

# Personal Computer Embedded Type Servo System Controller

## Simple Motion Board User's Manual (API Library)

---

-MR-EM340GF



# SAFETY PRECAUTIONS



---

(Read these precautions before using this product.)

Before using this product, please read this manual and the relevant manuals carefully and pay full attention to safety to handle the product correctly.

The precautions given in this manual are concerned with this product only.

In this manual, the safety precautions are classified into two levels: "⚠️ WARNING" and "⚠️ CAUTION".

 <b>WARNING</b>	Indicates that incorrect handling may cause hazardous conditions, resulting in death or severe injury.
 <b>CAUTION</b>	Indicates that incorrect handling may cause hazardous conditions, resulting in minor or moderate injury or property damage.

Under some circumstances, failure to observe the precautions given under "⚠️ CAUTION" may lead to serious consequences.

Observe the precautions of both levels because they are important for personal and system safety.

Make sure that the end users read this manual and then keep the manual in a safe place for future reference.

## [Design Precautions]

---

### **WARNING**

- Configure safety circuits externally to ensure that the entire system operates safely even when a fault occurs in the personal computer. Failure to do so may result in an accident due to an incorrect output or malfunction.
    - (1) Configure external safety circuits, such as an emergency stop circuit, protection circuit, and protective interlock circuit for forward/reverse operation or upper/lower limit positioning.
    - (2) If an incorrect home position return direction is set, motion control may continue without deceleration. To prevent machine damage caused by this, configure an interlock circuit external to the Simple Motion board.
    - (3) When the Simple Motion board detects an error, the motion slows down and stops or the motion rapidly stops, depending on the stop group setting in parameter. Set the parameter to meet the specifications of a positioning control system. In addition, set the home position return parameter and positioning data within the specified setting range.
  - For the operating status of each station after a communication failure, refer to manuals relevant to the network. Incorrect output or malfunction due to a communication failure may result in an accident.
  - When modifying data of a running Simple Motion board, configure an interlock in the program to ensure that the entire system will always operate safely. For other forms of control (such as program modification, parameter change, forced output, or operating status change) of a running Simple Motion board, read the relevant manuals carefully and ensure that the operation is safe before proceeding. Improper operation may damage machines or cause accidents. Determine corrective actions to be taken in case of a communication failure.
  - Especially, when a remote Simple Motion board is controlled, immediate action cannot be taken if a problem occurs in the Simple Motion board due to a communication failure. To prevent this, configure an interlock in the program, and determine corrective actions to be taken in case of a communication failure.
  - Do not write any data to the "system area" and "write-protect area" of the buffer memory in the Simple Motion board. Doing so may cause malfunction of the Simple Motion board. For the "system area", and "write-protect area", refer to the user's manual for the Simple Motion board.
-

## [Design Precautions]

---

### **WARNING**

- If a communication cable is disconnected, the network may be unstable, resulting in a communication failure of multiple stations. Configure an interlock in the program to ensure that the entire system will always operate safely even if communications fail. Failure to do so may result in an accident due to an incorrect output or malfunction.
  - To maintain the safety of the Simple Motion board against unauthorized access from external devices via the network, take appropriate measures. To maintain the safety against unauthorized access via the Internet, take measures such as installing a firewall.
  - If safety standards (ex., robot safety rules, etc.,) apply to the system using the Simple Motion board, servo amplifier and servomotor, make sure that the safety standards are satisfied.
  - Construct a safety circuit externally of the Simple Motion board or servo amplifier if the abnormal operation of the Simple Motion board or servo amplifier differs from the safety directive operation in the system.
- 

## [Design Precautions]

---

### **CAUTION**

- Do not install the control lines or communication cables together with the main circuit lines or power cables. Keep a distance of 100 mm or more between them. Failure to do so may result in malfunction due to noise.
  - After the personal computer is powered on or rebooted, the time taken for the Simple Motion board to enter the RUN status varies depending on the system configuration, parameter settings, and/or program size. Design circuits so that the entire system will always operate safely, regardless of the time.
  - Do not power off or reboot the personal computer during the setting registration. Doing so will make the data in the flash ROM undefined. The data need to be set in the buffer memory and to be written to the flash ROM again. Doing so may cause malfunction or failure of the Simple Motion board.
- 

## [Installation Precautions]

---

### **WARNING**

- Shut off the external power supply (all phases) used in the system before mounting or removing the Simple Motion board to or from the personal computer. Failure to do so may result in electric shock or cause the Simple Motion board to fail or malfunction.
  - Do not touch any connectors while power is on. Doing so may cause electric shock or malfunction.
-

## [Installation Precautions]

---

### **CAUTION**

- Use the Simple Motion board in an environment that meets the general specifications in the Simple Motion Board User's Manual. Failure to do so may result in electric shock, fire, malfunction, or damage to or deterioration of the product.
  - Fix the Simple Motion board securely with the board-fixing screw. Tighten the screws within the specified torque range. Undertightening can cause drop of the screw, short circuit, or malfunction. Overtightening can damage the screw and/or Simple Motion board, resulting in drop, short circuit, or malfunction. For the tightening torque of the board-fixing screws, refer to the manual supplied with the personal computer.
  - Do not directly touch any conductive parts and electronic components of the Simple Motion board. Hold the front panel or edge of the print board. Not holding by the front panel or edges may cause malfunction or failure of the Simple Motion board.
  - Do not disassemble or modify the Simple Motion board. Doing so may cause failure, malfunction, injury, or a fire.
  - Before handling the Simple Motion board, touch a conducting object such as a grounded metal to discharge the static electricity from the human body. Failure to do so may cause the Simple Motion board to fail or malfunction.
  - Install the Simple Motion board to a personal computer which is compliant with PCI Express® standard. Failure to do so may cause a failure or malfunction.
  - Securely insert the Simple Motion board into the slot following the board installation instruction of the personal computer. Incorrect insertion of the Simple Motion board may cause malfunction, failure, or drop of the board.
  - When installing the Simple Motion board, take care not to contact with other boards.
  - When installing the Simple Motion board, take care not to get injured by an implemented component or a surrounding member.
  - Handle the Simple Motion board in a place where static electricity will not be generated. Failure to do so may cause a failure or malfunction.
  - The Simple Motion board is included in an antistatic envelope. When storing or transporting it, be sure to put it in the antistatic envelope. Failure to do so may cause a failure or malfunction.
  - Do not drop or apply a strong impact to the Simple Motion board. Doing so may cause a failure or malfunction.
- 

## [Wiring Precautions]

---

### **WARNING**

- Shut off the external power supply (all phases) used in the system before installation and wiring. Failure to do so may result in electric shock or damage to the Simple Motion board.
  - After installation and wiring, attach the cover of the equipment the Simple Motion board is installed to before turning it on for operation. Failure to do so may result in electric shock.
-

## [Wiring Precautions]

---

### **CAUTION**

- Ground the controllers, servo amplifiers and servo motors embedded with a Simple Motion board with a ground resistance of 100 ohm or less. Do not use a common grounding with other equipment.
  - Check the rated voltage and signal layout before wiring to the Simple Motion board, and connect the cables correctly. Connecting a power supply with a different voltage rating or incorrect wiring may cause fire or failure.
  - Connectors must be correctly connected. Incomplete connections may cause short circuit, fire, or malfunction.
  - Securely connect the connector to the Simple Motion board. Poor contact may cause malfunction.
  - Do not install the control lines or communication cables together with the main circuit lines or power cables. Keep a distance of 100 mm or more between them. Failure to do so may result in malfunction due to noise.
  - Place the cables in a duct or clamp them. If not, dangling cable may swing or inadvertently be pulled, resulting in damage to the Simple Motion board or cables or malfunction due to poor contact. Do not clamp the extension cables with the jacket stripped. Doing so may change the characteristics of the cables, resulting in malfunction.
  - Check the interface type and correctly connect the cable. Incorrect wiring (connecting the cable to an incorrect interface) may cause failure of the Simple Motion board and external device.
  - When disconnecting the cable from the Simple Motion board, do not pull the cable by the cable part. For the cable with connector, hold the connector part of the cable. Pulling the cable connected to the Simple Motion board may result in malfunction or damage to the Simple Motion board or cable.
  - Prevent foreign matter such as dust or wire chips from entering the personal computer. Such foreign matter can cause a fire, failure, or malfunction.
  - For Ethernet cables to be used in the system, select the ones that meet the specifications in the user's manual of the Simple Motion board. If not, normal data transmission is not guaranteed.
- 

## [Startup and Maintenance Precautions]

---

### **WARNING**

- Shut off the external power supply (all phases) used in the system before cleaning or retightening the board-fixing screws. Failure to do so may result in electric shock or malfunction.
  - Turn off the external power supply for the system in all phases before installing the Simple Motion board to or removing it from the personal computer. Failure to do so may result in electric shock or cause the Simple Motion board to fail or malfunction.
  - Do not connect or disconnect any communication cable while power is on. Doing so may result in a malfunction.
-

## [Startup and Maintenance Precautions]

---

### **CAUTION**

---

- When modifying data of a running Simple Motion board, configure an interlock in the program to ensure that the entire system will always operate safely. For other forms of control (such as program modification, parameter change, forced output, or operating status change) of a running Simple Motion board, read the relevant manuals carefully and ensure that the operation is safe before proceeding. Improper operation may damage machines or cause accidents. Determine corrective actions to be taken in case of a communication failure.
  - Especially, when a remote Simple Motion board is controlled, immediate action cannot be taken if a problem occurs in the Simple Motion board due to a communication failure. To prevent this, configure an interlock in the program, and determine corrective actions to be taken in case of a communication failure.
  - Do not disassemble or modify the Simple Motion board. Doing so may cause failure, malfunction, injury, or a fire.
  - Use any radio communication device such as a cellular phone or PHS (Personal Handyphone System) more than 25 cm away in all directions from the Simple Motion board. Failure to do so may cause malfunction.
  - Shut off the external power supply (all phases) used in the system before mounting or removing the Simple Motion board. Failure to do so may cause the Simple Motion board to fail or malfunction.
  - Tighten the board-fixing screws within the specified torque range. Undertightening can cause drop of the component or wire, short circuit, or malfunction. Overtightening can damage the screw and/or Simple Motion board, resulting in drop, short circuit, or malfunction. For the tightening torque of the board-fixing screws, refer to the manual supplied with the personal computer.
  - After the first use of the product, do not mount/remove the Simple Motion board to/from the personal computer more than 50 times. Exceeding the limit of 50 times may cause malfunction.
  - Maintenance must be performed by qualified maintenance personnel with knowledge.
  - Before handling the Simple Motion board, touch a conducting object such as a grounded metal to discharge the static electricity from the human body. Failure to do so may cause the Simple Motion board to fail or malfunction.
  - The Simple Motion board is included in an antistatic envelope. When storing or transporting it, be sure to put it in the antistatic envelope. Failure to do so may cause a failure or malfunction.
  - The microprocessor built in the Simple Motion board will reach a high temperature during operation. Do not touch it directly when replacing the Simple Motion board. Doing so may result in a burn.
  - Before testing the operation, set a low speed value for the speed limit parameter so that the operation can be stopped immediately upon occurrence of a hazardous condition.
  - Confirm and adjust the program and each parameter before operation. Unpredictable movements may occur depending on the machine.
  - When using the absolute position system function, on starting up, and when the Simple Motion board or absolute position motor has been replaced, always perform a home position return.
  - Before starting the operation, confirm the brake function.
  - Do not perform a megger test (insulation resistance measurement) during inspection.
  - After maintenance and inspections are completed, confirm that the position detection of the absolute position detection function is correct.
  - Extreme adjustments and changes may lead to unstable operation, so never make them.
-

## [Startup and Maintenance Precautions]

---

### CAUTION

- Do not place the Simple Motion board or servo amplifier on metal that may cause a power leakage or wood, plastic or vinyl that may cause static electricity buildup. Doing so can cause malfunction or failure of the Simple Motion board.
- 

## [Operating Precautions]

---

### CAUTION

- When changing data and operating status, and modifying program of the running Simple Motion board, read relevant manuals carefully and ensure the safety before operation. Incorrect change or modification may cause system malfunction, damage to the machines, or accidents.
  - Do not power off or reboot the personal computer while the setting values in the buffer memory are being written to the flash ROM in the Simple Motion board. Doing so will make the data in the flash ROM undefined. The values need to be set in the buffer memory and written to the flash ROM again. Doing so also can cause malfunction or failure of the Simple Motion board.
  - Note that when the reference axis speed is specified for interpolation operation, the speed of the partner axis (2nd, 3rd, or 4th axis) may exceed the speed limit value.
  - Do not go near the machine during test operations or during operations such as teaching. Doing so may lead to injuries.
- 

## [Disposal Precautions]

---

### CAUTION

- When disposing of this product, treat it as industrial waste.
- 

## [Transportation Precautions]

---

### CAUTION

- The halogens (such as fluorine, chlorine, bromine, and iodine), which are contained in a fumigant used for disinfection and pest control of wood packaging materials, may cause failure of the product. Prevent the entry of fumigant residues into the product or consider other methods (such as heat treatment) instead of fumigation. The disinfection and pest control measures must be applied to unprocessed raw wood.
  - The Simple Motion board is a precision machine, so do not drop or apply strong impacts on it.
-



# CONDITIONS OF USE FOR THE PRODUCT

---

(1) Mitsubishi Simple Motion board ("the PRODUCT") shall be used in conditions;

- i) where any problem, fault or failure occurring in the PRODUCT, if any, shall not lead to any major or serious accident; and
- ii) where the backup and fail-safe function are systematically or automatically provided outside of the PRODUCT for the case of any problem, fault or failure occurring in the PRODUCT.

(2) The PRODUCT has been designed and manufactured for the purpose of being used in general industries.

MITSUBISHI SHALL HAVE NO RESPONSIBILITY OR LIABILITY (INCLUDING, BUT NOT LIMITED TO ANY AND ALL RESPONSIBILITY OR LIABILITY BASED ON CONTRACT, WARRANTY, TORT, PRODUCT LIABILITY) FOR ANY INJURY OR DEATH TO PERSONS OR LOSS OR DAMAGE TO PROPERTY CAUSED BY the PRODUCT THAT ARE OPERATED OR USED IN APPLICATION NOT INTENDED OR EXCLUDED BY INSTRUCTIONS, PRECAUTIONS, OR WARNING CONTAINED IN MITSUBISHI'S USER, INSTRUCTION AND/OR SAFETY MANUALS, TECHNICAL BULLETINS AND GUIDELINES FOR the PRODUCT.

("Prohibited Application")

Prohibited Applications include, but not limited to, the use of the PRODUCT in;

- Nuclear Power Plants and any other power plants operated by Power companies, and/or any other cases in which the public could be affected if any problem or fault occurs in the PRODUCT.
- Railway companies or Public service purposes, and/or any other cases in which establishment of a special quality assurance system is required by the Purchaser or End User.
- Aircraft or Aerospace, Medical applications, Train equipment, transport equipment such as Elevator and Escalator, Incineration and Fuel devices, Vehicles, Manned transportation, Equipment for Recreation and Amusement, and Safety devices, handling of Nuclear or Hazardous Materials or Chemicals, Mining and Drilling, and/or other applications where there is a significant risk of injury to the public or property.

Notwithstanding the above, restrictions Mitsubishi may in its sole discretion, authorize use of the PRODUCT in one or more of the Prohibited Applications, provided that the usage of the PRODUCT is limited only for the specific applications agreed to by Mitsubishi and provided further that no special quality assurance or fail-safe, redundant or other safety features which exceed the general specifications of the PRODUCTS are required. For details, please contact the Mitsubishi representative in your region.

## INTRODUCTION

---

Thank you for purchasing the personal computer embedded type servo system controllers.

This manual describes the API library required for the programming of the relevant products listed below.

Before using this product, please read this manual and the relevant manuals carefully and develop familiarity with the functions and performance of the personal computer embedded type servo system controller to handle the product correctly. When applying the program examples provided in this manual to an actual system, ensure the applicability and confirm that it will not cause system control problems.

Please make sure that the end users read this manual.

### Relevant products

---

MR-EM340GF

#### Point

Symbols used in this manual are shown below.

A serial No. is inserted in the "\*\*\*\*" mark.

- [Pr.\*\*]: Symbols indicating positioning parameter or home position return parameter items
  - [Da.\*\*]: Symbols indicating positioning data or block start data items
  - [Md.\*\*]: Symbols indicating monitor data items
  - [Cd.\*\*]: Symbols indicating control data items
-

# CONTENTS

SAFETY PRECAUTIONS .....	1
CONDITIONS OF USE FOR THE PRODUCT .....	7
INTRODUCTION .....	7
RELEVANT MANUALS .....	13
TERMS .....	13
<b>CHAPTER 1 SPECIFICATIONS</b> .....	<b>15</b>
<b>1.1 Operating Environment</b> .....	<b>15</b>
<b>1.2 Precautions</b> .....	<b>16</b>
<b>1.3 File Configuration</b> .....	<b>17</b>
<b>CHAPTER 2 CLASS/FUNCTION LIST</b> .....	<b>18</b>
<b>2.1 Class List</b> .....	<b>18</b>
<b>2.2 Public method/Data Method List</b> .....	<b>19</b>
MMC_NetworkModule class .....	19
MMC_Controller class .....	20
MMC_Axis class .....	22
MMC_lo class .....	23
MMC_SyncEncoder class .....	24
MC_FunctionBlock class .....	24
MMC_Label class .....	26
MMC_DeviceDriver class .....	26
<b>2.3 Function List</b> .....	<b>27</b>
Object generation functions .....	27
Basic library functions .....	27
<b>CHAPTER 3 API LIBRARY DETAILS (BASIC FUNCTIONS)</b> .....	<b>28</b>
<b>3.1 MMC_NetworkModule Class</b> .....	<b>30</b>
MMC_NetworkModule::Delete .....	31
<b>3.2 MMC_Controller Class</b> .....	<b>32</b>
MMC_Controller::GetAxis (C++) .....	34
MMC_Controller::GetAxis (C#) .....	35
MMC_Controller::GetSlavelo (C++) .....	36
MMC_Controller::GetSlavelo (C#) .....	37
MMC_Controller::ResetController .....	38
MMC_Controller::SetUserProgramReady .....	39
MMC_Controller::InitializeParameter .....	40
MMC_Controller::BackupParameter .....	41
MMC_Controller::SetInterruptParameter .....	42
MMC_Controller::EnableInterrupt .....	44
MMC_Controller::DisableInterrupt .....	45
MMC_Controller::RegisterIntCallback (C++) .....	46
MMC_Controller::RegisterIntCallback (C#) .....	47
MMC_Controller::UnregisterIntCallback .....	49
MMC_Controller::ResetIntEvent .....	50
MMC_Controller::SetIntEvent .....	51
MMC_Controller::WaitIntEvent .....	52
MMC_Controller::EnabledDMA .....	53

	MMC_Controller::DisableDMA	54
	MMC_Controller::ReadRemoteBufferMemory	55
	MMC_Controller::WriteRemoteBufferMemory	56
<b>3.3</b>	<b>MMC_Axis Class</b>	<b>57</b>
	MMC_Axis::SetPositioningData	59
	MMC_Axis::SetBlockStartData	60
	MMC_Axis::SetBlockConditionData	61
	MMC_Axis::GetPositioningData	62
	MMC_Axis::GetBlockStartData	63
	MMC_Axis::GetBlockConditionData	64
	MMC_Axis::StartPositioning	65
	MMC_Axis::StartBlockPositioning	66
	MMC_Axis::StopPositioning	67
	MMC_Axis::RestartPositioning	68
	MMC_Axis::WaitPositioningDone	69
	MMC_Axis::ResetPositioningDoneIntEvent	70
	MMC_Axis::SetPositioningDoneIntEvent	71
	MMC_Axis::WaitPositioningDoneIntEvent	72
	MMC_Axis::StartJog	73
	MMC_Axis::StopJog	74
	MMC_Axis::StartInching	75
	MMC_Axis::EnableMPG	76
	MMC_Axis::DisableMPG	77
	MMC_Axis::ChangeControlMode	78
	MMC_Axis::ChangeSpeed	79
	MMC_Axis::ChangePosition	80
	MMC_Axis::ResetError	81
	MMC_Axis::SetInterruptParameter	82
	MMC_Axis::ResetIntEvent	84
	MMC_Axis::SetIntEvent	85
	MMC_Axis::WaitIntEvent	86
<b>3.4</b>	<b>MMC_Io Class</b>	<b>87</b>
	Link device	88
<b>3.5</b>	<b>MMC_Label Class</b>	<b>89</b>
	Wait	90
<b>3.6</b>	<b>MMC_DeviceDriver Class</b>	<b>92</b>
	MMC_DeviceDriver::Delete	93
	MMC_DeviceDriver::Open	94
	MMC_DeviceDriver::Close	95
	MMC_DeviceDriver::StartInterrupt	96
	MMC_DeviceDriver::EndInterrupt	97
	MMC_DeviceDriver::StartDMA	98
	MMC_DeviceDriver::EndDMA	99
	MMC_DeviceDriver::SetBufferMemory	100
	MMC_DeviceDriver::GetBufferMemory	101
<b>3.7</b>	<b>Object Generation Functions</b>	<b>102</b>
	MmfCreateEM340GF (C++)	102
	MmfCreateEM340GF (C#)	103
	MmfCreatePciDevice (C++)	104
	MmfCreatePciDevice (C#)	105
<b>3.8</b>	<b>Basic Library Functions</b>	<b>106</b>

MmfGetLastError . . . . .	106
MmfCreateSemaphore . . . . .	107
MmfDeleteSemaphore . . . . .	108
MmfWaitSemaphore . . . . .	109
MmfReleaseSemaphore . . . . .	110
MmfCreateEvent . . . . .	111
MmfDeleteEvent . . . . .	112
MmfResetEvent . . . . .	113
MmfSetEvent . . . . .	114
MmfWaitEvent . . . . .	115
MmfCreateThread . . . . .	116
MmfDeleteThread . . . . .	117
MmfWaitThread . . . . .	118
MmfSetThreadPriority . . . . .	119
MmfResumeThread . . . . .	120
MmfGetExitCodeThread . . . . .	121
MmfExitThread . . . . .	122
MmfGetCurrentTime . . . . .	123
MmfCheckPassTime . . . . .	124
MmfWaitDelayTime . . . . .	125

---

## **CHAPTER 4 API LIBRARY DETAILS (ADVANCED SYNCHRONOUS CONTROL) 126**

<b>4.1 MMC_Controller Class . . . . .</b>	<b>127</b>
MMC_Controller::GetSyncEncoder . . . . .	128
MMC_Controller::CalcCamCommandPosition . . . . .	129
MMC_Controller::CalcCamCommandPositionPerCycle . . . . .	130
MMC_Controller::MakeRotaryCutterCam . . . . .	131
MMC_Controller::MakeEasyStrokeRatioCam . . . . .	132
MMC_Controller::MakeAdvancedStrokeRatioCam . . . . .	133
<b>4.2 MMC_Axis Class . . . . .</b>	<b>134</b>
MMC_Axis::StartSync . . . . .	135
MMC_Axis::StopSync . . . . .	136
MMC_Axis::ChangeSyncPosition . . . . .	137
MMC_Axis::MoveCamPosition . . . . .	138
<b>4.3 MMC_SyncEncoder Class . . . . .</b>	<b>139</b>
MMC_SyncEncoder::ResetSyncEncoderError . . . . .	140
MMC_SyncEncoder::ChangeSyncEncoderPosition . . . . .	141
MMC_SyncEncoder::DisableSyncEncoder . . . . .	142
MMC_SyncEncoder::EnableSyncEncoder . . . . .	143

---

## **CHAPTER 5 API LIBRARY DETAILS (FUNCTION BLOCK) 144**

<b>5.1 MC_FunctionBlock Class . . . . .</b>	<b>144</b>
Axis information class state diagram . . . . .	147
Function block units . . . . .	149
Function blocks for PLCopen Motion control . . . . .	150
MC_FunctionBlock::Update . . . . .	152
MC_Power class . . . . .	153
MCv_Home class . . . . .	155
MC_Stop class . . . . .	157
MC_MoveAbsolute class . . . . .	159

MC_MoveRelative class	162
MC_Reset class	165
MC_MoveAdditive class	167
MC_MoveVelocity class	169
MC_TorqueControl class	171
MC_SetPosition class	173
MC_SetOverride class	175
MC_ReadActualPosition class	177
MC_ReadStatus class	179
MC_ReadAxisInfo class	181
MC_ReadAxisError class	183
MCv_ReadServoParameter class	185
MCv_WriteServoParameter class	187
<b>5.2 MMC_Controller Class</b>	<b>189</b>
MMC_Controller::GetFbAxisRef	190
MMC_Controller::GetFbAxisReflo	191
<b>5.3 AXIS_REF Class</b>	<b>192</b>
<b>5.4 AXIS_REF_MOTION Class</b>	<b>193</b>
<b>5.5 AXIS_REF_IO Class</b>	<b>194</b>
<b>5.6 Object Generation Functions</b>	<b>195</b>
MmfCreateFunctionBlock	195
<b>CHAPTER 6 STRUCTURE LIST</b>	<b>196</b>
<b>6.1 MMST_PositioningData Structure</b>	<b>196</b>
<b>6.2 MMST_BlockStartData Structure</b>	<b>196</b>
<b>6.3 MMST_BlockConditionData Structure</b>	<b>196</b>
<b>6.4 MMST_InterruptParameter Structure</b>	<b>197</b>
<b>6.5 MMST_InterruptData Structure</b>	<b>197</b>
<b>6.6 MMST_CamPositionData Structure</b>	<b>197</b>
<b>6.7 MMST_RotaryCutterCamData Structure</b>	<b>197</b>
<b>6.8 MMST_EasyStrokeRatioCamData Structure</b>	<b>198</b>
<b>6.9 MMST_EasyStrokeRatioCamSectionData Structure</b>	<b>198</b>
<b>6.10 MMST_AdvancedStrokeRatioCamData Structure</b>	<b>198</b>
<b>6.11 MMST_AdvancedStrokeRatioCamSectionData Structure</b>	<b>198</b>
<b>CHAPTER 7 ERROR CODE LIST</b>	<b>199</b>
<b>7.1 Common Errors</b>	<b>199</b>
<b>7.2 Device Related Errors</b>	<b>200</b>
<b>7.3 Interrupt Related Errors</b>	<b>200</b>
<b>7.4 DMA Transmission Related Errors</b>	<b>201</b>
<b>7.5 SLMP Communication Function Related Errors</b>	<b>201</b>
<b>7.6 Function Block Related Errors</b>	<b>201</b>
<b>CHAPTER 8 LABEL LIST</b>	<b>203</b>
<b>APPENDICES</b>	<b>238</b>
<b>Appendix 1 Hierarchy Charts for Each Class</b>	<b>238</b>
<b>Appendix 2 Restrictions by the version</b>	<b>241</b>
<b>INDEX</b>	<b>242</b>

REVISIONS .....	244
WARRANTY .....	245
INFORMATION AND SERVICES .....	246
TRADEMARKS .....	246

# RELEVANT MANUALS

Manual name [manual number]	Description	Available form
Simple Motion Board User's Manual (API Library) [IB-0300330] (This manual)	API library and others that the host personal computer uses to control the Simple Motion board.	Print book e-Manual PDF
Simple Motion Board User's Manual (Startup) [IB-0300322]	Specifications, procedures before operation, system configuration, wiring, and operation examples of the Simple Motion board.	Print book e-Manual PDF
Simple Motion Board User's Manual (Application) [IB-0300324]	Functions, input/output signals, buffer memory, parameter settings, programming, and troubleshooting of the Simple Motion board.	Print book e-Manual PDF
Simple Motion Board User's Manual (Advanced Synchronous Control) [IB-0300326]	Functions and programming for the synchronous control of the Simple Motion board.	Print book e-Manual PDF
Simple Motion Board User's Manual (Network) [IB-0300328]	Functions, parameter settings, troubleshooting and buffer memory of CC-Link IE Field Network.	Print book e-Manual PDF



e-Manual refers to the Mitsubishi FA electronic book manuals that can be browsed using a dedicated tool.

e-Manual has the following features:

- Required information can be cross-searched in multiple manuals.
- Other manuals can be accessed from the links in the manual.
- The hardware specifications of each part can be found from the product figures.
- Pages that users often browse can be bookmarked.

## TERMS

Unless otherwise specified, this manual uses the following terms.

Term	Description
API version	Software version of the API library
API library	A general name for the library that creates the application on the host personal computer controlling the Simple Motion board
Axis	Another term for a servo amplifier
Buffer memory	A memory in the Simple Motion board, where data (such as setting values and monitoring values) are stored
CC-Link IE Field Network	A high-speed and large-capacity open field network that is based on Ethernet (1000BASE-T)
Device	A device (X, Y, RX, RY, or others) in the Simple Motion board
DMA transmission	Automatic data transfer between a buffer memory of the MR-EM340GF and a memory in the host personal computer
EM Software Development Kit	A product name for software development kit for Simple Motion board
Host personal computer	A general name for a personal computer which operates user programs
Label	A label that represents one of memory areas (I/O signals and buffer memory areas) specific to the Simple Motion board in a given character string
Link device	A device (RX, RY, RWr, or RWw) in a module on CC-Link IE Field Network
MR-EM340GF	Another term for the Simple Motion board compatible with CC-Link IE Field Network
Operation cycle	A motion operation cycle that is set in the inter-module synchronization cycle setting of the Simple Motion board
Remote device station	A station that exchanges I/O signals (bit data) and I/O data (word data) with another station by cyclic transmission. This station responds to a transient transmission request from another station.
Servo amplifier	A generic term for a drive unit Unless specified in particular, indicates the motor driver unit of the sequential command method which is controlled by the Simple Motion board (belonging to own station).
Simple Motion board	The abbreviation for the personal computer embedded type servo system controller Simple Motion board
SLMP	The abbreviation for Seamless Message Protocol. This protocol is used to access an SLMP-compatible device or a programmable controller connected to an SLMP-compatible device from an external device.
Software version	Software version of Simple Motion board firmware

Term	Description
User program	A general name for applications using the API library



# 1 SPECIFICATIONS

The API library is used to create applications on a host personal computer that controls the Simple Motion board (MR-EM340GF). The API library executes open and closed communication, positioning control etc. with the Simple Motion board.

## 1.1 Operating Environment

The operating environment and conditions for use of the API library are shown below.

- The operating system and compiler compatible with the API library are shown below.

Item	Details
Operating system <sup>*1</sup>	<ul style="list-style-type: none"><li>• Microsoft® Windows® 10 Enterprise (32-bit/64-bit)</li><li>• Microsoft® Windows® 10 Pro (32-bit/64-bit)</li><li>• Microsoft® Windows® 8.1 Enterprise (32-bit/64-bit)</li><li>• Microsoft® Windows® 8.1 Pro (32-bit/64-bit)</li><li>• Microsoft® Windows® 7 Enterprise SP1 (32-bit/64-bit)</li><li>• Microsoft® Windows® 7 Ultimate SP1 (32-bit/64-bit)</li><li>• Microsoft® Windows® 7 Professional SP1 (32-bit/64-bit)</li></ul>
Compiler	<ul style="list-style-type: none"><li>• Microsoft® Visual C++® 2015/2013/2012/2010</li><li>• Microsoft® Visual C#® 2015/2013/2012/2010</li></ul>

\*1 The operating environment of the C# API library is as follows.

For Windows 10, if .NET Framework 3.5 (including .NET 2.0 and 3.0) and .NET Framework 4.6 Advanced Services are invalid, they need to be valid.

For Windows 8.1, if .NET Framework 3.5 (including .NET 2.0 and 3.0) and .NET Framework 4.5 Advanced Services are invalid, they need to be valid.

For Windows 7, if .NET Framework 3.5 (including .NET 2.0 and 3.0) is invalid, it needs to be valid.

- The API library is provided in DLL format. When using the C++ API library, include the header files of the API library from the source files. When using the C# API library, add the following DLL references to Visual C# projects.

Language	32-bit	64-bit
C#	em3xx-std_x86_m.dll	em3xx-std_x64_m.dll

- Store the following DLL files in the same folder as the execution files.

Language	32-bit	64-bit
C++	em3xx-std_x86.dll em3xxdrv-pci_x86.dll	em3xx-std_x64.dll em3xxdrv-pci_x64.dll
C#	em3xx-std_x86_m.dll em3xx-std_x86.dll em3xxdrv-pci_x86.dll	em3xx-std_x64_m.dll em3xx-std_x64.dll em3xxdrv-pci_x64.dll

- When using the C++ API library, the following settings must be made.
  - Create source files with the file extension ".cpp".
  - Modify the project properties by setting "CompileAs" to "Default" or "Compile as C++ Code(/TP)".
- When using the C++ API library, define the following macros in the compiler options.

Operating system	Macro
Windows® 32-bit	"UseForWin32"
Windows® 64-bit	"UseForWin32", "_WIN64"

- Up to four Simple Motion boards can use the API library simultaneously.
- The API library type, and functions are declared by the following namespaces.

Language	Namespace
C++	MitsubishiElectric::EMSDK::Library
C#	MitsubishiElectric::EMSDK::ManagedLibrary

- The I/O mode of the servo amplifier MR-J4-GF supports only the function block class.
- When using the same VisualStudio IntelliSense function as the C++ in the C# development environment, store the following XML file in the same folder as the C# API library DLL file.

Language	32-bit	64-bit
C# API library	em3xx-std_x86_m.dll	em3xx-std_x64_m.dll
XML document for IntelliSense	em3xx-std_x86_m.xml	em3xx-std_x64_m.xml

- The following table shows the differences in the specifications between the C++ API library and the C# API library.

Item	C++	C#
Basic library	Supported.	Not supported by Semaphore function, Event function, Thread function, and Timer function. Use functions of .NET Framework.
Constant	Defined as a macro by using the #define.	Defined as a MMC_ConstLibrary class member.
MmfGetLastError	Defined as a global function.	Defined as a MMC_BasicLibrary class method.
MmfCreatePciDevice MmfCreateEM340GF MmfCreateFunctionBlock	Defined as a global function.	Defined as a MMC_CreateObject class method.
Object generation function	Specifies the pointer to the object pointer as an argument and returns the error code. example: errorCode = MmfCreatePciDevice( boardID, &deviceDriver );	Returns the object. Gets the error code using LastErrorNumber or MmfGetLastError. example: deviceDriver = MMC_CreateObject.MmfCreatePciDevice( boardID );
Label	Assigns and refers to a label. Example: axis->AxPrm.Unit = 3;	Assigns and refers to Value property of labels. Example: axis.AxPrm.Unit.Value = 3;
GetBufferMemory SetBufferMemory	Specifies the pointer to variables for the argument data.	Specifies one of the following types of array for the argument data. sbyte, short, int, byte, ushort, uint

### Point

- The Microsoft Security Advisory 3033929 must be installed to use the API library.
- The API library may not operate normally when anti-virus software is enabled.
- In order to access the PCI Express bus, this API library uses WinDriver produced by Jungo Connectivity.
- When using the C# API library, do not access members of objects after calling Delete method of device driver class, controller class or function block class. The C# API library will not operate normally.

## 1.2 Precautions

The following are precautions for using the API library.

- The disclosure of source code included in the API library to third-parties is prohibited.
- Source code in the API library may be modified as required. However, operation of the source code will not be guaranteed in cases where it is modified. When incorporating the API library in equipment, be sure to verify the user program for any problems beforehand.

# 1.3 File Configuration

The root folder, and file/folder configuration for the API library are shown below.

## Root folder

Find the root folder of the API library by the following procedure.

### Windows®7

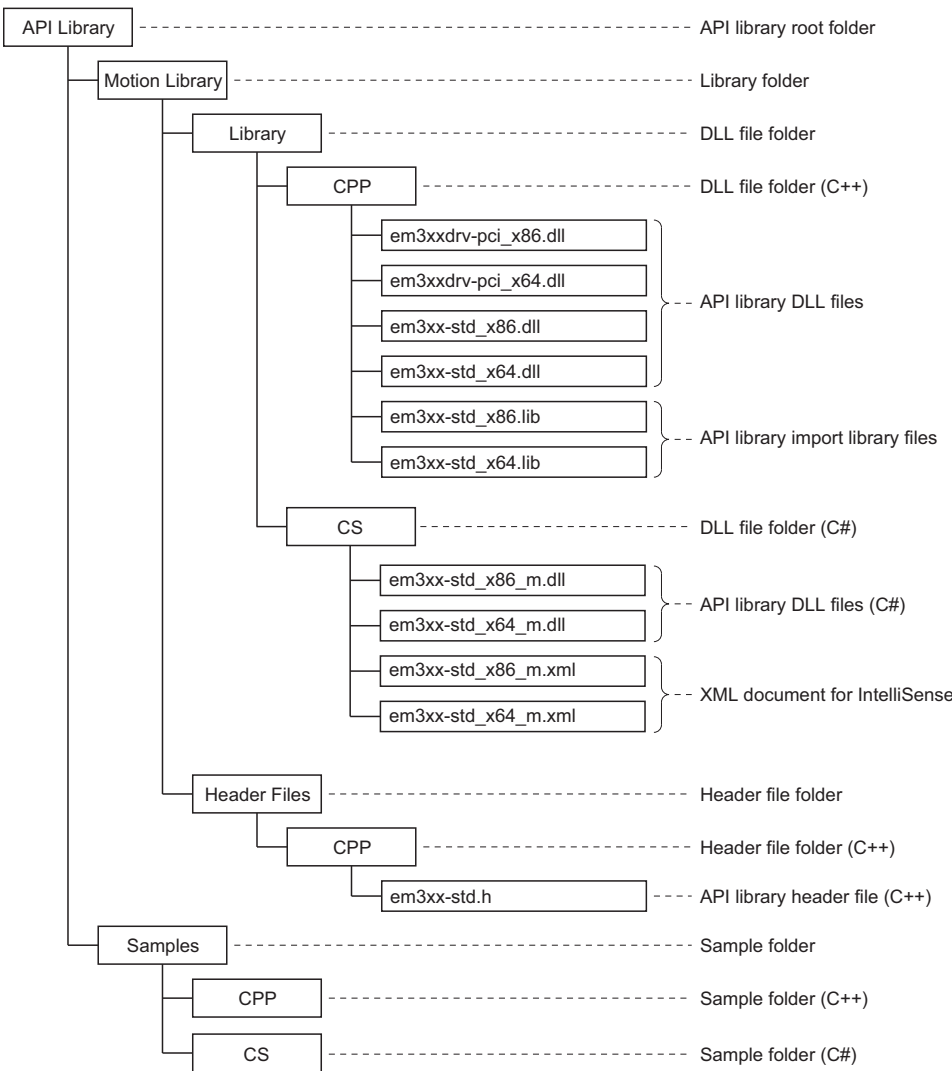
[Start]⇒[All programs]⇒[MELSOFT]⇒[EM Software Development Kit]⇒[EM API Library]

### Windows® 8.1 or later

[Start]⇒[All apps]⇒[MELSOFT]⇒[EM API Library]

## File/folder configuration

The file/folder configuration for the API library are shown below.



The API library version is written at the start of the header file for the C++ API library.

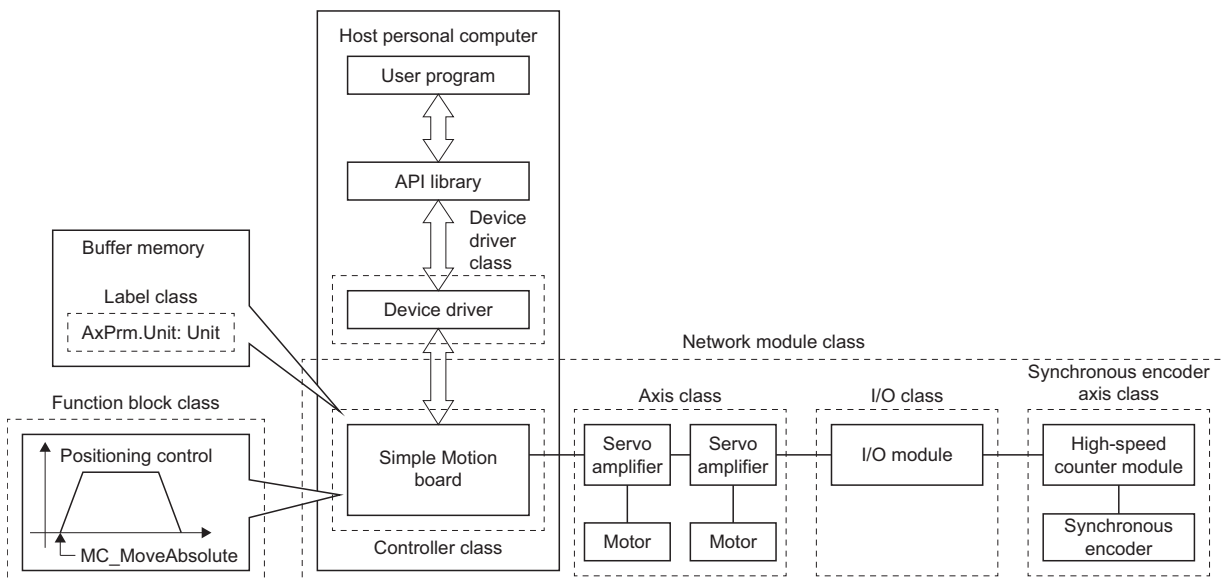
# 2 CLASS/FUNCTION LIST

## 2.1 Class List

The list of classes used in the API library is shown below.

Category	Class*1	Description	Reference
Network module class	MMC_NetworkModule	The base class of controller class, axis class, I/O class, and synchronous encoder axis class.	Page 19 MMC_NetworkModule class
Controller class	MMC_Controller	The class that controls the controller (MMC_Controller) and it is derived classes (MMC_EM340GF etc.). Controls such as the backup of execution data for the controller are executed by calling the method of this class. Objects of the controller class are generated with object generation functions, and deleted by Delete method.	Page 20 MMC_Controller class
Axis class	MMC_Axis	The class that controls the servo amplifier axis (MMC_Axis) and it is derived classes (MMC_J4GF, AXIS_REF etc.). Controls such as positioning start for the axis are executed by calling the method of this class. Get objects of the axis class by the method of the controller class.	Page 22 MMC_Axis class
I/O class	MMC_Io	The class that controls I/O modules (MMC_Io) and it is derived classes (MMC_CcieSlaveIo etc.). The link device area can be accessed through this class. Get objects of the I/O class by the get object method of the controller class.	Page 23 MMC_Io class
Synchronous encoder axis class	MMC_SyncEncoder	The class that controls the synchronous encoder axis (MMC_SyncEncoder) and it is derived classes (MMC_CcieSyncEncoder etc.). Controls such as changing the synchronous encoder axis current value for the synchronous encoder axis are executed by calling the method of this class. Get objects of the synchronous encoder axis class by the get object method of the controller class.	Page 24 MMC_SyncEncoder class
Function block class	MC_FunctionBlock	This class provides motion control functions for the specifications of the PLCopen Motion control function blocks. Objects of the function block class are generated with object generation functions, and deleted by Delete method.	Page 24 MC_FunctionBlock class
Label class	MMC_Label	This class provides functions that access the buffer memory of the Simple Motion board. The user program accesses the buffer memory by using the labels from the controller class and axis class as variables for the buffer memory.	Page 26 MMC_Label class
Device driver class	MMC_DeviceDriver	This class provides functions that access the Simple Motion board via a PCI Express connection. Controls such as starting the interrupt driver for the device driver are executed by calling the method of this class. Objects of the device driver class are generated with object generation functions, and deleted by Delete method.	Page 26 MMC_DeviceDriver class

\*1 MMC: Mitsubishi Motion Class



## 2.2 Public method/Data Method List

A list of the public methods and public data members used by the API library for each class is shown below.

Do not use any public methods and public data members other than those in the following lists. Using a public method or public data member that is not listed may cause the API library to operate incorrectly.

### MMC\_NetworkModule class

Refer to the following for details of MMC\_NetworkModule.

- Basic functions ([↩](#) Page 30 MMC\_NetworkModule Class)

#### Public method




Category	Name	Description	Reference
Object generation/ Delete method	Delete	Deletes object.	<a href="#">↩</a> Page 31 MMC_NetworkModule::Delete

#### Public data member






















Variable name	Data type	Initial value	Description
Tag	long long	0	A variable that the user may use freely. An integer value or pointer can be cast to long long type and stored.

# MMC\_Controller class

Refer to the following for details of MMC\_Controller class.

- Basic functions (  Page 32 MMC\_Controller Class)
- Advanced synchronous control (  Page 127 MMC\_Controller Class)
- Function block (  Page 189 MMC\_Controller Class)

## Public method

Category	Name	Description	Reference
Get object method	GetAxis	Gets the object of the axis class.	 Page 34 MMC_Controller::GetAxis (C++)  Page 35 MMC_Controller::GetAxis (C#)
	GetSlavelo	Gets the object of the I/O class.	 Page 36 MMC_Controller::GetSlavelo (C++)  Page 37 MMC_Controller::GetSlavelo (C#)
	GetSyncEncoder	Gets the object of the synchronous encoder axis class.	 Page 128 MMC_Controller::GetSyncEncoder
	GetFbAxisRef	Gets the object of the axis information class used in the function block class (Motion mode).	 Page 190 MMC_Controller::GetFbAxisRef
	GetFbAxisReflo	Gets the object of the axis information class used in the function block class (I/O mode).	 Page 191 MMC_Controller::GetFbAxisReflo
System method	ResetController	Executes remote RESET.	 Page 38 MMC_Controller::ResetController
	SetUserProgramReady	Sets the user program READY signal [Y0].	 Page 39 MMC_Controller::SetUserProgramReady
Backup method	InitializeParameter	Returns parameters to their initial factory values.	 Page 40 MMC_Controller::InitializeParameter
	BackupParameter	Performs backup of execution data (parameters etc.)	 Page 41 MMC_Controller::BackupParameter
Interrupt method	SetInterruptParameter	Sets the interrupt parameter.	 Page 42 MMC_Controller::SetInterruptParameter
	EnableInterrupt	Enables the interrupt output.	 Page 44 MMC_Controller::EnableInterrupt
	DisableInterrupt	Disables the interrupt output.	 Page 45 MMC_Controller::DisableInterrupt
	RegisterIntCallback	Registers the interrupt callback function.	 Page 46 MMC_Controller::RegisterIntCallback (C++)  Page 47 MMC_Controller::RegisterIntCallback (C#)
	UnregisterIntCallback	Unregisters the interrupt callback function.	 Page 49 MMC_Controller::UnregisterIntCallback
	ResetIntEvent	Sets the interrupt event to a nonsignaled state.	 Page 50 MMC_Controller::ResetIntEvent
	SetIntEvent	Sets the interrupt event to a signaled state.	 Page 51 MMC_Controller::SetIntEvent
	WaitIntEvent	Waits until the interrupt event is in a signaled state.	 Page 52 MMC_Controller::WaitIntEvent
DMA transmission method	EnableDMA	Enables DMA transmission.	 Page 53 MMC_Controller::EnableDMA
	DisableDMA	Disables DMA transmission.	 Page 54 MMC_Controller::DisableDMA
Synchronous control method	CalcCamCommandPosition	Calculates cam axis feed current value.	 Page 129 MMC_Controller::CalcCamCommandPosition
	CalcCamCommandPositionPerCycle	Calculates cam axis current value per cycle.	 Page 130 MMC_Controller::CalcCamCommandPositionPerCycle
	MakeRotaryCutterCam	Auto-generates the cam (central reference) for rotary cutter.	 Page 131 MMC_Controller::MakeRotaryCutterCam
	MakeEasyStrokeRatioCam	Auto-generates the easy stroke ratio cam.	 Page 132 MMC_Controller::MakeEasyStrokeRatioCam
	MakeAdvancedStrokeRatioCam	Auto-generates the advanced stroke ratio cam.	 Page 133 MMC_Controller::MakeAdvancedStrokeRatioCam



Category	Name	Description	Reference
SLMP communication method	ReadRemoteBufferMemory	Reads data from the buffer memory in the remote device station.	Page 55 MMC_Controller::ReadRemoteBufferMemory
	WriteRemoteBufferMemory	Writes data to the buffer memory in the remote device station.	Page 56 MMC_Controller::WriteRemoteBufferMemory

## Public data member































Variable name	Data type	Initial value	Description
LastErrorNumber	unsigned long	MMERR_NONE	Latest error code
LastDetailErrorNumber	unsigned long	MMERR_NONE	Latest detail error code

# MMC\_Axis class

Refer to the following for details of MMC\_Axis class.

- Basic functions (  Page 57 MMC\_Axis Class)
- Advanced synchronous control (  Page 134 MMC\_Axis Class)

## Public method

Category	Name	Description	Reference
Positioning data method	SetPositioningData	Sets the positioning data.	 Page 59 MMC_Axis::SetPositioningData
	SetBlockStartData	Sets the block start data.	 Page 60 MMC_Axis::SetBlockStartData
	SetBlockConditionData	Sets the condition data used by block start.	 Page 61 MMC_Axis::SetBlockConditionData
	GetPositioningData	Gets the positioning data.	 Page 62 MMC_Axis::GetPositioningData
	GetBlockStartData	Gets the block start data.	 Page 63 MMC_Axis::GetBlockStartData
	GetBlockConditionData	Gets the condition data used by block start.	 Page 64 MMC_Axis::GetBlockConditionData
Operation method	StartPositioning	Starts positioning control.	 Page 65 MMC_Axis::StartPositioning
	StartBlockPositioning	Starts advanced positioning control.	 Page 66 MMC_Axis::StartBlockPositioning
	StopPositioning	Stops axis.	 Page 67 MMC_Axis::StopPositioning
	RestartPositioning	Restarts stopped axis.	 Page 68 MMC_Axis::RestartPositioning
	WaitPositioningDone	Waits until completion of positioning control.	 Page 69 MMC_Axis::WaitPositioningDone
	ResetPositioningDoneIntEvent	Sets the positioning complete interrupt event to a nonsignaled state.	 Page 70 MMC_Axis::ResetPositioningDoneIntEvent
	SetPositioningDoneIntEvent	Sets the positioning complete interrupt event to a signaled state.	 Page 71 MMC_Axis::SetPositioningDoneIntEvent
	WaitPositioningDoneIntEvent	Waits until the positioning complete interrupt event is in a signaled state.	 Page 72 MMC_Axis::WaitPositioningDoneIntEvent
	StartJog	Starts JOG operation.	 Page 73 MMC_Axis::StartJog
	StopJog	Stops JOG operation.	 Page 74 MMC_Axis::StopJog
	StartInching	Starts inching operation.	 Page 75 MMC_Axis::StartInching
	EnableMPG	Enables manual pulse generator operation.	 Page 76 MMC_Axis::EnableMPG
	DisableMPG	Disables manual pulse generator operation.	 Page 77 MMC_Axis::DisableMPG
	ChangeControlMode	Changes control mode.	 Page 78 MMC_Axis::ChangeControlMode
	Change method	ChangeSpeed	Changes speed and acceleration/deceleration time.
ChangePosition		Changes target position and command speed.	 Page 80 MMC_Axis::ChangePosition
Error method	ResetError	Performs error reset.	 Page 81 MMC_Axis::ResetError
Interrupt method	SetInterruptParameter	Sets the interrupt parameter.	 Page 82 MMC_Axis::SetInterruptParameter
	ResetIntEvent	Sets the interrupt event to a nonsignaled state.	 Page 84 MMC_Axis::ResetIntEvent
	SetIntEvent	Sets the interrupt event to a signaled state.	 Page 85 MMC_Axis::SetIntEvent
	WaitIntEvent	Waits until the interrupt event is in a signaled state.	 Page 86 MMC_Axis::WaitIntEvent
Synchronous control method	StartSync	Starts synchronous control.	 Page 135 MMC_Axis::StartSync
	StopSync	Stops synchronous control.	 Page 136 MMC_Axis::StopSync
	ChangeSyncPosition	Changes current value during synchronous control.	 Page 137 MMC_Axis::ChangeSyncPosition
	MoveCamPosition	Moves cam axis during synchronous control.	 Page 138 MMC_Axis::MoveCamPosition




## Public data member

Variable name	Data type	Initial value	Description
LastErrorNumber	unsigned long	MMERR_NONE	Latest error code
LastDetailErrorNumber	unsigned long	MMERR_NONE	Latest detail error code
StationNumber	unsigned long	—	Station No. (1 to 16)

## MMC\_lo class

Refer to the following for details of MMC\_lo class.

- Basic functions (  Page 87 MMC\_lo Class)

## Public method

There is no public method.

## Public data member


Variable name	Data type	Initial value	Description
LastErrorNumber	unsigned long	MMERR_NONE	Latest error code
LastDetailErrorNumber	unsigned long	MMERR_NONE	Latest detail error code
StationNumber	unsigned long	—	Station No. (1 to 120)

## Label





Variable name	Data type	Description
bRX[]	bool	RX area
bRY[]	bool	RY area
wRX[]	unsigned short	RX area (for word access)
wRY[]	unsigned short	RY area (for word access)
dwRX[]	unsigned long	RX area (for double word access)
dwRY[]	unsigned long	RY area (for double word access)
wRWw[]	unsigned short	RWw area
wRWr[]	unsigned short	RWr area
dwRWw[]	unsigned long	RWw area (for double word access)
dwRWr[]	unsigned long	RWr area (for double word access)
wRWw[].b[]	bool	RWw area (for bit access)
wRWr[].b[]	bool	RWr area (for bit access)

## MMC\_SyncEncoder class

Refer to the following for details of MMC\_SyncEncoder class.

