Panasonic

Motion Controller

GM1 Series

Reference Manual

Instruction Edition

WUME-GM1PGR-05

2022.6

(MEMO)

Introduction

Thank you for purchasing a Panasonic product. Before you use the product, please carefully read through the installation instructions and the manuals, and understand them in detail to use the product properly.

Types of Manual

- There are different types of manuals for the GM1 series, as listed below. Please refer to a relevant manual for the unit and purpose of your use.
- The manuals can be downloaded from our website: https://industrial.panasonic.com/ac/e/motor/motion-controller/mc/gm1/index.jsp.

Manuals for GM1 series

Manual name	Manual code	Description
GM1 Controller RTEX User's Manual (Setup Edition)	WUME-GM1RTXSU	Explains wiring between the GM1 and its peripheral devices, installation method, and operation check method.
GM1 Controller EtherCAT User's Manual (Setup Edition)	WUME-GM1ETCSU	
GM1 Controller RTEX User's Manual (Operation Edition)	WUME-GM1RTXOP	Explains how to use GM Programmer and PANATERM Lite for GM, set up each function, create projects, and perform other operations.
GM1 Controller EtherCAT User's Manual (Operation Edition)	WUME-GM1ETCOP	
GM1 Series Reference Manual (Hardware Edition)	WUME-GM1H	Explains the functions and performance of each GM1 unit.
GM1 Series Reference Manual (Instruction Edition)	WUME-GM1PGR	Explains the specifications of each instruction that can be used with the GM1 Series.
GM1 Series Reference Manual (Analog I/O Unit)	WUME-GM1AIO	Explains the functions and performance of the GM1 Analog Expansion Unit.
GM1 Series Reference Manual (Pulse Output Unit)	WUME-GM1PG	Explains the functions and performance of the GM1 Pulse Output Unit.

Copyright / Trademarks

(MEMO)

Table of Contents

1 List of Instructions

1.1 List of Ladder Instructions

The following table lists contact and coil ladder instructions that can be used in ladder diagram programs for GM Programmer.

Name	Code	Description	Simulation (●: Supported, -: Not supported)	Page
NO contact		This instruction outputs a BOOL-type input from the left to the right. If the variable of the contact is TRUE, then the input value from the left is output. If the variable of the contact is FALSE, then FALSE is output.	●	"P.2-2"
NC contact		This instruction outputs the negated value of the BOOL-type input from the left to the right. If the variable of the contact is TRUE, then FALSE is output. If the variable of the contact is FALSE, then the input value from the left is output.	●	"P.2-3"
Rising edge detection		When a rising edge is detected in the BOOL-type input from the left, TRUE is output for one cycle only.	●	"P.2-3"
Falling edge detection		When a falling edge is detected in the BOOL-type input from the left, TRUE is output for one cycle only	●	"P.2-4"
Parallel NO contact	-	NO contacts can be wired in parallel. The contacts wired in parallel are treated as OR logic. If the output of one or more contacts is TRUE, TRUE is output.	●	"P.2-5"
Parallel NC contact	-	NC contacts can be wired in parallel. The contacts wired in parallel are treated as OR logic. If the output of one or more contacts is TRUE, TRUE is output.	●	"P.2-6"
Coil		A BOOL-type input from the left can be saved. If the input is TRUE, then TRUE is saved. If the input is FALSE, then FALSE is saved.	●	"P.2-7"
Negated coil		The negated value of the BOOL-type input from the left can be saved. If the input is TRUE, then FALSE is saved. If the input is FALSE, then TRUE is saved.	●	"P.2-7"
Set coil		If the BOOL-type input from the left becomes TRUE, TRUE is saved. It can be used together with the reset coil.	●	"P.2-8"
Reset coil		If the BOOL-type input from the left becomes TRUE, FALSE is saved. It can be used together with the set coil.	●	"P.2-9"
Execute Box		ST language programming is possible. If "Enter ST code here ..." is clicked, an input field using a multi-line ST will open.	●	"P.2-10"

1.2 List of Function Instructions

This section provides lists of the functions used by the GM Programmer. These functions can be used without declaring them with variables.

■ Basic instructions

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
MOVE	Substitution	Substitutes the input argument values with the output argument.	●	●	●	"P.3-4"
SIZEOF	Get the size	Outputs the size (in units of byte) of the input argument.	●	●	●	"P.3-4"
ADR	Get the address	Outputs the address of the input argument.	●	●	●	"P.3-5"

■ Arithmetic operation instructions

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
ADD	Addition	Adds the input arguments.	●	●	●	"P.3-7"
SUB	Subtraction	Subtracts the input arguments.	●	●	●	"P.3-8"
MUL	Multiplication	Multiplies the input arguments.	●	●	●	"P.3-9"
DIV	Division	Divides the input arguments.	●	●	●	"P.3-10"
MOD	Mod	Outputs the remainder of the input argument.	●	●	●	"P.3-11"

■ Boolean operation instructions

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
AND	Logical AND	Outputs the logical AND of the input arguments.	●	●	●	"P.3-12"
OR	Logical OR	Outputs the logical OR of the input arguments.	●	●	●	"P.3-13"
XOR	Exclusive OR	Outputs the Exclusive OR of the input arguments.	●	●	●	"P.3-14"
NOT	Negation	Outputs the negation of the input argument.	●	●	●	"P.3-13"

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
AND_THEN	Logical AND	Outputs the logical AND of the input arguments.	●	●	●	"P.3-15"
OR_ELSE	Logical OR	Outputs the logical OR of the input arguments.	●	●	●	"P.3-16"

■ Comparison operation instructions

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
EQ	“Equal” comparison	Compares the two input arguments and, if they are equal to each other, outputs TRUE.	●	●	●	"P.3-18"
NE	“Not Equal” comparison	Compares the two input arguments and, if they are not equal to each other, outputs TRUE.	●	●	●	"P.3-18"
LT	“Less Than” comparison	Compares the two input arguments and, if the first argument is less than the second argument, outputs TRUE.	●	●	●	"P.3-19"
LE	“Less Than or Equal” comparison	Compares the two input arguments and, if the first argument is less than the second argument or equal, outputs TRUE.	●	●	●	"P.3-20"
GT	“Greater Than” comparison	Compares the two input arguments and, if the first argument is greater than the second argument, outputs TRUE.	●	●	●	"P.3-21"
GE	“Greater Than Or Equal” comparison	Compares the two input arguments and, if the first argument is greater than the second argument or equal, outputs TRUE.	●	●	●	"P.3-22"

■ Bit shift instructions

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
SHL	Shift left	Shifts the input argument to the left by the specified number of bits. Inserts “0” from the least significant	●	●	●	"P.3-24"

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
		bit to the specified bit and outputs the data.				
SHR	Shift right	Shifts the input argument to the right by the specified number of bits. Inserts “0” from the most significant bit to the specified bit and outputs the data.	●	●	●	"P.3-25"
ROL	Rotate left	Shifts the input argument to the left by the specified number of bits. Inserts the value in excess from the most significant bit into the data starting from the least significant bit and outputs the data.	●	●	●	"P.3-25"
ROR	Rotate right	Shifts the input argument to the right by the specified number of bits. Inserts the value in excess from the least significant bit into the data starting from the most significant bit and outputs the data.	●	●	●	"P.3-26"

■ Numerical operation instructions

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
ABS	Absolute value	Outputs the absolute value.	●	●	●	"P.3-28"
SQRT	Square root	Outputs the the square root (√) of a number.	●	●	●	"P.3-28"
LN	Natural logarithm	Outputs the natural logarithm ($\log_e X$) of a number.	●	●	●	"P.3-29"
LOG	Common logarithm	Outputs the common logarithm ($\log_{10} X$) of a number.	●	●	●	"P.3-30"
EXP	Natural exponent	Outputs the natural exponent (e^X) of a number.	●	●	●	"P.3-31"
EXPT	Exponentiation	Outputs the exponentiation of a number (a^n).	●	●	●	"P.3-31"
SIN	Trigonometric function (sine)	Outputs the result of the sine function calculation.	●	●	●	"P.3-32"

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
COS	Trigonometric function (cosine)	Outputs the result of the cosine function calculation.	●	●	●	"P.3-33"
TAN	Trigonometric function (tangent)	Outputs the result of the tangent function calculation.	●	●	●	"P.3-34"
ASIN	Trigonometric function (arc sine)	Outputs the result of the arc sine function calculation.	●	●	●	"P.3-35"
ACOS	Trigonometric function (arc cosine)	Outputs the result of the arc cosine function calculation.	●	●	●	"P.3-36"
ATAN	Trigonometric function (arc tangent)	Outputs the result of the arc tangent function calculation.	●	●	●	"P.3-36"
SMC_PI	trigonometric constant	The conversion constants Pi, degree, and Radian are available.	●	●		"P.3-37"

■ Data type conversion instructions

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
<Type 1>_TO_<Type 2>	Data type conversion	Converts type 1 input argument to type 2.	●	●	●	"P.3-38"
TRUNC	Data type conversion	Changes the real number to the DINT-type data.	●	●	●	"P.3-45"
TRUNC_INT	Data type conversion	Changes the real number to the INT-type data.	●	●	●	"P.3-45"
BCD_TO_**	Data type conversion	Converts BCD data to binary data (BYTE type / INT type / WORD type / DWORD type).	●	●	●	"P.3-46"
**_TO_BCD	Data type conversion	Converts binary data (BYTE type / INT type / WORD type / DWORD type) to BCD data.	●	●	●	"P.3-49"
GRAY_TO_**	Data type conversion	Converts a Gray code to binary data (BYTE type / WORD type / DWORD type).	●	●	●	"P.3-51"
**_TO_GRAY	Data type conversion	Converts binary data (BYTE type / WORD type / DWORD type) to a Gray code.	●	●	●	"P.3-52"

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
BYTE_TO_HEXinASCII	Data type conversion	Converts a one-byte hexadecimal binary-coded value to a one-word ASCII code.	●	●	●	"P.3-54"
HEXinASCII_TO_BYTE	Data type conversion	Converts a one-word ASCII code to a one-byte hexadecimal binary-coded value.	●	●	●	"P.3-56"
MEM.Decode	Data type conversion	Converts data in units of byte to data in units of DWORD.	●	●	●	"P.3-57"
MEM.Encode	Data type conversion	Converts data in units of DWORD to data in units of byte.	●	●	●	"P.3-58"
MEM.PackArrayOfBoolToArrayOfByte	Data type conversion	Packs a BOOL type array into an array in bytes and copies a specified bit size data.	●	●	●	"P.3-60"
MEM.PackBitsTo**	Data type conversion	Packs BOOL type data and converts it to a BYTE, a WORD, or a DWORD.	●	●	●	"P.3-61"
MEM.PackBytesTo**	Data type conversion	Packs BYTE type data and converts it to one-word or one-dword data.	●	●	●	"P.3-67"
MEM.PackWordsToDword	Data type conversion	Packs WORD type data and converts it to a DWORD.	●	●	●	"P.3-68"
MEM.UnpackArrayOfByte	Data type conversion	Unpacks a BYTE type array to data in bits and copies the data to a specified BOOL array.	●	●	●	"P.3-69"

■ Bit operation instructions

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
EXTRACT	Bit extraction	Outputs a BOOL status at a specified bit of a DWORD.	●	●	●	"P.3-71"
PUTBIT	Bit change	Changes the status of a specified bit of a DWORD.	●	●	●	"P.3-72"
SWITCHBIT	Bit inversion	Inverts the status of a specified bit of a DWORD.	●	●	●	"P.3-73"
MEMUtils.BitCpy	Bit copying	Copies a specified size of bit data.	●	●	●	"P.3-73"

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
MEM.ReverseBitsIn**	Bit order change	Reverses the order of the bits of BYTE-, WORD-, or DWORD-type data and outputs the data of the bits in reverse order.	●	●	●	"P. 3-75"

■ Data manipulation instructions

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
SEL	Binary selector	Outputs “IN0” when the input argument G is FALSE and “IN1” when the input argument G is TRUE.	●	●	●	"P.3-78"
MUX	Multiplexer	Outputs the input argument value depending on the input argument K (0,1,2,…).	●	●	●	"P.3-79"
LIMIT	Limiter	Limits the value of the input argument IN between the input arguments MN and MX and outputs the data.	●	●	●	"P.3-80"
MAX	Maximum value	Outputs the maximum value of the input argument.	●	●	●	"P.3-81"
MIN	Minimum value	Outputs the minimum value of the input argument.	●	●	●	"P.3-81"
MEMUtils.Swap	Byte swapping	Swaps specified bytes (2, 4, or 8 bytes) in order.	●	●	●	"P. 3-82"
MEM.Compare	Memory comparison	Compares two specified memory block data pieces to determine whether they match	●	●	●	"P. 3-84"
MEM.FindBlock	メモリブロック検索	Searches memory block data for specified memory block data.	●	●	●	"P. 3-85"
MEM.FindByte	Find byte data	Searches specified memory block data for specified one-byte data.	●	●	●	"P. 3-86"
MEM.MemFill	Memory fill	Fills a specified size in data memory with a specified data value.	●	●	●	"P. 3-87"
MEM.MemMove	Memory copying	Copies a specified size in data memory onto copy destination data memory.	●	●	●	"P. 3-89"

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
MEM.High**	High byte/High WORD extraction	Outputs high byte / high WORD of an input value.	●	●	●	"P. 3-111"
MEM.Low**	Low byte/Low WORD extraction	Outputs low byte / low WORD of an input value.	●	●	●	"P. 3-91"
MEM.Reverse BYTEsIn**	Byte order change	Reverses the order of the bytes of WORD-, or DWORD-type data and outputs the data of the bytes in reverse order.	●	●	●	"P. 3-93"
MEM.Reverse WORDsInDWORD	WORD order change	Reverses the order of the bytes of DWORD-type data and outputs the data of the WORD in reverse order.	●	●	●	"P. 3-94"

■ Character string instructions

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
LEN/WLEN	Length of a character string	Outputs the length of a character string.	●	●	●	"P.3-96"
LEFT/WLEFT	Extracting characters from the left end	Extracts a character string consisting of the specified number of characters from the left of the character string.	●	●	●	"P.3-97"
RIGHT/ WRIGHT	Extracting characters from the right end	Extracts a character string consisting of the specified number of characters from the right of the character string.	●	●	●	"P.3-98"
MID/WMID	Extracting characters from the specified position	Extracts a character string consisting of the specified number of characters from the specified position of the character string.	●	●	●	"P.3-99"
CONCAT/ WCONCAT	Concatenating character strings	Concatenates two character strings.	●	●	●	"P.3-100"
INSERT/ WINSERT	Inserting a character string	Inserts another character string into the specified position of one character string.	●	●	●	"P.3-101"
DELETE/ WDELETE	Deleting a character string	Deletes a character string consisting of the specified number of characters from	●	●	●	"P.3-103"

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
		the specified position of the character string.				
REPLACE/ WREPLACE	Replacing a character string	Replaces a character string, consisting of the specified number of characters from the specified position of the character string, with another character string.	●	●	●	"P.3-104"
FIND/WFIND	Search for a character string	Searches for a specified character string in the character strings and outputs the position.	●	●	●	"P.3-105"
ConvertUTF16 toUTF8	Character code conversion	Converts a UTF-16 character string into a UTF-8 character string.	●	●	●	"P. 3-107"
ConvertUTF8t oUTF16	Character code conversion	Converts a UTF-8 character string into a UTF-16 character string.	●	●	●	"P. 3-108"

■ SD memory card slot instruction

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
SYS_GetSDC overState	Get SD card cover open / close state	Gets an open / close state of the card cover for the SD memory card slot.	●	●	-	"P.3-111"
SYS_GetSDAc cessRdy	Get SD card access ready state	Gets the state whether an access to the SD memory card is allowed.	●	●	-	"P.3-111"

■ CRC operation instructions

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
MEM.CRC16_ standard	CRC16	Calculates CRC16 checksum.	●	●	●	"P. 3-112"
MEM.CRC32	CRC32	Calculates CRC32 checksum.	●	●	●	"P. 3-113"

1.3 List of Function Block Instructions

This section provides lists of the function blocks used by the GM Programmer. These function blocks can be used with declaring the instances with variables.

1.3.1 Basic Instructions

■ Timer instructions

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
TON	Timer ON	Starts the timer when the input argument changes from FALSE to TRUE and, after an elapse of the specified time, the output argument outputs TRUE.	●	●	●	"P.4-2"
TOF	Timer OFF	Starts the timer when the input argument changes from TRUE to FALSE and, after an elapse of the specified time, the output argument outputs FALSE.	●	●	●	"P.4-3"
TP	Timer pulse	Starts the timer when the input argument changes from FALSE to TRUE until the specified time elapses. Outputs TRUE to the output argument while the timer keeps counting.	●	●	●	"P.4-4"
RTC	Realtime clock	Starts counting time from the specified date and time when the input argument changes from FALSE to TRUE. Outputs TRUE to the output argument while the clock keeps counting time.	●	●	●	"P.4-6"

■ Counter instructions

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
CTU	Up counter	Starts incrementing the counter value at the rising edge of the input argument CU and, after counting the specified number of count values, outputs TRUE.	●	●	●	"P.4-7"

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
CTD	Down counter	Starts decrementing from the specified number of count value at the rising edge of the input argument CD. Outputs TRUE when it reaches 0.	●	●	●	"P.4-8"
CTUD	Up-down counter	Starts incrementing the counter value at the rising edge of the input argument CU and, after counting the specified number of count values, outputs TRUE. Starts decrementing the counter value at the rising edge of the input argument CD and, when it reaches 0, outputs TRUE.	●	●	●	"P.4-9"

■ Edge detection instructions

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
R_TRIG	Rising edge detection	Outputs TRUE for one cycle only when detecting a rising edge.	●	●	●	"P.4-11"
F_TRIG	Falling edge detection	Outputs TRUE for one cycle only when detecting a falling edge.	●	●	●	"P.4-11"

■ Bistable circuit instructions

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
SR	Set-priority bistable circuit	If the input argument SET1 is TRUE, outputs TRUE. If the input argument RESET is TRUE, outputs FALSE. If both SET1 and RESET1 are TRUE, outputs TRUE	●	●	●	"P.4-13"
RS	Reset-priority bistable circuit	If the input argument SET1 is TRUE, outputs TRUE.	●	●	●	"P.4-14"

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
		If the input argument RESET is TRUE, outputs FALSE. If both SET1 and RESET1 are TRUE, outputs FALSE.				

■ Data Type Conversion Instructions

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
MEM.Unpack**	Data type conversion	Unpacks BYTE- , WORD-, or DWORD-type data to data in bits and outputs the data.	●	●	●	"P. 4-16"

■ Data manipulation instructions

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
LIN_TRAFO	linear conversion	Convert one range of numbers to another linearly.	●	●	●	"P. 4-22"
STATISTIC_REAL	Statistical data	Acquire the maximum, minimum, and average values of the input data (REAL type).	●	●	●	"P. 4-23"
LIMITALARM	Monitoring of input values	Monitor whether the input value is between LOW (lower limit) and HIGH (upper limit)	●	●	●	"P. 4-23"

■ Other instructions

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
BLINK	output of blinking signal	Switch the output argument OUT to TRUE or FALSE according to the setting time.	●	●	●	"P. 4-25"

1.3.2 Motion Control Function Blocks (Single Axis Control)

■ Servo ON

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
MC_Power	Servo ON	Sets the axis to the servo ON state to be ready for operation.	●	●	●	"P.5-2"

■ Home return

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
PMC_Home	Home return	Performs home return operation on the axis. Uses the home return function of the amplifier.	●	-	-	"P.5-4"
MC_Home	Home return	Performs home return operation on the axis. Uses the home return function of the amplifier.	-	●	●	"P.5-6"

■ Control switch

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
SMC_SetControllerMode	Control mode setup	Sets up the control mode for controlling the position, velocity, and torque.	●	●	-	"P.5-8"

■ Stop

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
MC_Stop	Forced stop	Causes the axis to make a deceleration stop. After stopping, the axis remains stopped while Execute is TRUE.	●	●	●	"P.5-9"
MC_Halt	Stop	Causes the axis to make a deceleration stop. After the axis is stopped or while the axis is being decelerated, other motion	●	●	●	"P.5-10"

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
		instructions can be executed.				

■ JOG / Inching

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
MC_Jog	Jogging	Causes the axis to keep traveling in a forward or reverse direction at a constant velocity while the input is TRUE.	●	●	●	"P.5-11"
SMC_Inch	Inching	Causes the axis to travel in a forward or reverse direction for a specified relative distance when the input turns TRUE.	●	●	●	"P.5-12"

■ Position control

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
MC_MoveAbsolute	Absolute value positioning	Causes the axis to travel to a position specified as an absolute position.	●	●	●	"P.5-14"
MC_MoveRelative	Relative value positioning	Causes the axis to travel to a position specified as a relative position.	●	●	●	"P.5-15"
MC_MoveAdditive	Change target position	Adds a relative distance to the target position of the immediately preceding instruction.	●	●	●	"P.5-16"
MC_MoveSuperImposed	Superimposed positioning	Adds a relative distance, a velocity, an acceleration, and a deceleration to the operations of the immediately preceding instruction.	●	●	●	"P.5-18"
MC_PositionProfile	Position profile move	Causes the axis to operate according to the profile data that consists of a combination of position and time.	●	●	●	"P.5-20"
SMC_MoveContinuousAbsolute	Absolute value position velocity move	Executes absolute value positioning and, after the axis reaches the target position, causes the axis	●	●	●	"P.5-24"

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
		to keep moving at a specified velocity.				
SMC_MoveContinuousRelative	Relative value position velocity move	Executes relative value positioning and, after the axis reaches the target position, causes the axis to keep moving at a specified velocity.	●	●	●	"P.5-25"

■ Velocity control

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
MC_MoveVelocity	Velocity control	Specifies the velocity of the axis.	●	●	●	"P.5-27"
MC_VelocityProfile	Velocity profile move	Causes the axis to operate according to the profile data that consists of a combination of time and velocity.	●	●	●	"P.5-28"
MC_AccelerationProfile	Acceleration profile move	Causes the axis to operate according to the profile data that consists of a combination of time and acceleration.	●	●	●	"P.5-29"

■ Torque control

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
PMC_SetTorque	Torque control	Specifies by % the torque of the axis.	●	●	●	"P.5-32"
SMC_SetTorque	Torque control	Specifies by Nm the torque of the axis.	-	●	●	"P.5-33"

1.3.3 Motion Control Function Blocks (Synchronous Control)

■ Cam operation

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
MC_CamIn	Start cam control	Starts cam synchronous operation.	●	●	●	"P.6-2"
MC_CamOut	Cancel cam operation	Cancels cam synchronous operation.	●	●	●	"P.6-5"
MC_CamTableSelect	Cam table selection	Specifies the cam table for cam synchronous operation.	●	●	●	"P.6-6"
SMC_GetTappetValue	Tappet output	Outputs the tappet set in the cam table.	●	●	●	"P.6-8"

■ Gear operation

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
MC_GearIn	Start gear operation	Starts gear synchronous operation.	●	●	●	"P.6-11"
MC_GearInPos	Position specified gear operation	Starts gear synchronous operation from the specified absolute position.	●	●	●	"P.6-12"
MC_GearOut	Cancel gear operation	Cancels the gear synchronous operation.	●	●	●	"P.6-14"

■ Phase correction

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
MC_Phasing	Master axis phase correction	Corrects the phase between the master and slave axes.	●	●	●	"P.6-16"

1.3.4 Motion Control Function Blocks (Interpolation Control)

■ Interpolation Control

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
PMC_ Interpolator2D	2-axis Interpolation Control	Specify the CNC pattern to perform 2-axis interpolation control.	●	●	●	"P.7-2"
PMC_ Interpolator3D	3-axis Interpolation Control	Specify the CNC pattern to perform 3-axis interpolation control.	●	●	●	"P.7-3"
PMC_NCDecoder	CNC Table Conversion	Convert the CNC table to executable format.	●	●	●	"P.7-5"

1.3.5 Motion Control Function Blocks (CNC Control)

■ G code decoding

Name	Function	Overview	RETX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
SMC_ NCDecoder	CNC Table Conversion	Convert the CNC table to executable format.	●	●	●	"P.8-2"

■ Pre-processing after decoding

Name	Function	Overview	RETX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
SMC_CheckVelocities	CNC table preprocessing	Check the angle between the paths and switch between P point control and C point control.	●	●	●	"P.8-6"
SMC_SmoothPath	CNC table preprocessing	Smooths the path of the specified CNC table.	●	●	●	"P.8-7"
SMC_RoundPath	CNC table preprocessing	Compensates with an arc between the paths in the specified CNC table.	●	●	●	"P.8-10"

■ Control calculation

Name	Function	Overview	RETX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
SMC_Interpolator	CNC control operation	Calculates discrete data for the control period from a continuous path.	●	●	●	"P.8-12"
SMC_PreAcknowledgeMFunction	M function	You can continue the operation without pausing by ignoring the description of the M function.	●	●	●	
SMC_GetMParameters	M function	Parameters can be received from the CNC processing unit that is paused by the M function.	●	●	●	

■ Control command & kinematics conversion

Name	Function	Overview	RETX	EtherCAT	Simulation (●: Supported, -: Not supported)	ページ
			(●: Supported, -: Not supported)			
SMC_ControlAxisByPos	Control command	Command the CNC position information to the motor.	●	●	●	"P.8-16"
SMC_TRAFO_Polar	Kinematics conversion	Convert 2D coordinates to polar coordinates.	●	●	●	"P.8-17"
SMC_TRAFOF_Polar	Kinematics conversion	Convert polar coordinates to 2D coordinates	●	●	●	"P.8-18"
SMC_TRAFO_Bipod_Arm	Kinematics conversion	Converts the XY coordinates of the hand of the Bipod robot to the angle of each axis motor.	●	●	●	"P.8-19"

1.3.6 Motion Control Function Blocks (Motion Communication Control)

■ RTEX

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
RTEX_ClearAmpAlarm	Clear amplifier alarm	Clears the amplifier's alarm.	●	-	-	"P.9-2"
RTEX_ReadAmpAlarm	Read amplifier alarm	Reads the amplifier's alarm.	●	-	-	"P.9-4"
RTEX_ReadAmpState	Amplifier alarm status	Reads the amplifier's alarm status.	●	-	-	"P.9-5"

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
RTEX_ReadAmpData	Amplifier monitor	Reads the amplifier's monitor data.	●	-	-	"P.9-6"
RTEX_ReadAmpParameter	Read amplifier parameter	Reads the amplifier's parameters.	●	-	-	"P.9-7"
RTEX_WriteAmpParameter	Write amplifier parameter	Writes the amplifier's parameters.	●	-	-	"P.9-7"
RTEX_WriteAmpEEPROM	Write amplifier EEPROM	Writes parameters of the servo amplifier to EEPROM.	●	-	-	"P.9-8"
RTEX_Reset	Reset RTEX	Resets the entire RTEX network.	●	-	-	"P.9-9"
RTEX_ClearAmpMultiTurnData	Clear the multi-turn data	Clears the multi-turn data of the amplifier.	●	-	-	"P.9-10"
RTEX_ClearAmpPositionalDeviation	Clear amplifier deviation counter	Clears the deviation counter of the amplifier.	●	-	-	"P.9-11"
RTEX_GetTrackingCommandError	Error	Measures the number of sent RTEX commands and the number of lost RTEX commands.	●	-	-	"P.9-12"
RTEX_ReadPot	Read NOT of amplifier	Reads the amplifier's NOT status.	●	-	-	"P.9-13"
RTEX_ReadNot	Read POT of amplifier	Reads the amplifier's POT status.	●	-	-	"P.9-13"

■ EtherCAT

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
ETC_CO_SdoRead	Read slave parameter	Reads EtherCAT slave parameters. Unlike ETC_CO_SdoRead4, this FB supports parameters longer than 4 bytes.	-	●	-	"P.9-15"
ETC_CO_SdoRead4	Read four bytes of slave parameter	Reads EtherCAT slave parameters. Unlike ETC_CO_SdoRead, this FB supports only parameters with 4 bytes or less.	-	●	-	"P.9-16"
ETC_CO_SdoReadDWord	Read double words of slave parameter	Just like ETC_CO_SdoRead4, this FB reads the EtherCAT slave parameters.	-	●	-	"P.9-17"

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
		The read data is stored in DWORD (dwData), not in an array.				
ETC_CO_SdoRead_Access	Read slave parameter index	Just like ETC_CO_SdoRead, this FB reads the EtherCAT slave parameters. By setting the xCompleteAccess input to TRUE and the bySubIndex input to 0, you can read complete indexes including all entries.	-	●	-	"P.9-18"
ETC_CO_SdoRead_Channel	Read priority specification of slave parameter	Reads all EtherCAT slave parameters.	-	●	-	"P.9-19"
ETC_CO_SdoWrite	Write slave parameter	Writes EtherCAT slave parameters. Unlike ETC_CO_SdoWrite4, this FB supports parameters longer than 4 bytes.	-	●	-	"P.9-21"
ETC_CO_SdoWrite4	Write four bytes of slave parameter	Writes EtherCAT slave parameters. Unlike ETC_CO_SdoWrite, this FB supports only parameters with 4 bytes or less.	-	●	-	"P.9-22"
ETC_CO_SdoWriteDWord	Write double words of slave parameter	Just like ETC_CO_SdoWrite4, this FB writes the EtherCAT slave parameters. The write data is transferred in DWORD (dwData), not in an array.	-	●	-	"P.9-24"
ETC_CO_SdoWrite_Access	Write slave parameter index	Just like ETC_CO_SdoWrite, this FB writes the EtherCAT slave parameters. By setting the xCompleteAccess input to TRUE and the bySubIndex input to 0, you can write complete indexes including all entries. By using the byChannelPriority (BYTE) input, you can specify the	-	●	-	"P.9-25"

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
		channel and priority using a CoE mailbox message.				
ReadIdentification	Read slave identification data	Reads identification data from EtherCAT slaves.	-	●	-	"P.9-26"
ReadMemory	Read slave memory	Reads the EtherCAT slave memory.	-	●	-	"P.9-27"
ReadNbrSlaves	Read the number of connected slaves	Reads the number of slaves currently connected.	-	●	-	"P.9-28"
WriteMemory	Write slave memory	Writes the EtherCAT slave memory.	-	●	-	"P.9-29"
PETC_ClearAmpPositionalDeviation	Clear amplifier deviation counter	Clears the deviation counter of the amplifier.	-	●	-	"P.9-30"
IoDrvEtherCAT.GetStatistics	Get EtherCAT frame statistics information	Gets the EtherCAT frame statistics information.	-	●	-	"P.9-31"

1.3.7 Motion Control Function Blocks (Auxiliary Function)

■ Motion auxiliary function (Monitoring)

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
MC_ReadActualPosition	Read actual position	Reads the actual position data of the axis.	●	●	●	"P.10-2"
MC_ReadActualVelocity	Read actual velocity	Reads the actual velocity of the axis.	●	●	●	"P.10-2"
PMC_ReadActualTorque	Read actual torque	Read the actual torque value of the axis.	●	●	-	"P.10-3"
MC_ReadActualTorque	Read actual torque	Reads the actual torque value of the axis.	-	●	-	"P.10-4"
MC_ReadAxisError	Read axis error	Gets general axis errors not related to function blocks.	-	●	●	"P.10-4"
MC_ReadStatus	Read status	Reads the status information of the axis.	●	●	●	"P.10-5"
SMC_InPosition	In-position judgment	Compares the actual position of the AMP with the command value and judges whether the	●	●	●	"P.10-7"

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
		position is within the specified range.				
SMC_ReadFBError	Read oldest error	Reads the oldest function block error information.	●	●	●	"P.10-8"
SMC_ClearFBError	Clear oldest error	Clears the oldest FB error information.	●	●	●	"P.10-9"
SMC_CheckAxisCommunication	Check axis communication state	Checks the communication state of the axis.	●	●	●	"P.10-10"
SMC_CheckLimits	Check exceeding limits	Checks whether the velocity, acceleration, or deceleration is in excess of the dynamic limit set value of the device.	●	●	●	"P.10-11"
SMC_GetMaxSetAccDec	Measure maximum acceleration / deceleration	Measures the maximum value of the axis acceleration / deceleration command.	●	●	●	"P.10-12"
SMC_GetMaxSetVelocity	Measure maximum velocity	Measures the maximum value of the axis velocity command.	●	●	●	"P.10-12"
SMC_GetTrackingError	Measure tracking error	Measures the tracking error of the actual position for the axis command position.	●	●	●	"P.10-13"
SMC_MeasureDistance	Measures turnaround travel distance	Measures the travel distance.	●	●	●	"P.10-14"
SMC_ReadSetPosition	Read axis set position	Reads the set command position of the axis.	●	●	●	"P.10-15"

■ Motion auxiliary function (Change / reset)

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
MC_Reset	Reset axis error	Resets the state transition error of the axis.	●	●	●	"P.10-16"
SMC3_ReinitDrive	Reinitialize axis	Restarts the servo amplifier / axis.	●	●	●	"P.10-16"
MC_SetPosition	Change actual position	Changes the actual command position of the axis.	●	●	●	"P.10-17"
SMC_ChangeDynamicLimits	Change axis settings	Change the dynamic limit of the axis.	●	●	●	"P.10-18"
SMC_ChangeGearingRatio	Change axis settings	Change the shaft type and gear ratio.	●	●	●	"P.10-19"

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
SMC_SetMovementType	Change axis settings	Change the type of the virtual axis.	●	●	●	"P.10-22"
SMC_SetRampType	Change axis settings	Change the speed ramp type of the axis.	●	●	●	"P.10-23"
SMC_SetSoftwareLimits	Change axis settings	Change the soft limit of the axis.	●	●	●	"P.10-25"

■ Motion auxiliary function (Other functions)

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
PMC_ReadLatchPosition	Monitor AMP latch position	Monitors the AMP latch position.	●	-	-	"P.10-27"
PMC_StopLatchPosition	Stop AMP latch monitoring	Stops the axis at the AMP latch position.	●	-	-	"P.10-29"
MC_TouchProbe	Enable AMP latch monitoring	Reads the axis position when a trigger signal occurs.	-	●	●	"P.10-32"
MC_AbortTrigger	Disable AMP latch monitoring	Aborts the trigger event (MC_TouchProbe).	-	●	●	"P.10-34"
MC_DigitalCamSwitch	Enable digital cam switch	Performs ON / OFF control on the digital output according to the axis position.	●	●	●	"P.10-35"
SMC_BacklashCompensation	Compensate backlash	Compensates the backlash.	●	●	●	"P.10-38"

1.3.8 Function Blocks (Others)

■ COM port (General-purpose communication)

The following table lists the function blocks that are used to perform general-purpose communication with the COM port.

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
COM.Open	Open COM port	Opens the COM port.	●	●	-	"P.11-4"

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
COM.Close	Close COM port	Closes the COM port.	●	●	-	"P.11-6"
COM.Read	Read COM port	Reads data from the COM port.	●	●	-	"P.11-7"
COM.Write	Write COM port	Writes data to the COM port.	●	●	-	"P.11-8"
COM.ERROR	Error ID	This is an enumeration type error ID that is output when the COM port (general-purpose communication) function block is executed.	●	●	-	"P.11-8"

■ COM port (Modbus COM)

The following table lists the instructions that are used to perform ModbusRTU communication with the COM port.

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
IoDrvModbusComPort	ModbusComPort device	This is a function block that controls the Modbus_Master_COM_Port device.	●	●	-	"P.11-10"
IoDrvModbus.ModbusChannel	Start sending Modbus command	Sends the command set in the Modbus Slave channel of the ModbusSlaveCOM_Port device.	●	●	-	"P.11-10"
IoDrvModbus.ModbusRequest	Modbus request	Processes the Modbus command specified by I/O without using the ModbusMasterComPort device.	●	●	-	"P.11-11"
IoDrvModbus.ModbusRequest2	Modbus request 2	Like the ModbusRequest, processes the Modbus command specified by I/O without using the ModbusMasterComPort device.	●	●	-	"P.11-12"
IoDrvModbus.ModbusSlaveComPort	ModbusSlaveComPort device	This is a function block that controls the Modbus_Slave_COM_Port device.	●	●	-	"P.11-14"
IoDrvModbus.MB_ErrorCodes	Error code	This is an enumeration type error code that is output when the function block for Modbus communication instruction	●	●	-	"P.11-14"

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
		that uses the COM port is executed.				

■ LAN port (IoDrvEthernet)

The following table lists the library functions that are used for the network interface to perform communication with the LAN port.

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
IoDrvEthernet	Ethernet device	This is a function block that acquires the status of the LANPort device.	●	●	-	"P.11-16"
IoDrvEthernet.IPARRAY_TO_INADDR	From array type to union type	This is a function that converts an array type IP address to an INADDR (union type).	●	●	●	"P.11-16"
IoDrvEthernet.IPARRAY_TO_IPSTRING	From array type to character string type	This is a function that converts an array type IP address to a character string type.	●	●	●	"P.11-17"
IoDrvEthernet.IPARRAY_TO_UDINT	From array type to UDINT type	This is a function that converts an array type IP address to a UDINT type.	●	●	●	"P.11-17"
IoDrvEthernet.IPSTRING_TO_UDINT	From character string type to UDINT type	This is a function that converts a character string type IP address to a UDINT type.	●	●	●	"P.11-18"
IoDrvEthernet.UDINT_TO_IPARRAY	From UDINT type to array type	This is a function that converts a UDINT type IP address to an array type.	●	●	●	"P.11-18"
IoDrvEthernet.UDINT_TO_IPSTRING	From UDINT type to character string type	This is a function that converts a UDINT type IP address to an array type.	●	●	●	"P.11-18"

■ LAN port (General-purpose communication)

The following table lists the library functions that are used to perform general-purpose communication with the LAN port using the TCP or UDP protocol.

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
NBS.TCP_Client	Connect to TCP client	Connects to the TCP/IP client.	●	●	-	"P.11-20"

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
NBS.TCP_Connection	Connect TCP	Establishes the connection of the client connecting to the connection port opened by TCP_Server.	●	●	-	"P.11-20"
NBS.TCP_Read	Receive TCP data	Acquires data received by the connection port that is established by TCP_Connection.	●	●	-	"P.11-21"
NBS.TCP_Server	Connect TCP server	Opens the specified port as a TCP/IP connection port.	●	●	-	"P.11-22"
NBS.TCP_Write	Send TCP data	Sends data to the connection port that is established by TCP_Connection.	●	●	-	"P.11-23"
NBS.UDP_Peer	Open UDP port	Opens the UDP/IP port.	●	●	-	"P.11-23"
NBS.UDP_Receive	Receive UDP data	Receives data to the connection handle acquired by UDP_Peer.	●	●	-	"P.11-24"
NBS.UDP_Send	Send UDP data	Sends data to the connection handle acquired by UDP_Peer.	●	●	-	"P.11-26"
NBS.ERROR	Error code	This is an enumeration type error code that is output when the function block for communication instruction that uses the LAN port is executed.	●	●	-	"P.11-25"

■ LAN port (Modbus TCP)

The following table lists the library functions that are used to perform ModbusTCP communication with the LAN port.

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
IoDrvModbusTCP	ModbusTCP device	This is a function block that controls the Modbus_TCP_Master device.	●	●	-	"P.11-28"
IoDrvModbusTCP.ModbusChannel	Start sending Modbus command	Sends the command set in the Modbus Slave channel of the ModbusTCP_Slave device.	●	●	-	"P.11-28"

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
IoDrvModbusTCP.ModbusRequest	Modbus request	Processes the Modbus command specified by I/O without using the Modbus_TCP_Slave device.	●	●	-	"P.11-29"
IoDrvModbusTCPSlave	ModbusTCPSlave device	This is a function block that controls the Modbus__TCP_Slave device.	●	●	-	"P.11-30"
IoDrvModbus.MB_ErrorCodes	Error code	This is an enumeration type error code that is output when the function block for Modbus communication instruction that uses the LAN port is executed.	●	●	-	"P.11-31"

■ LAN port (EtherNet/IP)

The following table lists instructions that are used to control EtherNet/IP scanner and adapter functions using the GM1 controller.

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
IoDrvEtherNetIP	EtherNet/IP scanner device	This is a function block that controls the EtherNet/IP scanner device.	●	●	-	"P.11-33"
RemoteAdapter	Remote adapter device	This is a function block for the Remote adapter device linked to the EtherNet/IP scanner device.	●	●	-	"P.11-34"
IoDrvEtherNetIPAdapter	EtherNet/IP adapter device	This is a function block that controls the EtherNet/IP adapter device.	●	●	-	"P.11-36"
Module	EtherNet/IP module device	This is a function block that controls the EtherNet/IP module device.	●	●	-	"P.11-37"
Apply_Attributes	Apply_Attributes service	This is a function block that calls Apply_Attributes service of the CIP object instance.	●	●	-	"P.11-38"
Generic_Service	Execute generic service	This is a function block that executes generic services with the EtherNet/IP adapter.	●	●	-	"P.11-39"

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
Get_Attribute_Single	Inquire specific attributes of a specific instance	This is a function block that inquires specific attributes of a specific instance of the CIP object	●	●	-	"P.11-40"
Get_Attributes_All	Inquire all attributes of a specific instance	This is a function block that inquires all attributes of a specific instance of the CIP object	●	●	-	"P.11-41"
Set_Attribute_Single	Set specific attributes of a specific instance	This is a function block that sets specific attributes of a specific instance of the CIP object	●	●	-	"P.11-42"
Set_Attributes_All	Set all attributes of a specific instance	This is a function block that sets all attributes of a specific instance of the CIP object	●	●	-	"P.11-43"
NOP	NOP service	This is a function block that executes the NOP service of a specific instance of the CIP object	●	●	-	"P.11-44"
Reset	Reset service	This is a function block that executes the Reset service of a specific instance of the CIP object	●	●	-	"P.11-45"
Start	Start service	This is a function block that executes the Start service of a specific instance of the CIP object	●	●	-	"P.11-45"
Stop	Stop service	This is a function block that executes the Stop service of a specific instance of the CIP object	●	●	-	"P.11-46"
ENIP.ERROR	Message service instruction error code	-	●	●	-	"P.11-47"
ENIP.CIPClass	Service class code	-	●	●	-	"P.11-49"

■ SD card operation (File operation)

Files in the SD card inserted in the SD memory card slot can be operated.

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
FILE.Open	Open file	Opens a file or creates a new file.	●	●	-	"P.11-52"
FILE.Close	Close file	Closes a file.	●	●	-	"P.11-53"

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
FILE.Read	Read file	Reads data from the file opened by the Open instruction.	●	●	-	"P.11-53"
FILE.Write	Write file	Writes data to the file opened by the Open instruction.	●	●	-	"P.11-54"
FILE.Flush	Flush file	Flushes buffer contents to the file opened by the Open instruction.	●	●	-	"P.11-55"
FILE.Copy	Copy file	Copies a file.	●	●	-	"P.11-56"
FILE.Rename	Rename file	Changes a file name.	●	●	-	"P.11-57"
FILE.Delete	Delete file	Deletes a file.	●	●	-	"P.11-58"
FILE.EOF	EOF of file	Determines whether the current offset of a file is EOF (End Of File) or not.	●	●	-	"P.11-58"
FILE.GetAttribute	Get file attribute	Gets file attributes (compressed, hidden, normal, read only).	●	●	-	"P.11-59"
FILE.GetPos	Get file offset	Gets the current offset of a file.	●	●	-	"P.11-60"
FILE.GetSize	Get file size	Gets the file size.	●	●	-	"P.11-60"
FILE.GetTime	Get file update time	Get the update time of a file	●	●	-	"P.11-61"
FILE.SetPos	Set file offset	Sets the offset of a file.	●	●	-	"P.11-62"

■ SD card operation (Directory operation)

Directories in the SD card inserted in the SD memory card slot can be operated.

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
FILE.DirCreate	Create directory	Creates a directory with a specified name.	●	●	-	"P.11-64"
FILE.DirOpen	Open directory	Opens a directory.	●	●	-	"P.11-64"
FILE.DirClose	Close directory	Closes a directory	●	●	-	"P.11-65"
FILE.DirCopy	Copy directory	Copies a directory.	●	●	-	"P.11-66"
FILE.DirRename	Rename directory	Renames a directory	●	●	-	"P.11-67"
FILE.DirRemove	Delete directory	Deletes a directory.	●	●	-	"P.11-67"
FILE.DirList	Directory list	Outputs a list of directories and files inside the directory.	●	●	-	"P.11-68"

■ Clock setting

The following table lists the function blocks that are used to set the clock of the GM1 Controller.

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
SYS_GetTime	Get time	This is a function block that gets the current local time	●	●	-	"P.11-70"
SYS_SetTime	Set time	This is a function block that sets the current local time.	●	●	-	"P.11-70"
SYS_GetTimezone	Get time zone information	This is a function block that gets the time zone information.	●	●	-	"P.11-71"
SYS_SetTimezone	Set time zone information	This is a function block that sets the time zone information.	●	●	-	"P.11-71"
SYS_DateConcat	Convert from UINT type to DATE type	This is a function that converts a UINT type date to a DATE type.	●	●	-	"P.11-72"
SYS_DateSplit	Convert from DATE type to UINT type	This is a function that converts a DATE type date to a UINT type.	●	●	-	"P.11-73"
SYS_DTConcat	Convert from UINT type to DT type	This is a function that converts a UINT type date and time to a DT type.	●	●	-	"P.11-73"
SYS_DTSplit	Convert from UINT type to DT type	This is a function that converts a UINT type date and time to a DT type.	●	●	-	"P.11-74"
SYS_GetDayOfWeek	Get day of the week	This is a function that gets the day of the week from the DATE type date.	●	●	-	"P.11-75"
SYS_TODConcat	Convert from UINT type to TOD type	This is a function that converts a UINT type time with milliseconds to a TOD type.	●	●	-	"P.11-75"
SYS_TODSplit	Convert from UINT type to TOD type	This is a function that converts a TOD type time with milliseconds to a UINT type.	●	●	-	"P.11-76"
ERROR	Clock instruction error code	-	●	●	-	"P.11-77"

■ System data

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
SYS_GetSystemError	Get system error	Gets the information of a system error that has occurred in the GM1 Controller.	●	●	-	"P.11-78"

■ PID control

This is a function block related to PID control of the GM1 controller.

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
PD	PD control	Performs PD control.	●	●	—	"P.11-79"
PID	PID control	Performs PID control.	●	●	—	"P.11-80"
PID_FIXCYCLE	PID control (any cycle time)	Performs PID control. Cycle time can be manually set.	●	●	—	"P.11-81"

■ Recipe Function

It is a method list of the function block RecipeManCommands of the recipe function.

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
CreateRecipe	Creating a recipe	Create a new recipe.	●	●	●	"P.11-84"
DeleteRecipe	Delete recipe	Delete the recipe.	●	●	●	"P.11-86"
LoadFromAndWriteRecipe	Read recipe file	Reads the value from the recipe file and writes to the recipe and the current value.	●	●	●	"P.11-87"
ReadAndSaveRecipe	Save to recipe file	Save the current value in the recipe and recipe file.	●	●	●	"P.11-89"
prvCompareRecipe	Recipe comparison	Compare the recipe with the current value.	●	●	●	"P.11-90"
ReloadRecipes	Reload recipe file	Read the recipe (inside) from the recipe file in the SD card.	●	●	●	"P.11-92"
GetRecipeCount	Get the number of recipes	Gets the number of recipes that belong to the recipe definition.	●	●	●	"P.11-93"

Name	Function	Overview	RTEX	EtherCAT	Simulation (●: Supported, -: Not supported)	Page
			(●: Supported, -: Not supported)			
GetRecipeNames	Get a list of recipe names	Gets a list of recipe names that belong to the recipe definition.	●	●	●	"P.11-94"
GetLastError	Get last error information	Gets the ReturnValues values for last processing.	●	●	●	"P.11-96"
GetLastInfo	Get last info information	Gets the InfoValues values for last processing.	●	●	●	"P.11-97"
ResetLastError	Reset last error information	Resets the value of GetLastError.	●	●	●	"P.11-99"
ResetLastInfo	Clear last info information	Resets the value of GetLastInfo.	●	●	●	"P.11-100"

1.3.9 Function Blocks (For the GM1 Pulse Output Unit)

The following table lists the function blocks used to control the GM1 Pulse Output Unit.

Name	Function	Overview	RTEX	EtherCAT	Simulation	Page
			(●: Supported, —: Not supported)			
PG_Power	Servo ON or OFF	Performs servo ON/OFF control.	●	●	-	"P.12-4"
PG_Jog	Jogging	Causes the axis to keep traveling in a forward or backward direction.	●	●	-	"P.12-5"
PG_MoveAbsolute	Absolute value positioning	Causes the axis to travel to a position specified as an absolute position.	●	●	-	"P.12-6"
PG_MoveRelative	Relative value positioning	Causes the axis to travel to a position specified as a relative position.	●	●	-	"P.12-7"
PG_LatchPosition	Latch relative positioning	Causes the axis to travel to the relative position specified by an external signal input.	●	●	-	"P.12-8"
PG_Pulser	Pulser operation	Enables constant speed operation for the axes using an external pulse input.	●	●	-	"P.12-10"
PG_Stop	Forced stop and deceleration stop	Causes the axis to make a forced stop or deceleration stop	●	●	-	"P.12-11"
PG_Home	Home return	Causes the axis to make a home return.	●	●	-	"P.12-12"

Name	Function	Overview	RTEX	EtherCAT	Simulation	Page
			(●: Supported, —: Not supported)			
PG_SetPosition	Elapsed value and feedback counter settings	Sets the elapsed value and the feedback counter to desired values.	●	●	-	"P.12-13"
PG_WriteParameter	Write parameters	Writes the parameters to the pulse output unit.	●	●	-	"P.12-14"
PG_ReadParameter	Read parameters	Reads the parameters from the pulse output unit.	●	●	-	"P.12-17"
PG_ClearError	Clear errors	Clears the limit error or the set value error of the pulse output unit.	●	●	-	"P.12-18"
PG_ReadStatus	Read status	Reads the status from the pulse output unit.	●	●	-	"P.12-18"

1.4 List of Function Block Instructions that Cannot Be Used with the GM1

■ Instructions not available for Modbus

The following function blocks in the IoDrvModbusTCP, IoDrvModbusTCPSlave, IoDrvModbus, and IoDrvModbusSerialSlave libraries are not available for the GM1 Controller.

Name	Function	Alternative function	Page
ModbusTCPSlaveBase	-	-	-
ModbusTCPSlaveUnit	-	-	-
ModbusTCPSlaveUnit_Diag	-	-	-
IoDrvModbusTCP_Diag	-	-	-
ModbusTCPSlave_Diag	-	-	-
ModbusTCPDeviceDiag	-	-	-
IoDrvModbusComPort_Diag	-	-	-
ModbusSlaveComPort_Diag	-	-	-
IoDrvModbusSerialSlave	-	-	-
ModbusSerialDeviceDiag	-	-	-
ModbusServer	-	-	-

■ Instructions not available for general-purpose communication

The following function blocks in the CAA NBS(Net Base Services) library are not available for the GM1 Controller.

Name	Function	Alternative function	Page
TCP_ReadBuffer	-	-	-
TCP_WriteBuffer	-	-	-
UDP_ReceiveBuffer	-	-	-
UDP_SendBuffer	-	-	-
DummyJob	-	-	-

■ Instructions not available for EtherNet/IP

The following function blocks in the IoDrvEtherNetIP and IoDrvEtherNetIPAdapter libraries are not available for the GM1 Controller.

Name	Function	Alternative function	Page
IoDrvEtherNetIP_diag	-	-	-
RemoteAdapter_diag	-	-	-
AdapterDiagnosis	-	-	-
IoDrvEtherNetIPAdapter_Diag	-	-	-
Module_Diag	-	-	-

■ Instructions not available for motion control

The following function blocks in the SM3_Basic library are not available for the GM1 Controller. Alternative functions are listed, if available.

Name	Function	Alternative function	Page
SMC_Commissioning	Commissioning status	Commissioning function of the GM Programmer	-
SMC_SetCustomRampType	Set acceleration / deceleration custom operation	-	-
SMC_CAM_ObjectManager	Manage cam data	-	-
SMC3_CommunicateDriveParameter	Communication setting	RTEX_ReadAmpParameter	"P.9-7"
SMC3_ReadDriveParameter	Read drive parameter	RTEX_ReadAmpParameter	"P.9-7"
SMC3_ReadParameter	Read parameter	RTEX_ReadAmpParameter	"P.9-7"
SMC3_WriteDriveParameter	Write drive parameter	RTEX_WriteAmpParameter	"P.9-7"
SMC3_WriteParameter	Write parameter	RTEX_WriteAmpParameter	"P.9-7"
MC_ReadBoolParameter	Read BOOL-type parameter	RTEX_ReadAmpParameter	"P.9-7"
MC_ReadParameter	Read parameter	RTEX_ReadAmpParameter	"P.9-7"
MC_WriteBoolParameter	Write BOOL-type parameter	RTEX_WriteAmpParameter	"P.9-7"
MC_WriteParameter	Write parameter	RTEX_WriteAmpParameter	"P.9-7"
SMC_VIRTUAL_AXIS	Set virtual axis	-	-
SMC3_BrakeStatus	Get brake status	-	-
SMC3_BrakeControl	Brake control	-	-
SMC3_PersistPosition	Persist actual axis position	-	-
SMC3_PersistPositionLogical	Persist logical axis position	-	-
SMC3_PersistPositionSingleturn	Persist actual axis position with a range	-	-
SMC_PerfStat	Calculate performance statistics	-	-
SMC_SeriesStat	Calculate increment statistics	-	-
SMC_AxisDiagnosticLog	Log axis parameter	-	-
FB_Template_Edge		-	-
FB_Template_EdgeAbort		-	-
FB_Template_EdgeAbortTimeout		-	-
SMC_StartupDrive		-	-
SMC_CAMBounds		-	-
SMC_CAMBounds_Pos		-	-
SMC_CamEditor		-	-
SMC_CamRegister		-	-
SMC_GetCamSlaveSetPosition		-	-

Name	Function	Alternative function	Page
SMC_ReadCAM		-	-
SMC_WriteCAM		-	-
SMC_PerfTimerSum		-	-
SMC_FollowPosition		-	-
SMC_FollowPositionVelocity		-	-
SMC_FollowSetValues		-	-
SMC_FollowVelocity		-	-
SMC_Homing		-	-
ETC_CO_SdoInfoGeEntryDescription	Read object name	-	-
ETC_CO_SdoInfoGetODList	Read object tree	-	-
ETC_CO_SdoInfoGetObjectDescription	Read object information	-	-
ReadEEpromData	Read slave EEPROM	-	-
ReadWriteEEprom	Read / write slave EEPROM	-	-

(MEMO)

2 Ladder Instructions

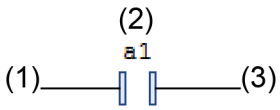
2.1 Ladder Instructions

This section describes ladder instructions that can be used for ladder diagram program (LD program).

