PLC MB3 地址 (十进制)	MODBUS 地址 (十进制)	MODBUS 地址 (十六进制)	定义
未定义	1000000	32768	0x8000	未使用区域
	…			
	1024575	45055	0xAFFF	未使用区域

■ SDH 区

SDH 区为系统保留区，内容皆为系统重要参数，请不要使用 0xB000 至 0xFFFF 的地址。以下为已定义的功能。

群组功能	PLC MB3 地址 (十进制)	MODBUS 地址 (十进制)	MODBUS 地址 (十六进制)	定义
系统参数	6000000	720896	0xB000	保留
	…			
	6040959	65535	0xFFFF	保留

A.5 MODBUS 设定范例

A

用户可以自行定义 DV 区和 DH 区的地址功能。在 MULTIPROG 中，PLC MB3 地址设定之后，就可以在对应的 MODBUS 地址存取到同一个内存空间。

在 MULTIPROG 中，PLC MB3 地址宣告格式与格式内容说明如下所示：

$$\underset{(1)}{\%M}\ \underset{(2)}{?}\ \underset{(3)}{3.}\ \underset{(4)}{addr}\ \underset{(5)}{.Y}$$

编号	说明
(1)	%M 为固定格式。
(2)	?表示该变量的数据长度。 X：BOOL 格式 B：BYTE 格式（8 bits) W：WORD 格式（16 bits) D：DWORD 格式（32 bits) L：LREAL 格式（64 bits) 注：关于 PLC 数据格式的详细说明，请参考章节 6.2.6 Data Type (数据型态)。
(3)	3.为固定格式。
(4)	addr 为 PLC MB3 地址（十进制)。
(5)	.Y 代表 PLC MB3 地址中的第几个位；Y 的范围为 0 ~ 7。 注：如果数据长度为 BOOL，则必需指定 PLC MB3 的地址中的第几个位。

■ MODBUS TCP/IP 范例 1

PLC MB3 区域地址设定为%MW3.160，其中 W 表示该数据长度为 WORD 格式（16 bits)，且 addr 为十进制的 $160_{(10)}$。

由于 PLC MB3 地址为 MODBUS 地址的两倍，所以如果要将该 PLC MB3 地址转换成 MODBUS 地址，则需要将 $160_{(10)}$ 除以 2，也就是 $80_{(10)}$，再将其转换成 16 进制则为 $50_{(16)}$。

PLC MB3 地址：

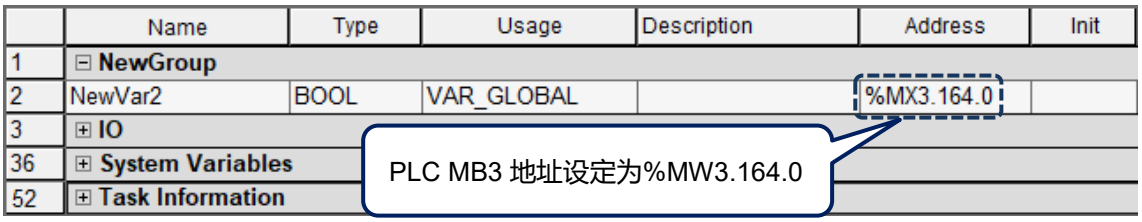
	Name	Type	Usage	Description	Address	Init
1	⊟ NewGroup					
2	NewVar1	INT	VAR_GLOBAL		%MW3.160	
3	⊞ IO					
36	⊞ System Variables					
52	⊞ Task Information					

PLC MB3 地址设定为%MW3.160

MODBUS 地址:

■ MODBUS TCP/IP 范例 2

若变量名称 NewVar2 为 BOOL 型态，将 PLC MB3 区域地址设定为%MX3.164.0，则 MODBUS 16 进制地址为 $52.0_{(16)}$。



(此页有意留为空白)

A

更新履历

发行日期	版本	更新章节	更新内容
June, 2020	V1.0 (第一版)		

关于[DXMC 使用手册]其它相关信息，可参考：

(1) DMARS 软件操作手册