- Advanced synchronous control function (  Page 139 MMC\_SyncEncoder Class)

### Public method


Category	Name	Description	Reference
Error method	ResetSyncEncoderError	Performs error reset of the synchronous encoder axis.	 Page 140 MMC_SyncEncoder::ResetSyncEncoderError
Change method	ChangeSyncEncoderPosition	Changes current value of the synchronous encoder axis.	 Page 141 MMC_SyncEncoder::ChangeSyncEncoderPosition
Enable/disable method	DisableSyncEncoder	Disables input from the synchronous encoder axis.	 Page 142 MMC_SyncEncoder::DisableSyncEncoder
	EnableSyncEncoder	Enables input from the synchronous encoder axis.	 Page 143 MMC_SyncEncoder::EnableSyncEncoder

### Public data member


Variable name	Data type	Initial value	Description
LastErrorNumber	unsigned long	MMERR_NONE	Latest error code
LastDetailErrorNumber	unsigned long	MMERR_NONE	Latest detail error code
AxisNumber	unsigned long	—	Synchronous encoder axis No. (1 to 16)

## MC\_FunctionBlock class

Refer to the following for details of MC\_FunctionBlock class.

- Function block (  Page 144 MC\_FunctionBlock Class)

### Public method

Category	Name	Description	Reference
Update method	Update	Updates the I/O data of the function block.	 Page 152 MC_FunctionBlock::Update

### Public data member

Variable name	Data type	Initial value	Description
Tag	long long	0	A variable that the user may use freely. An integer value or pointer can be cast to long long type and stored.

## Derived classes

Name	Description	Reference
MC_Power	Changes the servo amplifier of the specified axis to an operable state.	☞ Page 153 MC_Power class
MCv_Home	Executes home position return for the specified axis.	☞ Page 155 MCv_Home class
MC_Stop	Stops the specified axis.	☞ Page 157 MC_Stop class
MC_MoveAbsolute	Specifies the absolute target position of the specified axis and executes positioning.	☞ Page 159 MC_MoveAbsolute class
MC_MoveRelative	Moves the specified distance from the current position.	☞ Page 162 MC_MoveRelative class
MC_Reset	Cancels the errors of the specified axis.	☞ Page 165 MC_Reset class
MC_MoveAdditive	Adds the most recent relative position specified by the positioning command of the specified axis, and executes positioning.	☞ Page 167 MC_MoveAdditive class
MC_MoveVelocity	Executes speed control for the specified axis at the specified speed.	☞ Page 169 MC_MoveVelocity class
MC_TorqueControl	Executes torque control for the specified axis at the specified torque.	☞ Page 171 MC_TorqueControl class
MC_SetPosition	Changes the current position (command position and feedback position) of the specified axis.	☞ Page 173 MC_SetPosition class
MC_SetOverride	Changes the target speed of the specified axis.	☞ Page 175 MC_SetOverride class
MC_ReadActualPosition	Reads the current position of the specified axis.	☞ Page 177 MC_ReadActualPosition class
MC_ReadStatus	Returns the detailed state of the state diagram of the specified axis.	☞ Page 179 MC_ReadStatus class
MC_ReadAxisInfo	Reads the axis information of the specified axis.	☞ Page 181 MC_ReadAxisInfo class
MC_ReadAxisError	Reads the error No. of the specified axis.	☞ Page 183 MC_ReadAxisError class
MCv_ReadServoParameter	Reads the parameter value of the servo parameter No. of the specified axis.	☞ Page 185 MCv_ReadServoParameter class
MCv_WriteServoParameter	Changes the parameter value of the servo parameter No. of the specified axis.	☞ Page 187 MCv_WriteServoParameter class

## MMC\_Label class

Refer to the following for details of MMC\_Label class.

- Basic functions ( [Page 89 MMC\\_Label Class](#) )

### Public method

Category	Name	Description	Reference
Wait method	Wait	Waits by polling until the value of the label satisfies the specified conditions.	<a href="#">Page 90 Wait</a>

### Public data member

There is no public data member.

## MMC\_DeviceDriver class

Refer to the following for details of MMC\_DeviceDriver class.

- Basic functions ( [Page 92 MMC\\_DeviceDriver Class](#) )

### Public method

Category	Name	Description	Reference
Delete/generate object method	Delete	Deletes object.	<a href="#">Page 93 MMC_DeviceDriver::Delete</a>
Open and close method	Open	Opens device.	<a href="#">Page 94 MMC_DeviceDriver::Open</a>
	Close	Closes device.	<a href="#">Page 95 MMC_DeviceDriver::Close</a>
Interrupt method	StartInterrupt	Starts interrupt driver.	<a href="#">Page 96</a> MMC_DeviceDriver::StartInterrupt
	EndInterrupt	Ends interrupt driver.	<a href="#">Page 97</a> MMC_DeviceDriver::EndInterrupt
DMA transmission method	StartDMA	Starts DMA transmission driver.	<a href="#">Page 98 MMC_DeviceDriver::StartDMA</a>
	EndDMA	Ends DMA transmission driver.	<a href="#">Page 99 MMC_DeviceDriver::EndDMA</a>
Buffer memory access method	SetBufferMemory	Sets data to buffer memory.	<a href="#">Page 100</a> MMC_DeviceDriver::SetBufferMemory
	GetBufferMemory	Gets data from buffer memory.	<a href="#">Page 101</a> MMC_DeviceDriver::GetBufferMemory

### Public data member

Variable name	Data type	Initial value	Description
LastErrorNumber	unsigned long	MMERR_NONE	Latest error code
LastDetailErrorNumber	unsigned long	MMERR_NONE	Latest detail error code
BoardID	unsigned long	—	Board ID
Tag	long long	0	A variable that the user may use freely. An integer value or pointer can be cast to long long type and stored.

## 2.3 Function List

A list of the functions used by the API library is shown below.

### Object generation functions

Category	Function name	Description	Reference
Object generation function	MmfCreateEM340GF	Generates MMC_EM340GF class objects.	<a href="#">Page 102</a> <a href="#">MmfCreateEM340GF (C++)</a> <a href="#">Page 103</a> <a href="#">MmfCreateEM340GF (C#)</a>
	MmfCreatePciDevice	Generates PCI Express device driver class objects.	<a href="#">Page 104</a> <a href="#">MmfCreatePciDevice (C++)</a> <a href="#">Page 105</a> <a href="#">MmfCreatePciDevice (C#)</a>
	MmfCreateFunctionBlock	Generates function block class objects.	<a href="#">Page 195</a> <a href="#">MmfCreateFunctionBlock</a>

### Basic library functions

Category	Function name	Description	Reference
Error function	MmfGetLastError	Gets the error code.	<a href="#">Page 106 MmfGetLastError</a>
Semaphore function	MmfCreateSemaphore	Opens or creates the semaphore object.	<a href="#">Page 107</a> <a href="#">MmfCreateSemaphore</a>
	MmfDeleteSemaphore	Closes the handle of the semaphore object.	<a href="#">Page 108</a> <a href="#">MmfDeleteSemaphore</a>
	MmfWaitSemaphore	Returns control when the semaphore object becomes a signaled state.	<a href="#">Page 109</a> <a href="#">MmfWaitSemaphore</a>
	MmfReleaseSemaphore	Increases the count of the semaphore object by one.	<a href="#">Page 110</a> <a href="#">MmfReleaseSemaphore</a>
Event function	MmfCreateEvent	Opens or creates the event object.	<a href="#">Page 111 MmfCreateEvent</a>
	MmfDeleteEvent	Closes the handle of the event object.	<a href="#">Page 112 MmfDeleteEvent</a>
	MmfResetEvent	Sets the event object to a nonsignaled state.	<a href="#">Page 113 MmfResetEvent</a>
	MmfSetEvent	Sets the event object to a signaled state.	<a href="#">Page 114 MmfSetEvent</a>
	MmfWaitEvent	Returns control when the event object becomes a signaled state.	<a href="#">Page 115 MmfWaitEvent</a>
Thread function	MmfCreateThread	Creates the thread.	<a href="#">Page 116 MmfCreateThread</a>
	MmfDeleteThread	Closes the handle of the thread.	<a href="#">Page 117 MmfDeleteThread</a>
	MmfWaitThread	Returns control when the thread object becomes a signaled state.	<a href="#">Page 118 MmfWaitThread</a>
	MmfSetThreadPriority	Sets the priority value of the thread.	<a href="#">Page 119</a> <a href="#">MmfSetThreadPriority</a>
	MmfResumeThread	Decrements a thread's suspend count by one.	<a href="#">Page 120</a> <a href="#">MmfResumeThread</a>
	MmfGetExitCodeThread	Gets the exit code of the thread.	<a href="#">Page 121</a> <a href="#">MmfGetExitCodeThread</a>
	MmfExitThread	Exits the thread.	<a href="#">Page 122 MmfExitThread</a>
Timer function	MmfGetCurrentTime	Gets the time elapsed since system startup.	<a href="#">Page 123</a> <a href="#">MmfGetCurrentTime</a>
	MmfCheckPassTime	Checks the specified time has passed.	<a href="#">Page 124</a> <a href="#">MmfCheckPassTime</a>
	MmfWaitDelayTime	Cancels the execution of thread and specified time.	<a href="#">Page 125</a> <a href="#">MmfWaitDelayTime</a>

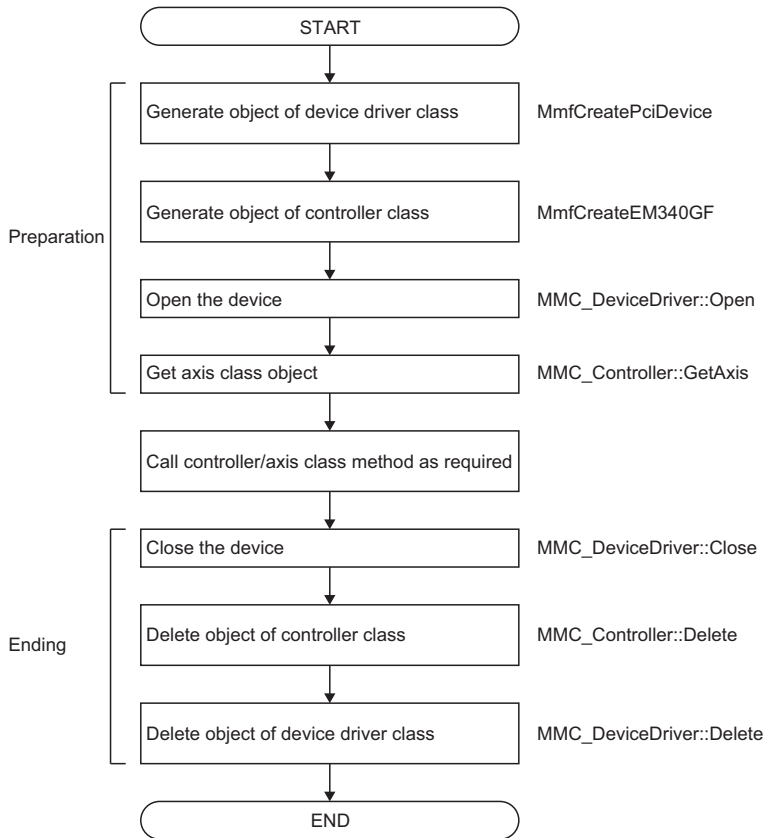
# 3 API LIBRARY DETAILS (BASIC FUNCTIONS)

This chapter describes the procedures for using the basic functions of the API library, and the relevant classes.

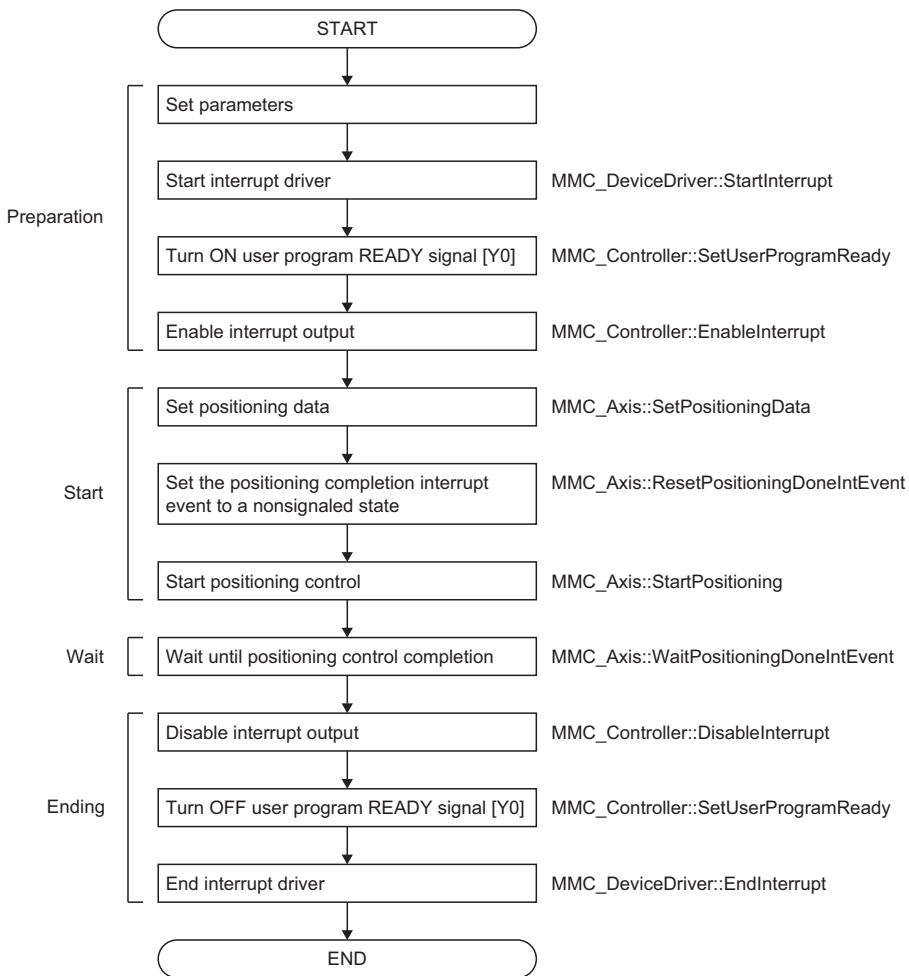
## Execution procedure

An example of the execution procedure is shown below.

### ■Preparation and ending procedure for using the methods of controller class and axis class



## ■ Procedure for starting positioning control, and waiting for positioning completion interrupt




# 3.1 MMC\_NetworkModule Class

This class controls the network module.

class MMC\_NetworkModule

## Member

### Public method

Category	Name	Description	Reference
Object generation/ Delete method	Delete	Deletes object.	 Page 31 MMC_NetworkModule::Delete

### Public data member

Variable name	Data type	Initial value	Description
Tag	long long	0	A variable that the user may use freely. An integer value or pointer can be cast to long long type and stored.

## Inheritance hierarchy

MMC\_NetworkModule



# MMC\_NetworkModule::Delete

Deletes axis object.

```
void Delete(  
    void  
);
```

## Detailed description

### ■Parameter

None.

### ■Return value

None.

### ■Point

Because the objects gotten by the get object method of the controller class (GetAxis, GetSlavelo, GetSyncEncoder, GetFbAxisRef) are automatically deleted at the time of controller class object delete, calling this method is not necessary.

### ■Supported version

API version	Software version
1.00	01

### ■Reference

None.

## 3.2 MMC\_Controller Class

This class controls the controller.

```
class MMC_Controller: public MMC_Master
```

### Member

#### Public method

Category	Name	Description	Reference
Get object method	GetAxis	Gets the object of the axis class.	<a href="#">Page 34 MMC_Controller::GetAxis (C++)</a> <a href="#">Page 35 MMC_Controller::GetAxis (C#)</a>
	GetSlavelo	Gets the object of the I/O class.	<a href="#">Page 36 MMC_Controller::GetSlavelo (C++)</a> <a href="#">Page 37 MMC_Controller::GetSlavelo (C#)</a>
System method	ResetController	Executes remote RESET.	<a href="#">Page 38 MMC_Controller::ResetController</a>
	SetUserProgramReady	Sets the user program READY signal [Y0].	<a href="#">Page 39 MMC_Controller::SetUserProgramReady</a>
Backup method	InitializeParameter	Returns parameters to their initial factory values.	<a href="#">Page 40 MMC_Controller::InitializeParameter</a>
	BackupParameter	Performs backup of execution data (parameters etc.)	<a href="#">Page 41 MMC_Controller::BackupParameter</a>
Interrupt method	SetInterruptParameter	Sets the interrupt parameter.	<a href="#">Page 42 MMC_Controller::SetInterruptParameter</a>
	EnableInterrupt	Enables the interrupt output.	<a href="#">Page 44 MMC_Controller::EnableInterrupt</a>
	DisableInterrupt	Disables the interrupt output.	<a href="#">Page 45 MMC_Controller::DisableInterrupt</a>
	RegisterIntCallback	Registers the interrupt callback function.	<a href="#">Page 46 MMC_Controller::RegisterIntCallback (C++)</a> <a href="#">Page 47 MMC_Controller::RegisterIntCallback (C#)</a>
	UnregisterIntCallback	Unregisters the interrupt callback function.	<a href="#">Page 49 MMC_Controller::UnregisterIntCallback</a>
	ResetIntEvent	Sets the interrupt event to a nonsignaled state.	<a href="#">Page 50 MMC_Controller::ResetIntEvent</a>
	SetIntEvent	Sets the interrupt event to a signaled state.	<a href="#">Page 51 MMC_Controller::SetIntEvent</a>
	WaitIntEvent	Waits until the interrupt event is in a signaled state.	<a href="#">Page 52 MMC_Controller::WaitIntEvent</a>
DMA transmission method	EnableDMA	Enables DMA transmission.	<a href="#">Page 53 MMC_Controller::EnableDMA</a>
	DisableDMA	Disables DMA transmission.	<a href="#">Page 54 MMC_Controller::DisableDMA</a>
SLMP communication method	ReadRemoteBufferMemory	Reads data from the buffer memory in the remote device station.	<a href="#">Page 55 MMC_Controller::ReadRemoteBufferMemory</a>
	WriteRemoteBufferMemory	Writes data to the buffer memory in the remote device station.	<a href="#">Page 56 MMC_Controller::WriteRemoteBufferMemory</a>

#### Public data member

Variable name	Data type	Initial value	Description
LastErrorNumber	unsigned long	MMERR_NONE	Latest error code
LastDetailErrorNumber	unsigned long	MMERR_NONE	Latest detail error code

#### Label

Refer to label list for labels. ([Page 203 LABEL LIST](#))

#### Point

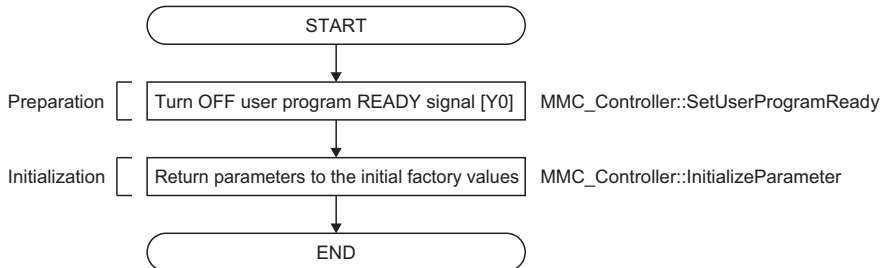
Generate controller class objects with object generation function.

## Inheritance hierarchy

MMC\_NetworkModule  
MMC\_Master  
MMC\_Controller

## Execution procedure

The procedure for returning parameters to their initial factory values is given as an example.



## MMC\_Controller::GetAxis (C++)

Gets axis class object.

```
unsigned long GetAxis(  
    unsigned long axisNumber,  
    MMC_Axis **axis  
);
```

### Detailed description

#### ■Parameter

Argument	Description
axisNumber [in]	Axis No. (1 to 16)
axis [out]	Pointer to the axis class object pointer.

#### ■Return value

Value	Description
MMC_OK	Function succeeded
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_CREATE_OBJECT	Failed to generate object.

#### ■Point

Because the objects gotten by this method are automatically deleted at the time of controller class object delete, calling the Delete method of the axis class is not necessary.

#### ■Supported version

API version	Software version
1.00	01

#### ■Reference

None.

## MMC\_Controller::GetAxis (C#)

Gets axis class object.

```
MMC_J4GF^ GetAxis(  
    uint axisNumber  
);
```

### Detailed description

#### Parameter

Argument	Description
axisNumber [in]	Axis No. (1 to 16)

#### Return value

Value	Description
Reference to the axis class object	Function succeeded
null	Failed to generate object. Call the MmfGetLastError function of MMC_BasicLibrary class and confirm the error details.

#### Point

Because the objects gotten by this method are automatically deleted at the time of controller class object delete, calling the Delete method of the axis class is not necessary.

#### Supported version

API version	Software version
1.30	01

#### Reference

None.

## MMC\_Controller::GetSlavelo (C++)

Gets I/O class object.

```
unsigned long GetSlavelo(  
    unsigned long stationNumber,  
    MMC_lo **slavelo  
);
```

### Detailed description

#### Parameter

Argument	Description
stationNumber [in]	Station No. (1 to 120)
slavelo [out]	Pointer to the I/O class object pointer.

#### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_CREATE_OBJECT	Failed to generate object.
MMERR_CONDITION_READY_SIGNAL_OFF	The READY signal [X0] is OFF.

#### Point

- Call this method after setting the parameter, calling the SetUserProgramReady method, and turning ON the user program READY signal [Y0].
- Because the objects gotten by this method are automatically deleted at the time of controller class object delete, calling the Delete method of the I/O class is not necessary.

#### Supported version

API version	Software version
1.00	01

#### Reference

None.

# MMC\_Controller::GetSlavelo (C#)

Gets I/O class object.

```
MMC_CcieSlavelo^ GetSlavelo(  
    uint stationNumber  
);
```

## Detailed description

### Parameter

Argument	Description
stationNumber [in]	Station No. (1 to 120)

### Return value

Value	Description
Reference to the I/O class object	Function succeeded
null	Failed to generate object. Call the MmfGetLastError function of MMC_BasicLibrary class and confirm the error details.

### Point

- Call this method after setting the parameter, calling the SetUserProgramReady method, and turning ON the user program READY signal [Y0].
- Because the objects gotten by this method are automatically deleted at the time of controller class object delete, calling the Delete method of the I/O class is not necessary.

### Supported version

API version	Software version
1.30	01

### Reference

None.

# MMC\_Controller::ResetController

Executes remote RESET.

```
unsigned long ResetController(  
    void  
);
```

## Detailed description

### ■Parameter

None.

### ■Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_TIMEOUT_01	The timeout time (30 seconds) has elapsed.

### ■Point

After writing "1EA5H" to "[Cd.1180] Remote RESET start", this method waits inside the method until the Simple Motion board turns ON the synchronous flag [X1].

### ■Supported version

API version	Software version
1.00	01

### ■Reference

None.



# MMC\_Controller::SetUserProgramReady

Sets the user program READY signal [Y0].

```
unsigned long SetUserProgramReady(  
    bool userProgramReady  
);
```

## Detailed description

### Parameter

Argument	Description
userProgramReady [in]	User program READY signal [Y0] (MMC_ON or MMC_OFF)

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_TIMEOUT_01	The timeout time (1 second) has elapsed.

### Point

- When manipulating the user program READY signal [Y0], use this method.
- This method waits inside the method until the Simple Motion board updates the READY signal [X0].
- This method writes the local time gotten from the operating system to "[Pr.1160] Startup time".

### Supported version

API version	Software version
1.00	01

### Reference

None.

# MMC\_Controller::InitializeParameter

Returns the data set in the buffer memory/internal memory and flash ROM/internal memory (nonvolatile) to their initial factory values.

```
unsigned long InitializeParameter(  
    void  
);
```

## Detailed description

### ■Parameter

None.

### ■Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_TIMEOUT_01	The timeout time (30 seconds) has elapsed.
MMERR_CONDITION_USER_PROGRAM_READY_ON	User program READY signal [Y0] is ON.
MMERR_CONDITION_BUSY_ON	BUSY signal [X10 to X1F] are ON.

### ■Point

- This method uses the "Parameter initialization function" of the Simple Motion board.
- This methods waits inside the method until "[Cd.2] Parameter initialization request" becomes "0".
- It takes a maximum of approximately 20 seconds for parameters initialization.

### ■Supported version

API version	Software version
1.00	01

### ■Reference

None.

# MMC\_Controller::BackupParameter

Writes execution data to flash ROM/internal memory (nonvolatile).

```
unsigned long BackupParameter(  
    void  
);
```

## Detailed description

### ■Parameter

None.

### ■Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_TIMEOUT_01	The timeout time (10 seconds) has elapsed.
MMERR_CONDITION_USER_PROGRAM_READY_ON	User program READY signal [Y0] is ON.
MMERR_CONDITION_BUSY_ON	BUSY signal [X10 to X1F] are ON.

### ■Point

- This method uses the "Execution data backup function" of the Simple Motion board.
- This method waits inside the method until "[Cd.1] Flash ROM write request" becomes "0".

### ■Supported version

API version	Software version
1.00	01

### ■Reference

None.

# MMC\_Controller::SetInterruptParameter

Sets the interrupt parameter.

```
unsigned long SetInterruptParameter(  
    unsigned long intNumber,  
    MMC_Label &label,  
    const MMST_InterruptParameter &interruptParameter  
);
```

## Detailed description

### Parameter

Argument	Description
intNumber [in]	Interrupt factor No. (9 to 64)
label [in]	Interrupt factor label
interruptParameter [in]	Interrupt parameter structure
interruptParameter.IntCondition [in]	Interrupt condition <sup>*1</sup>
interruptParameter.IntSensitivity [in]	Interrupt detection timing <sup>*2</sup>
interruptParameter.IntOutputDataSeting [in]	Factor details output setting <sup>*3</sup>
interruptParameter.IntTimeSetting [in]	Condition for condition completion continue <sup>*4</sup>
interruptParameter.IntJudgeValue1 [in]	Interrupt condition judge value 1
interruptParameter.IntJudgeValue2 [in]	Interrupt condition judge value 2
interruptParameter.IntTime [in]	Time for condition completion [ms]

\*1 Specify the following values for interrupt condition.

Value	Description
MMC_INT_CONDITION_DISABLE	Do not detect
MMC_INT_CONDITION_EQUAL	Equal to interrupt condition judge value 1
MMC_INT_CONDITION_NOT_EQUAL	Not equal to interrupt condition judge value 1
MMC_INT_CONDITION_GREATER	Larger than interrupt condition judge value 1
MMC_INT_CONDITION_LESS	Smaller than interrupt condition judge value 1
MMC_INT_CONDITION_WITHIN	Within the range of interrupt condition judge value 1, 2
MMC_INT_CONDITION_OUTSIDE	Outside the range of interrupt condition judge value 1, 2
MMC_INT_CONDITION_CHANGE_BIT	The value masked by interrupt condition judge value 1 is different from the previous value
MMC_INT_CONDITION_ALL_BIT_ON	All points of the bit masked by interrupt condition judge value 1 are ON.
MMC_INT_CONDITION_ALL_BIT_OFF	All points of the bit masked by interrupt condition judge value 1 are OFF.
MMC_INT_CONDITION_BIT_ON	A point of the bit masked by interrupt condition judge value 1 is ON.
MMC_INT_CONDITION_BIT_OFF	A point of the bit masked by interrupt condition judge value 1 is OFF.

\*2 Specify the following values for interrupt detection timing.

Value	Description
MMC_INT_SENSITIVITY_EDGE	Edge detection
MMC_INT_SENSITIVITY_LEVEL	Level detection

\*3 Specify the following values for factor details output setting.

Value	Description
MMC_INT_OUTPUT_DATA_SETTING_FIRST_DATA	First data
MMC_INT_OUTPUT_DATA_SETTING_LAST_DATA	Last data

\*4 Specify the following values for condition for condition completion continue.

Value	Description
MMC_INT_TIME_SETTING_RESTART	Start a new count at another interrupt condition completion
MMC_INT_TIME_SETTING_CONTINUE	Continue count

## Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.

## Point

- When a bool label is specified to an interrupt factor label, setting interrupt condition judge value 1 is not required because the API library automatically sets the mask value.
- When a bool label is specified to an interrupt factor label, specify one of the following to the interrupt condition.
  - MMC\_INT\_CONDITION\_CHANGE\_BIT
  - MMC\_INT\_CONDITION\_ALL\_BIT\_ON
  - MMC\_INT\_CONDITION\_ALL\_BIT\_OFF
  - MMC\_INT\_CONDITION\_BIT\_ON
  - MMC\_INT\_CONDITION\_BIT\_OFF
- The interrupt factor No. (1 to 8) is used inside the API library therefore it cannot be used by the user program.
- The interrupt parameter is enabled by turning OFF→ON the user program READY signal [Y0].
- When a value outside the set range is set to the member variable of a structure, the No. of the member variable (1 or more) is stored to the details error code.
- When MMC\_NULL\_LABEL is specified to label, the interrupt parameter of the interrupt No. specified by intNumber is initialized.
- The operation of the axis class SetInterruptParameter method is the same as the operation of this method.

## Supported version

API version	Software version
1.00	01

## Reference

None.

# MMC\_Controller::EnableInterrupt

Enables the interrupt output.

```
unsigned long EnableInterrupt(  
    void  
);
```

## Detailed description

### Parameter

None.

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_WIN_RESET_EVENT	An error occurred in the ResetEvent function (Win32API). Call the GetLastError function of Win32API and confirm the error details.

### Point

- When using the interrupt function, enable the interrupt output by the procedure below.

1. Call the StartInterrupt method of the device driver class.
2. Call the SetUserProgramReady method of the controller class and turn ON user program READY signal [Y0].
3. Call the EnableInterrupt method of the controller class.
  - This method writes "1: Interrupt enabled" to "[Cd.1100] Interrupt enable request".

### Supported version

API version	Software version
1.00	01

### Reference

DisableInterrupt (  Page 45 MMC\_Controller::DisableInterrupt)

# MMC\_Controller::DisableInterrupt

Disables the interrupt output.

```
unsigned long DisableInterrupt(  
    void  
);
```

## Detailed description

### ■Parameter

None.

### ■Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_WIN_RESET_EVENT	An error occurred in the ResetEvent function (Win32API). Call the GetLastError function of Win32API and confirm the error details.

### ■Point

- Before using the EndInterrupt method of the device driver class, call this method.
- This method writes "0: Disabled" to "[Cd.1100] Interrupt enable request".

### ■Supported version

API library	Software version
1.00	01

### ■Reference

EnableInterrupt (  Page 44 MMC\_Controller::EnableInterrupt)

# MMC\_Controller::RegisterIntCallback (C++)

Registers the interrupt callback function. The registered function is called back from the interrupt driver when an interrupt occurs.

```
unsigned long RegisterIntCallback(  
    void *interruptProc  
);
```

## Detailed description

### Parameter

Argument	Description
interruptProc [in]	Callback function pointer

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_INT_ALREADY_REREGISTER_CALLBACK	The interrupt callback function has already been registered. To change the interrupt callback function, call the UnregisterIntCallback method.

### Point

- When using C++, write the `__stdcall` declaration.
- Clearing the interrupt by "[Cd.1101] Interrupt factor reset request" in the callback function is not required.
- The callback function is called back from the interrupt driver, therefore, write the least code as possible without the infinite waiting processing.
- The callback function is called back before the interrupt factor occurrence waiting functions such as the `WaitIntEvent` function.
- The value of the free-running timer at the time of when the API library received the Simple Motion board interrupt is stored in the free-running timer of the `MMST_InterruptData` structure. The free-running timer increases by one every 250[μs].
- The interrupt callback function is also called when the following conditions occur. The axis where the condition occurs is stored as bit data (b0 to b15: Axis 1 to 16) in "[Md.1102] Interrupt factor details" of the `MMST_InterruptData` structure.

Interrupt factor	Interrupt factor No.
Error detection ("[Md.31] Status": b13) ON edge	1
Axis warning detection ("[Md.31] Status": b9) ON edge	2
BUSY signal [X10 to X1F] OFF edge	3
"[Md.1190] Controller in-position flag" ON edge	4

### Example

```
void RegisterIntCallbackSample(MMC_Controller *controller)  
{  
    /* Register interrupt callback function */  
    unsigned long returnCode = controller->RegisterIntCallback( InterruptProc );  
    if( returnCode != MMC_OK ){ /* Error processing */ }  
}  
  
/* Callback function */  
void __stdcall InterruptProc(MMST_InterruptData *interruptData)  
{  
    /* Processing at interrupt */  
}
```

### Supported version

API version	Software version
1.00	01

### Reference

UnregisterIntCallback (  Page 49 MMC\_Controller::UnregisterIntCallback)



# MMC\_Controller::RegisterIntCallback (C#)

Registers the interrupt callback function. The registered function is called back from the interrupt driver when an interrupt occurs.

```
uint RegisterIntCallback(
    INT_CB_FUNC^ int_CB_FUNC
);
```

## Detailed description

### Parameter

Argument	Description
int_CB_FUNC [in]	Reference to the delegate registered the callback function

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_INT_ALREADY_REREGISTER_CALLBACK	The interrupt callback function has already been registered. To change the interrupt callback function, call the UnregisterIntCallback method.

### Point

- When using C++, write the `__stdcall` declaration.
- Clearing the interrupt by "[Cd.1101] Interrupt factor reset request" in the callback function is not required.
- The callback function is called back from the interrupt driver, therefore, write the least code as possible without the infinite waiting processing.
- The callback function is called back before the interrupt factor occurrence waiting functions such as the `WaitIntEvent` function.
- The value of the free-running timer at the time of when the API library received the Simple Motion board interrupt is stored in the free-running timer of the `MMST_InterruptData` structure. The free-running timer increases by one every 250[μs].
- The interrupt callback function is also called when the following conditions occur. The axis where the condition occurs is stored as bit data (b0 to b15: Axis 1 to 16) in "[Md.1102] Interrupt factor details" of the `MMST_InterruptData` structure.

Interrupt factor	Interrupt factor No.
Error detection ("[Md.31] Status": b13) ON edge	1
Axis warning detection ("[Md.31] Status": b9) ON edge	2
BUSY signal [X10 to X1F] OFF edge	3
"[Md.1190] Controller in-position flag" ON edge	4

### Example

```
void RegisterIntCallbackSample( MMC_EM340GF controller )
{
    /* Register function to delegate */
    MMC_Controller.INT_CB_FUNC callback = InterruptProc;

    /* Register interrupt callback function */
    uint retCode = controller.RegisterIntCallback( callback );
    if( retCode != MMC_ConstLibrary.MMC_OK ){ /* Error processing */ }
}

/* Callback function */
void InterruptProc( MMST_InterruptData interruptData )
{
    /* Processing at interrupt */
}
```

### Supported version

API version	Software version
1.30	01

## ■Reference

UnregisterIntCallback (  Page 49 MMC\_Controller::UnregisterIntCallback)

# MMC\_Controller::UnregisterIntCallback

Unregisters the interrupt callback function.

```
unsigned long UnregisterIntCallback(  
    void  
);
```

## Detailed description

### Parameter

None.

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_INT_ALREADY_UNREGISTER_CALLBACK	The interrupt callback function has already been unregistered.

### Point

None.

### Supported version

API version	Software version
1.00	01

### Reference

RegisterIntCallbacks

([↩](#) Page 46 MMC\_Controller::RegisterIntCallback (C++), [↩](#) Page 47 MMC\_Controller::RegisterIntCallback (C#))

# MMC\_Controller::ResetIntEvent

Sets the interrupt event to a nonsignaled state. This function is used if interrupt events occurring prior to calling the WaitIntEvent method are to be disabled.

```
unsigned long ResetIntEvent(  
    unsigned long intNumber  
);
```

## Detailed description

### Parameter

Argument	Description
intNumber [in]	Interrupt factor No. (9 to 64)

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_INT_NOT_START_DRIVER	The interrupt driver is stopped. Call the StartInterrupt method.
MMERR_WIN_RESET_EVENT	An error occurred in the ResetEvent function (Win32API). Call the GetLastError function of Win32API and confirm the error details.

### Point

- The interrupt factor No. (1 to 8) is used inside the API library therefore it cannot be used by the user program.
- When using interrupt factor No. (9 to 64), the interrupt parameter must be set by the SetInterruptParameter method beforehand.
- The operation of the axis class ResetIntEvent method is the same as the operation of this method.

### Supported version

API version	Software version
1.00	01

### Reference

SetIntEvent (  Page 51 MMC\_Controller::SetIntEvent)

WaitIntEvent (  Page 52 MMC\_Controller::WaitIntEvent)

# MMC\_Controller::SetIntEvent

Sets the interrupt event to a signaled state. This function is used to release the standby status with the WaitIntEvent method at the timing of the user program, not the interrupt event of the Simple Motion board.

```
unsigned long SetIntEvent(  
    unsigned long intNumber  
);
```

## Detailed description

### Parameter

Argument	Description
intNumber [in]	Interrupt factor No. (9 to 64)

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_INT_NOT_START_DRIVER	The interrupt driver is stopped. Call the StartInterrupt method.
MMERR_WIN_SET_EVENT	An error occurred in the SetEvent function (Win32API). Call the GetLastError function of Win32API and confirm the error details.

### Point

- When the interrupt standby status is released by calling this method, an error occurs in the WaitIntEvent method.
- The interrupt factor No. (1 to 8) is used inside the API library therefore it cannot be used by the user program.
- When using interrupt factor No. (9 to 64), the interrupt parameter must be set by the SetInterruptParameter method beforehand.
- The operation of the axis class SetIntEvent method is the same as the operation of this method.

### Supported version

API version	Software version
1.00	01

### Reference

ResetIntEvent (  Page 50 MMC\_Controller::ResetIntEvent)

WaitIntEvent (  Page 52 MMC\_Controller::WaitIntEvent)

# MMC\_Controller::WaitIntEvent

Waits until the interrupt event is in a signaled state. This function is used to wait for the interrupt from the Simple Motion board for the designated interrupt factor.

```
unsigned long WaitIntEvent(  
    unsigned long intNumber,  
    unsigned long timeout  
);
```

## Detailed description

### Parameter

Argument	Description
intNumber [in]	Interrupt factor No. (9 to 64)
timeout [in]	Timeout time [ms](0 to 65535)

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_INT_NOT_START_DRIVER	The interrupt driver is stopped. Call the StartInterrupt method.
MMERR_WIN_WAIT_FOR_SINGLE_OBJECT	An error occurred in the WaitForSingleObject function (Win32API). Call the GetLastError function of Win32API and confirm the error details.
MMERR_INT_TERMINATE_DRIVER	The EndInterrupt method was called while the interrupt for the designated event factor was being confirmed.
MMERR_INT_TERMINATE_NOTIFY_EVENT	An error occurred in the interrupt event notification thread while the interrupt for the designated event factor was being confirmed.
MMERR_TIMEOUT_01	While the interrupt for the designated event factor was being waited, the set timeout time elapsed.
MMERR_INT_SET_HOST_APPLICATION_EVENT	While the interrupt for the designated event factor was being waited, a method which releases the standby status was called from the user program.

### Point

- When MMC\_INFINITE is designated as the timeout time, timeout is not checked. Instead, this function infinitely waits until the interrupt event occurs.
- The interrupt factor No. (1 to 8) is used inside the API library therefore it cannot be used by the user program.
- When using interrupt factor No. (9 to 64), the interrupt parameter must be set by the SetInterruptParameter method beforehand.
- The operation of the axis class WaitIntEvent method is the same as the operation of this method.

### Supported version

API library	Software version
1.00	01

### Reference

ResetIntEvent (  Page 50 MMC\_Controller::ResetIntEvent)

SetIntEvent (  Page 51 MMC\_Controller::SetIntEvent)

# MMC\_Controller::EnableDMA

Enables DMA transmission.

```
unsigned long EnableDMA(  
    void  
);
```

## Detailed description

### Parameter

None.

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_DMA_NOT_START_DRIVER	The DMA transmission driver is stopped. Call the StartDMA method.
MMERR_DMA_INVALID_ADDRESS	The DMA destination address is incorrect.
MMERR_TIMEOUT_01	The timeout time (1 second) has elapsed.

### Point

- When using the DMA transmission function, enable DMA transmission by the procedure below. If this method is not called by the following procedure, it returns MMERR\_DMA\_INVALID\_ADDRESS.

1. Call the StartDMA method of the device driver class.
2. Call the SetUserProgramReady method of the controller class and turn ON user program READY signal [Y0].
3. Call the EnableDMA method of the controller class.

- This method writes "Enabled" to "[Cd.1150] DMA transmission enable request".

### Supported version

API version	Software version
1.00	01

### Reference

DisableDMA ([🔗](#) Page 54 MMC\_Controller::DisableDMA)

# MMC\_Controller::DisableDMA

Disables DMA transmission.

```
unsigned long DisableDMA(  
    void  
);
```

## Detailed description

### Parameter

None.

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_TIMEOUT_01	The timeout time (1 second) has elapsed.

### Point

- Before calling the EndDMA method of the device driver class, call this method.
- This method writes "Disabled" to "[Cd.1150] DMA transmission enable request".

### Supported version

API version	Software version
1.00	01

### Reference

EnableDMA ([↩](#) Page 53 MMC\_Controller::EnableDMA)



# MMC\_Controller::ReadRemoteBufferMemory

Reads data from the buffer memory in the remote device station.

```
unsigned long ReadRemoteBufferMemory (  
    unsigned char netno,  
    unsigned char stno,  
    unsigned long offset,  
    unsigned short size,  
    void *data,  
    unsigned long timeout  
);
```

3

## Detailed description

### Parameter

Argument	Description
netno [in]	Network No. of the remote device station (0 to 239)
stno [in]	Station No. of the remote device station (0 to 255)
offset [in]	Start address of the remote device station buffer memory to be read
size [in]	Byte size of the read data (2 to 480)
data [out]	Pointer to the read data
timeout [in]	Timeout time [ms] (0 to 65535)

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_TIMEOUT_0□ □=1 to 9: Timeout location	The timeout time has elapsed.
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_WIN_WAIT_FOR_SINGLE_OBJECT	An error occurred in the WaitForSingleObject function (Win32API). Call the GetLastError function of Win32API and confirm the error details.
MMERR_WIN_RELEASE_SEMAPHORE	An error occurred in the ReleaseSemaphore function (Win32API). Call the GetLastError function of Win32API and confirm the error details.
MMERR_SLMP_SEND_RECEIVE_ERROR	An error occurred during SLMP communication. Since an end code of the SLMP response message is stored to the details error code, refer to the following to check the error contents and take actions. <a href="#">MELSEC iQ-R CC-Link IE Field Network User's Manual (Application)</a>
MMERR_SLMP_COMMUNICATION_RUNNING	A SLMP communication process is being executed. Execute again after other SLMP communication processes are completed.

### Point

- Specify the byte size of read data in even number.
- When MMC\_INFINITE is designated as the timeout time, timeout is not checked. Instead, this function infinitely waits for completion of the SLMP communication process.
- It is recommended to set 500 ms or more for the timeout time.

### Supported version

API version	Software version
1.30	03

### Reference

WriteRemoteBufferMemory (  Page 56 MMC\_Controller::WriteRemoteBufferMemory )

# MMC\_Controller::WriteRemoteBufferMemory

Writes data to the buffer memory in the remote device station.

```
unsigned long WriteRemoteBufferMemory(  
    unsigned char netno,  
    unsigned char stno,  
    unsigned long offset,  
    unsigned short size,  
    void *data,  
    unsigned long timeout  
);
```

## Detailed description

### Parameter

Argument	Description
netno [in]	Network No. of the remote device station (0 to 239)
stno [in]	Station No. of the remote device station (0 to 255)
offset [in]	Start address of the remote device station buffer memory to be written
size [in]	Byte size of the write data (2 to 480)
data [in]	Pointer to the write data
timeout [in]	Timeout time [ms] (0 to 65535)

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_TIMEOUT_0□ □=1 to 9: Timeout location	The timeout time has elapsed.
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_WIN_WAIT_FOR_SINGLE_OBJECT	An error occurred in the WaitForSingleObject function (Win32API). Call the GetLastError function of Win32API and confirm the error details.
MMERR_WIN_RELEASE_SEMAPHORE	An error occurred in the ReleaseSemaphore function (Win32API). Call the GetLastError function of Win32API and confirm the error details.
MMERR_SLMP_SEND_RECEIVE_ERROR	An error occurred during SLMP communication. Since an end code of the SLMP response message is stored to the details error code, refer to the following to check the error contents and take actions. <a href="#">MELSEC iQ-R CC-Link IE Field Network User's Manual (Application)</a>
MMERR_SLMP_COMMUNICATION_RUNNING	A SLMP communication process is being executed. Execute again after other SLMP communication processes are completed.

### Point

- Specify the byte size of the write data in even number.
- When MMC\_INFINITE is designated as the timeout time, timeout is not checked. Instead, this function infinitely waits for completion of the SLMP communication process.
- It is recommended to set 500 ms or more for the timeout time.

### Supported version

API version	Software version
1.30	03

### Reference

ReadRemoteBufferMemory (  Page 55 MMC\_Controller::ReadRemoteBufferMemory)

## 3.3 MMC\_Axis Class

This class controls the servo amplifier axis.

```
class MMC_Axis: public MMC_Slave
```

### Member

#### Public method

Category	Name	Description	Reference
Positioning data method	SetPositioningData	Sets the positioning data.	Page 59 MMC_Axis::SetPositioningData
	SetBlockStartData	Sets the block start data.	Page 60 MMC_Axis::SetBlockStartData
	SetBlockConditionData	Sets the condition data used by block start.	Page 61 MMC_Axis::SetBlockConditionData
	GetPositioningData	Gets the positioning data.	Page 62 MMC_Axis::GetPositioningData
	GetBlockStartData	Gets the block start data.	Page 63 MMC_Axis::GetBlockStartData
	GetBlockConditionData	Gets the condition data used by block start.	Page 64 MMC_Axis::GetBlockConditionData
Operation method	StartPositioning	Starts positioning control.	Page 65 MMC_Axis::StartPositioning
	StartBlockPositioning	Starts advanced positioning control.	Page 66 MMC_Axis::StartBlockPositioning
	StopPositioning	Stops axis.	Page 67 MMC_Axis::StopPositioning
	RestartPositioning	Restarts stopped axis.	Page 68 MMC_Axis::RestartPositioning
	WaitPositioningDone	Waits until completion of positioning control.	Page 69 MMC_Axis::WaitPositioningDone
	ResetPositioningDoneIntEvent	Sets the positioning complete interrupt event to a nonsignaled state.	Page 70 MMC_Axis::ResetPositioningDoneIntEvent
	SetPositioningDoneIntEvent	Sets the positioning complete interrupt event to a signaled state.	Page 71 MMC_Axis::SetPositioningDoneIntEvent
	WaitPositioningDoneIntEvent	Waits until the positioning complete interrupt event is in a signaled state.	Page 72 MMC_Axis::WaitPositioningDoneIntEvent
	StartJog	Starts JOG operation.	Page 73 MMC_Axis::StartJog
	StopJog	Stops JOG operation.	Page 74 MMC_Axis::StopJog
	StartInching	Starts inching operation.	Page 75 MMC_Axis::StartInching
	EnableMPG	Enables manual pulse generator operation.	Page 76 MMC_Axis::EnableMPG
	DisableMPG	Disables manual pulse generator operation.	Page 77 MMC_Axis::DisableMPG
	ChangeControlMode	Changes control mode.	Page 78 MMC_Axis::ChangeControlMode
Change method	ChangeSpeed	Changes speed and acceleration/deceleration time.	Page 79 MMC_Axis::ChangeSpeed
	ChangePosition	Changes target position and command speed.	Page 80 MMC_Axis::ChangePosition
Error method	ResetError	Performs error reset.	Page 81 MMC_Axis::ResetError
Interrupt method	SetInterruptParameter	Sets the interrupt parameter.	Page 82 MMC_Axis::SetInterruptParameter
	ResetIntEvent	Sets the interrupt event to a nonsignaled state.	Page 84 MMC_Axis::ResetIntEvent
	SetIntEvent	Sets the interrupt event to a signaled state.	Page 85 MMC_Axis::SetIntEvent
	WaitIntEvent	Waits until the interrupt event is in a signaled state.	Page 86 MMC_Axis::WaitIntEvent

#### Public data member

Variable name	Data type	Initial value	Description
LastErrorNumber	unsigned long	MMERR_NONE	Latest error code
LastDetailErrorNumber	unsigned long	MMERR_NONE	Latest detail error code
StationNumber	unsigned long	—	Station No. (1 to 16)

## Label

Refer to label list for labels. (☞ Page 203 LABEL LIST)

## Point

Get axis class objects with the GetAxis method of the controller class.

## Inheritance hierarchy

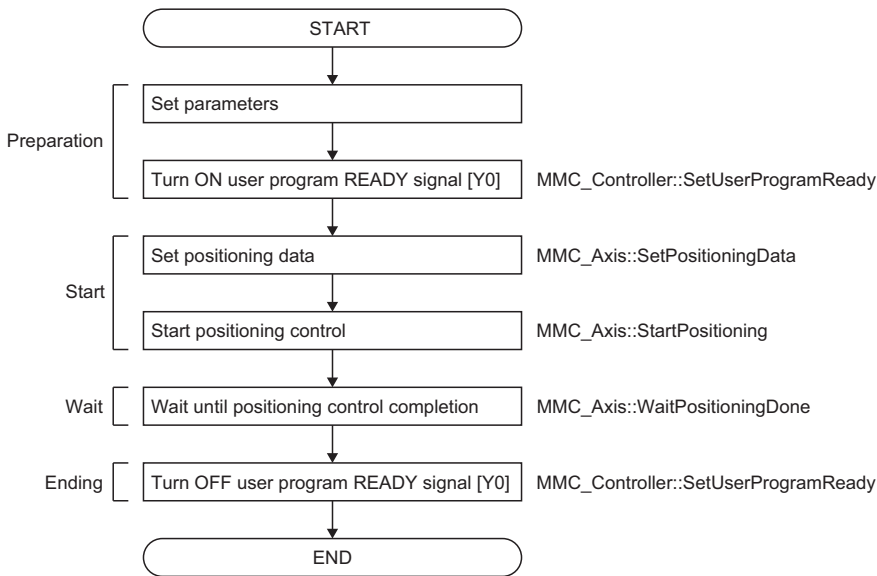
MMC\_NetworkModule

MMC\_Slave

MMC\_Axis

## Execution procedure

The procedure for starting positioning control and waiting for positioning completion by polling is given as an example.



# MMC\_Axis::SetPositioningData

Sets the positioning data ([Da.1] to [Da.10], [Da.20] to [Da.22], [Da.27] to [Da.29]).

```
unsigned long SetPositioningData(  
    unsigned long dataNo,  
    const MMST_PositioningData &positioningData  
);
```

## Detailed description

### Parameter

Argument	Description
dataNo [in]	Positioning data No. (1 to 600)
positioningData [in]	Positioning structure

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.


### Point

When a value outside the set range is set to the member variable of a structure, the No. of the member variable (1 or more) is stored to the details error code.

### Supported version

API version	Software version
1.00	01

### Reference

GetPositioningData (  Page 62 MMC\_Axis::GetPositioningData)

# MMC\_Axis::SetBlockStartData

Sets the block start data.

```
unsigned long SetBlockStartData(  
    unsigned long blockNo,  
    unsigned long pointNo,  
    const MMST_BlockStartData &blockStartData  
);
```

## Detailed description

### Parameter

Argument	Description
blockNo [in]	Block No.(0 to 4)
pointNo [in]	Point No.(1 to 50)
blockStartData [in]	Block start data structure

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.


### Point

When a value outside the set range is set to the member variable of a structure, the No. of the member variable (1 or more) is stored to the details error code.

### Supported version

API version	Software version
1.00	01

### Reference

GetBlockStartData (  Page 63 MMC\_Axis::GetBlockStartData)

# MMC\_Axis::SetBlockConditionData

Sets the condition data used by block start.

```
unsigned long SetBlockConditionData(  
    unsigned long blockNo,  
    unsigned long conditionNo,  
    const MMST_BlockConditionData &blockConditionData  
);
```

## Detailed description

### Parameter

Argument	Description
blockNo [in]	Block No. (0 to 4)
conditionNo [in]	Condition data No.(1 to 10)
blockConditionData [in]	Condition data structure

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.


### Point

When a value outside the set range is set to the member variable of a structure, the No. of the member variable (1 or more) is stored to the details error code.

### Supported version

API version	Software version
1.00	01

### Reference

GetBlockConditionData (  Page 64 MMC\_Axis::GetBlockConditionData)

# MMC\_Axis::GetPositioningData

Gets the positioning data ([Da.1] to [Da.10], [Da.20] to [Da.22], [Da.27] to [Da.29]).

```
Unsigned long GetPositioningData(  
    unsigned long dataNo,  
    MMST_PositioningData *positioningData  
);
```

## Detailed description

### Parameter

Argument	Description
dataNo [in]	Positioning data No.(1 to 600)
positioningData [in]	Pointer to positioning data structure

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.

### Point

None.

### Supported version

API version	Software version
1.00	01

### Reference

SetPositioningData ([↩](#) Page 59 MMC\_Axis::SetPositioningData)



# MMC\_Axis::GetBlockStartData

Gets the block start data.

```
unsigned long GetBlockStartData(  
    unsigned long blockNo,  
    unsigned long pointNo,  
    MMST_BlockStartData *blockStartData  
);
```

## Detailed description

### Parameter

Argument	Description
blockNo [in]	Block No. (0 to 4)
pointNo [in]	Point No.(1 to 50)
blockStartData [in]	Pointer to block start data structure

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.