2.1.1 NO Contact

If the variable corresponding to the contact is TRUE, then the input value is output. If the variable is FALSE, then FALSE is output.

■ Icon



■ Parameter

No.	Scope	Type	Description
(1)	Input	BOOL	Input to the NO contact
(2)	Variable name	BOOL	Variable that corresponds to the NO contact
(3)	Output	BOOL	Output from the NO contact

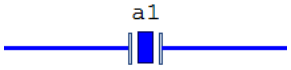
■ Input method

Use one of the following methods to input the NO contact.

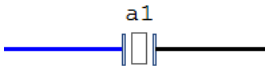
- From the tool box, select **Ladder elements**>**NO contact** and drag to "Start from here".
- Right-click on the network, and, from the displayed menu, select **Insert Contact**.
- Click the icon on the toolbar.
- From the menu, select **FBD / LD / IL**>**Insert Contact**.
- Press the shortcut keys <Ctrl+k> simultaneously.

■ Program example

If the variable (a1) corresponding to the NO contact is TRUE, then the value input to the NO contact (TRUE) is output as is.



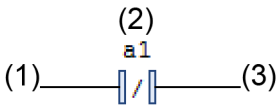
If the variable (a1) corresponding to the contact is FALSE, then FALSE is output.



2.1.2 NC Contact

If the variable corresponding to the contact is TRUE, then FALSE is output. If the variable is FALSE, then the input value is output.

■ Icon



■ Parameter

No.	Scope	Type	Description
(1)	Input	BOOL	Input to the NC contact
(2)	Variable name	BOOL	Variable that corresponds to the NC contact
(3)	Output	BOOL	Output from the NC contact

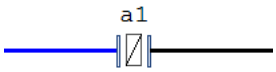
■ Input method

Use one of the following methods to input the NC contact.

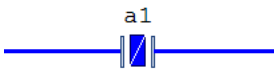
- From the tool box, select **Ladder elements**>**NC contact** and drag to "Start from here".
- Right-click on the network, and, from the displayed menu, select "Insert NC contact".
- Click the icon on the toolbar.
- From the menu, select **FBD / LD / IL**>**Insert NC contact**.

■ Program example

If the variable (a1) corresponding to the NC contact is TRUE, then FALSE is output.



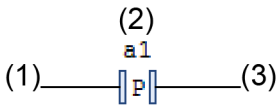
If the variable (a1) corresponding to the NC contact is FALSE, then the value input to the NC contact (TRUE) is output as is.



2.1.3 Rising Edge Detection Contact

If a rising edge is detected in the variable corresponding to the contact, then the input value is output for one cycle only.

■ Icon



■ Parameter

No.	Scope	Type	Description
(1)	Input	BOOL	Input to the contact
(2)	Variable name	BOOL	Variable that corresponds to the rising edge detection contact
(3)	Output	BOOL	Output from the contact

■ Input method

Input the rising edge detection contact by first inputting the NO contact and then changing the NO contact.

Select the NO contact just input and then perform one of the following operations.

- Right-click and, from the displayed menu, select **Edge detection** .
- From the menu, select **FBD / LD / IL**>**Edge detection** .
- Press the shortcut keys <Ctrl+e> simultaneously.
- Click the icon on the toolbar.

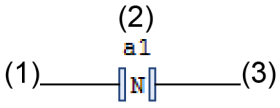
■ Program example

The following program is designed to detect the rising edge with the variable (a1) corresponding to the rising edge detection contact and to output TRUE for one cycle only.

2.1.4 Falling Edge Detection Contact

If a falling edge is detected in the variable corresponding to the contact, then the input value is output for one cycle only.

■ Icon



■ Parameter

No.	Scope	Type	Description
(1)	Input	BOOL	Input to the contact

No.	Scope	Type	Description
(2)	Variable name	BOOL	Variable that corresponds to the falling edge detection contact
(3)	Output	BOOL	Output from the contact

■ Input method

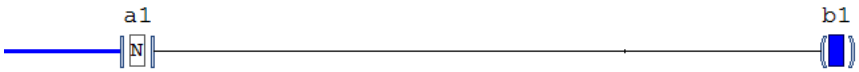
Input the falling edge detection contact by first inputting the NO contact and then changing the NO contact.

Select the NO contact just input and then perform one of the following operations.

- Right-click and, from the displayed menu, select **Edge detection** twice.
- From the menu, select **FBD / LD / IL>Edge detection** twice.
- Press the shortcut keys Ctrl+e simultaneously twice.
- Click the icon on the toolbar twice.

■ Program example

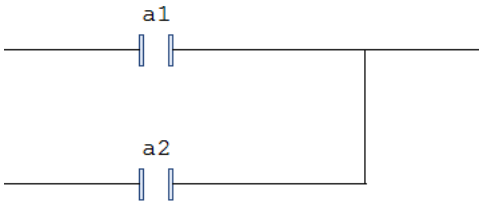
The following program is designed to detect the falling edge with the variable (a1) corresponding to the falling edge detection contact and to output TRUE for one cycle only.



2.1.5 Parallel NO Contact

NO contacts can be input in parallel to the initial contact. Of the contacts wired in parallel, if the output of one or more contacts is TRUE, TRUE is output.

■ Icon



■ Input method

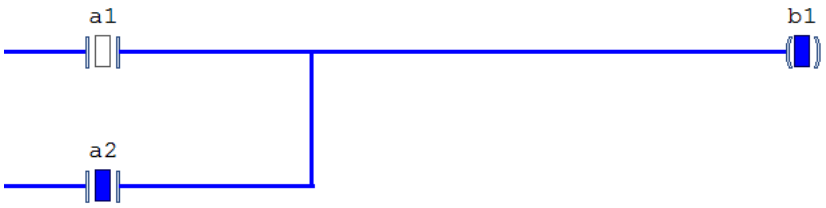
To input a parallel NO contact below the initial contact, select **Ladder elements >Parallel NO contact** from the tool box and drag to the position indicated with "▼" next to the contact.

Or, with the contact selected, perform one of the following operations.

- Right-click, and, from the displayed menu, select **Insert contact in parallel (below)**.
- From the menu, select **FBD/LD/IL>Insert contact in parallel (below)**.
- Press the shortcut keys <Ctrl+r> simultaneously.
- Click the icon on the toolbar.

■ **Program example**

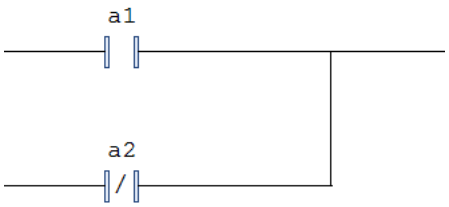
This program is designed to input one NO contact in parallel to the NO contact. TRUE is output because the NO contact below is TRUE.



2.1.6 Parallel NC Contact

NC contacts can be input in parallel to the initial contact. Of the contacts wired in parallel, if the output of one or more contacts is TRUE, TRUE is output.

■ **Icon**



■ **Input method**

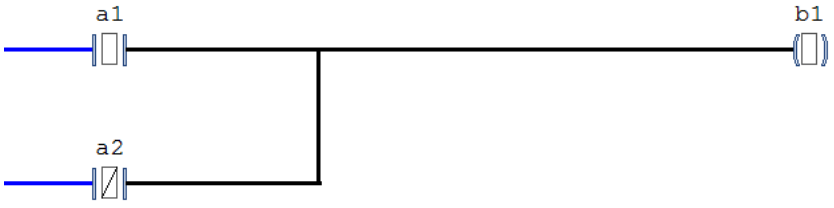
To input a parallel NC contact below the initial contact, select **Ladder elements** >**Parallel NC contact** from the tool box and drag to the position indicated with "▼" next to the contact.

Or, with the contact selected, perform one of the following operations.

- Right-click, and, from the displayed menu, select "Insert NC contact in parallel (below)".
- From the menu, select **FBD / LD / IL**>**Insert NC contact in parallel (below)**.
- Click the icon on the toolbar.

■ **Program example**

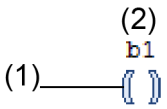
This program is designed to input one NC contact in parallel to the NO contact. FALSE is output because the outputs of both contacts are FALSE.



2.1.7 Coil

The input value is saved in the variable corresponding to the coil. If the input value is TRUE, then TRUE is saved. If the input value is FALSE, then FALSE is saved.

■ Icon



■ Parameter

No.	Scope	Type	Description
(1)	Input	BOOL	Input to the coil
(2)	Variable name	BOOL	Name of the variable that corresponds to the coil

■ Input method

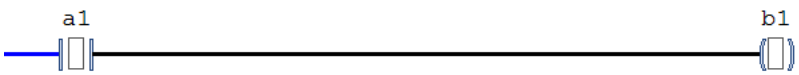
Use one of the following methods to input the coil.

- From the tool box, select **Ladder elements**> **Coil** and drag to "Add output or jump here" (when connecting to a contact).
- Right-click on the network, and, from the displayed menu, select **Insert Coil** .
- Click the icon on the tool bar.
- From the menu, select **FBD / LD / IL**>**Insert Coil** .
- Press the shortcut keys <Ctrl+a> simultaneously.
- Click the icon on the toolbar.

■ Program example

This program is designed to input the output from the NO contact to the coil.
TRUE is saved in the variable (b1) because the input to the coil is TRUE.

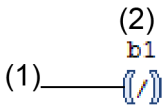
FALSE is saved in the variable (b1) because the input to the coil is FALSE.



2.1.8 Negated Coil

The negated value of the input is saved in the variable corresponding to the coil. If the input value is TRUE, then FALSE is saved. If the input value is FALSE, then TRUE is saved.

■ Icon



■ Parameter

No.	Scope	Type	Description
(1)	Input	BOOL	Input to the negated coil
(2)	Variable name	BOOL	Name of the variable that corresponds to the negated coil

■ Input method

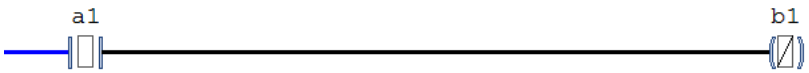
The negated coil can be input by inputting a coil and changing it.
With the input coil selected, perform one of the following operations.

- Right-click and, from the displayed menu, select **Negation**.
- From the menu, select **FBD / LD / IL**>**Negation**.
- Press the shortcut keys <Ctrl+n> simultaneously.
- Click the icon on the toolbar.

■ Program example

This program is designed to input the output from the NO contact to the negated coil.
FALSE is saved in the variable (b1) because the input to the coil is TRUE.

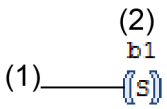
TRUE is saved in the variable (b1) because the input to the coil is FALSE.



2.1.9 Set Coil

When the input value turns TRUE, TRUE is saved in the variable corresponding to the coil.
TRUE is held until the input to the reset coil that corresponds to the same variable turns TRUE.

■ Icon



■ Parameter

No.	Scope	Type	Description
(1)	Input	BOOL	Input to the set coil.
(2)	Variable name	BOOL	Name of the variable that corresponds to the set coil

■ Input method

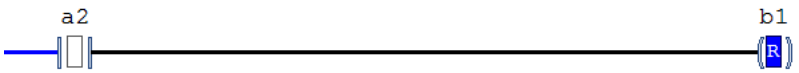
Use one of the following methods to input the set coil.

- From the tool box, select **Ladder elements**>**Set Coil** and drag to "Add output or jump here" (when connecting to a contact).
- Right-click on the network, and, from the displayed menu, select "Insert Set Coil ".
- Click the icon on the toolbar.
- From the menu, select **FBD / LD / IL**>**Insert Set Coil**.

■ Program example

This program is designed to input the output from the NO contact to the set coil and the reset coil.

TRUE is saved in the set coil variable (b1) because the input to the set coil is TRUE.



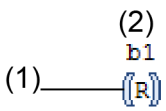
Info.

- Each set coil should be accompanied by a reset coil.

2.1.10 Reset Coil

When the input value turns TRUE, FALSE is saved in the variable corresponding to the coil. FALSE is held until the input to set coil that corresponds to the same variable turns TRUE.

■ Icon



■ Parameter

No.	Scope	Type	Description
(1)	Input	BOOL	Input to the reset coil.
(2)	Variable name	BOOL	Name of the variable that corresponds to the reset coil

■ Input method

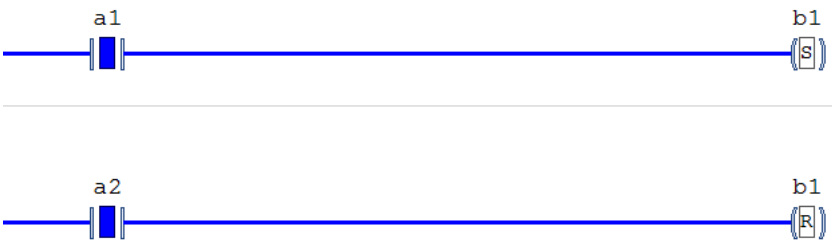
Use one of the following methods to input the reset coil.

- From the tool box, select **Ladder elements**>**Reset Coil** and drag to "Add output or jump here" (when connecting to a contact).
- Right-click on the network, and, from the displayed menu, select **Reset Coil**.
- Click the icon on the toolbar.
- From the menu, select **FBD / LD / IL**>**Insert Reset Coil** .

■ Program example

This program is designed to input the output from the NO contact to the set coil and the reset coil.

FALSE is saved in the variable (b1) because the input to the reset coil is TRUE.



Info.

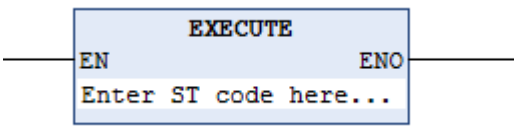
- Each set coil should be accompanied by a reset coil.

2.1.11 Execute Box

You can program in ST language by inserting an execute box in LD language.

If "Enter ST code here ..." is clicked, an input field using a multi-line ST will open.

■ Icon



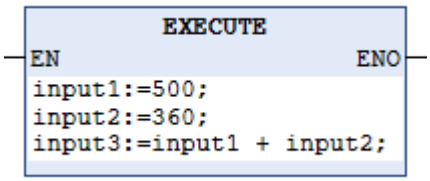
■ Input method

You can enter "Execute Box" by any of the following operations.

- Select **General→Execute** in the toolbox and drag it to the position of ◆ displayed next to the contact.
- Right-click on the network and select Insert "Execute Box" from the menu that appears
- Select **FBD/LD/IL>Execute Box** from the menu

■ Program example

When the EN condition is ON, the entered ST language code will be executed.



(MEMO)

3 Functions

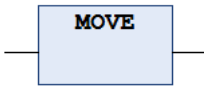
3.1 Basic Instructions

You can use basic instructions to assign the values of other variables to variables, specify addresses, and get sizes.

3.1.1 MOVE (Substitution)

This is a function that substitutes the value of a variable specified in the input for a variable specified in the output.

■ Icon



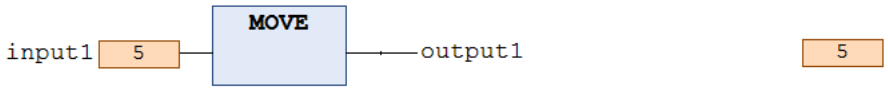
■ Parameter

Scope	Type	Description
Input	All	Specifies the variable of the substitution source.
Output	All	Specifies the variable of the substitution target.

■ Program example

This program is designed to substitute the value of input variable “input1” for the output variable “output1”.

LD program



ST program

```
output1 5 := MOVE(input1 5 );
```

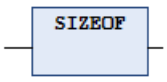
It is also possible to substitute the value using an operator (:=).

```
output1 5 := input1 5 ;
```

3.1.2 SIZEOF (Get the Size)

This is a function that outputs the size (number of bytes) of the input argument.

■ Icon



■ Parameter

Scope	Type	Description
Input	(Note 1)	Specifies the argument whose size is to be calculated.
Output	(Note 1)	Outputs the size of (1).

(Note 1) Usable data types

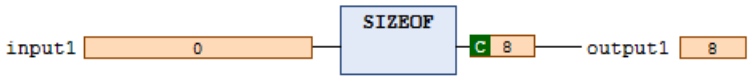
All standard data types

(BOOL, BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, REAL, LREAL, TIME, LTIME, DATE, TIME_OF_DAY, DATE_AND_TIME, STRING, WSTRING)

■ Program example

This program is designed to output the size of the ULINT type input variable “input1” to the UINT type output variable “output1”.

LD program



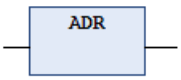
ST program

```
output1  8  := SIZEOF(input1  0 );
```

3.1.3 ADR (Get the Address)

This is a function that outputs the address of the variable.

■ Icon



■ Parameter

Scope	Type	Description
Input	(Note 1)	Input the variable from which to get the address.
Output	(Note 1)	Outputs the address (pointer) of the input variable.

(Note 1) Usable data types

All standard data types

(BOOL, BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, REAL, LREAL, TIME, LTIME, DATE, TIME_OF_DAY, DATE_AND_TIME, STRING, WSTRING)

■ Usable data type

All standard data types

(BOOL, BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, REAL, LREAL, TIME, LTIME, DATE, TIME_OF_DAY, DATE_AND_TIME, STRING, WSTRING)

■ Program example

This program is designed to output the address of the input variable “input1” to the output variable “output1”.

LD program

ST program

```
output1 16#F1D61884 := ADR(input1 0 );
```

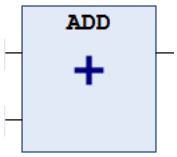
3.2 Arithmetic Operation Instructions

Arithmetic operation instructions can be used to perform calculation such as four arithmetic operations.

3.2.1 ADD (Addition)

This is a function that adds input arguments and outputs the sum.

■ Icon



■ Parameter

Scope	Type	Description
Input	(Note 1)	Specifies the variables to be added.
Output	(Note 1)	Outputs the sum of variables specified in the input.

(Note 1) Usable data type

BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE_AND_TIME

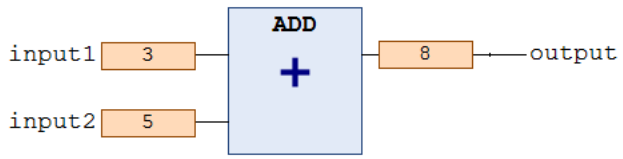
Time type data can be added in the following combinations.

- TIME + TIME = TIME
- TIME_OF_DAY + TIME = TIME_OF_DAY
- DATE_AND_TIME + TIME = DATE_AND_TIME

■ Program example

This program is designed to output the sum of input variables “input1” and “input2” to the output variable “output”.

LD program



ST program

It is possible to add the values using “+” operator.

```
output 8 := input1 3 + input2 5;
```

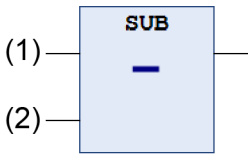
Info.

- If you want to increase input arguments in the LD program, right-click on the ADD function, and, on the displayed menu, select "Add Input".

3.2.2 SUB (Subtraction)

This is a function that subtracts input arguments and outputs the difference.

■ Icon



■ Parameter

Scope	Number	Type	Description
Input	(1), (2)	(Note 1)	Specifies the variables to be subtracted.
Output	-	(Note 1)	Outputs the value obtained by subtracting the input (2) from the input (1).

(Note 1) Usable data types

BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME

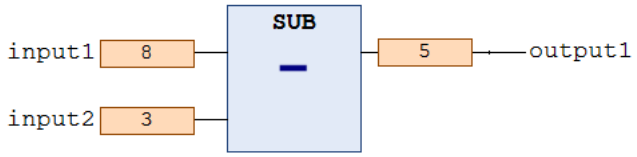
For time type data, subtraction can be performed in the following combinations. Note that negative time cannot be calculated.

- TIME - TIME = TIME
- DATE - DATE = TIME
- TOD - TIME = TOD
- TOD - TOD = TIME
- DT - TIME = DT
- DT - DT = TIME

■ Program example

This program is designed to output the difference between the input variables “input1” and “input2” to the output variable “output1”.

LD program



ST program

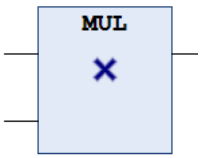
It is possible to subtract the values using “-” operator.

```
output1 5 := input1 8 - input2 3 ;
```

3.2.3 MUL (Multiplication)

This is a function that multiplies input arguments and outputs the product.

■ Icon



■ Parameter

Scope	Type	Description
Input	(Note 1)	Specifies the variables to be multiplied.
Output	(Note 1)	Outputs the product of variables specified in the input.

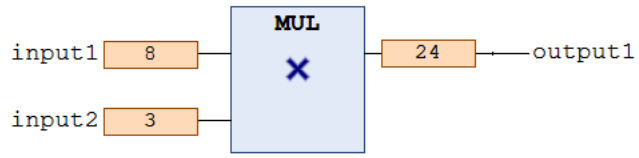
(Note 1) Usable data type

BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, REAL, LREAL, TIME

■ Program example

This program is designed to output the product of the input variables “input1” and “input2” to the output variable “output1”.

LD program



ST program

It is possible to multiply the values using “*” operator.

```
output1 24 := input1 8 * input2 3 ;
```

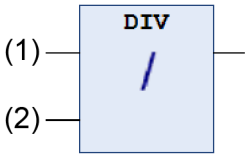
Info.

- If you want to increase input arguments in the LD program, right-click on the MUL function, and, on the displayed menu, select "Add Input".
- TIME type data cannot be multiplied by REAL type, LREAL type, or TIME type data.

3.2.4 DIV (Division)

This is a function that divides input arguments and outputs the quotient.

■ Icon



■ Parameter

Scope	No.	Type	Description
Input	(1), (2)	(Note 1)	Specifies the variables to be divided.
Output	-	(Note 1)	Outputs the quotient obtained by dividing the input (2) by the input (1).

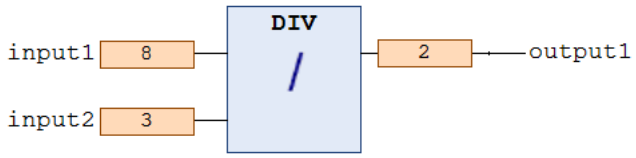
(Note 1) Usable data types

BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, REAL, LREAL, TIME

■ Program example

This program is designed to output the quotient of the INT type input variables “input1” and “input2” to the INT type output variable “output1”.

LD program



ST program

It is possible to divide the values using the division operator ("/") .

```
output1 2 := input1 8 / input2 3 ;
```

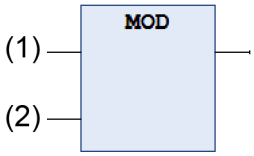
Info.

- TIME type variables can be divided by integer type variables.
- When a variable is divided by a DINT, LINT, REAL, or LREAL type variable, it can be checked if 0 is used in the calculation. (Refer to “Auto Check POU” in the “SMC Tool Introduction Guide”.)

3.2.5 MOD (Remainder)

This is a function that divides input arguments and outputs the remainder.

■ Icon



■ Parameter

Scope	No.	Type	Description
Input	(1), (2)	(Note 1)	Specifies the variables to be divided.
Output	-	(Note 1)	Outputs the remainder obtained by dividing the input (2) by the input (1).

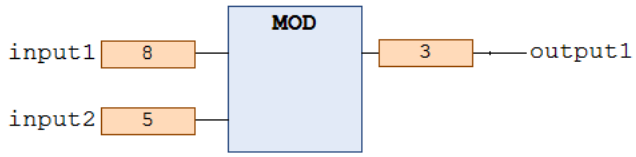
(Note 1) Usable data type

BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT

■ Program example

This program is designed to output the remainder obtained from dividing the INT type input variables “input1” and “input2” to the INT type output variable “output1”.

LD program



ST program

```
output1 3 := input1 8 MOD input2 5;
```

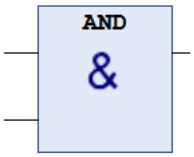
3.3 Boolean Operation Instructions

Boolean operation instructions can be used to perform bool operations such as logical AND or logical OR.

3.3.1 AND (Logical AND)

This is a function that outputs logical AND of the input arguments.

■ **Icon**



■ **Parameter**

Scope	Type	Description
Input	(Note 1)	Specifies the variables to be used to obtain logical AND.
Output	(Note 1)	Outputs the logical AND of the variables specified in the input.

(Note 1) Usable data type

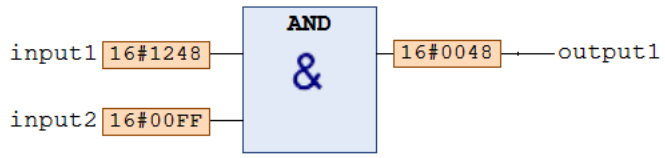
BOOL, BYTE, WORD, DWORD, LWORD

■ **Program example**

This program is designed to output the logical AND of the WORD type input variables “input1” and “input2” to the output variable “output1”.

The execution result is displayed in a hexadecimal number.

LD program



ST program

```
output1 16#0048 := input1 16#1248 AND input2 16#00FF;
```

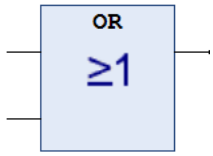
Info.

- If you want to increase input arguments in the LD program, right-click on the AND function, and, on the displayed menu, select "Add Input".

3.3.2 OR (Logical OR)

This is a function that outputs logical OR of the input arguments.

■ Icon



■ Parameter

Scope	Type	Description
Input	(Note 1)	Specifies the variables to be used to obtain logical OR.
Output	(Note 1)	Outputs the logical OR of the variables specified in the input.

(Note 1) Usable data type

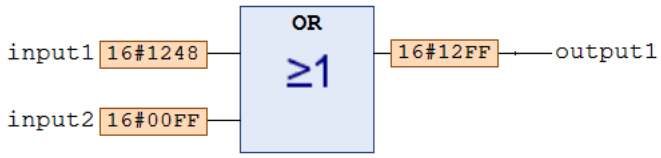
BOOL, BYTE, WORD, DWORD, LWORD

■ Program example

This program is designed to output the logical OR of the WORD type input variables “input1” and “input2” to the output variable “output1”.

The execution result is displayed in a hexadecimal number.

LD program



ST program

```
output1 16#12FF := input1 16#1248 OR input2 16#00FF;
```

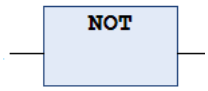
Info.

- If you want to increase input arguments in the LD program, right-click on the OR function, and, on the displayed menu, select "Add Input".

3.3.3 NOT (Negation)

This is a function that outputs the negation of the input argument.

■ **Icon**



■ **Parameter**

Scope	Type	Description
Input	(Note 1)	Specifies the variable to be used to obtain the negation.
Output	(Note 1)	Outputs the negation of the variable specified in the input.

(Note 1) Usable data type

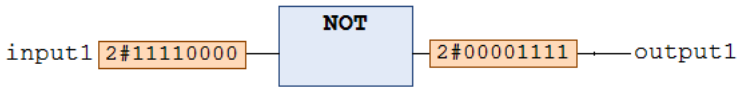
BOOL, BYTE, WORD, DWORD, LWORD

■ **Program example**

This program is designed to output the negation of the BYTE type input variable “input1” to the output variable “output1”.

The execution result is displayed in a binary number.

LD program



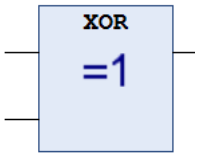
ST program

```
output1 2#00001111 := NOT input1 2#11110000;
```

3.3.4 XOR (Exclusive OR)

This is a function that outputs exclusive OR of the input arguments.

■ **Icon**



■ **Parameter**

Scope	Type	Description
Input	(Note 1)	Specifies the variables to be used to obtain exclusive OR.
Output	(Note 1)	Outputs the exclusive OR of the variables specified in the input.

Scope	Type	Description
		Outputs 0 if both input bits are 1 or 0. Outputs 1 if one of the two input bits is 1 and the other bit is 0.

(Note 1) Usable data type

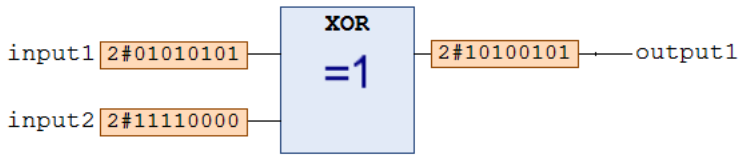
BOOL, BYTE, WORD, DWORD, LWORD

■ Program example

This program is designed to output the exclusive OR of the BYTE type input variables “input1” and “input2” to the output variable “output1”.

The execution result is displayed in a binary number.

LD program



ST program

```
output1 2#10100101 := input1 2#01010101 XOR input2 2#11110000;
```

3.3.5 AND_THEN (Logical AND)

This is a conditional AND evaluation function of the input operand.

■ Usable data types

BOOL, BIT

■ Program example

This program is designed to compare the value of the variable accessed by pwAddress (pointer) with wExpected if the pwAddress is not NULL and, if they are the same, substitute with the value of wNewValue.

As default values, “5” is stored in the variable “test1” accessed by pwAddress, “5” in wExpected, and “3” in wNewValue.

As an initial step, judgment is made whether pwAddress is NULL or not. Since it is not NULL, comparison is made between the value of “test1” and the value of wExpected as the next step. Since these two values are both “5”, TRUE is assigned. As a result, the value of wNewValue “3” is stored in the “test1” and the xFlag flag is set to TRUE.

ST program

[Declaration section]

```
VAR
    pwAddress   : POINTER TO WORD;
    wExpected   : WORD := 5;
    wNewValue   : WORD := 3;
    xFlag       : BOOL;
    testl       : WORD := 5;
END_VAR
```

[Implementation section]

```
pwAddress[16#F1D10BBE] := ADR(testl[16#0003]);

IF pwAddress[16#F1D10BBE] <> 0 AND_THEN pwAddress^[16#0003] = wExpected[16#0005] THEN
    pwAddress^[16#0003]:= wNewValue[16#0003];
    xFlag[TRUE] := TRUE;
ELSE
    xFlag[TRUE] := FALSE;
END_IF
```

Info.

- Expressions of other operands are executed only when the first operand is TRUE.
  Therefore, if no value is stored in pwAddress in the above example, the initial NULL judgment turns FALSE. As a result, no judgment is performed on operands after the AND_THEN operator.

3.3.6 OR_ELSE (Logical OR)

This is a conditional OR evaluation function of the input operand.

■ Usable data types

BOOL, BIT

■ Program example

16#000000FF is stored in the variable dw.
”dw.8” that represents bit 8 of dw is FALSE and “dw.1” that represents bit 1 is TRUE.
Therefore, the operation result flag bX is TRUE.
Note that the third input expression is not executed and bEver remains FALSE.

ST program

[Declaration section]

```
VAR
    bEver   : BOOL;
    bX      : BOOL;
    dw      : DWORD := 16#000000FF;
END_VAR
```

[Implementation section]

```
bEver[FALSE] := FALSE;
bX[TRUE] := dw[16#000000FF].8[FALSE] OR_ELSE dw[16#000000FF].1[TRUE] OR_ELSE dw[16#000000FF].1[TRUE] OR_ELSE (bEver[FALSE] := TRUE);
```

Info.

- In case of OR_ELSE, when one of the operands is evaluated TRUE, all other operator expressions are not evaluated.

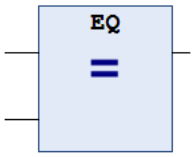
3.4 Comparison Operation Instructions

Comparison operation instructions can be used to compare two arguments.

3.4.1 EQ (“Equal” Comparison)

This is a function that compares two input arguments and determines if they are the same value.

■ **Icon**



■ **Parameter**

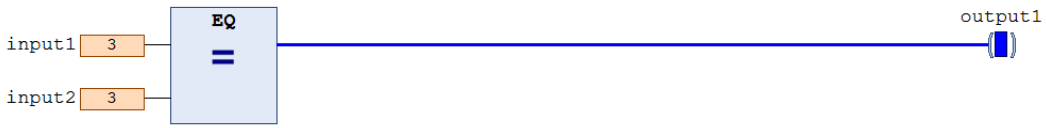
Scope	Type	Description
Input	All	Specifies the variables to be compared.
Output	BOOL	Outputs TRUE if the input variable values are the same. Outputs FALSE if they are different.

■ **Program example**

This program is designed to compare the input variables “input1” and “input2” and output the result to the output variable “output1”.

LD program

TRUE is output because the input variable values “input1” and “input2” are the same.



ST program

Use the operator (=) to compare the values.

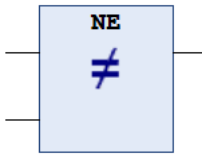
FALSE is output because the input variable values “input1” and “input2” are different.

```
output1 FALSE := (input1  3  = input2  5 );
```

3.4.2 NE (“Not Equal” Comparison)

This is a function that compares two input arguments and determines if they are not the same.

■ **Icon**



■ **Parameter**

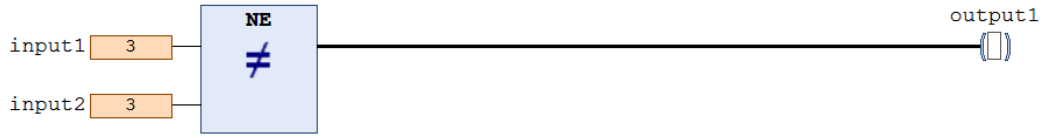
Scope	Type	Description
Input	All	Specifies the variables to be compared.
Output	BOOL	Outputs TRUE if the input variable values are different. Outputs FALSE if they are the same.

■ **Program example**

This program is designed to compare the input variables “input1” and “input2” and output the result to the output variable “output1”.

LD program

FALSE is output because the input variable values “input1” and “input2” are the same.



ST program

Use the operator (<>) to compare the values.

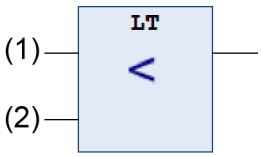
TRUE is output because the input variable values “input1” and “input2” are different.

```
output1 TRUE := input1 3 <> input2 5;
```

3.4.3 LT (“Less Than” Comparison)

This is a function that compares two input arguments and determines if the first argument is less than the second argument.

■ **Icon**



■ Parameter

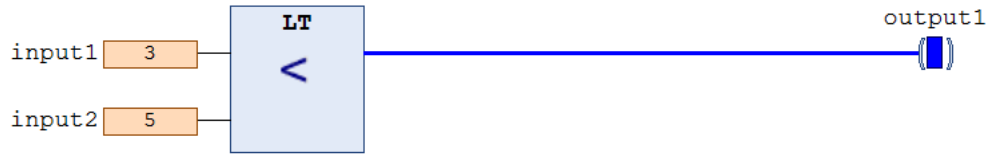
Scope	No.	Type	Description
Input	(1), (2)	All	Specifies the variables to be compared.
Output	-	BOOL	Outputs TRUE if the value of input (1) is less than the value of input (2). Otherwise, outputs FALSE.

■ Program example

This program is designed to compare the input variables “input1” and “input2” and output the result to the output variable “output1”.

LD program

TRUE is output because the input variable “input1” is less than the input variable “input2”.



ST program

Use the operator (<) to compare the values.

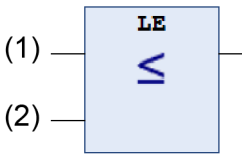
FALSE is output because the input variable “input1” is not less than the input variable “input2”.

```
output1 FALSE := input1 6 < input2 2;
```

3.4.4 LE (“Less Than or Equal” Comparison)

This is a function that compares two input arguments and determines if the first argument is less than or equal to the second argument.

■ Icon



■ Parameter

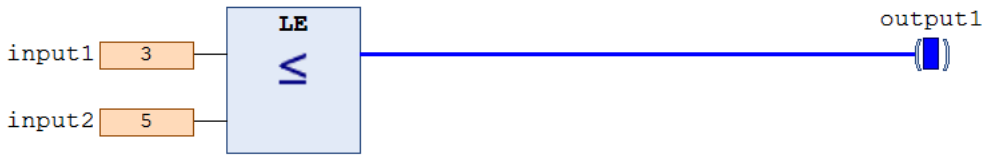
Scope	No.	Type	Description
Input	(1), (2)	All	Specifies the variables to be compared.
Output	-	BOOL	Outputs TRUE if the value of input (1) is less than or equal to the value of input (2). Otherwise, outputs FALSE.

■ Program example

This program is designed to compare the input variables “input1” and “input2” and output the result to the output variable “output1”.

LD program

TRUE is output because the input variable “input1” is less than or equal to the input variable “input2”.



ST program

Use the operator (<=) to compare the values.

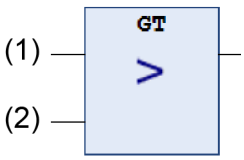
FALSE is output because the input variable “input1” is not less than or equal to the input variable “input2”.

```
output1 FALSE := input1 6 <= input2 2 ;
```

3.4.5 GT (“Greater Than” Comparison)

This is a function that compares two input arguments and determines if the first argument is greater than the second argument.

■ Icon



■ Parameter

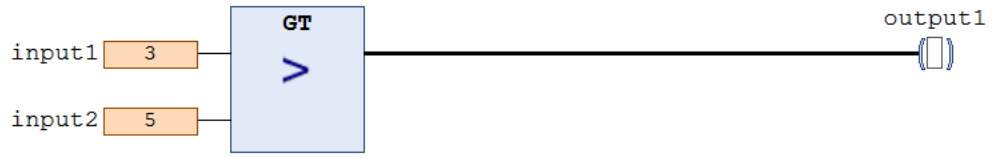
Scope	No.	Type	Description
Input	(1), (2)	All	Specifies the variables to be compared.
Output	-	BOOL	Outputs TRUE if the value of input (1) is greater than the value of input (2). Otherwise, outputs FALSE.

■ Program example

This program is designed to compare the input variables “input1” and “input2” and output the result to the output variable “output1”.

LD program

FALSE is output because the input variable “input1” is not greater than the input variable “input2”.



ST program

Use the operator (>) to compare the values.

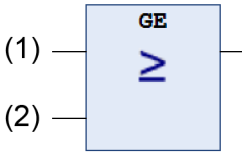
TRUE is output because the input variable “input1” is greater than the input variable “input2”.

```
output1 TRUE := input1 6 > input2 2;
```

3.4.6 GE (“Greater Than Or Equal” Comparison)

This is a function that compares two input arguments and determines if the first argument is greater than or equal to the second argument.

■ Icon



■ Parameter

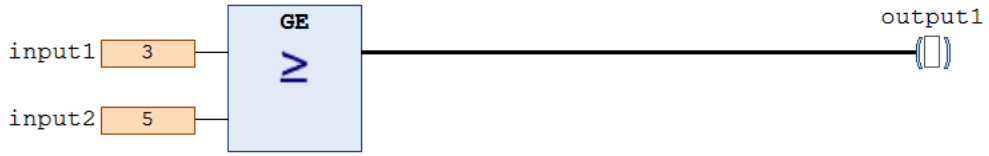
Scope	No.	Type	Description
Input	(1), (2)	All	Specifies the variables to be compared.
Output	-	BOOL	Outputs TRUE if the value of input (1) is greater than or equal to the value of input (2). Otherwise, outputs FALSE.

■ Program example

This program is designed to compare the input variables “input1” and “input2” and output the result to the output variable “output1”.

LD program

FALSE is output because the input variable “input1” is not greater than or equal to the input variable “input2”.



ST program

Use the operator (>=) to compare the values.

TRUE is output because the input variable “input1” is greater than or equal to the input variable “input2”.

```
output1 TRUE := input1 6 >= input2 2 ;
```

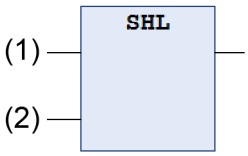
3.5 Bit Shift Instructions

Bit shift instructions can be used to perform bit shift operation on input arguments.

3.5.1 SHL (Shift Left)

This is a function that shifts the input argument to the left by the specified number of bits and outputs the shifted value. “0” is inserted from the least significant bit up to the bit position shifted by the shift quantity.

■ Icon



■ Parameter

Scope	No.	Type	Description
Input	(1)	(Note 1)	Specifies the variable on which bit shift is performed.
	(2)	(Note 1)	Specifies the number of times bit shift is performed (shift quantity).
Output	-	(Note 1)	Outputs the value bit shifted to the left from the value of input (1) by the quantity specified in the input (2).

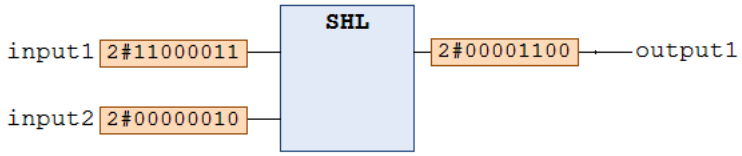
(Note 1) Usable data type

BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT

■ Program example

This program is designed to output the value that is shifted to the left from the value (2#11000011) of input variable “input1” by the number of bits (2 bits) specified in “input2” to the output variable “output1”.

LD program



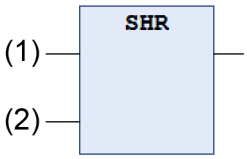
ST program

```
output1 2#00001100 := SHL(input1 2#11000011, input2 2#00000010);
```

3.5.2 SHR (Shift Right)

This is a function that shifts the input argument to the right by the specified number of bits and outputs the shifted value. “0” is inserted from the most significant bit up to the bit position shifted by the shift quantity.

■ Icon



■ Parameter

Scope	No.	Type	Description
Input	(1)	(Note 1)	Specifies the variable on which bit shift is performed.
	(2)	(Note 1)	Specifies the number of times bit shift is performed (shift quantity).
Output	-	(Note 1)	Outputs the value bit shifted to the right from the value of input (1) by the quantity specified in the input (2).

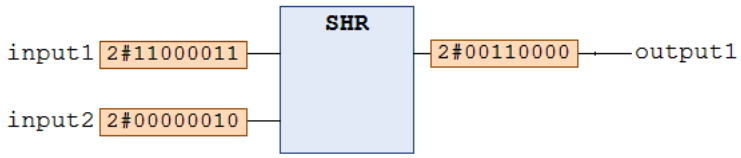
(Note 1) Usable data type

BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT

■ Program example

This program is designed to output the value that is shifted to the right from the value (2#11000011) of input variable “input1” by the number of bits (2 bits) specified in “input2” to the output variable “output1”.

LD program



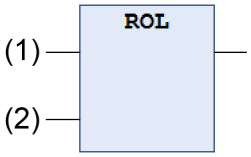
ST program

```
output1 2#00110000 := SHR(input1 2#11000011, input2 2#00000010);
```

3.5.3 ROL (Rotate Left)

This is a function that shifts the input argument to the left by the specified number of bits and outputs the shifted value. The bit value that has overflowed the most significant bit when the bit is shifted is inserted into the data starting from the least significant bit up to the bit position shifted by the shift quantity.

■ Icon



■ Parameter

Scope	No.	Type	Description
Input	(1)	(Note 1)	Specifies the variable on which bit shift is performed.
	(2)	(Note 1)	Specifies the number of times bit shift is performed (shift quantity).
Output	-	(Note 1)	Outputs the value rotated and shifted to the left from the value of input (1) by the quantity specified in the input (2).

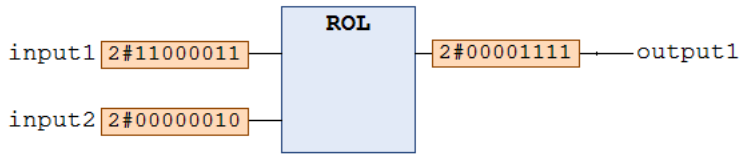
(Note 1) Usable data type

BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT

■ Program example

This program is designed to output the value that is rotated and shifted to the left from the value (2#11000011) of input variable “input1” by the number of bits (2 bits) specified in “input2” to the output variable “output1”.

LD program



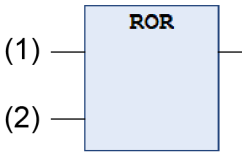
ST program

```
output1 2#00001111 := ROL(input1 2#11000011, input2 2#00000010);
```

3.5.4 ROR (Rotate Right)

This is a function that shifts the input argument to the right by the specified number of bits and outputs the shifted value. The bit value that has overflowed the least significant bit when the bit is shifted is inserted into the data starting from the most significant bit up to the bit position shifted by the shift quantity.

■ Icon



■ Parameter

Scope	No.	Type	Description
Input	(1)	(Note 1)	Specifies the variable on which bit shift is performed.
	(2)	(Note 1)	Specifies the number of times bit shift is performed (shift quantity).
Output	-	(Note 1)	Outputs the value rotated and shifted to the right from the value of input (1) by the quantity specified in the input (2).

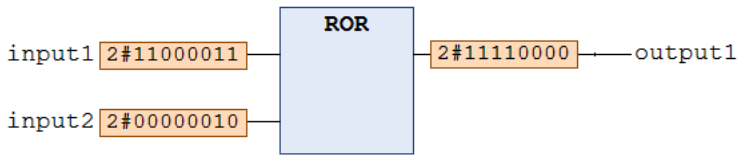
(Note 1) Usable data type

BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT

■ Program example

This program is designed to output the value that is rotated and shifted to the right from the value (2#11000011) of input variable “input1” by the number of bits (2 bits) specified in “input2” to the output variable “output1”.

LD program



ST program

```
output1 2#11110000 := ROR(input1 2#11000011, input2 2#00000010);
```

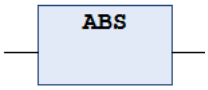
3.6 Numerical Operation Instructions

Numerical operation instructions can be used to perform various numerical calculations.

3.6.1 ABS (Absolute Value)

This is a function that outputs the absolute value of the input argument.

■ Icon



■ Parameter

Scope	Type	Description
Input	(Note 1)	Specifies the value from which to obtain the absolute value.
Output	(Note 1)	Outputs the absolute value of the input argument.

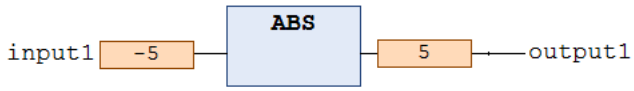
(Note 1) Usable data type

BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, REAL, LREAL

■ Program example

This program is designed to output the absolute value of the input variable “input1” to the output variable “output1”.

LD program



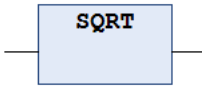
ST program

```
output1  5  := ABS(input1  -5  );
```

3.6.2 SQRT (Square Root)

This is a function that outputs the square root (√) of the input argument.

■ Icon



■ Parameter

Scope	Type	Description
Input	(Note 1)	Specifies the value from which to obtain the square root.
Output	(Note 2)	Outputs the square root of the input argument.

(Note 1) Usable data type

BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, REAL, LREAL

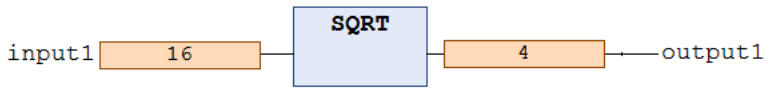
(Note 2) Usable data type

REAL (if the input is REAL), LREAL

■ Program example

This program is designed to output the square root of the input variable “input1” to the output variable “output1”.

LD program



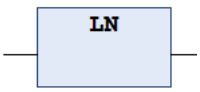
ST program

```
output1 4 := SQRT(input1 16);
```

3.6.3 LN (Natural Logarithm)

This is a function that outputs the natural logarithm ($\log_e X$) of the input argument (X).

■ Icon



■ Parameter

Scope	Type	Description
Input	(Note 1)	Specifies the value from which to obtain the natural logarithm.
Output	(Note 2)	Outputs the natural logarithm of the input argument.

(Note 1) Usable data type

BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, REAL, LREAL

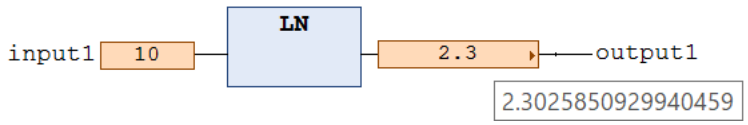
(Note 2) Usable data type

REAL (if the input is REAL), LREAL

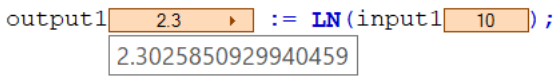
■ Program example

This program is designed to output the natural logarithm ($\log_e 10$) of the input variable “input1” (10) to the output variable “output1”.

LD program



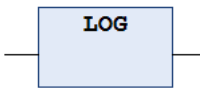
ST program



3.6.4 LOG (Common Logarithm)

This is a function that outputs the common logarithm ($\log_{10} X$) of the input argument (X).

■ Icon



■ Parameter

Scope	Type	Description
Input	(Note 1)	Specifies the value from which to obtain the common logarithm.
Output	(Note 2)	Outputs the common logarithm of the input argument.

(Note 1) Usable data type

BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, REAL, LREAL

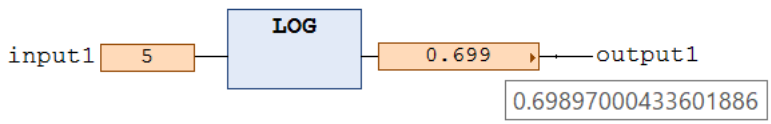
(Note 2) Usable data type

REAL (if the input is REAL), LREAL

■ Program example

This program is designed to output the common logarithm ($\log_{10} 5$) of the input variable “input1” (5) to the output variable “output1”.

LD program



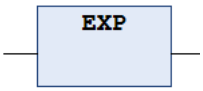
ST program

```
output1 0.699 := LOG(input1 5);
        0.69897000433601886
```

3.6.5 EXP (Natural Exponent)

This is a function that outputs the natural exponent (e^X) of the input argument (X).

■ Icon



■ Parameter

Scope	Type	Description
Input	(Note 1)	Specifies the value from which to obtain the natural exponent.
Output	(Note 2)	Outputs the natural exponent of the input argument.

(Note 1) Usable data type

BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, REAL, LREAL

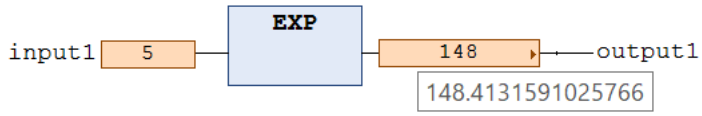
(Note 2) Usable data type

REAL (if the input is REAL), LREAL

■ Program example

This program is designed to output the natural exponent of the input variable “input1” to the output variable “output1”.

LD program



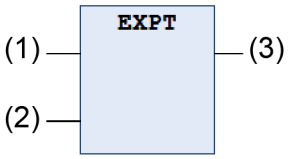
ST program

```
output1 148 := EXP(input1 5);
        148.4131591025766
```

3.6.6 EXPT (Exponentiation)

This is a function that outputs the exponentiation (a^n) of the input arguments (a, n).

■ Icon



■ Parameter

Scope	No.	Type	Description
Input	(1)	(Note 1)	Inputs the base of exponentiation.
	(2)	(Note 1)	Inputs the exponent of exponentiation.
Output	(3)	(Note 2)	Outputs the exponentiation obtained from the input arguments. Outputs a^n in the following case. Input (1): a Input (2): n

(Note 1) Usable data type

BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, REAL, LREAL

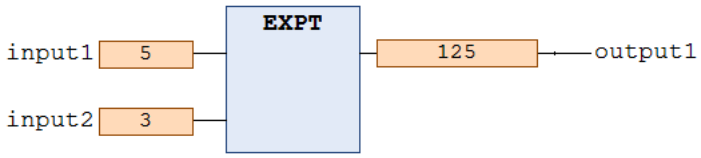
(Note 2) Usable data type

REAL (if the input is REAL), LREAL

■ Program example

This program is designed to output the exponentiation (5^3 = 125) obtained from the input variables “input1” and “input2” to the output variable “output1”.

LD program



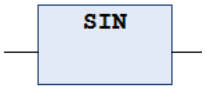
ST program

```
output1 125 := EXPT(input1 5, input2 3);
```

3.6.7 SIN (Trigonometric Function Sine)

This is a function that outputs the value of the trigonometric function sine. The unit of the input argument is radian.

■ Icon



■ Parameter

Scope	Type	Description
Input	(Note 1)	Specifies the value (unit: radian) from which to obtain the trigonometric function sine.
Output	(Note 2)	Outputs the value of sine of the input argument.

(Note 1) Usable data type

BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, REAL, LREAL

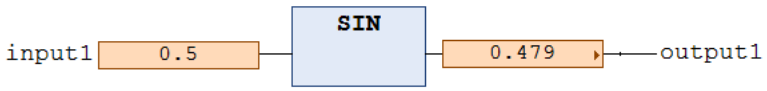
(Note 2) Usable data type

REAL (if the input is REAL), LREAL

■ Program example

This program is designed to output the value of the trigonometric function sine obtained from the input variable “input1” to the output variable ”output1”.