### Point

None.

### Supported version

API version	Software version
1.00	01

### Reference

SetBlockStartData (  Page 60 MMC\_Axis::SetBlockStartData)

# MMC\_Axis::GetBlockConditionData

Gets the condition data used by block start.

```
unsigned long GetBlockConditionData(  
    unsigned long blockNo,  
    unsigned long conditionNo,  
    MMST_BlockConditionData *blockConditionData  
);
```

## Detailed description

### Parameter

Argument	Description
blockNo [in]	Block No. (0 to 4)
conditionNo [in]	Condition data No. (1 to 10)
blockConditionData [in]	Pointer to condition data structure

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.


### Point

None.

### Supported version

API version	Software version
1.00	01

### Reference

SetBlockConditionData (  Page 61 MMC\_Axis::SetBlockConditionData)

# MMC\_Axis::StartPositioning

Starts positioning control.

```
unsigned long StartPositioning(  
    unsigned short startNo  
);
```

## Detailed description

### Parameter

Argument	Description
startNo [in]	Positioning start No.*1


\*1 Specify the following values for the positioning start No.

Value	Description
1 to 600	Positioning data No.
MMC_STNO_HOMING	Machine home position return
MMC_STNO_FAST_HOMING	High speed home position return
MMC_STNO_CHANGE_VALUE	Change current value
MMC_STNO_MULTIPLE_AXES	Start multiple axes simultaneously

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_TIMEOUT_01	The timeout time (1 second) has elapsed.
MMERR_CONDITION_READY_SIGNAL_OFF	The READY signal [X0] is OFF.
MMERR_CONDITION_BUSY_ON	BUSY signal [X10 to X1F] are ON.
MMERR_CONDITION_AXIS_ERROR_DETECTION	Error detection ("[Md.31] Status": b13) is ON.

### Point


- After writing ON to positioning start signal [Y10 to Y1F], this method returns control immediately.
- When the positioning start signal [Y10 to Y1F] is already ON, this method writes OFF to positioning start signal [Y10 to Y1F] and waits until start complete ("[Md.31] Status": b14) turns OFF before writing ON to positioning start signal [Y10 to Y1F].
- When using machine home position return, high speed home position return, change current value, and start multiple axes simultaneously for the positioning start No., the Simple Motion board function of the same name is used for starting positioning control. Refer to the following for details.  
 Simple Motion Board User's Manual (Application)
- This method cannot be used in conjunction with the pre-read start function. When using the pre-read start function, use a label when setting.


### Supported version

API version	Software version
1.00	01

### Reference

StartBlockPositioning ( Page 66 MMC\_Axis::StartBlockPositioning)

RestartPositioning ( Page 68 MMC\_Axis::RestartPositioning)

WaitPositioningDone ( Page 69 MMC\_Axis::WaitPositioningDone)

# MMC\_Axis::StartBlockPositioning

Starts advanced positioning control.

```
unsigned long StartBlockPositioning(  
    unsigned short startBlock,  
    unsigned short startPointNo  
);
```

## Detailed description

### Parameter

Argument	Description
startBlock [in]	Start block (0 to 4)
startPointNo [in]	Positioning start point No. (1 to 50)

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_TIMEOUT_01	The timeout time (1 second) has elapsed.
MMERR_CONDITION_READY_SIGNAL_OFF	The READY signal [X0] is OFF.
MMERR_CONDITION_BUSY_ON	BUSY signal [X10 to X1F] are ON.
MMERR_CONDITION_AXIS_ERROR_DETECTION	Error detection ("[Md.31] Status": b13) is ON.


### Point


- After writing ON to positioning start signal [Y10 to Y1F], this method returns control immediately.
- When the positioning start signal [Y10 to Y1F] is already ON, this method writes OFF to positioning start signal [Y10 to Y1F] and waits until start complete ("[Md.31] Status": b14) turns OFF before writing ON to positioning start signal [Y10 to Y1F].
- This method cannot be used in conjunction with the pre-read start function. When using the pre-read start function, use a label when setting.


### Supported version

API version	Software version
1.00	01

### Reference

StartPositioning (  Page 65 MMC\_Axis::StartPositioning )

RestartPositioning (  Page 68 MMC\_Axis::RestartPositioning )

WaitPositioningDone (  Page 69 MMC\_Axis::WaitPositioningDone )

# MMC\_Axis::StopPositioning

Stops axis.

```
unsigned long StopPositioning(  
    void  
);
```

## Detailed description

### Parameter

None.

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.

### Point

- This method uses "[Cd.180] Axis stop". In addition to positioning control, manual controls (JOG operation etc.), and expansion controls (speed-torque control etc.) can also use this method.
- This method returns control immediately after writing "1" to "[Cd.180] Axis stop".
- After calling this method, call the WaitPositioningDone method or WaitPositioningDoneIntEvent method. By calling the WaitPositioningDone method or WaitPositioningDoneIntEvent method, "0" is written to "[Cd.180] Axis stop". If the methods are not called, an error "Stop signal ON at start" (error code: 1908H) occurs at the next start.

### Supported version

API version	Software version
1.00	01

### Reference

StartPositioning ([↩](#) Page 65 MMC\_Axis::StartPositioning)

StartBlockPositioning ([↩](#) Page 66 MMC\_Axis::StartBlockPositioning)

# MMC\_Axis::RestartPositioning

Restarts stopped axis.

```
unsigned long RestartPositioning(  
    void  
);
```

## Detailed description

### Parameter

None.

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_CONDITION_READY_SIGNAL_OFF	The READY signal [X0] is OFF.
MMERR_CONDITION_AXIS_OPERATION_STATUS	"[Md.26] Axis operation status" is not stopped.
MMERR_CONDITION_AXIS_ERROR_DETECTION	Error detection ("[Md.31] Status": b13) is ON.

### Point

- This method returns control immediately after writing "1" to "[Cd.6] Restart command".
- If calling this method when the target value has already been reached, wait with WaitPositioningDone.

### Supported version

API version	Software version
1.00	01

### Reference

StartPositioning ([↩](#) Page 65 MMC\_Axis::StartPositioning)

StartBlockPositioning ([↩](#) Page 66 MMC\_Axis::StartBlockPositioning)

WaitPositioningDone ([↩](#) Page 69 MMC\_Axis::WaitPositioningDone)

# MMC\_Axis::WaitPositioningDone

Waits until completion of positioning control.

```
unsigned long WaitPositioningDone(  
    unsigned long checkMethod,  
    unsigned long timeout  
);
```

## Detailed description

### Parameter

Argument	Description
checkMethod [in]	Positioning control complete judge condition* <sup>1</sup>
timeout [in]	Timeout time [ms] (0 to 65535)

\*1 Specify the following values for positioning control complete judge condition.

Value	Description
MMC_POSITIONING_DONE_BUSY	Positioning complete when BUSY signal [X10 to X1F] turns OFF.
MMC_POSITIONING_DONE_INP	Positioning complete when BUSY signal [X10 to X1F] turns OFF and "[Md.1190] Controller in-position flag" turns ON.

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_TIMEOUT_01	The set timeout time has elapsed.
MMERR_AXIS_ERROR	An axis error occurred. Check "[Md.23] Axis error No."


### Point

- This method waits inside the method until positioning control is complete.
- When "0" is set to the timeout time, the method determines positioning control to be complete, and control is returned immediately. When positioning control is not complete, it returns MMERR\_TIMEOUT\_01.
- When MMC\_INFINITE is designated as the timeout time, timeout is not checked. Instead, this function infinitely waits for positioning completion.


### Supported version

API version	Software version
1.00	01

### Reference

StartPositioning (  Page 65 MMC\_Axis::StartPositioning)

StartBlockPositioning (  Page 66 MMC\_Axis::StartBlockPositioning)

RestartPositioning (  Page 68 MMC\_Axis::RestartPositioning)

## MMC\_Axis::ResetPositioningDoneIntEvent

Sets the positioning complete interrupt event to a nonsignaled state. This function is used if positioning complete interrupt events occurring prior to calling the WaitPositioningDoneIntEvent method are to be disabled.

```
unsigned long ResetPositioningDoneIntEvent(  
    void  
);
```

### Detailed description

#### Parameter

None.

#### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_INT_NOT_START_DRIVER	The interrupt driver is stopped. Call the StartInterrupt method.
MMERR_WIN_RESET_EVENT	An error occurred in the ResetEvent function (Win32API). Call the GetLastError function of Win32API and confirm the error details.


#### Point


None.

#### Supported version

API version	Software version
1.00	01

#### Reference

SetPositioningDoneIntEvent (  Page 71 MMC\_Axis::SetPositioningDoneIntEvent)

WaitPositioningDoneIntEvent (  Page 72 MMC\_Axis::WaitPositioningDoneIntEvent)



# MMC\_Axis::SetPositioningDoneIntEvent

Sets the positioning complete interrupt event to a signaled state. This function is used to release the standby status with the WaitPositioningDoneIntEvent method at the timing of the user program, not the interrupt event of the Simple Motion board.

```
unsigned long SetPositioningDoneIntEvent(  
    void  
);
```

## Detailed description

### Parameter

None.

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_INT_NOT_START_DRIVER	The interrupt driver is stopped. Call the StartInterrupt method.
MMERR_WIN_SET_EVENT	An error occurred in the SetEvent function (Win32API). Call the GetLastError function of Win32API and confirm the error details.


### Point


When the interrupt standby status is released by calling this method, an error occurs in the WaitPositioningDoneIntEvent method.

### Supported version

API version	Software version
1.00	01

### Reference

ResetPositioningDoneIntEvent (  Page 70 MMC\_Axis::ResetPositioningDoneIntEvent)

WaitPositioningDoneIntEvent (  Page 72 MMC\_Axis::WaitPositioningDoneIntEvent)

# MMC\_Axis::WaitPositioningDoneIntEvent

Waits until the positioning complete interrupt event is in a signaled state. This function is used to wait for the interrupt from the Simple Motion board for the designated interrupt factor.

```
unsigned long WaitPositioningDoneIntEvent(  
    unsigned long checkMethod,  
    unsigned long timeout  
);
```

## Detailed description

### Parameter

Argument	Description
checkMethod [in]	Positioning control complete judge condition <sup>*1</sup>
timeout [in]	Timeout time [ms] (0 to 65535)

\*1 Specify the following values for positioning control complete judge condition.

Value	Description
MMC_POSITIONING_DONE_BUSY	Positioning complete when BUSY signal [X10 to X1F] turns OFF.
MMC_POSITIONING_DONE_INP	Positioning complete when BUSY signal [X10 to X1F] turns OFF and "[Md.1190] Controller in-position flag" turns ON.

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_AXIS_ERROR	An axis error occurred. Check "[Md.23] Axis error No."
MMERR_INT_NOT_START_DRIVER	The interrupt driver is stopped. Call the StartInterrupt method.
MMERR_WIN_WAIT_FOR_SINGLE_OBJECT	An error occurred in the WaitForSingleObject function (Win32API). Call the GetLastError function of Win32API and confirm the error details.
MMERR_INT_TERMINATE_DRIVER	The EndInterrupt method was called while the interrupt for the designated event factor was being confirmed.
MMERR_INT_TERMINATE_NOTIFY_EVENT	An error occurred in the interrupt event notification thread while the interrupt for the designated event factor was being confirmed.
MMERR_TIMEOUT_01	While the interrupt for the designated event factor was being waited, the set timeout time elapsed.
MMERR_INT_SET_HOST_APPLICATION_EVENT	While the interrupt for the designated event factor was being waited, a method which releases the standby status was called from the user program.

### Point


When MMC\_INFINITE is designated as the timeout time, timeout is not checked. Instead, this function infinitely waits until the interrupt event occurs.

### Supported version

API version	Software version
1.00	01

### Reference

ResetPositioningDoneIntEvent (  Page 70 MMC\_Axis::ResetPositioningDoneIntEvent)

SetPositioningDoneIntEvent (  Page 71 MMC\_Axis::SetPositioningDoneIntEvent)

# MMC\_Axis::StartJog

Starts JOG operation.

```
unsigned long StartJog(  
    long speed  
);
```

## Detailed description


### Parameter

Argument	Description
speed [in]	JOG speed

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_CONDITION_READY_SIGNAL_OFF	The READY signal [X0] is OFF.
MMERR_CONDITION_BUSY_ON	BUSY signal [X10 to X1F] are ON.
MMERR_CONDITION_AXIS_ERROR_DETECTION	Error detection ("[Md.31] Status": b13) is ON.


### Point

- This method returns control immediately after turning ON "[Cd.181] Forward run JOG start", or "[Cd.182] Reverse run JOG start".
- For forward rotation, set a positive value to JOG speed. For reverse rotation, set a negative value to JOG speed.
- The setting range for JOG speed differs depending on the unit setting. Refer to the following for details.  
 Simple Motion Board User's Manual (Application)

### Supported version

API version	Software version
1.00	01

### Reference

StopJog ( Page 74 MMC\_Axis::StopJog)

# MMC\_Axis::StopJog

Stops JOG operation.

```
unsigned long StopJog(  
    void  
);
```

## Detailed description

### ■Parameter

None.

### ■Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.

### ■Point

This method returns control immediately after writing "0" to "[Cd.181] Forward run JOG start" and "[Cd.182] Reverse run JOG start".

### ■Supported version

API version	Software version
1.00	01

### ■Reference

StartJog ([↩](#) Page 73 MMC\_Axis::StartJog)

# MMC\_Axis::StartInching

Starts inching operation.

```
unsigned long StartInching(  
    long movementAmount  
);
```

## Detailed description

### Parameter

Argument	Description
movementAmount [in]	Inching movement amount (-65535 to -1, 1 to 65535)

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_TIMEOUT_01	The timeout time (1 second) has elapsed.
MMERR_AXIS_ERROR	An axis error occurred. Check "[Md.23] Axis error No."
MMERR_CONDITION_READY_SIGNAL_OFF	The READY signal [X0] is OFF.
MMERR_CONDITION_BUSY_ON	BUSY signal [X10 to X1F] are ON.
MMERR_CONDITION_AXIS_ERROR_DETECTION	Error detection ("[Md.31] Status": b13) is ON.

### Point

- After turning ON "[Cd.181] Forward run JOG start", or "[Cd.182] Reverse run JOG start", this method waits inside the method until inching operation is complete.
- For forward rotation, set a positive value to inching movement amount. For reverse rotation, set a negative value to inching movement amount.

### Supported version

API version	Software version
1.00	01

### Reference

None.

# MMC\_Axis::EnableMPG

Enables manual pulse generator operation.

```
unsigned long EnableMPG(  
    void  
);
```

## Detailed description

### ■Parameter

None.

### ■Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_TIMEOUT_01	The timeout time (1 second) has elapsed.
MMERR_AXIS_ERROR	An axis error occurred. Check "[Md.23] Axis error No."
MMERR_CONDITION_READY_SIGNAL_OFF	The READY signal [X0] is OFF.
MMERR_CONDITION_MPG_FLAG_ON	The "[Cd.21] Manual pulse generator enable flag" is ON.
MMERR_CONDITION_BUSY_ON	BUSY signal [X10 to X1F] are ON.
MMERR_CONDITION_AXIS_ERROR_DETECTION	Error detection ("[Md.31] Status": b13) is ON.

### ■Point

This method sets "1" to "[Cd.21] Manual pulse generator enable flag", and waits inside the method until manual pulse generator operation is enabled.

### ■Supported version

API version	Software version
1.00	01

### ■Reference

DisableMPG ([↩](#) Page 77 MMC\_Axis::DisableMPG)

# MMC\_Axis::DisableMPG

Disables manual pulse generator operation.

```
unsigned long DisableMPG(  
    void  
);
```

## Detailed description

### ■Parameter

None.

### ■Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_TIMEOUT_01	The timeout time (1 second) has elapsed.


### ■Point

This method sets "0" to "[Cd.21] Manual pulse generator enable flag", and waits inside the method until manual pulse generator operation is disabled.

### ■Supported version

API version	Software version
1.00	01

### ■Reference

EnableMPG ( Page 76 MMC\_Axis::EnableMPG)

# MMC\_Axis::ChangeControlMode

Changes control mode.

```
unsigned long ChangeControlMode(  
    unsigned short controlMode,  
    unsigned long timeout  
);
```

## Detailed description

### Parameter

Argument	Description
controlMode [in]	Control mode* <sup>1</sup>
timeout [in]	Timeout time [ms] (0 to 65535)

\*1 Specify the following values for control mode.

Value	Description
MMC_CONTROL_POSITION	Position control mode
MMC_CONTROL_SPEED	Speed control mode
MMC_CONTROL_TORQUE	Torque control mode

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_TIMEOUT_01	The set timeout time has elapsed.
MMERR_AXIS_ERROR	An axis error occurred. Check "[Md.23] Axis error No."
MMERR_CONDITION_BUSY_ON	BUSY signal [X10 to X1F] are ON.
MMERR_CONDITION_AXIS_ERROR_DETECTION	Error detection ("[Md.31] Status": b13) is ON.

### Point

- After setting "1" to "[Cd.138] Control mode switching request", this method waits inside the method until the value changes to "0".
- When MMC\_INFINITE is designated as the timeout time, timeout is not checked. Instead, this function infinitely waits for control mode switch.
- In speed-torque control the following labels are used.
  - [Cd.140] Command speed at speed control mode
  - [Cd.141] Acceleration time at speed control mode
  - [Cd.142] Deceleration time at speed control mode
  - [Cd.143] Command torque at torque control mode
  - [Cd.144] Torque time constant at torque control mode (Driving mode)
  - [Cd.145] Torque time constant at torque control mode (Regenerative mode)
  - [Cd.146] Speed limit value at torque control mode

### Supported version

API version	Software version
1.00	01

### Reference

None.



# MMC\_Axis::ChangeSpeed

Changes speed and acceleration/deceleration time.

```
unsigned long ChangeSpeed(  
    unsigned long speed,  
    unsigned long accTime,  
    unsigned long decTime  
);
```

## Detailed description

### Parameter

Argument	Description
speed [in]	[Cd.14] New speed value
accTime [in]	[Cd.10] New acceleration time value
decTime [in]	[Cd.11] New deceleration time value

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_TIMEOUT_01	The timeout time (1 second) has elapsed.
MMERR_AXIS_ERROR	An axis error occurred. Check "[Md.23] Axis error No."
MMERR_CONDITION_BUSY_OFF	BUSY signal [X10 to X1F] are OFF.

### Point

- After setting "1" to "[Cd.15] Speed change request", this method waits inside the method until the value changes to "0".
- When "0" is set to new acceleration time value and new deceleration time value, acceleration/deceleration time is not changed.
- Refer to the following for setting values (unit, setting range) of "[Cd.14] New speed value", "[Cd.10] New acceleration time value", and "[Cd.11] New deceleration time value".

 Simple Motion Board User's Manual (Application)

### Supported version

API version	Software version
1.00	01

### Reference

None.

# MMC\_Axis::ChangePosition

Changes target position and command speed.

```
unsigned long ChangePosition(  
    long position,  
    unsigned long speed,  
    unsigned long timeout  
);
```

## Detailed description


### Parameter

Argument	Description
position [in]	[Cd.27] Target position change value (New address)
speed [in]	[Cd.28] Target position change value (New speed)
timeout [in]	Timeout time [ms] (0 to 65535)

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_TIMEOUT_01	The set timeout time has elapsed.
MMERR_AXIS_ERROR	An axis error occurred. Check "[Md.23] Axis error No."
MMERR_CONDITION_BUSY_OFF	BUSY signal [X10 to X1F] are OFF.

### Point

- After setting "1" to "[Cd.29] Target position change request flag", this method waits inside the method until the value changes to "0".
- When "0" is set to new speed, command speed is not changed.
- Refer to the following for setting values (unit, setting range) of "[Cd.27] Target position change value (New address)", and "[Cd.28] Target position change value (New speed)".  
 Simple Motion Board User's Manual (Application)

### Supported version

API version	Software version
1.00	01

### Reference

None.

# MMC\_Axis::ResetError

Performs error reset.

```
unsigned long ResetError(  
    void  
);
```

## Detailed description

### ■Parameter

None.

### ■Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_TIMEOUT_01	The timeout time (1 second) has elapsed.

### ■Point

After setting "1" to "[Cd.5] Axis error reset", this method waits inside the method until the value changes to "0".

### ■Supported version

API version	Software version
1.00	01

### ■Reference

None.

# MMC\_Axis::SetInterruptParameter

Sets the interrupt parameter.

```
unsigned long SetInterruptParameter(
    unsigned long intNumber,
    MMC_Label &label,
    const MMST_InterruptParameter &interruptParameter
);
```

## Detailed description

### Parameter

Argument	Description
intNumber [in]	Interrupt factor No. (9 to 64)
label [in]	Interrupt factor label
interruptParameter [in]	Interrupt parameter structure
interruptParameter.IntCondition [in]	Interrupt condition <sup>*1</sup>
interruptParameter.IntSensitivity [in]	Interrupt detection timing <sup>*2</sup>
interruptParameter.IntOutputDataSeting [in]	Factor details output setting <sup>*3</sup>
interruptParameter.IntTimeSetting [in]	Condition for condition completion continue <sup>*4</sup>
interruptParameter.IntJudgeValue1 [in]	Interrupt condition judge value 1
interruptParameter.IntJudgeValue2 [in]	Interrupt condition judge value 2
interruptParameter.IntTime [in]	Time for condition completion [ms]

\*1 Specify the following values for interrupt condition.

Value	Description
MMC_INT_CONDITION_DISABLE	Do not detect
MMC_INT_CONDITION_EQUAL	Equal to interrupt condition judge value 1
MMC_INT_CONDITION_NOT_EQUAL	Not equal to interrupt condition judge value 1
MMC_INT_CONDITION_GREATER	Larger than interrupt condition judge value 1
MMC_INT_CONDITION_LESS	Smaller than interrupt condition judge value 1
MMC_INT_CONDITION_WITHIN	Within the range of interrupt condition judge value 1, 2
MMC_INT_CONDITION_OUTSIDE	Outside the range of interrupt condition judge value 1, 2
MMC_INT_CONDITION_CHANGE_BIT	The value masked by interrupt condition judge value 1 is different from the previous value
MMC_INT_CONDITION_ALL_BIT_ON	All points of the bit masked by interrupt condition judge value 1 are ON.
MMC_INT_CONDITION_ALL_BIT_OFF	All points of the bit masked by interrupt condition judge value 1 are OFF.
MMC_INT_CONDITION_BIT_ON	A point of the bit masked by interrupt condition judge value 1 is ON.
MMC_INT_CONDITION_BIT_OFF	A point of the bit masked by interrupt condition judge value 1 is OFF.

\*2 Specify the following values for interrupt detection timing.

Value	Description
MMC_INT_SENSITIVITY_EDGE	Edge detection
MMC_INT_SENSITIVITY_LEVEL	Level detection

\*3 Specify the following values for factor details output setting.

Value	Description
MMC_INT_OUTPUT_DATA_SETTING_FIRST_DATA	First data
MMC_INT_OUTPUT_DATA_SETTING_LAST_DATA	Last data

\*4 Specify the following values for condition for condition completion continue.

Value	Description
MMC_INT_TIME_SETTING_RESTART	Start a new count at another interrupt condition completion
MMC_INT_TIME_SETTING_CONTINUE	Continue count

## Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.

## Point

- When a bool label is specified to an interrupt factor label, setting interrupt condition judge value 1 is not required because the API library automatically sets the mask value.
- When a bool label is specified to an interrupt factor label, specify one of the following to the interrupt condition.
  - MMC\_INT\_CONDITION\_CHANGE\_BIT
  - MMC\_INT\_CONDITION\_ALL\_BIT\_ON
  - MMC\_INT\_CONDITION\_ALL\_BIT\_OFF
  - MMC\_INT\_CONDITION\_BIT\_ON
  - MMC\_INT\_CONDITION\_BIT\_OFF
- The interrupt factor No. (1 to 8) is used inside the API library therefore it cannot be used by the user program.
- The interrupt parameter is enabled by turning OFF→ON the user program READY signal [Y0].
- When a value outside the set range is set to the member variable of a structure, the No. of the member variable (1 or more) is stored to the details error code.
- When MMC\_NULL\_LABEL is specified to label, the interrupt parameter of the interrupt No. specified by IntNumber is initialized.
- The operation of the controller class SetInterruptParameter method is the same as the operation of this method.

## Supported version

API version	Software version
1.00	01

## Reference

None.

# MMC\_Axis::ResetIntEvent

Sets the interrupt event to a nonsignaled state. This function is used if interrupt events occurring prior to calling the WaitIntEvent method are to be disabled.

```
unsigned long ResetIntEvent(  
    unsigned long intNumber  
);
```

## Detailed description

### Parameter

Argument	Value
intNumber [in]	Interrupt factor No.*1

\*1 Specify the following values for interrupt factor No.

Value	Description
MMC_INTFACTOR_ERROR	Axis error occurrence
MMC_INTFACTOR_WARNING	Axis warning occurrence
9 to 64	Interrupt factor

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_INT_NOT_START_DRIVER	The interrupt driver is stopped. Call the StartInterrupt method.
MMERR_WIN_RESET_EVENT	An error occurred in the ResetEvent function (Win32API). Call the GetLastError function of Win32API and confirm the error details.


### Point


- The interrupt factor No. (1 to 8) is used inside the API library therefore it cannot be used by the user program.
- When using interrupt factor No. (9 to 64), the interrupt parameter must be set by the SetInterruptParameter method beforehand.
- The operation of the controller class ResetIntEvent method is the same as the operation of this method.

### Supported version

API version	Software version
1.00	01

### Reference

SetIntEvent (  Page 85 MMC\_Axis::SetIntEvent)

WaitIntEvent (  Page 86 MMC\_Axis::WaitIntEvent)

# MMC\_Axis::SetIntEvent

Sets the interrupt event to a signaled state. This function is used to release the standby status with the WaitIntEvent method at the timing of the user program, not the interrupt event of the Simple Motion board.

```
unsigned long SetIntEvent(  
    unsigned long intNumber  
);
```

## Detailed description

### Parameter

Argument	Description
intNumber [in]	Interrupt factor No.*1

\*1 Specify the following values for interrupt factor No.

Value	Description
MMC_INTFACTOR_ERROR	Axis error occurrence
MMC_INTFACTOR_WARNING	Axis warning occurrence
9 to 64	Interrupt factor

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_INT_NOT_START_DRIVER	The interrupt driver is stopped. Call the StartInterrupt method.
MMERR_WIN_SET_EVENT	An error occurred in the SetEvent function (Win32API). Call the GetLastError function of Win32API and confirm the error details.


### Point


- When the interrupt standby status is released by calling this method, an error occurs in the WaitIntEvent method.
- The interrupt factor No. (1 to 8) is used inside the API library therefore it cannot be used by the user program.
- When using interrupt factor No. (9 to 64), the interrupt parameter must be set by the SetInterruptParameter method beforehand.
- The operation of the controller class SetIntEvent method is the same as the operation of this method.

### Supported version

API version	Software version
1.00	01

### Reference

ResetIntEvent (  Page 84 MMC\_Axis::ResetIntEvent)

WaitIntEvent (  Page 86 MMC\_Axis::WaitIntEvent)

# MMC\_Axis::WaitIntEvent

Waits until the interrupt event is in a signaled state. This function is used to wait for the interrupt from the Simple Motion board for the designated interrupt factor.

```
unsigned long WaitIntEvent(  
    unsigned long intNumber,  
    unsigned long timeout  
);
```

## Detailed description

### Parameter

Argument	Description
intNumber [in]	Interrupt factor No. *1
timeout [in]	Timeout time [ms] (0 to 65535)

\*1 Specify the following values for interrupt factor No.

Value	Description
MMC_INTFACTOR_ERROR	Axis error occurrence
MMC_INTFACTOR_WARNING	Axis warning occurrence
9 to 64	Interrupt factor

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_INT_NOT_START_DRIVER	The interrupt driver is stopped. Call the StartInterrupt method.
MMERR_WIN_WAIT_FOR_SINGLE_OBJECT	An error occurred in the WaitForSingleObject function (Win32API). Call the GetLastError function of Win32API and confirm the error details.
MMERR_INT_TERMINATE_DRIVER	The EndInterrupt method was called while the interrupt for the designated event factor was being confirmed.
MMERR_INT_TERMINATE_NOTIFY_EVENT	An error occurred in the interrupt event notification thread while the interrupt for the designated event factor was being confirmed.
MMERR_TIMEOUT_01	While the interrupt for the designated event factor was being waited, the set timeout time elapsed.
MMERR_INT_SET_HOST_APPLICATION_EVENT	While the interrupt for the designated event factor was being waited, a method which releases the standby status was called from the user program.


### Point


- When MMC\_INFINITE is designated as the timeout time, timeout is not checked. Instead, this function infinitely waits until the interrupt event occurs.
- The interrupt factor No. (1 to 8) is used inside the API library therefore it cannot be used by the user program.
- When using interrupt factor No. (9 to 64), the interrupt parameter must be set by the SetInterruptParameter method beforehand.
- When waiting for positioning complete interrupt events (BUSY signal [X10 to X1F] OFF edge, "[Md.1190] Controller in-position flag" ON edge), use the WaitPositioningDoneIntEvent method instead of this method.
- The operation of the controller class WaitIntEvent method is the same as the operation of this method.

### Supported version

API version	Software version
1.00	01

### Reference

ResetIntEvent (  Page 84 MMC\_Axis::ResetIntEvent)

SetIntEvent (  Page 85 MMC\_Axis::SetIntEvent)



# 3.4 MMC\_Io Class

This class controls I/O modules.

```
class MMC_Io: public MMC_Slave
```

## Member

### Public method

There is no public method.

### Public data member

Variable name	Data type	Initial value	Description
LastErrorNumber	unsigned long	MMERR_NONE	Latest error code
LastDetailErrorNumber	unsigned long	MMERR_NONE	Latest detail error code
StationNumber	unsigned long	—	Station No. (1 to 120)

### Label

Variable name	Data type	Description
bRX[]	bool	RX area
bRY[]	bool	RY area
wRX[]	unsigned short	RX area (for word access)
wRY[]	unsigned short	RY area (for word access)
dwRX[]	unsigned long	RX area (for double word access)
dwRY[]	unsigned long	RY area (for double word access)
wRWw[]	unsigned short	RWw area
wRWr[]	unsigned short	RWr area
dwRWw[]	unsigned long	RWw area (for double word access)
dwRWr[]	unsigned long	RWr area (for double word access)
wRWw[].b[]	bool	RWw area (for bit access)
wRWr[].b[]	bool	RWr area (for bit access)

## Point

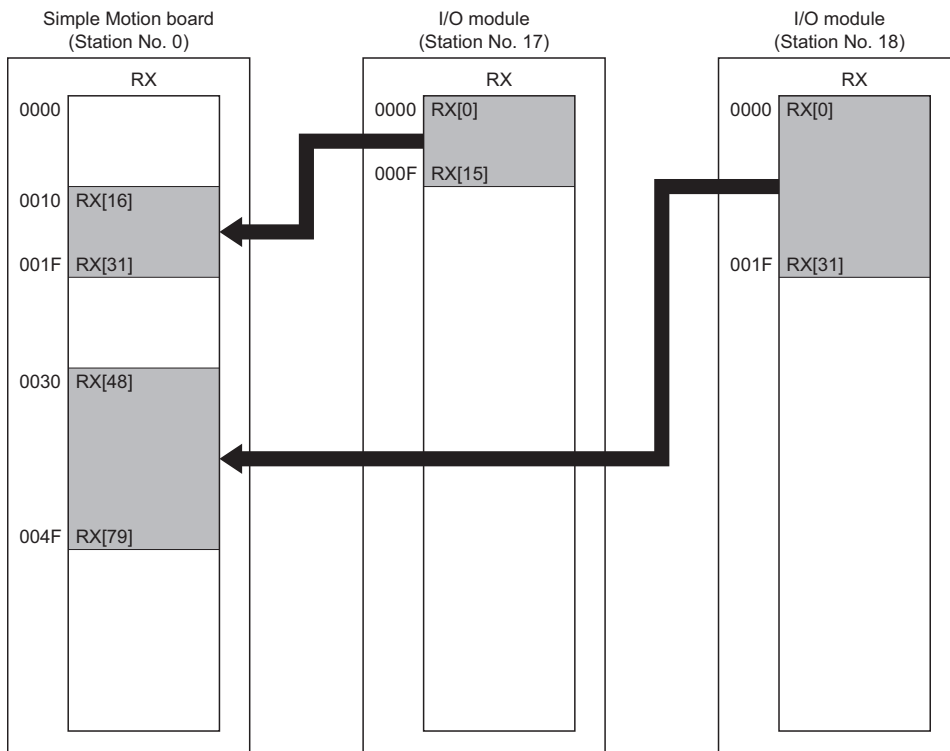
- Get I/O class objects using the GetSlavelo method of the controller class.
- When calling the GetSlavelo method of the controller class, labels for accessing the link device are automatically generated according to the parameters set in the Simple Motion board.
- Labels only retain an element the size of the area for each station. Do not access the elements outside the range.
- Access the RX areas and RY areas in 16-bit units by using the labels for word access (wRX, wRY).
- Access the RX areas, RY areas, RWw areas, and RWr areas in 32-bit units by using the labels for double word access (dwRX, dwRY, dwRWw, dwRWr).
- Access the RX areas, RY areas, RWw areas, and RWr areas in one-bit units by using the labels for bit access (wRWw[].b, wRWr[].b).

## Inheritance hierarchy

```
MMC_NetworkModule
  MMC_Slave
    MMC_Io
```

# Link device


Access the link devices allocated to I/O modules by using the labels automatically generated when getting objects.



When link devices are allocated as shown above by network parameters, controller class labels RX[16] to RX[31] and I/O class (station No. 17) labels RX[0] to RX[15] access the same area. (As does station No. 18 also.)

# 3.5 MMC\_Label Class


This class provides a function for accessing the buffer memory of the Simple Motion board.

The buffer memory of the Simple Motion board is given a name for each item which is registered as a label. By using labels, the buffer memory can be accessed in a way like using variables. Refer to the label list for the labels of each class. (  Page 203 LABEL LIST)

class MMC\_Label

## Member

### Public method

Category	Name	Description	Reference
Wait method	Wait	Waits by polling until the value of the label satisfies the specified conditions.	 Page 90 Wait

### Public data member

There is no public data member.

## Point

- When labels are array variables, do not access the elements outside the range.
- When specifying an argument of a function that outputs format strings such as the print function, cast them in label type.
- The buffer memory values cannot be referenced directly or edited by using Watch of the Visual Studio®. Store the values to variables before referencing or editing them.

## Example

```
void LabelSample(MMC_Axis *axis)
{
    /* Refer to "[Pr.1] Unit setting" */
    unsigned short unit = Axis->AxPrm.Unit;

    /* Display "[Pr.1] Unit setting" */
    printf( "%x", static_cast<unsigned short>(Axis->AxPrm.Unit) );

    /* Set "[Pr.1] Unit setting" */
    Axis->AxPrm.Unit = 0x0003;
}
```

## Inheritance hierarchy

MMC\_Label

# Wait

Waits by polling until the value of the label satisfies the specified conditions.

```
unsigned long MMC_LabelLongReadOnly::Wait(  
    unsigned long condition,  
    long value,  
    unsigned long timeout  
);  
unsigned long MMC_LabelULongReadOnly::Wait(  
    unsigned long condition,  
    unsigned long value,  
    unsigned long timeout  
);  
unsigned long MMC_LabelShortReadOnly::Wait(  
    unsigned long condition,  
    short value,  
    unsigned long timeout  
);  
unsigned long MMC_LabelUShortReadOnly::Wait(  
    unsigned long condition,  
    unsigned short value,  
    unsigned long timeout  
);  
unsigned long MMC_LabelBoolReadOnly::Wait(  
    unsigned long condition,  
    bool value,  
    unsigned long timeout  
);
```

## Detailed description

### Parameter

Argument	Description
condition [in]	Judge condition <sup>*1</sup>
value [in]	Judge value
timeout [in]	Timeout time [ms] (0 to 65535)

\*1 Specify the following values for judge condition.

Value	Description
MMC_WAIT_EQUAL	Equal to judge value
MMC_WAIT_NOT_EQUAL	Not equal to judge value
MMC_WAIT_LESS	Less than judge value
MMC_WAIT_LESS_EQUAL	Less than or equal to judge value
MMC_WAIT_GREATER	Greater than judge value
MMC_WAIT_GREATER_EQUAL	Greater than or equal to judge value

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_TIMEOUT_01	The set timeout time has elapsed.

### Point

- When MMC\_INFINITE is designated as the timeout time, timeout is not checked. Instead, this function infinitely waits for the establishment of conditions.
- For a bool label, specify one of the following to the judge condition.
  - MMC\_WAIT\_EQUAL
  - MMC\_WAIT\_NOT\_EQUAL

## ■ Example

```
void WaitServoON(MMC_Axis *axis)
{
    /* Wait until servo ON of servo status 1 turns ON */
    unsigned long returnCode;
    returnCode = axis->AxMntr.ServoStatus1_Servo.Wait( MMC_WAIT_EQUAL, MMC_ON, 10000 );
    if( returnCode != MMC_OK ){ /* Error processing */ }
}
```

## ■ Supported version

API version	Software version
1.00	01

## ■ Reference

None.

## 3.6 MMC\_DeviceDriver Class

This class provides a function for accessing the Simple Motion board via a PCI Express connection.

```
class MMC_DeviceDriver
```

### Member

#### Public method

Category	Name	Description	Reference
Delete/generate object method	Delete	Deletes object.	<a href="#">Page 93 MMC_DeviceDriver::Delete</a>
Open and close method	Open	Opens device.	<a href="#">Page 94 MMC_DeviceDriver::Open</a>
	Close	Closes device.	<a href="#">Page 95 MMC_DeviceDriver::Close</a>
Interrupt method	StartInterrupt	Starts interrupt driver.	<a href="#">Page 96 MMC_DeviceDriver::StartInterrupt</a>
	EndInterrupt	Ends interrupt driver.	<a href="#">Page 97 MMC_DeviceDriver::EndInterrupt</a>
DMA transmission method	StartDMA	Starts DMA transmission driver.	<a href="#">Page 98 MMC_DeviceDriver::StartDMA</a>
	EndDMA	Ends DMA transmission driver.	<a href="#">Page 99 MMC_DeviceDriver::EndDMA</a>
Buffer memory access method	SetBufferMemory	Sets data to buffer memory.	<a href="#">Page 100 MMC_DeviceDriver::SetBufferMemory</a>
	GetBufferMemory	Gets data from buffer memory.	<a href="#">Page 101 MMC_DeviceDriver::GetBufferMemory</a>

#### Public data member

Variable name	Data type	Initial value	Description
LastErrorNumber	unsigned long	MMERR_NONE	Latest error code
LastDetailErrorNumber	unsigned long	MMERR_NONE	Latest detail error code
BoardID	unsigned long	—	Board ID
Tag	long long	0	A variable that the user may use freely. An integer value or pointer can be cast to long long type and stored.

### Point

Generate device driver class objects with object generation functions.

### Inheritance hierarchy

MMC\_DeviceDriver

# MMC\_DeviceDriver::Delete

Deletes object.

```
void Delete(  
    void  
);
```

## Detailed description

### ■Parameter

None.

### ■Return value

None.

### ■Point

None.

### ■Supported version

API version	Software version
1.00	01

### ■Reference

None.

# MMC\_DeviceDriver::Open

Opens device.

```
unsigned long Open(  
    void  
);
```

## Detailed description

### Parameter

None.

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_REOPEN	The Open method is already called.
MMERR_DEV_DEVICE_DRIVER	An error occurred with a call of the device driver. Confirm that the device driver is installed.
MMERR_DEV_NOT_FOUND_BOARD	The Simple Motion board which has the designated board ID could not be found. Confirm the board ID selection (dip switch) of the Simple Motion board.
MMERR_DEV_UNSUPPORTED_DEVICE_DRIVER	The device driver is not a supported version. Use a API library that combines with the device driver contained in the "EM Software Development Kit".
MMERR_DEV_INACCESSIBLE	There is a Simple Motion board which cannot access the buffer memory.
MMERR_DEV_ALREADY_OTHER_PROCESS_OPEN	The device of the same board ID is already open by a different process.

### Point

- After calling the Open method for the Simple Motion board to be used, call the method which accesses the Simple Motion board.
- The same device of a board ID cannot be opened by multiple processes.
- Do not call the Open/Close method in straight succession.

### Supported version

API version	Software version
1.00	01

### Reference

Close ([↩](#) Page 95 MMC\_DeviceDriver::Close)



# MMC\_DeviceDriver::Close

Closes device.

```
unsigned long Close(  
    void  
);
```

## Detailed description

### ■Parameter

None.

### ■Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_DEV_DEVICE_DRIVER	An error occurred with a call of the device driver. Confirm that the device driver is installed.

### ■Point

Before ending the user program, call the Close method for Simple Motion boards that have been opened.

### ■Supported version

API version	Software version
1.00	01

### ■Reference

Open ([↩](#) Page 94 MMC\_DeviceDriver::Open)

# MMC\_DeviceDriver::StartInterrupt

Starts interrupt driver. This function is used when interrupt event wait functions are used for interrupt monitoring.

```
unsigned long StartInterrupt(  
    long priority  
);
```

## Detailed description

### Parameter

Argument	Description
priority [in]	Priority value

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_INT_ALREADY_START_DRIVER	The interrupt driver is already started.
MMERR_WIN_CREATE_EVENT	An error occurred in the CreateEvent function (Win32API). Call the GetLastError function of Win32API and confirm the error details.
MMERR_WIN_CREATE_THREAD	An error occurred in the CreateThread function (Win32API). Call the GetLastError function of Win32API and confirm the error details.
MMERR_WIN_SET_THREAD_PRIORITY	An error occurred in the SetThreadPriority function (Win32API). Call the GetLastError function of Win32API and confirm the error details.
MMERR_WIN_RESUME_THREAD	An error occurred in the ResumeThread function (Win32API). Call the GetLastError function of Win32API and confirm the error details.
MMERR_DEV_DEVICE_DRIVER	An error occurred with a call of the device driver. Confirm that the device driver is installed.

### Point

- Refer to the instruction manual of the operating system being used for details of the priority value.
- Call this method before calling the controller class EnableInterrupt method.

### Supported version

API version	Software version
1.00	01

### Reference

EndInterrupt (  Page 97 MMC\_DeviceDriver::EndInterrupt)

# MMC\_DeviceDriver::EndInterrupt

Ends interrupt driver.

```
unsigned long EndInterrupt(  
    void  
);
```

## Detailed description

### Parameter

None.

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_INT_ALREADY_END_DRIVER	The interrupt driver is already stopped.
MMERR_DEV_DEVICE_DRIVER	An error occurred with a call of the device driver. Confirm that the device driver is installed.
MMERR_WIN_SET_EVENT	An error occurred in the SetEvent function (Win32API). Call the GetLastError function of Win32API and confirm the error details.
MMERR_TIMEOUT_01	The timeout time (1 second) has elapsed while waiting for delete of the interrupt handler.
MMERR_WIN_CLOSE_HANDLE	An error occurred in the CloseHandle function (Win32API). Call the GetLastError function of Win32API and confirm the error details.
MMERR_WIN_GET_EXIT_CODE_THREAD	An error occurred in the GetExitCodeThread function (Win32API). Call the GetLastError function of Win32API and confirm the error details.

### Point

Call the controller class DisableInterrupt method before calling this method.

### Supported version

API version	Software version
1.00	01

### Reference

StartInterrupt (  Page 96 MMC\_DeviceDriver::StartInterrupt )

# MMC\_DeviceDriver::StartDMA

Starts DMA transmission driver.

```
unsigned long StartDMA(  
    void  
);
```

## Detailed description

### ■Parameter

None.

### ■Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_DMA_ALREADY_START_DRIVER	The DMA transmission driver is already started.
MMERR_DEV_DEVICE_DRIVER	An error occurred with a call of the device driver. Confirm that the device driver is installed.

### ■Point

Call this method before calling the controller class EnabledDMA method.

### ■Supported version

API version	Software version
1.00	01

### ■Reference

EndDMA ([↩](#) Page 99 MMC\_DeviceDriver::EndDMA)

# MMC\_DeviceDriver::EndDMA

Ends DMA transmission driver.

```
unsigned long EndDMA(  
    void  
);
```

## Detailed description

### ■Parameter

None.

### ■Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_DMA_ALREADY_END_DRIVER	The DMA transmission driver is already stopped.
MMERR_DEV_DEVICE_DRIVER	An error occurred with a call of the device driver. Confirm that the device driver is installed.

### ■Point

Call the controller class DisableDMA method before calling this method.

### ■Supported version

API version	Software version
1.00	01

### ■Reference

StartDMA ([↩](#) Page 98 MMC\_DeviceDriver::StartDMA)

# MMC\_DeviceDriver::SetBufferMemory

Sets any given data to any given address of the buffer memory.

```
unsigned long SetBufferMemory(  
    unsigned long offset,  
    unsigned long size,  
    const void *data  
);
```

## Detailed description

### Parameter

Argument	Description
offset [in]	Address offset (2-byte units) from the buffer memory top address
size [in]	Size (2-byte units) from the address offset
data [in]	Pointer to the variable that stores set data

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_DEV_ADDRESS_RANGE_OVER	The "offset" + "size" designated by the argument exceeds the size of the buffer memory.

### Point

None.

### Supported version

API version	Software version
1.00	01

### Reference

GetBufferMemory ([↗](#) Page 101 MMC\_DeviceDriver::GetBufferMemory)

# MMC\_DeviceDriver::GetBufferMemory

Gets set data from any given address of the buffer memory.

```
unsigned long GetBufferMemory(  
    unsigned long offset,  
    unsigned long size,  
    void *data  
);
```

## Detailed description

### Parameter

Argument	Description
offset [in]	Address offset (2-byte units) from the buffer memory top address
size [in]	Size (2-byte units) from the address offset
data [out]	Pointer to the variable that stores gotten data

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_DEV_ADDRESS_RANGE_OVER	The "offset" + "size" designated by the argument exceeds the size of the buffer memory.

### Point

None.

### Supported version

API version	Software version
1.00	01

### Reference

SetBufferMemory ([↩](#) Page 100 MMC\_DeviceDriver::SetBufferMemory)

## 3.7 Object Generation Functions

These functions generate the objects of the classes used by the API library. Use the object generation functions when generating objects.

### Point

- For objects generated using the object generation functions, delete them with the Delete method.
- Only one controller object and one device driver object can be generated per Simple Motion board.

### MmfCreateEM340GF (C++)

Generates the objects of the MMC\_EM340GF class.

```
unsigned long MmfCreateEM340GF(  
    MMC_DeviceDriver *deviceDriver,  
    MMC_EM340GF **controller  
);
```

#### Detailed description

##### Parameter

Argument	Description
deviceDriver [in]	Pointer to the device driver class object
controller [out]	Pointer to the controller class object

##### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_ALREADY_CREATED	The object of the same board ID has already been generated.
MMERR_CREATE_OBJECT	Failed to generate object.

##### Point

None.

##### Supported version

API version	Software version
1.00	01

##### Reference

None.



# MmfCreateEM340GF (C#)

Generates the objects of the MMC\_EM340GF class.

```
MMC_EM340GF^ MmfCreateEM340GF(  
    MMC_DeviceDriver deviceDriver  
);
```

## Detailed description

### Parameter

Argument	Description
deviceDriver [in]	Reference to the device driver class object

### Return value

Value	Description
Reference to the controller class objects	Function succeeded
null	Failed to generate object. Call the MmfGetLastError function of MMC_BasicLibrary class and confirm the error details.

### Point

None.

### Supported version

API version	Software version
1.30	01

### Reference

None.

# MmfCreatePciDevice (C++)

Generates the objects of the PCI Express device driver class.

```
unsigned long MmfCreatePciDevice(  
    unsigned long boardID,  
    MMC_DeviceDriver **deviceDriver  
);
```

## Detailed description

### Parameter

Argument	Description
boardID [in]	Board ID (0 to 3)
deviceDriver [out]	Pointer to the device driver class object pointer.

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_ALREADY_CREATED	The object of the same board ID has already been generated.
MMERR_CREATE_OBJECT	Failed to generate object.
MMERR_DEV_WIN_LOAD_LIBRARY	An error occurred in the LoadLibrary function (Win32API). Call the GetLastError function of Win32API and confirm the error details.
MMERR_DEV_WIN_GET_PROC_ADDRESS	An error occurred in the GetProcAddress function (Win32API). Call the GetLastError function of Win32API and confirm the error details.
MMERR_WIN_CREATE_SEMAPHORE	An error occurred in the CreateSemaphore function (Win32API). Call the GetLastError function of Win32API and confirm the error details.

### Point

None.

### Supported version

API version	Software version
1.00	01

### Reference

None.

# MmfCreatePciDevice (C#)

Generates the objects of the PCI Express device driver class.

```
MMC_PciWindows^ MmfCreatePciDevice(  
    uint boardID,  
);
```

## Detailed description

### Parameter

Argument	Description
boardID [in]	Board ID (0 to 3)

### Return value

Value	Description
Reference to the device driver class object	Function succeeded
null	Failed to generate object. Call the MmfGetLastError function of MMC_BasicLibrary class and confirm the error details.

### Point

None.

### Supported version

API version	Software version
1.30	01

### Reference

None.

## 3.8 Basic Library Functions

These functions are for using the basic functions provided by the operating system.

### MmfGetLastError

Gets the error codes of Basic Library functions and object generation functions.

```
unsigned long MmfGetLastError(  
    void  
);
```

#### Detailed description

##### ■Parameter

None.

##### ■Return value

Value	Description
Latest error code	Refer to error code list for details of error codes. ( <a href="#">Page 199 ERROR CODE LIST</a> )

##### ■Point

The error codes are saved at each process.

##### ■Supported version

API version	Software version
1.00	01

##### ■Reference

None.

# MmfCreateSemaphore

Creates or opens a named semaphore object.

```
unsigned long MmfCreateSemaphore(  
    const wchar_t *name,  
    HANDLE *hSemaphore  
);
```

## Detailed description

### Parameter

Argument	Description
name [in]	Name of object
hSemaphore [out]	Pointer to the variable that stores the handle of the semaphore object

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_WIN_CREATE_SEMAPHORE	An error occurred in the CreateSemaphore function (Win32API). Call the GetLastError function of Win32API and confirm the error details.

### Point

None.

### Supported version

API version	Software version
1.00	01

### Reference

MmfDeleteSemaphore (  Page 108 MmfDeleteSemaphore)

# MmfDeleteSemaphore

Closes the handle of an open semaphore object.

```
unsigned long MmfDeleteSemaphore(  
    HANDLE hSemaphore  
);
```

## Detailed description

### Parameter

Argument	Description
hSemaphore [in]	Handle of the semaphore object

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_WIN_CLOSE_HANDLE	An error occurred in the CloseHandle function (Win32API). Call the GetLastError function of Win32API and confirm the error details.

### Point

None.

### Supported version

API version	Software version
1.00	01

### Reference

MmfCreateSemaphore (  Page 107 MmfCreateSemaphore)

# MmfWaitSemaphore

Returns control when one of the following is established.

- The semaphore object becomes a signaled state.
- The timeout time elapses.

```
unsigned long MmfWaitSemaphore(  
    HANDLE hSemaphore,  
    unsigned long timeout  
);
```

## Detailed description

### Parameter

Argument	Description
hSemaphore [in]	Handle of the semaphore object
timeout [in]	Timeout time [ms] (0 to 65535)

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_TIMEOUT_01	The set timeout time has elapsed.
MMERR_WIN_WAIT_FOR_SINGLE_OBJECT	An error occurred in the WaitForSingleObject function (Win32API). Call the GetLastError function of Win32API and confirm the error details.

### Point

When MMC\_INFINITE is designated as the timeout time, timeout is not checked. Instead, this function infinitely waits for the semaphore object to become a signaled state.

### Supported version

API version	Software version
1.00	01

### Reference

MmfReleaseSemaphore (  Page 110 MmfReleaseSemaphore)

# MmfReleaseSemaphore

Increases the count of the specified semaphore object by one.

```
unsigned long MmfReleaseSemaphore(  
    HANDLE hSemaphore  
);
```

## Detailed description

### Parameter

Argument	Description
hSemaphore [in]	Handle of the semaphore object

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_WIN_RELEASE_SEMAPHORE	An error occurred in the ReleaseSemaphore function (Win32API). Call the GetLastError function of Win32API and confirm the error details.

### Point

None.

### Supported version

API version	Software version
1.00	01

### Reference

MmfWaitSemaphore (  Page 109 MmfWaitSemaphore)



# MmfCreateEvent

Creates or opens a named event object.

```
unsigned long MmfCreateEvent(  
    const wchar_t *name,  
    HANDLE *hEvent  
);
```

## Detailed description

### Parameter

Argument	Description
name [in]	Name of object
hEvent [out]	Pointer to the variable that stores the handle of the event object

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_WIN_CREATE_EVENT	An error occurred in the CreateEvent function (Win32API). Call the GetLastError function of Win32API and confirm the error details.

### Point

None.

### Supported version

API version	Software version
1.00	01

### Reference

MmfDeleteEvent ([↩](#) Page 112 MmfDeleteEvent)

# MmfDeleteEvent

Closes the handle of an open event object.

```
unsigned long MmfDeleteEvent(  
    HANDLE hEvent  
);
```

## Detailed description

### Parameter

Argument	Description
hEvent [in]	Handle of the event object

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_WIN_CLOSE_HANDLE	An error occurred in the CloseHandle function (Win32API). Call the GetLastError function of Win32API and confirm the error details.

### Point

None.

### Supported version

API version	Software version
1.00	01

### Reference

MmfCreateEvent ([↩](#) Page 111 MmfCreateEvent)

# MmfResetEvent

Sets the event object to a nonsignaled state.

```
unsigned long MmfResetEvent(  
    HANDLE hEvent  
);
```

## Detailed description

### Parameter

Argument	Description
hEvent [in]	Handle of the event object

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_WIN_RESET_EVENT	An error occurred in the ResetEvent function (Win32API). Call the GetLastError function of Win32API and confirm the error details.

### Point

None.

### Supported version

API version	Software version
1.00	01

### Reference

MmfSetEvent ([🔗](#) Page 114 MmfSetEvent)

MmfWaitEvent ([🔗](#) Page 115 MmfWaitEvent)

# MmfSetEvent

Sets the event object to a signaled state.

```
unsigned long MmfSetEvent(  
    HANDLE hEvent  
);
```

## Detailed description

### Parameter

Argument	Description
hEvent [in]	Handle of the event object

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_WIN_SET_EVENT	An error occurred in the SetEvent function (Win32API). Call the GetLastError function of Win32API and confirm the error details.

### Point

None.

### Supported version

API version	Software version
1.00	01

### Reference

MmfResetEvent ([↗](#) Page 113 MmfResetEvent)

MmfWaitEvent ([↗](#) Page 115 MmfWaitEvent)

# MmfWaitEvent

Returns control when one of the following is established.

- The event object becomes a signaled state.
- The timeout time elapses.

```
unsigned long MmfWaitEvent(  
    HANDLE hEvent,  
    unsigned long timeout  
);
```

## Detailed description

### Parameter

Value	Description
hEvent [in]	Handle of the event object
timeout [in]	Timeout time [ms] (0 to 65535)

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_TIMEOUT_01	The set timeout time has elapsed.
MMERR_WIN_WAIT_FOR_SINGLE_OBJECT	An error occurred in the WaitForSingleObject function (Win32API). Call the GetLastError function of Win32API and confirm the error details.

### Point

When MMC\_INFINITE is designated as the timeout time, timeout is not checked. Instead, this function infinitely waits for the event object to become a signaled state.

### Supported version

API version	Software version
1.00	01

### Reference

MmfResetEvent ([↗](#) Page 113 MmfResetEvent)

MmfSetEvent ([↗](#) Page 114 MmfSetEvent)

# MmfCreateThread

Creates thread.

```
unsigned long MmfCreateThread(  
    long priority,  
    unsigned long stackSize,  
    void *startAddress,  
    void *parameter,  
    HANDLE *hThread  
);
```

## Detailed description

### Parameter

Argument	Description
priority [in]	Thread priority value
stackSize [in]	Stack size [bytes]
startAddress [in]	Function of the thread
parameter [in]	Argument of the thread
hThread [out]	Pointer to the variable that stores the handle of the thread

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_WIN_CREATE_THREAD	An error occurred in the CreateThread function (Win32API). Call the GetLastError function of Win32API and confirm the error details.

### Point

None.

### Supported version

API version	Software version
1.00	01

### Reference

MmfDeleteThread ([↩](#) Page 117 MmfDeleteThread)

# MmfDeleteThread

Closes the handle of a thread.

```
unsigned long MmfDeleteThread(  
    HANDLE hThread  
);
```

## Detailed description

### Parameter

Argument	Description
hThread [in]	Handle of the thread

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_WIN_CLOSE_HANDLE	An error occurred in the CloseHandle function (Win32API). Call the GetLastError function of Win32API and confirm the error details.

### Point

None.

### Supported version

API version	Software version
1.00	01

### Reference

MmfCreateThread ( Page 116 MmfCreateThread)

# MmfWaitThread

Returns control when one of the following is established.

- The thread object becomes a signaled state.
- The timeout time elapses.

```
unsigned long MmfWaitThread (  
    HANDLE hThread,  
    unsigned long timeout  
);
```

## Detailed description

### Parameter

Value	Description
hThread [in]	Handle of the thread
timeout [in]	Timeout time [ms] (0 to 65535)

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_TIMEOUT_01	The set timeout time has elapsed.
MMERR_WIN_WAIT_FOR_SINGLE_OBJECT	An error occurred in the WaitForSingleObject function (Win32API). Call the GetLastError function of Win32API and confirm the error details.

### Point

When MMC\_INFINITE is designated as the timeout time, timeout is not checked. Instead, this function infinitely waits for the thread object to become a signaled state.

### Supported version

API version	Software version
1.00	01

### Reference

MmfResumeThread ([Page 120 MmfResumeThread](#))



# MmfSetThreadPriority

Sets the priority value of the thread.

```
unsigned long MmfSetThreadPriority(  
    HANDLE hThread,  
    long priority  
);
```

## Detailed description

### Parameter

Argument	Description
hThread [in]	Handle of the thread
priority [in]	Priority value of the thread

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_WIN_SET_THREAD_PRIORITY	An error occurred in the SetThreadPriority function (Win32API). Call the GetLastError function of Win32API and confirm the error details.

### Point

None.

### Supported version

API version	Software version
1.00	01

### Reference

None.

# MmfResumeThread

Decrements the suspend count of the thread by one.

```
unsigned long MmfResumeThread(  
    HANDLE hThread  
);
```

## Detailed description

### Parameter

Argument	Description
hThread [in]	Handle of the thread

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_WIN_RESUME_THREAD	An error occurred in the ResumeThread function (Win32API). Call the GetLastError function of Win32API and confirm the error details.

### Point

None.

### Supported version

API version	Software version
1.00	01

### Reference

MmfWaitThread ([↗](#) Page 118 MmfWaitThread)

# MmfGetExitCodeThread

Gets the exit code of the thread.

```
unsigned long MmfGetExitCodeThread(  
    HANDLE hThread,  
    unsigned long *exitCode  
);
```

## Detailed description

### Parameter

Argument	Description
hThread [in]	Handle of the thread
exitCode [out]	Pointer to the variable that stores the exit code

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_WIN_GET_EXIT_CODE_THREAD	An error occurred in the GetExitCodeThread function (Win32API). Call the GetLastError function of Win32API and confirm the error details.

### Point

None.

### Supported version

API version	Software version
1.00	01

### Reference

None.

# MmfExitThread

Exits the thread.

```
void MmfExitThread(  
    unsigned long exitCode  
);
```

## Detailed description

### ■Parameter

Argument	Description
exitCode [in]	Exit code

### ■Return value

None.

### ■Point

None.

### ■Supported version

API version	Software version
1.00	01

### ■Reference

None.

# MmfGetCurrentTime

Gets the time elapsed since system startup.

```
unsigned long MmfGetCurrentTime(  
    void  
);
```

## Detailed description

### ■Parameter

None.

### ■Return value

Elapsed time [ms] since the system startup.

### ■Point

None.

### ■Supported version

API version	Software version
1.00	01

### ■Reference

MmfCheckPassTime ([↩](#) Page 124 MmfCheckPassTime)

MmfWaitDelayTime ([↩](#) Page 125 MmfWaitDelayTime)

# MmfCheckPassTime

Checks the specified time has passed.

```
unsigned long MmfCheckPassTime(  
    unsigned long starttime,  
    unsigned long delaytime  
);
```

## Detailed description

### Parameter

Argument	Description
starttime [in]	Start time [ms]
delaytime [in]	Elapsed time [ms]

### Return value

Value	Description
MMC_OK	The specified time has elapsed.
MMC_NG	The specified time has not elapsed.

### Point

None.

### Supported version

API version	Software version
1.00	01

### Reference

MmfGetCurrentTime (  Page 123 MmfGetCurrentTime)

MmfWaitDelayTime (  Page 125 MmfWaitDelayTime)

# MmfWaitDelayTime

Cancels the execution of thread and specified time.

```
void MmfWaitDelayTime(  
    unsigned long delaytime  
);
```

## Detailed description

### Parameter

Argument	Description
delaytime [in]	Cancel time [ms]

### Return value

None.

### Point

None.

### Supported version

API version	Software version
1.00	01

### Reference

MmfGetCurrentTime ([↗](#) Page 123 MmfGetCurrentTime)

MmfCheckPassTime ([↗](#) Page 124 MmfCheckPassTime)

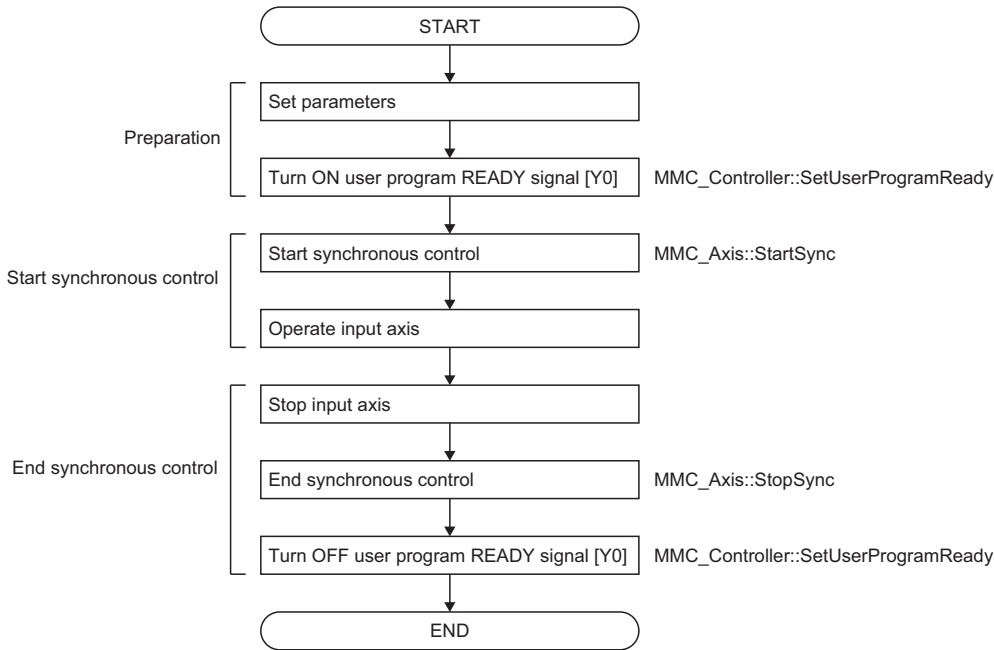
# 4 API LIBRARY DETAILS (ADVANCED SYNCHRONOUS CONTROL)

This chapter describes the procedures for using advanced synchronous control, and the specifications of the relevant classes. Refer to the following for details of advanced synchronous control.

📖 Simple Motion Board User's Manual (Advanced Synchronous Control)

## Execution procedure

The procedure for using the advanced synchronous control function is shown below.





# 4.1 MMC\_Controller Class

This class controls the controller.

```
class MMC_Controller: public MMC_Master
```

## Member

### Public method

Category	Name	Description	Reference
Get object method	GetSyncEncoder	Gets the object of the synchronous encoder axis class.	<a href="#">Page 128</a> MMC_Controller::GetSyncEncoder
Synchronous control method	CalcCamCommandPosition	Calculates cam axis feed current value.	<a href="#">Page 129</a> MMC_Controller::CalcCamCommandPosition
	CalcCamCommandPositionPerCycle	Calculates cam axis current value per cycle.	<a href="#">Page 130</a> MMC_Controller::CalcCamCommandPositionPerCycle
	MakeRotaryCutterCam	Auto-generates the cam (central reference) for rotary cutter.	<a href="#">Page 131</a> MMC_Controller::MakeRotaryCutterCam
	MakeEasyStrokeRatioCam	Auto-generates the easy stroke ratio cam.	<a href="#">Page 132</a> MMC_Controller::MakeEasyStrokeRatioCam
	MakeAdvancedStrokeRatioCam	Auto-generates the advanced stroke ratio cam.	<a href="#">Page 133</a> MMC_Controller::MakeAdvancedStrokeRatioCam

### Public data member

Variable name	Data type	Initial value	Description
LastErrorNumber	unsigned long	MMERR_NONE	Latest error code
LastDetailErrorNumber	unsigned long	MMERR_NONE	Latest detail error code

### Label

Refer to label list for labels. ([Page 203 LABEL LIST](#))

## Inheritance hierarchy

```
MMC_NetworkModule
  MMC_Master
    MMC_Controller
```

# MMC\_Controller::GetSyncEncoder

Gets the object of the synchronous encoder axis class.

```
unsigned long GetSyncEncoder(  
    unsigned long axisNumber,  
    MMC_SyncEncoder **syncEncoder  
);
```

## Detailed description

### Parameter

Argument	Description
axisNumber [in]	Synchronous encoder axis No. (1 to 16)
syncEncoder [out]	Pointer to the synchronous encoder axis class object pointer

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_CREATE_OBJECT	Failed to generate object.

### Point

Because the objects gotten by this method are automatically deleted at the time of controller class object delete, calling the Delete method of the synchronous encoder axis class is not necessary.

### Supported version

API version	Software version
1.00	01

### Reference

None.

# MMC\_Controller::CalcCamCommandPosition

Calculates cam axis feed current value.

```
unsigned long CalcCamCommandPosition(  
    const MMST_CamPositionData &camPositionData,  
    long *result  
);
```

## Detailed description

### Parameter

Argument	Description
camPositionData [in]	Cam position calculation data structure
result [out]	Pointer to the variable that stores cam position calculation result

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_TIMEOUT_01	The timeout time (1 second) has elapsed.

### Point

- This method uses the "cam position calculation function" of the Simple Motion board.
- This method waits inside the method until "[Cd.612] Cam position calculation request" changes to "0".
- For this method, the "[Cd.618] Cam position calculation: Cam axis feed current value" member variable of the structure is not used. (Setting it is not required.)

### Supported version

API version	Software version
1.00	01

### Reference

CalcCamCommandPositionPerCycle ([↩](#) Page 130 MMC\_Controller::CalcCamCommandPositionPerCycle)

# MMC\_Controller::CalcCamCommandPositionPerCycle

Calculates cam axis current value per cycle.

```
unsigned long CalcCamCommandPositionPerCycle(  
    const MMST_CamPositionData &camPositionData,  
    long *result  
);
```

## Detailed description

### Parameter

Argument	Description
camPositionData [in]	Cam position calculation data structure
result [out]	Pointer to the variable that stores cam position calculation result

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_TIMEOUT_01	The timeout time (1 second) has elapsed.

### Point

- This method uses the "cam position calculation function" of the Simple Motion board.
- This method waits inside the method until "[Cd.612] Cam position calculation request" changes to "0".

### Supported version

API version	Software version
1.00	01

### Reference

CalcCamCommandPosition (  Page 129 MMC\_Controller::CalcCamCommandPosition)

# MMC\_Controller::MakeRotaryCutterCam

Auto-generates the cam (central reference) for rotary cutter.

```
unsigned long MakeRotaryCutterCam(  
    unsigned short camNo,  
    const MMST_RotaryCutterCamData &rotaryCutterCamData,  
    unsigned short *asynchronousSpeedResult  
);
```

## Detailed description

### Parameter

Argument	Description
camNo [in]	Cam auto-generation cam No. (1 to 1024)
rotaryCutterCamData [in]	Data structure for auto-generation (rotary cutter cam (central reference))
asynchronousSpeedResult [out]	Pointer to the asynchronous speed result

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_TIMEOUT_01	The timeout time (5 seconds) has elapsed.

### Point

- This method uses the "Cam auto-generation function" of the Simple Motion board. Set the structure data specified by the argument as the parameters and execute cam auto-generation function.
- This method waits inside the method until "[Cd.608] Cam auto-generation request" changes to "0".

### Supported version

API version	Software version
1.00	01

### Reference

None.

# MMC\_Controller::MakeEasyStrokeRatioCam

Auto-generates the easy stroke ratio cam.

```
unsigned long MakeEasyStrokeRatioCam(  
    unsigned short camNo,  
    const MMST_EasyStrokeRatioCamData &easyStrokeRatioCamData,  
    const MMST_EasyStrokeRatioCamSectionData easyStrokeRatioCamSectionData[]  
);
```

## Detailed description

### Parameter

Argument	Description
camNo [in]	Cam auto-generation cam No. (1 to 1024)
easyStrokeRatioCamData [in]	Data structure for easy stroke ratio cam
easyStrokeRatioCamSectionData [in]	Pointer to section data structure array for easy stroke ratio cam

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_TIMEOUT_01	The timeout time (5 seconds) has elapsed.

### Point

- This method uses the "Cam auto-generation function" of the Simple Motion board. Set the structure data specified by the argument as the parameters and execute cam auto-generation function.
- This method waits inside the method until "[Cd.608] Cam auto-generation request" changes to "0".
- When a value outside the set range is set to the member variable of a structure, the No. of the member variable (1 or more) is stored to the details error code.

### Supported version

API version	Software version
1.00	01

### Reference

None.

# MMC\_Controller::MakeAdvancedStrokeRatioCam

Auto-generates the advanced stroke ratio cam.

```
unsigned long MakeAdvancedStrokeRatioCam(  
    unsigned short camNo,  
    const MMST_AdvancedStrokeRatioCamData &advancedStrokeRatioCamData,  
    const MMST_AdvancedStrokeRatioCamSectionData advancedStrokeRatioCamSectionData[]  
);
```

## Detailed description

### Parameter

Argument	Description
camNo [in]	Cam auto-generation cam No. (1 to 1024)
advancedStrokeRatioCamData [in]	Data structure for advanced stroke ratio cam
advancedStrokeRatioCamSectionData [in]	Pointer to section data structure array for advanced stroke ratio cam

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_TIMEOUT_01	The timeout time (5 seconds) has elapsed.

### Point

- This method uses the "Cam auto-generation function" of the Simple Motion board. Set the structure data specified by the argument as the parameters and execute cam auto-generation function.
- This method waits inside the method until "[Cd.608] Cam auto-generation request" changes to "0".
- When a value outside the set range is set to the member variable of a structure, the No. of the member variable (1 or more) is stored to the details error code.

### Supported version

API version	Software version
1.00	01

### Reference

None.





## 4.2 MMC\_Axis Class

This class controls the servo amplifier axis.

```
class MMC_Axis: public MMC_Slave
```

### Member


#### ■Public method

Category	Name	Description	Reference
Synchronous control method	StartSync	Starts synchronous control.	 Page 135 MMC_Axis::StartSync
	StopSync	Stops synchronous control.	 Page 136 MMC_Axis::StopSync
	ChangeSyncPosition	Changes current value during synchronous control.	 Page 137 MMC_Axis::ChangeSyncPosition
	MoveCamPosition	Moves cam axis during synchronous control.	 Page 138 MMC_Axis::MoveCamPosition

#### ■Public data member

Variable name	Data type	Initial value	Description
LastErrorNumber	unsigned long	MMERR_NONE	Latest error code
LastDetailErrorNumber	unsigned long	MMERR_NONE	Latest detail error code
StationNumber	unsigned long	—	Station No. (1 to 16)

#### ■Label

Refer to label list for labels. ( Page 203 LABEL LIST)

### Inheritance hierarchy

MMC\_NetworkModule

MMC\_Slave

MMC\_Axis



# MMC\_Axis::StartSync

Starts synchronous control.

```
unsigned long StartSync(  
    void  
);
```

## Detailed description

### Parameter

None.

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_TIMEOUT_01	The timeout time (10 seconds) has elapsed.
MMERR_AXIS_ERROR	An axis error occurred. Check "[Md.23] Axis error No."
MMERR_CONDITION_READY_SIGNAL_OFF	The READY signal [X0] is OFF.
MMERR_CONDITION_SYNC_START_ON	"[Cd.380] Synchronous control start" is ON.
MMERR_CONDITION_AXIS_ERROR_DETECTION	Error detection ("[Md.31] Status": b13) is ON.
MMERR_CONDITION_BUSY_ON	BUSY signal [X10 to X1F] are ON.


### Point

- After turning OFF→ON the applicable axis bit of "[Cd.380] Synchronous control start", this method waits inside the method until "[Md.26] Axis operation status" changes to "15: Synchronous control".
- It takes about 1 to 26 [ms] to start synchronous control because of the time it takes to confirm the response of the Simple Motion board.
- This method cannot be used in conjunction with the synchronous control analysis mode. When using the synchronous control analysis mode, use a label when setting.

### Supported version

API version	Software version
1.00	01

### Reference

StopSync (  Page 136 MMC\_Axis::StopSync)

# MMC\_Axis::StopSync

Stops synchronous control.

```
unsigned long StopSync(  
    void  
);
```

## Detailed description

### ■Parameter

None.

### ■Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_TIMEOUT_01	The timeout time (1 second) has elapsed.
MMERR_AXIS_ERROR	An axis error occurred. Check "[Md.23] Axis error No."

### ■Point

- After turning ON→OFF the applicable axis bit of "[Cd.380] Synchronous control start", this method waits inside the method until "[Md.26] Axis operation status" changes to "0: Standby", or "1: Stopped".
- With this method, synchronous control can be stopped even while the input axis is in operation. However, because the output axis comes to an immediate stop, we recommend stopping the input axis operation before stopping synchronous control.

### ■Supported version

API version	Software version
1.00	01

### ■Reference

StartSync ([🔗](#) Page 135 MMC\_Axis::StartSync)

# MMC\_Axis::ChangeSyncPosition

Changes current value of cam axis current value per cycle, current value per cycle after main gear, and current value per cycle after auxiliary shaft gear during synchronous control.

```
unsigned long ChangeSyncPosition(  
    unsigned short target,  
    long value  
);
```

## Detailed description

### Parameter

Argument	Description
target [in]	Change target <sup>*1</sup>
value [in]	[Cd.408] Synchronous control change value


\*1 Specify the following values for change target.

Value	Description
MMC_SYNC_POS_CAM_AXIS	Change cam axis current value per cycle
MMC_SYNC_POS_MAIN_SHAFT_GEAR	Change current value per cycle after main gear
MMC_SYNC_POS_AUX_SHAFT_GEAR	Change current value per cycle after auxiliary shaft gear

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_TIMEOUT_01	The timeout time (1 second) has elapsed.
MMERR_AXIS_ERROR	An axis error occurred. Check "[Md.23] Axis error No."
MMERR_CONDITION_AXIS_OPERATION_STATUS	"[Md.26] Axis operation status" is not in synchronous control.


### Point

- This method uses the "Synchronous control change function" of the Simple Motion board.
- This method waits inside the method until "[Cd.406] Synchronous control change request" changes to "0".
- Current value change is completed in one operation cycle.
- Refer to the following for setting values (unit, setting range) of "[Cd.408] Synchronous control change value".  
 Simple Motion Board User's Manual (Advanced Synchronous Control)

### Supported version

API version	Software version
1.00	01

### Reference

MoveCamPosition ( Page 138 MMC\_Axis::MoveCamPosition)

# MMC\_Axis::MoveCamPosition

Moves cam axis current value per cycle, or cam reference position during synchronous control.

```
unsigned long MoveCamPosition(  
    unsigned short target,  
    long value,  
    unsigned short reflectionTime  
);
```

## Detailed description

### Parameter

Argument	Description
target [in]	Change target* <sup>1</sup>
value [in]	[Cd.408] Synchronous control change value
reflectionTime [in]	[Cd.409] Synchronous control reflection time [ms]


\*1 Specify the following values for change target.

Value	Description
MMC_SYNC_MOVE_CAM_REFERENCE_POS	Cam axis reference position
MMC_SYNC_MOVE_CAM_POS_PER_CYCLE	Cam axis current value per cycle

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_TIMEOUT_01	The timeout time (synchronous control reflection time +10 seconds) has elapsed.
MMERR_AXIS_ERROR	An axis error occurred. Check "[Md.23] Axis error No.".
MMERR_CONDITION_AXIS_OPERATION_STATUS	"[Md.26] Axis operation status" is not in synchronous control.


### Point

- This method uses the "Synchronous control change function" of the Simple Motion board.
- This method waits inside the method until "[Cd.406] Synchronous control change request" changes to "0".
- Refer to the following for setting values (unit, setting range) of "[Cd.408] Synchronous control change value", and "[Cd.409] Synchronous control reflection time"  
 Simple Motion Board User's Manual (Advanced Synchronous Control)
- Because the cam axis feed current value also moves with the movement amount, when setting a large movement amount, make the reflection time larger as well.

### Supported version

API version	Software version
1.00	01

### Reference

ChangeSyncPosition (  Page 137 MMC\_Axis::ChangeSyncPosition)

# 4.3 MMC\_SyncEncoder Class

This class controls the synchronous encoder axis.

```
class MMC_SyncEncoder: public MMC_Slave
```

## Member

### Public method

Category	Name	Description	Reference
Error method	ResetSyncEncoderError	Performs error reset of the synchronous encoder axis.	Page 140 MMC_SyncEncoder::ResetSyncEncoderError
Change method	ChangeSyncEncoderPosition	Changes current value of the synchronous encoder axis.	Page 141 MMC_SyncEncoder::ChangeSyncEncoderPosition
Enable/disable method	DisableSyncEncoder	Disables input from the synchronous encoder axis.	Page 142 MMC_SyncEncoder::DisableSyncEncoder
	EnableSyncEncoder	Enables input from the synchronous encoder axis.	Page 143 MMC_SyncEncoder::EnableSyncEncoder

### Public data member

Variable name	Data type	Initial value	Description
LastErrorNumber	unsigned long	MMERR_NONE	Latest error code
LastDetailErrorNumber	unsigned long	MMERR_NONE	Latest detail error code
AxisNumber	unsigned long	—	Synchronous encoder axis No. (1 to 16)

### Label

Refer to label list for labels. (Page 203 LABEL LIST)

## Inheritance hierarchy

- MMC\_NetworkModule
  - MMC\_Slave
    - MMC\_SyncEncoder

# MMC\_SyncEncoder::ResetSyncEncoderError

Performs error reset of the synchronous encoder axis.

```
unsigned long ResetSyncEncoderError(  
    void  
);
```

## Detailed description

### ■Parameter

None.

### ■Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_TIMEOUT_01	The timeout time (1 second) has elapsed.

### ■Point

This method waits inside the method until the error reset is completed.

### ■Supported version

API version	Software version
1.00	01

### ■Reference

None.

# MMC\_SyncEncoder::ChangeSyncEncoderPosition

Changes the synchronous encoder axis current value and synchronous encoder axis current value per cycle.

```
unsigned long ChangeSyncEncoderPosition(  
    unsigned short startType,  
    long newPosition  
);
```

## Detailed description

### Parameter

Argument	Description
startType [in]	Start method*1
newPosition [in]	[Cd.322] Synchronous encoder axis current value setting address

\*1 Specify the following values for start method.

Value	Description
MMC_SYNC_ENC_START	Starts synchronous encoder axis control.
MMC_SYNC_ENC_LINKDEV_START	Wait to receive start request from link device.

### Return value


Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_CONDITION_SYNC_ENCODER_INVALID	The settings valid flag ("[Md.325] Synchronous encoder axis status": b0) is OFF.
MMERR_TIMEOUT_01	The timeout time (1 second) has elapsed.

### Point

- The operation for this method differs depending on the value set for synchronous encoder axis control start.

Value	Description
MMC_SYNC_ENC_START	For synchronous encoder axis control start, the synchronous encoder axis current value change is executed by calling the function. During this time, this method waits inside the method until the current value change is complete.
MMC_SYNC_ENC_LINKDEV_START	For synchronous encoder axis start request (link device), after calling the function, synchronous encoder axis control is started by link device input. During this time, this method returns control once in a state of waiting to receive input.

- Refer to the following for setting values (unit, setting range) of "[Cd.322] Synchronous encoder axis current value setting address".

 Simple Motion Board User's Manual (Advanced Synchronous Control)

### Supported version

API version	Software version
1.00	01

### Reference

None.

# MMC\_SyncEncoder::DisableSyncEncoder

Disables input from the synchronous encoder axis.

```
unsigned long DisableSyncEncoder(  
    unsigned short startType  
);
```

## Detailed description

### Parameter

Argument	Description
startType [in]	Start method <sup>*1</sup>

\*1 Specify the following values for start method.

Value	Description
MMC_SYNC_ENC_START	Starts synchronous encoder axis control.
MMC_SYNC_ENC_LINKDEV_START	Wait to receive start request from link device.

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_CONDITION_SYNC_ENCODER_INVALID	The settings valid flag ("[Md.325] Synchronous encoder axis status": b0) is OFF.
MMERR_TIMEOUT_01	The timeout time (1 second) has elapsed.

### Point

- The operation for this method differs depending on the value set for synchronous encoder axis control start.

Value	Description
MMC_SYNC_ENC_START	For synchronous encoder axis control start, the synchronous encoder axis is counter disabled by calling the function. During this time, this method waits inside the method until input from the synchronous encoder axis is disabled.
MMC_SYNC_ENC_LINKDEV_START	For synchronous encoder axis start request (link device), after calling the function, synchronous encoder axis is counter disabled by link device input. During this time, this method returns control once in a state of waiting to receive input.

- Smoothing, phase compensation, and rotation direction restriction processes continue. Therefore, when these processes are enabled, the input axis speed may not stop immediately.

### Supported version

API version	Software version
1.00	01

### Reference

EnableSyncEncoder (  Page 143 MMC\_SyncEncoder::EnableSyncEncoder)



# MMC\_SyncEncoder::EnableSyncEncoder

Enables input from the synchronous encoder axis.

```
unsigned long EnableSyncEncoder(  
    unsigned short startType  
);
```

## Detailed description

### Parameter

Argument	Description
startType [in]	Start method <sup>*1</sup>

\*1 Specify the following values for start method.

Value	Description
MMC_SYNC_ENC_START	Starts synchronous encoder axis control.
MMC_SYNC_ENC_LINKDEV_START	Wait to receive start request from link device.

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_CONDITION_SYNC_ENCODER_INVALID	The settings valid flag ("[Md.325] Synchronous encoder axis status": b0) is OFF.
MMERR_TIMEOUT_01	The timeout time (1 second) has elapsed.

### Point

The operation for this method differs depending on the value set for synchronous encoder axis control start.

Value	Description
MMC_SYNC_ENC_START	For synchronous encoder axis control start, the synchronous encoder axis is counter enabled by calling the function. During this time, this method waits inside the method until input from the synchronous encoder axis is enabled.
MMC_SYNC_ENC_LINKDEV_START	For synchronous encoder axis start request (link device), after calling the function, synchronous encoder axis is counter enabled by link device input. During this time, this method returns control once in a state of waiting to receive input.

### Supported version

API version	Software version
1.00	01

### Reference

DisableSyncEncoder (  Page 142 MMC\_SyncEncoder::DisableSyncEncoder)

# 5 API LIBRARY DETAILS (FUNCTION BLOCK)

For the function block class, the MR-J4-GF which is set to not only the motion mode but also the I/O mode can be used as the control target servo amplifier.

Refer to the servo amplifier instruction manual for details of the motion mode and the I/O mode.

<Motion mode>

High-level motion control, such as high-level positioning control and speed-torque control, can be used. Up to 16 axes can be controlled.

The station-specific mode of "Communication mode setting for CC-Link IE communication (PN03)" for MR-J4-GF is used with "0: Motion mode".

<I/O mode>

The servo amplifier built-in positioning function is used. Up to 120 stations can be controlled.

The station-specific mode of "Communication mode setting for CC-Link IE communication (PN03)" is used with "1: I/O mode".

## 5.1 MC\_FunctionBlock Class

This class provides Motion control functions for the specifications of the PLCopen Motion control function blocks.

```
class MC_FunctionBlock
```

### Member

#### Public method

Category	Name	Description	Reference
Update method	Update	Updates the I/O data of the function block.	Page 152 MC_FunctionBlock::Update

#### Public data member

Variable name	Data type	Initial value	Description
Tag	long long	0	A variable that the user may use freely. An integer value or pointer can be cast to long long type and stored.

#### Derived classes

○: Available, ×: Not available

Name	Description	Motion mode	I/O mode	Reference
MC_Power	Changes the servo amplifier of the specified axis to an operable state.	○	○	Page 153 MC_Power class
MCv_Home	Executes home position return for the specified axis.	○	○	Page 155 MCv_Home class
MC_Stop	Stops the specified axis.	○	○	Page 157 MC_Stop class
MC_MoveAbsolute	Specifies the absolute target position of the specified axis and executes positioning.	○	○	Page 159 MC_MoveAbsolute class
MC_MoveRelative	Moves the specified distance from the current position.	○	○	Page 162 MC_MoveRelative class
MC_Reset	Cancels the errors of the specified axis.	○	○	Page 165 MC_Reset class
MC_MoveAdditive	Adds the relative position specified by the positioning command of the specified axis immediately before, and executes positioning.	○	×	Page 167 MC_MoveAdditive class
MC_MoveVelocity	Executes speed control for the specified axis at the specified speed.	○	×	Page 169 MC_MoveVelocity class
MC_TorqueControl	Executes torque control for the specified axis at the specified torque.	○	×	Page 171 MC_TorqueControl class
MC_SetPosition	Changes the current position (command position and feedback position) of the specified axis.	○	×	Page 173 MC_SetPosition class
MC_SetOverride	Changes the target speed of the specified axis.	○	×	Page 175 MC_SetOverride class
MC_ReadActualPosition	Reads the current position of the specified axis.	○	○	Page 177 MC_ReadActualPosition class

Name	Description	Motion mode	I/O mode	Reference
MC_ReadStatus	Returns the detailed state of the state diagram of the specified axis.	○	○	Page 179 MC_ReadStatus class
MC_ReadAxisInfo	Reads the axis information of the specified axis.	○	○	Page 181 MC_ReadAxisInfo class
MC_ReadAxisError	Reads the error No. of the specified axis.	○	○	Page 183 MC_ReadAxisError class
MCv_ReadServoParameter	Reads the parameter value of the servo parameter No. of the specified axis.	×	○	Page 185 MCv_ReadServoParameter class
MCv_WriteServoParameter	Changes the parameter value of the servo parameter No. of the specified axis.	×	○	Page 187 MCv_WriteServoParameter class

## Point

- When executing a function block again, set the execution command (Execute, Enable) to OFF, and execute after calling the Update method.
- When using multiple function blocks and starting them all on the same axis simultaneously, operation will be abnormal.
- Operations are not guaranteed when using axis class methods such as PositioningStart method together with function blocks simultaneously.
- A MCv class name indicates a function block for vendor specifications.
- Function blocks use axis information classes (AXIS\_REF class).
- The slave stations that can be controlled are 1 to 16 stations (axes) in the motion mode and 1 to 120 stations in the I/O mode. Set the station No. of the servo amplifier within the setting range.
- For the function block class, if a value is not set to the input variable, the default value (0) is applied.
- When turning OFF the function block execution command (Execute) after turning it ON, be sure to turn it OFF after it is in an executing state (BUSY).
- Do not overwrite the output variable by user program. If overwritten, it may not operate normally.
- In this section, the bool values of the function block I/O are written as ON/OFF. According to the programming language, read them as follows.
  - C++
    - ON: true
    - OFF: false

### <I/O mode>

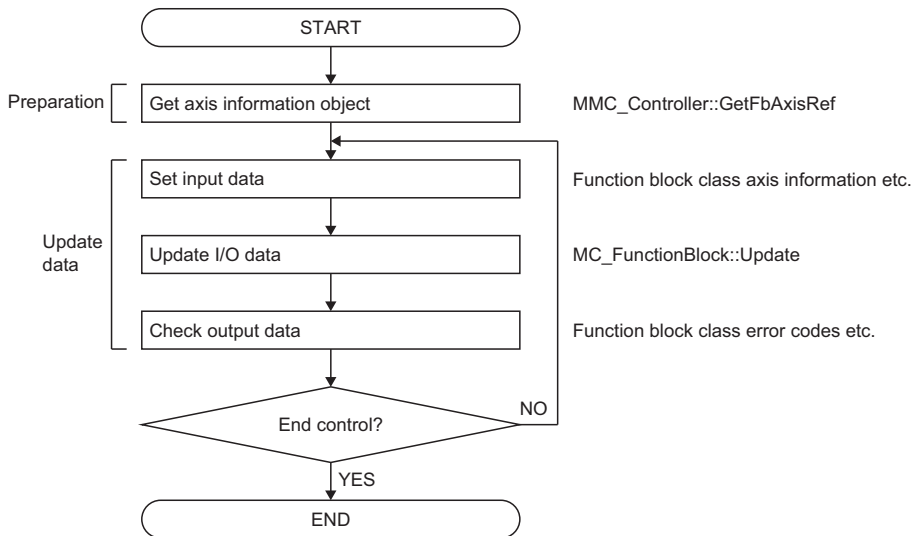
- This class accesses the servo parameter groups PA, PD, and PT. Before using function blocks, set the servo parameter "Parameter writing inhibit (PA19)" to "00ABH".
- This class accesses link devices. Before accessing the link devices with user programs, turn ON LinkDeviceUse of the AXIS\_REF class and check that LinkDeviceAccessible turns ON.
- The servo amplifier operates with the point table No. input method.

## Inheritance hierarchy

MC\_FunctionBlock

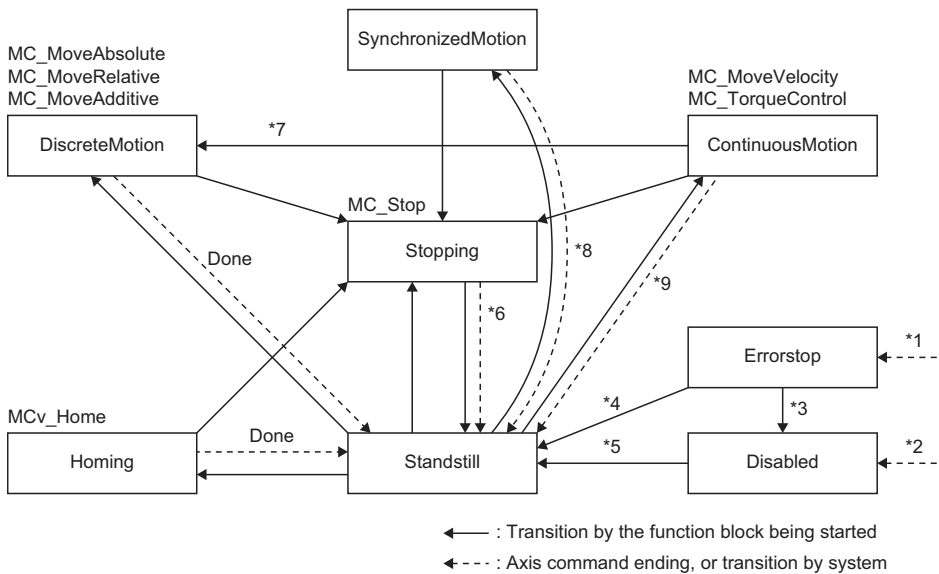
## Execution procedure

The procedure for using the functions of the function block class is shown below.



# Axis information class state diagram

The axis information class (AXIS\_REF class) state diagram is shown below. The axis is always in one of the defined states.  
<Motion mode>



- \*1 When an error occurs in the axis, transition occurs from any state.
- \*2 When Enable=OFF for MC\_Power and there is no error in the axis.
- \*3 MC\_Reset is executed and Status=OFF for MC\_Power.
- \*4 When MC\_Reset is executed and Enable=ON for MC\_Power, and Status=ON for MC\_Power.
- \*5 When Enable=ON for MC\_Power and Status=ON for MC\_Power.
- \*6 When Done=ON for MC\_Stop and Execute=OFF for MC\_Stop. (Excluding MC\_Power)
- \*7 When axis is stopped by the execution of Velocity=0 for MC\_MoveVelocity, or Torque=0 for MC\_TorqueControl.
- \*8 When "[Cd.380] Synchronous control start" is turned OFF.
- \*9 When "[Cd.181] Forward run JOG start" and "[Cd.182] Reverse run JOG start" are turned OFF.

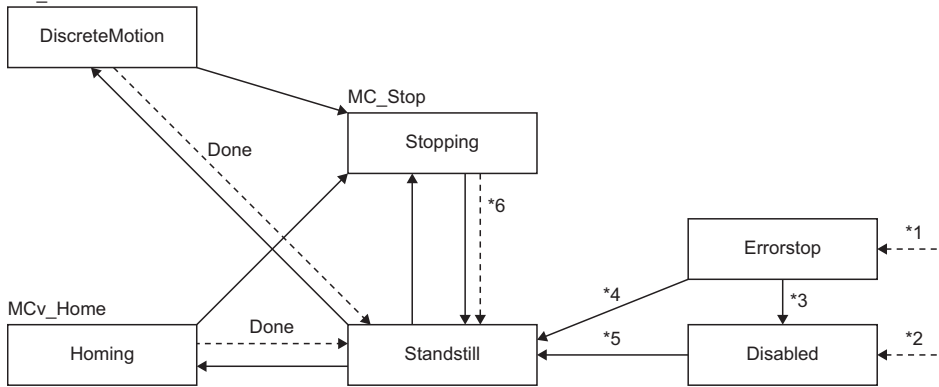
State	Description
Disabled	The initial state of the axis. Enable is OFF for MC_Power and no error is occurring in the axis.
Errorstop	Transitions when an error has occurred. In an error occurrence, the axis remains in this state.
Stopping	Transition by MC_Stop execution. While Execute is ON for MC_Stop, the axis remains in the Stopping state. "[Cd.380] Synchronous control start", "[Cd.181] Forward run JOG start", and "[Cd.182] Reverse run JOG start" do not turn OFF.
Homing	The axis is in home position return operation.
Standstill	MC_Power is ON and an error is not occurring in the axis.
DiscreteMotion	A function block for positioning control is being executed. Transition by MC_MoveAbsolute, MC_MoveRelative, and MC_Additive execution.
ContinuousMotion	A function block for continuous control is being executed. Transition by MC_MoveVelocity and MC_TorqueControl execution.
SynchronizedMotion	In synchronized control. Can not be used in this function block/

The axis state can be checked with the FbAxisStatus of axis information class (AXIS\_REF class). The following values are stored to FbAxisStatus.

Value	Description
MMC_FBST_DISABLED	Disabled
MMC_FBST_ERRORSTOP	Errorstop
MMC_FBST_STOPPING	Stopping
MMC_FBST_HOMING	Homing
MMC_FBST_STANDSTILL	Standstill
MMC_FBST_DISCRETE_MOTION	DiscreteMotion
MMC_FBST_CONTINUOUS_MOTION	ContinuousMotion

</I/O mode>

MC\_MoveAbsolute  
MC\_MoveRelative



← : Transition by the function block being started  
 ←- - - : Axis command ending, or transition by system

- \*1 When an error occurs in the axis, transition occurs from any state.
- \*2 When Enable=OFF for MC\_Power and there is no error in the axis.
- \*3 MC\_Reset is executed and Status=OFF for MC\_Power.
- \*4 When MC\_Reset is executed and Enable=ON for MC\_Power, and Status=ON for MC\_Power.
- \*5 When Enable=ON for MC\_Power and Status=ON for MC\_Power.
- \*6 When Done=ON for MC\_Stop and Execute=OFF for MC\_Stop. (Excluding MC\_Power)

State	Description
Disabled	The initial state of the axis. Enable is OFF for MC_Power and no error is occurring in the axis.
Errorstop	Transitions when an error has occurred. In an error occurrence, the axis remains in this state.
Stopping	Transition by MC_Stop execution. While Execute is ON for MC_Stop, the axis remains in the Stopping state.
Homing	The axis is in home position return operation.
Standstill	MC_Power is ON and an error is not occurring in the axis.
DiscreteMotion	A function block for positioning control is being executed. Transition by MC_MoveAbsolute and MC_MoveRelative.

The axis state can be checked with the FbAxisStatus of axis information class (AXIS\_REF class). The following values are stored to FbAxisStatus.

Value	Description
MMC_FBST_DISABLED	Disabled
MMC_FBST_ERRORSTOP	Errorstop
MMC_FBST_STOPPING	Stopping
MMC_FBST_HOMING	Homing
MMC_FBST_STANDSTILL	Standstill
MMC_FBST_DISCRETE_MOTION	DiscreteMotion

# Function block units

The following units are available in function block class. When a value is input with more digits after the decimal point than the valid amount of digits, the final valid digit is rounded to the nearest number.

<Motion mode>

Item		Unit			
		mm	inch	degree	pulse
Control unit <sup>*1</sup>		mm	inch	degree	pulse
Positioning range	Absolute method	-214748364.8 to 214748364.7[μm]	-21474.83648 to 21474.83647[inch]	0 to 359.99999[degree]	-2147483648 to 2147483647[pulse]
	Incremental method			-21474.83648 to 21474.83647[degree]	
	Current value change			0 to 359.99999[degree]	
Speed command	Positioning control	0.01 to 20000000.00 [mm/min]	0.001 to 2000000.000 [inch/min]	0.001 to 2000000.000 [degree/min] <sup>*2</sup>	1 to 1000000000[pulse/s]
	Speed control	-20000000.00 to 20000000.00 [mm/min]	-2000000.000 to 2000000.000[inch/min]	-2000000.000 to 2000000.000[degree/min] <sup>*3</sup>	-1000000000 to 1000000000[pulse/s]
	Torque control	0 to 20000000.00[mm/min]	0 to 2000000.000[inch/min]	0 to 2000000.000 [degree/min] <sup>*4</sup>	0 to 1000000000[pulse/s]
Acceleration/ deceleration time	Positioning control	1 to 8388608 [ms]			
	Speed control	0 to 65535 [ms]			

\*1 Control unit is set by the basic parameter "[Pr.1] Unit setting".

\*2 Range when "[Pr.83] Speed control 10 x multiplier setting for degree axis" is valid: 0.01 to 20000000.00[degree/min]

\*3 Range when "[Pr.83] Speed control 10 x multiplier setting for degree axis" is valid: -20000000.00 to 20000000.00[degree/min]

\*4 Range when "[Pr.83] Speed control 10 x multiplier setting for degree axis" is valid: 0 to 20000000.00[degree/min]

<I/O mode>

Item		Unit		
		mm	inch	pulse
Control unit <sup>*1</sup>		mm	inch	pulse
Positioning range		-999.999 to 999.999 [ × 10 <sup>STM</sup> ] [mm] <sup>*2</sup>	-999999 to 999999 [ × 10 <sup>(STM-4)</sup> ] [inch] <sup>*2</sup>	-999999 to 999999 [pulse]
Speed command		0.00 to 167772.15 [r/min or mm/s] <sup>*3</sup>		
Acceleration/deceleration time		0 to 20000 [ms]		

\*1 Control unit is set by the servo parameter "Command mode selection (PT01)".

\*2 STM=Feed length multiplication (PT03)

\*3 Set a value within the permissible rotation speed or permissible speed of the servo motor.

# Function blocks for PLCopen Motion control

A compatibility chart of the function blocks for PLCopen Motion control is shown below.

## Function blocks for PLCopen Motion control

Part	Category		Function block for PLCopen Motion control		API library
			Name	Description	
Part1&Part2	Control	Single axis	MC_Power	Operable	MC_Power
			MC_ReadParameter	Read parameter	Use labels for functions supported by function blocks for PLCopen Motion control. For supported labels, refer to label access <sup>*1</sup> (Page 151 Label access)
			MC_WriteParameter	Write parameter	
			MC_ReadActualVelocity	Read current velocity	
			MC_ReadActualTorque	Read current torque	Use labels for functions supported by function blocks for PLCopen Motion control. For supported labels, refer to label access (Page 151 Label access)
			MC_ReadDigitalInput	Read digital input	
			MC_ReadDigitalOutput	Read digital output	
			MC_WriteDigitalOutput	Write digital output	
			MC_ReadAxisError	Read axis error	MC_ReadAxisError
			MC_ReadActualPosition	Read current position	MC_ReadActualPosition
			MC_ReadAxisInfo	Read axis information	MC_ReadAxisInfo
			MC_SetPosition	Change current position	MC_SetPosition <sup>*1</sup>
			MC_SetOverride	Set override value	MC_SetOverride <sup>*1</sup>
			MC_Reset	Reset axis error	MC_Reset
	Operation	MC_Home	Home position return	MCv_Home	
		MC_Stop	Forced stop	MC_Stop	
		MC_MoveRelative	Relative value positioning	MC_MoveRelative	
		MC_MoveAbsolute	Absolute value positioning	MC_MoveAbsolute	
		MC_MoveAdditive	Change target position	MC_MoveAdditive <sup>*1</sup>	
		MC_MoveVelocity	Speed control	MC_MoveVelocity <sup>*1</sup>	
		MC_TorqueControl	Torque control	MC_TorqueControl <sup>*1</sup>	

\*1 It is not supported by the I/O mode.



## Label access

### ■ Motion mode

Function block name for PLCopen Motion control		Label name	Description	
MC_ReadAxisError		AxisErrorNo	[Md.23] Axis error No.	
MC_ReadParameter	CommandedPosition	TargetPosition	[Md.32] Target value	
	SWLimitPos	SoftwareStrokeUpperLimit	[Pr.12] Software stroke limit upper limit value	
	SWLimitNeg	SoftwareStrokeLowerLimit	[Pr.13] Software stroke limit lower limit value	
	MaxVelocityAppl	SpeedLimitValue	[Pr.8] Speed limit value	
	ActualVelocity		SpeedDuringCommand	[Md.122] Speed during command (during speed control only)
			AxisCommandSpeed	[Md.28] Axis feedrate (except for during speed control)
		MotorRotationSpeed	[Md.103] Motor rotation speed	
MC_WriteParameter	MaxVelocityAppl	SpeedLimitValue	[Pr.8] Speed limit value	
MC_ReadDigitalInput		Refer to MMC_lo class for details of labels for accessing the link device (☞ Page 87 MMC_lo Class)	Link device (RX area, RY area, RWw area, RWr area)	
MC_ReadDigitalOutput				
MC_WriteDigitalOutput				
MC_ReadActualPosition		CommandPosition	[Md.20] Feed current value	
		MachineCommandPosition	[Md.21] Machine feed value	
		ActualPosition	[Md.101] Real current value	
MC_ReadActualVelocity		SpeedDuringCommand	[Md.122] Speed during command (during speed control only)	
		AxisCommandSpeed	[Md.28] Axis feedrate (except for during speed control)	
MC_ReadActualTorque		TorqueDuringCommand	[Md.123] Torque during command (during torque control only)	
MC_ReadAxisInfo	HomeAbsSwitch	Axis□_DOG <sup>*1</sup>	Proximity dog signal ("[Md.30] External input signal": b6)	
	LimitSwitchPos	Axis□_FLS <sup>*1</sup>	Upper limit signal ("[Md.30] External input signal": b1)	
	LimitSwitchNeg	Axis□_RLS <sup>*1</sup>	Lower limit signal ("[Md.30] External input signal": b0)	
	CommunicationReady <sup>*2</sup>	AxisOperationStatus	[Md.26] Axis operation status	
	ReadyForPowerOn	ServoStatus1_Ready	READY ON ("[Md.108] Servo status1": b0)	
	PowerOn	ServoStatus1_Servo	Servo ON ("[Md.108] Servo status1": b1)	
	IsHomed <sup>*3</sup>	Status_HomingRequest	Home position return request flag ("[Md.31] Status": b3)	
	AxisWarning	Status_Warning	Axis warning detection ("[Md.31] Status": b9)	

\*1 □=1 to 16: Axis No.

\*2 The correspondence of CommunicationReady of MC\_ReadAxisInfo and "[Md.26] Axis operation status" is shown below.

CommunicationReady	[Md.26] Axis operation status
OFF	20
ON	Other than 20

\*3 The correspondence of IsHomed of MC\_ReadAxisInfo and Home position return request flag ("[Md.31] Status": b3) is shown below.

IsHomed	Home position return request flag
OFF	ON
ON	OFF

### ■ I/O mode

Function block name for PLCopen Motion control		Label name	Description
MC_ReadDigitalInput		Refer to MMC_lo class for details of labels for accessing the link device (☞ Page 87 MMC_lo Class)	Link device (RX area, RY area, RWw area, RWr area)
MC_ReadDigitalOutput			
MC_WriteDigitalOutput			

# MC\_FunctionBlock::Update

Updates the I/O data of the function block. Repeat the calling of this method for the function block to be executed until processing finishes.

```
void Update(  
    void  
);
```

## Detailed description

### ■Parameter

None.

### ■Return value

None.

### ■Point

- Call this method after setting the input data to the member variable.
- Output data such as error codes are stored to the member variable after executing this method.

### ■Supported version

API version	Software version
1.00	01

### ■Reference

None.

# MC\_Power class

Changes the servo amplifier of the specified axis to an operable state.

```
class MC_Power : public MC_FunctionBlock
```

## Member

### Input

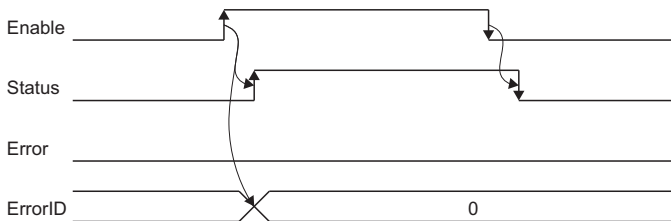
Name	Variable name	Data type	Fetch	Valid range	Description
Axis information	Axis	AXIS_REF*	—	—	Specifies the pointer to the axis information class object.
Enable	Enable	bool	Startup only	ON/OFF	While Enable is turned ON, axis control is enabled.