LD program



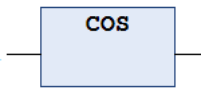
ST program

```
output1 0.479 := SIN(input1 0.5);
```

3.6.8 COS (Trigonometric Function Cosine)

This is a function that outputs the value of the trigonometric function cosine. The unit of the input argument is radian.

■ Icon



■ Parameter

Scope	Type	Description
Input	(Note 1)	Specifies the value (unit: radian) from which to obtain the trigonometric function cosine.
Output	(Note 2)	Outputs the value of cosine of the input argument.

(Note 1) Usable data type

BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, REAL, LREAL

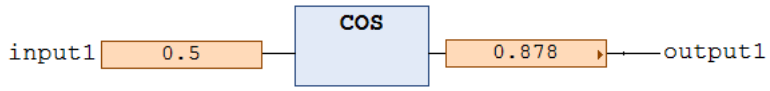
(Note 2) Usable data type

REAL (if the input is REAL), LREAL

■ Program example

This program is designed to output the value of the trigonometric function cosine obtained from the input variable “input1” to the output variable ”output1”.

LD program



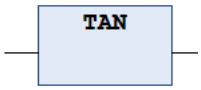
ST program

```
output1 0.878 := COS(input1 0.5);
```

3.6.9 TAN (Trigonometric Function Tangent)

This is a function that outputs the value of the trigonometric function tangent. The unit of the input argument is radian.

■ Icon



■ Parameter

Scope	Type	Description
Input	(Note 1)	Specifies the value (unit: radian) from which to obtain the trigonometric function tangent.
Output	(Note 2)	Outputs the value of tangent of the input argument.

(Note 1) Usable data type

BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, REAL, LREAL

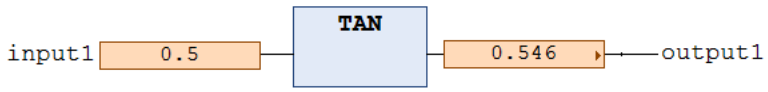
(Note 2) Usable data type

REAL (if the input is REAL), LREAL

■ Program example

This program is designed to output the value of the trigonometric function tangent obtained from the input variable “input1” to the output variable ”output1”.

LD program



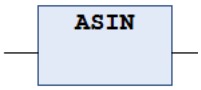
ST program

```
output1 0.546 := TAN(input1 0.5);
```

3.6.10 ASIN (Trigonometric Function Arc Sine)

This is a function that outputs the value of the trigonometric function arc sine. The unit of the input argument is radian.

■ Icon



■ Parameter

Scope	Type	Description
Input	(Note 1)	Specifies the value (unit: radian) from which to obtain the trigonometric function arc sine.
Output	(Note 2)	Outputs the value of arc sine of the input argument.

(Note 1) Usable data type

BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, REAL, LREAL

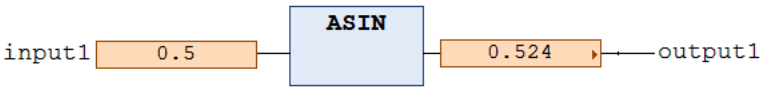
(Note 2) Usable data type

REAL (if the input is REAL), LREAL

■ Program example

This program is designed to output the value of the trigonometric function arc sine obtained from the input variable “input1” to the output variable ”output1”.

LD program



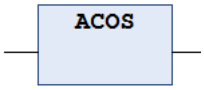
ST program

```
output1 0.524 := ASIN(input1 0.5);
```

3.6.11 ACOS (Trigonometric Function Arc Cosine)

This is a function that outputs the value of the trigonometric function arc cosine. The unit of the input argument is radian.

■ Icon



■ Parameter

Scope	Type	Description
Input	(Note 1)	Specifies the value (unit: radian) from which to obtain the trigonometric function arc cosine.
Output	(Note 2)	Outputs the value of arc cosine of the input argument.

(Note 1) Usable data type

BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, REAL, LREAL

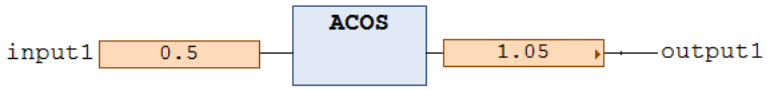
(Note 2) Usable data type

REAL (if the input is REAL), LREAL

■ Program example

This program is designed to output the value of the trigonometric function arc cosine obtained from the input variable “input1” to the output variable ”output1”.

LD program



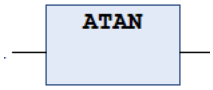
ST program

```
output1 1.05 := ACOS(input1 0.5);
```

3.6.12 ATAN (Trigonometric Function Arc Tangent)

This is a function that outputs the value of the trigonometric function arc tangent. The unit of the input argument is radian.

■ Icon



■ Parameter

Scope	Type	Description
Input	(Note 1)	Specifies the value (unit: radian) from which to obtain the trigonometric function arc tangent.
Output	(Note 2)	Outputs the value of arc tangent of the input argument.

(Note 1) Usable data type

BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, REAL, LREAL

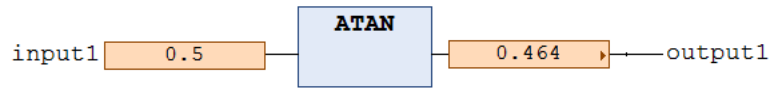
(Note 2) Usable data type

REAL (if the input is REAL), LREAL

■ Program example

This program is designed to output the value of the trigonometric function arc tangent obtained from the input variable “input1” to the output variable ”output1”.

LD program



ST program

```
output1 0.464 := ATAN(input1 0.5);
```

3.6.13 Triangular function operator constant

GM Programmer allows the use of the following constants.

Name	Value	Type	Description
SMC_PI	3.14159265358979	LREAL	Circumference ratio
SMC_FACTOR_DEG_TO_RAD	(SMC_PI/180)	LREAL	Convert angle (deg) to angle (rad)
SMC_FACTOR_RAD_TO_DEG	(180/SMC_PI)	LREAL	Convert angle (rad) to angle (deg)

3.7 Data Type Conversion Instructions

Data type conversion instructions can be used to convert the data type of a variable.

3.7.1 Type 1_TO_Type 2 (Type 1>Type 2 Conversion)

This is a function that converts the data type of the input argument Type 1 to another data type Type 2. Conversion from a larger size data type to a smaller size data type is not performed automatically. It is necessary to convert the data type using this instruction.

Parameter

Value	BOOL/BYTE/WORD/DWORD/LWORD/SINT/USINT INT/UINT/DINT/UDINT/LINT/ULINT/REAL/LREAL
Time	TIME/LTIME/TIME_OF_DAY/DATE/DATE_AND_TIME
Character string	STRING/WSTRING

Input ("Type 1")

Specifies the variable required to be converted

Output ("Type 2")

Outputs the converted variable

■ **Numerical value to numerical value type conversion**

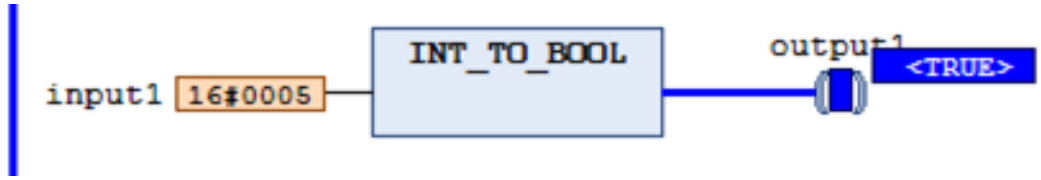
Main conversion examples

Numerical value ⇒ Numerical value	Conversion example		Description
	Input	Output	
INT_TO_BOOL	5	TRUE	If other than 0, outputs TRUE.
UINT_TO_SINT	300(16#012C)	44(16#2C)	Outputs lower eight bits out of the 16 bits of UINT.
REAL_TO_INT	3.5	4	Outputs data after rounding decimals to the nearest whole number.

■ **Program example**

INT_TO_BOOL

LD program

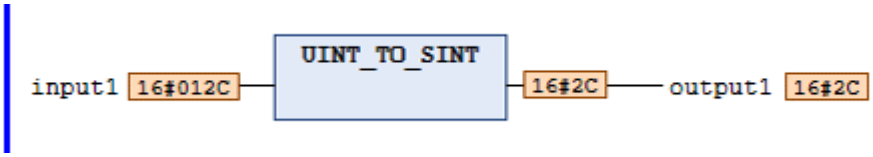


ST program

```
output1 TRUE :=INT_TO_BOOL(input1 16#0005 );
```

UINT_TO_SINT

LD program

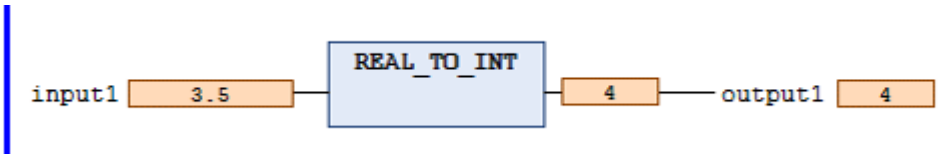


ST program

```
output1 16#2C :=UINT_TO_SINT(input1 16#012C );
```

REAL_TO_INT

LD program



ST program

```
output1 4 :=REAL_TO_INT(input1 3.5 );
```

■ Numerical value to time / Time to numerical value type conversion

Main conversion examples

Numerical value ⇒ Time	Conversion example		Description
	Input	Output	
UDINT_TO_TIME	16#098D 11B2	T#1d20h30m40s50ms	Outputs the TIME constant in the UDINT type.
UDINT_TO_TOD	16#0466 B774	TOD#20:30:40.500	Outputs the TOD constant in the UDINT type.
UDINT_TO_DATE	16#386D 4380	D#2000-1-1	Outputs the DATE constant in the UDINT type.
UDINT_TO_DT	16#386E 63F0	DT#2000-1-1-20:30:40	Outputs the DT constant in the UDINT type.
ULINT_TO_LTIME	16#0000 91BC CB43 3B26	LTIME#1d20h30m40s50ms60us70ns	Outputs the LTIME constant in the LTIME type.

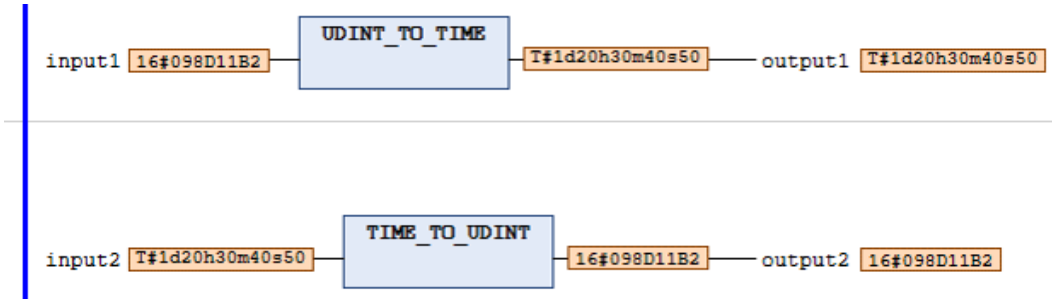
Time ⇒ Numerical value	Conversion example		Description
	Input	Output	
TIME_TO_UDINT	T#1d20h30m40s50ms	16#098D 11B2	Outputs the milliseconds from 0d0h0m0s.
TOD_TO_UDINT	TOD#20:30:40.500	16#0466 B774	Outputs the milliseconds from 00:00:00.

Time ⇒ Numerical value	Conversion example		Description
	Input	Output	
DATE_TO_UDINT	D#2000-1-1	16#386D 4380	Outputs the seconds from 1970-1-1.
DT_TO_UDINT	DT#2000-1-1-20:30:40	16#386E 63F0	Outputs the seconds from 1970-1-1-0:0:0.
LTIME_TO_ULINT	LTIME#1d20h30m40s50ms60us70ns	16#0000 91BC CB43 3B26	Outputs the nanoseconds from 0d0h0m0s0ms0us.

■ Program example

UDINT_TO_TIME/TIME_TO_UDINT

LD program

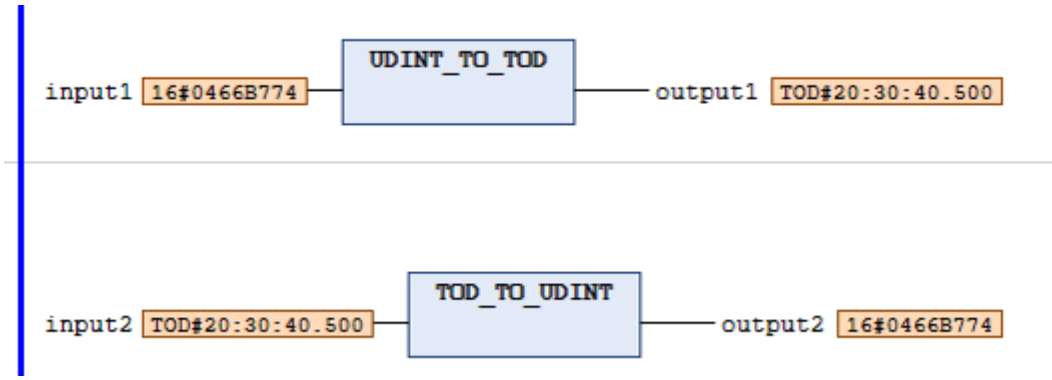


ST program

```
output1 T#1d20h30m40s50ms :=UDINT_TO_TIME(input1 16#098D11B2 );
output2 16#098D11B2 :=TIME_TO_UDINT(input2 T#1d20h30m40s50ms );
```

UDINT_TO_TOD/TOD_TO_UDINT

LD program

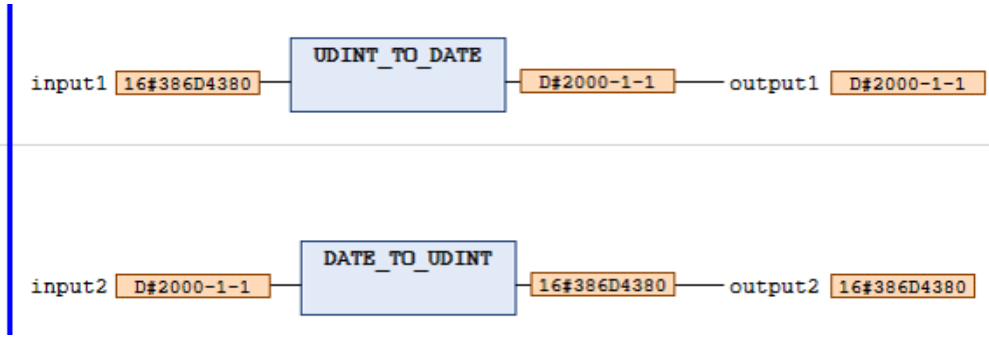


ST program

```
output1 TOD#20:30:40.500 :=UDINT_TO_TOD(input1 16#0466B774 );
output2 16#0466B774 :=TOD_TO_UDINT(input2 TOD#20:30:40.500 );
```

UDINT_TO_DATE/DATE_TO_UDINT

LD program

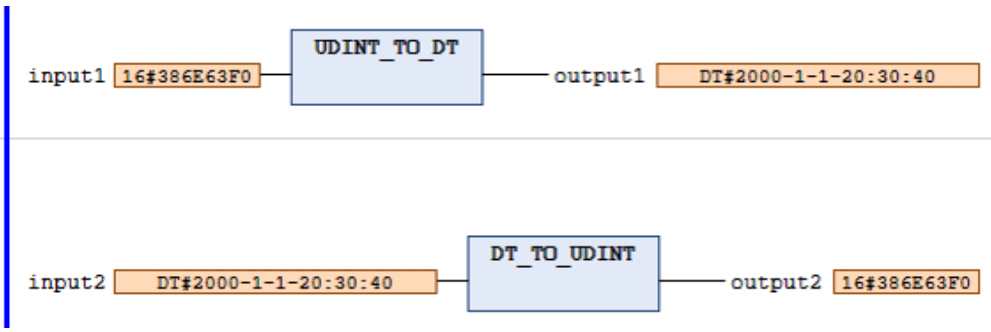


ST program

```
output1 D#2000-1-1 :=UDINT_TO_DATE(input1 16#386D4380 );
output2 16#386D4380 :=DATE_TO_UDINT(input2 D#2000-1-1 );
```

UDINT_TO_DT/DT_TO_UDINT

LD program

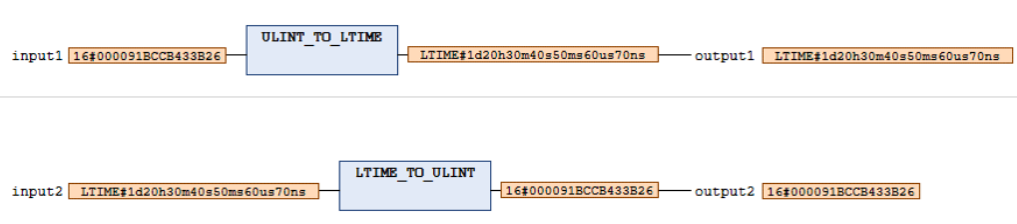


ST program

```
output1 DT#2000-1-1-20:30:40 :=UDINT_TO_DT(input1 16#386E63F0 );
output2 16#386E63F0 :=DT_TO_UDINT(input2 DT#2000-1-1-20:30:40 );
```

ULINT_TO_LTIME/LTIME_TO_ULINT

LD program



ST program

```
output1 LTIME#1d20h30m40s50ms60us70ns :=ULINT_TO_LTIME(input1 16#000091BCCB433B26 );
output2 16#000091BCCB433B26 :=LTIME_TO_ULINT(input2 LTIME#1d20h30m40s50ms60us70ns );
```

■ Numerical value to character string / Character string to numerical value type conversion

Main conversion examples

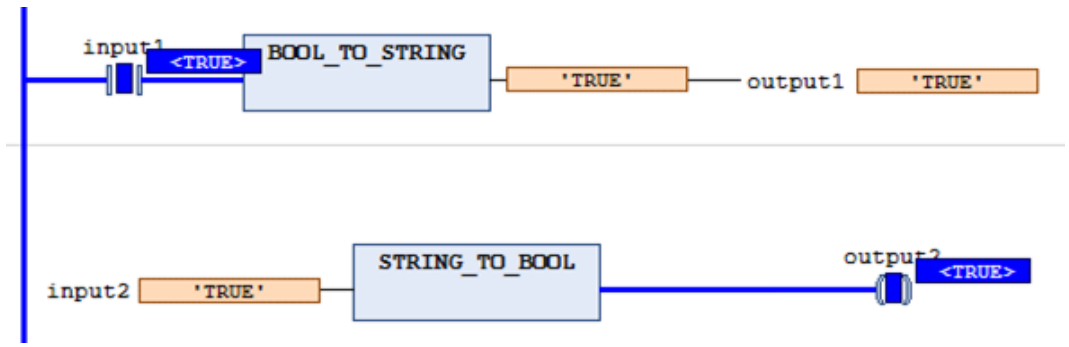
Numerical value ⇒ Character string	Conversion example		Description
	Input	Output	
BOOL_TO_STRING	TRUE	'TRUE'	Outputs 'TRUE' converted from TRUE / Outputs 'FALSE' converted from FALSE.
WORD_TO_STRING	16#3039(10#12345)	'12345'	Outputs the character string '12345' converted from the input value.
INT_TO_WSTRING	16#3039(10#12345)	"12345"	Outputs the character string "12345" converted from the input value.

Character string ⇒ Numerical value	Conversion example		Description
	Input	Output	
STRING_TO_BOOL	'TRUE'	TRUE	Outputs TRUE only when the character string 'TRUE' is input.
STRING_TO_WORD	'12345'	16#3039(10#12345)	Outputs a numerical value converted from the character string of the numerical value.
WSTRING_TO_INT	"12345"	16#3039(10#12345)	Outputs a numerical value converted from the character string of the numerical value.

■ Program example

BOOL_TO_STRING/STRING_TO_BOOL

LD program

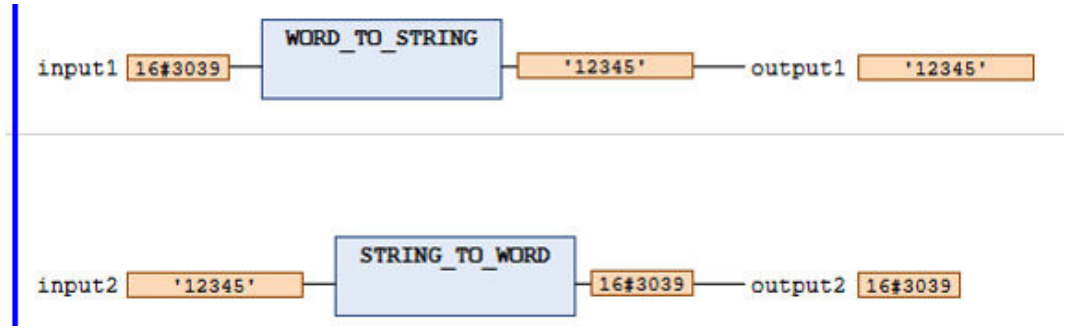


ST program

```
output1 'TRUE' :=BOOL_TO_STRING(input1 TRUE );
output2 TRUE :=STRING_TO_BOOL(input2 'TRUE' );
```

WORD_TO_STRING/STRING_TO_WORD

LD program

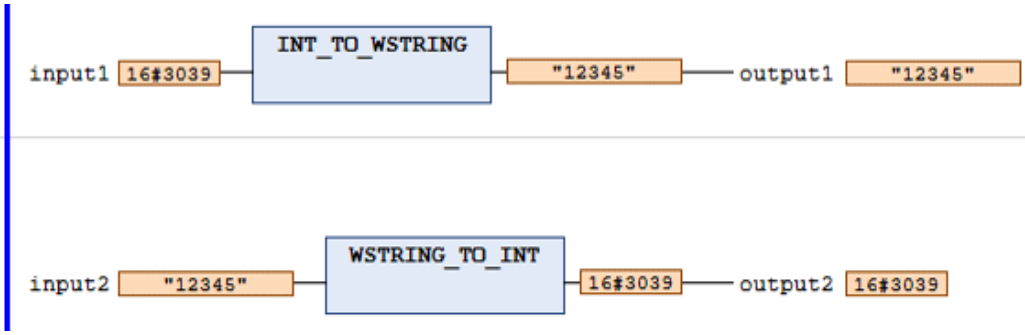


ST program

```
output1 '12345' :=WORD_TO_STRING(input1 16#3039 );
output2 16#3039 :=STRING_TO_WORD(input2 '12345' );
```

INT_TO_WSTRING/WSTRING_TO_INT

LD program



ST program

```
output1 "12345" :=INT_TO_WSTRING(input1 16#3039 );
output2 16#3039 :=WSTRING_TO_INT(input2 "12345" );
```

■ Time to character string / Character string to time type conversion

Main conversion examples

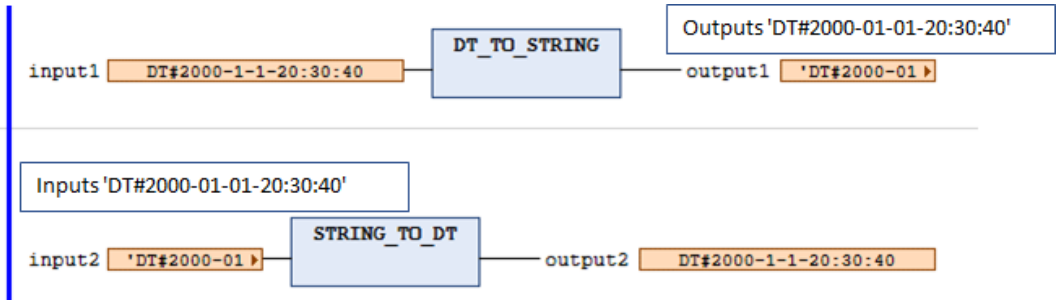
Time ⇒ Character string	Conversion example		Description
	Input	Output	
DT_TO_STRING	DT#2000-1-1-20:30:40	'DT#2000-01-01-20:30:40'	Outputs the DT constant in the STRING type.
TOD_TO_WSTRING	TOD#20:30:40.500	"TOD#20:30:40.500"	Outputs the TOD constant in the WSTRING type.

Character string ⇒ Time	Conversion example		Description
	Input	Output	
STRING_TO_DT	'DT#2000-01-01-20:30:40'	DT#2000-1-1-20:30:40	Outputs time converted from the character string of the time.
WSTRING_TO_TOD	"TOD#20:30:40.500"	TOD#20:30:40.500	Outputs time converted from the character string of the time.

■ **Program example**

DT_TO_STRING/STRING_TO_DT

LD program

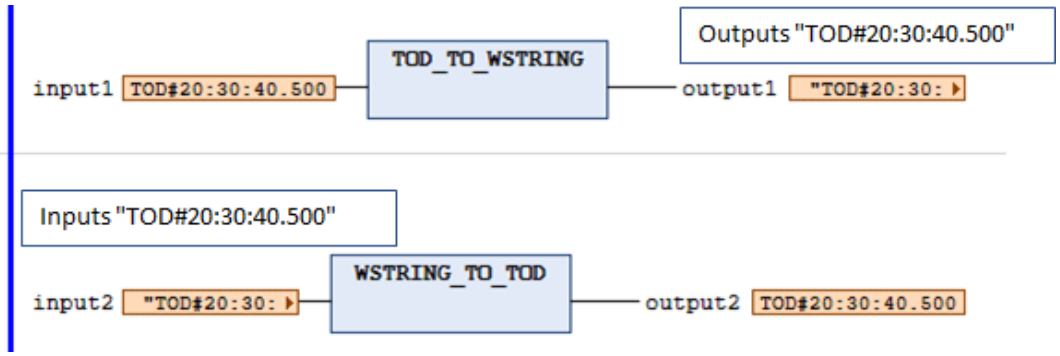


ST program

```
output1 'DT#2000-01 ▸ :=DT_TO_STRING(input1 DT#2000-1-1-20:30:40 );
output2 DT#2000-1-1-20:30:40 :=STRING_TO_DT(input2 'DT#2000-01 ▸ );
```

TOD_TO_WSTRING/WSTRING_TO_TOD

LD program



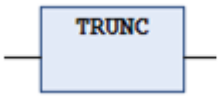
ST program

```
output1 "TOD#20:30: ▸ :=TOD_TO_WSTRING(input1 TOD#20:30:40.500 );
output2 TOD#20:30:40.500 :=WSTRING_TO_TOD(input2 "TOD#20:30: ▸ );
```

3.7.2 TRUNC (Real Number to DINT Conversion)

This is a function that converts a real number type input to a DINT type.

■ Icon



■ Parameter

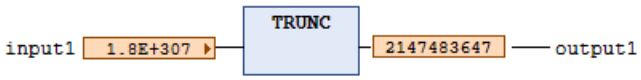
Scope	Type	Description
Input	REAL, LREAL	Real number type value
Output	DINT	Outputs the value converted to the DINT type from the input argument.

■ Program example

This program is designed to convert the LREAL type input variable “input1” to the DINT type output variable “output1” and output the converted data.

Input1 := 1.7976931348623157E+307;

LD program



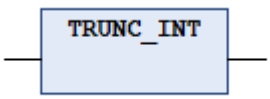
ST program

```
output1 2147483647 := TRUNC(input1 1.8E+307 );
```

3.7.3 TRUNC_INT (Real Number to INT Conversion)

This is a function that converts a real number type input to an INT type.

■ Icon



■ Parameter

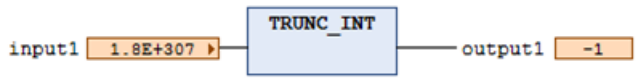
Scope	Type	Description
Input	REAL, LREAL	Real number type value
Output	INT	Outputs the value converted to the INT type from the input argument.

■ Program example

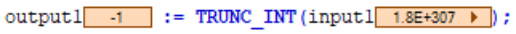
This program is designed to convert the LREAL type input variable “input1” to the INT type output variable “output1” and output the converted data.

Input1 := 1.7976931348623157E+307;

LD program



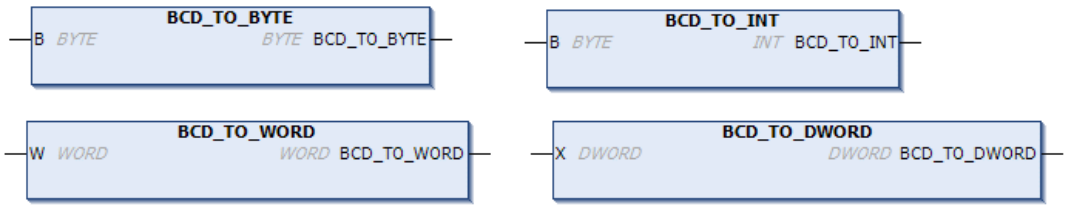
ST program



3.7.4 BCD_TO_** (BCD to Binary Conversion)

This is a function that converts an input in BCD format to a binary code of BYTE type, INT type, WORD type, or DWORD type.

■ Icon



■ Parameter

BCD_TO_BYTE

Scope	Name	Type	Description
Input	B	BYTE	The BCD code value to be converted
Output	BCD_TO_BYTE	BYTE	Outputs the value converted to a binary code from the input argument.

BCD_TO_INT

Scope	Name	Type	Description
Input	B	BYTE	The BCD code value to be converted
Output	BCD_TO_INT	INT	Outputs the value converted to a binary code from the input argument. Outputs 10#-1 when a value outside the effective range is input to the input B.

BCD_TO_WORD

Scope	Name	Type	Description
Input	W	WORD	The BCD code value to be converted
Output	BCD_TO_WORD	WORD	Outputs the value converted to a binary code from the input argument.

BCD_TO_DWORD

Scope	Name	Type	Description
Input	X	DWORD	The BCD code value to be converted
Output	BCD_TO_DWORD	DWORD	Outputs the value converted to a binary code from the input argument.

■ Program example

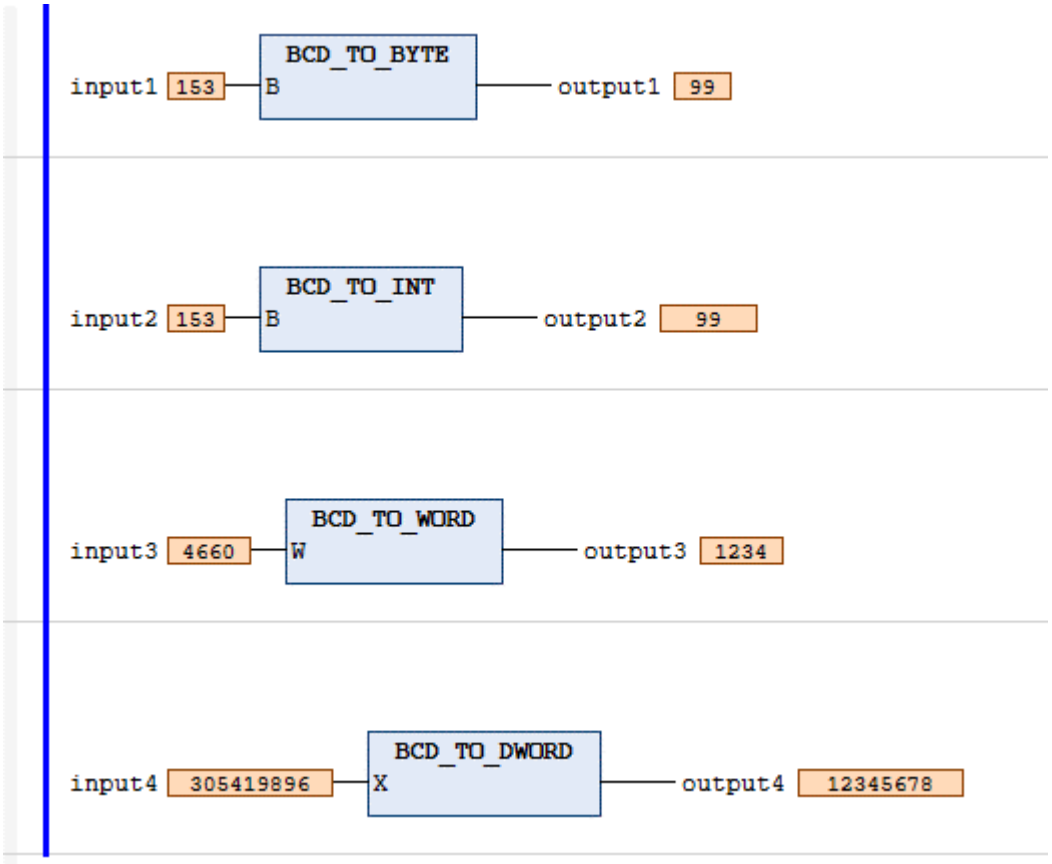
This program is designed to convert the BYTE type input variable “input1” to the BYTE type output variable “output1” and output the converted data.

This program is designed to convert the BYTE type input variable “input2” to the INT type output variable “output2” and output the converted data.

This program is designed to convert the WORD type input variable “input3” to the WORD type output variable “output3” and output the converted data.

This program is designed to convert the DWORD type input variable “input4” to the DWORD type output variable “output4” and output the converted data.

LD program



ST program

```
output1 99 := BCD_TO_BYTE(input1 153 );

output2 99 := BCD_TO_INT(input2 153 );

output3 1234 := BCD_TO_WORD(input3 4660 );

output4 12345678 := BCD_TO_DWORD(input4 305419896 );
```

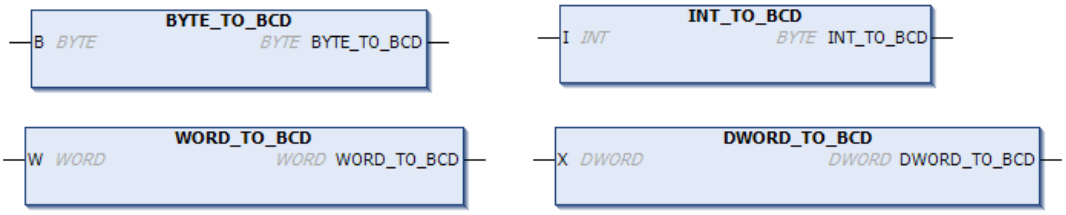
Note

- Do not input a value that is not a BCD array (including A, B, C, D, E, or F in hexadecimal notation).

3.7.5 **_TO_BCD (Binary to BCD Conversion)

This is a function that converts a binary code input of BYTE type, INT type, WORD type, or DWORD type to a BCD format value.

■ Icon



■ Parameter

BYTE_TO_BCD

Scope	Name	Type	Description
Input	B	BYTE	The binary code value to be converted. Effective range: 10#0 to 99
Output	BYTE_TO_BCD	BYTE	Outputs the value converted to the BCD code from the input argument.

INT_TO_BCD

Scope	Name	Type	Description
Input	I	INT	The binary code value to be converted. Effective range: 10#0 to 99
Output	INT_TO_BCD	BYTE	Outputs the value converted to the BCD code from the input argument. Outputs 16#FF when a value outside the effective range is input to the input I.

WORD_TO_BCD

Scope	Name	Type	Description
Input	W	WORD	The binary code value to be converted. Effective range: 10#0 to 9999
Output	WORD_TO_BCD	WORD	Outputs the value converted to the BCD code from the input argument.

DWORD_TO_BCD

Scope	Name	Type	Description
Input	X	DWORD	The binary code value to be converted. Effective range: 10#0 to 99999999
Output	DWORD_TO_BCD	DWORD	Outputs the value converted to the BCD code from the input argument.

■ Program example

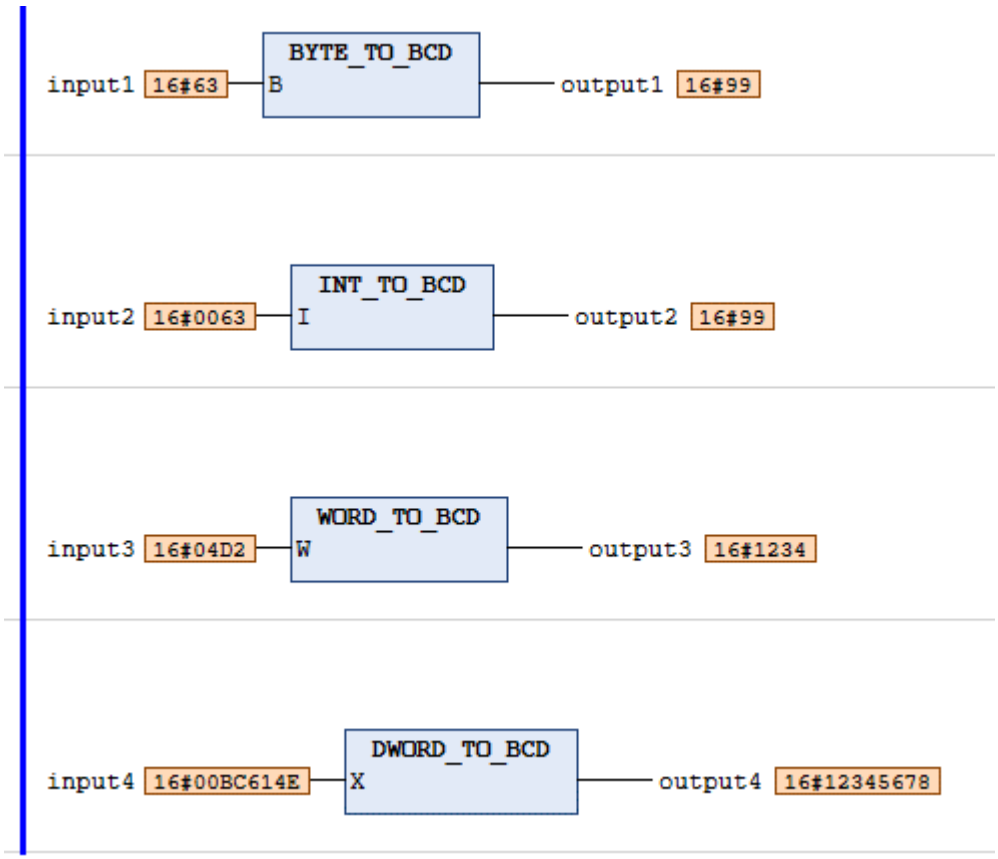
This program is designed to convert the BYTE type input variable “input1” to the BYTE type output variable “output1” and output the converted data.

This program is designed to convert the BYTE type input variable “input2” to the INT type output variable “output2” and output the converted data.

This program is designed to convert the WORD type input variable “input3” to the WORD type output variable “output3” and output the converted data.

This program is designed to convert the DWORD type input variable “input4” to the DWORD type output variable “output4” and output the converted data.

LD program



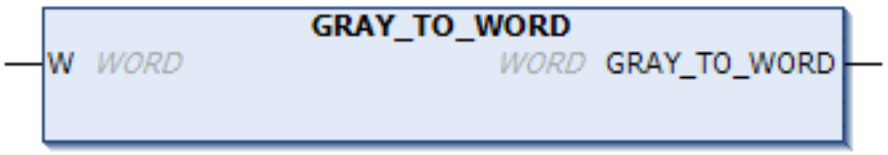
ST program

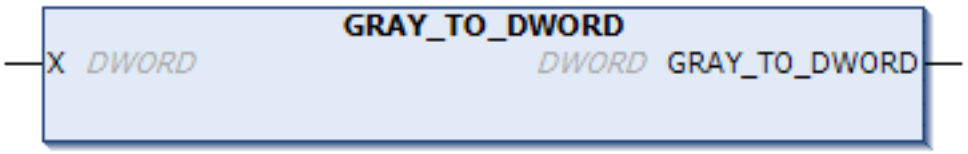
```
output1 16#99 := BYTE_TO_BCD(input1 16#63 );

output2 16#99 := INT_TO_BCD(input2 16#0063 );

output3 16#1234 := WORD_TO_BCD(input3 16#04D2 );

output4 16#12345678 := DWORD_TO_BCD(input4 16#00BC614E );
```

3.7.6 GRAY_TO_** (Gray Code to Binary Conversion)

This is a function that converts a Gray code input to a binary code of BYTE type, WORD type, or DWORD type.

■ **Icon**





■ **Parameter**

GRAY_TO_BYTE

Scope	Name	Type	Description
Input	B	BYTE	The BYTE type Gray code value to be converted
Output	GRAY_TO_BYTE	BYTE	A value converted to a BYTE type binary code from the input argument

GRAY_TO_WORD

Scope	Name	Type	Description
Input	W	WORD	The WORD-type Gray code value to be converted
Output	GRAY_TO_WORD	WORD	A value converted to a WORD-type binary code from the input argument

GRAY_TO_DWORD

Scope	Name	Type	Description
Input	X	DWORD	The DWORD type Gray code value to be converted
Output	GRAY_TO_DWORD	DWORD	A value converted to a DWORD type binary code from the input argument

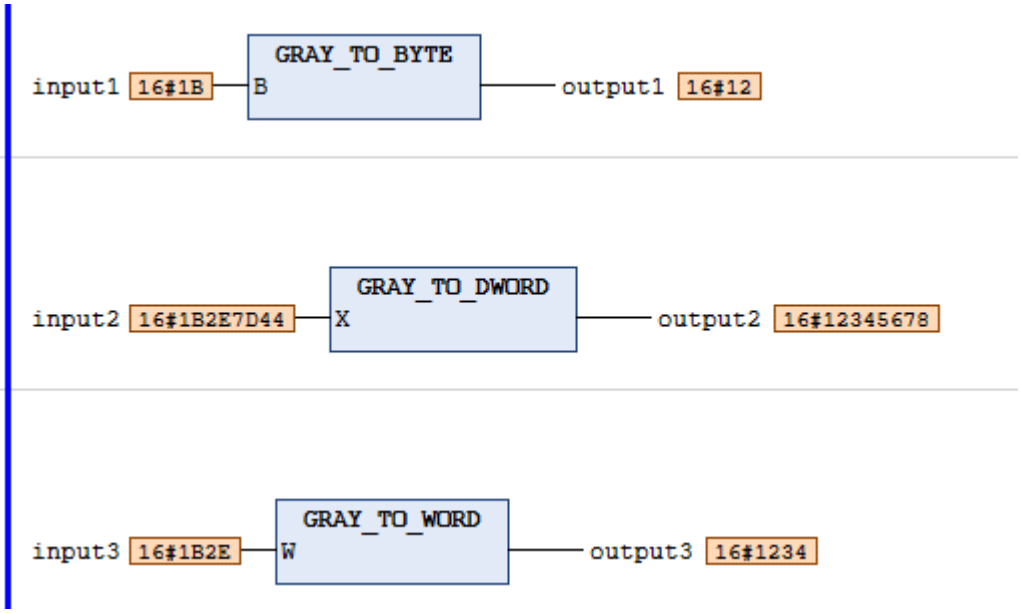
■ Program example

This program is designed to convert the BYTE type Gray code input variable “input1” to the BYTE type binary code output variable “output1” and output the converted data.

This program is designed to convert the DWORD type Gray code input variable “input2” to the DWORD type binary code output variable “output2” and output the converted data.

This program is designed to convert the WORD type Gray code input variable “input3” to the WORD type binary code output variable “output3” and output the converted data.

LD program



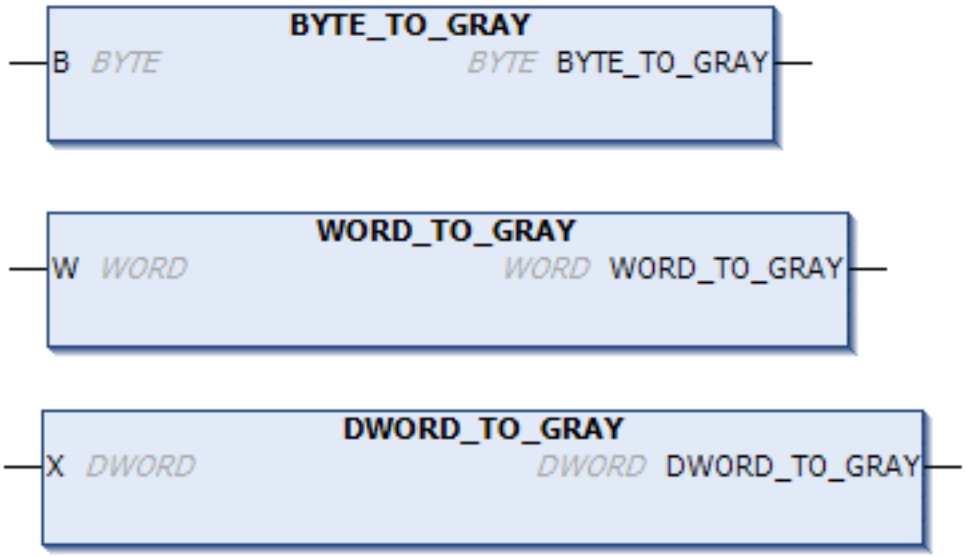
ST program

```
output1 16#12 :=GRAY_TO_BYTE(input1 16#1B );

output2 16#12345678 :=GRAY_TO_DWORD(input2 16#1B2E7D44 );

output3 16#1234 :=GRAY_TO_WORD(input3 16#1B2E );
```

3.7.7 **_TO_GRAY (Binary to Gray Code Conversion)

This is a function that converts a binary code input of BYTE type, WORD type, or DWORD type to a Gray code.

■ Icon



■ Parameter

BYTE_TO_GRAY

Scope	Name	Type	Description
Input	B	BYTE	The BYTE type binary code value to be converted
Output	BYTE_TO_GRAY	BYTE	A value converted to a BYTE type Gray code from the input argument

WORD_TO_GRAY

Scope	Name	Type	Description
Input	W	WORD	The WORD type binary code value to be converted
Output	WORD_TO_GRAY	WORD	A value converted to a WORD type Gray code from the input argument

DWORD_TO_GRAY

Scope	Name	Type	Description
Input	X	DWORD	The DWORD type binary code value to be converted
Output	DWORD_TO_GRAY	DWORD	A value converted to a DWORD type Gray code from the input argument

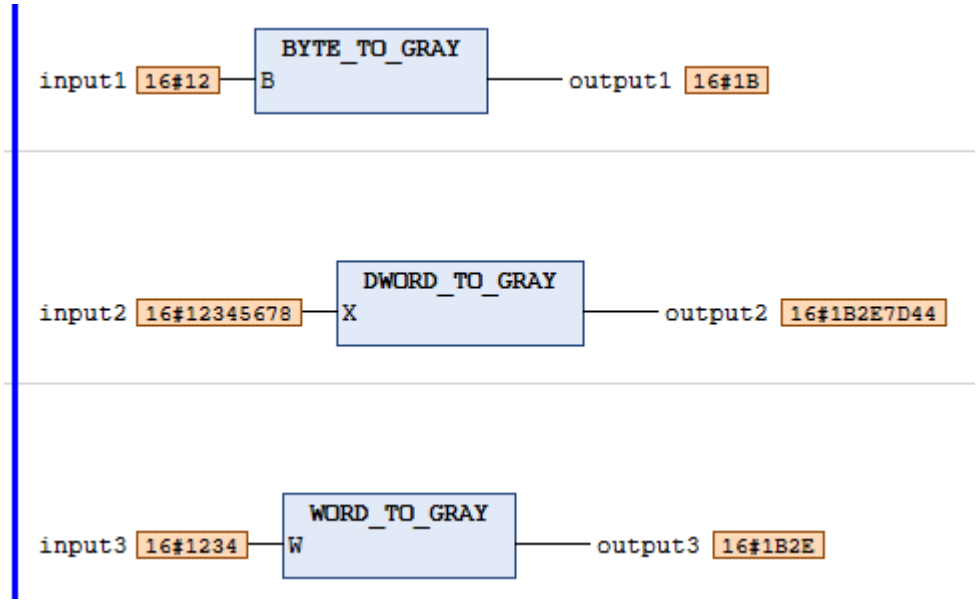
■ Program example

This program is designed to convert the BYTE type binary code input variable “input1” to the BYTE type Gray code output variable “output1” and output the converted data.

This program is designed to convert the DWORD type binary code input variable “input2” to the DWORD type Gray code output variable “output2” and output the converted data.

This program is designed to convert the WORD type binary code input variable “input3” to the WORD type Gray code output variable “output3” and output the converted data.

LD program



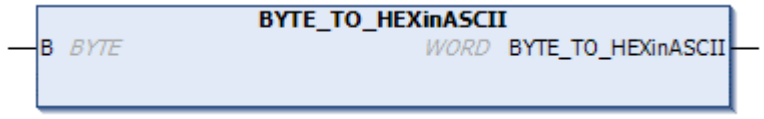
ST program

```
output1 16#1B :=BYTE_TO_GRAY(input1 16#12 );

output2 16#1B2E7D44 :=DWORD_TO_GRAY(input2 16#12345678 );

output3 16#1B2E :=WORD_TO_GRAY(input3 16#1234 );
```

3.7.8 BYTE_TO_HEXinASCII (Binary to ASCII Conversion)

This is a function that converts a one-byte hexadecimal binary-coded value to a one-word ASCII code.

■ Icon



■ Parameter

BYTE_TO_HEXinASCII

Scope	Name	Type	Description
Input	B	BYTE	The binary code value to be converted.

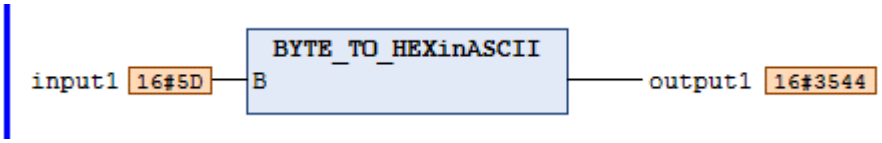
Scope	Name	Type	Description
Output	BYTE_TO_HEXinASCII	WORD	A value converted to an ASCII code from the input argument

■ Program example

This program is designed to convert the BYTE type input variable “input1” to the WORD type output variable “output1” and output the converted data.

input1 := 16#5D

LD program



ST program

```
output1 16#3544  :=  BYTE_TO_HEXinASCII (input1 16#5D );
```

Note

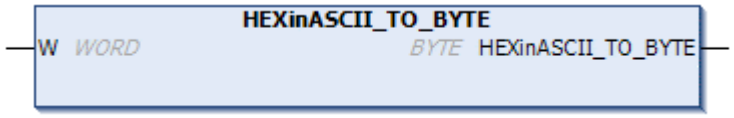
- **Convertible ASCII codes**

ASCII	Hexadecimal
0	0x30
1	0x31
2	0x32
3	0x33
4	0x34
5	0x35
6	0x36
7	0x37
8	0x38
9	0x39
A	0x41
B	0x42
C	0x43
D	0x44
E	0x45
F	0x46

3.7.9 HEXinASCII_TO_BYTE (ASCII to Binary Conversion)

This is a function that converts a one-word ASCII code to a one-byte hexadecimal binary-coded value.

■ Icon



■ Parameter

HEXinASCII_TO_BYTE

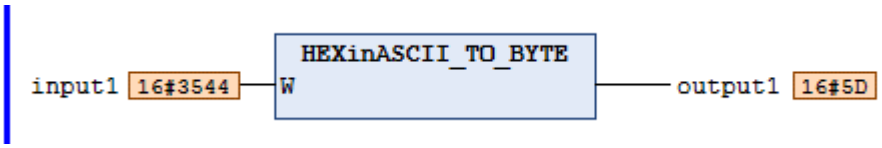
Scope	Name	Type	Description
Input	W	WORD	The ASCII code value to be converted
Output	HEXinASCII_TO_BYTE	BYTE	A value converted to a binary code from the input argument

■ Program example

This program is designed to convert the WORD type input variable “input1” to the BYTE type output variable “output1” and output the converted data.

input1 := 16#3544

LD program



ST program

```
output1 16#5D := HEXinASCII_TO_BYTE (input1 16#3544 );
```

Note

- **Convertible ASCII codes**

  Inputs other than those shown below cause 0 to be output.

16 進数	ASCII
0x30	0
0x31	1
0x32	2
0x33	3
0x34	4
0x35	5
0x36	6
0x37	7
0x38	8
0x39	9
0x41	A
0x42	B
0x43	C
0x44	D
0x45	E
0x46	F

3.7.10 MEM.Decode (4BYTE to DWORD Conversion)

This is a function that decodes data in units of byte to data in units of DWORD. The number of bytes that can be decoded is a multiple of 4 within the effective range 10#4 to 10#65532.

■ Icon

■ Parameter

Scope	Name	Type	Description
Input	pSource	POINTER TO BYTE	Start pointer to data in units of byte
Input	pDestination	POINTER TO DWORD	Start pointer to data in units of DWORD
Input	uiNumberOfBytes	UINT	Number of bytes to decode Effective range: 10#4 to 10#65532
Output	Decode	BOOL	Always outputs FALSE

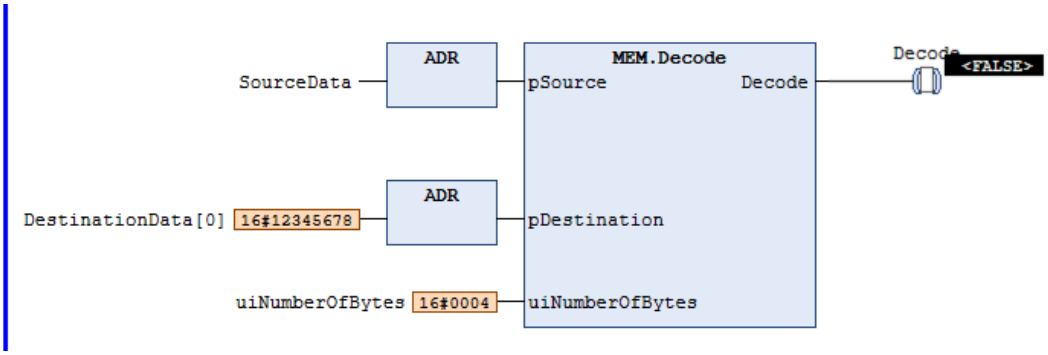
■ Program example

This program is designed to decode four-byte (uiNumberOfBytes) data of an input variable (SourceData[16#78,16#56,16#34,16#12]) into one-dword data (DestinationData[16#12345678]).

SourceData : ARRAY [0..3] OF BYTE := [16#78,16#56,16#34,16#12] (decode source data)

uiNumberOfBytes := 10#4 (16#4)

LD program



ST program

```
SourceData[3] 16#12;
SourceData[2] 16#34;
SourceData[1] 16#56;
SourceData[0] 16#78;

Decode FALSE := MEM.Decode(ADR(SourceData), ADR(DestinationData[0] 16#12345678 ),4);
```

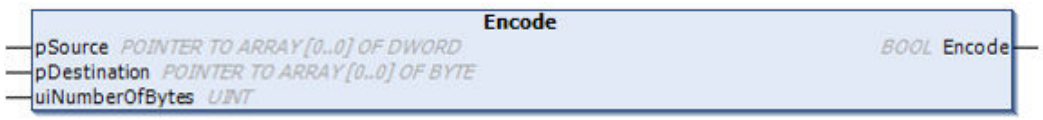
Note

- Use a multiple of 4 for the number of bytes to decode (uiNumberOfBytes). The bytes of data other than a multiple of 4 cannot be decoded and 0 is output.
- Do not set 0 (NULL) in the start pointer to decode source data (pSource) and the start pointer to decode destination data (pDestination). If set to NULL, an exception error occurs.

3.7.11 MEM.Encode (DWORD to 4BYTE Conversion)

This is a function that encodes data in units of DWORD into data in units of bytes. The number of bytes that can be encoded is a multiple of 4 within the effective range 10#4 to 10#65532.

■ Icon



■ Parameter

Scope	Name	Type	Description
Input	pSource	POINTER TO DWORD	Start pointer of data in units of DWORD
Input	pDestination	POINTER TO BYTE	Start pointer to data in units of byte
Input	uiNumberOfBytes	UINT	Number of bytes to encode Effective range: 10#4 to 10#65532
出力	Encode	BOOL	Number of bytes to encode Effective range: 10#4 to 10#65532

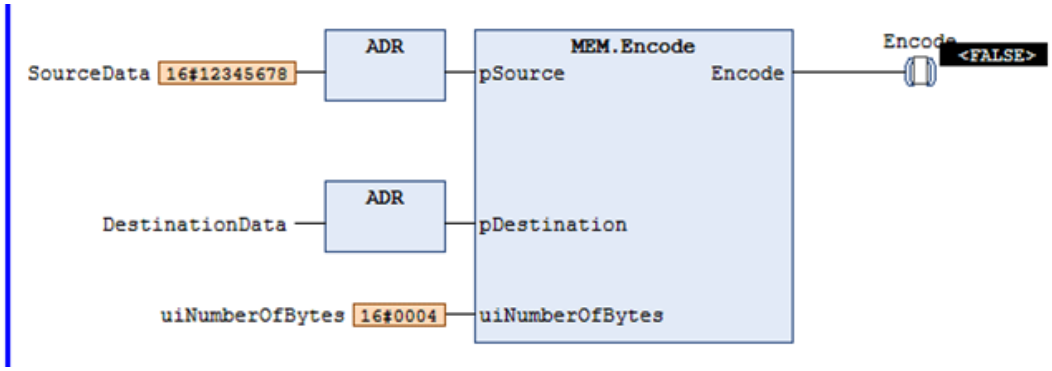
■ Program example

This program is designed to encode four-byte (uiNumberOfBytes) data of an input variable (SourceData[16#12345678]) into four-byte data (DestinationData[16#78,16#56,16#34,16#12]).

SourceData := 16#12345678

uiNumberOfBytes := 10#4 (16#4)

LD program



ST program

```
Encode FALSE := MEM.Encode(ADR(SourceData 16#12345678 ), ADR(DestinationData),4);

        DestinationData[3] 16#12 ;
        DestinationData[2] 16#34 ;
        DestinationData[1] 16#56 ;
        DestinationData[0] 16#78 ;
```

Note

- Use a multiple of 4 for the number of bytes to encode (uiNumberOfBytes). The bytes of data other than a multiple of 4 cannot be encoded and 0 is output.
- Do not set 0 (NULL) in the start pointer to encode source data (pSource) and the start pointer to encode destination data (pDestination). If set to NULL, an exception error occurs.

3.7.12 MEM.PackArrayOfBoolToArrayOfByte (BOOL Array to BYTE Conversion)

This is a function that packs a BOOL type array into an array in bytes and copies a specified bit size data. The function returns the number of bytes required for coping. The maximum copyable size is 65535 bits (approx. 8192 bytes).

■ Icon

■ Parameter

Scope	Name	Type	Description
Input	paxSource	POINTER TO BYTE	Starting address of BOOL type array data
Input	pabyDestination	POINTER TO BYTE	Starting address of Byte type data
Input	uiNumberOfBits	UINT	Number of bits to copy Effective range: 10#1 to 65535
出力	PackArrayOfBoolToArrayOfByte	UINT	Outputs the number of bytes required for coping

■ Program example

This program is designed to pack a 24-bit amount (uiNumberOfBits) of BOOL type copy source data (xbit) in bytes and copy the packed data to the copy destination (ArrayBlock).

The program returns the number of bytes required for coping.

ArrayBlock : ARRAY [0..4] OF BYTE := [5(0)]　(copy destination data: default value)

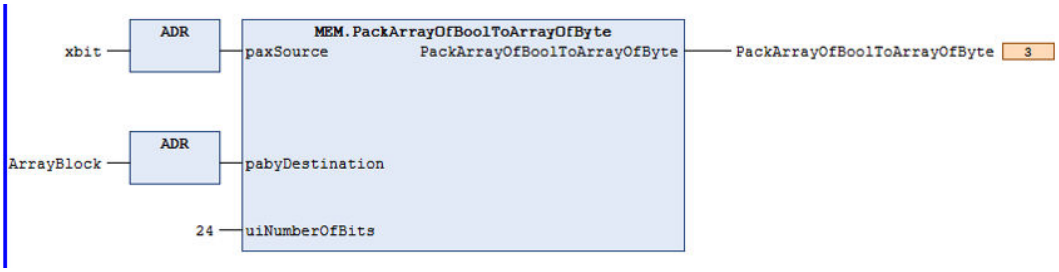
xbit[7:0] = 2#00010010[16#12]

xbit[15:8] = 2#00110100[16#34]

xbit[23:16] = 2#01010110[16#56]

uiNumberOfBits := 10#24

LD program



ST program

```
xbit[7] FALSE :=FALSE;  xbit[6] FALSE :=FALSE;  xbit[5] FALSE :=FALSE;  xbit[4] TRUE :=TRUE;
xbit[3] FALSE :=FALSE;  xbit[2] FALSE :=FALSE;  xbit[1] TRUE :=TRUE;    xbit[0] FALSE :=FALSE;
xbit[15] FALSE :=FALSE; xbit[14] FALSE :=FALSE; xbit[13] TRUE :=TRUE;   xbit[12] TRUE :=TRUE;
xbit[11] FALSE :=FALSE; xbit[10] TRUE :=TRUE;   xbit[9] FALSE :=FALSE;  xbit[8] FALSE :=FALSE;
xbit[23] FALSE :=FALSE; xbit[22] TRUE :=TRUE;   xbit[21] FALSE :=FALSE; xbit[20] TRUE :=TRUE;
xbit[19] FALSE :=FALSE; xbit[18] TRUE :=TRUE;  xbit[17] TRUE :=TRUE;   xbit[16] FALSE :=FALSE;

PackArrayOfBoolToArrayOfByte 16#0003  := MEM.PackArrayOfBoolToArrayOfByte(ADR(xbit), ADR(ArrayBlock), 24);

ArrayBlock[0] 16#12;
ArrayBlock[1] 16#34;
ArrayBlock[2] 16#56;
```

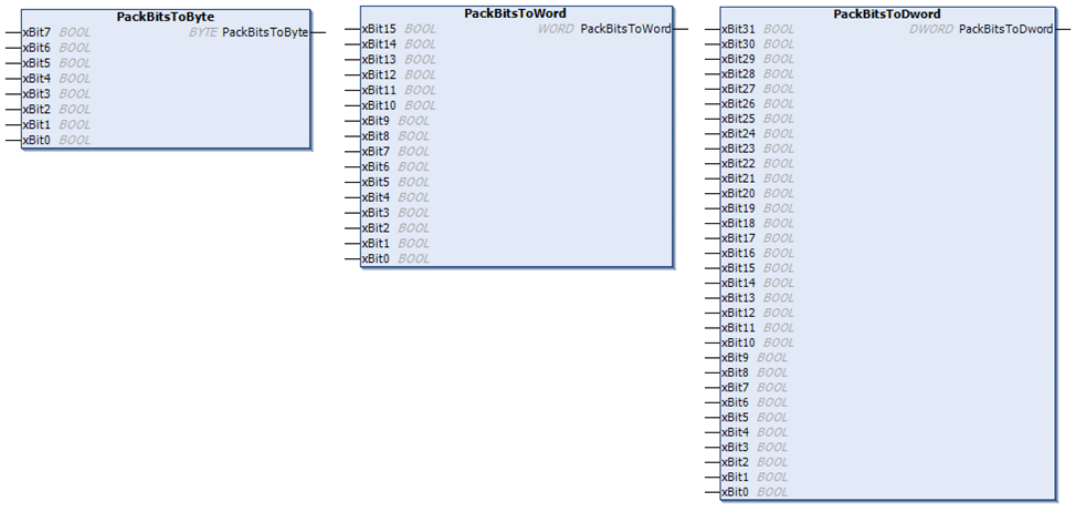
Note

- If the number of bits to be copied uiNumberOfBits = 0, the copying will not be carried out and the return value of the function will be PackArrayOfBoolToArrayOfByte = 0.

3.7.13 MEM.PackBitsTo**(Bit Data to BYTE/WORD/DWORD Conversion)

This is a function that packs input BOOL type data and outputs a BYTE, a WORD, or a DWORD.

■ Icon



■ Parameter

MEM.PackBitsToByte

Scope	Name	Type	Description
Input	xBit0~xBit7	BOOL	Bits to be packed Bit0 to Bit7
Output	PackBitsToByte	BYTE	A value of the packed input

MEM.PackBitsToWord

Scope	Name	Type	Description
Input	xBit0~xBit15	BOOL	Bits to be packed Bit0 to Bit15
Output	PackBitsToWord	WORD	A value of the packed input

MEM.PackBitsToDword

Scope	Name	Type	Description
Input	xBit0~xBit31	BOOL	Bits to be packed Bit0 to Bit31
Output	PackBitsToDword	DWORD	A value of the packed input

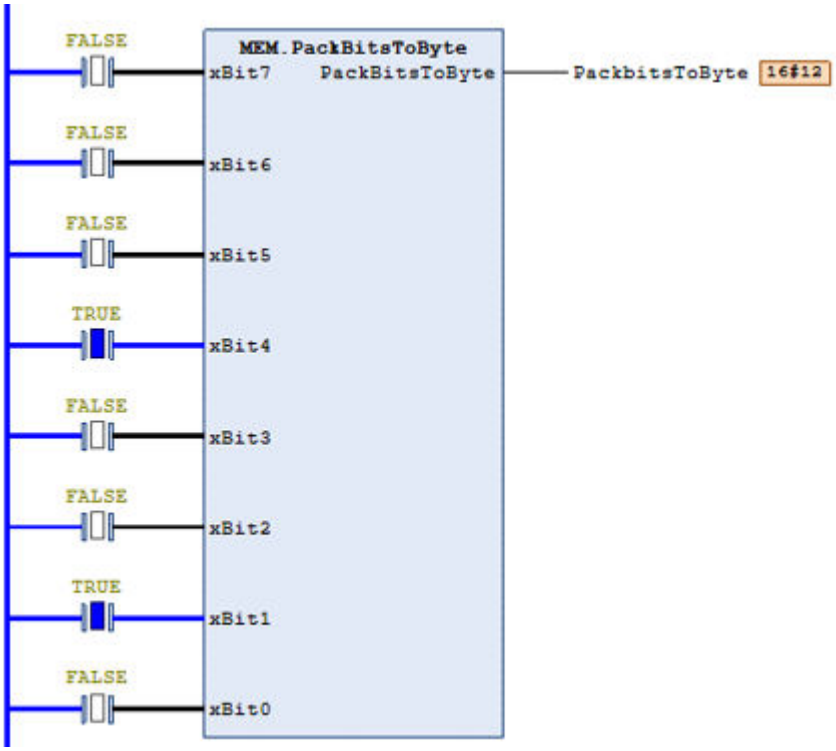
■ Program example 1

This program is designed to pack the BOOL type input variables xBit and output the packed data to the BYTE type output variable PackBitsToByte.

xBit4,xBit1 :=TRUE

Others :=FALSE

LD program



ST program

```
PackbitsToByte 16#12 :=MEM.PackbitsToByte (xbit[7] FALSE ,xbit[6] FALSE ,xbit[5] FALSE ,xbit[4] TRUE ,
                                           xbit[3] FALSE ,xbit[2] FALSE ,xbit[1] TRUE ,xbit[0] FALSE );
```

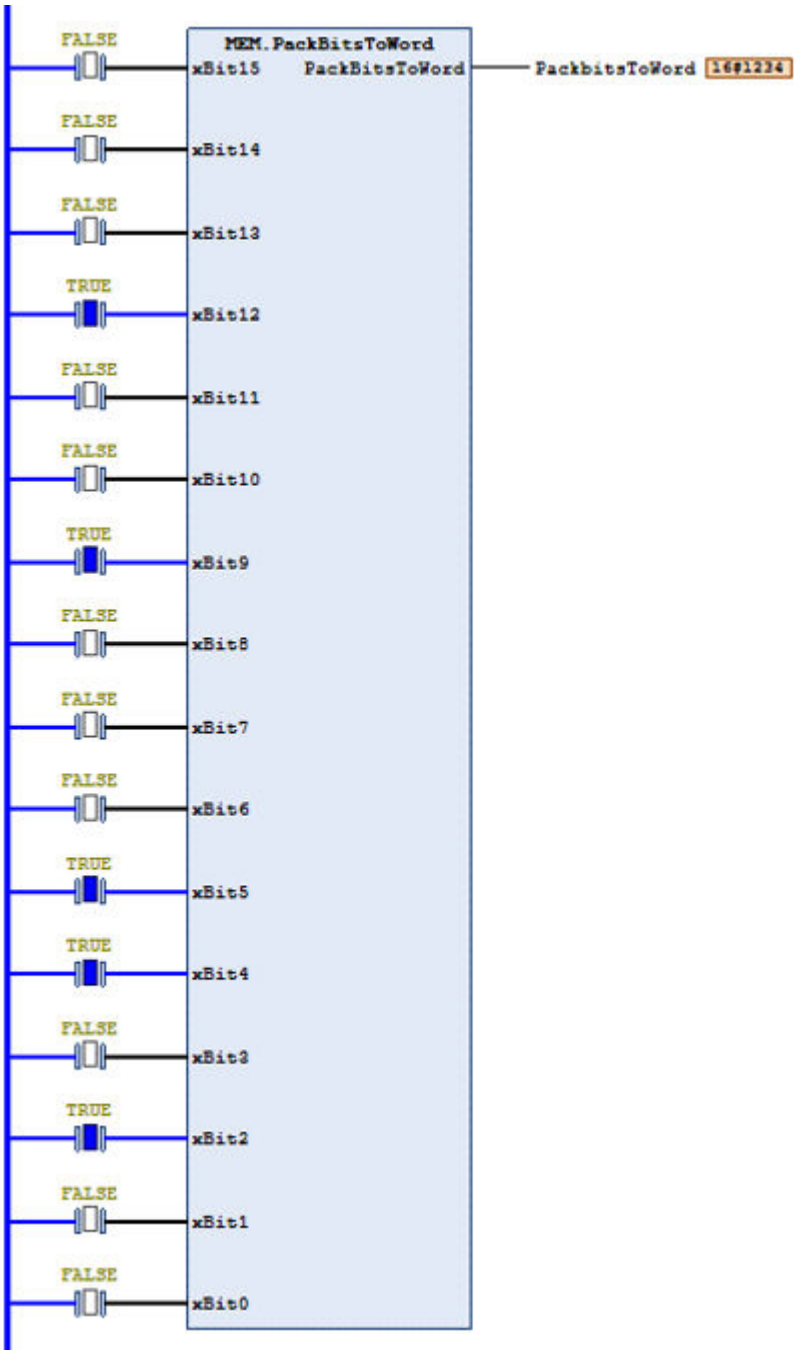
■ Program example 2

This program is designed to pack the BOOL type input variables xBit and output the packed data to the WORD type output variable PackBitsToWord.

xBit12,xBit9,xBit5,xBit4,xBit2 :=TRUE

Others :=FALSE

LD program



ST program

```
PackbitsToWord 16#1234 :=MEM.PackBitsToWord(xbit[15] FALSE ,xbit[14] FALSE ,xbit[13] FALSE ,xbit[12] TRUE ,
                                            xbit[11] FALSE ,xbit[10] FALSE ,xbit[9] TRUE ,xbit[8] FALSE ,
                                            xbit[7] FALSE ,xbit[6] FALSE ,xbit[5] TRUE ,xbit[4] TRUE ,
                                            xbit[3] FALSE ,xbit[2] TRUE ,xbit[1] FALSE ,xbit[0] FALSE );
```

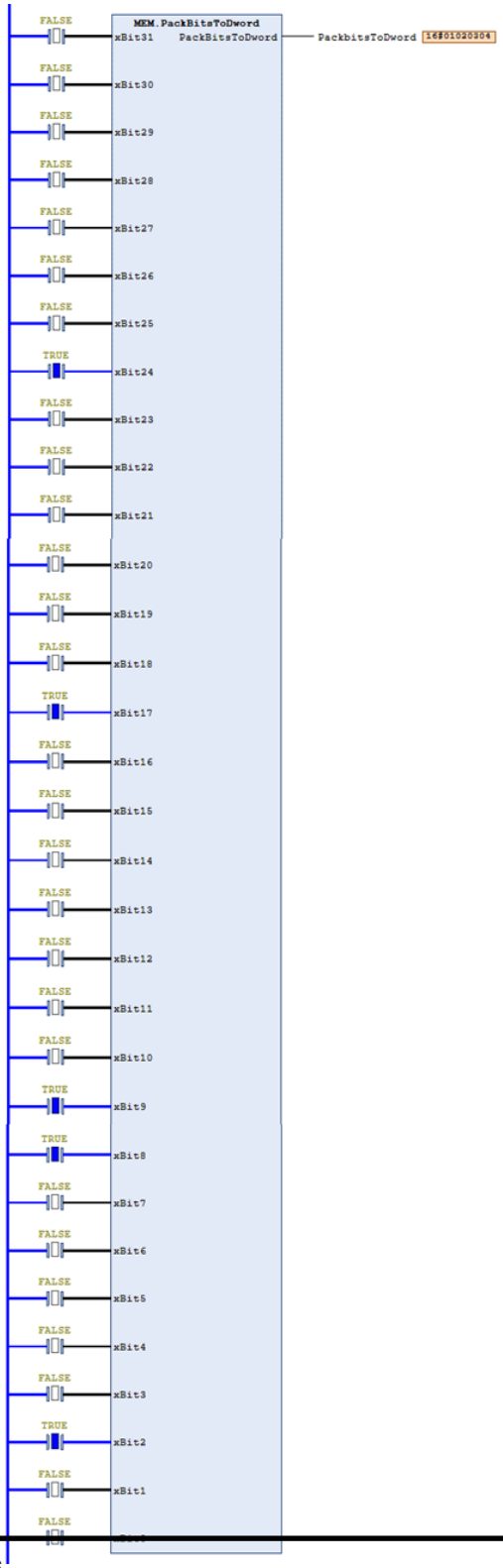
■ Program example 3

This program is designed to pack the BOOL type input variables xBit and output the packed data to the DWORD type output variable PackBitsToDword.

xBit24,xBit17,xBit9,xBit8,xBit2 :=TRUE

Others :=FALSE

LD program


MEM.PackBitsToDword
xBit31
PackBitsToDword
PackbitsToDword
16#01020304
FALSE xBit30
FALSE xBit29
FALSE xBit28
FALSE xBit27
FALSE xBit26
FALSE xBit25
TRUE xBit24
FALSE xBit23
FALSE xBit22
FALSE xBit21
FALSE xBit20
FALSE xBit19
FALSE xBit18
TRUE xBit17
FALSE xBit16
FALSE xBit15
FALSE xBit14
FALSE xBit13
FALSE xBit12
FALSE xBit11
FALSE xBit10
TRUE xBit9
TRUE xBit8
FALSE xBit7
FALSE xBit6
FALSE xBit5
FALSE xBit4
FALSE xBit3
TRUE xBit2
FALSE xBit1
FALSE

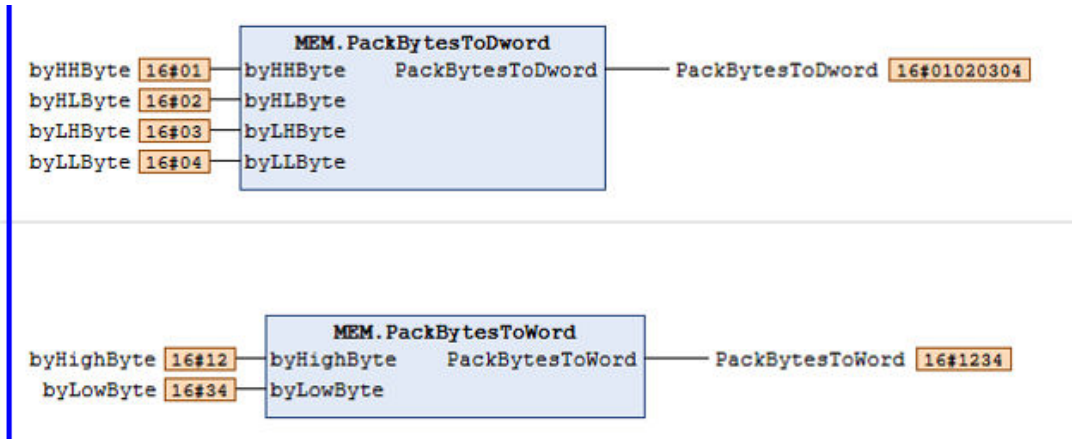
ST program

```
PackbitsToDword 16#01020304 :=MEM.PackBitsToDword(xbit[31] FALSE ,xbit[30] FALSE ,xbit[29] FALSE ,xbit[28] FALSE ,
                                                  xbit[27] FALSE ,xbit[26] FALSE ,xbit[25] FALSE ,xbit[24] TRUE ,
                                                  xbit[23] FALSE ,xbit[22] FALSE ,xbit[21] FALSE ,xbit[20] FALSE ,
                                                  xbit[19] FALSE ,xbit[18] FALSE ,xbit[17] TRUE ,xbit[16] FALSE ,
                                                  xbit[15] FALSE ,xbit[14] FALSE ,xbit[13] FALSE ,xbit[12] FALSE ,
                                                  xbit[11] FALSE ,xbit[10] FALSE ,xbit[9] TRUE ,xbit[8] TRUE ,
                                                  xbit[7] FALSE ,xbit[6] FALSE ,xbit[5] FALSE ,xbit[4] FALSE ,
                                                  xbit[3] FALSE ,xbit[2] TRUE ,xbit[1] FALSE ,xbit[0] FALSE );
```

3.7.14 MEM.PackBytesTo**(BYTE to WORD/DWORD Conversion)

This is a function that packs input BYTE type data and outputs one-word or one-dword data.

■ Icon



■ Parameter

PackBytesToWord

Scope	Name	Type	Description
Input	byHighByte	BYTE	PackBytesToWord
Input	byLowByte	BYTE	Low byte to be packed
Output	PackBytesToWord	WORD	A value of the packed input

PackBytesToDword

Scope	Name	Type	Description
Input	byHHByte	BYTE	HH byte to be packed
Input	byHLByte	BYTE	HL byte to be packed
Input	byLHByte	BYTE	LH byte to be packed
Input	byLLByte	BYTE	LL byte to be packed
Output	PackBytesToDword	DWORD	A value of the packed input

■ Program example

- PackBytesToDword

  This program is designed to pack the byHHByte, byHLByte, byLHByte, and byLLByte input variables of the BYTE type and output the packed data to the PackBytesToDword output variable of the DWORD type.

  byHHByte := 16#01 、byHLByte : = 16#02 、byLHByte := 16#03 、byLLByte := 16#04

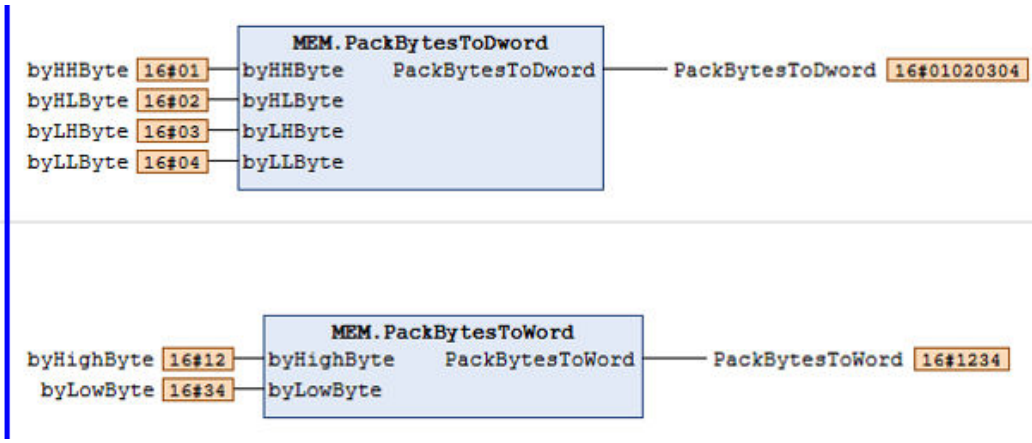
- PackBytesToWord

  This program is designed to pack the byHighByte and byLowByte input variables of the BYTE type and output the packed data to the PackBytesToWord output variable of the WORD type.

  byHighByte := 16#12

  byLowByte := 16#34

LD program



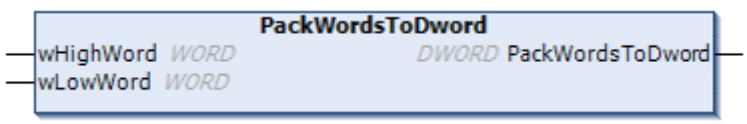
ST program

```
PackBytesToDword 16#01020304 :=
MEM.PackBytesToDword(byHHByte 16#01 ,byHLByte 16#02 ,byLHByte 16#03 ,byLLByte 16#04 );

PackBytesToWord 16#1234 :=
MEM.PackBytesToWord(byHighByte 16#12 ,byLowByte 16#34 );
```

3.7.15 MEM.PackWordsToDword (WORD to DWORD Conversion)

This is a function that packs input WORD type data and outputs a DWORD.

■ Icon



■ **Parameter**

Scope	Name	Type	Description
Input	wHighWord	WORD	High WORD to be packed
Input	wLowWord	WORD	Low WORD to be packed
Output	PackWordsToDword	DWORD	A value of the packed input

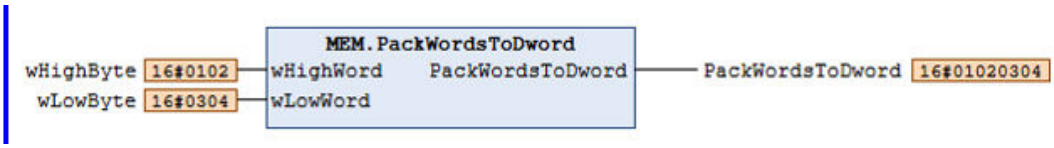
■ **Program example**

This program is designed to pack the wHighWord and wLowWord input variables of the WORD type and output the packed data to the PackWordsToDword output variable of the DWORD type.

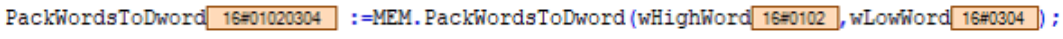
wHighWord := 16#0102

wLowWord := 16#0304

LD program



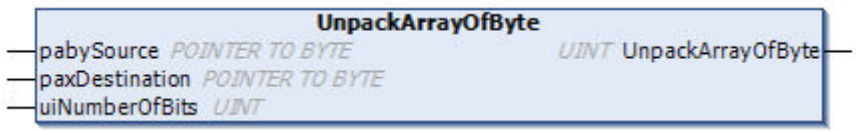
ST program



3.7.16 MEM.UnpackArrayOfByte (BYTE to BOOL Array Conversion)

This is a function that unpacks a BYTE type array to data in units of bits and copies a specified bit size of the data to a destination BOOL array. The function returns the number of bytes required for coping. The maximum copyable size is 65535 bits (approx. 8192 bytes).

■ **Icon**



■ **Parameter**

Scope	Name	Type	Description
Input	pabySource	POINTER TO BYTE	Starting address of Byte type data
Input	paxDestination	POINTER TO BYTE	Starting address of BOOL type array data
Input	uiNumberOfBits	UINT	Number of bits to copy Effective range: 10#1 to 65535
出力	UnpackArrayOfByte	UINT	Outputs the number of bytes required for coping

■ Program example

This program is designed to unpack BYTE type copy source data (ArrayBlock) to a 24-bit amount (uiNumberOfBits) in bits and copy the unpacked data to the copy destination (xbit).

The program returns the number of bytes required for coping.

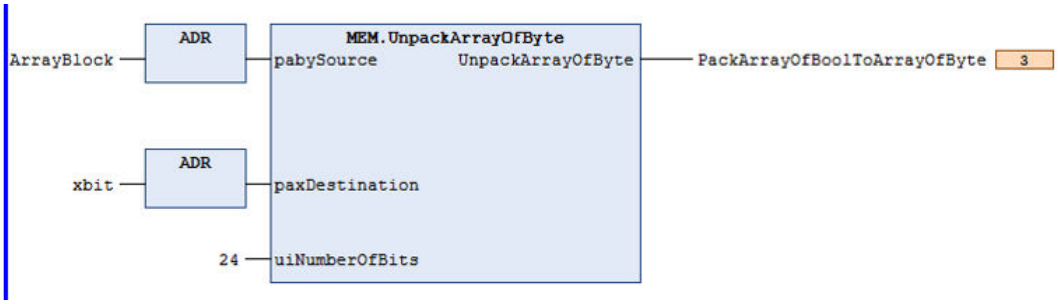
ArrayBlock[0] := 16#12

ArrayBlock[1] := 16#34

ArrayBlock[2] := 16#56

uiNumberOfBits := 10#24 (16#18)

LD program



ST program

```
ArrayBlock[0] 16#12 := 16#12;
ArrayBlock[1] 16#34 := 16#34;
ArrayBlock[2] 16#56 := 16#56;

UnpackArrayOfByte 16#0003 := MEM.UnpackArrayOfByte(ADR(ArrayBlock), ADR(xbit),24);

xbit[7] FALSE;  xbit[6] FALSE;  xbit[5] FALSE;  xbit[4] TRUE;
xbit[3] FALSE;  xbit[2] FALSE;  xbit[1] TRUE;   xbit[0] FALSE;
xbit[15] FALSE; xbit[14] FALSE; xbit[13] TRUE;  xbit[12] TRUE;
xbit[11] FALSE; xbit[10] TRUE;  xbit[9] FALSE;  xbit[8] FALSE;
xbit[23] FALSE; xbit[22] TRUE;  xbit[21] FALSE; xbit[20] TRUE;
xbit[19] FALSE; xbit[18] TRUE;  xbit[17] TRUE;  xbit[16] FALSE;
```

Note

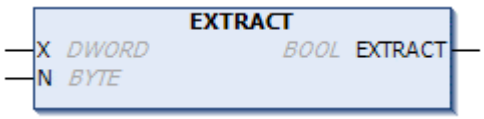
- If the number of bits to be copied uiNumberOfBits = 0, the copying will not be carried out and the return value of the function will be UnpackArrayOfByte = 0.

3.8 Bit operation instructions

3.8.1 EXTRACT (Bit Extraction)

This is a function that outputs the bit number N value (BOOL) of input value X (DWORD).

■ Icon



■ Parameter

Scope	Name	Type	Description
Input	X	DWORD	Input value
Input	N	BYTE	Number of the bit to be extracted. Effective range: 10#0 to 31
Output	EXTRACT	BOOL	The Nth bit value of the input value X

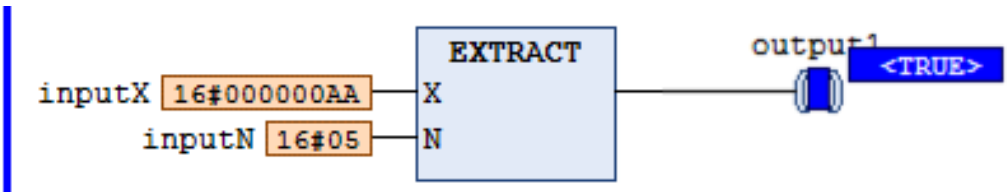
■ Program example

This program is designed to output the inputNth bit value of the inputX input variable of the DWORD type to the BOOL type output variable “output1”.

inputX := 16#AA (2#10101010)

inputN := 16#5

LD program



ST program

```
output1 TRUE := EXTRACT(inputX 16#000000AA ,inputN 16#05 );
```

Note

- The allowable range of the input value N (bit number) is 0 to 31 (bits).

3.8.2 PUTBIT (Bit Change)

This is a function that changes the value at bit number N of input value X (DWORD) to the B value and outputs a DWORD with the changed value at the bit number.

■ Icon

■ Parameter

Scope	Name	Type	Description
Input	X	DWORD	Input value
Input	N	BYTE	Number of the bit to be changed. Effective range: 10#0 to 31
Input	B	BOOL	Value of specified bit
出力	PUTBIT	DWORD	Value with the Nth bit of the input value X changed to the B value

■ Program example

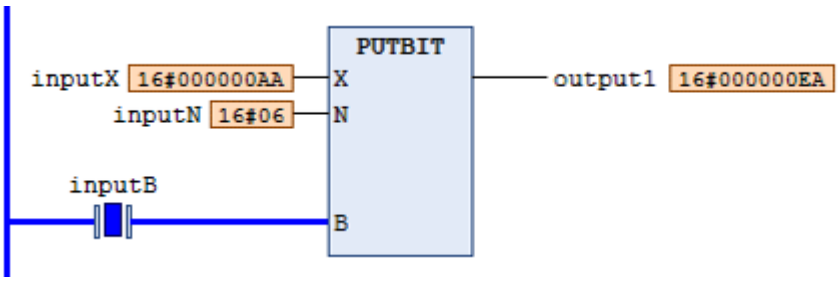
This program is designed to output a value with the inputNth bit value of the DWORD type inputX input variable changed to the inputB value to the DWORD type output variable “output1”.

inputX := 16#AA (2#10101010)

inputN := 16#6

inputB := TRUE

LD program



ST program

```
output1 16#000000EA := PUTBIT(inputX 16#000000AA, inputN 16#06, inputB TRUE);
```

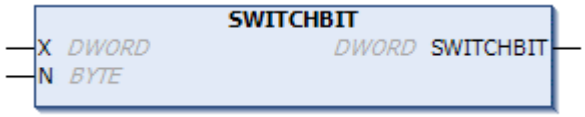
Note

- The allowable range of the input value N (bit number) is 0 to 31 (bits).

3.8.3 SWITCHBIT (Bit Inversion)

This is a function that inverts the bit number N value (0 to 1/1 to 0) of input value X (DWORD) and outputs the DWORD with the inverted value at the bit number.

■ Icon



■ Parameter

Scope	Name	Type	Description
Input	X	DWORD	Input value
Input	N	BYTE	Number of the bit to invert. Effective range: 10#0 to 31
Output	SWITCHBIT	DWORD	Value with the Nth bit value of the input value X inverted

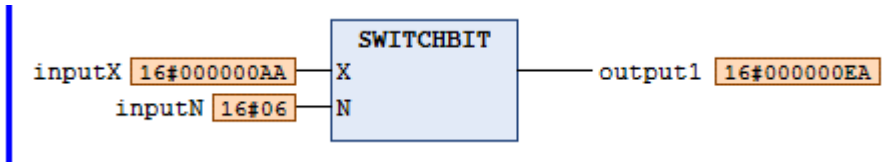
■ Program example

This program is designed to invert the inputNth bit value of the DWORD type inputX input variable and output a value with the inverted bit value to the DWORD type output variable “output1”.

inputX := 16#AA (2#10101010)

inputN := 10#6

LD program



LD program

```
output1 16#000000EA := SWITCHBIT(inputX 16#000000AA ,inputN 16#06 );
```

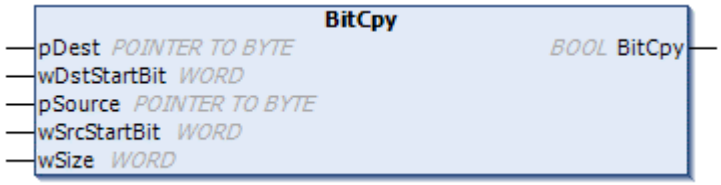
Note

- The allowable range of the input value N (bit number) is 0 to 31 (bits).

3.8.4 MEMUtils.BitCpy (Bit Copying)

This is a function that copies a specified size of bit data from copy source data.The maximum copyable size is 65535 bits (approx. 8191 bytes).

■ Icon



■ Parameter

Scope	Name	Type	Description
Input	pDest	POINTER TO BYTE	Start pointer to copy destination data
Input	wDstStartBit	WORD	Start bit position in copy destination data
Input	pSource	POINTER TO BYTE	Start pointer to copy source data
Input	wSrcStartBit	WORD	Start bit position in copy source data
Input	wSize	WORD	Bit size to copy. Effective range: 10#1 to 65535
Output	BitCpy	BOOL	Always outputs FALSE

■ Program example

This program is designed to copy 40 bits (wSize) in copy source data (SourceData) onto copy destination data (DestinationData).

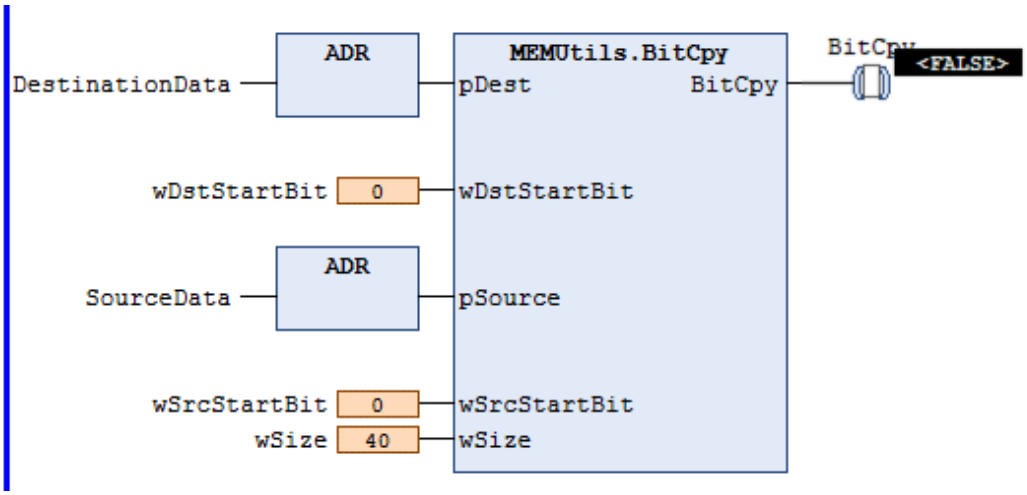
SourceData : ARRAY [0..4] OF BYTE := [1,2,3,4,5] (Copy source data)

DestinationData : ARRAY [0..4] OF BYTE := [5(0)] (Copy destination data)

wDstStartBit := 0 , wSrcStartBit := 0

wSize := 40

LD program



ST program

```
BitCpyFALSE := MEMUtils.BitCpy(ADR(DestinationData),0,ADR(SourceData ),0,40);

SourceData[0] 16#01;          DestinationData[0] 16#01;
SourceData[1] 16#02;          DestinationData[1] 16#02;
SourceData[2] 16#03;          DestinationData[2] 16#03;
SourceData[3] 16#04;          DestinationData[3] 16#04;
SourceData[4] 16#05;          DestinationData[4] 16#05;
```

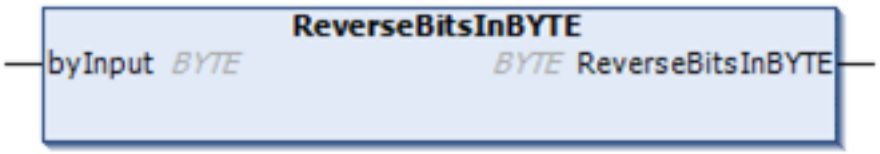
Note

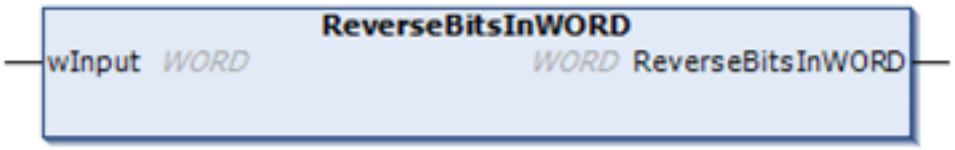
- If the wSize value is 0, the copying will not be carried out.
- If copying in units of byte is required, use the function in ”3.12.10 MEM.MemMove”.
- If any of the start bit positions are set to a value other than 0, the parameters need to be configured such that both the conditions below are satisfied.
  - wSize ≤ 65536 – wDstStartBit
  - wSize ≤ 65536 – wSrcStartBit

3.8.5 MEM.ReverseBitsIn** (Bit Order Change)

This is a function that reverses the order of the bits of input BYTE- , WORD-, or DWORD-type data and outputs the data of the bits in reverse order.

■ **Icon**





■ Parameter

ReverseBitsInBYTE

Scope	Name	Type	Description
Input	byInput	BYTE	Input value, BYTE type data
Output	ReverseBitsInBYTE	BYTE	Value in reverse bit order

ReverseBitsInWORD

Scope	Name	Type	Description
Input	wInput	WORD	Input value, WORD type data
Output	ReverseBitsInWORD	WORD	Value in reverse bit order

ReverseBitsInDWORD

Scope	Name	Type	Description
Input	dwInput	DWORD	Input value, DWORD type data
Output	ReverseBitsInDWORD	DWORD	Value in reverse bit order

■ Program example

- ReverseBitsInBYTE

  This program is designed to reverse the order of bits of the byInput input variable of the BYTE type and outputs the data of the bits in reverse order to the ReverseBitsInBYTE output variable of the BYTE type.

  byInput := 16#12

- ReverseBitsInDWORD

  This program is designed to reverse the order of bits of the dwInput input variable of the DWORD type and outputs the data of the bits in reverse order to the ReverseBitsInDWORD output variable of the DWORD type.

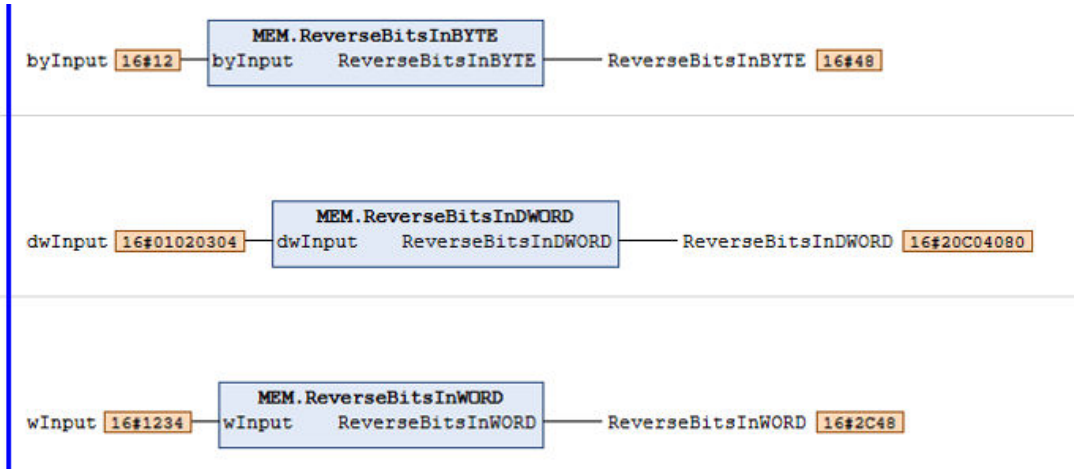
  dwInput := 16#01020304

- ReverseBitsInWORD

  This program is designed to reverse the order of bits of the wInput input variable of the WORD type and outputs the data of the bits in reverse order to the ReverseBitsInWORD output variable of the WORD type.

  wInput := 16#1234

LD program



ST program

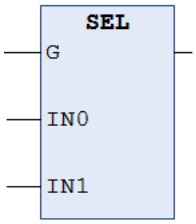
```
ReverseBitsInBYTE 16#48 :=MEM.ReverseBitsInBYTE(byInput 16#12 );

ReverseBitsInDWORD 16#20C04080  :=MEM.ReverseBitsInDWORD (dwInput 16#01020304 );

ReverseBitsInWORD 16#2C48  :=MEM.ReverseBitsInWORD(wInput 16#1234 );
```

3.9 Memory operation instructions

3.9.1 SEL (Binary Selector)

This is a function that outputs the value of the input argument IN0 or IN1 depending on whether the input argument G is true or false.

■ Icon



■ Parameter

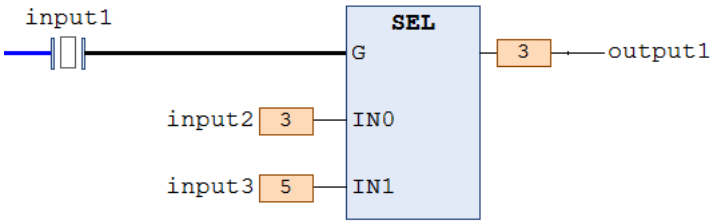
Scope	Name	Type	Description
Input	G	BOOL	Conditions for selecting the contents to be output
	IN0	All	Specifies the variable to be output if G is FALSE.
	IN1	All	Specifies the variable to be output if G is TRUE.
Output	-	All	Outputs the value of IN0 or IN1 depending on the value of G.

■ Program example

This program is designed to output the value of the input variable “input2” or “input3” to the output variable “output1” depending on the value of the input variable “input1”.

LD program

This program is designed to output the value of “input2” (IN0) because the value of “input1” is FALSE.



ST program

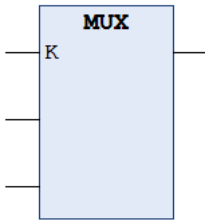
This program is designed to output the value of “input3” (IN1) to the "output1" because the value of “input1” is TRUE.

```
output1 5 := SEL(input1 TRUE ,input2 3 ,input3 5 );
```

3.9.2 MUX (Multiplexer)

This is a function that selectively outputs the input arguments depending on the value of the input argument K.

■ Icon



■ Parameter

Scope	Name	Type	Description
Input	K	(Note 1)	Specifies the value (K = 0, 1, 2...) to select the value to output.
	-	All	Specifies the value to be output depending on K.
Output	-	All	Outputs one of the input arguments depending on the value of K.

(Note 1) Usable data type

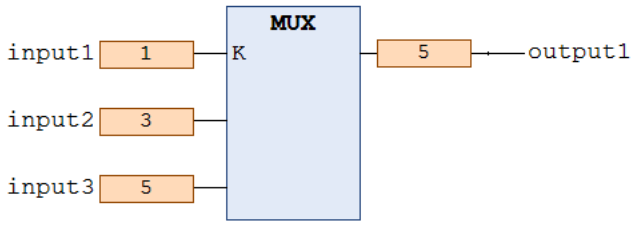
BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT

■ Program example

This program is designed to output the value of the input variable “input2” or “input3” to the output variable “output1” depending on the value of the input variable “input1”.

LD program

This program is designed to output the value of “input3” to “output1” depending on the value (1) of “input1”.



ST program

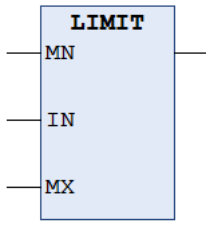
This program is designed to output the value of “input 2” to the “output 1” depending on the value (0) of “input 1”.

```
output1 3  := MUX(input1 0 , input2 3 , input3 5 );
```

3.9.3 LIMIT (Limiter)

This is a function that limits the input value with the lower and upper limit values and outputs a restricted value.

■ Icon



■ Parameter

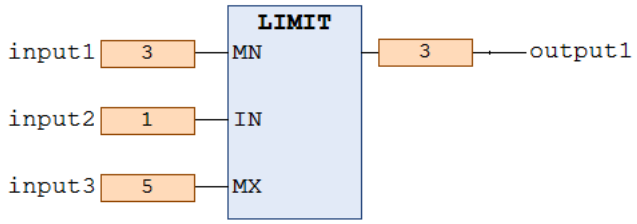
Scope	Name	Type	Description
Input	MN	All	Specifies the lower limit of the value to be output.
	IN	All	Specifies the input values to be restricted.
	MX	All	Specifies the upper limit of the value to be output.
Output	-	All	Outputs values according to the following conditions. IN ≤ MN: Outputs "MN". MN ≤ IN ≤ MX: Outputs "IN". MX ≤ IN: Outputs "MX".

■ Program example

This program is designed to limit the value range of the input variable “input2” with the input variable “input1” (lower limit) and the input variable “input3” (upper limit) and to output the limited value to the output variable “output1”.

LD program

This program is designed to output “3” to “output1” because the value (1) of “input2” (IN) is less than or equal to the lower limit (3) specified in “input1” (MN).



ST program

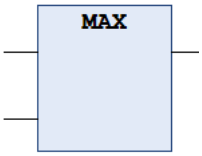
This program is designed to output “5” to “output1” because the value (8) of “input2” is greater than or equal to the upper limit (5) specified in “input3”.

```
output1 5 := LIMIT(input1 3, input2 8, input3 5);
```

3.9.4 MAX (Maximum Value)

This is a function that outputs the maximum value of the input arguments.

■ Icon



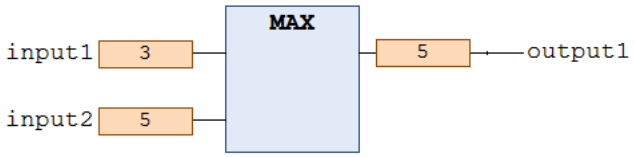
■ Parameter

Scope	Type	Description
Input	All	Specifies the values from which to obtain the maximum value.
Output	All	Outputs the maximum value of the input values.

■ Program example

This program is designed to output the maximum value of the input variables to the output variable “output1”.

LD program



ST program

```
output1 5  := MAX(input1 3 , input2 5 );
```

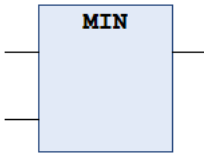
Info.

- If you want to increase input arguments in the LD program, right-click on the MAX function, and, on the displayed menu, select "Add Input".

3.9.5 MIN (Minimum Value)

This is a function that outputs the minimum value of the input arguments.

■ Icon



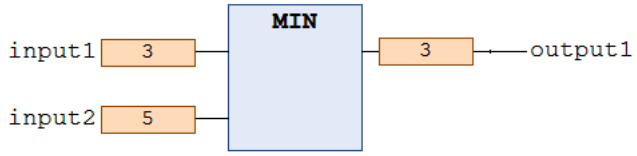
■ Parameter

Scope	Type	Description
Input	All	Specifies the values from which to obtain the minimum value.
Output	All	Outputs the minimum value of the input values.

■ Program example

This program is designed to output the minimum value of the input variables to the output variable “output1”.

LD program



ST program

```
output1 3 := MIN(input1 3 , input2 5 );
```

Info.

- If you want to increase input arguments in the LD program, right-click on the MIN function, and, on the displayed menu, select "Add Input".

3.9.6 MEMUtils.Swap (Byte Swapping)

This is a function that swaps specified bytes in order at a specified pointer to data in units of byte. The numbers of bytes that can be swapped are 2 bytes, 4 bytes, and 8 bytes.

■ Icon

■ Parameter

Scope	Name	Type	Description
Input	puc	POINTER TO BYTE	Start pointer at which byte swapping starts
Input	iSize	INT	The number of bytes to swap. Allowable inputs: 10#2/10#4/10#8
Output	Swap	INT	Status Successfully complete = 1 Error = -1

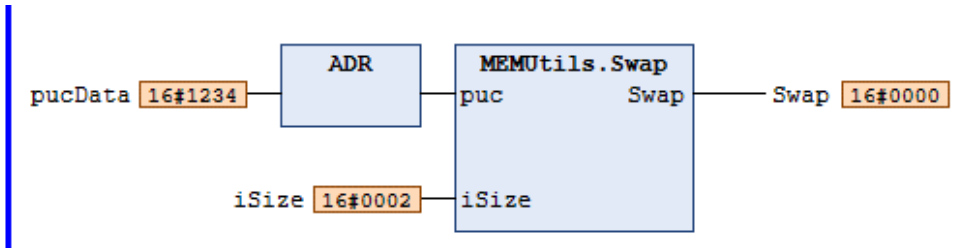
■ Program example

This program is designed to swap 2 bytes (iSize) in order in source data (pucData) in units of byte.