### Output

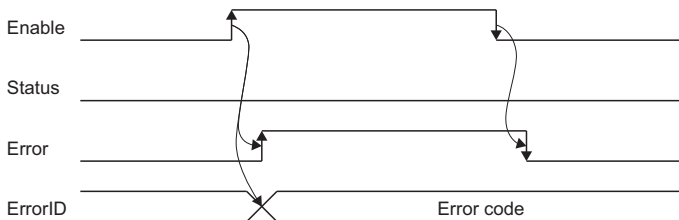
Name	Variable name	Data type	Initial value	Description
Operable	Status	bool	OFF	Indicates an operable state.
Error	Error	bool	OFF	When ON, this indicates an error occurrence in the function block.
Error code	ErrorID	unsigned short	0	Returns the error code of the error that occurred in the function block.

## Timing chart

### Normal completion




### Abnormal completion



## Error code

Value	Description
MMERR_FB_READY_SIGNAL_OFF	The READY signal [X0] is OFF.
MMERR_FB_SERVO_POWER_OFF	The control circuit power supply of the servo amplifier is OFF, or the servo amplifier is not connected.
MMERR_FB_AXIS_NUMBER	A value outside the range is set to the axis No.
MMERR_FB_UNIT	A value outside the range is set to the unit.
MMERR_FB_SERVO_PARAMETER_INACCESSIBLE	The servo parameter setting cannot be referred to.
MMERR_FB_LINK_DEVICE_INACCESSIBLE	Access to the link devices is disabled.

## Point

- When Enable turns ON, the function block initializes the information of the specified axis. Always use this function block when using the function blocks described in this manual.
- While Enable input is ON, the selected axis switches to a servo ON state.
- The axis state (FbAxisStatus) transitions from Disabled state to Standstill state.
- When the control circuit power supply of the servo amplifier is disconnected, the axis state (FbAxisStatus) transitions to Errorstop state.
- Refer to axis information class state diagram for axis state (FbAxisStatus). ( Page 147 Axis information class state diagram)

<I/O mode>

- When an error has occurred in this function block, no axis is controllable. Errors will occur in other function blocks and the function blocks will not function.

## Supported version

API version	Software version
1.10	01

## Inheritance hierarchy

MC\_FunctionBlock

MC\_Power

# MCv\_Home class

Executes home position return for the specified axis.

```
class MCv_Home : public MC_FunctionBlock
```

## Member

### Input

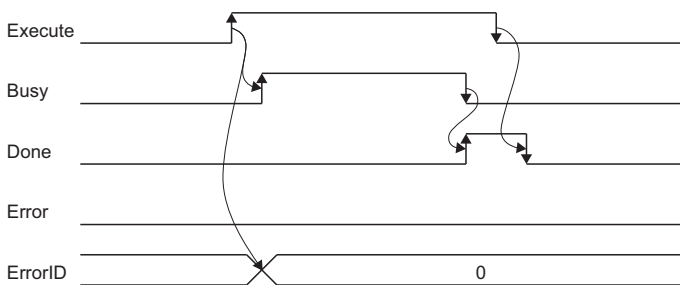
Name	Variable name	Data type	Fetch	Valid range	Description
Axis information	Axis	AXIS_REF*	—	—	Specifies the pointer to the axis information class object.
Execute command	Execute	bool	Startup only	ON/OFF	Executes the function block when ON.

### Output

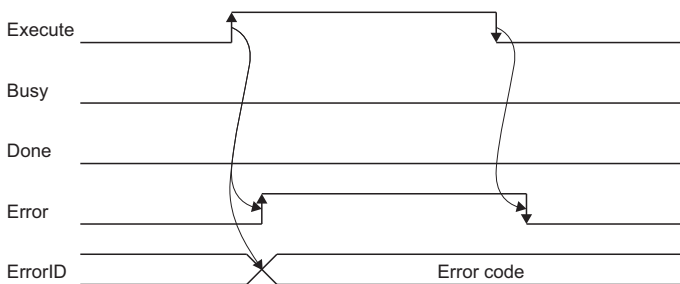
Name	Variable name	Data type	Initial value	Description
Execution complete	Done	bool	OFF	Indicates the completion of home position return.
Executing	Busy	bool	OFF	Indicates home position return is in operation.
Error	Error	bool	OFF	When ON, this indicates an error occurrence in the function block.
Error code	ErrorID	unsigned short	0	Returns the error code of the error that occurred in the function block.

## Timing chart

### Normal completion



### Abnormal completion



## Error code

Value	Description
MMERR_FB_AXIS_ERROR	An error occurred.
MMERR_FB_START_NOT_POSSIBLE	Positioning can not be started.
MMERR_FB_AXIS_STOPPED	The axis is in a Stopping state, or the stop command for the axis is ON.
MMERR_FB_AXIS_NUMBER	A value outside the range is set to the axis No.
MMERR_FB_LINK_DEVICE_INACCESSIBLE	Access to the link devices is disabled.
MMERR_FB_WARNING	A warning occurred.

## Point

- This function block executes the home position return of the specified axis based on the set home position return parameters.
- When Execute=ON, the function block is executed. During home position operation Busy is ON.
- When processing is completed normally, Done turns ON and Busy turns OFF.
- When the axis state (FbAxisStatus) is in Standstill state at the start, it is a Standstill state at completion.
- The home position return parameter must be set beforehand.

## Supported version

API version	Software version
1.10	01

## Inheritance hierarchy

MC\_FunctionBlock  
  MCv\_Home

# MC\_Stop class

Stops the specified axis.

```
class MC_Stop : public MC_FunctionBlock
```

## Member

### Input

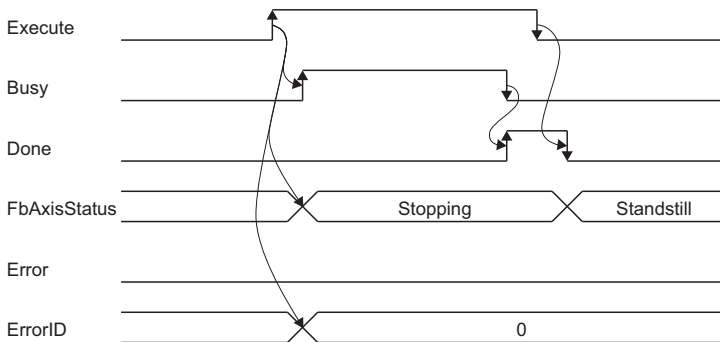
Name	Variable name	Data type	Fetch	Valid range	Description
Axis information	Axis	AXIS_REF*	—	—	Specifies the pointer to the axis information class object.
Execute command	Execute	bool	Startup only	ON/OFF	Executes the function block when ON.

### Output

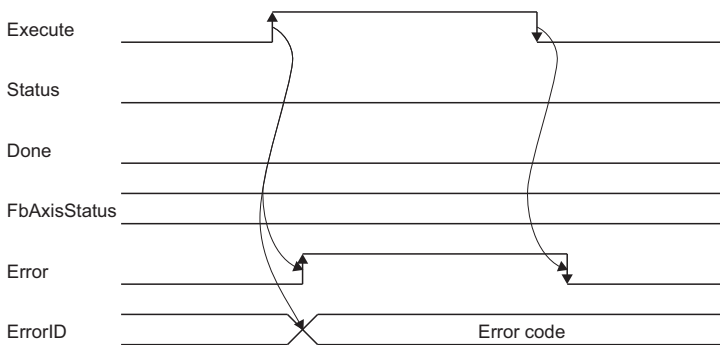
Name	Variable name	Data type	Initial value	Description
Execution complete	Done	bool	OFF	Indicates a speed of 0.
Executing	Busy	bool	OFF	Indicates speed is decreasing to 0.
Error	Error	bool	OFF	When ON, this indicates an error occurrence in the function block.
Error code	ErrorID	unsigned short	0	Returns the error code of the error that occurred in the function block.

## Timing chart

### Normal completion



### Abnormal completion



## Error code

Value	Description
MMERR_FB_AXIS_ERROR	An error occurred.
MMERR_FB_AXIS_NUMBER	A value outside the range is set to the axis No.
MMERR_FB_LINK_DEVICE_INACCESSIBLE	Access to the link devices is disabled.

## Point

- This function block stops the control of the specified axis and transitions to the Stopping state. The operation function blocks being executed are cancelled.
- When Execute=ON, the function block is executed. If processing starts normally, Busy turns ON.
- When processing is completed and the axis stops, Done turns ON.
- Until the speed of the axis is 0, no other function blocks can be executed.
- The axis state (FbAxisStatus) transitions to the Stopping state. While Execute=ON, or while speed is other than 0, the function block remains in a Stopping state. When Done turns ON, and Execute turns OFF, the function block transitions to a Standstill state.
- The deceleration time at stopping is the deceleration time specified in the positioning control function block (MC\_MoveAbsolute, MC\_MoveRelative, MC\_MoveAdditive), or continuous control function block (MC\_MoveVelocity) being executed. During torque control (MC\_TorqueControl), the function block stops immediately.

## Supported version

API version	Software version
1.10	01

## Inheritance hierarchy

MC\_FunctionBlock  
MC\_Stop



# MC\_MoveAbsolute class

Specifies the absolute target position of the specified axis and executes positioning.

```
class MC_MoveAbsolute : public MC_FunctionBlock
```

## Member

### Input

Name	Variable name	Data type	Fetch	Valid range	Description
Axis information	Axis	AXIS_REF*	—	—	Specifies the pointer to the axis information class object.
Execute command	Execute	bool	Startup only	ON/OFF	Executes the function block when ON.
Positioning data No.	PositionDataNo	short		1 to 100	<Motion mode> Specifies the positioning data No. that stores the positioning data. <I/O mode> Specifies the point table No. that stores the positioning data.
Target position	Position	double		*1	Sets the absolute target position.
Speed	Velocity	double			Sets the speed command value at the time of positioning.
Acceleration time	Acceleration	long		1 to 8338608 [ms]	<Motion mode> Sets the time to reach the speed limit value from speed 0. <I/O mode> Sets the time to reach the rated rotation speed of the servo motor.
Deceleration time	Deceleration	long			<Motion mode> Sets the time to speed 0 from the speed limit value. <I/O mode> Sets the time to stop from the rated rotation speed of the servo motor.
Rotation direction	Direction	short		1: Forward 2: Reverse 3: Shortest path	<Motion mode> Specifies rotation direction. <I/O mode> Any setting value is ignored.

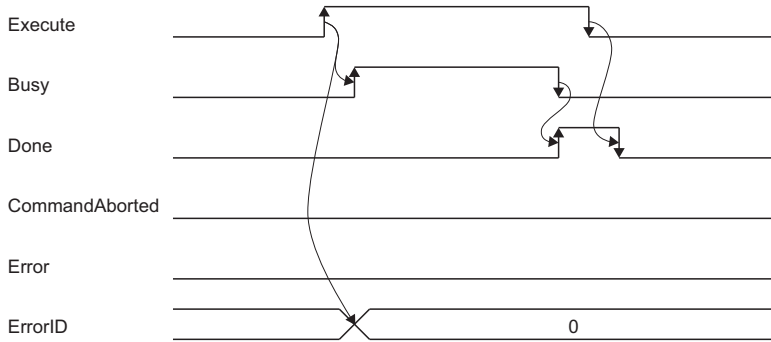
\*1 Refer to function block units. (Page 149 Function block units)

### Output

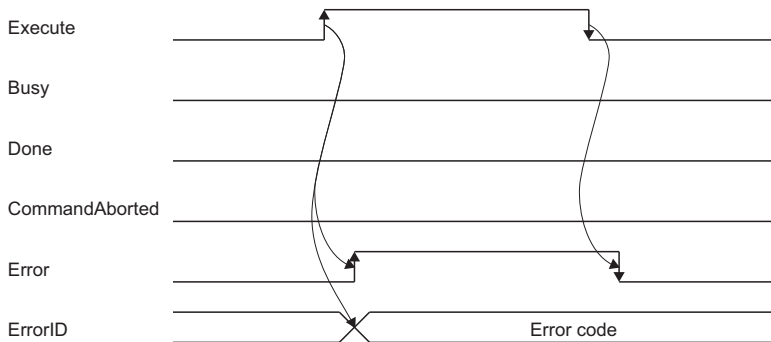
Name	Variable name	Data type	Initial value	Description
Execution complete	Done	bool	OFF	Indicates the target position has been reached.
Executing	Busy	bool	OFF	Indicates the function block is in operation.
Execution cancel	CommandAborted	bool	OFF	Indicates the execution is cancelled by another function block.
Error	Error	bool	OFF	When ON, this indicates an error occurrence in the function block.
Error code	ErrorID	unsigned short	0	Returns the error code of the error that occurred in the function block.

## Timing chart

### ■ Normal completion



### ■ Abnormal completion



## Error code

Value	Description
MMERR_FB_AXIS_ERROR	An error occurred.
MMERR_FB_START_NOT_POSSIBLE	Positioning can not be started.
MMERR_FB_AXIS_STOPPED	The axis is in a Stopping state, or the stop command for the axis is ON.
MMERR_FB_AXIS_NUMBER	A value outside the range is set to the axis No.
MMERR_FB_POSITIONIG_DATA_NUMBER	A value outside the range is set to the positioning data No.
MMERR_FB_POSITION	A value outside the range is set to the target position, and movement amount.
MMERR_FB_VELOCITY	A value outside the range is set to the speed.
MMERR_FB_ACCDEC	A value outside the range is set to the acceleration/deceleration time.
MMERR_FB_DIRECTION	A value outside the range is set to the rotation direction.
MMERR_FB_READ_OR_WRITE_PARAMETER	Reading or writing the parameter failed.
MMERR_FB_INSTRUCTION_RUNNING	Another function block is executing the instruction code.
MMERR_FB_LINK_DEVICE_INACCESSIBLE	Access to the link devices is disabled.
MMERR_FB_WARNING	A warning occurred.

## Point

- This function block executes positioning of the specified axis to the set absolute target position.
- When Execute=ON, the function block is executed. If processing starts normally, Busy turns ON.
- When processing is completed and the axis completes positioning, Done turns ON.
- When the path to the target position is determined uniquely, Direction input is ignored.
- The axis state (FbAxisStatus) during positioning control transitions to the DiscreteMotion state.

### <Motion mode>

- When executing this function block during the execution of continuous control function blocks (MC\_MoveVelocity, MC\_TorqueControl), execute with the axis in a stopped state.
- This class uses one point of positioning data. Use positioning data by setting a position data No. (a No. that is not being used for any other application) to be used in this class.
- When used with positioning operation function blocks, the operation for function blocks that are executed after is the operation of the target position change function. When the accumulated movement amount from the position of execution of the first function block exceeds "-2147483648 to 2147483647", an error occurs.
- When executing the function block in a state with "[Cd.183] Execution prohibition flag" turned ON, start the positioning control by turning OFF "[Cd.183] Execution prohibition flag" while the function block execute command (Execute) is ON. If the function block execute command (Execute) is turned OFF before "[Cd.183] Execution prohibition flag" is turned OFF, the function block is in a state where it has received start, and positioning control cannot be cancelled by turning OFF Execute only. When cancelling, use MC\_Stop.

### <I/O mode>

- This class uses one point of a point table. Use a point table No. by setting a No. (that is not being used for any other application) to be used in this class.
- For an axis on which this function block is in execution, another MC\_MoveAbsolute or MC\_MoveRelative cannot be executed. Doing so causes an error for the function block executed later, and the function block in execution continues the operation.
- If the function block has been completed in the state where Error=ON and ErrorID=MMERR\_FB\_WARNING by a warning, clearing the warning will clear ErrorID to zero.

## Supported version

API version	Software version
1.10	01

## Inheritance hierarchy

```

MC_FunctionBlock
  MC_MoveAbsolute
  
```

# MC\_MoveRelative class

Moves the specified distance from the current position.

```
class MC_MoveRelative : public MC_FunctionBlock
```

## Member

### Input

Name	Variable name	Data type	Fetch	Valid range	Description
Axis information	Axis	AXIS_REF*	—	—	Specifies the pointer to the axis information class object.
Execute command	Execute	bool	Startup only	ON/OFF	Executes the function block when ON.
Positioning data No.	PositionDataNo	short		1 to 100	<Motion mode> Specifies the positioning data No. that stores the positioning data. <I/O mode> Specifies the point table No. that stores the positioning data.
Movement amount	Distance	double		*1	Sets the movement amount.
Speed	Velocity	double			Sets the speed command value at the time of positioning.
Acceleration time	Acceleration	long		1 to 8338608 [ms]	<Motion mode> Sets the time to reach the speed limit value from speed 0. <I/O mode> Sets the time to reach the rated rotation speed of the servo motor.
Deceleration time	Deceleration	long			<Motion mode> Sets the time to speed 0 from the speed limit value. <I/O mode> Sets the time to stop from the rated rotation speed of the servo motor.

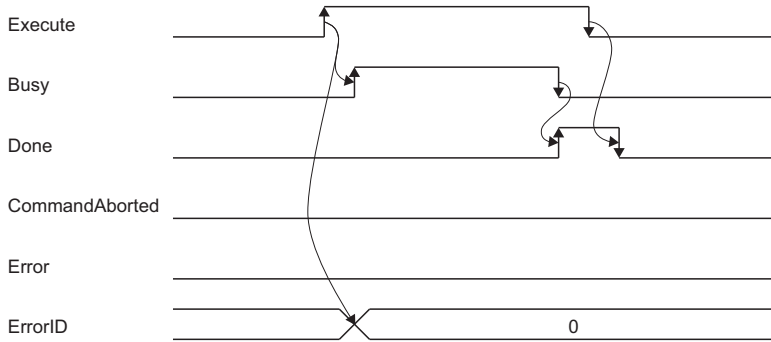
\*1 Refer to function block units. (☞ Page 149 Function block units)

### Output

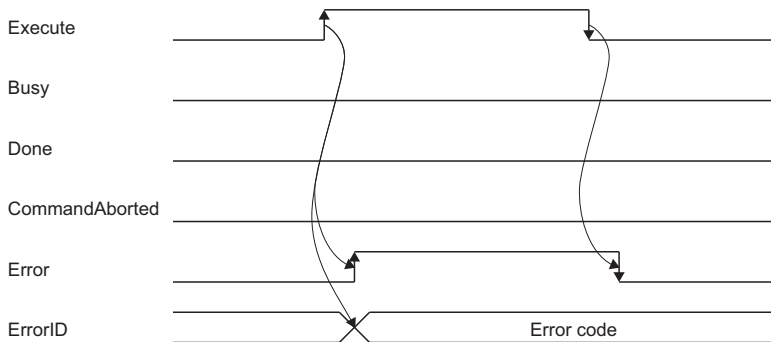
Name	Variable name	Data type	Initial value	Description
Execution complete	Done	bool	OFF	Indicates the target position has been reached.
Executing	Busy	bool	OFF	Indicates the function block is in operation.
Execution cancel	CommandAborted	bool	OFF	Indicates the execution is cancelled by another function block.
Error	Error	bool	OFF	When ON, this indicates an error occurrence in the function block.
Error code	ErrorID	unsigned short	0	Returns the error code of the error that occurred in the function block.

## Timing chart

### ■ Normal completion



### ■ Abnormal completion



## Error code

Value	Description
MMERR_FB_AXIS_ERROR	An error occurred.
MMERR_FB_START_NOT_POSSIBLE	Positioning can not be started.
MMERR_FB_AXIS_STOPPED	The axis is in a Stopping state, or the stop command for the axis is ON.
MMERR_FB_AXIS_NUMBER	A value outside the range is set to the axis No.
MMERR_FB_POSITIONING_DATA_NUMBER	A value outside the range is set to the positioning data No.
MMERR_FB_POSITION	A value outside the range is set to the target position, and movement amount.
MMERR_FB_VELOCITY	A value outside the range is set to the speed.
MMERR_FB_ACCDEC	A value outside the range is set to the acceleration/deceleration time.
MMERR_FB_READ_OR_WRITE_PARAMETER	Reading or writing the parameter failed.
MMERR_FB_INSTRUCTION_RUNNING	Another function block is executing the instruction code.
MMERR_FB_LINK_DEVICE_INACCESSIBLE	Access to the link devices is disabled.
MMERR_FB_WARNING	A warning occurred.

## Point

- This function block moves the specified distance from the current command position of the specified axis.
- When Execute=ON, the function block is executed. If processing starts normally, Busy turns ON.
- When processing is completed and the axis completes positioning, Done turns ON.
- The axis state (FbAxisStatus) during positioning control transitions to the DiscreteMotion state.

### <Motion mode>

- When executing this function block during the execution of continuous control function blocks (MC\_MoveVelocity, MC\_TorqueControl), execute with the axis in a stopped state.
- This class uses one point of positioning data. Use positioning data by setting a position data No. (a No. that is not being used for any other application) to be used in this class.
- When used with positioning operation function blocks, the operation for function blocks that are executed after is the operation of the target position change function. When the accumulated movement amount from the position of execution of the first function block exceeds "-2147483648 to 2147483647", an error occurs.
- When executing the function block in a state with "[Cd.183] Execution prohibition flag" turned ON, start the positioning control by turning OFF "[Cd.183] Execution prohibition flag" while the function block execute command (Execute) is ON. If the function block execute command (Execute) is turned OFF before "[Cd.183] Execution prohibition flag" is turned OFF, the function block is in a state where it has received start, and positioning control cannot be cancelled by turning OFF Execute only. When cancelling, use MC\_Stop.

### <I/O mode>

- This class uses one point of a point table. Use a point table No. by setting a No. (that is not being used for any other application) to be used in this class.
- For an axis on which this function block is in execution, another MC\_MoveRelative or MC\_MoveAbsolute cannot be executed. Doing so causes an error for the function block executed later, and the function block in execution continues the operation.
- If the function block has been completed in the state where Error=ON and ErrorID=MMERR\_FB\_WARNING by a warning, clearing the warning will clear ErrorID to zero.

## Supported version

API version	Software version
1.10	01

## Inheritance hierarchy

MC\_FunctionBlock  
  MC\_MoveRelative

# MC\_Reset class

Cancels the errors of the specified axis.

```
class MC_Reset : public MC_FunctionBlock
```

## Member

### Input

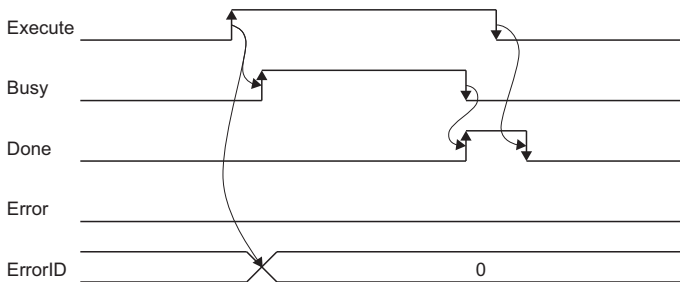
Name	Variable name	Data type	Fetch	Valid range	Description
Axis information	Axis	AXIS_REF*	—	—	Specifies the pointer to the axis information class object.
Execute command	Execute	bool	Startup only	ON/OFF	Executes the function block when ON.

### Output

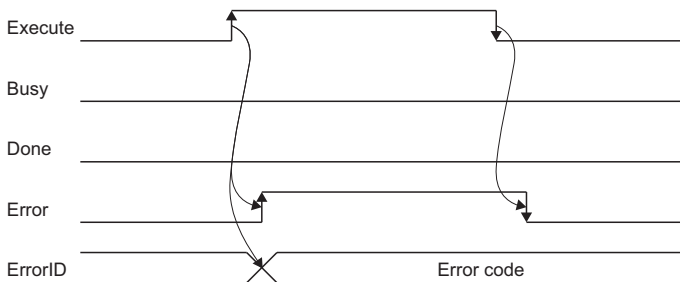
Name	Variable name	Data type	Initial value	Description
Execution complete	Done	bool	OFF	Indicates the reset is complete.
Executing	Busy	bool	OFF	Indicates the function block is in operation.
Error	Error	bool	OFF	When ON, this indicates an error occurrence in the function block.
Error code	ErrorID	unsigned short	0	Returns the error code of the error that occurred in the function block.

## Timing chart

### Normal completion



### Abnormal completion



## Error code

Value	Description
MMERR_FB_AXIS_NUMBER	A value outside the range is set to the axis No.
MMERR_FB_LINK_DEVICE_INACCESSIBLE	Access to the link devices is disabled.

## Point

- When Execute=ON, the function block is executed. If processing starts normally, Busy turns ON.
- When the cancelling of errors of the axis is completed, Done turns ON.
- When execute is turned ON with the error factor of the axis still remaining, the error is not cancelled. In this case Busy stays ON. Turn OFF Execute, remove the error factor, then turn Execute ON again. For how to eliminate error causes, refer to the following or the servo amplifier instruction manual.

 Simple Motion Board User's Manual (Application)

<Motion mode>

- This function block also clears warnings.

## Supported version

API version	Supported version
1.10	01

## Inheritance hierarchy

MC\_FunctionBlock

MC\_Reset



# MC\_MoveAdditive class

Adds the specified relative position to the most recent positioning command of the specified axis, and executes positioning.

```
class MC_MoveAdditive : public MC_FunctionBlock
```

## Member

### Input

Name	Variable name	Data type	Fetch	Valid range	Description
Axis information	Axis	AXIS_REF*	—	—	Specifies the pointer to the axis information class object.
Execute command	Execute	bool	Startup only	ON/OFF	Executes the function block when ON.
Positioning data No.	PositionDataNo	short		1 to 100	Specifies the positioning data No. that stores the positioning data.
Movement amount	Distance	double		*1	Sets the relative movement amount.
Target speed	Velocity	double			Sets the feedrate at the time of positioning.
Acceleration time	Acceleration	long		1 to 8338608 [ms]	Sets the time to reach the speed limit value from speed 0.
Deceleration time	Deceleration	long			Sets the time to speed 0 from the speed limit value.

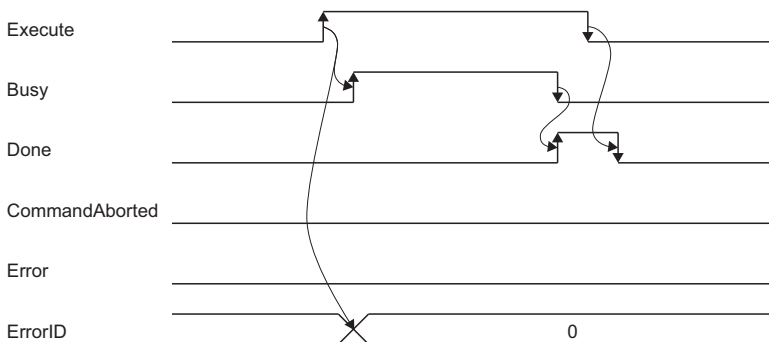
\*1 Refer to function block units. (Page 149 Function block units)

### Output

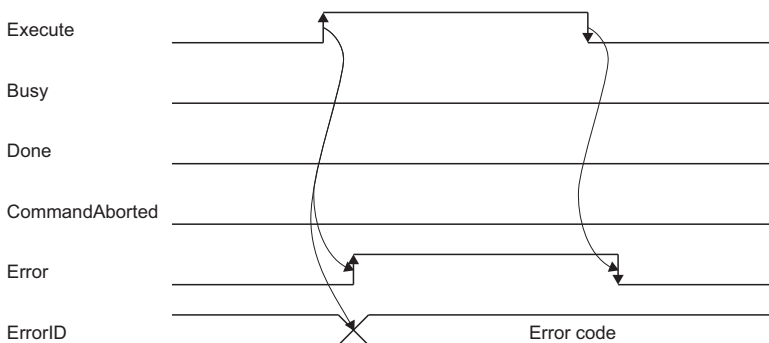
Name	Variable name	Data type	Initial value	Description
Execution complete	Done	bool	OFF	Indicates the target position has been reached.
Executing	Busy	bool	OFF	Indicates the function block is in operation.
Execution cancel	CommandAborted	bool	OFF	Indicates the execution is cancelled by another function block.
Error	Error	bool	OFF	When ON, this indicates an error occurrence in the function block.
Error code	ErrorID	unsigned short	0	Returns the error code of the error that occurred in the function block.

## Timing chart

### Normal completion



### Abnormal completion



## Error code

Value	Description
MMERR_FB_AXIS_ERROR	An error occurred.
MMERR_FB_START_NOT_POSSIBLE	Positioning can not be started.
MMERR_FB_AXIS_STOPPED	The axis is in a Stopping state, or the stop command for the axis is ON.
MMERR_FB_AXIS_NUMBER	A value outside the range is set to the axis No.
MMERR_FB_POSITIONIG_DATA_NUMBER	A value outside the range is set to the positioning data No.
MMERR_FB_POSITION	A value outside the range is set to the target position, and movement amount.
MMERR_FB_VELOCITY	A value outside the range is set to the speed.
MMERR_FB_ACCDEC	A value outside the range is set to the acceleration/deceleration time.

## Point

- This function block adds the specified relative position to the most recent positioning command of the specified axis, and executes positioning.
- When Execute=ON, the function block is executed. If processing starts normally, Busy turns ON.
- When processing is completed and the axis completes positioning, Done turns ON.
- The axis state (FbAxisStatus) during positioning control is the DiscreteMotion state.
- This function block can be used in a Standstill state or DiscreteMotion state. It cannot be used in the ContinuousMotion state.
- When executing this function block during the execution of continuous control function blocks (MC\_MoveVelocity, MC\_TorqueControl), execute with the axis in a stopped state.
- This class uses one point of positioning data. Use positioning data by setting a position data No. (a No. that is not being used for any other application) to be used in this class.
- When used with positioning operation function blocks, the operation for function blocks that are executed after is the operation of the target position change function. When the accumulated movement amount from the position of execution of the first function block exceeds "-2147483648 to 2147483647", an error occurs.
- When executing the function block in a state with "[Cd.183] Execution prohibition flag" turned ON, start the positioning control by turning OFF "[Cd.183] Execution prohibition flag" while the function block execute command (Execute) is ON. If the function block execute command (Execute) is turned OFF before "[Cd.183] Execution prohibition flag" is turned OFF, the function block is in a state where it has received start, and positioning control cannot be cancelled by turning OFF Execute only. When cancelling, use MC\_Stop.

## Supported version

API version	Software version
1.00	01

## Inheritance hierarchy

MC\_FunctionBlock  
  MC\_MoveAdditive

# MC\_MoveVelocity class

Executes speed control for the specified axis at the specified speed.

```
class MC_MoveVelocity : public MC_FunctionBlock
```

## Member

### Input

Name	Variable name	Data type	Fetch	Valid range	Description
Axis information	Axis	AXIS_REF*	—	—	Specifies the pointer to the axis information class object.
Execute command	Execute	bool	Startup only	ON/OFF	Executes the function block when ON.
Target speed	Velocity	double		*1	Sets the command speed.
Acceleration time	Acceleration	unsigned short		0 to 65535	Sets the time to reach the speed limit value from speed 0.
Deceleration time	Deceleration	unsigned short		[ms]	Sets the time to speed 0 from the speed limit value.
Rotation direction	Direction	short		1: Forward 2: Reverse	Specifies rotation direction.

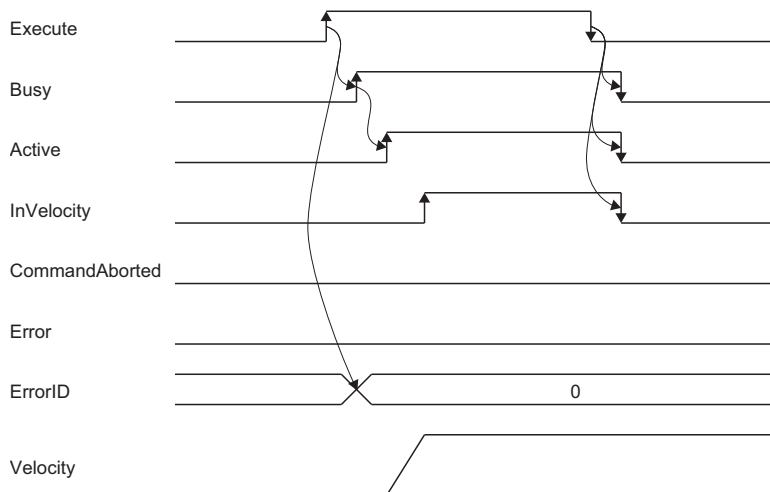
\*1 Refer to function block units. (Page 149 Function block units)

### Output

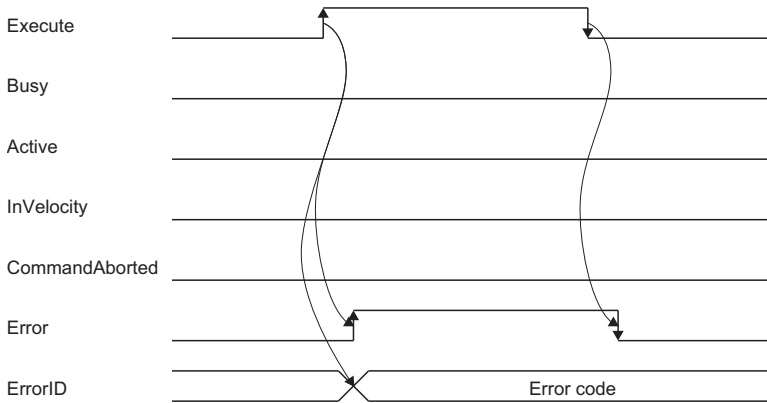
Name	Variable name	Data type	Initial value	Description
Target speed reached	InVelocity	bool	OFF	Indicates the specified speed has been reached.
Executing	Busy	bool	OFF	Indicates the function block is in operation.
Controlling	Active	bool	OFF	Indicates the function block is controlling the axis.
Execution cancel	CommandAborted	bool	OFF	Indicates the execution is cancelled by another function block.
Error	Error	bool	OFF	When ON, this indicates an error occurrence in the function block.
Error code	ErrorID	unsigned short	0	Returns the error code of the error that occurred in the function block.

## Timing chart

### Normal completion



## ■ Abnormal completion



## Error code

Value	Description
MMERR_FB_AXIS_ERROR	An error occurred.
MMERR_FB_START_NOT_POSSIBLE	Positioning can not be started.
MMERR_FB_AXIS_STOPPED	The axis is in a Stopping state, or the stop command for the axis is ON.
MMERR_FB_AXIS_NUMBER	A value outside the range is set to the axis No.
MMERR_FB_VELOCITY	A value outside the range is set to the speed.
MMERR_FB_ACCDEC	A value outside the range is set to the acceleration/deceleration time.
MMERR_FB_DIRECTION	A value outside the range is set to the rotation direction.

## Point

- When Execute=ON, the function block is executed. If processing starts normally, Busy turns ON.
- When the axis is in speed control mode, Active turns ON. When the target speed is reached, InVelocity turns ON. Once the target speed is reached and InVelocity turns ON, the state is retained until Execute is turned OFF, or control is cancelled.
- The axis state (FbAxisStatus) is the ContinuousMotion state.
- The execution is cancelled and control changes with a new MC\_MoveVelocity or MC\_TorqueControl.
- To stop operation use MC\_Stop. When control is cancelled, CommandAborted turns ON.
- CommandAborted is turned OFF by Execute=OFF.
- This function block cannot be executed while positioning control function blocks (MC\_MoveAbsolute, MC\_MoveRelative, MC\_MoveAdditive) are being executed.
- When switching from speed control mode to torque control mode the motor speed may temporarily fluctuate. For this reason, switching from speed control mode to torque control mode is done after motor is stopped.

## Supported version

API version	Software version
1.00	01

## Inheritance hierarchy

MC\_FunctionBlock  
 MC\_MoveVelocity

# MC\_TorqueControl class

Executes torque control for the specified axis at the specified torque.

```
class MC_TorqueControl : public MC_FunctionBlock
```

## Member

### Input

Name	Variable name	Data type	Fetch	Valid range	Description
Axis information	Axis	AXIS_REF*	—	—	Specifies the pointer to the axis information class object.
Execute command	Execute	bool	Startup only	ON/OFF	Executes the function block when ON.
Target torque	Torque	double		-1000.0 to 1000.0[%]	Sets command torque. Set the ratio in units of % against the rated torque of the servo motor being used.
Forward torque time constant	TorqueRampFwd	unsigned short		0 to 65535 [ms]	Sets the time to reach the torque limit value from torque 0.
Reverse torque time constant	TorqueRampRev	unsigned short			Sets the time to torque 0 from the torque limit value.
Speed limit	Velocity	double		*1	Sets the speed limit value when in torque control mode.
Rotation direction	Direction	short		1: Forward 2: Reverse	Specifies rotation direction.

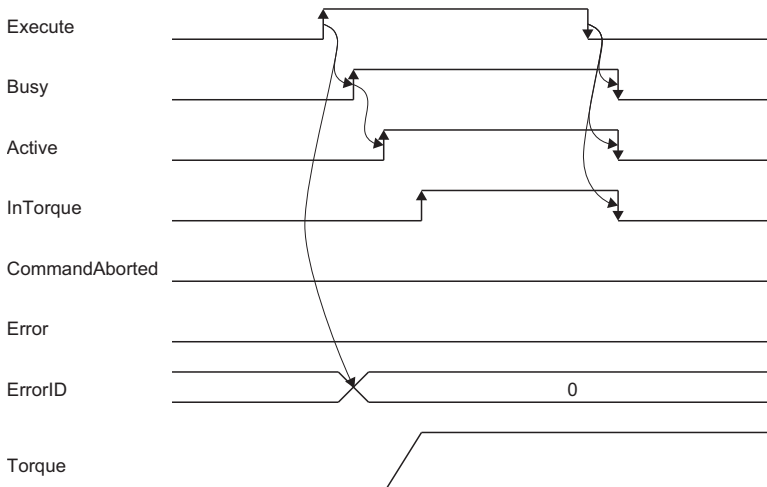
\*1 Refer to function block units. (☞ Page 149 Function block units)

### Output

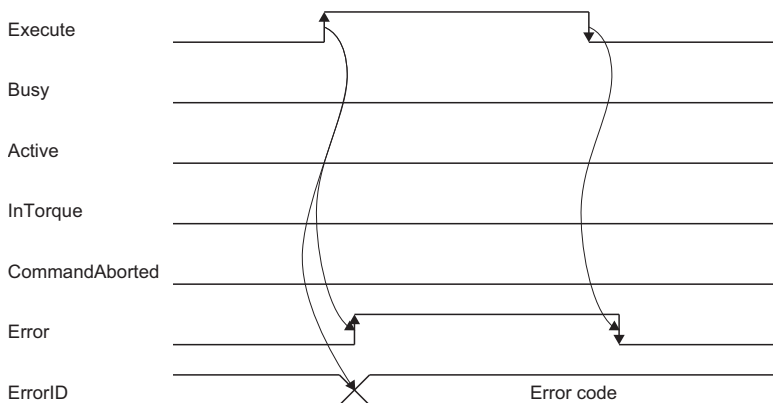
Name	Variable name	Data type	Initial value	Description
Target torque reached	InTorque	bool	OFF	Indicates the specified torque has been reached.
Executing	Busy	bool	OFF	Indicates the function block is in operation.
Controlling	Active	bool	OFF	Indicates the function block is controlling the axis.
Execution cancel	CommandAborted	bool	OFF	Indicates the execution is cancelled by another function block.
Error	Error	bool	OFF	When ON, this indicates an error occurrence in the function block.
Error code	ErrorID	unsigned short	0	Returns the error code of the error that occurred in the function block.

## Timing chart

### Normal completion



## ■ Abnormal completion



## Error code

Value	Description
MMERR_FB_AXIS_ERROR	An error occurred.
MMERR_FB_START_NOT_POSSIBLE	Positioning can not be started.
MMERR_FB_AXIS_STOPPED	The axis is in a Stopping state, or the stop command for the axis is ON.
MMERR_FB_AXIS_NUMBER	A value outside the range is set to the axis No.
MMERR_FB_VELOCITY	A value outside the range is set to the speed.
MMERR_FB_ACCDEC	A value outside the range is set to the acceleration/deceleration time.
MMERR_FB_DIRECTION	A value outside the range is set to the rotation direction.
MMERR_FB_TORQUE	A value outside the range is set to the target torque.

## Point

- When Execute=ON, the function block is executed. If processing starts normally, Busy turns ON.
- When the axis is in torque control mode, Active turns ON. When the target torque is reached, InTorque turns ON. Once the target torque is reached and InTorque turns ON, the state is retained until Execute is turned OFF, or control is cancelled.
- The axis state (FbAxisStatus) is the ContinuousMotion state.
- The execution is cancelled and control changes with a new MC\_TorqueControl or MC\_MoveVelocity.
- To stop operation use MC\_Stop. When control is cancelled, CommandAborted turns ON.
- CommandAborted is turned OFF by Execute=OFF.
- This function block cannot be executed while positioning control function blocks (MC\_MoveAbsolute, MC\_MoveRelative, MC\_MoveAdditive) are being executed.
- The relationship between the target torque setting value and the direction in which torque occurs in the motor differs depending on the settings of servo parameters "Rotation direction selection/travel direction selection (PA14)" and "Function selection C-B POL reflection selection at torque control (PC29)". Refer to the following for details.  
[📖 Servo amplifier Instruction Manual](#)
- The rotation direction (Direction) of this function block is the direction when the servo parameter "Function selection C-B POL reflection selection at torque control (PC29)" is set to "0: Enabled".

## Supported version

API version	Software version
1.00	01

## Inheritance hierarchy

MC\_FunctionBlock  
 MC\_TorqueControl

# MC\_SetPosition class

Changes the current position (command position and feedback position) of the specified axis.

```
class MC_SetPosition : public MC_FunctionBlock
```

## Member

### Input

Name	Variable name	Data type	Fetch	Valid range	Description
Axis information	Axis	AXIS_REF*	—	—	Specifies the pointer to the axis information class object.
Execute command	Execute	bool	Startup only	ON/OFF	Executes the function block when ON.
Target position	Position	double		*1	Sets the target position.
Relative position selection	Relative	bool		ON/OFF	Sets relative distance when Relative=ON, and absolute position with Relative=OFF.

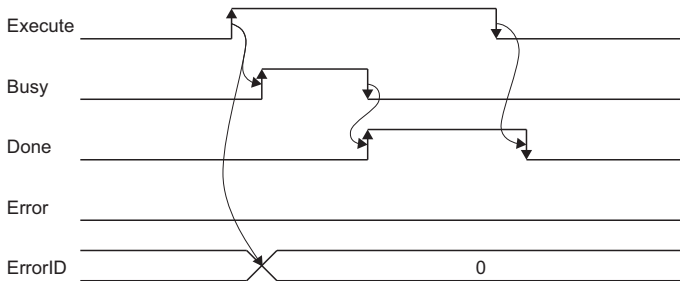
\*1 Refer to function block units. (Page 149 Function block units)

### Output

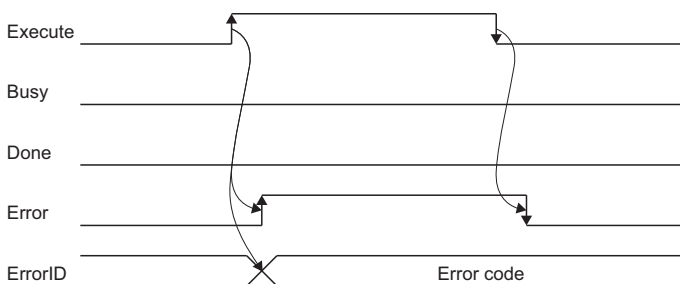
Name	Variable name	Data type	Initial value	Description
Execution complete	Done	bool	OFF	Indicates the current value change is complete.
Executing	Busy	bool	OFF	Indicates the function block is in operation.
Error	Error	bool	OFF	When ON, this indicates an error occurrence in the function block.
Error code	ErrorID	unsigned short	0	Returns the error code of the error that occurred in the function block.

## Timing chart

### Normal completion



### Abnormal completion



## Error code

Value	Description
MMERR_FB_AXIS_ERROR	An error occurred.
MMERR_FB_START_NOT_POSSIBLE	Positioning can not be started.
MMERR_FB_AXIS_STOPPED	The axis is in a Stopping state, or the stop command for the axis is ON.
MMERR_FB_AXIS_NUMBER	A value outside the range is set to the axis No.
MMERR_FB_POSITION	A value outside the range is set to the target position, and movement amount.

## Point

- When Relative is ON, the current position is changed to a position that adds the target position (relative position) to the current position.
- When Relative is OFF, the current position is changed to the target position (absolute position).
- When Execute=ON, the function block is executed. If processing starts normally, Busy turns ON.
- When processing is completed and the current position is changed, Done turns ON.
- Execute this function block with the axis in a stopped state.

## Supported version

API version	Software version
1.00	01

## Inheritance hierarchy

MC\_FunctionBlock  
MC\_SetPosition



# MC\_SetOverride class

Changes the target speed of the specified axis.

```
class MC_SetOverride : public MC_FunctionBlock
```

## Member

### Input

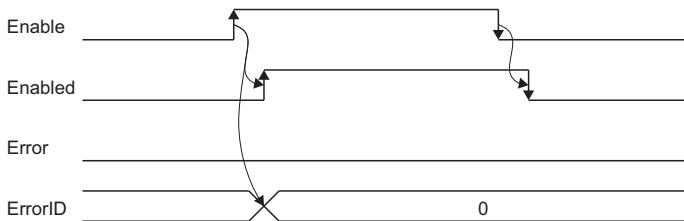
Name	Variable name	Data type	Fetch	Valid range	Description
Axis information	Axis	AXIS_REF*	—	—	Specifies the pointer to the axis information class object.
Enable	Enable	bool	Startup only	ON/OFF	Executes the function block when ON.
Speed override coefficient	VelFactor	double	All times	0.00 to 3.00	Continuously fetched while Enable=ON. Sets the override coefficient of speed.

### Output

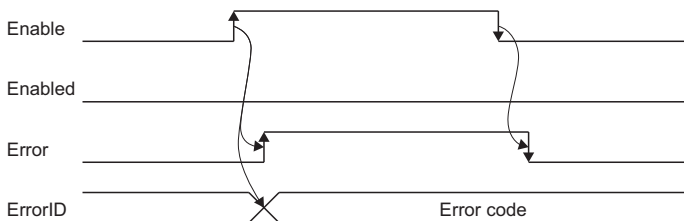
Name	Variable name	Data type	Initial value	Description
Enabled	Enabled	bool	OFF	Turns ON when override value is set normally.
Error	Error	bool	OFF	When ON, this indicates an error occurrence in the function block.
Error code	ErrorID	unsigned short	0	Returns the error code of the error that occurred in the function block.

## Timing chart

### Normal completion



### Abnormal completion



## Error code

Value	Description
MMERR_FB_AXIS_NUMBER	A value outside the range is set to the axis No.
MMERR_FB_VELOCITY_FACTOR	A value outside the range is set to the speed override coefficient.

## Point

- This function block changes the speed of the current positioning to a speed which has the override coefficient "0.00(0%) to 3.00(300%)" applied to it.
- When Enable=ON, the function block is executed. When the override coefficient is enabled, Enabled turns ON.
- The initial value of the speed override coefficient is 1.00.
- When the value 0.00 is set to the speed override coefficient, the axis stops without transitioning to the Standstill state.

## Supported version

API version	Software version
1.00	01

## Inheritance hierarchy

MC\_FunctionBlock

MC\_SetOverride

# MC\_ReadActualPosition class

Reads the current position of the specified axis.

```
class MC_ReadActualPosition : public MC_FunctionBlock
```

## Member

### ■Input

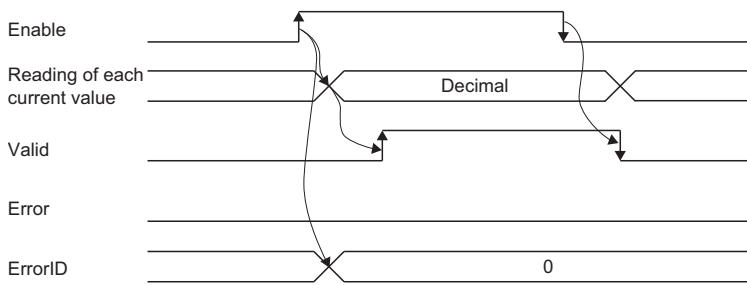
Name	Variable name	Data type	Fetch	Valid range	Description
Axis information	Axis	AXIS_REF*	—	—	Specifies the pointer to the axis information class object.
Enable	Enable	bool	Startup only	ON/OFF	Executes the function block when ON.

### ■Output

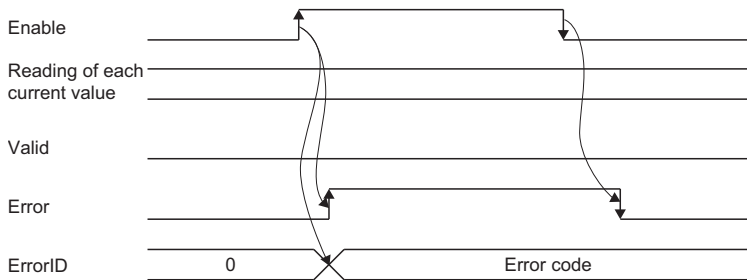
Name	Variable name	Data type	Initial value	Description
Output value valid	Valid	bool	OFF	While this device is ON, this indicates that the output value is valid.
Error	Error	bool	OFF	When ON, this indicates an error occurrence in the function block.
Error code	ErrorID	unsigned short	0	Returns the error code of the error that occurred in the function block.
Feed current value/ Command position	Position	double	0	<Motion mode> Returns the feed current value of the selected axis. </O mode> Returns the command position of the selected axis.
Machine feed value	MachinePosition	double	0	<Motion mode> Returns the machine feed value of the selected axis. </O mode> Always outputs "0".
Current position	RealPosition	double	0	<Motion mode> Returns the real current value of the selected axis. </O mode> Returns the current position of the selected axis.

## Timing chart

### Normal completion



### Abnormal completion



## Error code

Value	Description
MMERR_FB_AXIS_NUMBER	A value outside the range is set to the axis No.

## Point

- This function block reads the current position of the specified axis.  
Motion mode: Feed current value, machine feed value, real current value  
I/O mode: Commanded position, current position
- The unit of the read data is converted.
- The function block is executed when Enable=ON, and the current position is read.
- Read data is always updated while Valid=ON.

## Supported version

API version	Software version
1.10	01

## Inheritance hierarchy

MC\_FunctionBlock  
MC\_ReadActualPosition

# MC\_ReadStatus class

Returns the detailed state of the state diagram of the specified axis.

```
class MC_ReadStatus : public MC_FunctionBlock
```

## Member

### Input

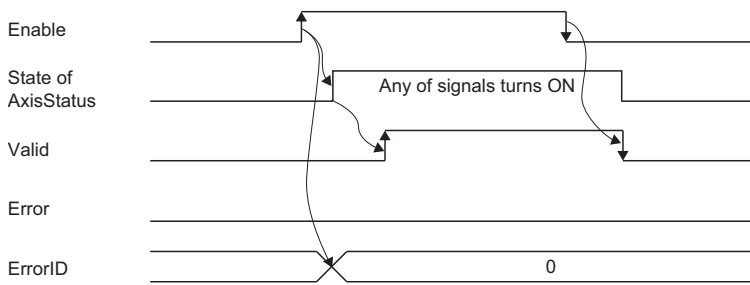
Name	Variable name	Data type	Fetch	Valid range	Description
Axis information	Axis	AXIS_REF*	—	—	Specifies the pointer to the axis information class object.
Enable	Enable	bool	Startup only	ON/OFF	Executes the function block when ON.

### Output

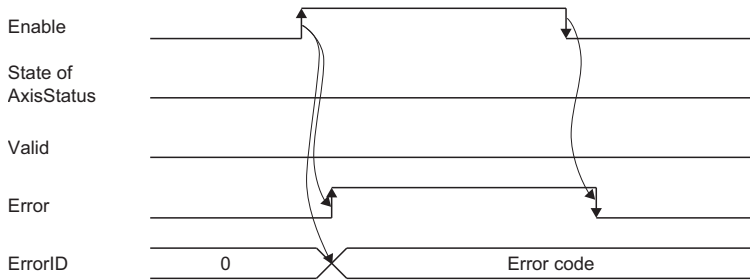
Name	Variable name	Data type	Initial value	Description
Output value valid	Valid	bool	OFF	While this device is ON, this indicates that the output value is valid.
Error	Error	bool	OFF	When ON, this indicates an error occurrence in the function block.
Error code	ErrorID	unsigned short	0	Returns the error code of the error that occurred in the function block.
ErrorStop state	ErrorStop	bool	OFF	Indicates that the axis is in the ErrorStop state. ( <a href="#">Page 147 Axis information class state diagram</a> )
Disabled state	Disabled	bool	OFF	Indicates that the axis is in the Disabled state. ( <a href="#">Page 147 Axis information class state diagram</a> )
Stopping state	Stopping	bool	OFF	Indicates that the axis is in the Stopping state. ( <a href="#">Page 147 Axis information class state diagram</a> )
Homing state	Homing	bool	OFF	Indicates that the axis is in the Homing state. ( <a href="#">Page 147 Axis information class state diagram</a> )
Standstill state	Standstill	bool	OFF	Indicates that the axis is in the Standstill state. ( <a href="#">Page 147 Axis information class state diagram</a> )
DiscreteMotion state	DiscreteMotion	bool	OFF	Indicates that the axis is in the DiscreteMotion state. ( <a href="#">Page 147 Axis information class state diagram</a> )
ContinuousMotion state	ContinuousMotion	bool	OFF	Indicates that the axis is in the ContinuousMotion state. ( <a href="#">Page 147 Axis information class state diagram</a> )
SynchronizedMotion state	SynchronizedMotion	bool	OFF	Indicates that the axis is in the SynchronizedMotion state. ( <a href="#">Page 147 Axis information class state diagram</a> )

## Timing chart

### ■ Normal completion



### ■ Abnormal completion



## Error code

Value	Description
MMERR_FB_AXIS_NUMBER	A value outside the range is set to the axis No.