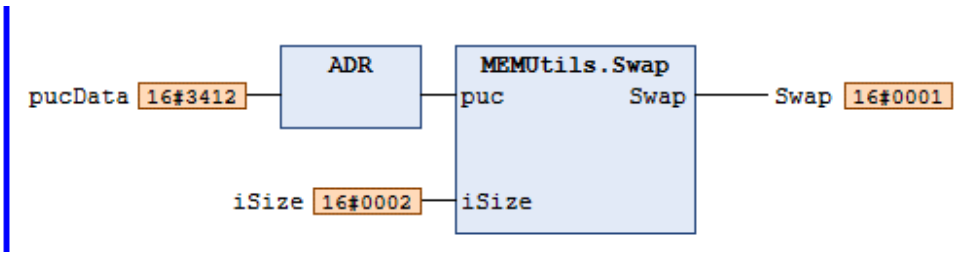
pucData := 16#1234 (Data to swap)

iSize := 2

LD program



Execution result



ST program

```
Swap 16#0001 := MEMUtils.Swap(ADR(pucData 16#1234 ),2);
```

Execution result

```
Swap_before 16#1234 ;
Swap_after 16#3412 ;
```

Note

- If iSize is set to a value other than 2/4/8, byte swapping will not be carried out and the return value (Swap) of the function will be -1.

3.9.7 MEM.Compare (Memory Comparison)

This is a function that compares two specified memory block data pieces. When the memory block data pieces match each other, the function outputs 0. If they do not match, the function outputs the first location at which they differ.

■ Icon

■ Parameter

Scope	Name	Type	Description
Input	pMemoryBlockA	POINTER TO BYTE	Start pointer to data A to compare
Input	pMemoryBlockB	POINTER TO BYTE	Start pointer to data B to compare
Input	uiNumberOfBytes	UINT	Number of data bytes to compare Effective range: 10#1 to 10#65534
Output	Compare	UINT	0 = Data pieces match / Other = First location (BYTE) at which data pieces differ

■ Program example

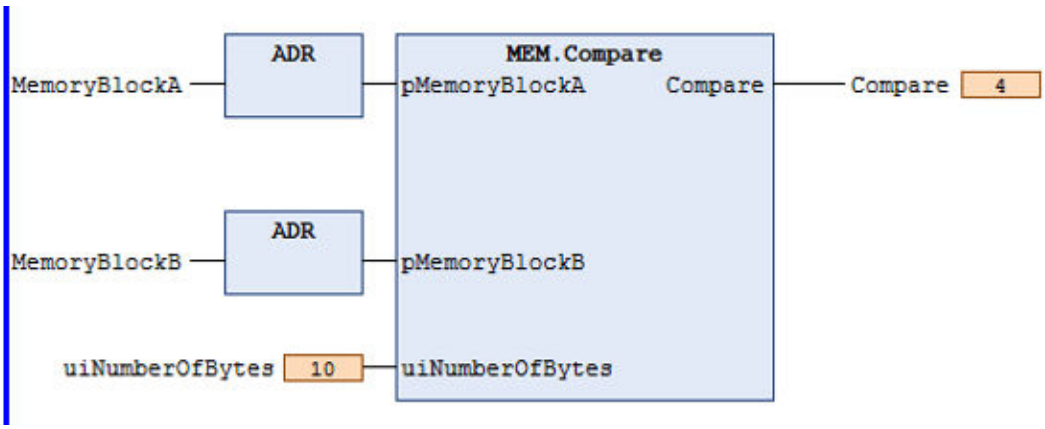
This program is designed to compare two specified pieces of BYTE type data (MemoryBlockA/ MemoryBlockB).

MemoryBlockA : ARRAY[0..9] OF BYTE := 0,1,2,3,4,5,6,7,8,9

MemoryBlockB : ARRAY[0..9] OF BYTE := 0,1,2,0,4,5,0,7,8,9

uiNumberOfBytes := 10

LD program



ST program

```
Compare   4   :=MEM.Compare(ADR(MemoryBlockA),ADR(MemoryBlockB),10);
```

Note

- If the number of data bytes to compare (uiNumberOfBytes) is 0, the function does not compare the two data pieces and returns 0.
- The function does not operate properly if the number of data bytes to compare (uiNumberOfBytes) is set to 65535 bytes. Thus, do not use that byte size.

3.9.8 MEM.FindBlock(Memory block search)

This is a function that searches memory block data A for memory block data B. If the target data is found, the function outputs the location at which the target data starts. If the target data is not found, the function outputs 0.

■ Icon

■ Parameter

Scope	Name	Type	Description
Input	pMemoryBlockA	POINTER TO BYTE	Start pointer to the data to search
Input	uiLengthBlockA	UINT	Number of bytes of the data to search Effective range: 10#1 to 10#65535
Input	pMemoryBlockB	POINTER TO BYTE	Start pointer to the data to find
Input	uiLengthBlockB	UINT	Number of bytes of the data to find Effective range: 10#1 to 10#65535
Output	FindBlock	UINT	0 = Data not found / Other = Location (BYTE) at which the found data starts

■ Program example

This program is designed to search specified BYTE type data (MemoryBlockA) for specified BYTE type data (MemoryBlockB).

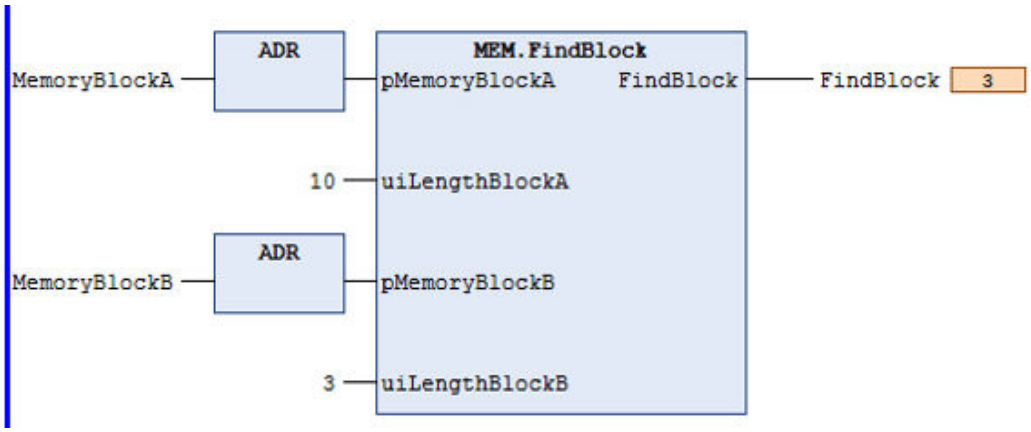
pMemoryBlockA : ARRAY[0..9] OF BYTE := 1,2,3,4,5,1,2,3,4,5

pMemoryBlockB : ARRAY[0..2] OF BYTE := 3,4,5

uiLengthBlockA := 10

uiLengthBlockB := 3

LD program



ST program

```
FindBlock 3 :=MEM.FindBlock(ADR(MemoryBlockA),10,ADR(MemoryBlockB),3);
```

Note

- Do not use this function with 0 set in the number of bytes of the data to find (uiLengthBlockB). If uiLengthBlockB = 0, the return value of the function is 16#FFFF.

3.9.9 MEM.FindByte (Find Byte Data)

This is a function that searches specified memory block data for specified one-byte data. If the target data is found, the function outputs the location at which the target data starts. If the target data is not found, the function outputs 0.

■ Icon

■ Parameter

Parameter	Name	Type	Description
Input	pMemoryBlock	POINTER TO BYTE	Start pointer to the data to search
Input	uiLength	UINT	Number of bytes of the data to search Effective range: 10#1 to 10#65534
Input	byValue	BYTE	Data to find Effective range: 10#0 to 10#255

Parameter	Name	Type	Description
Output	FindByte	UINT	0 = Data not found / Other = Location (BYTE) at which the found data starts

■ Program example

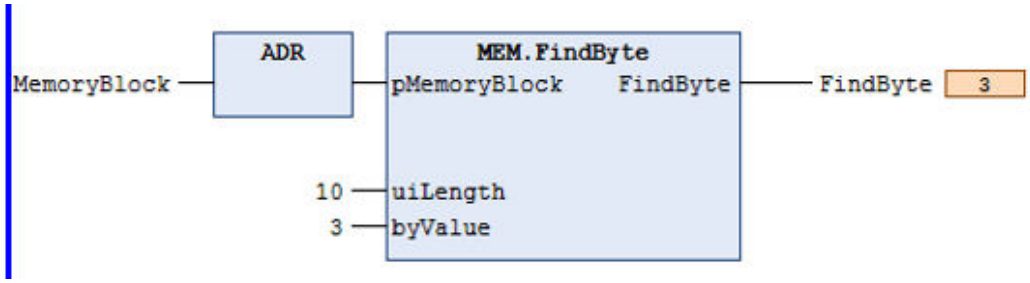
This program is designed to search specified BYTE type data (MemoryBlock) for BYTE type data (byValue).

pMemoryBlock : ARRAY[0..9] OF BYTE := 1,2,3,4,5,1,2,3,4,5

byValue := 3

uiLength := 10

LD program



ST program

```
FindByte 3 :=MEM.FindByte(ADR(MemoryBlock),10,3);
```

Note

- If the number of bytes of the data to search (uiLength) is 0, the function does not search the data and returns 0.
- The function does not operate properly if the number of bytes of the data to search (uiLength) is set to 65535 bytes. Thus, do not use that byte size.

3.9.10 MEM.MemFill (Memory Fill)

This is a function that fills a specified size in data memory with a specified data value.

■ Icon

■ Parameter

MemFill

Scope	Name	Type	Description
Input	pMemoryBlock	POINTER TO BYTE	Starting address of data to fill
Input	uiLength	UINT	Number of bytes to fill Effective range: 10#1 to 65534
Input	byFillValue	BYTE	Data value with which to fill the data Effective range: 10#0 to 255
Output	MemFill	BOOL	TRUE = Filling completed

■ Program example

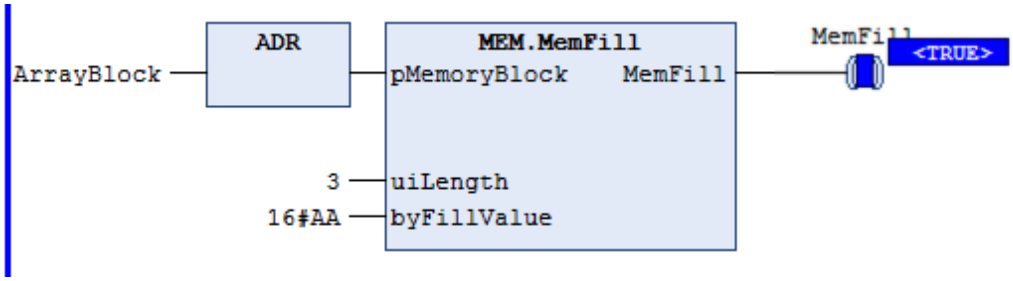
This program is designed to fill a three-byte size (uiLength) in data to fill (ArrayBlock) with the 16#AA data value (byFillValue).

ArrayBlock : ARRAY [0..4] OF BYTE := [5(0)] (data to fill: default value)

uiLength := 10#3

byFillValue := 16#AA (data value with which to fill the data)

LD program



ST program

```
MemFill TRUE := MEM.MemFill(ADR(ArrayBlock[0] 16#AA ),3,16#AA);

ArrayBlock[0] 16#AA ;
ArrayBlock[1] 16#AA ;
ArrayBlock[2] 16#AA ;
ArrayBlock[3] 16#00 ;
ArrayBlock[4] 16#00 ;
```

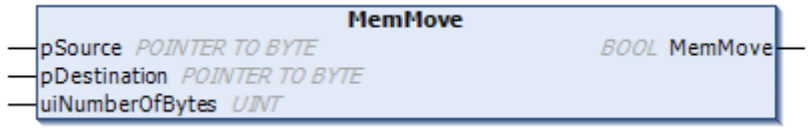
Note

- The function does not operate properly if the number of bytes to fill (uiLength) is set to the maximum 65535 bytes. Thus, do not use that byte size.
- If the number of bytes to fill (uiLength) is 0, the data filling will not be carried out.
- If the start pointer to data to fill (pMemoryBlock) is set to 0 (NULL), the function returns FALSE.

3.9.11 MEM.MemMove (Memory Copying)

This is a function that copies a specified size in data memory onto copy destination data memory.

■ Icon



■ Parameter

MemMove

Scope	Name	Type	Description
Input	pSource	POINTER TO BYTE	Copy source data starting address
Input	pDestination	POINTER TO BYTE	Copy destination data starting address
Input	uiNumberOfBytes	UINT	Number of bytes to copy Effective range: 10#1 to 65534
Output	MemMove	BOOL	TRUE = Copying completed

■ Program example

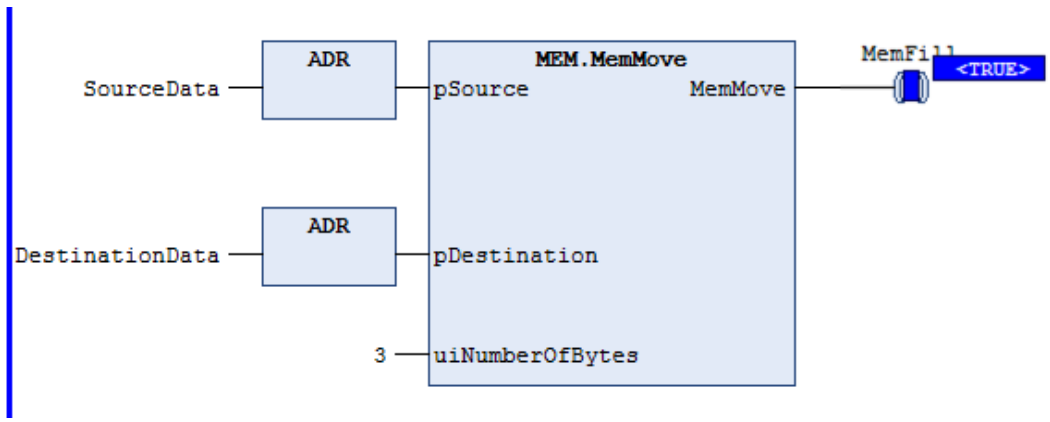
This program is designed to copy 3 bytes (uiNumberOfBytes) in copy source data (SourceData) onto copy destination data (DestinationData).

SourceData : ARRAY [0..4] OF BYTE := [1,2,3,4,5] (Copy source data)

DestinationData : ARRAY [0..4] OF BYTE := [5(0)] (Copy destination data: default value)

uiNumberOfBytes := 3

LD program



ST program

```
MemMove TRUE := MEM.MemMove(ADR(SourceData), ADR(DestinationData),3);

SourceData[0] 16#01;      DestinationData[0] 16#01;
SourceData[1] 16#02;      DestinationData[1] 16#02;
SourceData[2] 16#03;      DestinationData[2] 16#03;
SourceData[3] 16#04;      DestinationData[3] 16#00;
SourceData[4] 16#05;      DestinationData[4] 16#00;
```

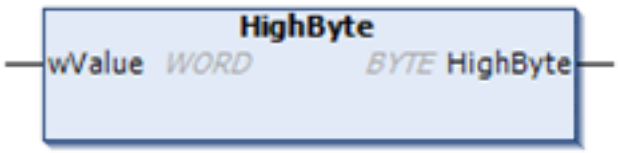
Note

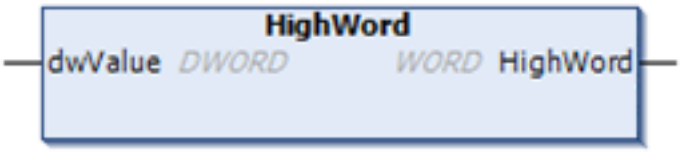
- The function does not operate properly if the number of bytes to copy (uiNumberOfBytes) is set to the maximum 65535 bytes. Thus, do not use that byte size.
- If the number of bytes to copy (uiNumberOfBytes) is 0, the copying will not be carried out.
- If any of the start pointer to copy source data (pSource) and the start pointer to copy destination data (pDestination) are set to 0 (NULL), the function returns FALSE.
- If copying in units of bit is required, use the function in ”3.12.4 MEMUtils.BitCpy”.

3.9.12 EM.High** (High Byte/High WORD Extraction)

This is a function that outputs high byte / high WORD of the input value.

■ Icon





■ Parameter

HighByte

Scope	Name	Type	Description
Input	wValue	WORD	Input value of WORD type
Output	HighByte	BYTE	Outputs high byte of the input value

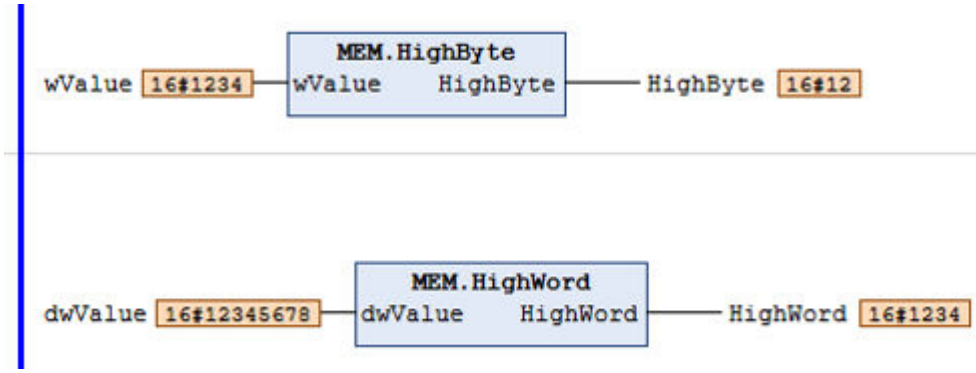
HighWord

Scope	Name	Type	Description
Input	dwValue	DWORD	Input value of DWORD type
Output	HighWord	WORD	Outputs high WORD of the input value

■ Program example

This program is designed to output the high byte (16#12) of the wValue input variable (16#1234) of the WORD type to the HighByte output variable of the BYTE type.
This program is designed to output the high WORD (16#1234) of the dwValue input variable (16#12345678) of the DWORD type to the HighWord output variable of the WORD type.

LD program



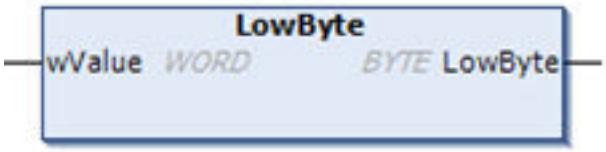
ST program

```
HighByte 16#12 :=MEM.HighByte(wValue 16#1234 );

HighWord 16#1234 :=MEM.HighWord(dwValue 16#12345678 );
```

3.9.13 MEM.Low** (Low Byte/Low WORD Extraction)

This is a function that outputs low byte / low WORD of the input value.

■ Icon



■ Parameter

LowByte

Scope	Name	Type	Description
Input	wValue	WORD	Input value of WORD type
Output	LowByte	BYTE	Outputs low byte of the input value

LowWord

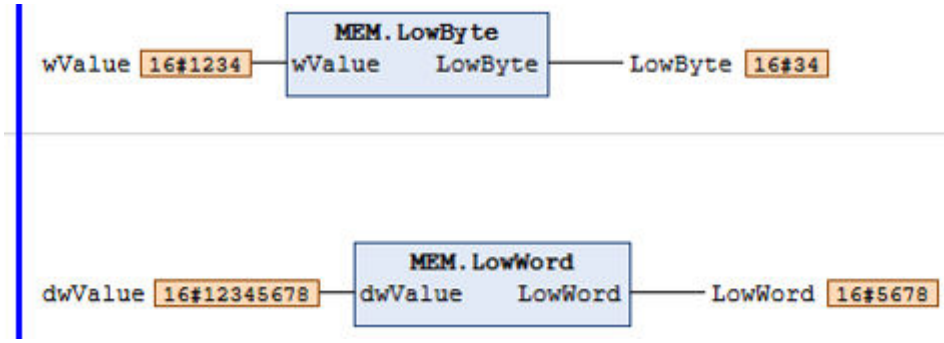
Scope	Name	Type	Description
Input	dwValue	DWORD	Input value of DWORD type
Output	LowWord	WORD	Outputs low WORD of the input value

■ Program example

This program is designed to output the low byte (16#34) of the wValue input variable (16#1234) of the WORD type to the LowByte output variable of the BYTE type.

This program is designed to output the low WORD (16#5678) of the dwValue input variable (16#12345678) of the DWORD type to the LowWord output variable of the WORD type.

LD program



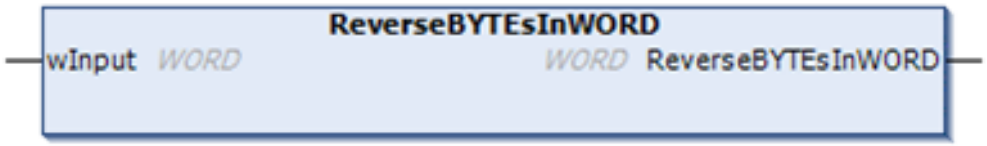
ST program

```
LowByte 16#34 :=MEM.LowByte(wValue 16#1234 );

LowWord 16#5678 :=MEM.LowWord(dwValue 16#12345678 );
```

3.9.14 MEM.ReverseBYTEsIn** (Byte Order Change)

This is a function that reverses the order of the bytes of input WORD-, or DWORD-type data and outputs the data of the bytes in reverse order.

■ Icon



■ Parameter

ReverseBYTEsInWORD

Scope	Name	Type	Description
Input	wInput	WORD	Input value, WORD type data
Output	ReverseBYTEsInWORD	WORD	Value in reverse byte order

ReverseBYTEsInDWORD

Scope	Name	Type	Description
Input	dwInput	DWORD	Input value, DWORD type data
Output	ReverseBYTEsInDWORD	DWORD	Value in reverse byte order

■ Program example

- ReverseBYTEsInDWORD

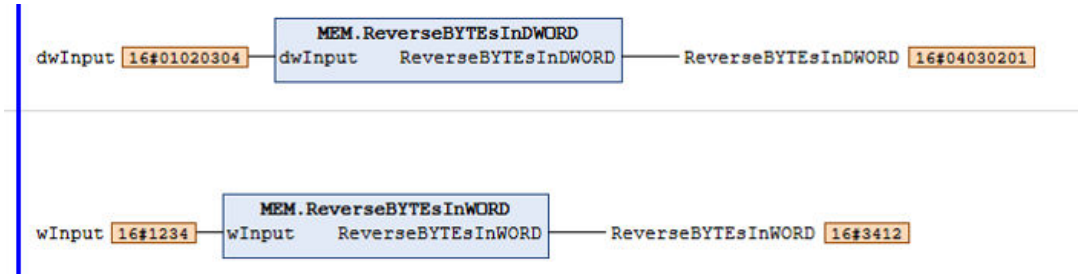
  This program is designed to reverse the order of bytes of the dwInput input variable of the DWORD type and outputs the data of the bytes in reverse order to the ReverseBYTEsInDWORD output variable of the DWORD type.

  dwInput := 16#01020304
- ReverseBYTEsInWORD

  This program is designed to reverse the order of bytes of the wInput input variable of the WORD type and outputs the data of the bytes in reverse order to the ReverseBYTEsInWORD output variable of the WORD type.

wInput := 16#1234

LD program



ST program

```
ReverseBYTEsInDWORD 16#04030201 :=MEM.ReverseBYTEsInDWORD(dwInput 16#01020304 );

ReverseBYTEsInWORD 16#3412 :=MEM.ReverseBYTEsInWORD(wInput 16#1234 );
```

3.9.15 MEM.ReverseWORDsInDWORD (WORD Order Change)

This is a function that reverses the order of the WORDs of input DWORD-type data and outputs the data of the WORDs in reverse order.

■ Icon

■ Parameter

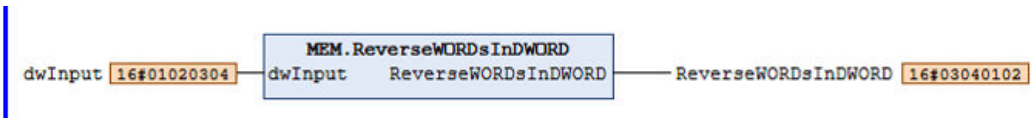
Scope	Name	Type	Description
Input	dwInput	DWORD	Input value, DWORD type data
Output	ReverseWORDsInDWORD	DWORD	Value in reverse WORD order

■ Program example

This program is designed to reverse the order of WORDs of the dwInput input variable of the DWORD type and outputs the data of the WORDs in reverse order to the ReverseWORDsInDWORD output variable of the DWORD type.

dwInput := 16#01020304

LD program



ST program

```
ReverseWORDsInDWORD 16#03040102 :=MEM.ReverseWORDsInDWORD(dwInput 16#01020304 );
```

3.10 Character string instructions

Character string instructions can be used to perform various operations on character strings.

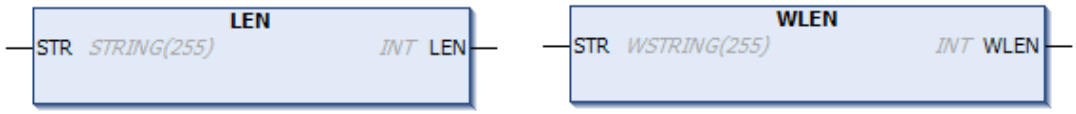
There is no limit to the length of a STRING type string, but the string functions described in this chapter only process lengths of 1 to 255 characters.

Do not use a string longer than 256 characters in the function input.

3.10.1 LEN/WLEN (string length)

This is a function that outputs the length of a character string.

■ Icon



■ Parameter

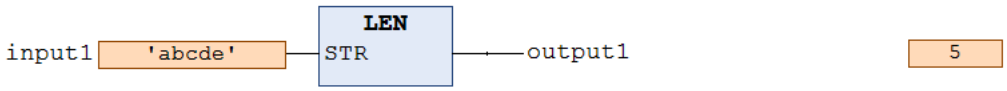
Scope	Name	Type	Description
Input	STR	STRING(255)/ WSTRING(255)	Specifies the character string from which to obtain the length.
Output	LEN/WLEN	INT	Outputs the character string length of the input argument.

■ Program example

This program is designed to output the character string length of the input variable “input1” to the output variable ”output1”.

This is a program example for the function LEN.

LD program

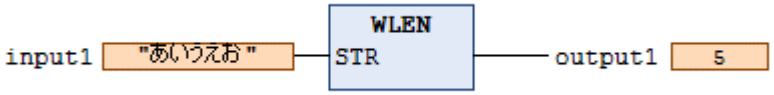


ST program

```
output1 5 := LEN(input1 'abcde');
```

This is a program example for the function WLEN.

LD program



ST program

```
output1  5  :=WLEN(input1  "あいうえお"  );
```

3.10.2 LEFT/WLEFT (extract text from left edge)

This is a function that extracts a character string consisting of the specified number of characters from the left end of the character string and outputs the extracted data.

■ Icon

■ Parameter

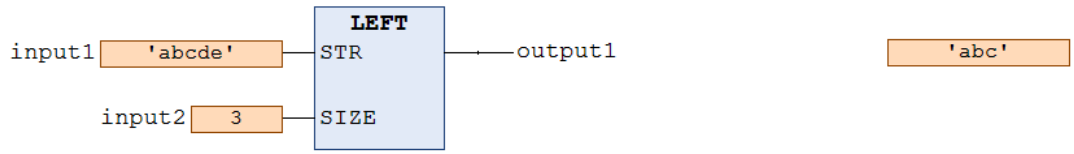
Scope	Name	Type	Description
Input	STR	STRING(255)/ WSTRING(255)	Specifies the character string from which a character string is to be extracted.
	SIZE	INT	Specifies the number of characters to be extracted from the left.
Output	LEFT/ WLEFT	STRING(255)/ WSTRING(255)	Extracts a character string consisting of the number of characters specified in SIZE from STR and outputs the extracted data.

■ Program example

This program extracts the character string of the number of characters (3 characters) specified by input2 from the character string of the input variable input1 from the left end and outputs it to the output variable output1.

This is a program example for the function LEFT.

LD program

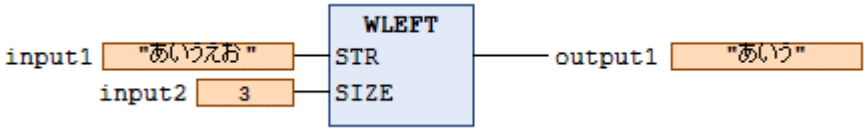


ST program

```
output1  'abc'  := LEFT(input1  'abcde' , input2  3 );
```

This is a program example for the function WLEFT.

LD program



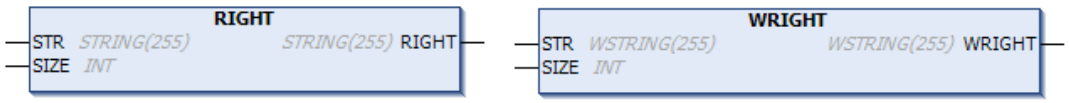
ST program

```
output1 "あいう" :=WLEFT(input1 "あいうえお",input2 3);
```

3.10.3 RIGHT/WRIGHT (Extract text from the right end)

This is a function that extracts a character string consisting of the specified number of characters from the right end of the character string and outputs the extracted data.

■ Icon



■ Parameter

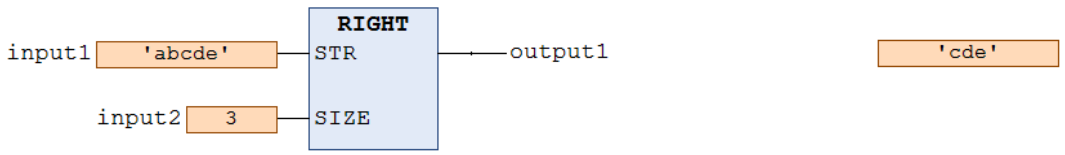
Scope	Name	Type	Description
Input	STR	STRING(255)/ WSTRING(255)	Specifies the character string from which a character string is to be extracted.
	SIZE	INT	Specifies the number of characters to be extracted from the right.
Output	RIGHT/ WRIGHT	STRING(255)/ WSTRING(255)	Extracts a character string consisting of the number of characters specified in SIZE from STR and outputs the extracted data.

■ Program example

This program is designed to extract a character string consisting of the number of characters (3 characters) specified in “input2” from the right end of the character string of the input variable “input1” and to output the extracted character string to the output variable ”output1”.

This is a program example for the function RIGHT.

LD program

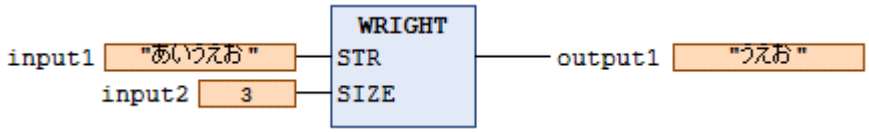


ST program

```
output1 'cde' := RIGHT(input1 'abcde', input2 3);
```

This is a program example for the function WRIGHT.

LD program



ST program

```
output1 "うえお" :=WRIGHT(input1 "あいうえお",input2 3);
```

3.10.4 MID/WMID (extract string from specified position)

This function extracts a specified number of characters from the right end of a character string and outputs it.

■ Icon

■ Parameter

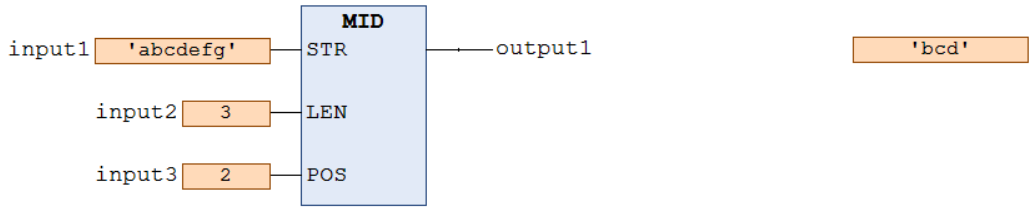
Scope	Name	Type	Description
Input	STR	STRING(255)/ WSTRING(255)	Specifies the character string from which a character string is to be extracted.
	LEN	INT	Specifies the number of characters to be extracted.
	POS	INT	Specified the position from which extraction is to be started.
Output	MID/WMID	STRING(255)/ WSTRING(255)	Extracts a character string consisting of the number of characters specified in LEN from STR starting from the position specified in POS and outputs the extracted data.

■ Program example

This program is designed to extract a character string consisting of the number of characters (3 characters) specified in “input2” from the character string of the input variable “input1”, starting from the position (2nd character from the left end) specified in “input3”, and to output the extracted data to the output variable ”output1”.

This is a program example for the function MID.

LD program

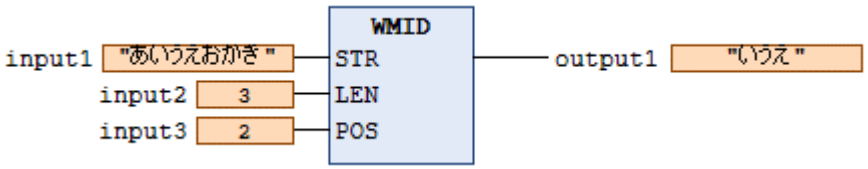


ST program

```
output1 'bcd' := MID(input1 'abcdefg', input2 3, input3 2);
```

This is a program example for the function WMID.

LD program



ST program

```
output1 "いうえ" :=WMID(input1 "あいうえおかき",input2 3,input3 2);
```

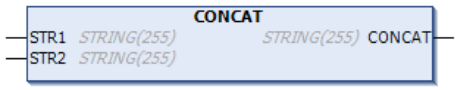
Info.

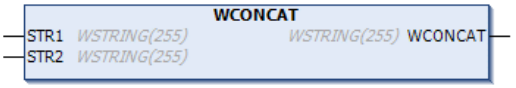
If POS = 0, it is not extracted.

3.10.5 CONCAT/WCONCAT (string concatenation)

This is a function that concatenates the character strings.

■ Icon





■ Parameter

Scope	Name	Type	Description
Input	STR1	STRING(255)/ WSTRING(255)	Specifies the character string to be concatenated.
	STR2	STRING(255)/ WSTRING(255)	Specifies the character string to be concatenated.

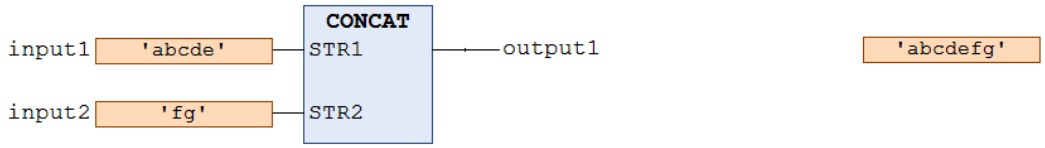
Scope	Name	Type	Description
Output	CONCAT/ WCONCAT	STRING(255)/ WSTRING(255)	Concatenate the STR2 character string to the right of the STR1 character string and output the concatenated data.

■ Program example

This program is designed to concatenate the character string of “input2” to the character string of the input variable “input1” and to output the concatenated data to the output variable ”output1”.

This is a program example for the function CONCAT.

LD program

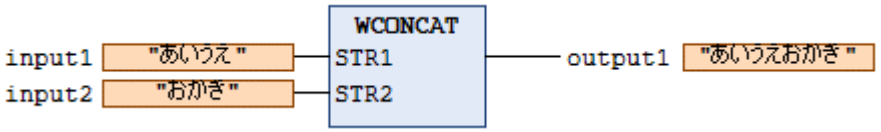


ST program

```
output1 'abcdefg' := CONCAT(input1 'abcde', input2 'fg');
```

This is a program example for the function WCONCAT.

LD program



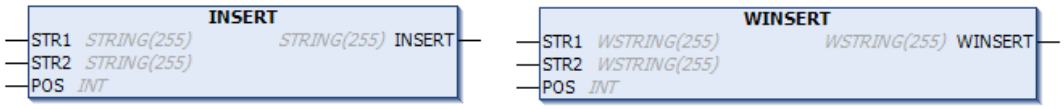
ST program

```
output1 "あいうえおかき" :=WCONCAT(input1 "あいうえ",input2 "おかき");
```

3.10.6 INSERT/WINSERT (Inserting a Character String)

This is a function that inserts a character string in the specified position and outputs the inserted data.

■ Icon



■ Parameter

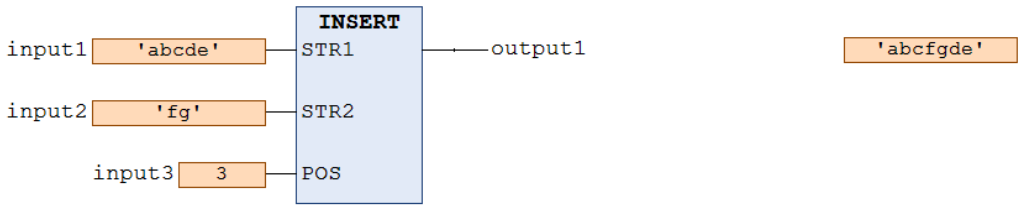
Scope	Name	Type	Description
Input	STR1	STRING(255)/ WSTRING(255)	Specifies the character string in which a character string is to be inserted.
	STR2	STRING(255)/ WSTRING(255)	Specifies the character string to be inserted.
	POS	INT	Specifies the position to be inserted. n-th character from the left
Output	INSERT/ WINSERT	STRING(255)/ WSTRING(255)	Insert the string of STR2 into the position of POS in the string of STR1 and output

■ Program example

This program is designed to insert the character string of “input2” in the position (3rd character from the left end) specified in “input3” from the left of the the character string of the input variable “input1” and to output the inserted data to the output variable ”output1”.

This is a program example for the function INSERT.

LD program

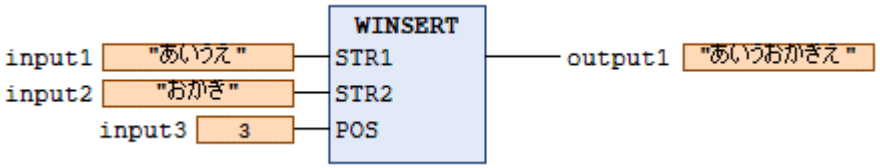


ST program

```
output1 'abcfgde' := INSERT(input1 'abcde', input2 'fg', input3 3);
```

This is a program example for the function WINSERT.

LD program



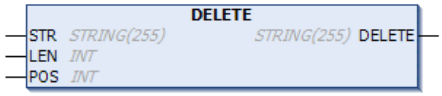
ST program

```
output1 "あいうおかきえ" :=WINSERT(input1 "あいうえ",input2 "おかき",input3 3);
```

3.10.7 DELETE/WDELETE (delete string)

This is a function that deletes a character string from the specified position and outputs the deleted data.

■ Icon



■ Parameter

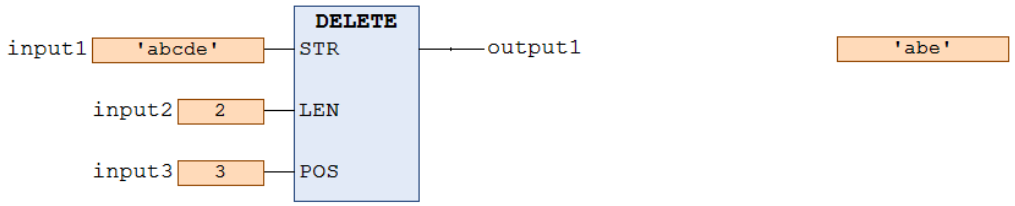
Scope	Name	Type	Description
Input	STR	STRING(255)/ WSTRING(255)	Specifies the character string from which a character string is to be deleted.
	LEN	INT	Specifies the length of the character string to be deleted.
	POS	INT	Specified the position from which deletion is to be started. n-th character from the left
Output	DELETE/ WDELETE	STRING(255)/ WSTRING(255)	Deletes a character string consisting of the number of characters specified in LEN from the left end of the STR character string starting from the position specified in POS and outputs the deleted data.

■ Program example

This program is designed to delete a character string consisting of the number of characters (2 characters) specified in “input2” from the character string of the input variable “input1” starting from the position (3rd character from the left) specified in “input3” and to output the deleted data to the output variable “output1”.

This is a program example for the function DELETE.

LD program

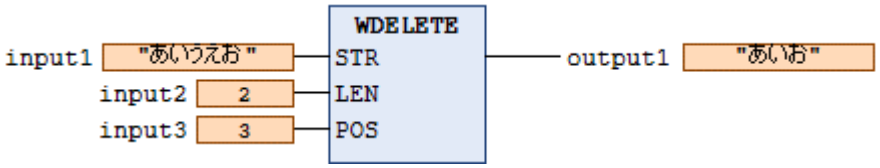


ST program

```
output1 'abe' := DELETE(input1 'abcde', input2 2, input3 3);
```

This is a program example for the function WDELETE.

LD program



ST program

```
output1 "あいお" :=WDELETE(input1 "あいうえお", input2 2, input3 3);
```

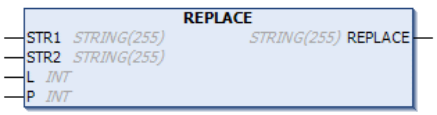
Info.

If POS = 0, LEN is used with a setting of -1.

3.10.8 REPLACE/WREPLACE (replace string)

This is a function that replaces the character strings and outputs the replaced character strings.

■ Icon



■ Parameter

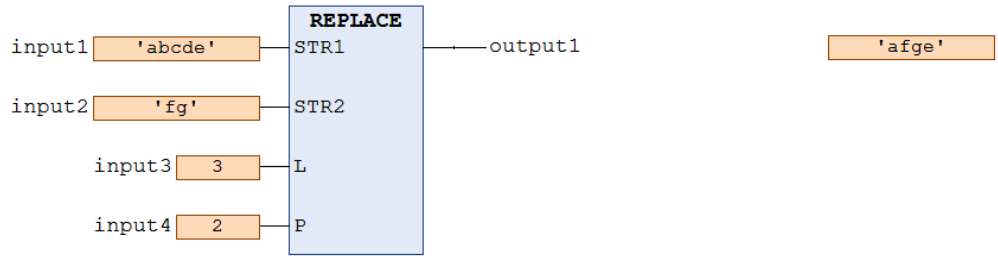
Scope	Name	Type	Description
Input	STR1	STRING(255)/ WSTRING(255)	Specifies the character string to be replaced.
	STR2	STRING(255)/ WSTRING(255)	Specifies the character string to be added by replacement.
	L	INT	Specifies the number of characters to be deleted by replacement.
	P	INT	Specify where to add STR2 text by substitution
Output	REPLACE/ WREPLACE	STRING(255)/ WSTRING(255)	Replaces the number of characters specified in L with the character string specified in STR2 from the left end of the character string specified in STR1 starting from the position specified in P and outputs the replaced data.

■ Program example

This program is designed to replace a character string consisting of the number of characters specified in “input3” with the character string specified in “input2” from the position specified in “input4” in the character string of the input variable “input1” and to output the replaced data to the output variable “output1”.

This is a program example for the function REPLACE.

LD program

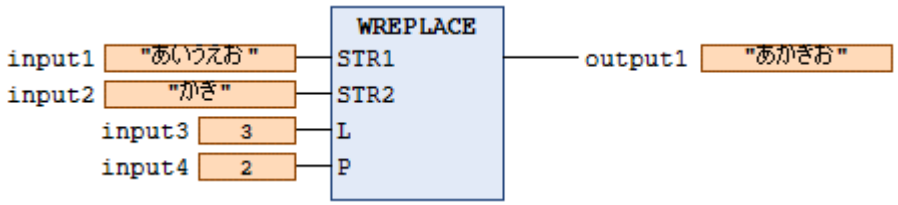


ST program

```
output1 'afge' := REPLACE(input1 'abcde', input2 'fg',
                          input3 3, input4 2);
```

This is a program example for the function WREPLACE.

LD program



ST program

```
output1 "あかきお" :=WREPLACE(input1 "あいうえお", input2 "かき", input3 3, input4 2);
```

Info.

If P = 0, L is used with a setting of -1.

3.10.9 FIND/WFIND (find text)

This is a function that searches for a specified character string and outputs the searched position.

■ **Icon**

■ Parameter

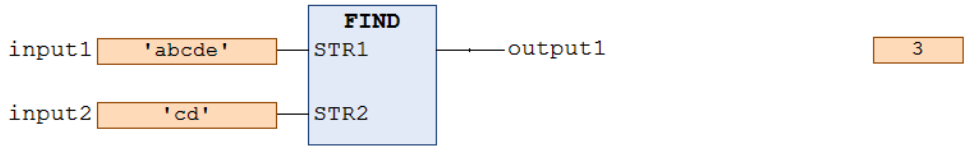
Scope	Name	Type	Description
Input	STR1	STRING(255)/ WSTRING(255)	Specify text to extract
	STR2	STRING(255)/ WSTRING(255)	Specifies the number of characters to extract
Output	FIND/ WFIND	INT	Searches for the character string specified in STR2 in the character string specified in STR1 and outputs the position from the left end.

■ Program example

This program is designed to search for the character string specified in “input2” in the character string of the input variable “input1” and to output the position from the left to the output variable ”output1”.

This is a program example for the function FIND.

LD program

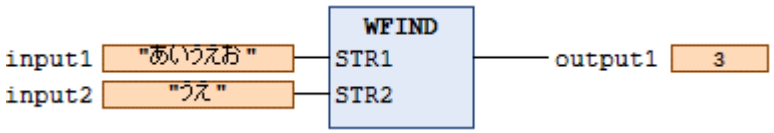


ST program

```
output1 3 := FIND(input1 'abcde', input2 'cd');
```

This is a program example for the function WFIND.

LD program



ST program

```
output1 3 :=WFIND(input1 "あいうえお",input2 "うえ");
```

Info.

- Outputs 0 if the character string is not found.
- If the character string is found in multiple places, the position found first (the leftmost position) is output.

3.10.10 ConvertUTF16toUTF8 (UTF-16 → UTF-8)

This is a function that converts a UTF-16 character string into a UTF-8 character string. Input a target storage size (dwTargetBufferSize) based on [(input WORD type data volume x 3) + 1(end code)].

■ Icon

■ Parameter

ConvertUTF16toUTF8

Scope	Name	Type	Description
Input	sourceStart	POINTER TO WORD	Start pointer to the UTF16 character string to be converted
Input	targetStart	POINTER TO BYTE	Start pointer to the converted UTF8 character string
Input	dwTargetBufferSize	DWORD	Target storage size (unit byte)
Input	bStrictConversion	BOOL	TRUE = An error is output when data that is not convertible is present TRUE = An error is output when data that is not convertible is present
Output	ConvertUTF16toUTF8	UDINT	Error identification (refer to ConvertUTF16toUTF8 return values)

ConvertUTF16toUTF8 return values

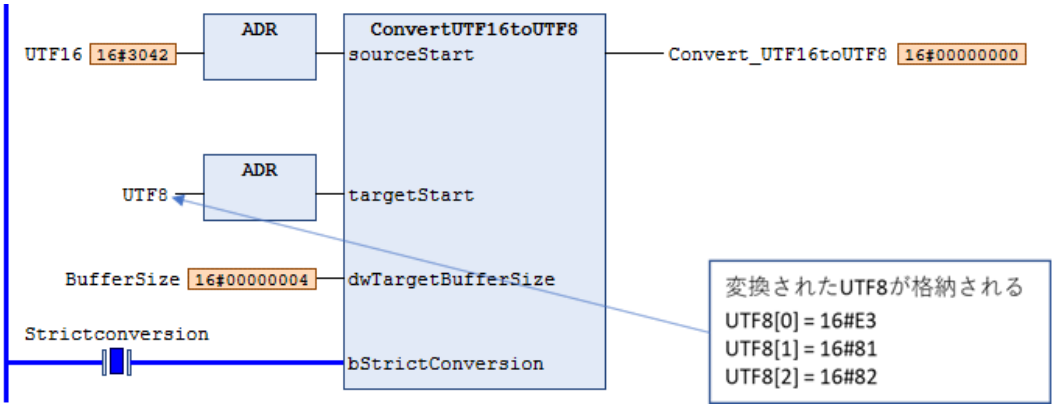
Return value	Name	Description
16#0000	ERR_OK	No error
16#0002	ERR_PARAMETER	Parameter error
16#40A1	ERR_TARGET_EXHAUSTED	Error in stored data buffer size
16#40A2	ERR_SOURCE_ILLEGAL	Data that is not convertible is present

■ Program example

This program is designed to convert a UTF16 input variable of the WORD type to a UTF8 character string.

```
UTF16 := 16#3042          (UTF16 that represents “あ”)
BufferSize := 4           (input WORD type data volume 1 WORD x 3) + 1 = 4
Strictconversion := TRUE
```

LD program



ST program

```
Convert_UTF16toUTF8 16#00000000 :=ConvertUTF16toUTF8(sourceStart:=ADR(UTF16),
                                  targetStart:=ADR(UTF8),
                                  dwTargetBufferSize:=BufferSize 16#00000004,
                                  bStrictConversion:=StrictConversion TRUE );

                                  UTF8_0 16#F08DFAC2 :=ADR(UTF8[0] 16#E3 );
                                  UTF8_1 16#F08DFAC3 :=ADR(UTF8[1] 16#81 );
                                  UTF8_2 16#F08DFAC4 :=ADR(UTF8[2] 16#82 );
```

Note

- Take care of input in the Unicode disuse / not used areas. (Otherwise, data may not be output properly.)
- Always set the error detection function for data that cannot be input (bStrictConversion) to TRUE.
- If the target storage buffer size (dwTargetBuffer) is not proper, return value = 16#40A1 is output.
- In the UTF16 data to be converted, 16#0000 serves as end-of-file (EOF). Thus, the UTF16 string from the start pointer (sourceStart) data to 16#0000 is converted.

3.10.11 ConvertUTF8toUTF16(UTF-8 → UTF-16)

This is a function that converts a UTF-8 character string into a UTF-16 character string. Input a target storage size (dwTargetBufferSize) based on [(input BYTE type data volume x 2) + 2(end code)].

■ Icon

■ Parameter

ConvertUTF8toUTF16

Scope	Name	Type	Description
Input	sourceStart	POINTER TO BYTE	Start pointer of the UTF8 character string to be converted
Input	targetStart	POINTER TO WORD	Start pointer of the converted UTF16 character string
Input	dwTargetBufferSize	DWORD	Target storage size (unit byte)
Input	bStrictConversion	BOOL	TRUE = An error is output when data that is not convertible is present FALSE = An error is not output even if data that is not convertible is present
Output	ConvertUTF8toUTF16	UDINT	Error identification (refer to ConvertUTF8toUTF16 return values)

ConvertUTF8toUTF16 return values

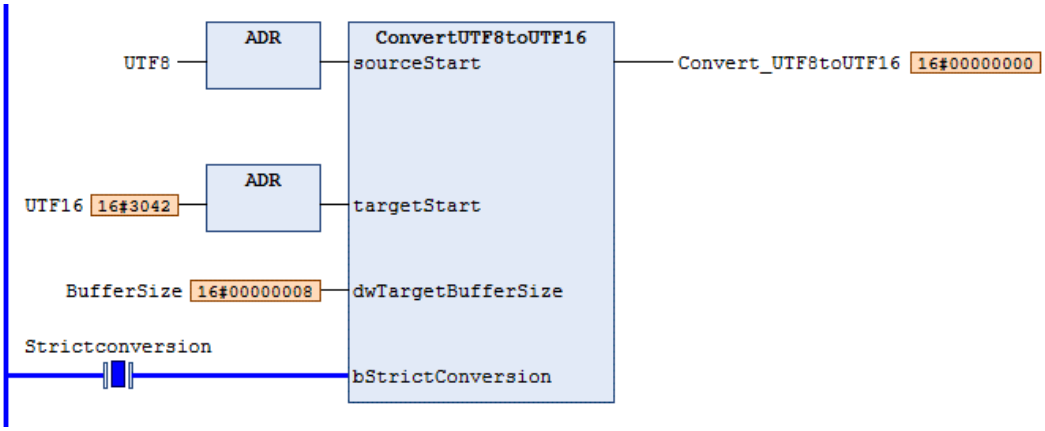
Return value	Name	Description
16#0000	ERR_OK	No error
16#0002	ERR_PARAMETER	Parameter error
16#40A1	ERR_TARGET_EXHAUSTED	Error in stored data buffer size
16#40A2	ERR_SOURCE_ILLEGAL	Data that is not convertible is present

■ Program example

This program is designed to convert a UTF8 input variable of the BYTE type to a UTF16 character string.

```
UTF8[0] := 16#E3        (UTF8 that represents "あ")
UTF8[1] := 16#81
UTF8[2] := 16#82
BufferSize := 8         (input BYTE type data volume 3 bytes x 2) + 2 = 8
Strictconversion := TRUE
```

LD program



ST program

```
Convert_UTF8toUTF16 16#00000000 :=ConvertUTF8toUTF16(sourceStart:=ADR(UTF8),
                                                     targetStart:=ADR(UTF16[0] 16#3042 ),
                                                     dwTargetBufferSize:=BufferSize 16#00000008 ,
                                                     bStrictConversion:=StrictConversion TRUE );

UTF8_0 16#F08DFADB :=ADR(UTF8[0] 16#E3 );
UTF8_1 16#F08DFADC :=ADR(UTF8[1] 16#81 );
UTF8_2 16#F08DFADD :=ADR(UTF8[2] 16#82 );
```

Note

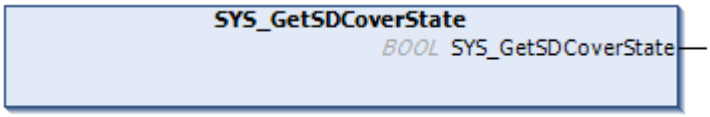
- Take care of input in the Unicode disuse / not used areas. (Otherwise, data may not be output properly.)
- Always set the error detection function for data that cannot be input (bStrictConversion) to TRUE.
- If a UTF8 character string that is not convertible is input, the 16#FFFD data is stored in the converted UTF16 data.
- With bStrictConversion = TRUE, an error (return value: 16#40A2) will occur in response to input of data that cannot be represented.
- If the target storage buffer size (dwTargetBuffer) is not proper, return value = 16#40A1 is output.
- In the UTF8 data to be converted, 16#00 serves as end-of-file (EOF). Thus, the UTF8 string from the start pointer (sourceStart) data to 16#00 is converted.