## Point

- This function block reads the state of the specified axis.
- The function block is executed when Enable=ON, and the state is consecutively read.
- When the state is normally read, any of the outputs that indicates a state turns ON.
- Read data is always updated while Valid=ON.

<I/O mode>

- The output labels ContinuousMotion and SynchronizedMotion are always OFF.

## Supported version

API version	Software version
1.10	01

## Inheritance hierarchy

MC\_FunctionBlock  
 MC\_ReadStatus

# MC\_ReadAxisInfo class

Reads the axis information of the specified axis.

```
class MC_ReadAxisInfo : public MC_FunctionBlock
```

## Member

### ■Input

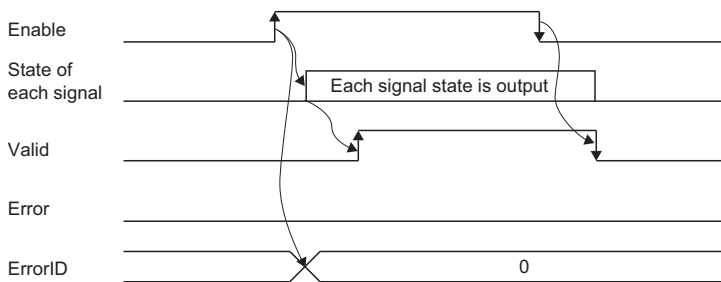
Name	Variable name	Data type	Fetch	Valid range	Description
Axis information	Axis	AXIS_REF*	—	—	Specifies the pointer to the axis information class object.
Enable	Enable	bool	Startup only	ON/OFF	Executes the function block when ON.

### ■Output

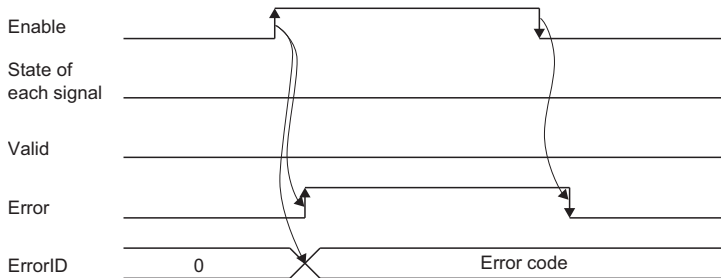
Name	Variable name	Data type	Initial value	Description
Output value valid	Valid	bool	OFF	While this device is ON, this indicates that the output value is valid.
Error	Error	bool	OFF	When ON, this indicates an error occurrence in the function block.
Error code	ErrorID	unsigned short	0	Returns the error code of the error that occurred in the function block.
Proximity dog signal	HomeAbsSwitch	bool	OFF	Indicates the state of the proximity dog signal.
Positive limit signal	LimitSwitchPos	bool	OFF	Indicates the state of the hardware stroke limit signal in the positive direction.
Negative limit signal	LimitSwitchNeg	bool	OFF	Indicates the state of the hardware stroke limit signal in the negative direction.
Communication ready	CommunicationReady	bool	OFF	Indicates the communication ready state.
Ready for operation	ReadyForPowerOn	bool	OFF	Indicates the ready for operation state.
Operable	PowerOn	bool	OFF	Indicates the operable state.
Home position valid	IsHomed	bool	OFF	Indicates that the home position return is completed.
Axis warning	AxisWarning	bool	OFF	Indicates the axis warning state.

## Timing chart

### Normal completion



### Abnormal completion



## Error code

Value	Description
MMERR_FB_AXIS_NUMBER	A value outside the range is set to the axis No.

## Point

- This function block is executed when Enable=ON, and the axis information of the specified axis is read.
  - Read data is always updated while Valid=ON.
- <I/O mode>
- When the positioning control function block is executed, updates of read data will temporarily stop. When the positioning control is started again, updates will also be restarted.
  - When this function block is used with "sensor input method selection" of the servo parameter "Function selection D-4 (PD41)" set to "0: Input from servo amplifier", the positive limit signal (LimitSwitchPos) and the negative limit signal (LimitSwitchNeg) can be read with software version A3 or later of the servo amplifier MR-J4-GF.

## Supported version

API version	Software version
1.10	01

## Inheritance hierarchy

```

MC_FunctionBlock
  MC_ReadAxisInfo
  
```



# MC\_ReadAxisError class

Reads the error No. of the specified axis.

```
class MC_ReadAxisError : public MC_FunctionBlock
```

## Member

### Input

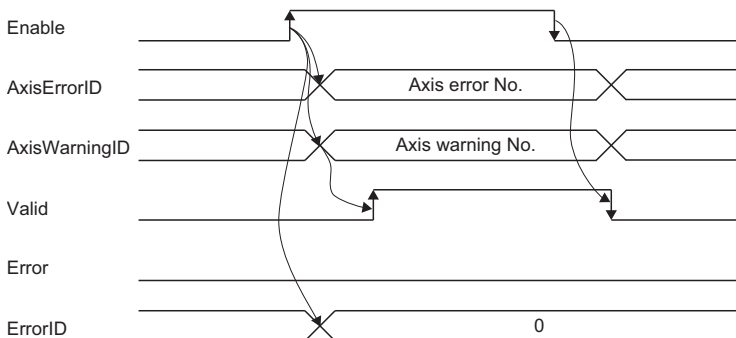
Name	Variable name	Data type	Fetch	Valid range	Description
Axis information	Axis	AXIS_REF*	—	—	Specifies the pointer to the axis information class object.
Enable	Enable	bool	Startup only	ON/OFF	Executes the function block when ON.

### Output

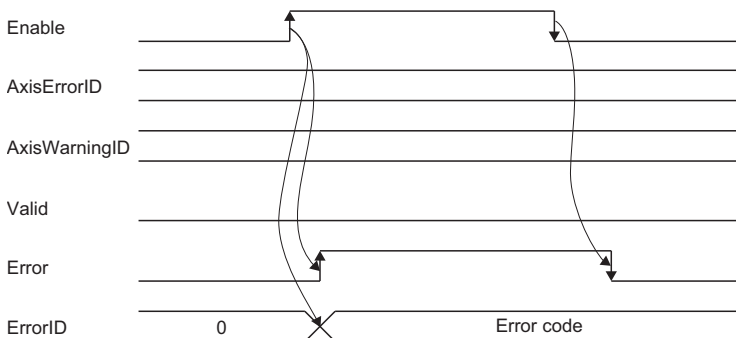
Name	Variable name	Data type	Initial value	Description
Output value valid	Valid	bool	OFF	While this device is ON, this indicates that the output value is valid.
Error	Error	bool	OFF	When ON, this indicates an error occurrence in the function block.
Error code	ErrorID	unsigned short	0	Returns the error code of the error that occurred in the function block.
Axis error No.	AxisErrorID	unsigned short	0	Returns the error code of the axis.
Axis warning No.	AxisWarningID	unsigned short	0	Returns the warning code of the axis.

## Timing chart

### Normal completion



### Abnormal completion



## Error code

Value	Description
MMERR_FB_AXIS_NUMBER	A value outside the range is set to the axis No.

## Point

- This function block reads the error No. and warning No. of the specified axis.
- The function block is executed when Enable=ON, and the error No. and warning No. of the specified axis are read.
- Read data is always updated while Valid=ON.
- When no error or warning has occurred, "0" is returned.

<I/O mode>

- When an error and a warning simultaneously occur, this function block reads only an error No.
- When the positioning control function block is executed, updates of read data will temporarily stop. When the positioning control is started again, updates will also be restarted.

## Supported version

API version	Software version
1.10	01

## Inheritance hierarchy

MC\_FunctionBlock

MC\_ReadAxisError

# MCv\_ReadServoParameter class

Reads the parameter value of the servo parameter No. of the specified axis.

```
class MCv_ReadServoParameter : public MC_FunctionBlock
```

## Member

### Input

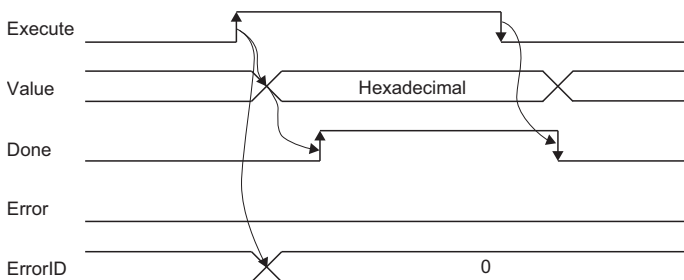
Name	Variable name	Data type	Fetch	Valid range	Description
Axis information	Axis	AXIS_REF*	—	—	Specifies the pointer to the axis information class object.
Execution command	Execute	bool	Startup only	ON/OFF	Executes the function block when ON.
Object index	ObjectIndex	unsigned short	Startup only	2001H to 25A0H	Specifies an object index No. of the servo parameter.

### Output

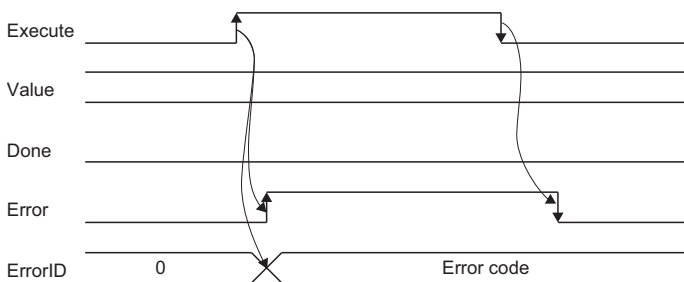
Name	Variable name	Data type	Initial value	Description
Execution completion	Done	bool	OFF	While this device is ON, this indicates that the output value is valid.
Error	Error	bool	OFF	When ON, this indicates an error occurrence in the function block.
Error code	ErrorID	unsigned short	0	Returns the error code of the error that occurred in the function block.
Read value	Value	unsigned short	0	Outputs the value read from the specified parameter.

## Timing chart

### Normal completion



### Abnormal completion



## Error code

Value	Description
MMERR_FB_AXIS_NUMBER	A value outside the range is set to the axis No.
MMERR_FB_OBJECT_INDEX	A value outside the range is set to the object index No.
MMERR_FB_READ_OR_WRITE_PARAMETER	Reading or writing the parameter failed.
MMERR_FB_INSTRUCTION_RUNNING	Another function block is executing the instruction code.
MMERR_FB_LINK_DEVICE_INACCESSIBLE	Access to the link devices is disabled.

## Point

- This function block reads the parameter value of the servo parameter No. of the specified axis.
- The function block is executed when Execute=ON, and the specified parameter value is read.
- Done turns ON when reading of the parameter is completed. The read data is held while Done=ON.
- Multiple parameters cannot be simultaneously read for one axis.
- This function block cannot be executed while MCv\_WriteServoParameter is being executed.

## Supported version

API version	Software version
1.10	01

## Inheritance hierarchy

MC\_FunctionBlock  
  MCv\_ReadServoParameter

## Reference

 Page 187 MCv\_WriteServoParameter class

# MCv\_WriteServoParameter class

Changes the parameter value of the servo parameter No. of the specified axis.

```
class MCv_WriteServoParameter : public MC_FunctionBlock
```

## Member

### Input

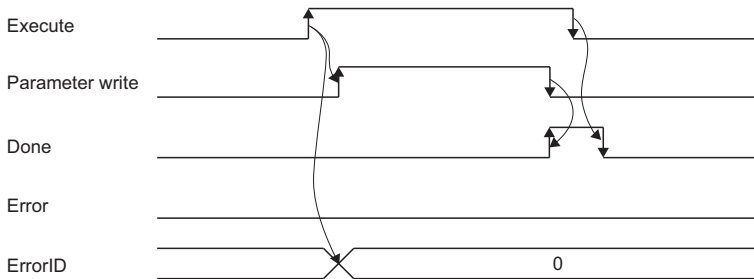
Name	Variable name	Data type	Fetch	Valid range	Description
Axis information	Axis	AXIS_REF*	—	—	Specifies the pointer to the axis information class object.
Execution command	Execute	bool	Startup only	ON/OFF	Executes the function block when ON.
Object index	ObjectIndex	unsigned short	Startup only	2001H to 25A0H	Specifies an object index No. of the servo parameter.
ROM write selection	StoreParameter	bool	Startup only	ON/OFF	Selects whether to write the parameter in a non-volatile memory.
Setting value	Value	unsigned short	Startup only	—	Specifies a setting value of the specified parameter.

### Output

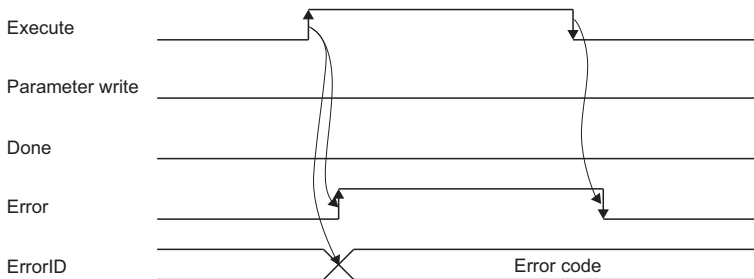
Name	Variable name	Data type	Initial value	Description
Execution completion	Done	bool	OFF	Indicates that writing to the parameter is completed.
Error	Error	bool	OFF	When ON, this indicates an error occurrence in the function block.
Error code	ErrorID	unsigned short	0	Returns the error code of the error that occurred in the function block.

## Timing chart

### Normal completion



### Abnormal completion



## Error code

Value	Description
MMERR_FB_AXIS_NUMBER	A value outside the range is set to the axis No.
MMERR_FB_OBJECT_INDEX	A value outside the range is set to the object index No.
MMERR_FB_READ_OR_WRITE_PARAMETER	Reading or writing the parameter failed.
MMERR_FB_INSTRUCTION_RUNNING	Another function block is executing the instruction code.
MMERR_FB_LINK_DEVICE_INACCESSIBLE	Access to the link devices is disabled.

## Point

- This function block changes the parameter value of the servo parameter No. of the specified axis.
- The function block is executed when Execute=ON and the specified parameter value is changed.
- Done turns ON when writing to the parameter is completed.
- Turn ON StoreParameter to store the parameters in ROM. Specify StoreParameter for each parameter. When the parameters are not stored in ROM, the data before change is restored at the next power-on.
- Multiple parameters cannot be simultaneously read for one axis.
- This function block cannot be executed while MCv\_ReadServoParameter is being executed.

## Supported version

API version	Software version
1.10	01

## Inheritance hierarchy

MC\_FunctionBlock  
    MCv\_WriteServoParameter

## Reference

 Page 185 MCv\_ReadServoParameter class


## 5.2 MMC\_Controller Class

This class controls the controller.

```
class MMC_Controller: public MMC_Master
```

### Member


#### ■Public method

Category	Name	Description	Reference
Get object method	GetFbAxisRef	Gets the object of the axis information class used in the function block class.	 Page 190 MMC_Controller::GetFbAxisRef

#### ■Public data member

Variable name	Data type	Initial value	Description
LastErrorNumber	unsigned long	MMERR_NONE	Latest error code
LastDetailErrorNumber	unsigned long	MMERR_NONE	Latest detail error code

#### ■Label

Refer to label list for labels. ( Page 203 LABEL LIST)

### Point

Generate controller class objects with object generation functions.

### Inheritance hierarchy

```
MMC_NetworkModule
  MMC_Master
    MMC_Controller
```

# MMC\_Controller::GetFbAxisRef

Gets the object of the axis information class (Motion mode) used in the function block class.

```
unsigned long GetFbAxisRef(  
    unsigned long axisNumber,  
    AXIS_REF **axisRef  
);
```

## Detailed description

### Parameter

Argument	Description
axisNumber [in]	Axis No. (1 to 16)
axisRef [out]	Pointer to the axis information class object pointer.

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_CREATE_OBJECT	Failed to generate object.

### Point

- Because the axis information class objects gotten by this method are automatically deleted at the time of controller class object delete, deleting this method is not necessary.
- This method gets the object of the axis information class (Motion mode). To get the axis information class (I/O mode), use the GetFbAxisRefIo method.

### Supported version

API version	Software version
1.00	01

### Reference

None.



# MMC\_Controller::GetFbAxisReflo

Gets the object of the axis information class (I/O mode) used in the function block class.

```
unsigned long GetFbAxisReflo(  
    unsigned long stationNumber,  
    AXIS_REF **axisRef  
);
```

## Detailed description

### Parameter

Argument	Description
stationNumber [in]	Station No. (1 to 120)
axisRef [out]	Pointer to the axis information class object pointer.

### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_CREATE_OBJECT	Failed to generate object.
MMERR_LINK_DEVICE_SETTING	Either RX/Ry or RWw/RWr setting in network configuration settings incorrect.
MMERR_CONDITION_READY_SIGNAL_OFF	The READY signal [X0] is OFF.

### Point

- Call this method after setting the parameter, calling the SetUserProgramReady method, and turning ON the user program READY signal [Y0].
- Because the axis information class objects gotten by this method are automatically deleted at the time of controller class object delete, deleting this method is not necessary.
- This method gets the object of the axis information class (I/O mode). To get the axis information class (Motion mode), use the GetFbAxisRef method.

### Supported version

API version	Software version
1.10	01

### Reference

None.

## 5.3 AXIS\_REF Class

This is the axis information class used in the function block class.

```
class AXIS_REF : public MMC_Slave
```

### Member

#### ■Public method

There is no public method.

#### ■Public data member

Variable name	Data type	Initial value	Description
FbAxisStatus	short	MMC_FBST_DISABLED	Outputs the axis state based on the PLCopen state diagram.

#### ■Label

Refer to label list for labels. ([↗](#) Page 203 LABEL LIST)

### Point

- Get axis information class objects using the GetFbAxisRef method or GetFbAxisReflo method of the controller class.
- Refer to axis information class state transition for axis state (FbAxisStatus). ([↗](#) Page 147 Axis information class state diagram)

### Inheritance hierarchy

MMC\_NetworkModule

MMC\_Slave

AXIS\_REF

## 5.4 AXIS\_REF\_MOTION Class

This is the axis information class used in the function block class.

```
class AXIS_REF_MOTION : public AXIS_REF
```

### Member


#### ■Public method

There is no public method.

#### ■Public data member

There is no public data member.

#### ■Label

Refer to label list for labels. ( Page 203 LABEL LIST)

### Point

- Get axis information class objects (Motion mode) using the GetFbAxisRef method of the controller class.

### Inheritance hierarchy

MMC\_NetworkModule

  MMC\_Slave

    AXIS\_REF

      AXIS\_REF\_MOTION

### Reference

 Page 190 MMC\_Controller::GetFbAxisRef

# 5.5 AXIS\_REF\_IO Class

This is the axis information class used in the function block class.

```
class AXIS_REF_IO : public AXIS_REF
```

## Member

### Public method

There is no public method.

### Public data member

Variable name	Data type	Initial value	Description
LinkDeviceUse	bool	OFF	Temporarily releases link devices used in the library. Use this label only when accessing the link devices with user programs. Access the link devices after LinkDeviceAccessible turns ON. To restart the library operation, turn OFF LinkDeviceUse.
LinkDeviceAccessible	bool	OFF	Outputs the state of the access to the link devices in the library. ON: Accessible OFF: Inaccessible

### Label

Variable name	Data type	Description
bRX[]	bool	RX area
bRY[]	bool	RY area
wRX[]	unsigned short	RX area (for word access)
wRY[]	unsigned short	RY area (for word access)
dwRX[]	unsigned long	RX area (for double word access)
dwRY[]	unsigned long	RY area (for double word access)
wRWw[]	unsigned short	RWw area
wRWr[]	unsigned short	RWr area
dwRWw[]	unsigned long	RWw area (for double word access)
dwRWr[]	unsigned long	RWr area (for double word access)
wRWw[].b[]	bool	RWw area (for bit access)
wRWr[].b[]	bool	RWr area (for bit access)

## Point

- Get axis information class objects (I/O mode) using the GetFbAxisReflo method of the controller class.
- When calling the GetFbAxisReflo method of the controller class, labels for accessing the link device are automatically generated according to the parameters set in the Simple Motion board.
- Labels only retain an element the size of the area for each station. Do not access the elements outside the range.
- Access the RX areas and RY areas in 16-bit units by using the labels for word access (wRX, wRY).
- Access the RX areas, RY areas, RWw areas, and RWr areas in 32-bit units by using the labels for double word access (dwRX, dwRY, dwRWw, dwRWr).
- Access the RX areas, RY areas, RWw areas, and RWr areas in one-bit units by using the labels for bit access (wRWw[].b, wRWr[].b).

## Inheritance hierarchy

```
MMC_NetworkModule
  MMC_Slave
    AXIS_REF
      AXIS_REF_IO
```

## Reference

Page 191 MMC\_Controller::GetFbAxisReflo

# 5.6 Object Generation Functions

## MmfCreateFunctionBlock

Generates function block class objects.

```
unsigned long MmfCreateFunctionBlock(  
    unsigned long fbID,  
    MC_FunctionBlock **functionBlock  
);
```

### Detailed description

#### Parameter

Argument	Description
fbID [in]	Function block ID*1
functionBlock [out]	Pointer to the function block class object pointer.

\*1 Specify the following values for function block ID.

Value	Description
MMC_FBID_MC_Power	MC_Power
MMC_FBID_MCv_Home	MCv_Home
MMC_FBID_MC_Stop	MC_Stop
MMC_FBID_MC_MoveAbsolute	MC_MoveAbsolute
MMC_FBID_MC_MoveRelative	MC_MoveRelative
MMC_FBID_MC_Reset	MC_Reset
MMC_FBID_MC_MoveAdditive	MC_MoveAdditive
MMC_FBID_MC_MoveVelocity	MC_MoveVelocity
MMC_FBID_MC_TorqueControl	MC_TorqueControl
MMC_FBID_MC_SetPosition	MC_SetPosition
MMC_FBID_MC_SetOverride	MC_SetOverride
MMC_FBID_MC_ReadActualPosition	MC_ReadActualPosition
MMC_FBID_MC_ReadStatus	MC_ReadStatus
MMC_FBID_MC_ReadAxisInfo	MC_ReadAxisInfo
MMC_FBID_MC_ReadAxisError	MC_ReadAxisError
MMC_FBID_MCv_ReadServoParameter	MCv_ReadServoParameter
MMC_FBID_MCv_WriteServoParameter	MCv_WriteServoParameter

#### Return value

Value	Description
MMC_OK	Function succeeded
MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
MMERR_CREATE_OBJECT	Failed to generate object.

#### Point

- Specify the ID of the generated function block to the function block ID.
- For objects generated using this function, delete them with the Delete method.

#### Supported version

API version	Software version
1.00	01

#### Reference

None.

# 6 STRUCTURE LIST

Refer to the following for details of the data for each structure.

📖 Simple Motion Board User's Manual (Application)

📖 Simple Motion Board User's Manual (Advanced Synchronous Control)

## 6.1 MMST\_PositioningData Structure

The following are the positioning data structures used by positioning data settings.

Variable name	Data type	Name
OperationPattern	unsigned short	[Da.1] Operation pattern
ControlMethod	unsigned short	[Da.2] Control method
AccelerationTimeNo	unsigned short	[Da.3] Acceleration time No.
DecelerationTimeNo	unsigned short	[Da.4] Deceleration time No.
PositioningAddress	long	[Da.6] Positioning address/movement amount
ArcAddress	long	[Da.7] Arc address
CommandSpeed	unsigned long	[Da.8] Command speed
DwellTime	unsigned short	[Da.9] Dwell time/JUMP destination positioning data No.
Mcode	unsigned short	[Da.10] M code/Condition data No./Number of LOOP to LEND repetitions
InterpolatedAxisNo1	unsigned long	[Da.20] Axis to be interpolated No.1
InterpolatedAxisNo2	unsigned long	[Da.21] Axis to be interpolated No.2
InterpolatedAxisNo3	unsigned long	[Da.22] Axis to be interpolated No.3
McodeOnTiming	unsigned short	[Da.27] M code ON signal output timing
AbsDirectionInDegrees	unsigned short	[Da.28] ABS direction in degrees
InterpolationSpeed	unsigned short	[Da.29] Interpolation speed designation method

## 6.2 MMST\_BlockStartData Structure

The following are the block start data structures used by block start data settings.

Variable name	Data type	Name
Shape	unsigned short	[Da.11] Shape
StartDataNo	unsigned short	[Da.12] Start data No.
SpecialStartInstruction	unsigned short	[Da.13] Special start instruction
Parameter	unsigned short	[Da.14] Parameter

## 6.3 MMST\_BlockConditionData Structure

The following are the condition data structures used by condition data settings.

Variable name	Data type	Name
ConditionTarget	unsigned short	[Da.15] Condition target
ConditionOperator	unsigned short	[Da.16] Condition operator
Address	unsigned long	[Da.17] Address
Parameter1	unsigned long	[Da.18] Parameter 1
Parameter2	unsigned long	[Da.19] Parameter 2
NumberOfStartingAxes	unsigned long	[Da.23] Number of simultaneous starting axes
StartingAxesNo1	unsigned long	[Da.24] Simultaneous starting axis No.1
StartingAxesNo2	unsigned long	[Da.25] Simultaneous starting axis No.2
StartingAxesNo3	unsigned long	[Da.26] Simultaneous starting axis No.3

## 6.4 MMST\_InterruptParameter Structure

The following are the interrupt parameter structures used by interrupt parameter settings.

Variable name	Data type	Name
IntCondition	unsigned long	Interrupt condition
IntSensitivity	unsigned long	Interrupt detection timing
IntOutputDataSeting	unsigned long	Factor details output setting
IntTimeSetting	unsigned long	Condition for condition completion continue
IntJudgeValue1	unsigned long	Interrupt condition judge value 1
IntJudgeValue2	unsigned long	Interrupt condition judge value 2
IntTime	unsigned short	Time for condition completion [ms]

## 6.5 MMST\_InterruptData Structure

The following are the interrupt data structures used by interrupt callback functions.

Variable name	Data type	Name
BoardID	unsigned long	Board ID (0 to 3)
FreerunningTime	unsigned short	Free-running timer
IntFactor[2]	unsigned long	[Md.1101] Interrupt factor (1 to 64)
IntFactorDetail[64]	unsigned long	[Md.1102] Interrupt factor details

## 6.6 MMST\_CamPositionData Structure

The following are the cam position calculation data structures used by cam position calculation functions.

Variable name	Data type	Name
CamNo	unsigned short	[Cd.613] Cam position calculation: Cam No.
Stroke	long	[Cd.614] Cam position calculation: Stroke amount
LengthPerCycle	unsigned long	[Cd.615] Cam position calculation: Cam axis length per cycle
ReferencePosition	long	[Cd.616] Cam position calculation: Cam reference position
CommandPositionPerCycle	unsigned long	[Cd.617] Cam position calculation: Cam axis current value per cycle
CommandPosition	long	[Cd.618] Cam position calculation: Cam axis feed current value

## 6.7 MMST\_RotaryCutterCamData Structure

The following are the auto-generation data structures used by cam auto-generation (rotary cutter cam(central reference)).

Variable name	Data type	Name
Resolution	unsigned long	[Cd.611] Cam auto-generation data
AutoGenerationOption	unsigned short	Resolution
SyncSectionAccelerationRatio	short	Auto-generation option
SheetLength	unsigned long	Synchronous section acceleration ratio
SheetSyncWidth	unsigned long	Sheet length
SyncAxisLength	unsigned long	Sheet synchronization width
SyncPositionAdjustment	long	Synchronous axis length
AccelerationDecelerationWidth	unsigned long	Synchronous position adjustment
NumberOfCutter	unsigned short	Acceleration/deceleration width
		Number of cutter

## 6.8 MMST\_EasyStrokeRatioCamData Structure

The following are the auto-generation data structures used by cam auto-generation (easy stroke ratio cam).

Variable name	Data type	Name	
Resolution	unsigned long	[Cd.611] Cam auto-generation data	Resolution
CamAxisLengthPerCycle	unsigned long		Cam axis length per cycle
CamDataStartingPoint	unsigned long		Cam data starting point
NumberOfSections	unsigned short		Number of sections

## 6.9 MMST\_EasyStrokeRatioCamSectionData Structure

The following are the section data structures used by cam auto-generation (easy stroke ratio cam).

Variable name	Data type	Name		
CamCurveType	unsigned short	[Cd.611] Cam auto-generation data	Section	Cam curve type
EndPoint	unsigned long			End point
Stroke	long			Stroke

## 6.10 MMST\_AdvancedStrokeRatioCamData Structure

The following are the auto-generation data structures used by cam auto-generation (advanced stroke ratio cam).

Variable name	Data type	Name	
Resolution	unsigned long	[Cd.611] Cam auto-generation data	Resolution
CamAxisLengthPerCycle	unsigned long		Cam axis length per cycle
CamStrokeAmount	unsigned long		Cam stroke amount
UnitSetting	unsigned short		Unit setting
CamDataStartingPoint	unsigned long		Cam data starting point
NumberOfSections	unsigned short		Number of sections

## 6.11 MMST\_AdvancedStrokeRatioCamSectionData Structure

The following are the section data structures used by cam auto-generation (advanced stroke ratio cam).

Variable name	Data type	Name		
CamCurveType	unsigned short	[Cd.611] Cam auto-generation data	Section	Cam curve type
EndPoint	unsigned long			End point
Stroke	long			Stroke
CurveApplicableRangeP1	unsigned short			Curve applicable range (P1)
CurveApplicableRangeP2	unsigned short			Curve applicable range (P2)
AccDecRangeCompensationL1	unsigned short			Acceleration/deceleration range compensation (Range L1)
AccDecRangeCompensationL2	unsigned short			Acceleration/deceleration range compensation (Range L2)



# 7 ERROR CODE LIST

## 7.1 Common Errors

Value	Constant definition	Occurrence cause/Remedy
FFFFFFFFH	MMERR_NONE	An error has not occurred once.
01000001H	MMERR_ARGUMENT_0□ □=1 to 9: Argument location	The argument is outside the set range.
01000002H		
01000003H		
01000004H		
01000005H		
01000006H		
01000007H		
01000008H		
01000009H		
01000011H	MMERR_TIMEOUT_0□ □=1 to 9: Timeout location	The timeout time has elapsed.
01000012H		
01000013H		
01000014H		
01000015H		
01000016H		
01000017H		
01000018H		
01000019H		
01000020H	MMERR_CREATE_OBJECT	Failed to generate object.
01000021H	MMERR_ALREADY_CREATED	The object of the same board ID has already been generated.
01000040H	MMERR_AXIS_ERROR	An axis error occurred. Check "[Md.23] Axis error No.".
01000041H	MMERR_LINK_DEVICE_SETTING	Either RX/RX or RWw/RWw setting in network configuration settings is incorrect.
01000100H	MMERR_CONDITION_READY_SIGNAL_OFF	The READY signal [X0] is OFF.
01000101H	MMERR_CONDITION_BUSY_ON	BUSY signal [X10 to X1F] are ON.
01000102H	MMERR_CONDITION_BUSY_OFF	BUSY signal [X10 to X1F] are OFF.
01000103H	MMERR_CONDITION_USER_PROGRAM_READY_ON	User program READY signal [Y0] is ON.
01000104H	MMERR_CONDITION_USER_PROGRAM_READY_OFF	User program READY signal [Y0] is OFF.
01000105H	MMERR_CONDITION_POSITIONING_START_ON	Positioning start signal [Y10 to Y1F] are ON.
01000106H	MMERR_CONDITION_AXIS_OPERATION_STATUS	"[Md.26] Axis operation status" is incorrect.
01000107H	MMERR_CONDITION_AXIS_ERROR_DETECTION	Error detection ("[Md.31] Status": b13) is ON.
01000108H	MMERR_CONDITION_START_COMPLETE_ON	Start complete ("[Md.31] Status": b14) is ON.
01000109H	MMERR_CONDITION_POSITIONING_COMPLETE_ON	Positioning complete ("[Md.31] Status": b15) is ON.
0100010AH	MMERR_CONDITION_RESTART_COMMAND_ON	"[Cd.6] Restart command" is ON.
0100010BH	MMERR_CONDITION_MPG_FLAG_ON	The "[Cd.21] Manual pulse generator enable flag" is ON.
0100010CH	MMERR_CONDITION_JOG_START_ON	"[Cd.181] Forward run JOG start"/"[Cd.182] Reverse run JOG start" is ON.
0100010DH	MMERR_CONDITION_SYNC_START_ON	"[Cd.380] Synchronous control start" is ON.
0100010EH	MMERR_CONDITION_SYNC_ENCODER_INVALID	The settings valid flag ("[Md.325] Synchronous encoder axis status": b0) is OFF.
01010000H	MMERR_WIN_CREATE_SEMAPHORE	An error occurred in the CreateSemaphore function (Win32API). Call the GetLastError function of Win32API and confirm the error details.
01010001H	MMERR_WIN_CLOSE_HANDLE	An error occurred in the CloseHandle function (Win32API). Call the GetLastError function of Win32API and confirm the error details.
01010002H	MMERR_WIN_WAIT_FOR_SINGLE_OBJECT	An error occurred in the WaitForSingleObject function (Win32API). Call the GetLastError function of Win32API and confirm the error details.
01010003H	MMERR_WIN_RELEASE_SEMAPHORE	An error occurred in the ReleaseSemaphore function (Win32API). Call the GetLastError function of Win32API and confirm the error details.

Value	Constant definition	Occurrence cause/Remedy
01010010H	MMERR_WIN_CREATE_EVENT	An error occurred in the CreateEvent function (Win32API). Call the GetLastError function of Win32API and confirm the error details.
01010011H	MMERR_WIN_RESET_EVENT	An error occurred in the ResetEvent function (Win32API). Call the GetLastError function of Win32API and confirm the error details.
01010012H	MMERR_WIN_SET_EVENT	An error occurred in the SetEvent function (Win32API). Call the GetLastError function of Win32API and confirm the error details.
01010020H	MMERR_WIN_CREATE_THREAD	An error occurred in the CreateThread function (Win32API). Call the GetLastError function of Win32API and confirm the error details.
01010021H	MMERR_WIN_SET_THREAD_PRIORITY	An error occurred in the SetThreadPriority function (Win32API). Call the GetLastError function of Win32API and confirm the error details.
01010022H	MMERR_WIN_RESUME_THREAD	An error occurred in the ResumeThread function (Win32API). Call the GetLastError function of Win32API and confirm the error details.
01010023H	MMERR_WIN_GET_EXIT_CODE_THREAD	An error occurred in the GetExitCodeThread function (Win32API). Call the GetLastError function of Win32API and confirm the error details.

## 7.2 Device Related Errors

Value	Constant definition	Occurrence cause/Remedy
02000000H	MMERR_DEV_REOPEN	The Open method is already called.
02000001H	MMERR_DEV_UNOPEN	The Open method of the device driver class has not been called.
02000010H	MMERR_DEV_NOT_FOUND_BOARD	The Simple Motion board which has the designated board ID could not be found. Confirm the board ID selection (dip switch) of the Simple Motion board.
02000011H	MMERR_DEV_UNSUPPORT_DEVICE_DRIVER	The device driver is not a supported version. Use a API library that combines with the device driver contained in the "EM Software Development Kit".
02000012H	MMERR_DEV_DEVICE_DRIVER	An error occurred with a call of the device driver. Confirm that the device driver is installed.
02000013H	MMERR_DEV_INACCESSIBLE	There is a Simple Motion board which cannot access the buffer memory.
02000014H	MMERR_DEV_ALREADY_OTHER_PROCESS_OPEN	The device of the same board ID is already open by a different process.
02000020H	MMERR_DEV_ADDRESS_RANGE_OVER	The "offset" + "size" designated by the argument exceeds the size of the buffer memory.
02010000H	MMERR_DEV_WIN_LOAD_LIBRARY	An error occurred in the LoadLibrary function (Win32API). Call the GetLastError function of Win32API and confirm the error details.
02010001H	MMERR_DEV_WIN_GET_PROC_ADDRESS	An error occurred in the GetProcAddress function (Win32API). Call the GetLastError function of Win32API and confirm the error details.

## 7.3 Interrupt Related Errors

Value	Constant definition	Occurrence cause/Remedy
03000000H	MMERR_INT_ALREADY_START_DRIVER	The interrupt driver is already started.
03000001H	MMERR_INT_ALREADY_END_DRIVER	The interrupt driver is already stopped.
03000010H	MMERR_INT_NOT_START_DRIVER	The interrupt driver is stopped. Call the StartInterrupt method.
03000011H	MMERR_INT_TERMINATE_DRIVER	The EndInterrupt method was called while the interrupt for the designated event factor was being confirmed.
03000012H	MMERR_INT_TERMINATE_NOTIFY_EVENT	An error occurred in the interrupt event notification thread while the interrupt for the designated event factor was being confirmed.
03000013H	MMERR_INT_SET_HOST_APPLICATION_EVENT	While the interrupt for the designated event factor was being waited, a method which releases the standby status was called from the user program.
03000020H	MMERR_INT_ALREADY_REREGISTER_CALLBACK	The interrupt callback function has already been registered. To change the interrupt callback function, call the UnregisterIntCallback method.
03000021H	MMERR_INT_ALREADY_UNREREGISTER_CALLBACK	The interrupt callback function has already been unregistered.

## 7.4 DMA Transmission Related Errors

Value	Constant definition	Occurrence cause/Remedy
03010000H	MMERR_DMA_ALREADY_START_DRIVER	The DMA transmission driver is already started.
03010001H	MMERR_DMA_ALREADY_END_DRIVER	The DMA transmission driver is already stopped.
03010010H	MMERR_DMA_NOT_START_DRIVER	The DMA transmission driver is stopped. Call the StartDMA method.
03010011H	MMERR_DMA_INVALID_ADDRESS	The DMA destination address is incorrect.

## 7.5 SLMP Communication Function Related Errors

Value	Constant definition	Occurrence cause/Remedy
03030001H	MMERR_SLMP_SEND_RECEIVE_ERROR	An error occurred during SLMP communication. Since an end code of the SLMP response message is stored to the details error code, refer to the following to check the error contents and take actions. <a href="#">MELSEC iQ-R CC-Link IE Field Network User's Manual (Application)</a>
03030002H	MMERR_SLMP_COMMUNICATION_RUNNING	A SLMP communication process is being executed. Execute again after other SLMP communication processes are completed.

## 7.6 Function Block Related Errors

Value	Constant definition	Occurrence cause/Remedy
1100H	MMERR_FB_AXIS_NUMBER	An axis information error occurred. Review the axis information value. <ul style="list-style-type: none"> <li>A value outside the range is set to the axis No.</li> <li>The axis No. which is different from the previous input is set.</li> <li>NULL is set to the axis information.</li> <li>The station-specific mode is not supported.</li> </ul>
1102H	MMERR_FB_POSITIONING_DATA_NUMBER	A value outside the range is set to the positioning data No. Correct the value of the positioning data No. within 1 to 100.
1103H	MMERR_FB_POSITION	A value outside the range is set to the target position, and movement amount. Review the value of the target position and the movement amount.
1104H	MMERR_FB_VELOCITY	A value outside the range is set to the speed. Review the speed value.
1105H	MMERR_FB_ACCDEC	A value outside the range is set to the acceleration/deceleration time. Review the value of the acceleration time or the deceleration time.
1106H	MMERR_FB_UNIT	A value outside the range is set to the unit. Correct the unit setting to mm, inch, or pulse.
1107H	MMERR_FB_DIRECTION	A value outside the range is set to the rotation direction. Review the rotation direction value.
1108H	MMERR_FB_TORQUE	A value outside the range is set to the target torque. Review the target torque value.
1109H	MMERR_FB_VELOCITY_FACTOR	A value outside the range is set to the speed override coefficient. Correct the speed override coefficient value within 0.00 to 3.00.
110BH	MMERR_FB_OBJECT_INDEX	A value outside the range is set to the object index. Review the value of the object index No.
110DH	MMERR_FB_SERVO_PARAMETER_INACCESSIBLE	The servo parameter setting cannot be referred to. Correct the value of the servo parameter PA19 to "00ABH" or "10ABH".
1200H	MMERR_FB_READY_SIGNAL_OFF	The READY signal [X0] is OFF. Eliminate the error of the controller or servo amplifier and execute the function block again.
1201H	MMERR_FB_SERVO_POWER_OFF	The control circuit power supply of the servo amplifier is OFF, or the servo amplifier is not connected. Check if the power of the servo amplifier is ON or a communication cable is connected to the servo amplifier.
1202H	MMERR_FB_AXIS_ERROR	An error occurred. Eliminate the error and execute the function block again.

Value	Constant definition	Occurrence cause/Remedy
1203H	MMERR_FB_START_NOT_POSSIBLE	Positioning can not be started. Execute the function block again after the control in execution is completed or turning OFF the start signal.
1204H	MMERR_FB_AXIS_STOPPED	The axis is in a Stopping state, or the stop command for the axis is ON. Execute the function block again after changing the axis status to Standstill or turning OFF the stop command.
1205H	MMERR_FB_READ_OR_WRITE_PARAMETER	Reading or writing the parameter failed. Check if the parameter No. or object index is correct or the setting data is within the range.
1206H	MMERR_FB_INSTRUCTION_RUNNING	Another function block is executing the instruction code. Check the completion of the function block executing the instruction code, and then execute the target function block.
1207H	MMERR_FB_LINK_DEVICE_INACCESSIBLE	Access to the link devices is disabled. Turn OFF the link device use flag (LinkDeviceUse) of the AXIS_REF class, and execute the target function block.
1300H	MMERR_FB_WARNING	A warning occurred.










# 8 LABEL LIST

The following is a list of labels.

Category	Label name	Data type	Name
MMC_Controller	AxPrm	MMC_EM340AxPrm[16] (  Page 208 MMC_EM340AxPrm)	Parameter
	AxMntr	MMC_EM340AxMntr[16] (  Page 209 MMC_EM340AxMntr)	Axis monitor data
	SysMntr1	MMC_EM340SysMntr1 (  Page 211 MMC_EM340SysMntr1)	System monitor data
	AxCtrl1	MMC_EM340AxCtrl1[16] (  Page 211 MMC_EM340AxCtrl1)	Axis control data
	SysCtrl	MMC_EM340SysCtrl (  Page 213 MMC_EM340SysCtrl)	System control data
	PositData1	MMC_EM340PositData1[16] (  Page 214 MMC_EM340PositData1)	Positioning data
	BlockStartData	MMC_EM340BlockStartData[16] (  Page 215 MMC_EM340BlockStartData)	Block start data
	AxCtrl2	MMC_EM340AxCtrl2[16] (  Page 215 MMC_EM340AxCtrl2)	Axis control data2
	SysMntr2	MMC_EM340SysMntr2 (  Page 215 MMC_EM340SysMntr2)	System monitor data2
	SvInpAxPrm	MMC_EM340SvInpAxPrm[16] (  Page 215 MMC_EM340SvInpAxPrm)	Servo input axis parameters
	SvInpAxMntr	MMC_EM340SvInpAxMntr[16] (  Page 216 MMC_EM340SvInpAxMntr)	Servo input axis monitor data
	SyncEncAxPrm	MMC_EM340SyncEncAxPrm[16] (  Page 216 MMC_EM340SyncEncAxPrm)	Synchronous encoder axis parameters
	SyncEncAxCtrl	MMC_EM340SyncEncAxCtrl[16] (  Page 216 MMC_EM340SyncEncAxCtrl)	Synchronous encoder axis control data
	SyncEncAxMntr	MMC_EM340SyncEncAxMntr[16] (  Page 216 MMC_EM340SyncEncAxMntr)	Synchronous encoder axis monitor data
	SyncSysCtrl	MMC_EM340SyncSysCtrl (  Page 217 MMC_EM340SyncSysCtrl)	Synchronous control
	SyncCtrlPrm	MMC_EM340SyncAxPrm[16] (  Page 217 MMC_EM340SyncAxPrm)	Synchronous control parameters
	SyncCtrlMntr	MMC_EM340SyncAxMntr[16] (  Page 218 MMC_EM340SyncAxMntr)	Synchronous control monitor data
	SyncCtrlAxCtrl	MMC_EM340SyncAxCtrl[16] (  Page 219 MMC_EM340SyncAxCtrl)	Synchronous control axis control data
	CamCalculation	MMC_EM340CamCalculation (  Page 219 MMC_EM340CamCalculation)	Cam position calculation
	MarkSignalPrm	MMC_EM340MarkSignalPrm[16] (  Page 220 MMC_EM340MarkSignalPrm)	Mark detection parameters
	MarkSignalCtrl	MMC_EM340MarkSignalCtrl[16] (  Page 220 MMC_EM340MarkSignalCtrl)	Mark detection control data
	MarkSignalMntr	MMC_EM340MarkSignalMntr[16] (  Page 220 MMC_EM340MarkSignalMntr)	Mark detection monitor data
	CmnPrm	MMC_EM340CmnPrm (  Page 220 MMC_EM340CmnPrm)	Common parameter
	SvNetPrm	MMC_EM340SvNetPrm[20] (  Page 220 MMC_EM340SvNetPrm)	Servo network configuration parameters
	SvNetMntr	MMC_EM340SvNetMntr[20] (  Page 220 MMC_EM340SvNetMntr)	Servo network configuration monitor data
	AxMntr2	MMC_EM340AxMntr2[16] (  Page 220 MMC_EM340AxMntr2)	Axis monitor data 2
	PositData2	MMC_EM340PositData2[16] (  Page 220 MMC_EM340PositData2)	Positioning data

Category	Label name	Data type	Name
MMC_Controller	StartHistory	MMC_EM340StartHistory ( <a href="#">Page 221</a> MMC_EM340StartHistory)	Start history
	Interrupt	MMC_EM340Interrupt ( <a href="#">Page 221</a> MMC_EM340Interrupt)	Interrupt
	OptionalBitMntr	MMC_EM340OptionalBitMntr ( <a href="#">Page 222</a> MMC_EM340OptionalBitMntr)	Optional bit monitor
	AllAxMntr	MMC_EM340AllAxMntr ( <a href="#">Page 222</a> MMC_EM340AllAxMntr)	All axes monitor
	DirectCtrl	MMC_EM340DirectCtrl ( <a href="#">Page 222</a> MMC_EM340DirectCtrl)	Direct control
	CamOperation	MMC_EM340CamOperation ( <a href="#">Page 223</a> MMC_EM340CamOperation)	Cam data operation
	UserWatchDog	MMC_EM340UserWatchDog ( <a href="#">Page 224</a> MMC_EM340UserWatchDog)	User watchdog
	ControllerInPosition	MMC_EM340ControllerInPosition ( <a href="#">Page 224</a> MMC_EM340ControllerInPosition)	Controller in-position
	BoardMntr	MMC_EM340BoardMntr ( <a href="#">Page 224</a> MMC_EM340BoardMntr)	Board system monitor
	RemoteOperation	MMC_EM340RemoteOperation ( <a href="#">Page 224</a> MMC_EM340RemoteOperation)	Remote operation
	BitDevice	MMC_EM340BitDevice ( <a href="#">Page 224</a> MMC_EM340BitDevice)	I/O signal
	DmaTransfer	MMC_EM340DmaTransfer ( <a href="#">Page 224</a> MMC_EM340DmaTransfer)	DMA transmission
	TimeSetting	MMC_EM340TimeSetting ( <a href="#">Page 225</a> MMC_EM340TimeSetting)	Time setting
	MMC_EM340GF	SyncEncAxPrmViaLinkDev	MMC_EM340SyncEncAxPrmViaLinkDev[16] ( <a href="#">Page 225</a> MMC_EM340SyncEncAxPrmViaLinkDev)
SysMntr3		MMC_EM340SysMntr3 ( <a href="#">Page 225</a> MMC_EM340SysMntr3)	System monitor data3
IEFieldInfo		MMC_EM340IEFieldInfo ( <a href="#">Page 226</a> MMC_EM340IEFieldInfo)	For network
LinkDevCHGBit		MMC_EM340LinkDevCHGBit[16] ( <a href="#">Page 228</a> MMC_EM340LinkDevCHGBit)	Link device external signal assignment parameters (bit device)
LinkDevCHGWord		MMC_EM340LinkDevCHGWord[16] ( <a href="#">Page 232</a> MMC_EM340LinkDevCHGWord)	Link device external signal assignment parameters (word device)
ServoObject		MMC_EM340ServoObject[16] ( <a href="#">Page 232</a> MMC_EM340ServoObject)	Servo object specification area
SlaveMonitor		MMC_EM340SlaveMonitor[16] ( <a href="#">Page 232</a> MMC_EM340SlaveMonitor)	Monitor data for slave device operation
SlaveControl		MMC_EM340SlaveControl[16] ( <a href="#">Page 233</a> MMC_EM340SlaveControl)	Control data for slave device operation
ExCamData		MMC_EM340ExCamData ( <a href="#">Page 233</a> MMC_EM340ExCamData)	Extended cam data
ExAutomaticCam		MMC_EM340ExAutomaticCam ( <a href="#">Page 235</a> MMC_EM340ExAutomaticCam)	Extended automatic cam
Ethernet		MMC_EM340Ethernet ( <a href="#">Page 237</a> MMC_EM340Ethernet)	Ethernet communication

Category	Label name	Data type	Name
MMC_Axis	Ready	bool	R: READY
	SynchronizationFlag	bool	R: Synchronization flag
	Busy	bool	R: BUSY
	PositioningStart	bool	RW: Positioning start
	ControllerInPositionRange	unsigned short	RW: [Pr.1190] Controller in-position range
	ControllerInPositionFlag	unsigned short	R: [Md.1190] Controller in-position flag
	SynchronousControlStart	bool	RW: [Cd.380] Synchronous control start
	SynchronousControlAnalysisMode	bool	RW: [Cd.381] Synchronous control analysis mode
	FLS_OperationDevice	bool	Upper limit signal (FLS) operation device
	RLS_OperationDevice	bool	Lower limit signal (RLS) operation device
	DOG_OperationDevice	bool	Proximity dog signal (DOG) operation device
	STOP_OperationDevice	bool	Stop signal (STOP) operation device
	AxPrm	MMC_EM340AxPrm ( <a href="#">Page 208</a> MMC_EM340AxPrm)	Parameter
	AxMntr	MMC_EM340AxMntr ( <a href="#">Page 209</a> MMC_EM340AxMntr)	Axis monitor data
	AxCtrl1	MMC_EM340AxCtrl1 ( <a href="#">Page 211</a> MMC_EM340AxCtrl1)	Axis control data
	PositData1	MMC_EM340PositData1 ( <a href="#">Page 214</a> MMC_EM340PositData1)	Positioning data
	BlockStartData	MMC_EM340BlockStartData ( <a href="#">Page 215</a> MMC_EM340BlockStartData)	Block start data
	AxCtrl2	MMC_EM340AxCtrl2 ( <a href="#">Page 215</a> MMC_EM340AxCtrl2)	Axis control data2
	SvInpAxPrm	MMC_EM340SvInpAxPrm ( <a href="#">Page 215</a> MMC_EM340SvInpAxPrm)	Servo input axis parameters
	SvInpAxMntr	MMC_EM340SvInpAxMntr ( <a href="#">Page 216</a> MMC_EM340SvInpAxMntr)	Servo input axis monitor data
	SyncCtrlPrm	MMC_EM340SyncAxPrm ( <a href="#">Page 217</a> MMC_EM340SyncAxPrm)	Synchronous control parameters
	SyncCtrlMntr	MMC_EM340SyncAxMntr ( <a href="#">Page 218</a> MMC_EM340SyncAxMntr)	Synchronous control monitor data
	SyncCtrlAxCtrl	MMC_EM340SyncAxCtrl ( <a href="#">Page 219</a> MMC_EM340SyncAxCtrl)	Synchronous control axis control data
	MarkSignalPrm	MMC_EM340MarkSignalPrm ( <a href="#">Page 220</a> MMC_EM340MarkSignalPrm)	Mark detection parameters
	MarkSignalCtrl	MMC_EM340MarkSignalCtrl ( <a href="#">Page 220</a> MMC_EM340MarkSignalCtrl)	Mark detection control data
	MarkSignalMntr	MMC_EM340MarkSignalMntr ( <a href="#">Page 220</a> MMC_EM340MarkSignalMntr)	Mark detection monitor data
	AxMntr2	MMC_EM340AxMntr2 ( <a href="#">Page 220</a> MMC_EM340AxMntr2)	Axis monitor data 2
	PositData2	MMC_EM340PositData2 ( <a href="#">Page 220</a> MMC_EM340PositData2)	Positioning data
	Interrupt	MMC_EM340Interrupt ( <a href="#">Page 221</a> MMC_EM340Interrupt)	Interrupt

Category	Label name	Data type	Name
MMC_J4GF	LinkDevCHGBit	MMC_EM340LinkDevCHGBit (  Page 228 MMC_EM340LinkDevCHGBit)	Link device external signal assignment parameters (bit device)
	LinkDevCHGWord	MMC_EM340LinkDevCHGWord (  Page 232 MMC_EM340LinkDevCHGWord)	Link device external signal assignment parameters (word device)
	ServoObject	MMC_EM340ServoObject (  Page 232 MMC_EM340ServoObject)	Servo object specification area
	SlaveMonitor	MMC_EM340SlaveMonitor (  Page 232 MMC_EM340SlaveMonitor)	Monitor data for slave device operation
	SlaveControl	MMC_EM340SlaveControl (  Page 233 MMC_EM340SlaveControl)	Control data for slave device operation
MMC_SyncEncoder	SyncEncAxPrm	MMC_EM340SyncEncAxPrm (  Page 216 MMC_EM340SyncEncAxPrm)	Synchronous encoder axis parameters
	SyncEncAxCtrl	MMC_EM340SyncEncAxCtrl (  Page 216 MMC_EM340SyncEncAxCtrl)	Synchronous encoder axis control data
	SyncEncAxMntr	MMC_EM340SyncEncAxMntr (  Page 216 MMC_EM340SyncEncAxMntr)	Synchronous encoder axis monitor data
MMC_CcieSyncEncoder	SyncEncAxPrmViaLinkDev	MMC_EM340SyncEncAxPrmViaLinkDev (  Page 225 MMC_EM340SyncEncAxPrmViaLinkDev)	Synchronous encoder axis parameters via link device
MMC_CcieSlavelo	bRX	bool[]	R: RX area
	bRY	bool[]	RW: RY area
	wRX	unsigned short[]	R: RX area (for word access)
	wRY	unsigned short[]	RW: RY area (for word access)
	dwRX	unsigned long[]	R: RX area (for double word access)
	dwRY	unsigned long[]	RW: RY area (for double word access)
	wRWw	unsigned short[]	RW: RWw area
	wRWr	unsigned short[]	R: RWr area
	dwRWw	unsigned long[]	RW: RWw area (for double word access)
	dwRWr	unsigned long[]	R: RWr area (for double word access)
AXIS_REF_IO	Ready	bool	R: READY
	SynchronizationFlag	bool	R: Synchronization flag
	bRX	bool[]	R: RX area
	bRY	bool[]	RW: RY area
	wRX	unsigned short[]	R: RX area (for word access)
	wRY	unsigned short[]	RW: RY area (for word access)
	dwRX	unsigned long[]	R: RX area (for double word access)
	dwRY	unsigned long[]	RW: RY area (for double word access)
	wRWw	unsigned short[]	RW: RWw area
	wRWr	unsigned short[]	R: RWr area
	dwRWw	unsigned long[]	RW: RWw area (for double word access)
	dwRWr	unsigned long[]	R: RWr area (for double word access)
AXIS_REF_MOTION	Ready	bool	R: READY
	SynchronizationFlag	bool	R: Synchronization flag
	Busy	bool	R: BUSY
	PositioningStart	bool	RW: Positioning start
	ControllerInPositionRange	unsigned short	RW: [Pr.1190] Controller in-position range
	ControllerInPositionFlag	unsigned short	R: [Md.1190] Controller in-position flag



Category	Label name	Data type	Name
AXIS_REF_MOTION	SynchronousControlStart	bool	RW: [Cd.380] Synchronous control start
	SynchronousControlAnalysisMode	bool	RW: [Cd.381] Synchronous control analysis mode
	FLS_OperationDevice	bool	RW: [Cd.44] External input signal operation device: Upper limit signal (FLS)
	RLS_OperationDevice	bool	RW: [Cd.44] External input signal operation device: Lower limit signal (RLS)
	DOG_OperationDevice	bool	RW: [Cd.44] External input signal operation device: Proximity dog signal (DOG)
	STOP_OperationDevice	bool	RW: [Cd.44] External input signal operation device: Stop signal (STOP)
	AxPrm	MMC_EM340AxPrm ( <a href="#">Page 208</a> MMC_EM340AxPrm)	Parameter
	AxMntr	MMC_EM340AxMntr ( <a href="#">Page 209</a> MMC_EM340AxMntr)	Axis monitor data
	AxCtrl1	MMC_EM340AxCtrl1 ( <a href="#">Page 211</a> MMC_EM340AxCtrl1)	Axis control data
	PositData1	MMC_EM340PositData1 ( <a href="#">Page 214</a> MMC_EM340PositData1)	Positioning data
	BlockStartData	MMC_EM340BlockStartData ( <a href="#">Page 215</a> MMC_EM340BlockStartData)	Block start data
	AxCtrl2	MMC_EM340AxCtrl2 ( <a href="#">Page 215</a> MMC_EM340AxCtrl2)	Axis control data2
	SvInpAxPrm	MMC_EM340SvInpAxPrm ( <a href="#">Page 215</a> MMC_EM340SvInpAxPrm)	Servo input axis parameters
	SvInpAxMntr	MMC_EM340SvInpAxMntr ( <a href="#">Page 216</a> MMC_EM340SvInpAxMntr)	Servo input axis monitor data
	SyncCtrlPrm	MMC_EM340SyncAxPrm ( <a href="#">Page 217</a> MMC_EM340SyncAxPrm)	Synchronous control parameters
	SyncCtrlMntr	MMC_EM340SyncAxMntr ( <a href="#">Page 218</a> MMC_EM340SyncAxMntr)	Synchronous control monitor data
	SyncCtrlAxCtrl	MMC_EM340SyncAxCtrl ( <a href="#">Page 219</a> MMC_EM340SyncAxCtrl)	Synchronous control axis control data
	MarkSignalPrm	MMC_EM340MarkSignalPrm ( <a href="#">Page 220</a> MMC_EM340MarkSignalPrm)	Mark detection parameters
	MarkSignalCtrl	MMC_EM340MarkSignalCtrl ( <a href="#">Page 220</a> MMC_EM340MarkSignalCtrl)	Mark detection control data
	MarkSignalMntr	MMC_EM340MarkSignalMntr ( <a href="#">Page 220</a> MMC_EM340MarkSignalMntr)	Mark detection monitor data
	AxMntr2	MMC_EM340AxMntr2 ( <a href="#">Page 220</a> MMC_EM340AxMntr2)	Axis monitor data 2
	PositData2	MMC_EM340PositData2 ( <a href="#">Page 220</a> MMC_EM340PositData2)	Positioning data
	Interrupt	MMC_EM340Interrupt ( <a href="#">Page 221</a> MMC_EM340Interrupt)	Interrupt

## MMC\_EM340AxBPrm

Label name	Data type	Name
Unit	unsigned short	RW: [Pr.1] Unit setting
UnitMagnification	unsigned short	RW: [Pr.4] Unit magnification (AM)
PulsesPerRotation	unsigned long	RW: [Pr.2] Number of pulses per rotation (AP)
MovementAmountPerRotation	unsigned long	RW: [Pr.3] Movement amount per rotation (AL)
BiasSpeed	unsigned long	RW: [Pr.7] Bias speed at start
SpeedLimitValue	unsigned long	RW: [Pr.8] Speed limit value
AccelerationTime0	unsigned long	RW: [Pr.9] Acceleration time 0
DecelerationTime0	unsigned long	RW: [Pr.10] Deceleration time 0
BacklashAmount	unsigned short	RW: [Pr.11] Backlash compensation amount
SoftwareStrokeUpperLimit	long	RW: [Pr.12] Software stroke limit upper limit value
SoftwareStrokeLowerLimit	long	RW: [Pr.13] Software stroke limit lower limit value
SoftwareStrokeLimitMode	unsigned short	RW: [Pr.14] Software stroke limit selection
ManualControlSoftwareStrokeLimitValid	unsigned short	RW: [Pr.15] Software stroke limit valid/invalid setting
CommandInPositionWidth	unsigned long	RW: [Pr.16] Command in-position width
TorqueLimit	unsigned short	RW: [Pr.17] Torque limit setting value
M_CodeOutputTiming	unsigned short	RW: [Pr.18] M code ON signal output timing
SpeedSwitchingMode	unsigned short	RW: [Pr.19] Speed switching mode
InterpolationSpeedDesignation	unsigned short	RW: [Pr.20] Interpolation speed designation method
V_CommandPosition	unsigned short	RW: [Pr.21] Feed current value during speed control
InputSignalLogic	unsigned short	RW: [Pr.22] Input signal logic selection
SignalLogic_RLS	bool	RW: [Pr.22] Input signal logic selection Lower limit signal
SignalLogic_FLS	bool	RW: [Pr.22] Input signal logic selection Upper limit signal
SignalLogic_STOP	bool	RW: [Pr.22] Input signal logic selection Stop signal
SignalLogic_CHG	bool	RW: [Pr.22] Input signal logic selection External command/Switching signal
SignalLogic_DOG	bool	RW: [Pr.22] Input signal logic selection Proximity dog signal
VP_Mode	unsigned short	RW: [Pr.81] Speed-position function selection
ForcedStopSetting	unsigned short	RW: [Pr.82] Forced stop valid/invalid selection
AccelerationTime1	unsigned long	RW: [Pr.25] Acceleration time 1
AccelerationTime2	unsigned long	RW: [Pr.26] Acceleration time 2
AccelerationTime3	unsigned long	RW: [Pr.27] Acceleration time 3
DecelerationTime1	unsigned long	RW: [Pr.28] Deceleration time 1
DecelerationTime2	unsigned long	RW: [Pr.29] Deceleration time 2
DecelerationTime3	unsigned long	RW: [Pr.30] Deceleration time 3
JogSpeedLimit	unsigned long	RW: [Pr.31] JOG speed limit value
JogAccelerationTime	unsigned short	RW: [Pr.32] JOG operation acceleration time selection
JogDecelerationTime	unsigned short	RW: [Pr.33] JOG operation deceleration time selection
AccelerationDecelerationMode	unsigned short	RW: [Pr.34] Acceleration/deceleration process selection
S_CurveRatio	unsigned short	RW: [Pr.35] S-curve ratio
RapidStopDecelerationTime	unsigned long	RW: [Pr.36] Rapid stop deceleration time
StopGroup1RapidStop	unsigned short	RW: [Pr.37] Stop group 1 Rapid stop selection
StopGroup2RapidStop	unsigned short	RW: [Pr.38] Stop group 2 Rapid stop selection
StopGroup3RapidStop	unsigned short	RW: [Pr.39] Stop group 3 Rapid stop selection
PositioningCompleteOutputTime	unsigned short	RW: [Pr.40] Positioning complete signal output time
CircularInterpolationErrorRange	unsigned long	RW: [Pr.41] Allowable circular interpolation error width
DegreeAxisSpeedMode	unsigned short	RW: [Pr.83] Speed control 10 × multiplier setting for degree axis
RestartAllowableRange	unsigned long	RW: [Pr.84] Restart allowable range when servo OFF to ON

Label name	Data type	Name
SpdTrq_Operation	unsigned short	RW: [Pr.90] Operation setting for speed-torque control mode
HomingMethod	unsigned short	RW: [Pr.43] Home position return method
HomingDirection	unsigned short	RW: [Pr.44] Home position return direction
HomePosition	long	RW: [Pr.45] Home position address
HomingSpeed	unsigned long	RW: [Pr.46] Home position return speed
HomingAccelerationTime	unsigned short	RW: [Pr.51] Home position return acceleration time selection
HomingDecelerationTime	unsigned short	RW: [Pr.52] Home position return deceleration time selection
HomingIncompletionMode	unsigned short	RW: [Pr.55] Operation setting for incompletion of home position return
OperationCycleSetting	unsigned short	RW: [Pr.96] Operation cycle setting
FLS_SignalSelection	unsigned short	RW: [Pr.116] FLS signal selection
RLS_SignalSelection	unsigned short	RW: [Pr.117] RLS signal selection
DOG_SignalSelection	unsigned short	RW: [Pr.118] DOG signal selection
STOP_SignalSelection	unsigned short	RW: [Pr.119] STOP signal selection
MPG_SpeedLimitMode	unsigned short	RW: [Pr.122] Manual pulse generator speed limit mode
MPG_SpeedLimitValue	unsigned long	RW: [Pr.123] Manual pulse generator speed limit value

## MMC\_EM340AxMntr

Label name	Data type	Name
CommandPosition	long	R: [Md.20] Feed current value
MachineCommandPosition	long	R: [Md.21] Machine feed value
CommandSpeed	unsigned long	R: [Md.22] Feedrate
AxisErrorNo	unsigned short	R: [Md.23] Axis error No.
AxisWarningNo	unsigned short	R: [Md.24] Axis warning No.
M_Code	unsigned short	R: [Md.25] Valid M code
AxisOperationStatus	short	R: [Md.26] Axis operation status
CurrentSpeed	unsigned long	R: [Md.27] Current speed
AxisCommandSpeed	unsigned long	R: [Md.28] Axis feedrate
VP_MovementAmount	long	R: [Md.29] Speed-position switching control positioning movement amount
ExternalInputSignal	unsigned short	R: [Md.30] External input signal
RLS_Signal	bool	R: [Md.30] External input/output signal Lower limit signal
FLS_Signal	bool	R: [Md.30] External input/output signal Upper limit signal
STOP_Signal	bool	R: [Md.30] External input/output signal Stop signal
CHG_Signal	bool	R: [Md.30] External input/output signal External command signal
DOG_Signal	bool	R: [Md.30] External input/output signal Proximity dog signal
Status	unsigned short	R: [Md.31] Status
Status_SpeedControl	bool	R: [Md.31] Status Speed control flag
Status_VP_Latch	bool	R: [Md.31] Status Speed-position switching latch flag
Status_CommandInPosition	bool	R: [Md.31] Status Command in-position flag
Status_HomingRequest	bool	R: [Md.31] Status Home position return request flag
Status_HomingComplete	bool	R: [Md.31] Status Home position return complete flag
Status_PV_Latch	bool	R: [Md.31] Status Speed-position switching latch flag
Status_Warning	bool	R: [Md.31] Status Warning detection
Status_SpeedChange0	bool	R: [Md.31] Status Speed change 0 flag
Status_M_Code	bool	R: [Md.31] Status M code ON signal
Status_Error	bool	R: [Md.31] Status Error detection

Label name	Data type	Name
Status_PositioningStart	bool	R: [Md.31] Status Start complete
Status_PositioningComplete	bool	R: [Md.31] Status Positioning complete
TargetPosition	bool	R: [Md.32] Target value
TargetSpeed	unsigned long	R: [Md.33] Target speed
MPG_CarryoverMovementAmount	long	R: [Md.62] Manual pulse generator operation carry-over amount of movement
ForwardTorqueLimit	unsigned short	R: [Md.35] Torque limit stored value/forward torque limit stored value
SpecialStartInstructionCode	unsigned short	R: [Md.36] Special start data instruction code setting value
SpecialStartInstructionParameter	unsigned short	R: [Md.37] Special start data instruction parameter setting value
StartPositioningDataNo	unsigned short	R: [Md.38] Start positioning data No. setting value
InSpeedLimitFlag	unsigned short	R: [Md.39] In speed limit flag
InSpeedChangeFlag	unsigned short	R: [Md.40] In speed change processing flag
SpecialStartRepetitionCounter	unsigned short	R: [Md.41] Special start repetition counter
ControlSystemRepetitionCounter	unsigned short	R: [Md.42] Control system repetition counter
CurrentStartDataPointer	unsigned short	R: [Md.43] Start data pointer being executed
CurrentPositioningDataNo	unsigned short	R: [Md.44] Positioning data No. being executed
CurrentBlockNo	unsigned short	R: [Md.45] Block No. being executed
LatestPositioningDataNo	unsigned short	R: [Md.46] Last executed positioning data No.
CurrentPositioningIdentifier	unsigned short	R: [Md.47] Positioning data being executed Positioning identifier
CurrentM_Code	unsigned short	R: [Md.47] Positioning data being executed M code
CurrentDwellTime	unsigned short	R: [Md.47] Positioning data being executed Dwell time
PositioningOption	unsigned short	R: [Md.47] Positioning data being executed Positioning option
CurrentCommandSpeed	unsigned long	R: [Md.47] Positioning data being executed Command speed
CurrentPositioningAddress	long	R: [Md.47] Positioning data being executed Positioning address
CurrentArcAddress	long	R: [Md.47] Positioning data being executed Arc address
ActualPosition	long	R: [Md.101] Real current value
DeviationCounter	long	R: [Md.102] Deviation counter value
MotorRotationSpeed	long	R: [Md.103] Motor rotation speed
MotorCurrentValue	short	R: [Md.104] Motor current value
HomingOperationStatus	unsigned short	R: [Md.514] Home position return operation status
ServoStatus2	unsigned short	R: [Md.119] Servo status2
ServoStatus2_ZeroPointPass	bool	R: [Md.119] Servo status2 Zero point pass
ServoStatus2_ZeroSpeed	bool	R: [Md.119] Servo status2 Zero speed
ServoStatus2_SpeedLimit	bool	R: [Md.119] Servo status2 Speed limit
ServoStatus2_PID_Control	bool	R: [Md.119] Servo status2 PID control
ServoStatus1	unsigned short	R: [Md.108] Servo status1
ServoStatus1_Ready	bool	R: [Md.108] Servo status1 READY ON
ServoStatus1_Servo	bool	R: [Md.108] Servo status1 Servo ON
ServoStatus1_ControlMode	bool[2]	R: [Md.108] Servo status1 Control mode
ServoStatus1_GainSwitching	bool	R: [Md.108] Servo status1 Gain switching
ServoStatus1_FullyClosedLoopControlSwitching	bool	R: [Md.108] Servo status1 Fully closed loop control switching
ServoStatus1_Alarm	bool	R: [Md.108] Servo status1 Servo alarm
ServoStatus1_InPosition	bool	R: [Md.108] Servo status1 In-position
ServoStatus1_TorqueLimit	bool	R: [Md.108] Servo status1 Torque limit
ServoStatus1_AbsolutePositionLost	bool	R: [Md.108] Servo status1 Absolute position lost
ServoStatus1_Warning	bool	R: [Md.108] Servo status1 Servo warning

Label name	Data type	Name
Statusword	unsigned short	R: [Md.117] Statusword
SemiFullyClosedLoopStatus	unsigned short	R: [Md.113] Semi/Fully closed loop status
ServoAlarm	unsigned short	R: [Md.114] Servo alarm
ReverseTorqueLimit	unsigned short	R: [Md.120] Reverse torque limit stored value
SpeedDuringCommand	long	R: [Md.122] Speed during command
TorqueDuringCommand	short	R: [Md.123] Torque during command
CurrentInterpolatedAxis	unsigned long	R: [Md.47] Positioning data being executed interpolated axis
DecelerationStartFlag	unsigned short	R: [Md.48] Deceleration start flag

## MMC\_EM340SysMntr1

Label name	Data type	Name
TestModeStatus	short	R: [Md.1] In test mode flag
FW_Version	unsigned long	R: [Md.130] F/W version
OperationTime	unsigned short	R: [Md.134] Operation time
MaximumOperationTime	unsigned short	R: [Md.135] Maximum operation time
DigitalOscilloscopeStatus	short	R: [Md.131] Digital oscilloscope running
FlashRomWriteCount	unsigned long	R: [Md.19] Number of write accesses to flash ROM
ForcedStopInput	unsigned short	R: [Md.50] Forced stop input
AmplifierLessOperationStatus	unsigned short	R: [Md.51] Amplifier-less operation mode status
OperationCycle	unsigned short	R: [Md.132] Operation cycle setting
OperationCycleOver	unsigned short	R: [Md.133] Operation cycle over flag

## MMC\_EM340AxCtrl1

Label name	Data type	Name
PositioningStartNo	unsigned short	RW: [Cd.3] Positioning start No.
PositioningStartingPointNo	unsigned short	RW: [Cd.4] Positioning starting point No.
ResetAxisError	unsigned short	RW: [Cd.5] Axis error reset
Restart	unsigned short	RW: [Cd.6] Restart command
Clear_M_Code	unsigned short	RW: [Cd.7] M code OFF request
ExternalCommandValid	unsigned short	RW: [Cd.8] External command valid
NewPosition	long	RW: [Cd.9] New current value
NewAccelerationTime	unsigned long	RW: [Cd.10] New acceleration time value
NewDecelerationTime	unsigned long	RW: [Cd.11] New deceleration time value
AccelerationTimeChangeValid	unsigned short	RW: [Cd.12] Acceleration/deceleration time change value during speed change, enable/disable selection
Override	unsigned short	RW: [Cd.13] Positioning operation speed override
NewSpeed	unsigned long	RW: [Cd.14] New speed value
ChangeSpeed	unsigned short	RW: [Cd.15] Speed change request
InchingMovementAmount	unsigned short	RW: [Cd.16] Inching movement amount
JOG_Speed	unsigned long	RW: [Cd.17] JOG speed
InterruptOperation	unsigned short	RW: [Cd.18] Interrupt request during continuous operation
ClearHomingRequestFlag	unsigned short	RW: [Cd.19] Home position return request flag OFF request
MPG_InputMagnification	unsigned long	RW: [Cd.20] Manual pulse generator 1 pulse input magnification
EnableMPG	unsigned short	RW: [Cd.21] Manual pulse generator enable flag
ForwardNewTorque	unsigned short	RW: [Cd.22] New torque value/forward new torque value
VP_NewMovementAmount	unsigned long	RW: [Cd.23] Speed-position switching control movement amount change register
EnableVP_Switching	unsigned short	RW: [Cd.24] Speed-position switching enable flag


Label name	Data type	Name
PV_NewSpeed	unsigned long	RW: [Cd.25] Position-speed switching control speed change register
EnablePV_Switching	unsigned short	RW: [Cd.26] Position-speed switching enable flag
TargetNewPosition	long	RW: [Cd.27] Target position change value (New address)
TargetNewSpeed	unsigned long	RW: [Cd.28] Target position change value (New speed)
ChangeTargetPosition	unsigned short	RW: [Cd.29] Target position change request flag
SimultaneousStartNo	unsigned short	RW: [Cd.30] Simultaneous starting own axis start data No.
SimultaneousStartNoAxis1	unsigned short	RW: [Cd.31] Simultaneous starting axis start data No.1
SimultaneousStartNoAxis2	unsigned short	RW: [Cd.32] Simultaneous starting axis start data No.2
SimultaneousStartNoAxis3	unsigned short	RW: [Cd.33] Simultaneous starting axis start data No.3
StepMode	unsigned short	RW: [Cd.34] Step mode
StepValid	unsigned short	RW: [Cd.35] Step valid flag
StepStartInformation	unsigned short	RW: [Cd.36] Step start information
Skip	unsigned short	RW: [Cd.37] Skip command
TeachingDataSelection	unsigned short	RW: [Cd.38] Teaching data selection
TeachingPositioningDataNo	unsigned short	RW: [Cd.39] Teaching positioning data No.
ABS_DirectionInDegrees	unsigned short	RW: [Cd.40] ABS direction in degrees
RequestServoOff	unsigned short	RW: [Cd.100] Servo OFF command
TorqueOutputValue	unsigned short	RW: [Cd.101] Torque output setting value
ReadWriteServoParameter	unsigned short	RW: [Cd.130] Servo parameter read/write request
ServoParameterNo	unsigned short	RW: [Cd.131] Parameter No. (Object index for servo parameters to be changed)
ChangeData	unsigned short	RW: [Cd.132] Change data
SwitchSemiFullyClosed	unsigned short	RW: [Cd.133] Semi/Fully closed loop switching request
ChangeGain	unsigned short	RW: [Cd.108] Gain changing command flag
SwitchTorqueFunction	unsigned short	RW: [Cd.112] Torque change function switching request
ReverseNewTorque	unsigned short	RW: [Cd.113] New reverse torque value
SwitchPI_PID	unsigned short	RW: [Cd.136] PI-PID switching request
VP_PV_Device	unsigned short	RW: [Cd.45] Speed-position switching device selection
SwitchVP_PV	unsigned short	RW: [Cd.46] Speed-position switching command
SimultaneousStartAxis	unsigned long	RW: [Cd.43] Simultaneous starting axis
SwitchControlMode	unsigned short	RW: [Cd.138] Control mode switching request
ControlModeSetting	unsigned short	RW: [Cd.139] Control mode setting
Spd_CommandSpeed	long	RW: [Cd.140] Command speed at speed control mode
Spd_AccelerationTime	unsigned short	RW: [Cd.141] Acceleration time at speed control mode
Spd_DecelerationTime	unsigned short	RW: [Cd.142] Deceleration time at speed control mode
Trq_CommandTorque	short	RW: [Cd.143] Command torque at torque control mode
Trq_TorqueTimeConstDrivingMode	unsigned short	RW: [Cd.144] Torque time constant at torque control mode (Driving mode)
Trq_TorqueTimeConstRegenerativeMode	unsigned short	RW: [Cd.145] Torque time constant at torque control mode (Regenerative mode)
Trq_SpeedLimit	unsigned long	RW: [Cd.146] Speed limit value at torque control mode

## MMC\_EM340SysCtrl

Label name	Data type	Name
WriteFlashRom	unsigned short	RW: [Cd.1] Flash ROM write request
InitializeParameter	unsigned short	RW: [Cd.2] Parameter initialization request
DecelerationFlagValid	unsigned short	RW: [Cd.41] Deceleration start flag valid
DecelerationStopMode	unsigned short	RW: [Cd.42] Stop command processing for deceleration stop selection
SwitchAmplifierLessMode	unsigned short	RW: [Cd.137] Amplifier-less operation mode switching request
ExternalInputOperationDevice1	unsigned short	RW: [Cd.44] External input signal operation device (Axis 1 to 4)
Axis1_FLS	bool	RW: Axis 1 Upper limit signal (FLS)
Axis1_RLS	bool	RW: Axis 1 Lower limit signal (RLS)
Axis1_DOG	bool	RW: Axis 1 Proximity dog signal (DOG)
Axis1_STOP	bool	RW: Axis 1 Stop signal (STOP)
Axis2_FLS	bool	RW: Axis 2 Upper limit signal (FLS)
Axis2_RLS	bool	RW: Axis 2 Lower limit signal (RLS)
Axis2_DOG	bool	RW: Axis 2 Proximity dog signal (DOG)
Axis2_STOP	bool	RW: Axis 2 Stop signal (STOP)
Axis3_FLS	bool	RW: Axis 3 Upper limit signal (FLS)
Axis3_RLS	bool	RW: Axis 3 Lower limit signal (RLS)
Axis3_DOG	bool	RW: Axis 3 Proximity dog signal (DOG)
Axis3_STOP	bool	RW: Axis 3 Stop signal (STOP)
Axis4_FLS	bool	RW: Axis 4 Upper limit signal (FLS)
Axis4_RLS	bool	RW: Axis 4 Lower limit signal (RLS)
Axis4_DOG	bool	RW: Axis 4 Proximity dog signal (DOG)
Axis4_STOP	bool	RW: Axis 4 Stop signal (STOP)
ExternalInputOperationDevice2	unsigned short	RW: [Cd.44] External input signal operation device (Axis 5 to 8)
Axis5_FLS	bool	RW: Axis 5 Upper limit signal (FLS)
Axis5_RLS	bool	RW: Axis 5 Lower limit signal (RLS)
Axis5_DOG	bool	RW: Axis 5 Proximity dog signal (DOG)
Axis5_STOP	bool	RW: Axis 5 Stop signal (STOP)
Axis6_FLS	bool	RW: Axis 6 Upper limit signal (FLS)
Axis6_RLS	bool	RW: Axis 6 Lower limit signal (RLS)
Axis6_DOG	bool	RW: Axis 6 Proximity dog signal (DOG)
Axis6_STOP	bool	RW: Axis 6 Stop signal (STOP)
Axis7_FLS	bool	RW: Axis 7 Upper limit signal (FLS)
Axis7_RLS	bool	RW: Axis 7 Lower limit signal (RLS)
Axis7_DOG	bool	RW: Axis 7 Proximity dog signal (DOG)
Axis7_STOP	bool	RW: Axis 7 Stop signal (STOP)
Axis8_FLS	bool	RW: Axis 8 Upper limit signal (FLS)
Axis8_RLS	bool	RW: Axis 8 Lower limit signal (RLS)
Axis8_DOG	bool	RW: Axis 8 Proximity dog signal (DOG)
Axis8_STOP	bool	RW: Axis 8 Stop signal (STOP)
ExternalInputOperationDevice3	unsigned short	RW: [Cd.44] External input signal operation device (Axis 9 to 12)
Axis9_FLS	bool	RW: Axis 9 Upper limit signal (FLS)
Axis9_RLS	bool	RW: Axis 9 Lower limit signal (RLS)
Axis9_DOG	bool	RW: Axis 9 Proximity dog signal (DOG)
Axis9_STOP	bool	RW: Axis 9 Stop signal (STOP)
Axis10_FLS	bool	RW: Axis 10 Upper limit signal (FLS)
Axis10_RLS	bool	RW: Axis 10 Lower limit signal (RLS)
Axis10_DOG	bool	RW: Axis 10 Proximity dog signal (DOG)

Label name	Data type	Name
Axis10_STOP	bool	RW: Axis 10 Stop signal (STOP)
Axis11_FLS	bool	RW: Axis 11 Upper limit signal (FLS)
Axis11_RLS	bool	RW: Axis 11 Lower limit signal (RLS)
Axis11_DOG	bool	RW: Axis 11 Proximity dog signal (DOG)
Axis11_STOP	bool	RW: Axis 11 Stop signal (STOP)
Axis12_FLS	bool	RW: Axis 12 Upper limit signal (FLS)
Axis12_RLS	bool	RW: Axis 12 Lower limit signal (RLS)
Axis12_DOG	bool	RW: Axis 12 Proximity dog signal (DOG)
Axis12_STOP	bool	RW: Axis 12 Stop signal (STOP)
ExternalInputOperationDevice4	unsigned short	RW: [Cd.44] External input signal operation device (Axis 13 to 16)
Axis13_FLS	bool	RW: Axis 13 Upper limit signal (FLS)
Axis13_RLS	bool	RW: Axis 13 Lower limit signal (RLS)
Axis13_DOG	bool	RW: Axis 13 Proximity dog signal (DOG)
Axis13_STOP	bool	RW: Axis 13 Stop signal (STOP)
Axis14_FLS	bool	RW: Axis 14 Upper limit signal (FLS)
Axis14_RLS	bool	RW: Axis 14 Lower limit signal (RLS)
Axis14_DOG	bool	RW: Axis 14 Proximity dog signal (DOG)
Axis14_STOP	bool	RW: Axis 14 Stop signal (STOP)
Axis15_FLS	bool	RW: Axis 15 Upper limit signal (FLS)
Axis15_RLS	bool	RW: Axis 15 Lower limit signal (RLS)
Axis15_DOG	bool	RW: Axis 15 Proximity dog signal (DOG)
Axis15_STOP	bool	RW: Axis 15 Stop signal (STOP)
Axis16_FLS	bool	RW: Axis 16 Upper limit signal (FLS)
Axis16_RLS	bool	RW: Axis 16 Lower limit signal (RLS)
Axis16_DOG	bool	RW: Axis 16 Proximity dog signal (DOG)
Axis16_STOP	bool	RW: Axis 16 Stop signal (STOP)
ForcedStopInput	unsigned short	RW: [Cd.158] Forced stop input
VirtualSvOperation	unsigned short	RW: [Cd.700] Virtual servo amplifier operation setting
VirtualSvOperationStation	unsigned short	RW: [Cd.701] Virtual servo amplifier operation station

## MMC\_EM340PositData1


Label name	Data type	Name
PositDataEle1	MMC_EM340PositDataEle1[600] (  Page 214 MMC_EM340PositDataEle1)	Positioning data element

## MMC\_EM340PositDataEle1



Label name	Data type	Name
Identifier	unsigned short	RW: Positioning identifier [Da.1] Operation pattern [Da.2] Control method [Da.3] Acceleration time No. [Da.4] Deceleration time No.
M_Code	unsigned short	RW: [Da.10] M code/Condition data No./Number of LOOP to LEND repetitions
DwellTime	unsigned short	RW: [Da.9] Dwell time/JUMP destination positioning data No.
Option	unsigned short	RW: Positioning option [Da.27] M code ON signal output timing [Da.28] ABS direction in degrees [Da.29] Interpolation speed designation method
CommandSpeed	unsigned long	RW: [Da.8] Command speed
Position_MovementAmount	long	RW: [Da.6] Positioning address/movement amount
ArcPosition	long	RW: [Da.7] Arc address



## MMC\_EM340BlockStartData

Label name	Data type	Name
StartBlock	MMC_EM340StartBlock[5] (  Page 215 MMC_EM340StartBlock)	Start block

## MMC\_EM340StartBlock

Label name	Data type	Name
StartData	MMC_EM340StartData (  Page 215 MMC_EM340StartData)	Start data
ConditionData	MMC_EM340ConditionData[10] (  Page 215 MMC_EM340ConditionData)	Condition data

## MMC\_EM340StartData

Label name	Data type	Name
Shape_StartDataNo	unsigned short[50]	RW: [Da.11] Shape, [Da.12] Start data No.
SpecialStartParameter	unsigned short[50]	RW: [Da.13] Special start instruction, [Da.14] Parameter

## MMC\_EM340ConditionData

Label name	Data type	Name
Condition	unsigned short	RW: [Da.15] Condition target, [Da.16] Condition operator
BufferMemoryNo	unsigned long	RW: [Da.17] Address
Parameter1	unsigned long	RW: [Da.18] Parameter 1
Parameter2	unsigned long	RW: [Da.19] Parameter 2
SimultaneousStartAxis	unsigned long	RW: Number of simultaneous starting axes [Da.23] Number of simultaneous starting axes [Da.24] Simultaneous starting axis No.1 [Da.25] Simultaneous starting axis No.2 [Da.26] Simultaneous starting axis No.3

## MMC\_EM340AxCtrl2

Label name	Data type	Name
StopAxis	unsigned short	RW: [Cd.180] Axis stop
StartForwardJOG	unsigned short	RW: [Cd.181] Forward run JOG start
StartReverseJOG	unsigned short	RW: [Cd.182] Reverse run JOG start
ProhibitPositioning	unsigned short	RW: [Cd.183] Execution prohibition flag

## MMC\_EM340SysMntr2

Label name	Data type	Name
ModuleInformation	unsigned short	R: [Md.59] Module information

## MMC\_EM340SvInpAxPrm

Label name	Data type	Name
Type	unsigned short	RW: [Pr.300] Servo input axis type
SmoothingTimeConstant	unsigned short	RW: [Pr.301] Servo input axis smoothing time constant
PhaseCompensationAdvanceTime	long	RW: [Pr.302] Servo input axis phase compensation advance time
PhaseCompensationTimeConstant	unsigned short	RW: [Pr.303] Servo input axis phase compensation time constant
RotationRestriction	unsigned short	RW: [Pr.304] Servo input axis rotation direction restriction

## MMC\_EM340SvInpAxMntr

Label name	Data type	Name
CommandPosition	long	R: [Md.300] Servo input axis current value
Speed	long	R: [Md.301] Servo input axis speed
PhaseCompensationAmount	long	R: [Md.302] Servo input axis phase compensation amount
RotationRestrictionAmount	long	R: [Md.303] Servo input axis rotation direction restriction amount

## MMC\_EM340SyncEncAxPrm

Label name	Data type	Name
Type	unsigned short	RW: [Pr.320] Synchronous encoder axis type
Unit	unsigned short	RW: [Pr.321] Synchronous encoder axis unit setting
UnitConversionNumerator	long	RW: [Pr.322] Synchronous encoder axis unit conversion: Numerator
UnitConversionDenominator	long	RW: [Pr.323] Synchronous encoder axis unit conversion: Denominator
LengthPerCycle	long	RW: [Pr.324] Synchronous encoder axis length per cycle
SmoothingTimeConstant	unsigned short	RW: [Pr.325] Synchronous encoder axis smoothing time constant
PhaseCompensationAdvanceTime	long	RW: [Pr.326] Synchronous encoder axis phase compensation advance time
PhaseCompensationTimeConstant	unsigned short	RW: [Pr.327] Synchronous encoder axis phase compensation time constant
RotationRestriction	unsigned short	RW: [Pr.328] Synchronous encoder axis rotation direction restriction
SyncEncViaCPU_Resolution	long	RW: [Pr.329] Resolution of synchronous encoder via buffer memory

## MMC\_EM340SyncEncAxCtrl

Label name	Data type	Name
StartControl	unsigned short	RW: [Cd.320] Synchronous encoder axis control start
ControlMethod	unsigned short	RW: [Cd.321] Synchronous encoder axis control method
NewPosition	long	RW: [Cd.322] Synchronous encoder axis current value setting address
ResetError	unsigned short	RW: [Cd.323] Synchronous encoder axis error reset
EnableSyncEncViaCPU	unsigned short	RW: [Cd.324] Connection command of synchronous encoder via buffer memory
SyncEncViaCPU_InputValue	long	RW: [Cd.325] Input value for synchronous encoder via buffer memory

## MMC\_EM340SyncEncAxMntr

Label name	Data type	Name
CommandPosition	long	R: [Md.320] Synchronous encoder axis current value
CommandPositionPerCycle	long	R: [Md.321] Synchronous encoder axis current value per cycle
Speed	long	R: [Md.322] Synchronous encoder axis speed
PhaseCompensationAmount	long	R: [Md.323] Synchronous encoder axis phase compensation amount
RotationRestrictionAmount	long	R: [Md.324] Synchronous encoder axis rotation direction restriction amount
Status	unsigned short	R: [Md.325] Synchronous encoder axis status
Status_SettingValid	bool	R: [Md.325] Synchronous encoder axis status Setting valid flag

Label name	Data type	Name
Status_ConnectingValid	bool	R: [Md.325] Synchronous encoder axis status Connecting valid flag
Status_CounterEnable	bool	R: [Md.325] Synchronous encoder axis status Counter enable flag
Status_CurrentValueSetRequest	bool	R: [Md.325] Synchronous encoder axis status Current value setting request flag
Status_Error	bool	R: [Md.325] Synchronous encoder axis status Error detection flag
Status_Warning	bool	R: [Md.325] Synchronous encoder axis status Warning detection flag
Status_StartRequest	bool	R: [Md.325] Synchronous encoder axis status Start request flag
ErrorNo	unsigned short	R: [Md.326] Synchronous encoder axis error No.
WarningNo	unsigned short	R: [Md.327] Synchronous encoder axis warning No.

## MMC\_EM340SyncSysCtrl

Label name	Data type	Name
Start	unsigned short	RW: [Cd.380] Synchronous control start
AnalysisMode	unsigned short	RW: [Cd.381] Synchronous control analysis mode

## MMC\_EM340SyncAxPrm

Label name	Data type	Name
MainInputAxisNo	unsigned short	RW: [Pr.400] Main input axis No.
SubInputAxisNo	unsigned short	RW: [Pr.401] Sub input axis No.
MainShaftCompositeGear	unsigned short	RW: [Pr.402] Composite main shaft gear
MainShaftGearNumerator	long	RW: [Pr.403] Main shaft gear: Numerator
MainShaftGearDenominator	long	RW: [Pr.404] Main shaft gear: Denominator
MainShaftClutchControlMode	unsigned short	RW: [Pr.405] Main shaft clutch control setting
MainShaftClutchReferencePosition	unsigned short	RW: [Pr.406] Main shaft clutch reference address setting
MainShaftClutchOnPosition	long	RW: [Pr.407] Main shaft clutch ON address
MainShaftMovementAmountBeforeClutchOn	long	RW: [Pr.408] Movement amount before main shaft clutch ON
MainShaftClutchOffPosition	long	RW: [Pr.409] Main shaft clutch OFF address
MainShaftMovementAmountBeforeClutchOff	long	RW: [Pr.410] Movement amount before main shaft clutch OFF
MainShaftClutchSmoothMethod	unsigned short	RW: [Pr.411] Main shaft clutch smoothing method
MainShaftClutchSmoothTimeConstant	unsigned short	RW: [Pr.412] Main shaft clutch smoothing time constant
MainShaftClutchOnSlipAmount	unsigned long	RW: [Pr.413] Slippage at main shaft clutch ON
MainShaftClutchOffSlipAmount	unsigned long	RW: [Pr.414] Slippage at main shaft clutch OFF
AuxShaftAxisNo	unsigned short	RW: [Pr.418] Auxiliary shaft axis No.
AuxShaftCompositeGear	unsigned short	RW: [Pr.419] Composite auxiliary shaft gear
AuxShaftGearNumerator	long	RW: [Pr.420] Auxiliary shaft gear: Numerator
AuxShaftGearDenominator	long	RW: [Pr.421] Auxiliary shaft gear: Denominator
AuxShaftClutchControlMode	unsigned short	RW: [Pr.422] Auxiliary shaft clutch control setting
AuxShaftClutchReferencePosition	unsigned short	RW: [Pr.423] Auxiliary shaft clutch reference address setting
AuxShaftClutchOnPosition	long	RW: [Pr.424] Auxiliary shaft clutch ON address
AuxShaftMovementAmountBeforeClutchOn	long	RW: [Pr.425] Movement amount before auxiliary shaft clutch ON
AuxShaftClutchOffPosition	long	RW: [Pr.426] Auxiliary shaft clutch OFF address
AuxShaftMovementAmountBeforeClutchOff	long	RW: [Pr.427] Movement amount before auxiliary shaft clutch OFF
AuxShaftClutchSmoothMethod	unsigned short	RW: [Pr.428] Auxiliary shaft clutch smoothing method

Label name	Data type	Name
AuxShaftClutchSmoothTimeConstant	unsigned short	RW: [Pr.429] Auxiliary shaft clutch smoothing time constant
AuxShaftClutchOnSlipAmount	unsigned long	RW: [Pr.430] Slippage at auxiliary shaft clutch ON
AuxShaftClutchOffSlipAmount	unsigned long	RW: [Pr.431] Slippage at auxiliary shaft clutch OFF
SpeedChangeGear	unsigned short	RW: [Pr.434] Speed change gear
SpeedChangeGearSmoothTimeConstant	unsigned short	RW: [Pr.435] Speed change gear smoothing time constant
SpeedChangeRatioNumerator	long	RW: [Pr.436] Speed change ratio: Numerator
SpeedChangeRatioDenominator	long	RW: [Pr.437] Speed change ratio: Denominator
CamAxisCycleUnit	unsigned short	RW: [Pr.438] Cam axis cycle unit setting
CamAxisLengthPerCycleChangeMode	unsigned short	RW: [Pr.442] Cam axis length per cycle change setting
CamAxisLengthPerCycle	unsigned long	RW: [Pr.439] Cam axis length per cycle
CamNo	unsigned short	RW: [Pr.440] Cam No.
CamStrokeAmount	long	RW: [Pr.441] Cam stroke amount
CamAxisPhaseCompensationAdvanceTime	long	RW: [Pr.444] Cam axis phase compensation advance time
CamAxisPhaseCompensationTimeConstant	unsigned short	RW: [Pr.445] Cam axis phase compensation time constant
SyncCtrlDecelerationTime	unsigned short	RW: [Pr.446] Synchronous control deceleration time
OutputAxisSmoothTimeConstant	unsigned short	RW: [Pr.447] Output axis smoothing time constant
MainShaftGearPositionPerCycleMode	unsigned short	RW: [Pr.460] Setting method of current value per cycle after main shaft gear
AuxShaftGearPositionPerCycleMode	unsigned short	RW: [Pr.461] Setting method of current value per cycle after auxiliary shaft gear
CamPositionRestorationMode	unsigned short	RW: [Pr.462] Cam axis position restoration object
CamReferencePositionMode	unsigned short	RW: [Pr.463] Setting method of cam reference position
CamPositionPerCycleMode	unsigned short	RW: [Pr.464] Setting method of cam axis current value per cycle
MainShaftGearInitialPositionPerCycle	unsigned long	RW: [Pr.465] Current value per cycle after main shaft gear (initial setting)
AuxShaftGearInitialPositionPerCycle	unsigned long	RW: [Pr.466] Current value per cycle after auxiliary shaft gear (initial setting)
CamInitialReferencePosition	long	RW: [Pr.467] Cam reference position (initial setting)
CamInitialPositionPerCycle	unsigned long	RW: [Pr.468] Cam axis current value per cycle (initial setting)

## MMC\_EM340SyncAxMntr



Label name	Data type	Name
MainShaftCompositeGearPosition	long	R: [Md.400] Current value after composite main shaft gear
MainShaftGearPositionPerCycle	unsigned long	R: [Md.401] Current value per cycle after main shaft gear
AuxShaftGearPositionPerCycle	unsigned long	R: [Md.402] Current value per cycle after auxiliary shaft gear
CamPhaseCompensationAmount	long	R: [Md.406] Cam axis phase compensation amount
CamPositionPerCycle	unsigned long	R: [Md.407] Cam axis current value per cycle
CamReferencePosition	long	R: [Md.408] Cam reference position
CamCommandPosition	long	R: [Md.409] Cam axis feed current value
CurrentCamNo	unsigned short	R: [Md.410] Execute cam No.
CurrentCamStrokeAmount	long	R: [Md.411] Execute cam stroke amount
CurrentCamAxisLengthPerCycle	long	R: [Md.412] Execute cam axis length per cycle
MainShaftClutchStatus	unsigned short	R: [Md.420] Main shaft clutch ON/OFF status
MainShaftClutchSmoothingStatus	unsigned short	R: [Md.421] Main shaft clutch smoothing status
MainShaftClutchAccumulativeSlippage	long	R: [Md.422] Main shaft clutch slippage (accumulative)
AuxShaftClutchStatus	unsigned short	R: [Md.423] Auxiliary shaft clutch ON/OFF status
AuxShaftClutchSmoothingStatus	unsigned short	R: [Md.424] Auxiliary shaft clutch smoothing status

Label name	Data type	Name
AuxShaftClutchAccumulativeSlippage	long	R: [Md.425] Auxiliary shaft clutch slippage (accumulative)

## MMC\_EM340SyncAxCtrl

Label name	Data type	Name
CommandMainShaftClutch	unsigned short	RW: [Cd.400] Main shaft clutch command
CommandMainShaftClutchInvalid	unsigned short	RW: [Cd.401] Main shaft clutch control invalid command
CommandMainShaftClutchForcedOff	unsigned short	RW: [Cd.402] Main shaft clutch forced OFF command
CommandAuxShaftClutch	unsigned short	RW: [Cd.403] Auxiliary shaft clutch command
CommandAuxShaftClutchInvalid	unsigned short	RW: [Cd.404] Auxiliary shaft clutch control invalid command
CommandAuxShaftClutchForcedOff	unsigned short	RW: [Cd.405] Auxiliary shaft clutch forced OFF command
RequestSyncCtrlChange	unsigned short	RW: [Cd.406] Synchronous control change request
SyncCtrlChangeCommand	unsigned short	RW: [Cd.407] Synchronous control change command
SyncCtrlChangeValue	long	RW: [Cd.408] Synchronous control change value
SyncCtrlReflectionTime	unsigned short	RW: [Cd.409] Synchronous control reflection time

## MMC\_EM340CamCalculation

Label name	Data type	Name
Ctrl	MMC_EM340Ctrl (  Page 219 MMC_EM340Ctrl)	Cam position calculation control data
Mntr	MMC_EM340Mntr (  Page 219 MMC_EM340Mntr)	Cam position calculation monitor data

## MMC\_EM340Ctrl

Label name	Data type	Name
Request	unsigned short	RW: [Cd.612] Cam position calculation request
CamNo	unsigned short	RW: [Cd.613] Cam position calculation: Cam No.
Stroke	long	RW: [Cd.614] Cam position calculation: Stroke amount
LengthPerCycle	unsigned long	RW: [Cd.615] Cam position calculation: Cam axis length per cycle
ReferencePosition	long	RW: [Cd.616] Cam position calculation: Cam reference position
CommandPositionPerCycle	unsigned long	RW: [Cd.617] Cam position calculation: Cam axis current value per cycle
CommandPosition	long	RW: [Cd.618] Cam position calculation: Cam axis feed current value

## MMC\_EM340Mntr

Label name	Data type	Name
Result	long	R: [Md.600] Cam position calculation result

## MMC\_EM340MarkSignalPrm

Label name	Data type	Name
CompensationTime	short	RW: [Pr.801] Mark detection signal compensation time
Type	short	RW: [Pr.802] Mark detection data type
AxisNo	unsigned short	RW: [Pr.803] Mark detection data axis No.
BufferMemoryNo	unsigned long	RW: [Pr.804] Mark detection data buffer memory No.
LatchDataRangeUpperLimit	long	RW: [Pr.805] Latch data range upper limit value
LatchDataRangeLowerLimit	long	RW: [Pr.806] Latch data range lower limit value
DetectionMode	short	RW: [Pr.807] Mark detection mode setting
LinkDeviceType	unsigned short	RW: [Pr.808] Mark detection signal link device type
LinkDeviceNo	unsigned short	RW: [Pr.809] Mark detection signal link device No.
LinkDeviceBitSpecification	unsigned short	RW: [Pr.810] Mark detection signal link device bit specification
DetectionDirection	unsigned short	RW: [Pr.811] Mark detection signal detection direction setting

## MMC\_EM340MarkSignalCtrl

Label name	Data type	Name
ClearDetectionNumber	unsigned short	RW: [Cd.800] Number of mark detection clear request
InvalidDetection	unsigned short	RW: [Cd.801] Mark detection invalid flag
ChangeLatchDataRange	unsigned short	RW: [Cd.802] Latch data range change request

## MMC\_EM340MarkSignalMntr

Label name	Data type	Name
DetectionCounter	unsigned short	R: [Md.800] Number of mark detection counter
DetectionData	long[32]	R: [Md.801] Mark detection data storage area
DetectionMonitor	unsigned short	R: [Md.802] Mark detection signal monitor

## MMC\_EM340CmnPrm

Label name	Data type	Name
MaxAxes	unsigned short	RW: [Pr.152] Maximum number of control axes

## MMC\_EM340SvNetPrm

Label name	Data type	Name
ConnectedDevice	unsigned long	RW: [Pr.100] Connected device
VirtualSvSetting	unsigned short	RW: [Pr.101] Virtual servo amplifier setting


## MMC\_EM340SvNetMntr

Label name	Data type	Name
ConnectedDevice	unsigned long	R: [Md.105] Connected device

## MMC\_EM340AxMntr2

Label name	Data type	Name
PreReadingAnalysisStatus	unsigned short	R: [Md.503] Pre-reading data analysis status
ExternalSignalMonitor	unsigned short	R: [Md.900] External command signal monitor



## MMC\_EM340PositData2

Label name	Data type	Name
PositDataEle2	MMC_EM340PositDataEle2[600] (  Page 221 MMC_EM340PositDataEle2)	Positioning data element

## MMC\_EM340PositDataEle2

Label name	Data type	Name
InterpolationAxis	unsigned long	RW: Axis to be interpolated [Da.20] Axis to be interpolated No.1 [Da.21] Axis to be interpolated No.2 [Da.22] Axis to be interpolated No.3

## MMC\_EM340StartHistory

Label name	Data type	Name
Info	MMC_EM340Info (  Page 221 MMC_EM340Info)	Start history information
Data	MMC_EM340Data[64] (  Page 221 MMC_EM340Data)	Start history (0 to 63)




## MMC\_EM340Info

Label name	Data type	Name
StartHistoryPointer	unsigned short	R: [Md.8] Start history pointer

## MMC\_EM340Data

Label name	Data type	Name
StartInfo	unsigned short	R: [Md.3] Start information
StartNo	unsigned short	R: [Md.4] Start No.
StartYearMonth	unsigned short	R: [Md.54] Start Year:month
StartDayHour	unsigned short	R: [Md.5] Start Day:hour
StartMinSec	unsigned short	R: [Md.6] Start Minute:second
StartMsec	unsigned short	R: [Md.60] Start milli second
ErrorNo	unsigned short	R: [Md.7] Error judgement

## MMC\_EM340Interrupt

Label name	Data type	Name
IntMode	unsigned short	RW: [Pr.1100] Interrupt mode selection
InterruptPrm	MMC_EM340InterruptPrm[64] (  Page 221 MMC_EM340InterruptPrm)	Interrupt parameter
InterruptCmd	MMC_EM340InterruptCmd (  Page 221 MMC_EM340InterruptCmd)	Interrupt control data
InterruptMntr	MMC_EM340InterruptMntr (  Page 222 MMC_EM340InterruptMntr)	Interrupt monitor data

## MMC\_EM340InterruptPrm

Label name	Data type	Name
IntFactor	unsigned long	RW: [Pr.1101] Interrupt factor setting
IntDataCondition	unsigned short	RW: [Pr.1102] Interrupt data condition
IntJudgeCondition	unsigned short	RW: [Pr.1103] Interrupt judge condition
IntJudgeValue	unsigned long[2]	RW: [Pr.1104] Interrupt condition judge value 1, [Pr.1105] Interrupt condition judge value 2
IntTime	unsigned short	RW: [Pr.1106] Time for condition completion
IntPointer	unsigned long	RW: [Pr.1107] Interrupt pointer



## MMC\_EM340InterruptCmd

Label name	Data type	Name
IntEnable	unsigned short	RW: [Cd.1100] Interrupt enable request
IntReset	unsigned short	RW: [Cd.1101] Interrupt factor reset request


## MMC\_EM340InterruptMntr

Label name	Data type	Name
IntFactorOut	unsigned short[4]	R: [Md.1100] During interrupt factor output
IntFactor	unsigned long[2]	R: [Md.1101] Interrupt factor
IntFactorDetail	unsigned long[64]	R: [Md.1102] Interrupt factor details

## MMC\_EM340OptionalBitMntr

Label name	Data type	Name
OptionalBitMntrPrm	MMC_EM340OptionalBitMntrPrm[2] (  Page 222 MMC_EM340OptionalBitMntrPrm)	Optional bit monitor output parameter
OptionalBitMntrData	MMC_EM340OptionalBitMntrData (  Page 222 MMC_EM340OptionalBitMntrData)	Optional bit monitor data

## MMC\_EM340OptionalBitMntrPrm

Label name	Data type	Name
OptionalBitMntrSetting	MMC_EM340OptionalBitMntrSetting[32] (  Page 222 MMC_EM340OptionalBitMntrSetting)	Optional bit monitor output setting

## MMC\_EM340OptionalBitMntrSetting

Label name	Data type	Name
OutputAddress	unsigned long	RW: [Pr.1110] Optional bit output 1 address, [Pr.1112] Optional bit output 2 address
OutputBitNumber	unsigned short	RW: [Pr.1111] Optional bit output 1 bit No., [Pr.1113] Optional bit output 2 bit No.