3.11 SD Memory Card Slot Instruction

3.11.1 SYS_GetSDCoverState (Get SD Card Cover Open / Close State)

This is a function that gets an open / close state of the card cover for the SD memory card slot.

■ Icon



■ Parameter

Scope	Name	Type	Description
Output	SYS_GetSDCoverState	BOOL	TRUE: The card cover is closed. FALSE: The card cover is open.

3.11.2 SYS_GetSDAccessRdy (Get SD Card Access Ready State)

This is a function block that gets the state whether an access to the SD memory card is allowed.

■ Icon

■ Parameter

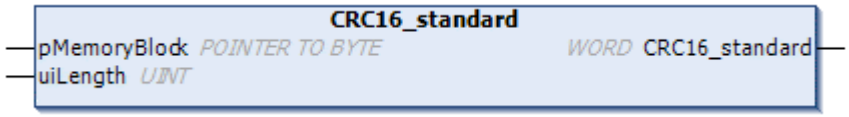
Scope	Name	Type	Description
Output	SYS_GestSDAccessRdy	BOOL	TRUE: Access to the SD memory card is enabled. FALSE: Access to the SD memory card is disabled.

3.12 CRC operation instructions

3.12.1 MEM.CRC16_standard (CRC16)

This is a function that calculates the CRC16 checksum.

■ **Icon**



■ **Parameter**

CRC16_standard

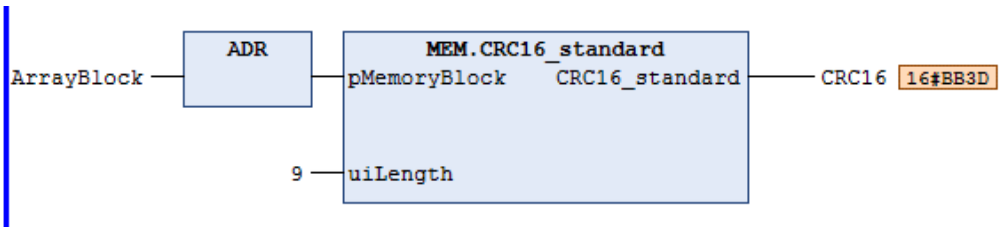
Scope	Name	Type	Description
Input	pMemoryBlock	POINTER TO BYTE	Start pointer to the memory block to calculate the checksum
Input	uiLength	UINT	Number of bytes to be calculated Effective range: 10#1 to 10#65534
Output	CRC16_standard	WORD	Calculated CRC16 result

■ **Program example**

This program is designed to calculate the CRC16 checksum of 9 bytes (uiLength) in memory block data (MemoryBlock) and output the result (16#BB3D) to CRC16_standard.

```
ArrayBlock := ARRAY [0..8] OF BYTE :   = [16#31,16#32,16#33,16#34,16#35,16#36,16#37,16#38,16#39]
                                        (= [STRING(10) := '123456789'])
uiLength :=9
```

LD program



ST program

```
CRC16 16#BB3D := MEM.CRC16_standard(ADR(ArrayBlock), 9);
```

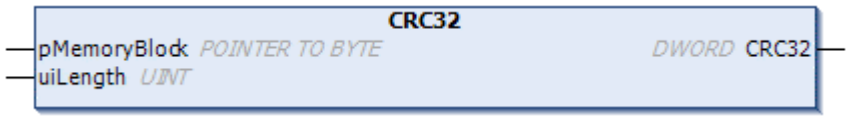
Note

- The function does not operate properly if the number of bytes to calculate (uiLength) is set to 65535 bytes or more. Thus, do not use that byte size.

3.12.2 MEM.CRC32(CRC32)

This is a function that calculates the CRC32 checksum.

■ Icon



■ Parameter

CRC32

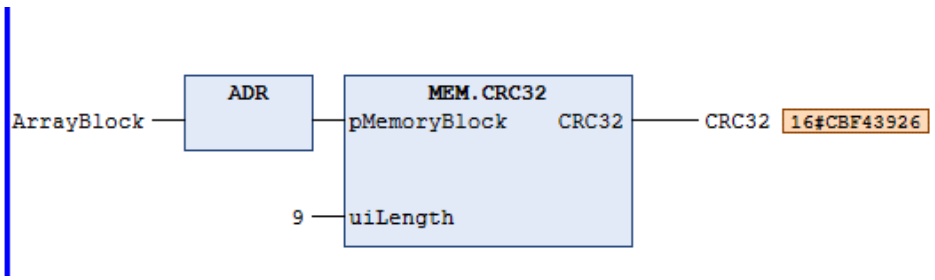
Scope	Name	Type	Description
Input	pMemoryBlock	POINTER TO BYTE	Start pointer to the memory block to calculate the checksum
Input	uiLength	UINT	Number of bytes to be calculated Effective range: 10#1 to 10#65534
Output	CRC32	DWORD	Calculated CRC32 result

■ Program example

This program is designed to calculate the CRC32 checksum of 9 bytes (uiLength) in memory block data (MemoryBlock) and output the result (16#CBF43926) to CRC32.

```
ArrayBlock := ARRAY [0..8] OF BYTE :  = [16#31,16#32,16#33,16#34,16#35,16#36,16#37,16#38,16#39]
                                        (= [STRING(10) := '123456789'])
uiLength :=9
```

LD program



ST program

```
CRC32 16#CBF43926 := MEM.CRC32(ADR(ArrayBlock),9);
```

Note

- The function does not operate properly if the number of bytes to calculate (uiLength) is set to 65535 bytes or more. Thus, do not use that byte size.

4 Function Blocks (Basic Instructions)

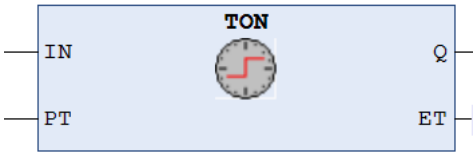
4.1 Timer Instructions

Timer instructions can be used to perform timer operations.

4.1.1 TON (Timer ON)

This is a function block (FB) that starts the timer when the input becomes TRUE. After a specified time elapses, the output becomes TRUE.

■ Icon



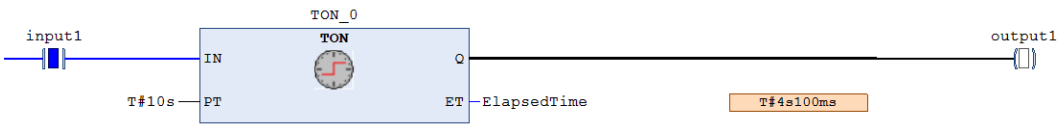
■ Parameter

Scope	Name	Type	Description
Input	IN	BOOL	Starts the timer when FALSE becomes TRUE and the timer continues counting while it remains TRUE. Resets the timer when it becomes FALSE.
	PT	TIME	Specifies the timer time.
Output	Q	BOOL	Outputs TRUE when the time specified in the input argument PT elapses.
	ET	TIME	Specifies the elapsed time of the timer.

■ Program example

This program is designed to start the timer when the input variable “input1” becomes TRUE and, after an elapse of 10 seconds, to cause the output variable “output1” to become TRUE. The instance name is TON_0.

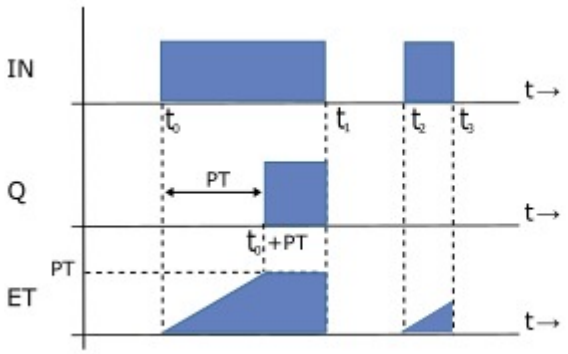
LD program



ST program

```
TON_0(IN TRUE := input1 TRUE, PT T#10s := T#10S,
      Q FALSE => output1 FALSE, ET T#3s659ms => ElapsedTime T#3s659ms);
```

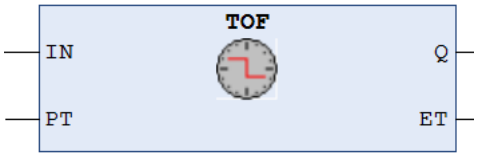
■ **Time-sequence diagram**



4.1.2 TOF (Timer OFF)

This is a function block (FB) that starts the timer when the input becomes FALSE. After a specified time elapses, the output becomes FALSE.

■ **Icon**



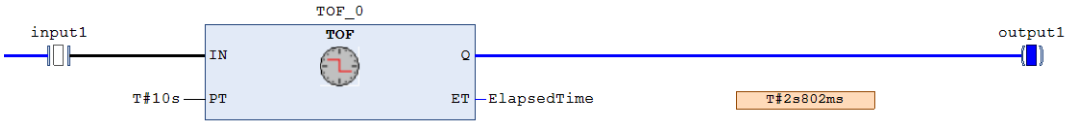
■ **Parameter**

Scope	Name	Type	Description
Input	IN	BOOL	Starts the timer when TRUE becomes FALSE and the timer continues counting while it remains FALSE. Resets the timer when it becomes TRUE.
	PT	TIME	Specifies the timer time.
Output	Q	BOOL	Outputs FALSE when the time specified in the input argument PT elapses.
	ET	TIME	Specifies the elapsed time of the timer.

■ **Program example**

This program is designed to start the timer when the input variable “input1” changes from TRUE to FALSE and, after an elapse of 10 seconds, to cause the output variable “output1” to become FALSE. The instance name is TOF_0.

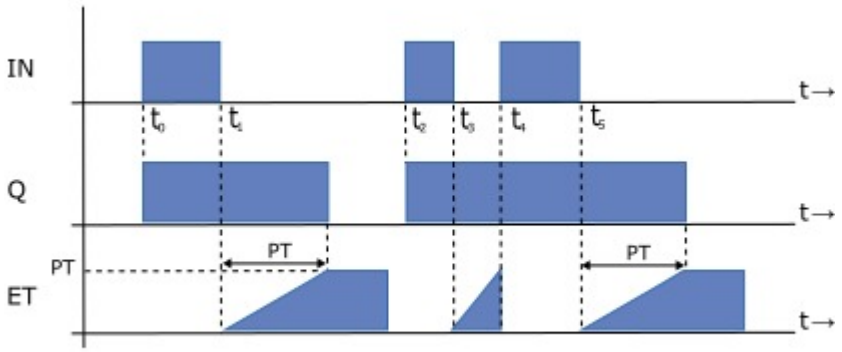
LD program



ST program

```
TOF_0(IN FALSE := input1 FALSE,
      PT T#10s := T#10S,
      Q TRUE => output1 TRUE,
      ET T#2s113ms => ElapsedTime T#2s113ms);
```

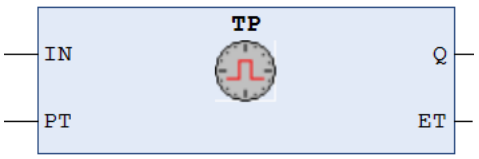
■ **Time-sequence diagram**



4.1.3 TP (Timer Pulse)

This is a function block that starts the timer at the rising edge. The output remains TRUE while the timer keeps counting. After a specified time elapses, the output becomes FALSE.

■ **Icon**



■ **Parameter**

Scope	Name	Type	Description
Input	IN	BOOL	Starts the timer when FALSE changes to TRUE (rising edge).

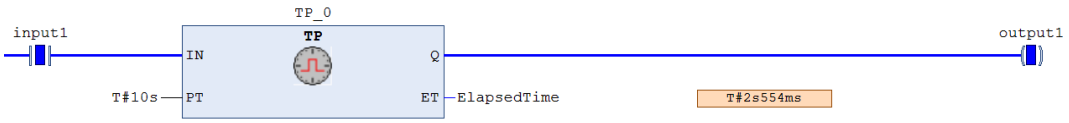
Scope	Name	Type	Description
			Resets the timer when the timer expires and TRUE changes to FALSE .
	PT	TIME	Specifies the timer time.
Output	Q	BOOL	Outputs TRUE from when the timer is started until when the time specified in the input argument PT elapses. Outputs FALSE after the specified time elapses.
	ET	TIME	Specifies the elapsed time of the timer.

■ Program example

This program is designed to start the timer when the input variable “input1” changes from FALSE to TRUE and, during the time from when the timer is started to when the timer expires (for 10 seconds), to cause the output variable “output1” to remain TRUE.

The instance name is TP_0.

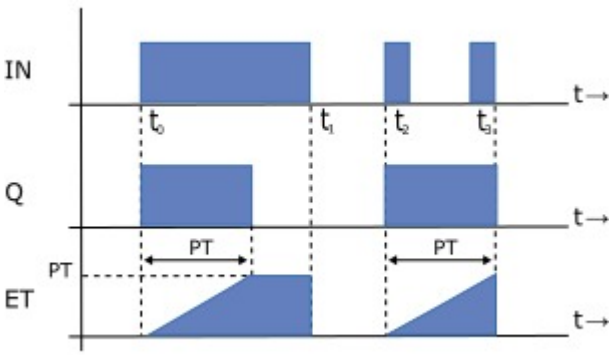
LD program



ST program

```
TP_0(IN TRUE := input1 TRUE,
     PT T#10s := T#10S,
     Q TRUE => output1 TRUE,
     ET T#2s822ms => ElapsedTime T#2s822ms);
```

■ Time-sequence diagram



4.1.4 RTC (Realtime Clock)

This is a function block that starts counting time at the rising edge starting from the specified date and time. The output remains TRUE while the time counting continues. After a specified time elapses, the output becomes FALSE.

■ Icon

■ Parameter

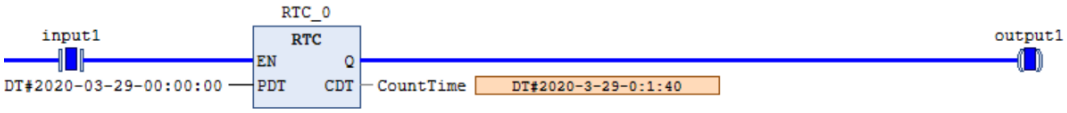
Scope	Name	Type	Description
Input	EN	BOOL	Starts counting time from the date and time specified in the input argument PDT when FALSE changes to TRUE (rising edge). When TRUE changes to FALSE, DT#1970-01-01-00:00:00 is set in the output argument CDT.
	PDT	DATE_AND_TIME	Date and time when time counting starts
Output	Q	BOOL	Outputs TRUE while time counting continues.
	CDT	DATE_AND_TIME	Outputs the time count time from the date and time specified in the input argument PDT.

■ Program example

This program is designed to start counting time, starting from 0 o'clock of March 29, 2020, when the input variable “input1” changes from FALSE to TRUE, and, to cause the output variable “output1” to remain TRUE while time counting continues.

The instance name is RTC_0.

LD program



ST program

```
RTC_0(EN TRUE := input1 TRUE,
      PDT DT#2020-3-29-0:0:0 := DT#2020-03-29-00:00:00,
      Q TRUE => output1 TRUE,
      CDT DT#2020-3-29-0:1:31 => CountTime DT#2020-3-29-0:1:31);
```

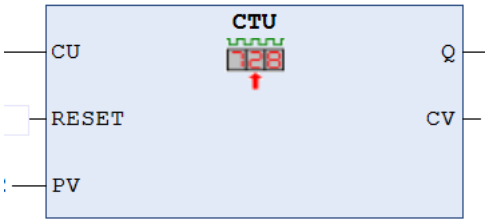
4.2 Counter Instructions

Counter instructions can be used to perform counter operations.

4.2.1 CTU (Up Counter)

This is a function block that increments the counter value by 1 every time the rising edge occurs.

■ Icon



■ Parameter

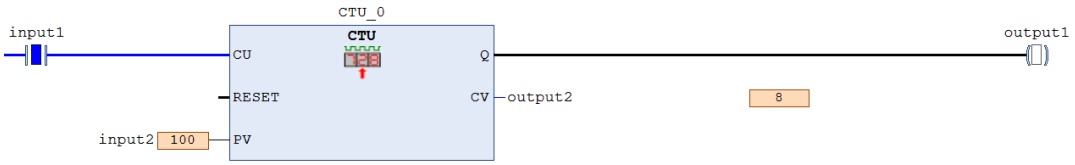
Scope	Name	Type	Description
Input	CU	BOOL	Increments the value of the output argument CV by 1 when FALSE changes to TRUE (rising edge).
	RESET	BOOL	If TRUE, 0 is set in the output argument CV.
	PV	WORD	Target value of CV
Output	Q	BOOL	Outputs TRUE when the CV value reaches the PV value.
	CV	WORD	Outputs the current counter value.

■ Program example

This program is designed to increment the value of the output variable “output2” by 1 every time the input variable “input1” changes from FALSE to TRUE. The program is designed to cause the output variable “output1” to change to TRUE when the value (100) of the input variable “input2” is counted up.

The instance name is CTU_0.

LD program



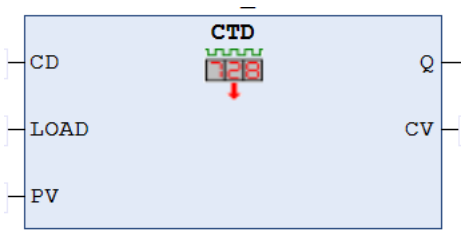
ST program

```
CTU_0(
     CU TRUE := input1 TRUE,
     RESET:= ,
     PV 100 := 100,
     Q FALSE => output1 FALSE,
     CV 8 => output2 8 );
```

4.2.2 CTD (Down Counter)

This is a function block that decrements the counter value by 1 every time the rising edge occurs.

■ Icon



■ Parameter

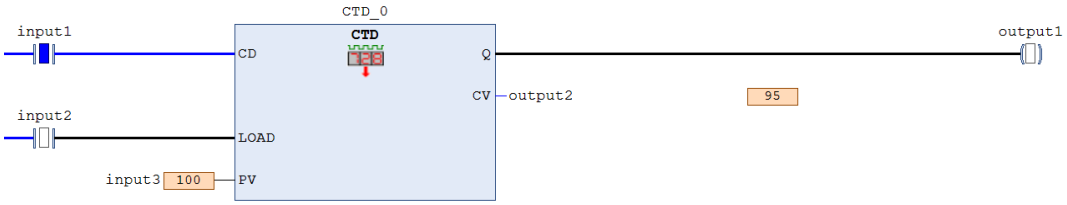
Scope	Name	Type	Description
Input	CD	BOOL	Decrements the value of the output argument CV by 1 when FALSE changes to TRUE (rising edge).
	LOAD	BOOL	If TRUE, the value specified in PV is set in the output argument CV.
	PV	WORD	Initial value of the counter value
Output	Q	BOOL	Outputs TRUE when the CV value becomes 0.
	CV	WORD	Outputs the current counter value.

■ Program example

This program is designed to decrement the value of the output variable “output2” by 1 every time the input variable “input1” changes from FALSE to TRUE, and to cause the output variable “output1” to change to TRUE when the value becomes 0. The initial value (100) to count down from is specified in the input variable “input3”.

The instance name is CTD_0.

LD program



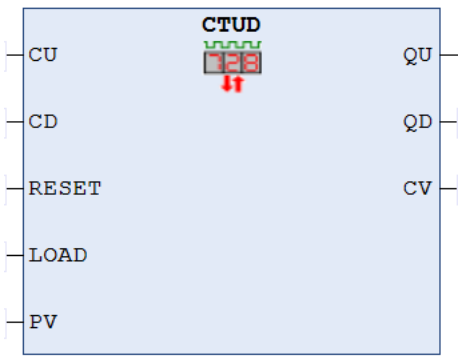
ST program

```
CTD_0(
    CD TRUE := input1 TRUE,
    LOAD FALSE := input2 FALSE,
    PV 100 := input3 100,
    Q FALSE => output1 FALSE,
    CV 95 => output2 95);
```

4.2.3 CTUD (Up-down Counter)

This is a function block that increments or decrements the counter value by 1 every time the rising edge occurs.

■ Icon



■ Parameter

Scope	Name	Type	Description
Input	CU	BOOL	Increments the value of the output argument CV by 1 when FALSE changes to TRUE (rising edge).
	CD	BOOL	Decrements the value of the output argument CV by 1 when FALSE changes to TRUE (rising edge).
	RESET	BOOL	If TRUE, 0 is set in the output argument CV.
	LOAD	BOOL	If TRUE, the value specified in PV is set in the output argument CV.
	PV	WORD	Initial value of the counter value
Output	QU	BOOL	Outputs TRUE when the CV value reaches the PV value.

Scope	Name	Type	Description
	QD	BOOL	Outputs TRUE when the CV value becomes 0.
	CV	WORD	Outputs the current counter value.

■ Program example

Every time the input variable “input1” changes from FALSE to TRUE, the value of the output variable “output3” is incremented by 1.

Every time the input variable “input2” changes from FALSE to TRUE, the value of the output variable “output3” is decremented by 1

When the output variable “output3” becomes greater than or equal to the input variable “input5”, the output variable “output1” becomes TRUE.

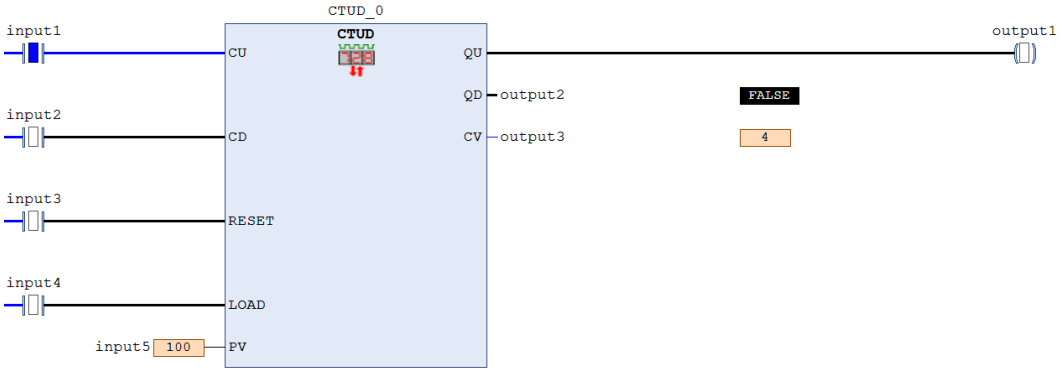
When the output variable “output3” becomes 0, the output variable “output2” becomes TRUE.

When the input variable “input3” becomes TRUE, the output variable “output3” becomes 0.

When the input variable “input4” becomes TRUE, the value (100) of the input variable “input5” is set in the output variable “output3”.

The instance name is CTUD_0.

LD program



ST program

```
CTUD_0(
    CU TRUE := input1 TRUE,
    CD FALSE := input2 FALSE,
    RESET FALSE := input3 FALSE,
    LOAD FALSE := input4 FALSE,
    PV 100 := input5 100,
    QU FALSE => output1 FALSE,
    QD FALSE => output2 FALSE,
    CV 3 => output3 3 );
```

4.3 Edge Detection Instructions

Edge detection instructions can be used to perform edge detection.

4.3.1 R_TRIG (Rising Edge Detection)

This is a function block that detects a rising edge.

■ **Icon**

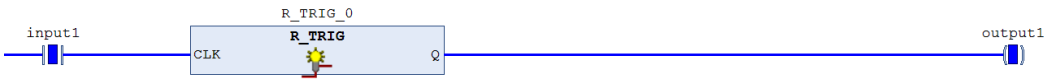
■ **Parameter**

Scope	Name	Type	Description
Input	CLK	BOOL	Input that detects a rising edge
Output	Q	BOOL	Outputs TRUE for one cycle only when a rising edge is detected in the input argument CLK.

■ **Program example**

When the input variable “input1” changes from FALSE to TRUE, the output variable “output1” becomes TRUE for one cycle only.
The instance name is R_TRIG_0.

LD program



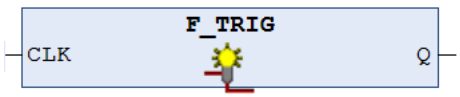
ST program

```
R_TRIG_0(
    CLK TRUE := input1 TRUE,
    Q TRUE => output1 TRUE);
```

4.3.2 F_TRIG (Falling Edge Detection)

This is a function block that detects a falling edge.

■ **Icon**



■ Parameter

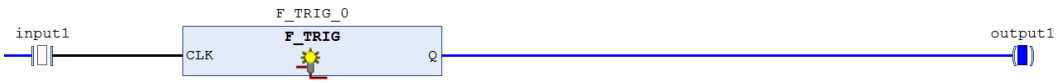
Scope	Name	Type	Description
Input	CLK	BOOL	Input that detects a falling edge
Output	Q	BOOL	Outputs TRUE for one cycle only when a falling edge is detected in the input argument CLK.

■ Program example

When the input variable “input1” changes from FALSE to TRUE, the output variable “output1” becomes TRUE for one cycle only.

The instance name is F_TRIG_0.

LD program



ST program

```
F_TRIG_0(
    CLK FALSE := input1 FALSE,
    Q TRUE => output1 TRUE );
```

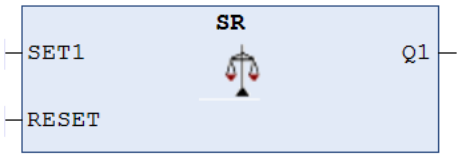
4.4 Bistable Circuit Instructions

Bistable circuit instructions can be used to perform edge detection.

4.4.1 SR (Set-priority Bistable Circuit)

This is a function block that realizes a bistable (flip-flop) circuit. The priority is given to the set input.

■ Icon



■ Parameter

Scope	Name	Type	Description
Input	SET1	BOOL	Specifies the set input for a bistable circuit.
	RESET	BOOL	Specifies the reset input for a bistable circuit.
Output	Q1	BOOL	When the input argument SET1 becomes TRUE, outputs and holds TRUE. When the input argument RESET becomes TRUE, outputs and holds FALSE. When both SET1 and RESET1 are TRUE, outputs and holds TRUE.

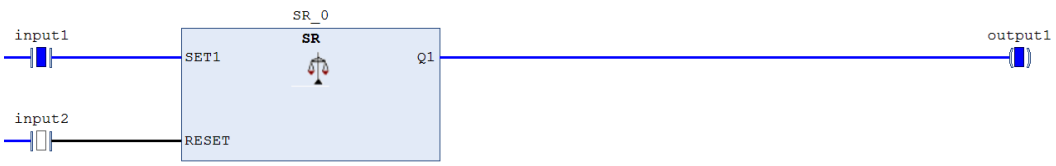
■ Program example

When the input variable “input1” becomes TRUE, the output variable “output1” becomes TRUE.
Even if the input variable “input1” becomes FALSE, “output1” remains TRUE.

When the input variable “input1” is FALSE and if input variable “input2” becomes TRUE, the output variable “output1” becomes FALSE.

The instance name is SR_0.

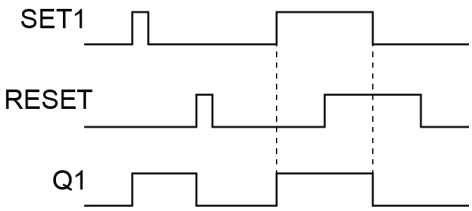
LD program



ST program

```
SR_0(
    SET1 TRUE := input1 TRUE,
    RESET FALSE := input2 FALSE,
    Q1 TRUE => output1 TRUE);
```

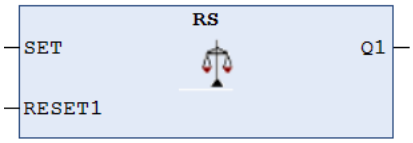
■ **Time-sequence diagram**



4.4.2 RS (Reset-priority Bistable Circuit)

This is a function block that realizes a bistable (flip-flop) circuit. The priority is given to the reset input.

■ **Icon**



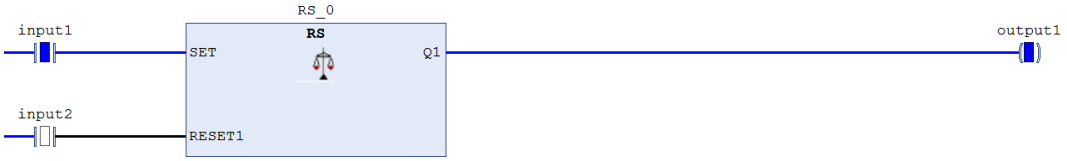
■ **Parameter**

Scope	Name	Type	Description
Input	SET1	BOOL	Specifies the set input for a bistable circuit.
	RESET	BOOL	Specifies the reset input for a bistable circuit.
Output	Q1	BOOL	When the input argument SET1 becomes TRUE, outputs and holds TRUE. When the input argument RESET becomes TRUE, outputs and holds FALSE. When both SET1 and RESET1 are TRUE, outputs and holds FALSE.

■ **Program example**

When the input variable “input1” becomes TRUE, the output variable “output1” becomes TRUE.
Even if the input variable “input1” becomes FALSE, “output1” remains TRUE.
When the input variable “input1” is FALSE and if the input variable “input2” becomes TRUE, the output variable “output1” becomes FALSE.
The instance name is RS_0.

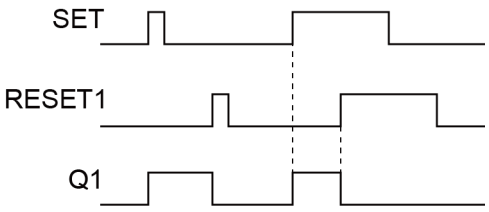
LD program



ST program

```
RS_0(
    SET TRUE := input1 TRUE,
    RESET1 FALSE := input2 FALSE,
    Q1 TRUE => output1 TRUE);
```

■ **Time-sequence diagram**

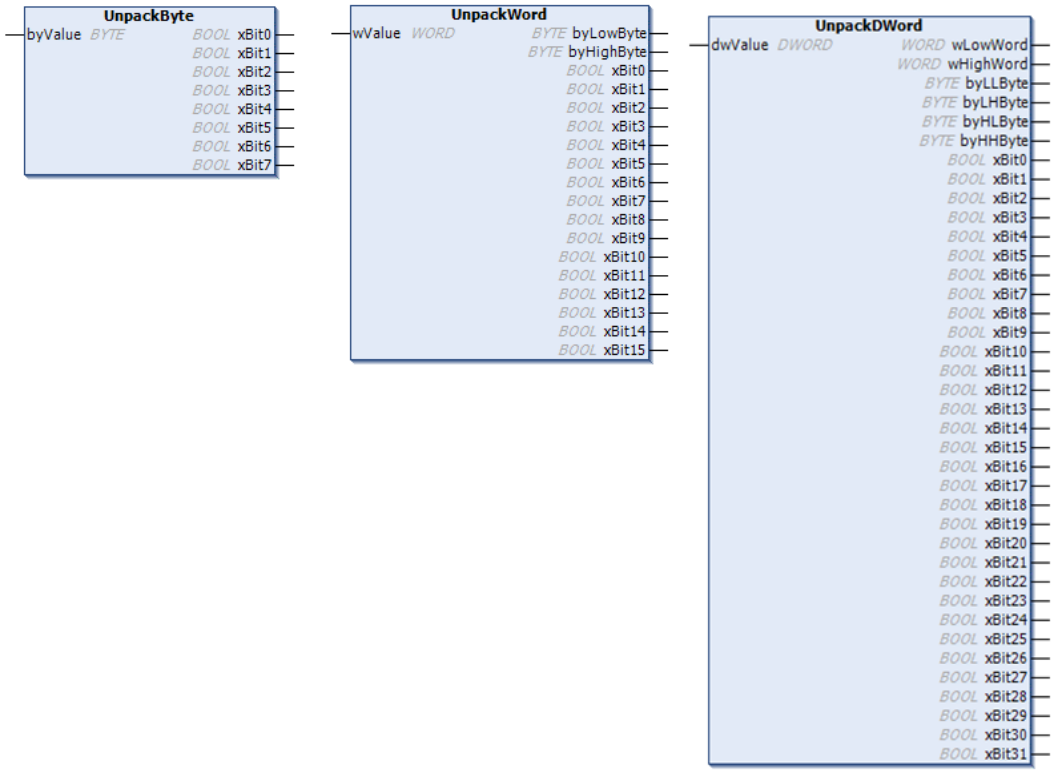


4.5 Data Type Conversion Instructions

4.5.1 MEM.Unpack** (BYTE/WORD/DWORD to Bit Data Conversion)

This is a function that unpacks input BYTE-, WORD-, or DWORD-type data to data in bits and outputs the data.

■ **Icon**



■ **Parameter**

UnpackByte

Scope	Name	Type	Description
Input	byValue	BYTE	BYTE type data to be unpacked
Output	xBit0～xBit7	BOOL	A value representing the input value unpacked in bits

UnpackWord

Scope	Name	Type	Description
Input	wValue	BYTE	WORD type data to be unpacked
Output	byHighByte	BYTE	High byte unpacked from the input value

Scope	Name	Type	Description
Output	byLowByte	BYTE	Low byte unpacked from the input value
Output	xBit0～xBit15	BYTE	A value representing the input value unpacked in bits

UnpackDWord

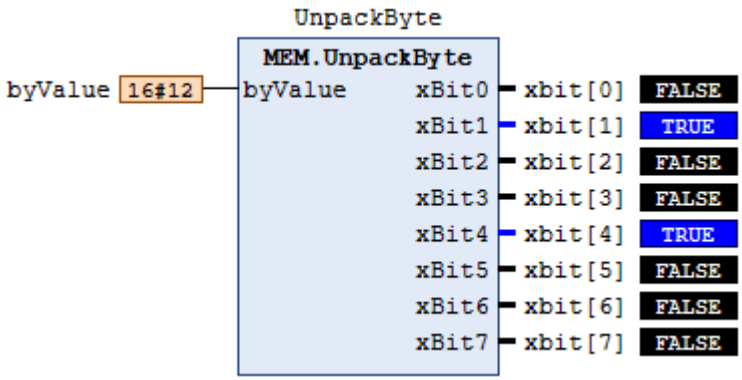
Scope	Name	Type	Description
Input	dwValue	DWORD	DWORD type data to be unpacked
Output	wHighWord	WORD	High WORD unpacked from the input value
Output	wLowWord	WORD	Low WORD unpacked from the input value
Output	byHHByte	BYTE	HH byte unpacked from the input value
Output	byHLByte	BYTE	HL byte unpacked from the input value
Output	byLHByte	BYTE	LH byte unpacked from the input value
Output	byLLByte	BYTE	LL byte unpacked from the input value
Output	xBit0～xBit31	BOOL	A value representing the input value unpacked in bits

■ Program example 1

This program is designed to unpack the byValue input variable of the BYTE type to pieces of data of the BOOL type and outputs them to the xBit0 to xBit7 output variables of the BOOL type.

byValue := 16#12

LD program



ST program

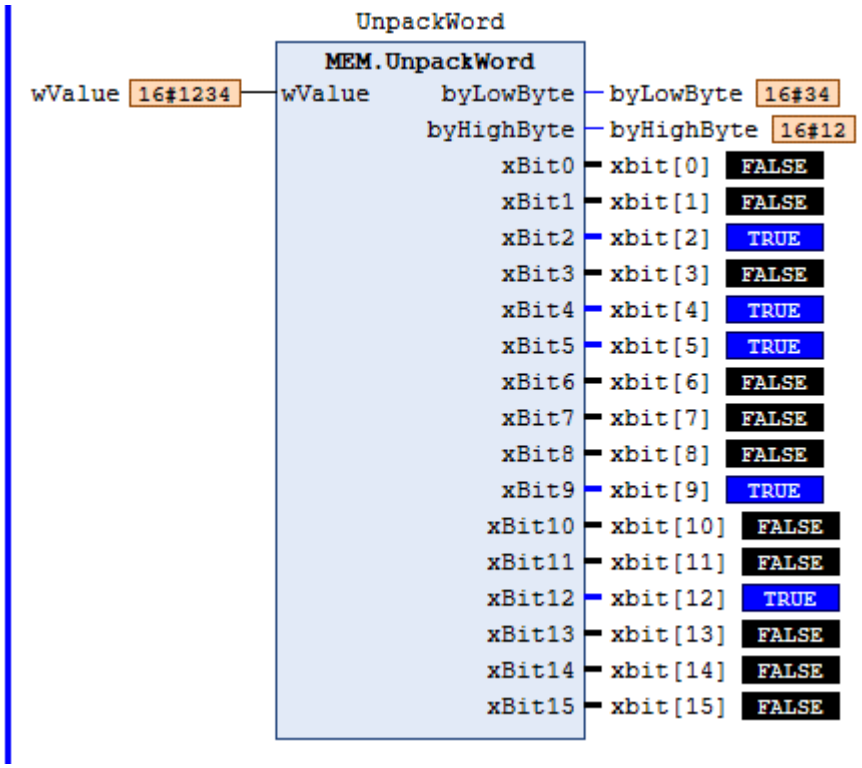
```
UnpackByte(byvalue 16#12 :=byValue 16#12 ,
           xBit7 FALSE =>xbit[7] FALSE ,
           xBit6 FALSE =>xbit[6] FALSE ,
           xBit5 FALSE =>xbit[5] FALSE ,
           xBit4 TRUE =>xbit[4] TRUE ,
           xBit3 FALSE =>xbit[3] FALSE ,
           xBit2 FALSE =>xbit[2] FALSE ,
           xBit1 TRUE =>xbit[1] TRUE ,
           xBit0 FALSE =>xbit[0] FALSE );
```

■ Program example 2

This program is designed to unpack the wValue input variable of the WORD type to pieces of data of the BOOL and BYTE types and outputs them to the xBit0 to xBit15 output variables of the BOOL type and the byHighByte and byLowByte output variables of the BYTE type.

wValue := 16#1234

LD program



ST program

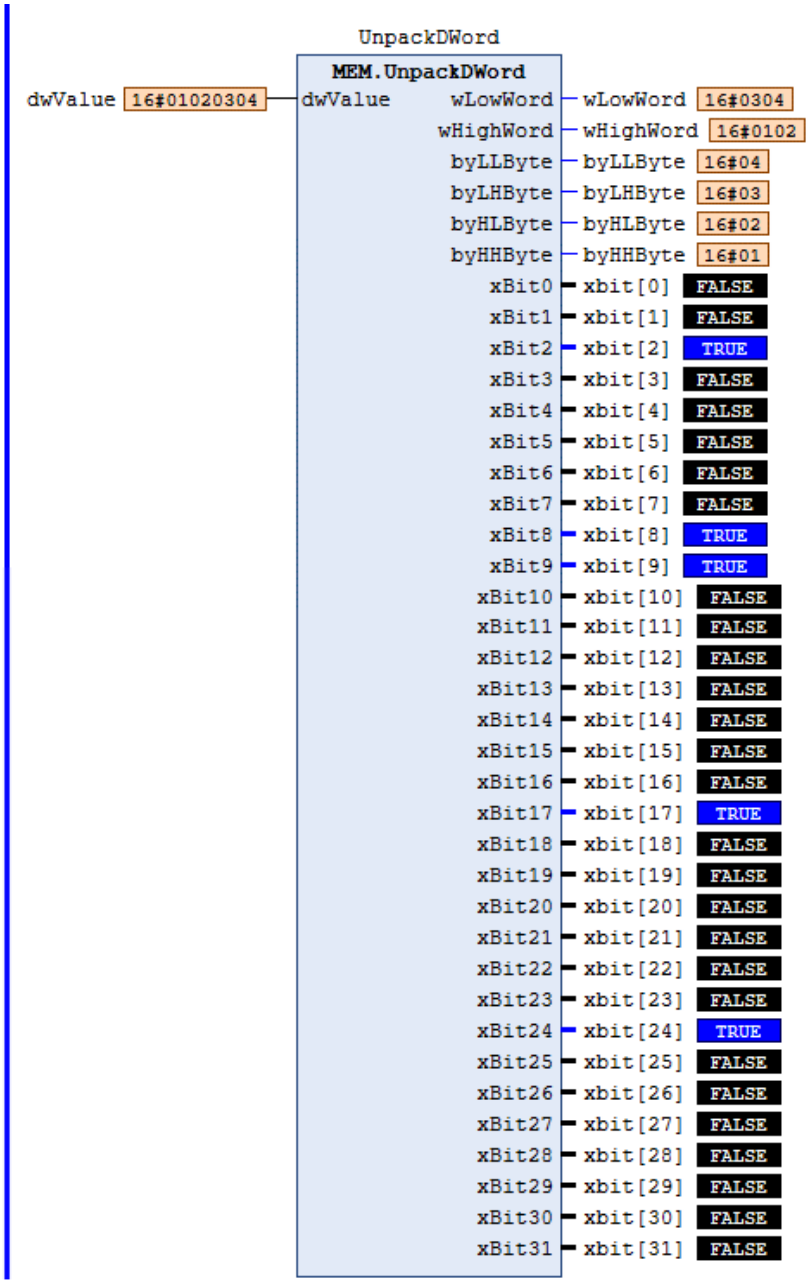
```
UnpackWord(wValue 16#1234 :=wValue 16#1234 ,
           byHighByte 16#12 =>byHighByte 16#12 ,byLowByte 16#34 =>byLowByte 16#34 ,
           xBit15 FALSE =>xbit[15] FALSE ,
           xBit14 FALSE =>xbit[14] FALSE ,
           xBit13 FALSE =>xbit[13] FALSE ,
           xBit12 TRUE =>xbit[12] TRUE ,
           xBit11 FALSE =>xbit[11] FALSE ,
           xBit10 FALSE =>xbit[10] FALSE ,
           xBit9 TRUE =>xbit[9] TRUE ,
           xBit8 FALSE =>xbit[8] FALSE ,
           xBit7 FALSE =>xbit[7] FALSE ,
           xBit6 FALSE =>xbit[6] FALSE ,
           xBit5 TRUE =>xbit[5] TRUE ,
           xBit4 TRUE =>xbit[4] TRUE ,
           xBit3 FALSE =>xbit[3] FALSE ,
           xBit2 TRUE =>xbit[2] TRUE ,
           xBit1 FALSE =>xbit[1] FALSE ,
           xBit0 FALSE =>xbit[0] FALSE );
```

■ Program example 3

This program is designed to unpack the dwValue input variable of the DWORD type to pieces of data of the BOOL, BYTE and WORD types and outputs them to the xBit0 to xBit31 output variables of the BOOL type, the wHighWord and wLowWord output variables of the WORD type, and the byHHByte, byHLByte, byLHByte, and byLLByte output variables of the BYTE type.

dwValue := 16#01020304

LD program


UnpackDWord
MEM.UnpackDWord
dwValue 16#01020304
dwValue
wLowWord wLowWord 16#0304
wHighWord wHighWord 16#0102
byLLByte byLLByte 16#04
byLHByte byLHByte 16#03
byHLByte byHLByte 16#02
byHHByte byHHByte 16#01
xBit0 xbit[0] FALSE
xBit1 xbit[1] FALSE
xBit2 xbit[2] TRUE
xBit3 xbit[3] FALSE
xBit4 xbit[4] FALSE
xBit5 xbit[5] FALSE
xBit6 xbit[6] FALSE
xBit7 xbit[7] FALSE
xBit8 xbit[8] TRUE
xBit9 xbit[9] TRUE
xBit10 xbit[10] FALSE
xBit11 xbit[11] FALSE
xBit12 xbit[12] FALSE
xBit13 xbit[13] FALSE
xBit14 xbit[14] FALSE
xBit15 xbit[15] FALSE
xBit16 xbit[16] FALSE
xBit17 xbit[17] TRUE
xBit18 xbit[18] FALSE
xBit19 xbit[19] FALSE
xBit20 xbit[20] FALSE
xBit21 xbit[21] FALSE
xBit22 xbit[22] FALSE
xBit23 xbit[23] FALSE
xBit24 xbit[24] TRUE
xBit25 xbit[25] FALSE
xBit26 xbit[26] FALSE
xBit27 xbit[27] FALSE
xBit28 xbit[28] FALSE
xBit29 xbit[29] FALSE
xBit30 xbit[30] FALSE
xBit31 xbit[31] FALSE

ST program

```
UnpackDWord(dwValue 16#01020304 :=dwValue 16#01020304 ,
            wHighWord 16#0102 =>wHighWord 16#0102 ,wLowWord 16#0304 =>wLowWord 16#0304 ,
            byHHByte 16#01 =>byHHByte 16#01 ,byHLByte 16#02 =>byHLByte 16#02 ,
            byLHByte 16#03 =>byLHByte 16#03 ,byLLByte 16#04 =>byLLByte 16#04 ,
            xBit31 FALSE =>xbit[31] FALSE ,
            xBit30 FALSE =>xbit[30] FALSE ,
            xBit29 FALSE =>xbit[29] FALSE ,
            xBit28 FALSE =>xbit[28] FALSE ,
            xBit27 FALSE =>xbit[27] FALSE ,
            xBit26 FALSE =>xbit[26] FALSE ,
            xBit25 FALSE =>xbit[25] FALSE ,
            xBit24 TRUE =>xbit[24] TRUE ,
            xBit23 FALSE =>xbit[23] FALSE ,
            xBit22 FALSE =>xbit[22] FALSE ,
            xBit21 FALSE =>xbit[21] FALSE ,
            xBit20 FALSE =>xbit[20] FALSE ,
            xBit19 FALSE =>xbit[19] FALSE ,
            xBit18 FALSE =>xbit[18] FALSE ,
            xBit17 TRUE =>xbit[17] TRUE ,
            xBit16 FALSE =>xbit[16] FALSE ,
            xBit15 FALSE =>xbit[15] FALSE ,
            xBit14 FALSE =>xbit[14] FALSE ,
            xBit13 FALSE =>xbit[13] FALSE ,
            xBit12 FALSE =>xbit[12] FALSE ,
            xBit11 FALSE =>xbit[11] FALSE ,
            xBit10 FALSE =>xbit[10] FALSE ,
            xBit9 TRUE =>xbit[9] TRUE ,
            xBit8 TRUE =>xbit[8] TRUE ,
            xBit7 FALSE =>xbit[7] FALSE ,
            xBit6 FALSE =>xbit[6] FALSE ,
            xBit5 FALSE =>xbit[5] FALSE ,
            xBit4 FALSE =>xbit[4] FALSE ,
            xBit3 FALSE =>xbit[3] FALSE ,
            xBit2 TRUE =>xbit[2] TRUE ,
            xBit1 FALSE =>xbit[1] FALSE ,
            xBit0 FALSE =>xbit[0] FALSE );
```

4.6 Data manipulation instructions

You can process the data using data manipulation instructions.

4.6.1 LIN_TRAFO (linear conversion)

Convert one range of numbers to another linearly.

It is calculated by the following formula.

OUT = OUT_MIN ＋((IN－IN_MIN)×(OUT_MAX-OUT_MIN)/(IN_MAX-IN_MIN))

Set the setting value to IN_MIN ≤ IN ≤ IN_MAX, and set the maximum input value (IN_MAX) to be not equal to the minimum input value (IN_MIN).

■ **Icon**

■ **Parameter**

Scope	Name	Type	Default value	Description
Input	IN	REAL	0	Input value
	IN_MIN	REAL	0	Minimum input range
	IN_MAX	REAL	0	Maximum input range
	OUT_MIN	REAL	0	Minimum output range
	OUT_MAX	REAL	0	Maximum output range
Output	OUT	REAL	0	Converted output value
	ERROR	BOOL	FALSE	TRUE：IN_MIN=IN_MAX Or IN is outside of the input range (IN<IN_MIN or IN >_IN_MAX)

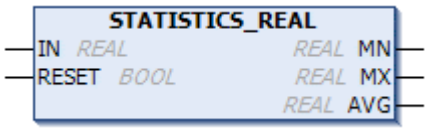
Note

- Do not set the input range (IN_MAX-IN_MIN) ≥ REAL maximum value (3.402823E+38)
- Do not set the output range (OUT_MAX-OUT_MIN) ≥ REAL maximum value (3.402823E+38).
- Do not set the same range (IN_MIN = OUT_MIN and IN_MAX = OUT_MAX).
- The REAL type is divided into the mantissa and an exponent, so if you increase the input range and the output range, an error will occur.

4.6.2 STATISTICS_REAL (maximum, minimum, and average input values)

Acquire the maximum, minimum, and average values of the input data (REAL type). The input value is added and updated for each execution timing. Resetting will return the maximum, minimum, and average values to their default values.

■ Icon



■ Parameter

Scope	Name	Type	Default value	Description
Input	IN	REAL	0	Input value
	RESET	BOOL	FALSE	TRUE: Reset Set MN, MX, AVG to the default values
Output	MN	REAL	-3.402823466 E+38	Minimum value
	MX	REAL	3.402823466 E+38	Maximum value
	AVG	REAL	0	Average Value

4.6.3 LIMITALARM (Monitoring of input values)

Monitor whether the input value is between LOW (lower limit) and HIGH (upper limit)

■ Icon

■ Parameter

Scope	Name	Type	Default value	Description
Input	IN	INT	0	Input value
	HIGH	INT	0	Upper limit value(Note 1)
	LOW	INT	0	Lower limit value(Note 1)
Output	O	BOOL	FALSE	TRUE: Input value (IN) is greater than HIGH, FALSE: IN is equal to or less than HIGH

Scope	Name	Type	Default value	Description
	U	BOOL	FALSE	TRUE: Input value (IN) is less than LOW, FALSE: IN is equal to or greater than LOW
	IL	BOOL	FALSE	TRUE: Input value (IN) is within the range of LOW to HIGH FALSE: If either the output argument O or U is TRUE

(Note 1) Set LOW < HIGH to use.

4.7 Other instructions

You can use other instructions

4.7.1 BLINK (output of blinking signal)

Switch the output argument OUT to TRUE or FALSE according to the setting time.

■ **Icon**

■ **Parameter**

Scope	Name	Type	Default value	Description
Input	ENABLE	BOOL	FALSE	TRUE: Start the pulse output. FALSE: The pulse output is stopped and the output OUT is maintained.(Note 1)
	TIMELOW	TIME	T#0ms	Time that is FALSE(Note 2)
	TIMEHIGH	TIME	T#0ms	Time that is TRUE(Note 2)
Output	OUT	BOOL	FALSE	Switch between TRUE and FALSE at the specified time. If the output at the start of the pulse is FALSE, it starts with TRUE, and if it is TRUE, it starts with FALSE.

(Note 1) When pulse is stopped (ENABLE = FALSE), the value of the output value OUT at that time is maintained.

(Note 2) When executing with the default value (T#0ms), the timing at which the pulse signal OUT switches is 1 scan.

(MEMO)

5 Motion Control Function Blocks (Single Axis Control)

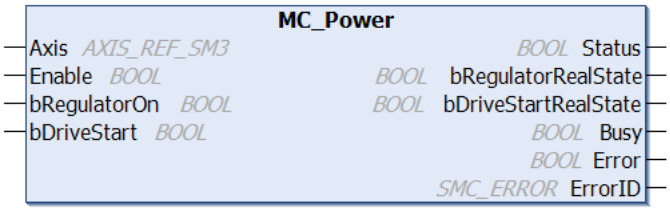
This section describes motion control function blocks for the single axis.

5.1 Servo ON

5.1.1 MC_Power (Servo ON)

This is a function block (FB) that sets the axis to the servo ON state to be ready for operation.

■ Icon

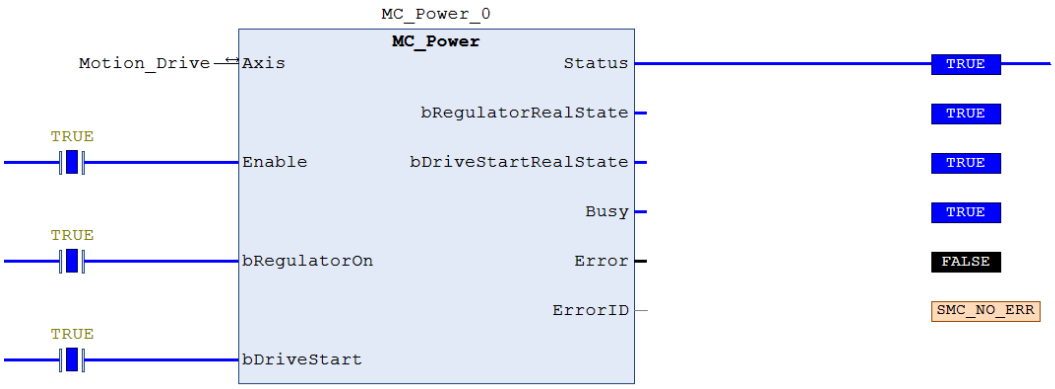


■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	Enable	BOOL	FALSE	TRUE: The FB can be executed.
	bRegulatorOn	BOOL	FALSE	TRUE: Servo ON FALSE: Servo OFF
	bDriveStart	BOOL	FALSE	TRUE: Quick stop is disabled. For the GM1 Controller, fix to TRUE.
Output	Status	BOOL	FALSE	TRUE: The axis can be executed.
	bRegulatorRealState	BOOL	FALSE	TRUE: The FB is ready to be executed.
	bDriveStartRealState	BOOL	FALSE	TRUE: Operation is not stopped due to quick stop.
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	FALSE	An error ID is output.

■ Program example

The following program is designed to set the axis that corresponds to the input variable Motion_Drive to the servo ON state.



Info.

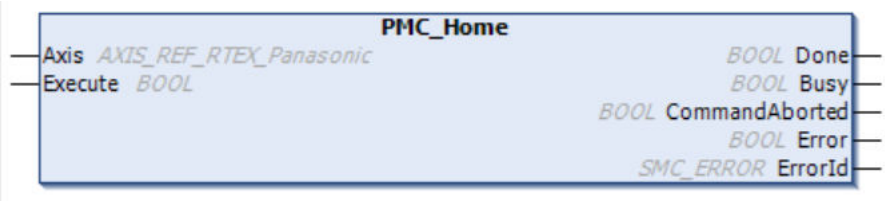
- If the input arguments Enable, bRegulatorOn, and bDriveStart are TRUE and the output argument Status is FALSE, a hardware problem may occur.
- Reference manual
  GM1 Controller RTEX User's Manual (Operation Edition)

5.2 Home Return

5.2.1 PMC_Home (Home Return)

This is a function block (FB) that performs home return of the axis. The home return function of the servo amplifier is used.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_RTEX_Panasonic	-	Specifies the axis.
Input	Execute	BOOL	FALSE	TRUE: Starts execution at the rising edge. FALSE: Stops processing.
Output	Done	BOOL	FALSE	TRUE: Execution is completed and transitioned to the Standstill state.
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	CommandAborted	BOOL	FALSE	TRUE: An interruption from other FB has occurred.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	An error ID is output.

■ Execution operation

- Execute = TRUE: Starts the home return mode. Execute = FALSE: Ends the home return mode.
- When PMC_Home is successfully completed (when Done changes to TRUE), the home return mode is automatically ended.
- When PMC_Home is abnormally terminated (when Error changes to TRUE), end the home return mode by setting Execute = FALSE and making a call.

■ Execution errors

The PMC_Home function block outputs the following errors.

Error	Description
SMC_WRONG_CONTROLLER_MODE	Executed in a mode other than the position control mode. Change to SMC_position using SMC_SetControllerMode.
SMC_DI_HOMING_ERROR	The version of the amplifier paired with an absolute encoder is lower than V1.24.
	Trigger setting is incorrect.
	Amplifier parameters (Pr4.00 to Pr4.07) are incorrect.
	Abnormal state in HOME, POT, or NOT is detected.
	The home return cannot be completed even if POT and NOT settings were inverted three times or more.
	The home return was completed at an incorrect position.
SMC_MS_DIRECTION_NOT_APPLICABLE	The return direction setting is incorrect.
SMC_AXIS_NOT_READY_FOR_MOTION	The axis is in a state (Stopping, Disabled, or Errorstop) where PMC_Home cannot be executed.
SMC_REGULATOR_OR_START_NOT_SET	The servo was turns OFF and the brake was applied.
SMC_3SH_INVALID_VELACC_VALUES	The input target velocity, home return creep speed, acceleration, or deceleration is incorrect.
SMC_AXIS_REF_CHANGED_DURING_OPERATION	The Axis was changed during operation.

■ Execution conditions

- As the PMC_Home function block uses the RTEX home return command, it cannot be executed together with PMC_ReadLatchPosition or PMC_StopLatchPosition.
- If PMC_Home is executed while PMC_ReadLatchPosition or PMC_StopLatchPosition is being executed, the CommandAborted parameter becomes TRUE. Furthermore, if PMC_Home of another instance is executed while one PMC_Home is being executed, the CommandAborted parameter of the PMC_Home executed later becomes TRUE.

■ Amplifier parameter conditions

When using PMC_Home, set amplifier parameters as shown in the following table.

Parameter	Parameter name	Setting A	Setting B
Pr4.00	SI1 input selection	SI-MON5	SI-MON5
Pr4.01	SI2 input selection	POT	
Pr4.02	SI3 input selection	NOT	
Pr4.03	SI4 input selection	SI-MON1	SI-MON1
Pr4.04	SI5 input selection	HOME	HOME
Pr4.05	SI6 input selection	EXT2	POT
Pr4.06	SI7 input selection	EXT3	NOT
Pr4.07	SI8 input selection	SI-MON4	SI-MON4

Return methods that can be executed for the settings A and B are as shown in the following table.

Return method	Setting A	Setting B
DOG method 1	○	○(Note 2)
DOG method 2	×(Note 1)	○
DOG method 3	○	○(Note 2)
Limit method 1	○	○(Note 2)
Limit method 2	×(Note 1)	○
Home return method	○	○(Note 2)
Stop-on-contact method 1	○	○(Note 2)
Stop-on-contact method 2	○	○
Data setting method	○	○
High-speed home return method	○	○

(Note 1) When using POT, NOT, or HOME as a home reference trigger, assign them as follows.
HOME: SI5 input selection
POT: SI6 input selection
NOT: SI7 input selection

(Note 2) When EXT2 or EXT3 is used as a home reference trigger, it can be used only for the above setting A.

Info.

- Reference manual
  GM1 Controller RTEX User's Manual (Operation Edition)

5.2.2 MC_Home (Home Return)

This is a function block (FB) that performs home return.

■ Icon

■ Parameter

Scope	Name	Type	Default	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	Execute	BOOL	FALSE	TRUE: Starts execution at the rising edge.

Scope	Name	Type	Default	Description
	Position	LREAL	0	Set value of the absolute position when the reference signal is detected
Output	Done	BOOL	FALSE	TRUE: Stopping is completed.
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	CommandAborted	BOOL	FALSE	TRUE: An interruption from other FB has occurred.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	Error ID output

5.3 Control Switch

5.3.1 SMC_SetControllerMode (Control Mode Setting)

This is a function block (FB) that sets up the control mode for controlling the position, velocity, and torque.

■ **Icon**

■ **Parameter**

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	bExecute	BOOL	FALSE	Sets up the control mode at the rising edge.
	nControllerMode	SMC_CONTROLLER_MODE	SMC_position	Specifies the control mode.
Output	bDone	BOOL	FALSE	TRUE: Control mode setup is completed.
	bBusy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	bError	BOOL	FALSE	TRUE: An error has occurred within the FB.
	nErrorID	SMC_ERROR	0	An error ID is output.

■ **SMC_CONTROLLER_MODE (Enumeration type)**

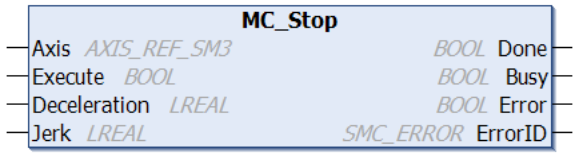
Name	Value	Description
SMC_nocontrol	0	Usage prohibited
SMC_torque	1	Torque mode
SMC_velocity	2	Velocity mode
SMC_position	3	Position mode
SMC_current	4	Usage prohibited

5.4 Stop

5.4.1 MC_Stop (Forced Stop)

This is a function block (FB) that causes the axis to make a deceleration stop. After stopping, the axis remains stopped while Execute is TRUE. While the axis is being decelerated or while it is stopped, other function blocks cannot be executed.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	Execute	BOOL	FALSE	Starts execution at the rising edge. While it is TRUE, other FB cannot be executed.
	Deceleration	LREAL	0	Specifies the deceleration (u/s^2).
	Jerk	LREAL	0	Specifies the jerk (u/s^3).
Output	Done	BOOL	FALSE	TRUE: The axis velocity has reached 0.
	Busy	BOOL	FALSE	TRUE: The FB is in operation.
	Error	BOOL	FALSE	TRUE: An error has occurred.
	ErrorID	SMC_ERROR	0	An error ID is output.

Info.

- In the torque control mode (SMC_torque), the axis cannot be stopped using MC_Stop. For stopping methods, refer to PMC_SetTorque.
- Reference manual

  GM1 Controller RTEX User's Manual (Operation Edition)

  GM1 Controller EtherCAT User's Manual (Operation Edition)

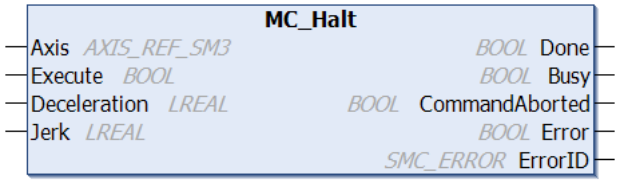
REFERENCE

5.8.1 PMC_SetTorque (Torque Control)

5.4.2 MC_Halt (Halt)

This is a function block (FB) that causes the axis to make a deceleration stop. After the axis is stopped or while the axis is being decelerated, other motion instructions can be executed.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	Execute	BOOL	FALSE	Starts execution at the rising edge.
	Deceleration	LREAL	0	Specifies the deceleration (u/s^2).
	Jerk	LREAL	0	Specifies the jerk (u/s^3).
Output	Done	BOOL	FALSE	TRUE: The axis velocity has reached 0.
	Busy	BOOL	FALSE	TRUE: The FB is in operation.
	CommandAborted	BOOL	FALSE	TRUE: An interruption from other FB has occurred.
	Error	BOOL	FALSE	TRUE: An error has occurred.
	ErrorID	SMC_ERROR	0	Error ID output

Info.

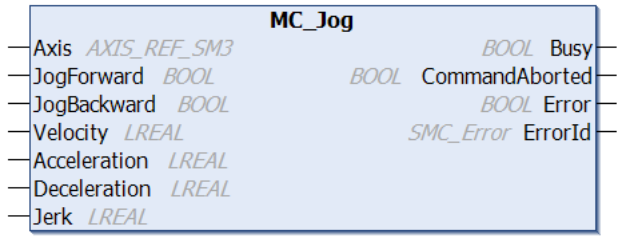
- Reference manual
  GM1 Controller RTEX User's Manual (Operation Edition)
  GM1 Controller EtherCAT User's Manual (Operation Edition)

5.5 JOG / Inching

5.5.1 MC_Jog (Jogging)

This is a function block (FB) that causes the axis to keep traveling in a forward or reverse direction at a constant velocity. While the input is TRUE, the axis keeps traveling in a forward or reverse direction at a constant velocity.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	JogForward	BOOL	FALSE	While JogForward is TRUE, the axis travels in a forward direction. If JogBackaward is TRUE at the same time, the axis does not operate.
	JogBackward	BOOL	FALSE	While JogBackward is TRUE, the axis travels in a reverse direction. If JogForward is TRUE at the same time, the axis does not operate.
	Velocity	LREAL	0	Specifies the velocity (u/s).
	Acceleration	LREAL	0	Specifies the acceleration (u/s^2).
	Deceleration	LREAL	0	Specifies the deceleration (u/s^2).
	Jerk	LREAL	0	Specifies the jerk (u/s^3).
Output	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	CommandAborted	BOOL	FALSE	TRUE: An interruption from other FB has occurred.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	An error ID is output.

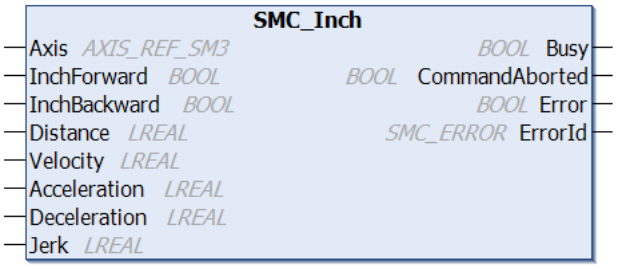
Info.

- Reference manual
  GM1 Controller RTEX User's Manual (Operation Edition)
  GM1 Controller EtherCAT User's Manual (Operation Edition)

5.5.2 SMC_Inch (Inching)

This is a function block (FB) that causes the axis to travel in a forward or reverse direction for a specified relative distance. When the input turns TRUE, the axis travels in a forward or reverse direction for a specified relative distance.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	InchForward	BOOL	FALSE	When the input changes from FALSE to TRUE, the axis starts traveling in a forward direction for the distance specified in Distance. When the input changes to FALSE before the axis travels the specified distance, the axis stops traveling. If InchBackaward is TRUE at the same time, the axis does not operate.
	InchBackward	BOOL	FALSE	When the input changes from FALSE to TRUE, the axis starts traveling in a reverse direction for the distance specified in Distance. When the input changes to FALSE before the axis travels the specified distance, the axis stops traveling. If InchForward is TRUE at the same time, the axis does not operate.
	Distance	LREAL	0	Specifies the travel distance (u).
	Velocity	LREAL	0	Specifies the velocity (u/s).
	Acceleration	LREAL	0	Specifies the acceleration (u/s^2).

Scope	Name	Type	Initial	Description
	Deceleration	LREAL	0	Specifies the deceleration (u/s^2).
	Jerk	LREAL	0	Specifies the jerk (u/s^3).
Output	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	CommandAborted	BOOL	FALSE	TRUE: An interruption from other FB has occurred.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	An error ID is output.

Info.

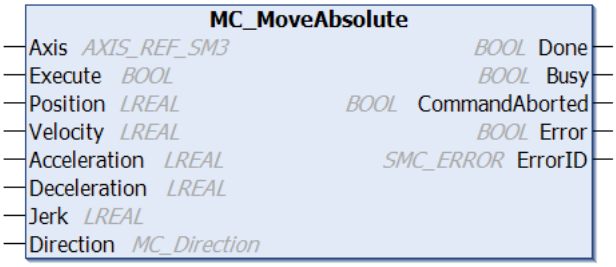
- Reference manual
  GM1 Controller RTEX User's Manual (Operation Edition)
  GM1 Controller EtherCAT User's Manual (Operation Edition)

5.6 Position Control

5.6.1 MC_MoveAbsolute (Absolute Value Positioning)

This is a function block (FB) that causes the axis to travel to a position specified as an absolute position.

■ **Icon**



■ **Parameter**

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	Execute	BOOL	FALSE	Starts execution at the rising edge.
	Position	LREAL	0	Specifies the target position (u).
	Velocity	LREAL	0	Specifies the maximum velocity (u/s).
	Acceleration	LREAL	0	Specifies the acceleration (u/s^2).
	Deceleration	LREAL	0	Specifies the deceleration (u/s^2).
	Jerk	LREAL	0	Specifies the jerk (u/s^3).
	Direction	MC_Direction	shortest	Specifies the traveling direction of the axis. Direction can be specified only for the modulo type. For the finite axis, the specification is ignored.
Output	Done	BOOL	FALSE	TRUE: The target position has been reached.
	Busy	BOOL	FALSE	TRUE: The FB is in operation.
	CommandAborted	BOOL	FALSE	TRUE: An interruption from other FB has occurred.
	Error	BOOL	FALSE	TRUE: An error has occurred.
	ErrorID	SMC_ERROR	0	An error ID is output.

■ MC_Direction (Enumeration type)

Name	Value	Description
positive	1	Travels in the positive direction.
negative	-1	Travels in the negative direction.
shortest	0	Travels in the shortest direction from the current command position to the target command position when MC_MoveAbsolute is executed.
fastest	3	Travels in the fastest direction from the current command position to the target command position when MC_MoveAbsolute is executed. If the axis is being driven by another function block when MC_MoveAbsolute is executed, selects the fastest direction within the GM1. If the axis is being stopped when MC_MoveAbsolute is executed, makes the same movement as for the shortest.
current	2	Travels to the current direction. If the axis is being driven by another function block when MC_MoveAbsolute is executed, travels in the same direction. If the axis is being stopped when MC_MoveAbsolute is executed, travels in the direction moved by the previously executed function block.

Info.

- Reference manual
  GM1 Controller RTEX User's Manual (Operation Edition)
  GM1 Controller EtherCAT User's Manual (Operation Edition)

5.6.2 MC_MoveRelative (Relative Value Positioning)

This is a function block (FB) that causes the axis to travel to a position specified as a relative position.

■ Icon

■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	Execute	BOOL	FALSE	Starts execution at the rising edge.
	Distance	LREAL	0	Specifies the relative distance (u).
	Velocity	LREAL	0	Specifies the maximum velocity (u/s).
	Acceleration	LREAL	0	Specifies the acceleration (u/s^2).
	Deceleration	LREAL	0	Specifies the deceleration (u/s^2).
	Jerk	LREAL	0	Specifies the jerk (u/s^3).
Output	Done	BOOL	FALSE	TRUE: The axis has traveled the specified relative distance.
	Busy	BOOL	FALSE	TRUE: FB operation is in progress.
	CommandAborted	BOOL	FALSE	TRUE: An interruption from other FB has occurred.
	Error	BOOL	FALSE	TRUE: An error has occurred.
	ErrorID	SMC_ERROR	0	An error ID is output.

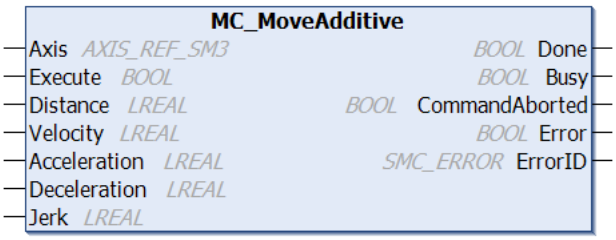
Info.

- Reference manual
  GM1 Controller RTEX User's Manual (Operation Edition)
  GM1 Controller EtherCAT User's Manual (Operation Edition)

5.6.3 MC_MoveAdditive (Target Position Change)

This is a function block (FB) that adds a relative distance to the target position of the immediately preceding instruction. No addition is made to the velocity, acceleration, or deceleration; they change to the specified values.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	Execute	BOOL	FALSE	Starts execution at the rising edge.
	Distance	LREAL	0	Specifies the relative distance (u) to be added.
	Velocity	LREAL	0	Specifies the maximum velocity (u/s).
	Acceleration	LREAL	0	Specifies the acceleration (u/s^2).
	Deceleration	LREAL	0	Specifies the deceleration (u/s^2).
	Jerk	LREAL	0	Specifies the jerk (u/s^3).
Output	Done	BOOL	FALSE	TRUE: The axis has traveled the specified relative distance.
	Busy	BOOL	FALSE	TRUE: The FB is in operation.
	CommandAborted	BOOL	FALSE	TRUE: An interruption from other FB has occurred.
	Error	BOOL	FALSE	TRUE: An error has occurred.
	ErrorID	SMC_ERROR	0	An error ID is output.

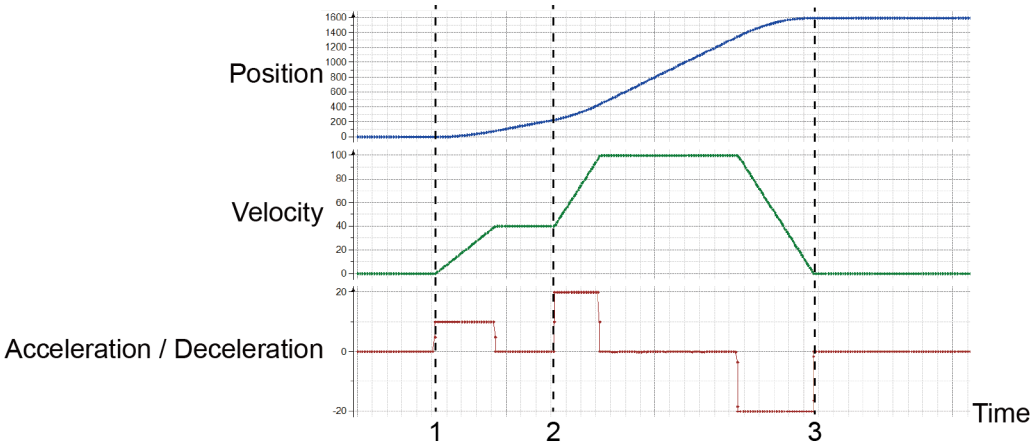
■ Operations when the function block is executed

This example shows the position, velocity, acceleration, and deceleration traces when MC_MoveAdditive is executed while MC_MoveRelative (Relative Value Positioning) is being executed.

Function block input parameters

Execution sequence	Function blocks	Distance (Relative distance)	Velocity (Velocity)	Acceleration (Acceleration)	Deceleration (Deceleration)
1	MC_MoveRelative	1000	40	10	10
2	MC_MoveAdditive	600	100	20	20

Trace



1. MC_MoveRelative is started.
2. MC_MoveAdditive is started.
   MC_MoveRelative is interrupted at the timing when MC_MoveAddive is started. The output parameter CommandAborted of MC_MoveRelative turns TRUE.
   The velocity and acceleration / deceleration change to values (velocity: 100, acceleration / deceleration: 20) specified by MC_MoveAdditive.
3. MC_MoveAdditive is completed.
   The axis travels to the position where the relative distance specified by MC_MoveAdditive is added to the position specified by MC_MoveRelative (1000+600=1600).

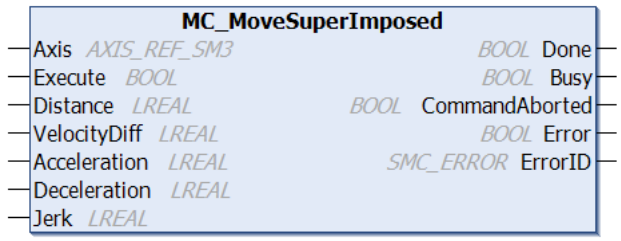
Info.

- Use the MC_MoveSuperImposed function block to cause the axis to travel after adding the velocity, acceleration, and deceleration to the previously executed instruction.
- By setting Distance = 0, it is possible to change the velocity , acceleration, deceleration, and jerk during axis operation.

5.6.4 MC_MoveSuperImposed (Superimposed positioning)

This is a function block (FB) that adds a relative distance, a velocity, an acceleration, and a deceleration to the operations of the immediately preceding instruction.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	Execute	BOOL	FALSE	Starts execution at the rising edge.
	Distance	LREAL	0	Specifies the relative distance (u).
	VelocityDiff	LREAL	0	Specifies the maximum velocity (u/s).
	Acceleration	LREAL	0	Specifies the acceleration (u/s^2).
	Deceleration	LREAL	0	Specifies the deceleration (u/s^2).
	Jerk	LREAL	0	Specifies the jerk (u/s^3).
Output	Done	BOOL	FALSE	TRUE: The axis has traveled the specified relative distance.
	Busy	BOOL	FALSE	TRUE: The FB is in operation.
	CommandAborted	BOOL	FALSE	TRUE: An interruption from other FB has occurred.
	Error	BOOL	FALSE	TRUE: An error has occurred.
	ErrorID	SMC_ERROR	0	An error ID is output.

Note

- When MC_MoveSuperImposed is being executed (Busy = TRUE), be sure to call at every interval.

  If a call is not made, the axis may perform an unexpected operation.

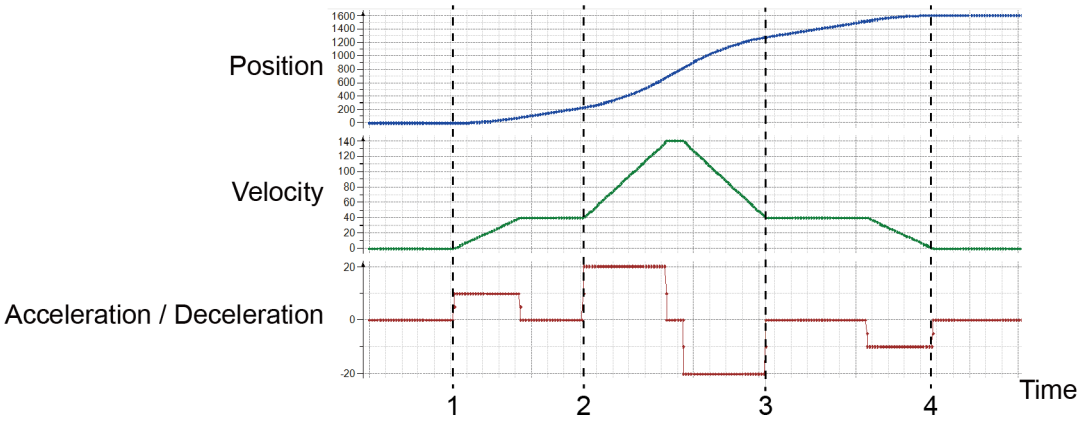
■ Operations when the function block is executed

This example shows the position, velocity, and acceleration traces when MC_MoveSuperImposed is executed while MC_MoveRelative (Relative Value Positioning) is being executed.

Function block input parameters

Execution sequence	Function blocks	Distance (Relative distance)	Velocity (Velocity)	Acceleration (Acceleration)	Deceleration (Acceleration)
1	MC_MoveRelative	1000	40	10	10
2	MC_MoveSuperImposed	600	100	20	20

Trace



1. MC_MoveRelative is started.
2. MC_MoveSuperImposed is started.
   The specified velocity, acceleration, and deceleration are added at the timing when MC_MoveSuperImposed is started.
3. MC_MoveSuperImposed is completed.
4. MC_MoveRelative is completed.
   The axis travels to the position where the relative distance specified by MC_MoveSuperImposed is added to the position specified by MC_MoveRelative (1000+600=1600).

Info.

- Use the MC_MoveAdditive function block to cause the axis to travel at the specified values without adding the velocity, acceleration, and deceleration to the previously executed instruction.

5.6.5 MC_PositionProfile (Position Profile Move)

This is a function block (FB) that causes the axis to operate according to the profile data that consists of a combination of position and time.

■ Icon

■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
	TimePosition	MC_TP_REF	-	Specifies the time/position profile.
Input	Execute	BOOL	FALSE	Starts execution at the rising edge.
	ArraySize	INT	0	A member of the Input / output TimePosition. Specifies the number of points to be executed by FB in the array of time and position that are specified by MC_TP_ARRAY.
	PositionScale	LREAL	1	Position scaling
	Offset	LREAL	0	Position offset
Output	Done	BOOL	FALSE	TRUE: The movements specified by the profile are completed.
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	CommandAborted	BOOL	FALSE	TRUE: An interruption from other FB has occurred.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	An error ID is output.

■ MC_TP_REF (Structure)

Member	Type	Description
Number_of_pairs	INT	Not used
isAbsolute	BOOL	Methods of specifying the position of profile data TRUE: Specified in an absolute value. FALSE: Specified in a relative value.
MC_TP_Array	ARRAY [1..100] OF SMC_TP	Time and position profile data (1st point to 100th point)

■ SMC_TP (Structure)

Member	Type	Description
delta_time	TIME	Time of the profile data
position	LREAL	Position of the profile data

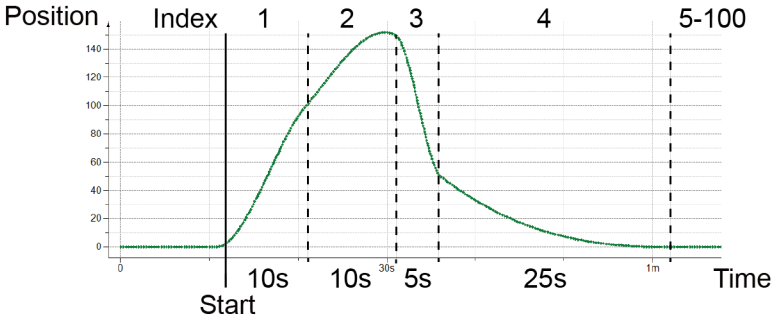
■ Operations when the function block is executed

This example shows time and position traces when the MC_PositionProfile function block is executed with the following parameter settings.

Parameter

Index of the MC_TP_Array array, a member of the input TimePosition	Delta_time (time)	Position (absolute position)
1	Time#10s	100
2	Time#10s	150
3	Time#5s	50
4	Time#25s	0
5 to 100	Time#0ms	0

Trace



Note

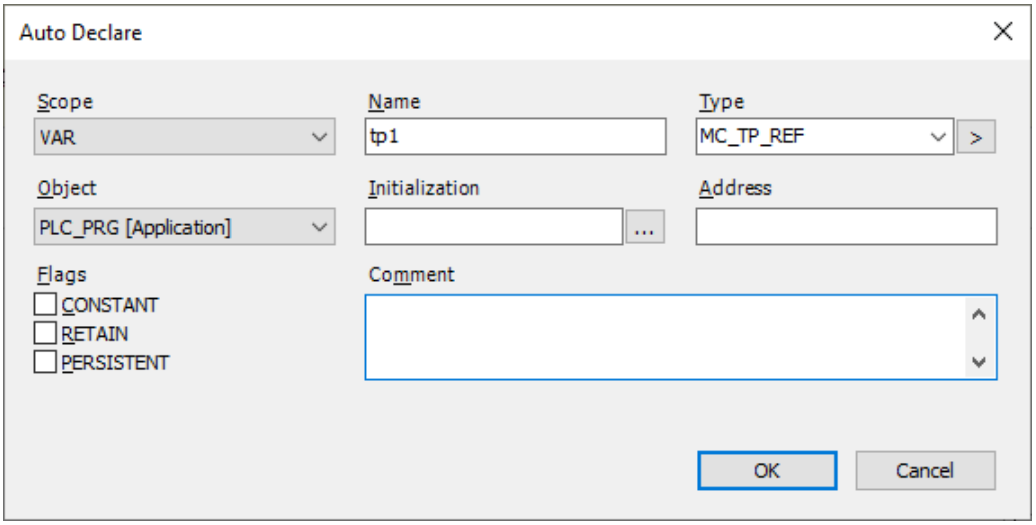
- While the axis keeps driving, do not specify delta_time to 0 ms. Otherwise, it may cause an abnormal operation.

5.6.6 Default Setting for Variables of the MC_TP_REF Type Structure

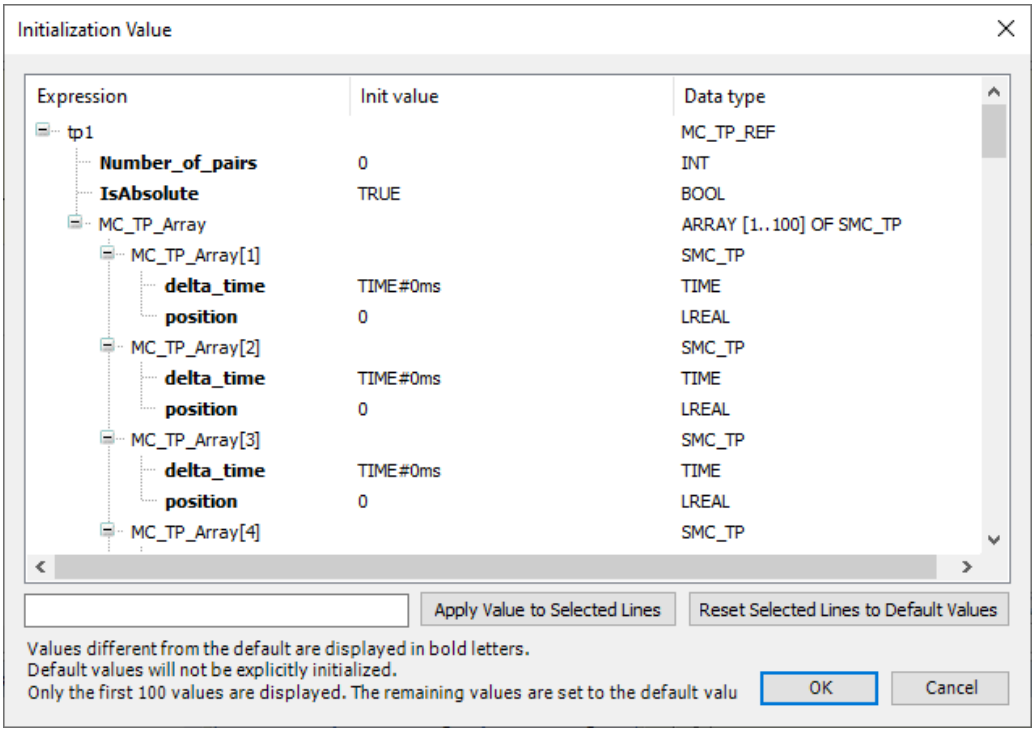
To enter the value of the input TimePosition, it is necessary to make default setting for variables of the MC_TP_REF type structure.

Procedure

1. When the input variable to TimePosition is declared, "Automatic Declaration" dialog box is displayed. Click ... displayed next to the "Initial Value" field.



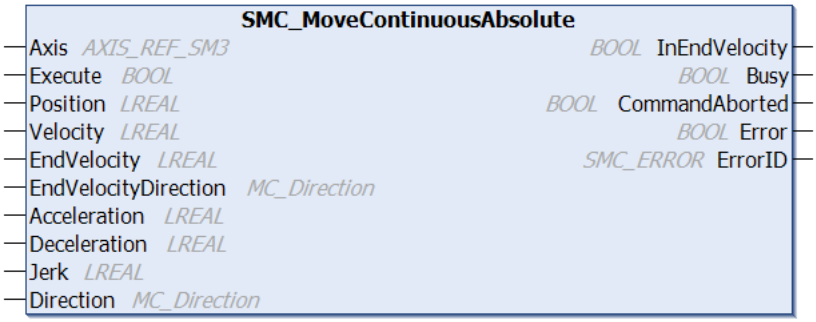
2. The "Initial Value" dialog box is displayed and, on the dialog box, you can set the default value for every member of the variable type (MC_TP_REF).



5.6.7 SMC_MoveContinuousAbsolute (Absolute Value Position Velocity Move)

This is a function block (FB) that executes absolute value positioning and, after the axis reaches the target position, causes the axis to keep moving at a specified velocity.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	Execute	BOOL	FALSE	Starts execution at the rising edge.
	Position	LREAL	0	Specifies the target position (u).
	Velocity	LREAL	0	Specifies the velocity (u/s) until the axis reaches the target position.
	EndVelocity	LREAL	0	Specifies the velocity (u/s) after the axis reaches the target position.
	EndVelocityDirection	MC_Direction	current	Specifies the traveling direction after the axis reaches the target. Specifies either “positive”, “negative”, or “current”. If “fastest” or “shortest” is specified, an error occurs.
	Acceleration	LREAL	0	Specifies the acceleration (u/s^2).
	Deceleration	LREAL	0	Specifies the deceleration (u/s^2).
	Jerk	LREAL	0	Specifies the jerk (u/s^3).
	Direction	MC_Direction	shortest	Specifies the traveling direction of the axis until the axis reaches the target position. Possible to specify only for the modulo type. For the finite axis, the specification is ignored.
Output	InEndVelocity	BOOL	FALSE	TRUE: The axis has reached the target position (Position).
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.

Scope	Name	Type	Initial	Description
	CommandAborted	BOOL	FALSE	TRUE: An interruption from other FB has occurred.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	An error ID is output.

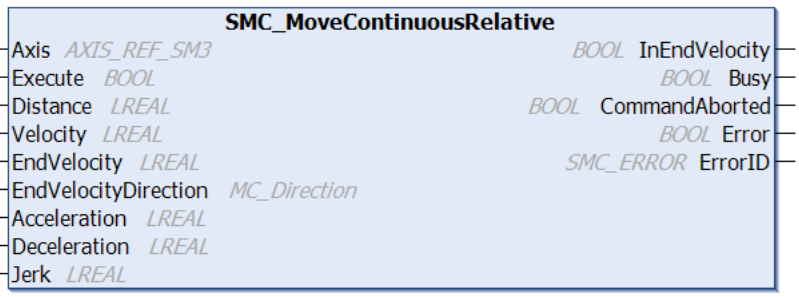
■ MC_Direction (Enumeration type)

Name	Value	Description
positive	1	Travels in the positive direction.
negative	-1	Travels in the negative direction.
shortest	0	Travels in the shortest direction from the current command position to the target command position when SMC_MoveContinuousAbsolute is executed.
fastest	3	Travels in the fastest direction from the current command position to the target command position when SMC_MoveContinuousAbsolute is executed. If the axis is being driven by another function block when SMC_MoveContinuousAbsolute is executed, selects the fastest direction within the GM1. If the axis is being stopped when SMC_MoveContinuousAbsolute is executed, makes the same movement as for the shortest.
current	2	Travels to the current direction. If the axis is being driven by another function block when SMC_MoveContinuousAbsolute is executed, travels in the same direction. If the axis is being stopped when SMC_MoveContinuousAbsolute is executed, travels in the direction moved by the previously executed function block.

5.6.8 SMC_MoveContinuousRelative (Relative Value Position Velocity Move)

This is a function block (FB) that executes relative value positioning and, after the axis reaches the target position, causes the axis to keep moving at a specified velocity.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	Execute	BOOL	FALSE	Starts execution at the rising edge.
	Distance	LREAL	0	Specifies the relative distance (u).
	Velocity	LREAL	0	Specifies the velocity (u/s).
	EndVelocity	LREAL	0	Specifies the velocity (u/s) after the axis travels the relative distance.
	EndVelocityDirection	MC_Direction	current	Specifies the traveling direction after the axis travels the relative distance. Specifies either “positive”, “negative”, or “current”. If “fastest” or “shortest” is specified, an error occurs.
	Acceleration	LREAL	0	Specifies the acceleration (u/s^2).
	Deceleration	LREAL	0	Specifies the deceleration (u/s^2).
	Jerk	LREAL	0	Specifies the jerk (u/s^3).
Output	InEndVelocity	BOOL	FALSE	TRUE: The axis has traveled the specified relative distance and has reached the specified velocity.
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	CommandAborted	BOOL	FALSE	TRUE: An interruption from other FB has occurred.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	An error ID is output.

■ MC_Direction (Enumeration type)

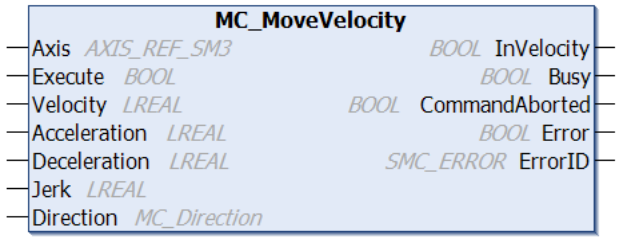
Name	Value	Description
positive	1	Travels in the positive direction.
negative	-1	Travels in the negative direction.
shortest	0	Not available. Do not specify this.
fastest	3	Not available. Do not specify this.
current	2	Travels to the current direction. Possible to use only for the modulo axis.

5.7 Velocity Control

5.7.1 MC_MoveVelocity (Velocity Control)

This is a function block (FB) that specifies the velocity of the axis. The axis keeps moving at the specified velocity and direction.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	Execute	BOOL	FALSE	Starts execution at the rising edge.
	Velocity	LREAL	0	Specifies the velocity (u/s).
	Acceleration	LREAL	0	Specifies the acceleration (u/s^2).
	Deceleration	LREAL	0	Specifies the deceleration (u/s^2).
	Jerk	LREAL	0	Specifies the jerk (u/s^3).
	Direction	MC_Direction	current	Specifies the traveling direction of the axis. “positive”, “negative”, or “current” (An error occurs when “fastest” or “shortest” is selected)
Output	InVelocity	BOOL	FALSE	TRUE: The axis has reached the specified velocity for the first time.
	Busy	BOOL	FALSE	TRUE: The FB is in operation.
	CommandAborted	BOOL	FALSE	TRUE: An interruption from other FB has occurred.
	Error	BOOL	FALSE	TRUE: An error has occurred.
	ErrorID	SMC_ERROR	0	An error ID is output.

■ MC_Direction (Enumeration type)

Name	Value	Description
positive	1	Travels in the positive direction.
negative	-1	Travels in the negative direction.

Name	Value	Description
shortest	0	Not available. Do not specify this.
fastest	3	Not available. Do not specify this.
current	2	Travels to the current direction. Possible to use only for the modulo axis.

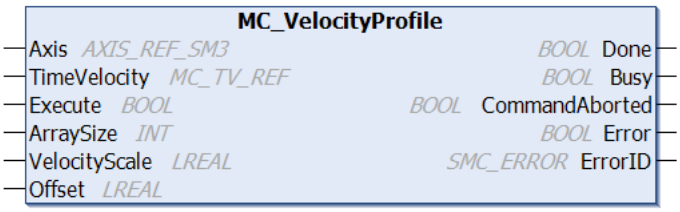
Info.

- To stop speed control, execute "5.4.1 MC_Stop (Forced Stop)" or "5.4.2 MC_Halt (Halt)".
- It can be executed not only in speed control mode but also in position control mode.
- Reference manual
  GM1 Controller RTEX User's Manual (Operation Edition)
  GM1 Controller EtherCAT User's Manual (Operation Edition)

5.7.2 MC_VelocityProfile (Velocity Profile Movement)

This is a function block (FB) that causes the axis to operate according to the profile data that consists of a combination of time and velocity.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
	TimeVelocity	MC_TV_REF	-	Specifies the time / velocity profile.
Input	Execute	BOOL	FALSE	Starts execution at the rising edge.
	ArraySize	INT	0	A member of input TimeVelocity. Specifies the number of points to be executed by FB in the array of time and velocity that are specified by MC_TV_ARRAY.
	VelocityScale	LREAL	1	Velocity scaling
	Offset	LREAL	0	Velocity offset (u)
Output	Done	BOOL	FALSE	TRUE: The movements specified by the profile are completed.
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.

Scope	Name	Type	Initial	Description
	CommandAborted	BOOL	FALSE	TRUE: An interruption from other FB has occurred.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	An error ID is output.

■ MC_TV_REF (Structure)

Member	Type	Description
Number_of_pairs	INT	Not used
isAbsolute	BOOL	Methods of specifying the velocity of profile data TRUE: Specified in an absolute value. FALSE: Specified in a relative value.
MC_TV_Array	ARRAY [1..100] OF SMC_TV	Time and velocity profile data (1st point to 100th point)

■ SMC_TV (Structure)

Member	Type	Description
delta_time	TIME	Time of the profile data
velocity	LREAL	Velocity of profile data

Regarding the method for entering defaults for variables of the MC_TV_REF Type Structure, refer to “Default Setting for Variables of the MC_TP_REF Type Structure”.

Note

- While the axis keeps driving, do not specify delta_time to 0 ms.
  Otherwise, it may cause an abnormal operation.

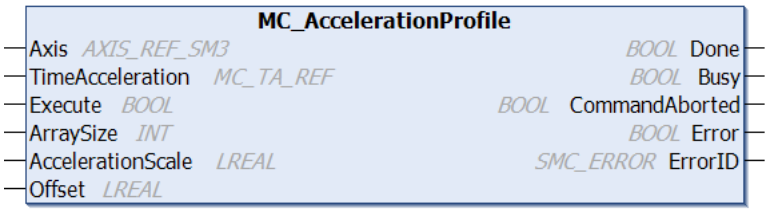
REFERENCE

5.6.6 Default Setting for Variables of the MC_TP_REF Type Structure

5.7.3 MC_AccelerationProfile (Acceleration Profile Movement)

This is a function block (FB) that causes the axis to operate according to the profile data that consists of a combination of time and acceleration.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
	TimeAcceleration	MC_TA_REF	-	Specifies the time / acceleration profile.
Input	Execute	BOOL	FALSE	Starts execution at the rising edge.
	ArraySize	INT	0	A member of the input TimeAccleration. Specifies the number of points to be executed by FB in the array of time and acceleration that are specified by MC_TA_ARRAY.
	AccelerationScale	LREAL	1	Acceleration scaling
	Offset	LREAL	0	Acceleration offset (u)
Output	Done	BOOL	FALSE	TRUE: The movements specified by the profile are completed.
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	CommandAborted	BOOL	FALSE	TRUE: An interruption from other FB has occurred.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	An error ID is output.

■ MC_TA_REF (Structure)

Member	Type	Description
Number_of_pairs	INT	Not used
isAbsolute	BOOL	Methods of specifying the acceleration of profile data TRUE: Specified in an absolute value. FALSE: Specified in a relative value.
MC_TA_Array	ARRAY [1..100] OF SMC_TA	Time and acceleration profile data (1st point to 100th point)

■ SMC_TA (Structure)

Member	Type	Description
delta_time	TIME	Time of the profile data

Member	Type	Description
Acceleration	LREAL	Acceleration of profile data

Regarding the method for entering defaults for variables of the MC_TA_REF Type Structure, refer to “Default Setting for Variables of the MC_TP_REF Type Structure”.

Note

- While the axis keeps driving, do not specify delta_time to 0 ms.
  Otherwise, it may cause an abnormal operation.

REFERENCE

5.6.6 Default Setting for Variables of the MC_TP_REF Type Structure

5.8 Torque Control

5.8.1 PMC_SetTorque (Torque Control)

Specifies by % the torque of the axis.

■ **Icon**

■ **Parameter**

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	bEnable	BOOL	FALSE	TRUE: The FB can be executed.
	fTorque	LREAL	0	Specifies the ratio (%) to the rated torque
Output	bBusy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	bError	BOOL	FALSE	TRUE: An error has occurred within the FB.
	nErrorID	SMC_ERROR	0	An error ID is output.

Info.

- This function block can be executed only in the torque control mode (SMC_torque). Change to the torque control mode in advance using MC_SetControllerMode.
- Confirm that the MINAS firmware is the latest version.
- When stopping the axis while the axis is being controlled in the torque control mode (SMC_torque), set fTorque to 0 and execute again. Or, change to the position control mode using SMC_SetControllerMode and then stop the axis using MC_Stop.
- When performing torque control for MINAS A5B/A6B, the speed is limited by the Max motorspeed (16#6080:00) set in the PDO mapping. Set the speed in advance.
- Reference manual

  GM1 Controller RTEX User's Manual (Operation Edition)

  GM1 Controller EtherCAT User's Manual (Operation Edition)

REFERENCE

5.3.1 SMC_SetControllerMode (Control Mode Setting)
5.4.1 MC_Stop (Forced Stop)

5.8.2 SMC_SetTorque (Torque Control)

Specifies by Nm the torque of the axis.

■ Icon

■ Parameter

Scope	Name	Type	Default	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	bEnable	BOOL	FALSE	TRUE: The FB can be executed.
	fTorque	LREAL	0	Specifies the torque (N·m, N)
Output	bBusy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	bError	BOOL	FALSE	TRUE: An error has occurred within the FB.
	nErrorID	SMC_ERROR	0	Error ID output

Info.

- This function block can be executed only in the torque control mode (SMC_torque). Change to the torque control mode in advance using SMC_SetControllerMode
- Confirm that the MINAS firmware is the latest version.
- When stopping the axis while the axis is being controlled in the torque control mode (SMC_torque), set fTorque to 0 and execute again. Or, change to the position control mode (SMC_position) using SMC_SetControllerMode and then stop the axis using MC_Stop.
- When performing torque control for MINAS A5B/A6B, the speed is limited by the Max motorspeed (16#6080:00) set in the PDO mapping. Set the speed in advance.
- Reference manual

  GM1 Controller RTEX User's Manual (Operation Edition)

  GM1 Controller EtherCAT User's Manual (Operation Edition)

(MEMO)

6 Motion Control Function Blocks (Synchronous Control)

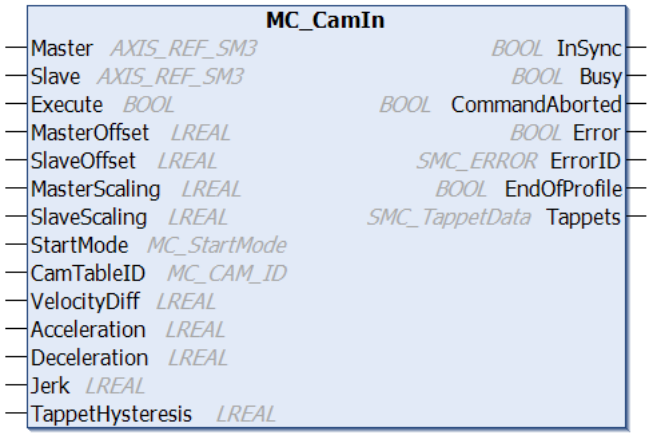
This section describes motion control function blocks to perform synchronous processing.

6.1 Cam Operation

6.1.1 MC_CamIn (Start Cam Operation)

This is a function block (FB) that starts cam synchronous operation. The master axis and the slave axis operate in synchronization according to the cam table.

■ Icon



■ Parameter

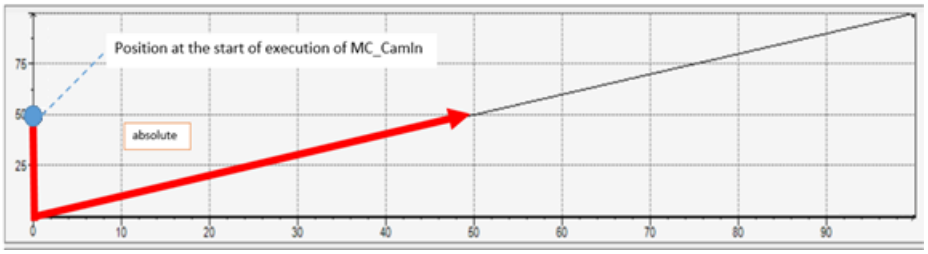
Scope	Name	Type	Initial	Comment
Input / output	Master	AXIS_REF_SM3	-	Specifies the master axis.
	Slave	AXIS_REF_SM3	-	Specifies the slave axis.
Input	Execute	BOOL	FALSE	Starts execution at the rising edge.
	MasterOffset	LREAL	0	Master axis table offset
	SlaveOffset	LREAL	0	Slave axis table offset
	MasterScaling	LREAL	1	Master axis profile scaling factor
	SlaveSacling	LREAL	1	Slave axis profile scaling factor
	StartMode	MC_StartMode	absolute	Start mode
	CamTableID	MC_CAM_ID	0	Cam table ID Specifies the output for MC_CamTableSelect.
	VelocityDiff	LREAL	0	Specifies the maximum velocity difference (u/s) when StartMode is set to ramp_in.
	Acceleration	LREAL	0	Specifies the acceleration (u/s^2) when StartMode is set to ramp_in.
	Deceleration	LREAL	0	Specifies the deceleration (u/s^2) when StartMode is set to ramp_in.

Scope	Name	Type	Initial	Comment
	Jerk	LREAL	0	Specifies jerk (u/s^3) when StartMode is set to ramp_in.
	TappetHysteresis	LREAL	0	Specifies the hysteresis value (position) of the tappet.
Output	InSync	BOOL	FALSE	TRUE: The cam is in synchronization for the first time.
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	CommandAborted	BOOL	FALSE	TRUE: An interruption from other FB has occurred.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	An error ID is output.
	EndOfProfile	BOOL	FALSE	A pulse is output every time the cam profile period of the slave axis ends.
	Tappets	SMC_TAPPETDATA		Output tappet data Used as the input for SMC_GetTappetValue.

■ MC_StartMode (Enumeration type)

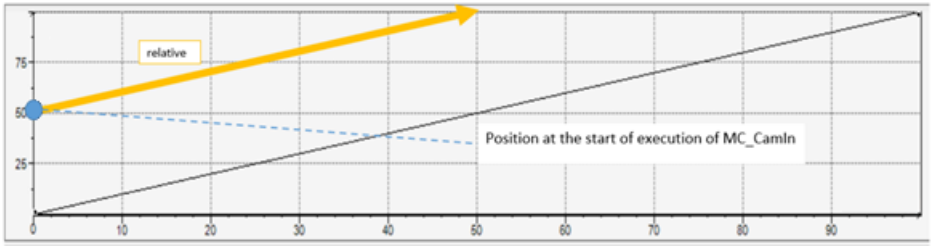
- absolute

  If the slave axis position is offset from the position on the cam table when MC_CamIn is executed, the position jumps. When starting a new cycle, the cam does not consider the current slave axis position. Therefore, if the slave axis position relative to the mater axis at the start point is offset from the slave axis position relative to the mater axis at the end point, the slave axis position may jump.



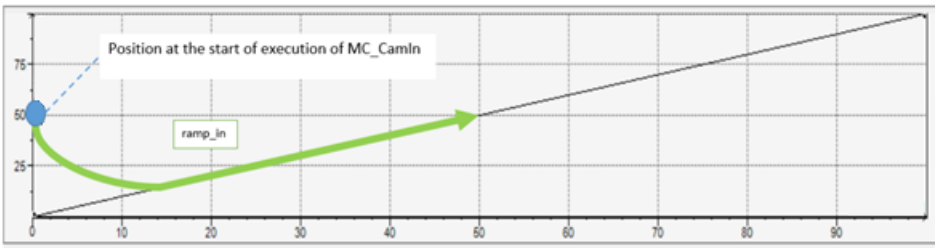
- relative

  If the slave axis position is offset from the position on the cam table when MC_CamIn is executed, the position jumps. In consideration of the current slave axis position, new cam operation is started. The slave axis position at the time when the previous cycle is completed is reflected as SlaveOffset on the cam operation. It must be noted, however, that the position may jump if the master axis position is not 0 at the start position.



- ramp_in, ramp_in_neg, ramp_in_pos

  When MC_CamIn is executed, acceleration and deceleration is performed according to VelocityDiff, Acceleration, and Deceleration to reach the synchronized state (InSync). If the slave axis is a modulo axis, correction is made only in the positive direction when the mode is set to ramp_in_pos, while correction is made only in the negative direction when the mode is set to ramp_in_neg. With the finite axis, ramp_in_pos and ramp_in_neg are treated as ramp_in.



The final StartMode is determined by MC_CamIn.StartMode and MC_CamTableSelect.SlaveAbsolute.