## MMC\_EM340OptionalBitMntrData

Label name	Data type	Name
MonitorData	unsigned long[2]	R: [Md.1110] Optional bit monitor 1, [Md.1111] Optional bit monitor 2

## MMC\_EM340AllAxMntr

Label name	Data type	Name
ErrorAllAxis	unsigned long	R: [Md.1130] Error detection (all axes)
ControllerInPositionAllAxis	unsigned long	R: [Md.1131] Controller in-position (all axes)
WarningAllAxis	unsigned long	R: [Md.1132] Axis warning detection (all axes)
DirectCtrlEnableFlagAllAxis	unsigned long	R: [Md.1134] Direct control enable flag (all axes)

## MMC\_EM340DirectCtrl

Label name	Data type	Name
DirectCtrlPrm	MMC_EM340DirectCtrlPrm[16] (  Page 223 MMC_EM340DirectCtrlPrm)	Direct control setting parameter
DirectCtrlMntr	MMC_EM340DirectCtrlMntr[16] (  Page 223 MMC_EM340DirectCtrlMntr)	Direct control monitor data
DirectCtrlData	MMC_EM340DirectCtrlData[16] (  Page 223 MMC_EM340DirectCtrlData)	Control data for direct control
DirectCtrlCmdBuff	MMC_EM340DirectCtrlCmdBuff[16] (  Page 223 MMC_EM340DirectCtrlCmdBuff)	Direct control command buffer



## MMC\_EM340DirectCtrlPrm

Label name	Data type	Name
UseBufferNumber	unsigned short	RW: [Pr.1210] Direct control number of used buffers


## MMC\_EM340DirectCtrlMntr

Label name	Data type	Name
OperationStatus	short	R: [Md.1210] Direct control operation status
EnableFlag	unsigned short	R: [Md.1211] Direct control enable flag
CurrentBufferNumberPos	unsigned short	R: [Md.1212] Direct control buffer No. being executed (position control)
MaximumBufferNumberPos	unsigned short	R: [Md.1213] Direct control maximum buffer No. (position control)

## MMC\_EM340DirectCtrlData

Label name	Data type	Name
StartReq	unsigned short	RW: [Cd.1210] Direct control start request
WriteBufferNumberPos	unsigned short	RW: [Cd.1212] Direct control writing complete buffer No. (position control)

## MMC\_EM340DirectCtrlCmdBuff

Label name	Data type	Name
DirectCtrlCmdBuffPos	MMC_EM340DirectCtrlCmdBuffPos (  Page 223 MMC_EM340DirectCtrlCmdBuffPos)	Direct control command buffer (position control)

## MMC\_EM340DirectCtrlCmdBuffPos

Label name	Data type	Name
PositionCommandData	long[16]	RW: [Cd.1220] Position command data

## MMC\_EM340CamOperation

Label name	Data type	Name
Request	unsigned short	RW: [Cd.600] Cam data operation request
CamNo	unsigned short	RW: [Cd.601] Operation cam No.
FirstPosition	unsigned short	RW: [Cd.602] Cam data first position
Size	unsigned short	RW: [Cd.603] Number of cam data operation points
Format	unsigned short	RW: [Cd.604] Cam data format
CamResolution	unsigned short	RW: [Cd.605] Cam resolution/coordinate number
StartingPoint	unsigned short	RW: [Cd.606] Cam data starting point

## MMC\_EM340UserWatchDog

Label name	Data type	Name
StartCounter	unsigned short	RW: [Cd.1140] Watchdog timer start counter
CheckCounter	unsigned short	RW: [Cd.1141] Watchdog check counter
Timer	unsigned short	R: [Md.1140] Watchdog timer

## MMC\_EM340ControllerInPosition

Label name	Data type	Name
Range	unsigned short[16]	RW: [Pr.1190] Controller in-position range
Flag	unsigned short[16]	R: [Md.1190] Controller in-position flag

## MMC\_EM340BoardMntr

Label name	Data type	Name
BoardInfo	unsigned short	R: [Md.1200] Board information
FpgaVersion	unsigned short	R: [Md.1201] FPGA version
ExTransferSize	unsigned long	R: [Md.1202] Expansion transmission maximum size
CrcErrorLocation	unsigned short	R: [Md.1203] CRC error occurrence location
ProductionNo	char[16]	R: [Md.138] Production No.
LinkDeviceAreaInfoReady	unsigned short	R: [Md.1204] Link device offset/size information output status

## MMC\_EM340RemoteOperation

Label name	Data type	Name
Reset	unsigned short	RW: [Cd.1180] Remote RESET start

## MMC\_EM340BitDevice

Label name	Data type	Name
Input	unsigned long	R: Input signal
Ready	bool	R: READY
SynchronizationFlag	bool	R: Synchronization flag
Busy	bool[16]	R: BUSY (Axis No. 1 to 16)
Output	unsigned long	RW: Output signal
UserProgramReady	bool	RW: User program ready
AllAxisServoOn	bool	RW: All axis servo ON
PositioningStart	bool[16]	RW: Positioning start (Axis No. 1 to 16)



## MMC\_EM340DmaTransfer

Label name	Data type	Name
DestinationAddress	unsigned long[2]	RW: [Pr.1150] DMA destination address
AllocateSize	unsigned short	RW: [Pr.1151] DMA transmission allocated size
RxTopNumber	unsigned short	RW: [Pr.1152] RX transmission No.
RxSize	unsigned short	RW: [Pr.1153] RX transmission points
RWrTopNumber	unsigned short	RW: [Pr.1154] RWr transmission No.
RWrSize	unsigned short	RW: [Pr.1155] RWr transmission points
EnableRequest	unsigned short	RW: [Cd.1150] DMA transmission enable request
Status	unsigned short	R: [Md.1150] DMA transmission status
MntrDataSize	unsigned short	R: [Md.1151] Monitor data transmission size

## MMC\_EM340TimeSetting

Label name	Data type	Name
StartupTime	unsigned long	RW: [Pr.1160] Startup time

## MMC\_EM340SyncEncAxPrmViaLinkDev

Label name	Data type	Name
SyncEncAx	MMC_EM340LinkDevCHGWordInfoSync (  Page 225 MMC_EM340LinkDevCHGWordInfoSync)	Synchronous encoder axis
SyncEncStartReq	MMC_EM340LinkDevInfoSyncEnc (  Page 225 MMC_EM340LinkDevInfoSyncEnc)	Synchronous encoder axis start request

## MMC\_EM340LinkDevCHGWordInfoSync

Label name	Data type	Name
Type	unsigned short	RW: [Pr.710] Synchronous encoder axis: Link device type
FirstNo	unsigned short	RW: [Pr.711] Synchronous encoder axis: Link device start No.
CntMax	unsigned long	RW: [Pr.713] Synchronous encoder axis: Ring counter maximum value
CntMin	unsigned long	RW: [Pr.714] Synchronous encoder axis: Ring counter minimum value
CntDirection	unsigned short	RW: [Pr.712] Synchronous encoder axis: Link device count direction setting







## MMC\_EM340LinkDevInfoSyncEnc

Label name	Data type	Name
Type	unsigned short	RW: [Pr.1010] Synchronous encoder axis start request: Link device type
FirstNo	unsigned short	RW: [Pr.1011] Synchronous encoder axis start request: Link device start No.
BitSpecification	unsigned short	RW: [Pr.1012] Synchronous encoder axis start request: Link device bit specification
LogicSetting	unsigned short	RW: [Pr.1013] Synchronous encoder axis start request: Link device logic setting

## MMC\_EM340SysMntr3

Label name	Data type	Name
VartualSv	unsigned short	R: [Md.700] Virtual servo amplifier connected station monitor
MaxAxesSettingValue	unsigned short	R: [Md.180] Maximum number of control axes setting value

## MMC\_EM340IEFieldInfo

Label name	Data type	Name
bRX	bool[16384]	R: RX area
bRY	bool[16384]	RW: RY area
wRX	unsigned short[1024]	R: RX area (for word access)
wRY	unsigned short[1024]	RW: RY area (for word access)
dwRX	unsigned long[512]	R: RX area (for double word access)
dwRY	unsigned long[512]	RW: RY area (for double word access)
wRWw	unsigned short[8192]	RW: RWw area
wRWr	unsigned short[8192]	R: RWr area
dwRWw	unsigned long[4096]	RW: RWw area (for double word access)
dwRWr	unsigned long[4096]	R: RWr area (for double word access)
SB	bool[512]	RW: SB area
SW	unsigned short[512]	RW: SW area
RXArealInfo	MMC_EM340OffsetSizeRX[121] (  Page 226 MMC_EM340OffsetSizeRX)	RX offset/size information
RYArealInfo	MMC_EM340OffsetSizeRY[121] (  Page 226 MMC_EM340OffsetSizeRY)	RY offset/size information
RWwArealInfo	MMC_EM340OffsetSizeRWw[121] (  Page 226 MMC_EM340OffsetSizeRWw)	RWw offset/size information
RWrArealInfo	MMC_EM340OffsetSizeRWr[121] (  Page 226 MMC_EM340OffsetSizeRWr)	RWr offset/size information
OwnInfo	MMC_EM340StationInfoOwn (  Page 227 MMC_EM340StationInfoOwn)	Own station information
SystemInfo	MMC_EM340StationInfo[121] (  Page 227 MMC_EM340StationInfo)	Other station information
Completion_CommunicationPath_network	bool[239]	R: Communication path defined state
Val_Port2Line_ErrorOccurrenceRate_Max	unsigned short[121]	R: PORT2 line error occurrence rate (max.)
Val_Port2Line_ErrorOccurrenceRate_Present	unsigned short[121]	R: PORT2 line error occurrence rate (present)

## MMC\_EM340OffsetSizeRX

Label name	Data type	Name
Val_Offset	unsigned short	R: RX offset
Val_Size	unsigned short	R: RX size

## MMC\_EM340OffsetSizeRY

Label name	Data type	Name
Val_Offset	unsigned short	R: RY offset
Val_Size	unsigned short	R: RY size

## MMC\_EM340OffsetSizeRWw

Label name	Data type	Name
Val_Offset	unsigned short	R: RWw offset
Val_Size	unsigned short	R: RWw size

## MMC\_EM340OffsetSizeRWr

Label name	Data type	Name
Val_Offset	unsigned short	R: RWr offset
Val_Size	unsigned short	R: RWr size

## MMC\_EM340StationInfoOwn

Label name	Data type	Name
Val_CardInfo	MMC_EM340CardInfoOwn ( <a href="#">Page 227 MMC_EM340CardInfoOwn</a> )	Network card information
Val_ControllerInfo	MMC_EM340ControllerInfoOwn ( <a href="#">Page 227 MMC_EM340ControllerInfoOwn</a> )	Controller information

## MMC\_EM340CardInfoOwn

Label name	Data type	Name
Val_ManufacturerCode	unsigned short	R: Manufacturer code (Own station)
Val_ModelType	unsigned short	R: Model type (Own station)
Val_ModelCode	unsigned short	R: Model code (Own station)
Val_Version	unsigned short	R: Version (Own station)

## MMC\_EM340ControllerInfoOwn

Label name	Data type	Name
Val_Flag	unsigned short	R: Controller valid/invalid flag (Own station)
Val_ManufacturerCode	unsigned short	R: Manufacturer code (Own station)
Val_ModelType	unsigned short	R: Model type (Own station)
Val_ModelCode	unsigned short	R: Model code (Own station)
Val_Version	unsigned short	R: Version (Own station)
Val_ModelName	unsigned short[10]	R: Model name string (Own station)
Val_SpecificInformation	unsigned short[2]	R: Vendor-specific device information (Own station)

## MMC\_EM340StationInfo

Label name	Data type	Name
Val_CardInfo	MMC_EM340CardInfo ( <a href="#">Page 227 MMC_EM340CardInfo</a> )	Network card information
Val_ControllerInfo	MMC_EM340ControllerInfo ( <a href="#">Page 227 MMC_EM340ControllerInfo</a> )	Controller information

## MMC\_EM340CardInfo

Label name	Data type	Name
Val_ManufacturerCode	unsigned short	R: Manufacturer code
Val_ModelType	unsigned short	R: Model type
Val_ModelCode	unsigned short	R: Model code
Val_Version	unsigned short	R: Version

## MMC\_EM340ControllerInfo

Label name	Data type	Name
Val_Flag	unsigned short	R: Controller valid/invalid flag
Val_ManufacturerCode	unsigned short	R: Manufacturer code
Val_ModelType	unsigned short	R: Model type
Val_ModelCode	unsigned short	R: Model code
Val_Version	unsigned short	R: Version
Val_ModelName	unsigned short[10]	R: Model name string
Val_SpecificInformation	unsigned short[2]	R: Vendor-specific device information

## MMC\_EM340LinkDevCHGBit

Label name	Data type	Name
EMI	MMC_EM340LinkDevInfoEMI ( Page 228 MMC_EM340LinkDevInfoEMI)	Forced stop signal (EMI)
FLS	MMC_EM340LinkDevInfoFLS ( Page 229 MMC_EM340LinkDevInfoFLS)	Upper limit signal (FLS)
RLS	MMC_EM340LinkDevInfoRLS ( Page 229 MMC_EM340LinkDevInfoRLS)	Lower limit signal (RLS)
DOG	MMC_EM340LinkDevInfoDOG ( Page 229 MMC_EM340LinkDevInfoDOG)	Proximity dog signal (DOG)
STOP	MMC_EM340LinkDevInfoSTOP ( Page 229 MMC_EM340LinkDevInfoSTOP)	Stop signal (STOP)
PositStartReq	MMC_EM340LinkDevInfoPositStart ( Page 229 MMC_EM340LinkDevInfoPositStart)	External positioning start request
ChangeSpeedReq	MMC_EM340LinkDevInfoChangeSpeed ( Page 230 MMC_EM340LinkDevInfoChangeSpeed)	External speed change request
SkipReq	MMC_EM340LinkDevInfoSkip ( Page 230 MMC_EM340LinkDevInfoSkip)	Skip request
VP_Req	MMC_EM340LinkDevInfoVP ( Page 230 MMC_EM340LinkDevInfoVP)	Speed-position control switching request
MainShaftClutchControlReq	MMC_EM340LinkDevInfoMainShaft ( Page 230 MMC_EM340LinkDevInfoMainShaft)	Main shaft clutch control request
AuxShaftClutchControlReq	MMC_EM340LinkDevInfoAuxShaft ( Page 230 MMC_EM340LinkDevInfoAuxShaft)	Auxiliary shaft clutch control request
Block7000StartReq	MMC_EM340LinkDevInfoBlock7000 ( Page 231 MMC_EM340LinkDevInfoBlock7000)	Block No.7000 start request
Block7001StartReq	MMC_EM340LinkDevInfoBlock7001 ( Page 231 MMC_EM340LinkDevInfoBlock7001)	Block No.7001 start request
Block7002StartReq	MMC_EM340LinkDevInfoBlock7002 ( Page 231 MMC_EM340LinkDevInfoBlock7002)	Block No.7002 start request
Block7003StartReq	MMC_EM340LinkDevInfoBlock7003 ( Page 231 MMC_EM340LinkDevInfoBlock7003)	Block No.7003 start request
Block7004StartReq	MMC_EM340LinkDevInfoBlock7004 ( Page 231 MMC_EM340LinkDevInfoBlock7004)	Block No.7004 start request

## MMC\_EM340LinkDevInfoEMI

Label name	Data type	Name
Type	unsigned short	RW: [Pr.900] Forced stop signal (EMI): Link device type
FirstNo	unsigned short	RW: [Pr.901] Forced stop signal (EMI): Link device start No.
BitSpecification	unsigned short	RW: [Pr.902] Forced stop signal (EMI): Link device bit specification
LogicSetting	unsigned short	RW: [Pr.903] Forced stop signal (EMI): Link device logic setting

## MMC\_EM340LinkDevInfoFLS

Label name	Data type	Name
Type	unsigned short	RW: [Pr.910] Upper limit signal (FLS): Link device type
FirstNo	unsigned short	RW: [Pr.911] Upper limit signal (FLS): Link device start No.
BitSpecification	unsigned short	RW: [Pr.912] Upper limit signal (FLS): Link device bit specification
LogicSetting	unsigned short	RW: [Pr.913] Upper limit signal (FLS): Link device logic setting

## MMC\_EM340LinkDevInfoRLS

Label name	Data type	Name
Type	unsigned short	RW: [Pr.920] Lower limit signal (RLS): Link device type
FirstNo	unsigned short	RW: [Pr.921] Lower limit signal (RLS): Link device start No.
BitSpecification	unsigned short	RW: [Pr.922] Lower limit signal (RLS): Link device bit specification
LogicSetting	unsigned short	RW: [Pr.923] Lower limit signal (RLS): Link device logic setting

## MMC\_EM340LinkDevInfoDOG

Label name	Data type	Name
Type	unsigned short	RW: [Pr.930] Proximity dog signal (DOG): Link device type
FirstNo	unsigned short	RW: [Pr.931] Proximity dog signal (DOG): Link device start No.
BitSpecification	unsigned short	RW: [Pr.932] Proximity dog signal (DOG): Link device bit specification
LogicSetting	unsigned short	RW: [Pr.933] Proximity dog signal (DOG): Link device logic setting

## MMC\_EM340LinkDevInfoSTOP

Label name	Data type	Name
Type	unsigned short	RW: [Pr.940] Stop signal (STOP): Link device type
FirstNo	unsigned short	RW: [Pr.941] Stop signal (STOP): Link device start No.
BitSpecification	unsigned short	RW: [Pr.942] Stop signal (STOP): Link device bit specification
LogicSetting	unsigned short	RW: [Pr.943] Stop signal (STOP): Link device logic setting

## MMC\_EM340LinkDevInfoPositStart

Label name	Data type	Name
Type	unsigned short	RW: [Pr.950] External positioning start request: Link device type
FirstNo	unsigned short	RW: [Pr.951] External positioning start request: Link device start No.
BitSpecification	unsigned short	RW: [Pr.952] External positioning start request: Link device bit specification
LogicSetting	unsigned short	RW: [Pr.953] External positioning start request: Link device logic setting

## MMC\_EM340LinkDevInfoChangeSpeed

Label name	Data type	Name
Type	unsigned short	RW: [Pr.960] External speed change request: Link device type
FirstNo	unsigned short	RW: [Pr.961] External speed change request: Link device start No.
BitSpecification	unsigned short	RW: [Pr.962] External speed change request: Link device bit specification
LogicSetting	unsigned short	RW: [Pr.963] External speed change request: Link device logic setting

## MMC\_EM340LinkDevInfoSkip

Label name	Data type	Name
Type	unsigned short	RW: [Pr.970] Skip request: Link device type
FirstNo	unsigned short	RW: [Pr.971] Skip request: Link device start No.
BitSpecification	unsigned short	RW: [Pr.972] Skip request: Link device bit specification
LogicSetting	unsigned short	RW: [Pr.973] Skip request: Link device logic setting

## MMC\_EM340LinkDevInfoVP

Label name	Data type	Name
Type	unsigned short	RW: [Pr.980] Speed-position control switching request: Link device type
FirstNo	unsigned short	RW: [Pr.981] Speed-position control switching request: Link device start No.
BitSpecification	unsigned short	RW: [Pr.982] Speed-position control switching request: Link device bit specification
LogicSetting	unsigned short	RW: [Pr.983] Speed-position control switching request: Link device logic setting

## MMC\_EM340LinkDevInfoMainShaft

Label name	Data type	Name
Type	unsigned short	RW: [Pr.990] Main shaft clutch control request: Link device type
FirstNo	unsigned short	RW: [Pr.991] Main shaft clutch control request: Link device start No.
BitSpecification	unsigned short	RW: [Pr.992] Main shaft clutch control request: Link device bit specification
LogicSetting	unsigned short	RW: [Pr.993] Main shaft clutch control request: Link device logic setting

## MMC\_EM340LinkDevInfoAuxShaft

Label name	Data type	Name
Type	unsigned short	RW: [Pr.1000] Auxiliary shaft clutch control request: Link device type
FirstNo	unsigned short	RW: [Pr.1001] Auxiliary shaft clutch control request: Link device start No.
BitSpecification	unsigned short	RW: [Pr.1002] Auxiliary shaft clutch control request: Link device bit specification
LogicSetting	unsigned short	RW: [Pr.1003] Auxiliary shaft clutch control request: Link device logic setting



## MMC\_EM340LinkDevInfoBlock7000

Label name	Data type	Name
Type	unsigned short	RW: [Pr.1020] Block No.7000 start request: Link device type
FirstNo	unsigned short	RW: [Pr.1021] Block No.7000 start request: Link device start No.
BitSpecification	unsigned short	RW: [Pr.1022] Block No.7000 start request: Link device bit specification
LogicSetting	unsigned short	RW: [Pr.1023] Block No.7000 start request: Link device logic setting

## MMC\_EM340LinkDevInfoBlock7001

Label name	Data type	Name
Type	unsigned short	RW: [Pr.1030] Block No.7001 start request: Link device type
FirstNo	unsigned short	RW: [Pr.1031] Block No.7001 start request: Link device start No.
BitSpecification	unsigned short	RW: [Pr.1032] Block No.7001 start request: Link device bit specification
LogicSetting	unsigned short	RW: [Pr.1033] Block No.7001 start request: Link device logic setting

## MMC\_EM340LinkDevInfoBlock7002

Label name	Data type	Name
Type	unsigned short	RW: [Pr.1040] Block No.7002 start request: Link device type
FirstNo	unsigned short	RW: [Pr.1041] Block No.7002 start request: Link device start No.
BitSpecification	unsigned short	RW: [Pr.1042] Block No.7002 start request: Link device bit specification
LogicSetting	unsigned short	RW: [Pr.1043] Block No.7002 start request: Link device logic setting

## MMC\_EM340LinkDevInfoBlock7003

Label name	Data type	Name
Type	unsigned short	RW: [Pr.1050] Block No.7003 start request: Link device type
FirstNo	unsigned short	RW: [Pr.1051] Block No.7003 start request: Link device start No.
BitSpecification	unsigned short	RW: [Pr.1052] Block No.7003 start request: Link device bit specification
LogicSetting	unsigned short	RW: [Pr.1053] Block No.7003 start request: Link device logic setting

## MMC\_EM340LinkDevInfoBlock7004

Label name	Data type	Name
Type	unsigned short	RW: [Pr.1060] Block No.7004 start request: Link device type
FirstNo	unsigned short	RW: [Pr.1061] Block No.7004 start request: Link device start No.
BitSpecification	unsigned short	RW: [Pr.1062] Block No.7004 start request: Link device bit specification
LogicSetting	unsigned short	RW: [Pr.1063] Block No.7004 start request: Link device logic setting

## MMC\_EM340LinkDevCHGWord

Label name	Data type	Name
MPG	MMC_EM340LinkDevCHGWordInfoMPG (  Page 232 MMC_EM340LinkDevCHGWordInfoMPG)	Manual pulse generator input

## MMC\_EM340LinkDevCHGWordInfoMPG

Label name	Data type	Name
Type	unsigned short	RW: [Pr.700] Manual pulse generator input: Link device type
FirstNo	unsigned short	RW: [Pr.701] Manual pulse generator input: Link device start No.
CntMax	unsigned long	RW: [Pr.703] Manual pulse generator input: Ring counter maximum value
CntMin	unsigned long	RW: [Pr.704] Manual pulse generator input: Ring counter minimum value
CntDirection	unsigned short	RW: [Pr.702] Manual pulse generator input: Link device count direction setting

## MMC\_EM340ServoObject

Label name	Data type	Name
AnySendPDO1	unsigned long	RW: [Pr.500] Optional send PDO 1
AnySendPDO2	unsigned long	RW: [Pr.501] Optional send PDO 2
AnySendPDO3	unsigned long	RW: [Pr.502] Optional send PDO 3
AnySendPDO4	unsigned long	RW: [Pr.503] Optional send PDO 4
AnyRecvPDO1	unsigned long	RW: [Pr.506] Optional recv PDO 1
AnyRecvPDO2	unsigned long	RW: [Pr.507] Optional recv PDO 2
AnyRecvPDO3	unsigned long	RW: [Pr.508] Optional recv PDO 3
AnyRecvPDO4	unsigned long	RW: [Pr.509] Optional recv PDO 4
AnySDO1	unsigned long	RW: [Pr.512] Optional SDO 1
AnySDO2	unsigned long	RW: [Pr.513] Optional SDO 2
AnySDO3	unsigned long	RW: [Pr.514] Optional SDO 3
AnySDO4	unsigned long	RW: [Pr.515] Optional SDO 4






## MMC\_EM340SlaveMonitor

Label name	Data type	Name
AnySDOResult1	unsigned long	R: [Md.160] Optional SDO transfer result 1
AnySDOResult2	unsigned long	R: [Md.161] Optional SDO transfer result 2
AnySDOResult3	unsigned long	R: [Md.162] Optional SDO transfer result 3
AnySDOResult4	unsigned long	R: [Md.163] Optional SDO transfer result 4
AnySDOStatus1	unsigned short	R: [Md.164] Optional SDO transfer status 1
AnySDOStatus2	unsigned short	R: [Md.165] Optional SDO transfer status 2
AnySDOStatus3	unsigned short	R: [Md.166] Optional SDO transfer status 3
AnySDOStatus4	unsigned short	R: [Md.167] Optional SDO transfer status 4
AnyRecvPDOData1	unsigned long[2]	R: [Md.170] Optional receive PDO data 1
AnyRecvPDOData2	unsigned long[2]	R: [Md.171] Optional receive PDO data 2
AnyRecvPDOData3	unsigned long[2]	R: [Md.172] Optional receive PDO data 3
AnyRecvPDOData4	unsigned long[2]	R: [Md.173] Optional receive PDO data 4
RestorationStatus	unsigned short	R: [Md.190] Controller current value restoration complete status

## MMC\_EM340SlaveControl

Label name	Data type	Name
SvPrmReq	unsigned short	RW: [Cd.120] Servo parameter operation request
Size	unsigned short	RW: [Cd.121] Parameter size
Offset	unsigned short	RW: [Cd.122] Parameter offset
ReqData	unsigned short[1024]	RW: [Cd.125] Request data
RestorationReq	unsigned short	RW: [Cd.126] Controller current value restoration request
ABSLostDesignation	unsigned short	RW: [Cd.127] Absolute position disappearance designation
AnySendPDOData1	unsigned long[2]	RW: [Cd.170] Optional send PDO data 1
AnySendPDOData2	unsigned long[2]	RW: [Cd.171] Optional send PDO data 2
AnySendPDOData3	unsigned long[2]	RW: [Cd.172] Optional send PDO data 3
AnySendPDOData4	unsigned long[2]	RW: [Cd.173] Optional send PDO data 4
SendReq1	unsigned short	RW: [Cd.160] Command sending request 1
SendReq2	unsigned short	RW: [Cd.161] Command sending request 2
SendReq3	unsigned short	RW: [Cd.162] Command sending request 3
SendReq4	unsigned short	RW: [Cd.163] Command sending request 4
AnySDOSend1	unsigned short[63]	RW: [Cd.164] Optional SDO transfer data 1
AnySDOSend2	unsigned short[63]	RW: [Cd.165] Optional SDO transfer data 2
AnySDOSend3	unsigned short[63]	RW: [Cd.166] Optional SDO transfer data 3
AnySDOSend4	unsigned short[63]	RW: [Cd.167] Optional SDO transfer data 4

## MMC\_EM340ExCamData

Label name	Data type	Name
StrokeRatioData	MMC_EM340StrokeRatioDataEx[32768] (  Page 233 MMC_EM340StrokeRatioDataEx)	Stroke ratio data format (1 to 32768)
CoordinateData	MMC_EM340CoordinateDataEx[65535] (  Page 233 MMC_EM340CoordinateDataEx)	Coordinate data format (1 to 65535)
RotaryCutter	MMC_EM340RotaryCutter (  Page 234 MMC_EM340RotaryCutter)	Rotary cutter cam
SimpleStroke	MMC_EM340SimpleStroke (  Page 234 MMC_EM340SimpleStroke)	Simple stroke ratio cam
DetailsStroke	MMC_EM340DetailsStroke (  Page 234 MMC_EM340DetailsStroke)	Details stroke ratio cam

## MMC\_EM340StrokeRatioDataEx

Label name	Data type	Name
Value	long	RW: [Cd.607] Cam data value Stroke ratio


## MMC\_EM340CoordinateDataEx

Label name	Data type	Name
InputValue	unsigned long	RW: [Cd.607] Cam data value Input value
OutputValue	long	RW: [Cd.607] Cam data value Output value

## MMC\_EM340RotaryCutter

Label name	Data type	Name
Resolution	unsigned long	RW: [Cd.607] Cam data value Resolution
Option	unsigned short	RW: [Cd.607] Cam data value Auto-generation option
SyncSectionAccelerationRatio	short	RW: [Cd.607] Cam data value Synchronous section acceleration ratio
SheetLength	unsigned long	RW: [Cd.607] Cam data value Sheet length
SheetSyncWidth	unsigned long	RW: [Cd.607] Cam data value Sheet synchronization width
SyncAxisLength	unsigned long	RW: [Cd.607] Cam data value Synchronous axis length
SyncPointAdjustment	long	RW: [Cd.607] Cam data value Synchronous position adjustment
AccelerationWidth	unsigned long	RW: [Cd.607] Cam data value Acceleration/ deceleration width
CutterNum	unsigned short	RW: [Cd.607] Cam data value Number of cutter
AsyncSpeedResult	unsigned short	RW: [Cd.607] Cam data value Asynchronous speed result


## MMC\_EM340SimpleStroke

Label name	Data type	Name
Resolution	unsigned long	RW: [Cd.607] Cam data value Resolution
CamAxisLengthPerCycle	unsigned long	RW: [Cd.607] Cam data value Cam axis length per cycle
StartingPoint	unsigned long	RW: [Cd.607] Cam data value Cam data starting point
SheetLength	unsigned short	RW: [Cd.607] Cam data value Number of sections
SimpleSection	MMC_EM340SimpleSection[32] (  Page 234 MMC_EM340SimpleSection)	Easy stroke ratio cam data for auto-generation

## MMC\_EM340SimpleSection

Label name	Data type	Name
CurveType	unsigned short	RW: [Cd.607] Cam data value Cam curve type
EndPoint	unsigned long	RW: [Cd.607] Cam data value End point
Stroke	long	RW: [Cd.607] Cam data value Stroke




## MMC\_EM340DetailsStroke

Label name	Data type	Name
Resolution	unsigned long	RW: [Cd.607] Cam data value Resolution
CamAxisLengthPerCycle	unsigned long	RW: [Cd.607] Cam data value Cam axis length per cycle
CamStrokeAmount	unsigned long	RW: [Cd.607] Cam data value Cam stroke amount
Unit	unsigned short	RW: [Cd.607] Cam data value Unit setting
StartingPoint	unsigned long	RW: [Cd.607] Cam data value Cam data starting point
SectionNum	unsigned short	RW: [Cd.607] Cam data value Number of sections
DetailsSection	MMC_EM340DetailsSection[360] (  Page 235 MMC_EM340DetailsSection)	Advanced stroke ratio cam data for auto-generation

## MMC\_EM340DetailsSection

Label name	Data type	Name
CurveType	unsigned short	RW: [Cd.607] Cam data value Cam curve type
EndPoint	unsigned long	RW: [Cd.607] Cam data value End point
Stroke	long	RW: [Cd.607] Cam data value Stroke
CurveScopeP1	unsigned short	RW: [Cd.607] Cam data value Curve applicable range (P1)
CurveScopeP2	unsigned short	RW: [Cd.607] Cam data value Curve applicable range (P2)
AccelerationRangeCompensationL1	short	RW: [Cd.607] Cam data value Acceleration/ deceleration range compensation (Range L1)
AccelerationRangeCompensationL2	short	RW: [Cd.607] Cam data value Acceleration/ deceleration range compensation (Range L2)


## MMC\_EM340ExAutomaticCam

Label name	Data type	Name
Request	unsigned short	RW: [Cd.608] Cam auto-generation request
CamNo	unsigned short	RW: [Cd.609] Cam auto-generation cam No.
Type	unsigned short	RW: [Cd.610] Cam auto-generation type
RotaryCutter	MMC_EM340RotaryCutterAuto (  Page 235 MMC_EM340RotaryCutterAuto)	Rotary cutter cam
SimpleStroke	MMC_EM340SimpleStrokeAuto (  Page 236 MMC_EM340SimpleStrokeAuto)	Simple stroke ratio cam
DetailsStroke	MMC_EM340DetailsStrokeAuto (  Page 236 MMC_EM340DetailsStrokeAuto)	Details stroke ratio cam

## MMC\_EM340RotaryCutterAuto

Label name	Data type	Name
Resolution	unsigned long	RW: [Cd.611] Cam auto-generation data Resolution
Option	unsigned short	RW: [Cd.611] Cam auto-generation data Auto-generation option
SyncSectionAccelerationRatio	short	RW: [Cd.611] Cam auto-generation data Synchronous section acceleration ratio
SheetLength	unsigned long	RW: [Cd.611] Cam auto-generation data Sheet length
SheetSyncWidth	unsigned long	RW: [Cd.611] Cam auto-generation data Sheet synchronization width
SyncAxisLength	unsigned long	RW: [Cd.611] Cam auto-generation data Synchronous axis length
SyncPointAdjustment	long	RW: [Cd.611] Cam auto-generation data Synchronous position adjustment
AccelerationWidth	unsigned long	RW: [Cd.611] Cam auto-generation data Acceleration/ deceleration width
CutterNum	unsigned short	RW: [Cd.611] Cam auto-generation data Number of cutter
AsyncSpeedResult	unsigned short	RW: [Cd.611] Cam auto-generation data Asynchronous speed result


## MMC\_EM340SimpleStrokeAuto

Label name	Data type	Name
Resolution	unsigned long	RW: [Cd.611] Cam auto-generation data Resolution
CamAxisLengthPerCycle	unsigned long	RW: [Cd.611] Cam auto-generation data Cam axis length per cycle
StartingPoint	unsigned long	RW: [Cd.611] Cam auto-generation data Cam data starting point
SectionNum	unsigned short	RW: [Cd.611] Cam auto-generation data Number of sections
SimpleSection	MMC_EM340SimpleSectionAuto[32] (  Page 236 MMC_EM340SimpleSectionAuto)	Easy stroke ratio cam data for auto-generation

## MMC\_EM340SimpleSectionAuto

Label name	Data type	Name
CurveType	unsigned short	RW: [Cd.611] Cam auto-generation data Cam curve type
EndPoint	unsigned long	RW: [Cd.611] Cam auto-generation data End point
Stroke	long	RW: [Cd.611] Cam auto-generation data Stroke




## MMC\_EM340DetailsStrokeAuto

Label name	Data type	Name
Resolution	unsigned long	RW: [Cd.611] Cam auto-generation data Resolution
CamAxisLengthPerCycle	unsigned long	RW: [Cd.611] Cam auto-generation data Cam axis length per cycle
CamStrokeAmount	unsigned long	RW: [Cd.611] Cam auto-generation data Cam stroke amount
Unit	unsigned short	RW: [Cd.611] Cam auto-generation data Unit setting
StartingPoint	unsigned long	RW: [Cd.611] Cam auto-generation data Cam data starting poin
SectionNum	unsigned short	RW: [Cd.611] Cam auto-generation data Number of sections
DetailsSection	MMC_EM340DetailsSectionAuto[360] (  Page 236 MMC_EM340DetailsSectionAuto)	Advanced stroke ratio cam data for auto-generation

## MMC\_EM340DetailsSectionAuto

Label name	Data type	Name
CurveType	unsigned short	RW: [Cd.611] Cam auto-generation data Cam curve type
EndPoint	unsigned long	RW: [Cd.611] Cam auto-generation data End point
Stroke	unsigned long	RW: [Cd.611] Cam auto-generation data Stroke
CurveScopeP1	unsigned short	RW: [Cd.611] Cam auto-generation data Curve applicable range (P1)
CurveScopeP2	unsigned short	RW: [Cd.611] Cam auto-generation data Curve applicable range (P2)
AccelerationRangeCompensationL1	short	RW: [Cd.611] Cam auto-generation data Acceleration/ deceleration range compensation (Range L1)
AccelerationRangeCompensationL2	short	RW: [Cd.611] Cam auto-generation data Acceleration/ deceleration range compensation (Range L2)

## MMC\_EM340Ethernet

Label name	Data type	Name
EthernetPrm	MMC_EM340EthernetPrm (  Page 237 MMC_EM340EthernetPrm)	Ethernet communication parameter
EthernetCmd	MMC_EM340EthernetCmd (  Page 237 MMC_EM340EthernetCmd)	Ethernet communication command
EthernetMntr	MMC_EM340EthernetMntr (  Page 237 MMC_EM340EthernetMntr)	Ethernet communication monitor

## MMC\_EM340EthernetPrm

Label name	Data type	Name
IP_Address	unsigned long	RW: [Pr.1170] IP address
SubnetMask	unsigned long	RW: [Pr.1171] Subnet mask
DefaultGateway	unsigned long	RW: [Pr.1172] Default gateway
DirectAccessInValidate	unsigned short	RW: [Pr.1173] Disable direct connection with MELSOFT

## MMC\_EM340EthernetCmd

Label name	Data type	Name
BackupRequest	unsigned short	RW: [Cd.1170] Ethernet communication parameter backup request

## MMC\_EM340EthernetMntr

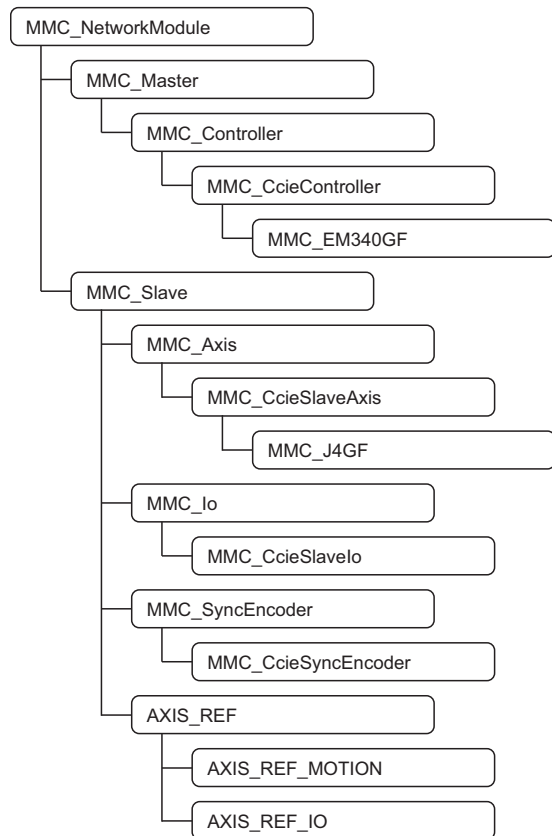
Label name	Data type	Name
IP_Address	unsigned long	R: [Md.1170] IP address
SubnetMask	unsigned long	R: [Md.1171] Subnet mask
DefaultGateway	unsigned long	R: [Md.1172] Default gateway
MAC_Address	unsigned long[2]	R: [Md.1173] MAC address
LinkStatus	unsigned short	R: [Md.1174] Link status

# APPENDICES

## Appendix 1 Hierarchy Charts for Each Class

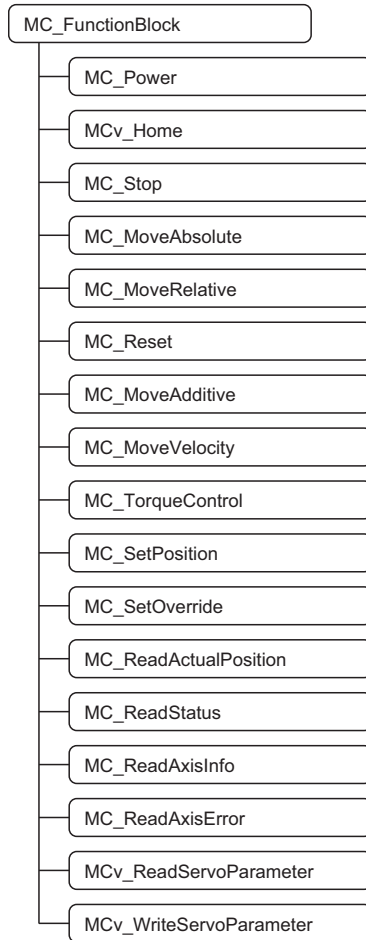
The hierarchy charts for the classes used by the API library are shown below.

### Network module class

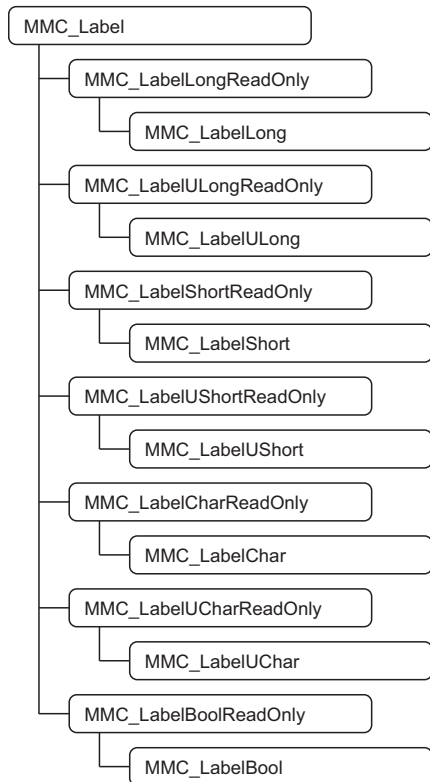




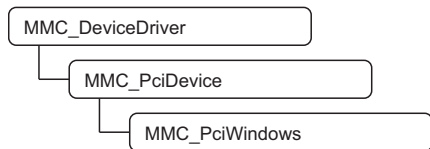
## Function black class



## Label class



## Device driver class



## Appendix 2 Restrictions by the version

There is a restriction in the function that can be used by the API version.

Function/Item	Software version	API version	Description
Function block	—	Ver.1.10	MR-J4-GF I/O mode
Direct control	Ver.02	Ver.1.10	Direct control is supported.
SLMP communication function	Ver.03	Ver.1.30	SLMP communication function is supported.
C#	—	Ver.1.30	Microsoft Visual C# is supported.

# INDEX

---

## A

---

AXIS_REF Class	192
AXIS_REF_IO Class	194
AXIS_REF_MOTION Class	193

## B

---

BackupParameter	41
Basic library functions	27

## C

---

CalcCamCommandPosition	129
CalcCamCommandPositionPerCycle	130
ChangeControlMode	78
ChangePosition	80
ChangeSpeed	79
ChangeSyncEncoderPosition	141
ChangeSyncPosition	137
Close	95

## D

---

Delete	31,93
DisableDMA	54
DisableInterrupt	45
DisableMPG	77
DisableSyncEncoder	142

## E

---

EnableDMA	53
EnableInterrupt	44
EnableMPG	76
EnableSyncEncoder	143
EndDMA	99
EndInterrupt	97

## G

---

GetAxis (C#)	35
GetAxis (C++)	34
GetBlockConditionData	64
GetBlockStartData	63
GetBufferMemory	101
GetFbAxisRef	190
GetFbAxisReflo	191
GetPositioningData	62
GetSlavelo (C#)	37
GetSlavelo (C++)	36
GetSyncEncoder	128

## I

---

InitializeParameter	40
---------------------	----

## L

---

Label	203
-------	-----

## M

---

MakeAdvancedStrokeRatioCam	133
MakeEasyStrokeRatioCam	132
MakeRotaryCutterCam	131
MC_FunctionBlock class	24,144
MC_MoveAbsolute class	159
MC_MoveAdditive class	167
MC_MoveRelative class	162
MC_MoveVelocity class	169
MC_Power class	153
MC_ReadActualPosition class	177
MC_ReadAxisError class	183
MC_ReadAxisInfo class	181
MC_ReadStatus class	179
MC_Reset class	165
MC_SetOverride class	175
MC_SetPosition class	173
MC_Stop class	157
MC_TorqueControl class	171
MCv_Home class	155
MCv_ReadServoParameter class	185
MCv_WriteServoParameter class	187
MMC_Axis class	22,57,134
MMC_Controller class	20,32,127,189
MMC_DeviceDriver class	26,92
MMC_Io class	23,87
MMC_Label class	26,89
MMC_NetworkModule class	19,30
MMC_SyncEncoder class	24,139
MmfCheckPassTime	124
MmfCreateEM340GF (C#)	103
MmfCreateEM340GF (C++)	102
MmfCreateEvent	111
MmfCreateFunctionBlock	195
MmfCreatePciDevice (C#)	105
MmfCreatePciDevice (C++)	104
MmfCreateSemaphore	107
MmfCreateThread	116
MmfDeleteEvent	112
MmfDeleteSemaphore	108
MmfDeleteThread	117
MmfExitThread	122
MmfGetCurrentTime	123
MmfGetExitCodeThread	121
MmfGetLastError	106
MmfReleaseSemaphore	110
MmfResetEvent	113
MmfResumeThread	120
MmfSetEvent	114
MmfSetThreadPriority	119
MmfWaitDelayTime	125
MmfWaitEvent	115
MmfWaitSemaphore	109
MmfWaitThread	118
MMST_AdvancedStrokeRatioCamData	198
MMST_AdvancedStrokeRatioCamSectionData	198
MMST_BlockConditionData	196
MMST_BlockStartData	196
MMST_CamPositionData	197
MMST_EasyStrokeRatioCamData	198
MMST_EasyStrokeRatioCamSectionData	198

MMST_InterruptData . . . . .	197
MMST_InterruptParameter . . . . .	197
MMST_PositioningData . . . . .	196
MMST_RotaryCutterCamData . . . . .	197
MoveCamPosition . . . . .	138

## O

---

Object generation functions . . . . .	27,102
Open . . . . .	94

## R

---

ReadRemoteBufferMemory . . . . .	55
RegisterIntCallback (C#) . . . . .	47
RegisterIntCallback (C++) . . . . .	46
ResetController . . . . .	38
ResetError . . . . .	81
ResetIntEvent . . . . .	50,84
ResetPositioningDoneIntEvent . . . . .	70
ResetSyncEncoderError . . . . .	140
RestartPositioning . . . . .	68

## S

---

SetBlockConditionData . . . . .	61
SetBlockStartData . . . . .	60
SetBufferMemory . . . . .	100
SetInterruptParameter . . . . .	42,82
SetIntEvent . . . . .	51,85
SetPositioningData . . . . .	59
SetPositioningDoneIntEvent . . . . .	71
SetUserProgramReady . . . . .	39
StartBlockPositioning . . . . .	66
StartDMA . . . . .	98
StartInching . . . . .	75
StartInterrupt . . . . .	96
StartJog . . . . .	73
StartPositioning . . . . .	65
StartSync . . . . .	135
StopJog . . . . .	74
StopPositioning . . . . .	67
StopSync . . . . .	136

## U

---

UnregisterIntCallback . . . . .	49
Update . . . . .	152

## W

---

Wait . . . . .	90
WaitIntEvent . . . . .	52,86
WaitPositioningDone . . . . .	69
WaitPositioningDoneIntEvent . . . . .	72
WriteRemoteBufferMemory . . . . .	56

# REVISIONS

\* The manual number is given on the bottom left of the back cover

Revision date	*Manual number	Description
August 2016	IB(NA)-0300330-A	First edition
June 2017	IB(NA)-0300330-B	■ Added functions MR-J4-GF I/O mode, Direct control ■ Added or modified parts SAFETY PRECAUTIONS, TERMS, Section 1.1, 2.2, 3.2, 3.4, 4.1, Chapter 5, Section 5.1, 5.2, 5.3, 5.4, 5.5, 5.6, 7.1, 7.2, 7.3, 7.4, 7.5, Chapter 8, Appendix 1, 2
February 2018	IB(NA)-0300330-C	■ Added functions Microsoft® Visual C#®, SLMP communication function ■ Added or modified parts SAFETY PRECAUTIONS, TERMS, Section 1.1, 1.3, 2.2, 3.2, 3.7, 3.8, 7.5, Appendix 2
January 2024	IB(NA)-0300330-D	■ Added or modified parts INFORMATION AND SERVICES, TRADEMARKS

Japanese manual number: IB-0300329-D

This manual confers no industrial property rights of any other kind, nor does it confer any patent licenses. Mitsubishi Electric Corporation cannot be held responsible for any problems involving industrial property rights which may occur as a result of using the contents noted in this manual.

©2016 MITSUBISHI ELECTRIC CORPORATION

# WARRANTY

---

Please confirm the following product warranty details before using this product.

## **1. Gratis Warranty Term and Gratis Warranty Range**

If any faults or defects (hereinafter "Failure") found to be the responsibility of Mitsubishi occurs during use of the product within the gratis warranty term, the product shall be repaired at no cost via the sales representative or Mitsubishi Service Company.

However, if repairs are required onsite at domestic or overseas location, expenses to send an engineer will be solely at the customer's discretion. Mitsubishi shall not be held responsible for any re-commissioning, maintenance, or testing on-site that involves replacement of the failed module.

[Gratis Warranty Term]

The gratis warranty term of the product shall be for one year after the date of purchase or delivery to a designated place. Note that after manufacture and shipment from Mitsubishi, the maximum distribution period shall be six (6) months, and the longest gratis warranty term after manufacturing shall be eighteen (18) months. The gratis warranty term of repair parts shall not exceed the gratis warranty term before repairs.

[Gratis Warranty Range]

- (1) The range shall be limited to normal use within the usage state, usage methods and usage environment, etc., which follow the conditions and precautions, etc., given in the instruction manual, user's manual and caution labels on the product.
- (2) Even within the gratis warranty term, repairs shall be charged for in the following cases.
  1. Failure occurring from inappropriate storage or handling, carelessness or negligence by the user. Failure caused by the user's hardware or software design.
  2. Failure caused by unapproved modifications, etc., to the product by the user.
  3. When the Mitsubishi product is assembled into a user's device, Failure that could have been avoided if functions or structures, judged as necessary in the legal safety measures the user's device is subject to or as necessary by industry standards, had been provided.
  4. Failure that could have been avoided if consumable parts (battery, backlight, fuse, etc.) designated in the instruction manual had been correctly serviced or replaced.
  5. Failure caused by external irresistible forces such as fires or abnormal voltages, and Failure caused by force majeure such as earthquakes, lightning, wind and water damage.
  6. Failure caused by reasons unpredictable by scientific technology standards at time of shipment from Mitsubishi.
  7. Any other failure found not to be the responsibility of Mitsubishi or that admitted not to be so by the user.

## **2. Onerous repair term after discontinuation of production**

- (1) Mitsubishi shall accept onerous product repairs for seven (7) years after production of the product is discontinued. Discontinuation of production shall be notified with Mitsubishi Technical Bulletins, etc.
- (2) Product supply (including repair parts) is not available after production is discontinued.

## **3. Overseas service**

Overseas, repairs shall be accepted by Mitsubishi's local overseas FA Center. Note that the repair conditions at each FA Center may differ.

## **4. Exclusion of loss in opportunity and secondary loss from warranty liability**

Regardless of the gratis warranty term, Mitsubishi shall not be liable for compensation to:

- (1) Damages caused by any cause found not to be the responsibility of Mitsubishi.
- (2) Loss in opportunity, lost profits incurred to the user by Failures of Mitsubishi products.
- (3) Special damages and secondary damages whether foreseeable or not, compensation for accidents, and compensation for damages to products other than Mitsubishi products.
- (4) Replacement by the user, maintenance of on-site equipment, start-up test run and other tasks.

## **5. Changes in product specifications**

The specifications given in the catalogs, manuals or technical documents are subject to change without prior notice.

# INFORMATION AND SERVICES

---

For further information and services, please contact your local Mitsubishi Electric sales office or representative.  
Visit our website to find our locations worldwide.

MITSUBISHI ELECTRIC Factory Automation Global Website

Locations Worldwide

[www.MitsubishiElectric.com/fa/about-us/overseas/](http://www.MitsubishiElectric.com/fa/about-us/overseas/)

## TRADEMARKS

---

Ethernet is a registered trademark of Fuji Xerox Co., Ltd. in Japan.

Intel Core is either a trademark or a registered trademark of Intel Corporation and its subsidiaries in the United States and/or other countries.

Microsoft, Windows, Visual C++, Visual C#, and Visual Studio are trademarks of the Microsoft group of companies.

PCI Express is either a registered trademark or a trademark of PCI-SIG.

WinDriver is the trademark of Jungo Connectivity Ltd. in Israel.

PLCopen and related logos are registered trademarks of PLCopen®.

The company names, system names and product names mentioned in this manual are either registered trademarks or trademarks of their respective companies.

In some cases, trademark symbols such as <sup>™</sup> or <sup>®</sup> are not specified in this manual.





IB(NA)-0300330-D(2401)MEE

MODEL: EM340-U-API-E

MODEL CODE: 1XB054

## **MITSUBISHI ELECTRIC CORPORATION**

HEAD OFFICE: TOKYO BLDG., 2-7-3, MARUNOUCHI, CHIYODA-KU, TOKYO 100-8310, JAPAN  
NAGOYA WORKS: 1-14, YADA-MINAMI 5-CHOME, HIGASHI-KU, NAGOYA 461-8670, JAPAN

When exported from Japan, this manual does not require application to the  
Ministry of Economy, Trade and Industry for service transaction permission.

Specifications subject to change without notice.