MC_CamIn.StartMode	MC_CamTableSelect.SlaveAbsolute	StartMode
absolute	TRUE	absolute
absolute	FALSE	relative
relative	TRUE	relative
relative	FALSE	relative
ramp_in	TRUE	ramp_in absolute
ramp_in	FALSE	ramp_in relative
ramp_in_pos	TRUE	ramp_in_pos absolute
ramp_in_pos	FALSE	ramp_in_pos relative
ramp_in_neg	TRUE	ramp_in_neg absolute
ramp_in_neg	FALSE	ramp_in_neg relative

■ Changing the scale and offset in the cam table (MasterOffset, SlaveOffset, MasterScaling, SlaveSacling)

A cam table can be created by using the Cam Table Editor on the GM Programmer. The cam table created by using the Cam Table Editor determines the relationship between the master axis position and the slave axis position that operate on the POU,

SlavePosition = CAM(MasterPosition)

The scale and offset in the cam table can be changed on the POU by setting the MC_CamIn parameters including MasterOffset, SlaveOffset, sterScaling, and SlaveSacling. At that time, the slave axis position is determined as follows according to the setting of each parameter.

SlavePosition = SlaveSacling * CAM(MasterPosition * MasterScaling + MasterOffset) + SlaveOffset

■ Tappet

For information on setting the tappet and TappetHysteresis of MC_CamIn, refer to the description on SMC_GetTappetValue

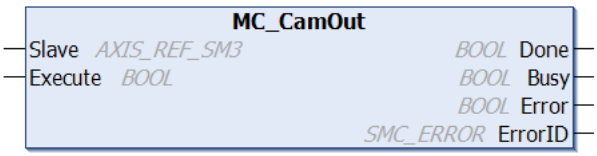
■ SMC_TAPPETDATA (Structure)

Member	Type	Description
pTaps	ARRAY [0..2] OF POINTER TO SMC_CAMTappet	Used internally for the output of MC_CamIn and for the input of SMC_GetTappetValue.
dwCycleTime	DWORD	
byChannels	BYTE	
bRestart	BOOL	

6.1.2 MC_CamOut (Cancel Cam Operation)

This is a function block (FB) that cancels synchronous operation of the cam. Synchronized operation between the master axis and slave axis is canceled.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Slave	AXIS_REF_SM3	-	Specifies the slave axis to be released.
Input	Execute	BOOL	FALSE	Starts execution at the rising edge.
Output	Done	BOOL	FALSE	TRUE: Synchronization cancellation is completed.
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	An error ID is output.

Info.

- The slave axis operation continues even after the cam operation is canceled. Execute MC_Halt or MC_Stop to stop the slave axis.
- Reference manual

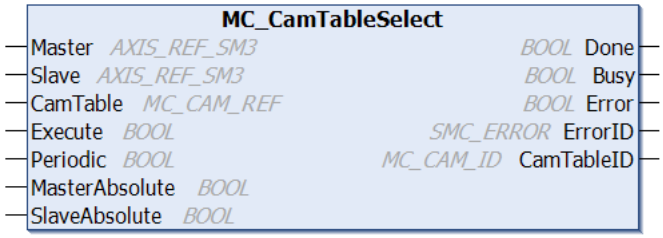
  GM1 Controller RTEX User's Manual (Operation Edition)

  GM1 Controller EtherCAT User's Manual (Operation Edition)

6.1.3 MC_CamTableSelect (Select Cam Table)

This is a function block (FB) that specifies the cam table to be used for synchronous operation of the cam. When the cam table to be used for synchronized operation between the master axis and slave axis is selected, a cam table ID is output.

■ Icon



■ Parameter

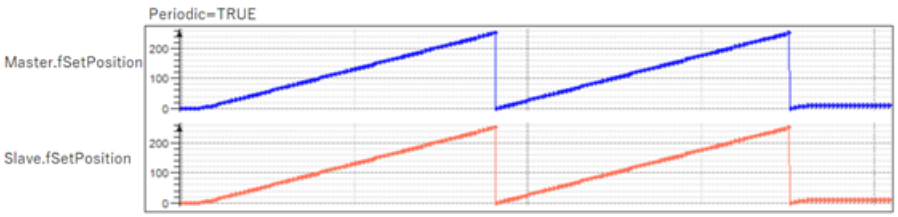
Scope	Name	Type	Initial	Description
Input / output	Master	AXIS_REF_SM3	-	Specifies the master axis.
	Slave	AXIS_REF_SM3	-	Specifies the slave axis.
	CamTable	MC_CAM_REF	-	Specifies the cam table.
Input	Execute	BOOL	FALSE	Starts execution at the rising edge.
	Periodic	BOOL	TRUE	Movement of the slave axis TRUE: Repeat execution FALSE: 1-period execution
	MasterAbsolute	BOOL	TRUE	TRUE: Absolute position of the master axis FALSE: Relative position of the master axis
	SlaveAbsolute	BOOL	TRUE	TRUE: Absolute position of the slave axis FALSE: Relative position of the slave axis
Output	Done	BOOL	FALSE	TRUE: Selection is completed.
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.

Scope	Name	Type	Initial	Description
	ErrorID	SMC_ERROR	0	An error ID is output.
	CamTableID	MC_CAM_ID		Cam table ID Used as the input for MC_CamIn.

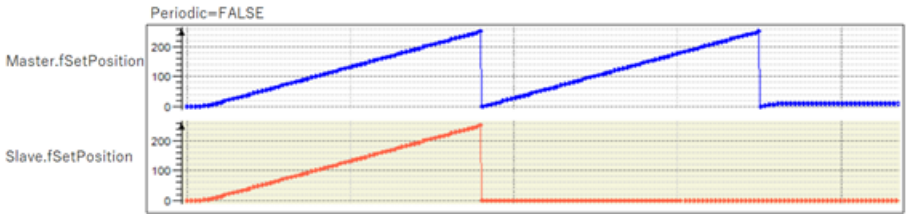
■ Periodic (Periodic cam control)

If Periodic of MC_CamTableSelect is set to TRUE, cam operation is repeatedly performed. The cam is automatically restarted when reaching the end position. If Periodic is set to FALSE, when the master axis reaches the end position, EndOfProfile of MC_CamIn changes to TRUE and the slave axis stops at the current position. When the master axis enters the cam position range again, the slave axis starts moving according to the cam table.

If periodic = TRUE

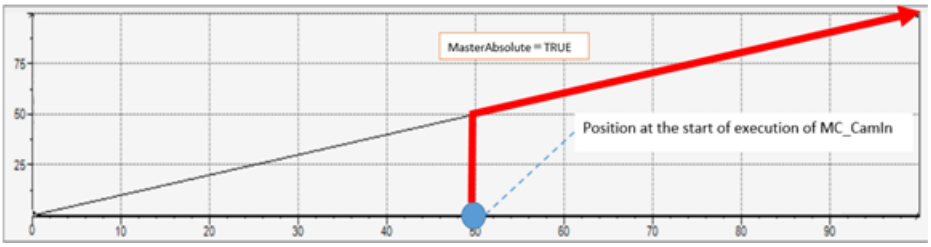


If periodic = FALSE

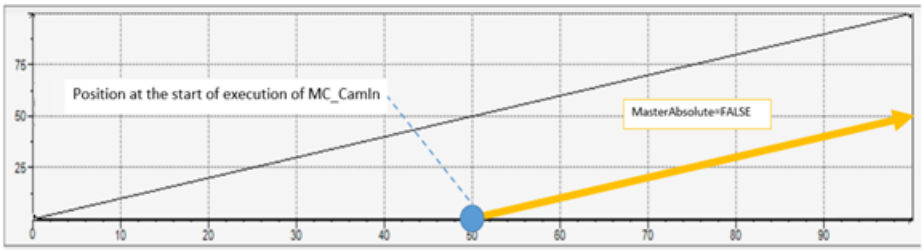


■ MasterAbsolute

If MasterAbsolute is set to TRUE, the cam starts from the current master axis position. Any position within the cam table range specified for the master axis can be set as the starting position. If the starting position is outside the cam table range, an error occurs.



If MasterAbsolute is set to FALSE, the cam is relocated to the current master axis position. The zero point of the master axis in the cam table is relocated to the current master axis position. This mode is allowed only when the value 0 is within the master axis range.



■ SlaveAbsolute

Refer to MC_CamIn.

Info.

- Reference manual
  GM1 Controller RTEX User's Manual (Operation Edition)
  GM1 Controller EtherCAT User's Manual (Operation Edition)

6.1.4 SMC_GetTappetValue (Tappet Output)

This is a function block (FB) that outputs the tappet data set using the cam table. Specify the output Tappets of the MC_CamIn function block as the input for this FB and obtain the tappet output for one track.

■ Icon

■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Tappets	SMC_TAPPETDATA	-	Specifies the output Tappets of MC_CamIn.
Input	iID	INT	0	Track ID of the tappet
	bInitValue	BOOL	FALSE	Default of the tappet
	bSetInitValueAtReset	BOOL	FALSE	TRUE: The tappet output is Initialized when MC_CamIn is executed. FALSE: The tappet output is held when MC_CamIn is executed.
Output	bTappet	BOOL	FALSE	Tappet output

■ SMC_TAPPETDATA (Structure)

Member	Type	Description
pTaps	ARRAY [0..2] OF POINTER TO SMC_CAMTappet	Used internally for the output of MC_CamIn and for the input of SMC_GetTappetValue.
dwCycleTime	DWORD	
byChannels	BYTE	
bRestart	BOOL	

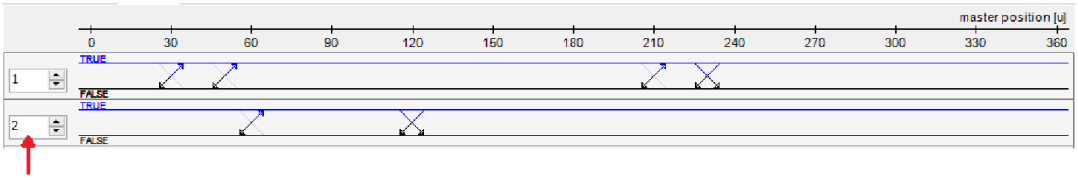
■ Tappet settings

Set tappet settings on the "Tappet" tab or "Tappet Table" tab window.

The settings that are set on these two tab windows are linked to each other. They can be set from either window.

Set switches required for the tracks (100 max).
The following is an example where three switches are set for two tracks.

"Tappet"



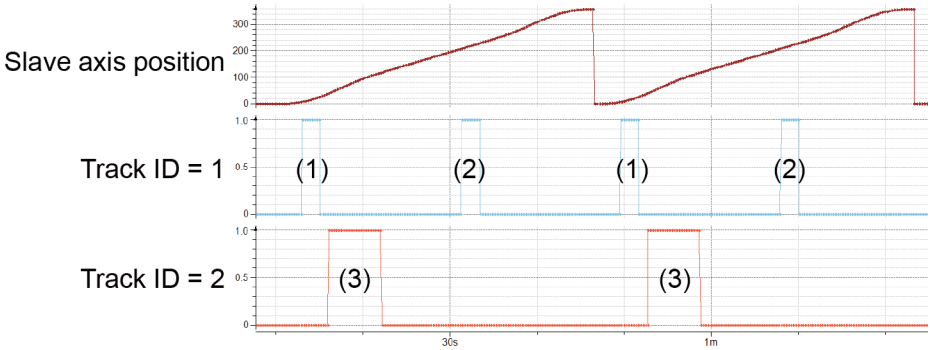
"Tappet Table"

	Track ID	X	positive pass	negative pass	
	1				
		30	switch ON	switch OFF	Switch (1)
		50	switch ON	switch OFF	
		210	switch ON	switch OFF	Switch (2)
		230	switch OFF	switch OFF	
	2				
		60	switch ON	switch OFF	Switch (3)
		120	switch OFF	switch OFF	

■ Operations when the function block is executed

The operations that take place when, after setting the tappet, the SMC_GetTappetValue function block is executed and the tappet (bTappet) is output. The axis is set to the modulo (modulo value: 360).

Trace
(1) to (3) are switch numbers.



■ **TappetHysteresis of MC_CamIn**

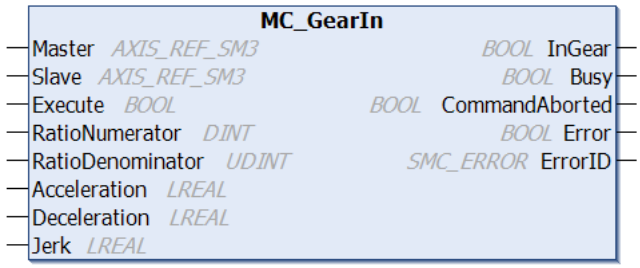
By setting TappetHysteresis, mechanical chattering can be filtered. When the cam reaches the tappet position, tappet processing is performed. Once the cam moves out of the tappet position range set in TappetHysteresis, tappet processing is not performed unless the cam reaches the tappet position again.

6.2 Gear Operation

6.2.1 MC_GearIn (Start Gear Operation)

This is a function block (FB) that starts synchronous operation of the gears. Specify the gear ratio between the master axis and slave axis and start moving the gears for synchronous operation of the gears.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Master	AXIS_REF_SM3	-	Specifies the master axis.
	Slave	AXIS_REF_SM3	-	Specifies the slave axis.
Input	Execute	BOOL	FALSE	Starts execution at the rising edge.
	RatioNumerator	DINT	1	Specifies the gear ratio (numerator).
	RatioDenominator	UDINT	1	Specifies the gear ratio (denominator).
	Acceleration	LREAL	0	Maximum acceleration (u/s^2) until gear synchronization is completed
	Deceleration	LREAL	0	Maximum deceleration (u/s^2) until gear synchronization is completed
	Jerk	LREAL	0	Maximum jerk (u/s^3) until gear synchronization is completed
Output	InGear	BOOL	FALSE	TRUE: Gear synchronization is completed.
	Busy	BOOL	FALSE	TRUE: The FB is in operation.
	CommandAborted	BOOL	FALSE	TRUE: An interruption from other FB has occurred.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	An error ID is output.

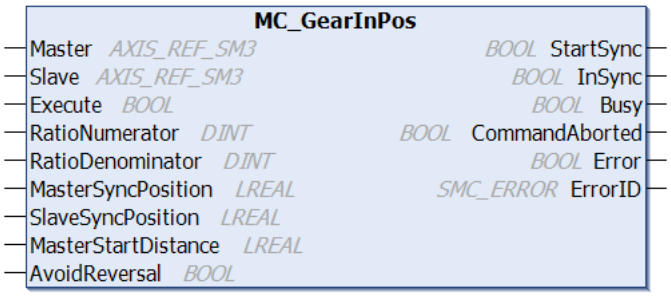
Info.

- Reference manual
  GM1 Controller RTEX User's Manual (Operation Edition)
  GM1 Controller EtherCAT User's Manual (Operation Edition)

6.2.2 MC_GearInPos (Position Specified Gear Operation)

This is a function block (FB) that starts synchronous operation of the gears from the specified absolute position.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Master	AXIS_REF_SM3	-	Specifies the master axis.
	Slave	AXIS_REF_SM3	-	Specifies the slave axis.
Input	Execute	BOOL	FALSE	Starts execution at the rising edge.
	RatioNumerator	DINT	1	Specifies the gear ratio (numerator).
	RatioDenominator	DINT	1	Specifies the gear ratio (denominator).
	MasterSyncPosition	LREAL	0	Master axis position to start synchronization
	SlaveSyncPosition	LREAL	0	Slave axis position to start synchronization
	MasterStartDistance	LREAL	0	When the master axis moves forward from the MasterSyncPosition position for the distance specified by MasterStartDistance, the slave axis starts moving for the synchronized operation with the master axis. If MasterStartDistance is zero, the slave axis immediately starts moving for synchronization.
	AvoidReversal	BOOL	FALSE	● Axis setting: When set to "Finite" AvoidReversal = TRUE

Scope	Name	Type	Initial	Description
				● Axis setting: When set to "Modulo" AvoidReversal = FALSE
Output	StartSync	BOOL	FALSE	TRUE: Gear synchronization is started.
	InSync	BOOL	FALSE	TRUE: Gear synchronization is completed.
	Busy	BOOL	FALSE	TRUE: The FB is in operation.
	CommandAborted	BOOL	FALSE	TRUE: An interruption from other FB has occurred.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	An error ID is output.

■ Operations when the function block is executed

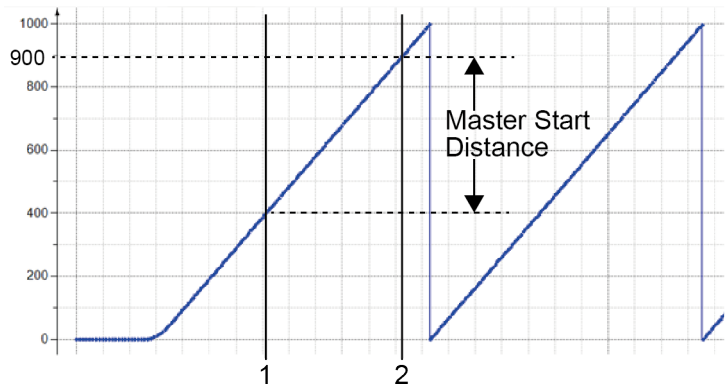
This example shows the trace when the MC_GearInPos function block is executed with the following conditions.

Execution condition

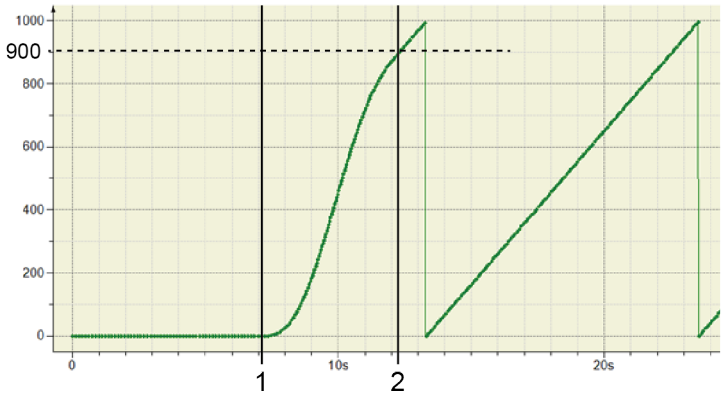
Item	Dis
Master axis type	Modulo (modulo value = 1000)
Slave axis type	Modulo (modulo value = 1000)
Gear ratio	1 : 1
Input MasterSyncPostion	900
Input SlaveSyncPositon	900
Input MasterStartDistance	500

When the master axis position reaches 900 and the salve axis position reaches 900, the master axis starts to synchronize with the slave axis. When the master axis passes the position 400, which is obtained by deducting 500 (MasterStartDistane) from 900 (synchronization start position of the master axis), the slave axis starts traveling to synchronize with the master axis. At this time, velocity, acceleration, and deceleration are automatically determined.

Position of the master axis



Position of the slave axis



■ AvoidReversal

By setting AvoidReversal, the slave axis can be restricted on reverse rotation. If AvoidReversal is set to TRUE, an error occurs under the following conditions.

1. Gear ratio is negative.

   If the gear ratio is negative (for example, RatioNumerator = -1, RatioDenominator = 1), when the axis reaches the position set in GearInPos.StartSync while the slave axis is operating in forward rotation, an error (SMC_GIP_SLAVE_REVERSAL_CANNOT_BE_AVOIDED) occurs
2. The slave axis is rotating in reverse to the rotation of the master axis before the start of synchronization

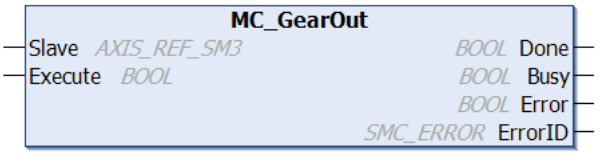
   When the axis reaches the gear synchronization start position set in StartSync while the slave axis is operating in reverse rotation, an error (SMC_GIP_SLAVE_REVERSAL_CANNOT_BE_AVOIDED) occurs
3. Correction of the slave axis is not completed within five cycles.

   Gear synchronization completion (InSync) is not achieved within five cycles after reaching the gear synchronization start (StartSync), an error occurs.

6.2.3 MC_GearOut (Cancel Gear Operation)

This is a function block (FB) that cancels synchronous operation of the gears. Synchronized gear operation between the master axis and slave axis is canceled.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Slave	AXIS_REF_SM3	-	Specifies the axis.
Input	Execute	BOOL	FALSE	Starts execution at the rising edge.
Output	Done	BOOL	FALSE	TRUE: Synchronization cancellation is completed.
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	An error ID is output.

Info.

- The slave axis operation continues even after the gear operation is canceled. Execute MC_Halt or MC_Stop to stop the slave axis.
- Reference manual

  GM1 Controller RTEX User's Manual (Operation Edition)

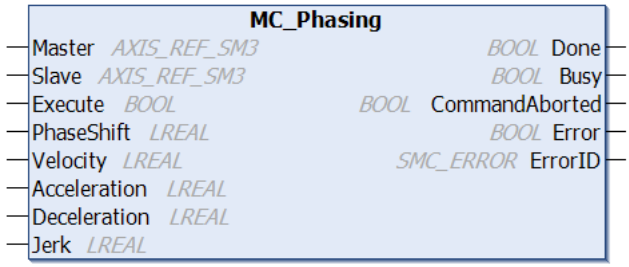
  GM1 Controller EtherCAT User's Manual (Operation Edition)

6.3 Phase Correction

6.3.1 MC_Phasing (Master Axis Phase Correction)

This is a function block (FB) that performs phase correction between the master axis and slave axis. Phase synchronous operation can be performed by making phase correction for the master axis.

■ **Icon**



■ **Parameter**

Scope	Name	Type	Initial	Description
Input / output	Master	AXIS_REF_SM3	-	Specifies the master axis.
	Slave	AXIS_REF_SM3	-	Specifies the slave axis.
Input	Execute	BOOL	FALSE	Starts execution at the rising edge.
	PhaseShift	LREAL	0	Specifies the phase between the master axis and slave axis.
	Velocity	LREAL	0	Specifies the velocity (u/s).
	Acceleration	LREAL	0	Specifies the acceleration (u/s^2).
	Deceleration	LREAL	0	Specifies the deceleration (u/s^2).
	Jerk	LREAL	0	Specifies the jerk (u/s^3).
Output	Done	BOOL	FALSE	TRUE: Phase correction is completed.
	Busy	BOOL	FALSE	TRUE: The FB is in operation.
	CommandAborted	BOOL	FALSE	TRUE: An interruption from other FB has occurred.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	An error ID is output.

Info.

- Reference manual
  GM1 Controller RTEX User's Manual (Operation Edition)
  GM1 Controller EtherCAT User's Manual (Operation Edition)

7 Motion Control Function Blocks (Interpolation Control)

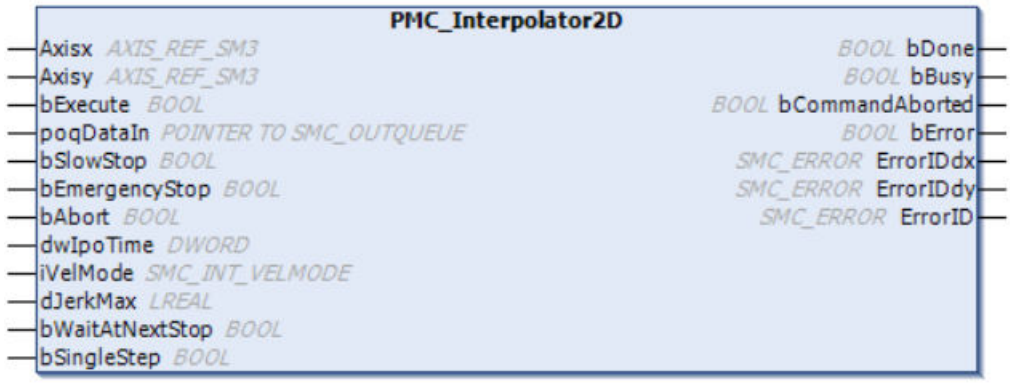
This section describes function blocks used to perform interpolation control using the CNC program.

7.1 Interpolation Control

7.1.1 PMC_Interpolator2D (2-axis Interpolation Control)

This function block (FB) performs 2-axis interpolation control according to the specified CNC table.

■ Icon



■ Parameter

Scope	Name	Type	Default value	Description
I/O	Axisx	AXIS_REF_SM3	-	Specifies the x-axis.
	Axisy	AXIS_REF_SM3	-	Specifies the y-axis.
Input	bExecute	BOOL	FALSE	Starts execution at the rising edge.
	poqDataIn	POINTER TO SMC_OUTQUEUE	-	Specifies a pointer to the CNC table.
	bSlowStop	BOOL	FALSE	TRUE: A pause is executed. Deceleration stop is executed according to the velocity profile (iVelMode). FALSE: The pause is canceled.
	bEmergencyStop	BOOL	FALSE	TRUE: An emergency stop is executed. FALSE: The emergency stop is canceled.
	bAbort	BOOL	FALSE	TRUE: Execution of the FB is stopped.
	dwIpoTime	DWORD	0	MotionTask interval (µsec)
	iVelMode	SMC_INT_VELMODE	TRAPEZOID	Specifies a velocity profile.
	dJerkMax	LREAL	LREAL	Specifies the maximum value of jerk.

Scope	Name	Type	Default value	Description
				This parameter must be specified when QUADRATIC is selected for the velocity profile (iVelMode).
	bWaitAtNextStop(Note 1)	BOOL	BOOL	TRUE: A pause is executed in the table where the velocity between paths becomes zero. The conditions that cause the velocity between paths to become zero are set in bSingleStep or dAngleMode. FALSE: The pause is canceled.
	bSingleStep(Note 1)	BOOL	BOOL	TRUE: All connections between paths are established through deceleration stop.
Output	bCommandAborted	BOOL	FALSE	TRUE: An interruption is caused by another FB.
	bBusy	BOOL	-	TRUE: Execution of the FB is not completed.
	bDone	BOOL	FALSE	TRUE: Output is completed.
	bError	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorIDdx	SMC_ERROR	SMC_NO_ERROR	Error ID output during x-axis movement processing
	ErrorIDdy	SMC_ERROR	SMC_NO_ERROR	Error ID output during y-axis movement processing
	ErrorID	SMC_ERROR	SMC_NO_ERROR	Error ID output during interpolation control operation

(Note 1) When both bWaitAtNextStop and bSingleStep are set to TRUE, they may not work properly, so please do not use them together.

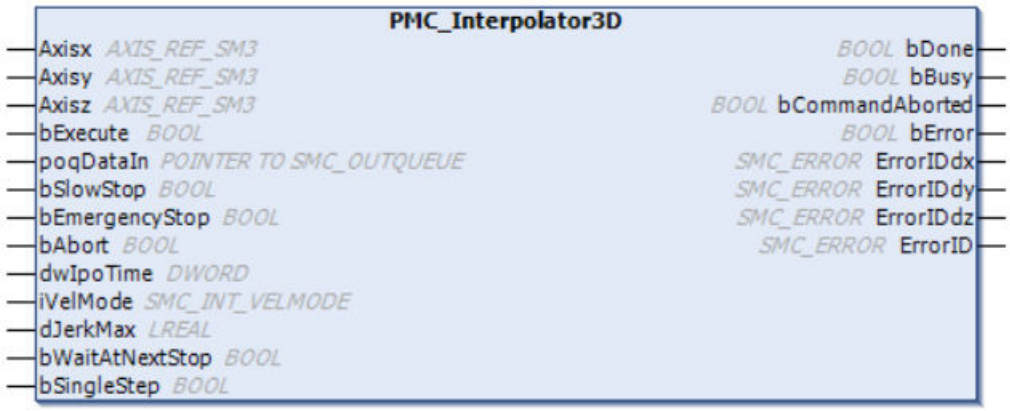
SMC_INT_VELMODE (Enumeration type)

Name	Value	Description
TRAPEZOID	0	Trapezoid
SIGMOID	1	Sin2
SIGMOID_LIMIT	2	Sin2 (limit)
QUADRATIC	3	Quadratic
QUADRATIC_SMOOTH	4	Quadratic (smooth)

7.1.2 PMC_Interpolator3D (3-axis Interpolation Control)

This function block (FB) performs 3-axis interpolation control according to the specified CNC table.

Icon



Parameter

Scope	Name	Type	Default value	Description
I/O	Axisx	AXIS_REF_SM3	-	Specifies the x-axis.
	Axisy	AXIS_REF_SM3	-	Specifies the y-axis.
	Axisz	AXIS_REF_SM3	-	Specifies the z-axis.
Input	bExecute	BOOL	FALSE	Starts execution at the rising edge.
	poqDataIn	POINTER TO SMC_OUTQUEUE	-	Specifies a pointer to the CNC table.
	bSlowStop	BOOL	FALSE	TRUE: A pause is executed. Deceleration stop is executed according to the velocity profile (iVelMode). FALSE: The pause is canceled.
	bEmergencyStop	BOOL	FALSE	TRUE: An emergency stop is executed. FALSE: The emergency stop is canceled.
	bAbort	BOOL	FALSE	TRUE: Execution of the FB is stopped.
	dwIpoTime	DWORD	0	MotionTask interval (µsec)
	iVelMode	SMC_INT_VELMODE	TRAPEZOID	Specifies a velocity profile.
	dJerkMax	LREAL	LREAL	Specifies the maximum value of jerk. This parameter must be specified when QUADRATIC is selected for the velocity profile (iVelMode).
	bWaitAtNextStop(Note 1)	BOOL	BOOL	TRUE: A pause is executed in the table where the velocity between paths becomes zero. The conditions that cause the velocity between paths to become

Scope	Name	Type	Default value	Description
				zero are set in bSingleStep or dAngleMode. FALSE: The pause is canceled.
	bSingleStep(Note 1)	BOOL	BOOL	TRUE: All connections between paths are established through deceleration stop.
Output	bCommandAborted	BOOL	FALSE	TRUE: An interruption is caused by another FB.
	bBusy	BOOL	-	TRUE: Execution of the FB is not completed.
	bDone	BOOL	FALSE	TRUE: Output is completed.
	bError	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorIDdx	X-axis error ID	SMC_NO_ERROR	Error ID output during x-axis movement processing
	ErrorIDdy	Y-axis error ID	SMC_NO_ERROR	Error ID output during y-axis movement processing
	ErrorIDdz	Z-axis error ID	SMC_NO_ERROR	Error ID output during z-axis movement processing
	ErrorID	SMC_ERROR	SMC_NO_ERROR	Error ID output during interpolation control operation

(Note 1) When both bWaitAtNextStop and bSingleStep are set to TRUE, they may not work properly, so please do not use them together.

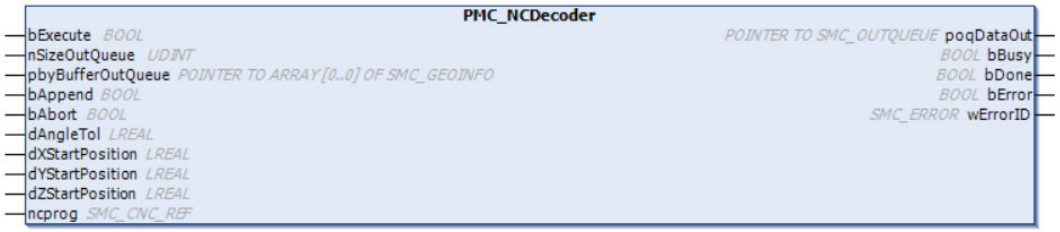
SMC_INT_VELMODE (Enumeration type)

Name	Value	Description
TRAPEZOID	0	Trapezoid
SIGMOID	1	Sin2
SIGMOID_LIMIT	2	Sin2 (limit)
QUADRATIC	3	Quadratic
QUADRATIC_SMOOTH	4	Quadratic (smooth)

7.1.3 PMC_NCDecoder (CNC Table Conversion)

This function block (FB) decodes the specified SMC_CNC_REF value to SMC_OUTQUEUE.

Icon



Parameter

Scope	Name	Type	Default value	Description
I/O	ncprogIn	SMC_CNC_REF	-	Specifies the SMC_CNC_REF value to be decoded.
Input	bExecute	BOOL	FALSE	Starts execution at the rising edge.
	nSizeOutQueue	UDINT	-	Specifies a buffer size. We recommend that a buffer be created and the sizeof operator be specified as shown in the following example. ExampleBuf: ARRAY [0..50] OF SMC_GEOINFO; nSizeOutQueue:=sizeof(ExampleBuf)
	pbyBufferOutQueue	POINTER TO ARRAY [0..0] OF SMC_GEOINFO	-	Specifies the memory space for SMC_OUTUEUE. We recommend that array SMC_GEOINFO be defined and an address be specified as shown in the following example. ExampleBuf: ARRAY [0..50] OF SMC_GEOINFO; (Buffer that can store 50 path elements) pbyBufferOutQueue:=ADR(ExampleBuf)
	dXstartPosition	LREAL	0	Specifies the position of the x-axis at the start of movement(Note 1).
	dYstartPosition	LREAL	0	Specifies the position of the y-axis at the start of movement(Note 1).
	dZstartPosition	LREAL	0	Specifies the position of the z-axis at the start of movement(Note 1).
	bAppend	BOOL	FALSE	TRUE: Decoded data of ncprogIn is appended to the end of poqDataOut without resetting the poqDataOut data within the FB at the rising edge as specified by bExecute.
	bAbort	BOOL	FALSE	TRUE: Execution of the FB is stopped.

Scope	Name	Type	Default value	Description
Output	poqDataOut	POINTER TO SMC_OUTQUEUE	-	Pointer to SMC_OUTQUEUE which manages decoded SMC_GEOINFO objects
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	Done	BOOL	FALSE	TRUE: Output is completed.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	Error ID output

(Note 1) We recommend that fSetPosition be entered. If the entered value and the actual position differ, there is a risk that the axis may move suddenly.

(MEMO)

8 Motion Control Function Blocks (CNC Control)

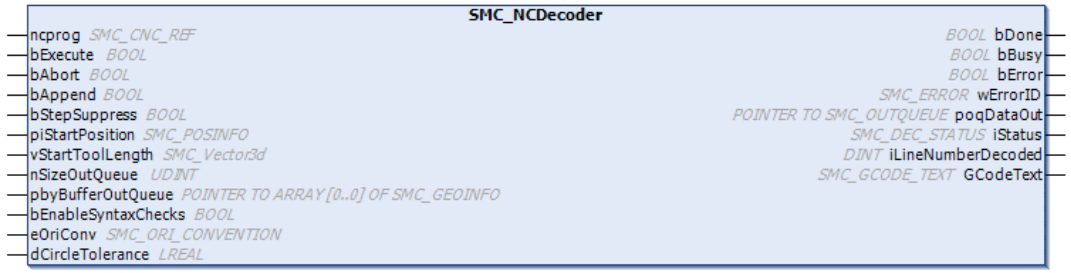
This section describes function blocks used to perform control using the CNC program.

8.1 G code decoding

8.1.1 SMC_NCDecoder (CNC Table Conversion)

This function block (FB) decodes a specified CNC program (SMC_CNC_REF) to a list of SMC_GEOINFO objects (SMC_OUTQUEUE). In each cycle, one line of the program is decoded. Execute the function block by MotionTask.

■ Icon



■ Parameter

Scope	Name	Type	Default value	Description
I/O	ncprog	SMC_CNC_REF	-	Specifies the SMC_CNC_REF value to be decoded.
Input	bExecute	BOOL	FALSE	Starts execution at the rising edge.
	bAbort	BOOL	FALSE	TRUE: Execution of the FB is stopped.
	bAppend	BOOL	FALSE	TRUE: At the rising edge as specified by bExecute, the poqDataOut data within the FB is not reset.(Note 1) Decoded data of ncprogIn is appended to the end of poqDataOut.
	bStepSuppress	BOOL	FALSE	TRUE: Lines of the CNC program starting with “/” are ignored.
	piStartPosition	SMC_POSINFO	-	Start position of the path(Note 2)
	vStartToolLength	SMC_Vector3d	dX=0, dY=0, dZ=0	Start tool length. Even if this parameter is set, nothing is reflected in operation. Do not use.
	nSizeOutQueue	UDINT	0	Specifies the size of the data buffer to which the list of SMC_GEOINFO structure objects will be written. This buffer must be able to hold at least five SMC_GEOINFO objects. If the size of the buffer is not satisfactory, no error occurs and the FB is not executed.

Scope	Name	Type	Default value	Description
				The buffer size may be predefined, but may be changed only during a reset. [Declaration example] ExampleBuf: ARRAY[0..49] OF SMC_GEOINFO; (An array of five or more elements is required) [Example of acquiring appropriate buffer size] nSizeOutQueue:=SIZEOF(ExampleBuf);
	pbyBufferOutQueue	POINTER TO ARRAY [0..0] OF SMC_GEOINFO	-	Specifies an address of the memory space for SMC_OUTUEUE. We recommend that array SMC_GEOINFO be defined and an address be specified as shown in the following example. [Declaration example] ExampleBuf: ARRAY[0..49] OF SMC_GEOINFO; (An array of five or more elements is required) [Writing example] pbyBufferOutQueue:=ADR(ExampleBuf);
	bEnableSyntaxChecks	BOOL	FALSE	TRUE: Detects invalid G-code and wrong CNC program, and stops with the occurrence of an error.
	eOriConv	SMC_ORI_CONVENTION	ADDAXES	A definition for specifying the order in which the coordinates A/B/C are rotated by coordinate system conversion G54/G55/G56
	dCircleTolerance	LREAL	0	Tolerance to determine whether the definition of a circle makes sense(Note 3)
Output	bDone	BOOL	FALSE	TRUE: Decode output is completed.
	bBusy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	bError	BOOL	FALSE	TRUE: An error has occurred within the FB.
	wErrorID	SMC_ERROR	SMC_NO_ERROR	Error ID output
	poqDataOut	POINTER TO SMC_OUTQUEUE	-	Pointer to the CNC table which manages decoded SMC_GEOINFO objects
	iStatus	SMC_DEC_STATUS	WAIT_PROG	Current status
	iLineNumberDecoded	DINT	0	The 0-based line number of the CNC file that is completely decoded
	GCodeText	SMC_GCODE_TEXT	-	The G-Code text that is currently decoded

(Note 1) To decode the CNC table including circular interpolation written by center-point technique with bAppend = TRUE, explicitly specify G98/G99 in the CNC table.

(Note 2) We recommend that fSetPosition be entered. If the entered value and the current value greatly differ, there is a risk that the axis may move suddenly.

(Note 3) Determined according to the following rules:

Definition via target-position and radius: If the distance between start- and end-position is greater than 2 * the radius + MAX(dCircleTolerance, 1e-06), the circle will be converted into a line.

Definition via target- and center-position: Let x be the maximum of the distance between start- and center-position and the distance between target- and center-position. If those distances differ by more than MAX(dCircleTolerance, 0.1 * x), the circle will be converted into a line.

SMC_POSINFO (Structure)

This is a structure that describes the positions of coordinate axes including additional axes for a particular position point.

Information on the path written in G code is output as position information for control at every cycle from the SMC_Interpolator.

Information about the output is written by this structure.

Name	Type	Default value	Description
iFrameNo	INT	0	Frame number (Additional information not relevant for the SoftMotion modules may be stored by the user.)
wAuxData	WORD	7	Axes to be calculated by the Interpolator: TRUE = Enabled, FALSE = Disabled bit0 = X axis, bit1 = Y axis, bit2 = Z axis, bit3 and subsequent bits are not used.
wSProfile	WORD	0	Not used
dX	LREAL	0	X-position in coordinate system
dY	LREAL	0	Y-position in coordinate system
dZ	LREAL	0	Z-position in coordinate system
dA	LREAL	0	Not used
dB	LREAL	0	Not used
dC	LREAL	0	Not used
dA1	LREAL	0	Not used
dA2	LREAL	0	Not used
dA3	LREAL	0	Not used
dA4	LREAL	0	Not used
dA5	LREAL	0	Not used
dA6	LREAL	0	Not used

■ Example

- For 2-axis interpolation control, wAuxData = 10#3 and values are set in dX: target position X for next cycle and dY: target position Y for next cycle. The other parameters are not used.
- For 3-axis interpolation control, wAuxData = 10#7 and values are set in dX: target position X for next cycle, dY: target position Y for next cycle, and dZ: target position Z for next cycle. The other parameters are not used.

SMC_ORI_CONVENTION (Enumeration type)

Input values need to be specified when coordinate conversion (G54, G55, G56) is executed.

While parallel translation (X, Y, Z) of the coordinate system is executed with any input value, rotation of the coordinate system requires an input value that sets the derided order in rotation to be specified.

Name	Type	Value	Description
ADDAXES	INT	0	Rotation of the coordinate system is not executed (default value).
ZYZ	INT	1	The coordinate system rotates around the Z axis > The coordinate system rotates around the Y axis > The coordinate system rotates around the Z axis
ZYX	INT	2	The coordinate system rotates around the Z axis > The coordinate system rotates around the Y axis > The coordinate system rotates around the X axis
XYZ	INT	3	The coordinate system rotates around the X axis > The coordinate system rotates around the Y axis > The coordinate system rotates around the Z axis

SMC_DEC_STATUS (Enumeration type)

Name	Type	Value	Description
WAIT_PROG	INT	0	Waiting program
READ_WORD	INT	1	Program decoding in progress
PROG_READ	INT	2	Program decoding completed

SMC_GCODE_TEXT (Structure)

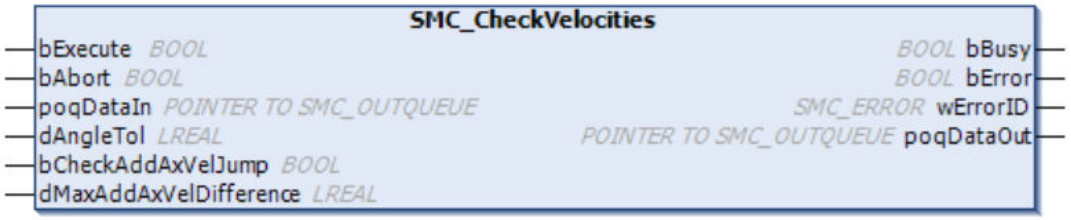
Name	Type	Default value	Description
str	STRING(80)	''	Outputs the G-Code text that is currently decoded
iLineNumber	DINT	0	Line number
bNewLine	BOOL	FALSE	TRUE: A new line has been decoded.
bClearList	BOOL	FALSE	TRUE: When the NCDecoder has been started from new, a buffer that may store the last lines needs to be emptied.

8.2 Pre-processing after decoding

8.2.1 SMC_CheckVelocities (Check Angle between Paths)

This function block (FB) is used to check an angle between paths and perform P-point control (without deceleration stop between paths) or C-point control (with deceleration stop between paths) according to the formed angle. If the SMC_OUTQUEUE has not been created by the editor, but by the program (e.g. SMC_NCDecoder), this FB has to be called straight before each call to the SMC_Interpolator. Execute the function block by MotionTask.

■ **Icon**



■ **Parameter**

Scope	Name	Type	Default value	Description
Input	bExecute	BOOL	FALSE	Starts execution of the FB at the rising edge.
	bAbort	BOOL	FALSE	TRUE: Execution of the FB is stopped.(Note 1) Do not use this if you want to stop movement in midstream.
	poqDataIn	POINTER TO SMC_OUTQUEUE	-	A pointer to the CNC table
	dAngleTol	LREAL	0.001	Tolerance angle up to which P-point control is performed
	bCheckAddAxVelJump	BOOL	FALSE	TRUE: Additional axes velocities are checked. Even if this parameter is set, nothing is reflected in operation. Do not use.
	dMaxAddAxVelDifference	LREAL	0	Maximum allowed velocity difference (u/s) Even if this parameter is set, nothing is reflected in operation. Do not use.
Output	bBusy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	bError	BOOL	FALSE	TRUE: An error has occurred within the FB.

Scope	Name	Type	Default value	Description
	wErrorID	SMC_ERROR	SMC_NO_ERROR	Error ID output
	poqDataOut	POINTER TO SMC_OUTQUEUE	-	A pointer to SMC_OUTQUEUE that has checked the angle between the paths(Note 2)

(Note 1) The abort function operates only before the completion of SMC_NCDecoder or when the G code “G75 (timing synchronization with SMC_Interpolator)” is used.

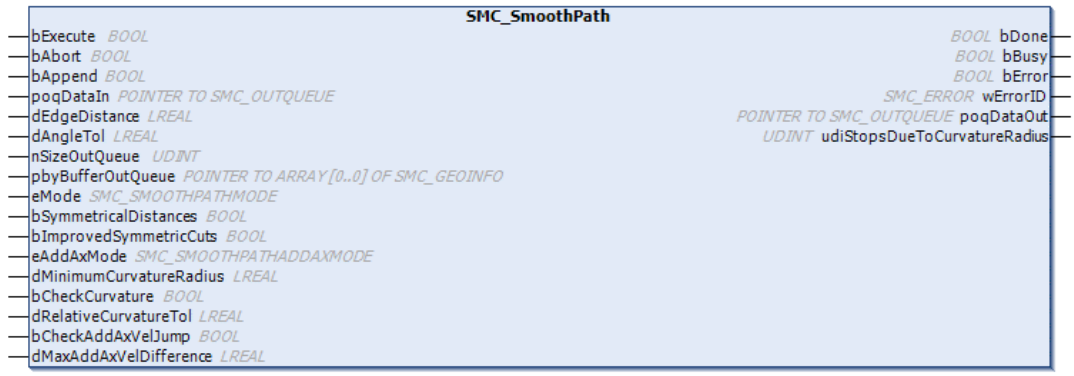
If you want to stop axial movement in midstream, do not use bAbort but use the argument described in “7.2.3 SMC_Interpolator”.

(Note 2) For the poqDataIn value, do not specify a pointer to values other than the CNC table. Otherwise, SMC_CheckVelocities and subsequent processes will not be executed.

8.2.2 SMC_SmoothPath (path smoothing)

This function block can smooth bends in the path of the specified CNC table. G51 and G50 in the G-code are used to perform smoothing.Unlike SMC_RoundPath, the path around the bend is also subject to smoothing. This FB must be run before running SMC_Interpolator. Execute the function block by MotionTask.

■ Icon



■ Parameter

Scope	Name	Type	Default value	Description
Input	bExecute	BOOL	FALSE	Starts execution at the rising edge.
	bAbort	BOOL	FALSE	TRUE: Execution of the FB is stopped.
	bAppend	BOOL	FALSE	TRUE: At the rising edge as specified by bExecute, the poqDataOut data within the FB is not reset. Decoded data of ncprogIn is appended to the end of poqDataOut.

Scope	Name	Type	Default value	Description
				(Note 1)
	poqDataIn	POINTER TO SMC_OUTQUEUE	-	A pointer to the CNC table(Note 2)
	dEdgeDistance	LREAL	0	Set the radius of curvature of the smoothing process to be added to parameter D of G-code G51.
	dAngleTol	LREAL	'0.001	Set the tolerance for the path-to-path angle at which smoothing is not performed
	nSizeOutQueue	UDINT	0	Specifies the size of the data buffer to which the list of SMC_GEOINFO structure objects will be written. This buffer must be able to hold at least five SMC_GEOINFO objects. If the size of the buffer is not satisfactory, no error occurs and the FB is not executed. The buffer size may be predefined, but may be changed only during a reset. [Declaration example] ExampleBuf: ARRAY[0..49] OF SMC_GEOINFO; (An array of five or more elements is required) [Example of acquiring appropriate buffer size] nSizeOutQueue:=SIZEOF(ExampleBuf);
	pbyBufferOutQueue	POINTER TO ARRAY [0..0] OF SMC_GEOINFO	-	Refers to the first byte of memory space allocated to the SMC_OUTQUEUE structure, which must be at least as large as defined in nSizeOutQueue (Note 2) Size only be changed during predefinition or resetting [Declaration example] ExampleBuf: ARRAY[0..49] OF SMC_GEOINFO; (An array of five or more elements is required) [Writing example] pbyBufferOutQueue:=ADR(ExampleBuf);
	eMode	SMC_SMOOTHPATHMODE	SP_SPLINE3	Element type applied to path smoothing
	bSymmetricalDistances	BOOL	TRUE	Half the length of the short side of the two sides that form the angle and the smoothed radius of curvature D are compared, and the smaller value is taken as D'. TRUE: The smoothed radius of curvature is set to the value of D' FALSE: The value set in D is used

Scope	Name	Type	Default value	Description
	bImprovedSymmetricCuts	BOOL	FALSE	The setting is reflected when bSymmetricalDistances = TRUE. TRUE:For comparison of the radius of curvature of bSymmetrical Distances at the second and subsequent turns, the radius of curvature used for the judgment at the first corner is applied.
	eAddAxMode	SMC_SMOOTHPATHADDAXMODE	SPAA_LATE	It does not affect the operation even if set. Do not use.
	dMinimumCurvatureRadius	LREAL	0	If the spline inserted in the smoothing path contains a position in which the radius of curvature is less than this parameter, it is not smoothed and the original path bend is used
	bCheckCurvature	BOOL	FALSE	TRUE: Check whether the curvature of adjacent elements is equal. If not equal, the path is smoothed.
	dRelativeCurvatureTol	LREAL	0.001	It does not affect the operation even if set. Do not use.
	bCheckAddAxVelJump	BOOL	FALSE	It does not affect the operation even if set. Do not use.
	dMaxAddAxVelDifference	LREAL		It does not affect the operation even if set. Do not use.
Output	bDone	BOOL	FALSE	TRUE：Completion of smoothing of input data
	bBusy	BOOL	FALSE	TRUE： Execution of the FB is not completed.
	bError	BOOL	FALSE	TRUE： An error has occurred within the FB.
	wErrorID	SMC_ERROR	SMC_NO_ERROR	Error ID output
	poqDataOut	POINTER TO SMC_OUTQUEUE	-	Pointer to the SMC_OUTQUEUE that executed the smoothing process
	udiStopsDueToCurvatureRadius	UDINT	0	Number of bends that could not be smoothed due to the dMinimumCurvatureRadius setting

(Note 1) When using the bAppend function, set it to TRUE at the same time as bAppend of SMC_NCDecoder.

(Note 2) If the input variable is not set correctly, the operation is not executed when there is no error and when bBusy = TRUE.

(Note 3) For details, refer to “G50, G51, G52”

SMC_SMOOTHPATHMODE (Enumeration type)

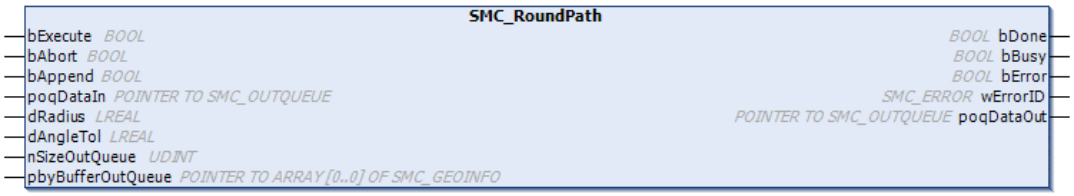
Name	Description
SP_SPLINE3	A 3rd order spline with different tangent lengths is inserted to define the spline. The length is dependent on the length of the adjacent object.

Name	Description
SP_SPLINE5	A 5th order spline is inserted.
SP_SPLINE3_CV	A 3rd order spline with different tangent lengths is inserted to define the spline. The length is dependent on the length of the portion of the cut adjacent object. For two adjacent line objects, SP_SPLINE3_CV stays inside the convex groove of the original path.
SP_SPLINE5_CV	A 5th order spline with different tangent lengths is inserted to define the spline. The length is dependent on the length of the portion of the cut adjacent object.
SP_SPLINE5_MIN_CURVATURE	Multiplication with the 5th order polynomial of the minimum curvature.

8.2.3 SMC_RoundPath (Arc correction between paths)

This function block can correct between paths in the specified CNC table with an arc. To perform smoothing, use G52 and G50 in Gcode. Unlike SMC_SmoothPath, arc correction is applied only between the paths in the specified section. This FB must be run before running SMC_Interpolator. Execute the function block by MotionTask.

■ Icon



■ Parameter

Scope	Name	Type	Default value	Description
Input	bExecute	BOOL	FALSE	Starts execution at the rising edge.
	bAbort	BOOL	FALSE	TRUE: Execution of the FB is stopped.
	bAppend	BOOL	FALSE	TRUE: At the rising edge as specified by bExecute, the poqDataOut data within the FB is not reset. Decoded data of ncprogIn is appended to the end of poqDataOut. (Note 1)
	poqDataIn	POINTER TO SMC_OUTQUEUE	-	A pointer to the CNC table(Note 2)
	dRadius	LREAL	0	Set the radius of curvature of arc correction to be added to parameter D of G code G52.(Note 3)

Scope	Name	Type	Default value	Description
	dAngleTol	LREAL	'0.001	Sets the tolerance for the angle between paths where arc correction is not performed.
	nSizeOutQueue	UDINT	0	Specifies the size of the data buffer to which the list of SMC_GEOINFO structure objects will be written. This buffer must be able to hold at least five SMC_GEOINFO objects. If the size of the buffer is not satisfactory, no error occurs and the FB is not executed. The buffer size may be predefined, but may be changed only during a reset. [Declaration example] ExampleBuf: ARRAY[0..49] OF SMC_GEOINFO; (An array of five or more elements is required) [Example of acquiring appropriate buffer size] nSizeOutQueue:=SIZEOF(ExampleBuf);
	pbyBufferOutQueue	POINTER TO ARRAY [0..0] OF SMC_GEOINFO	-	Refers to the first byte of memory space allocated to the SMC_OUTQUEUE structure, which must be at least as large as defined in nSizeOutQueue.(Note 2) Size only be changed during predefinition or resetting [Declaration example] ExampleBuf: ARRAY[0..49] OF SMC_GEOINFO; (An array of five or more elements is required) [Writing example] pbyBufferOutQueue:=ADR(ExampleBuf);
Output	bDone	BOOL	FALSE	TRUE：Completion of smoothing of input data
	bBusy	BOOL	FALSE	TRUE： Execution of the FB is not completed.
	bError	BOOL	FALSE	TRUE： An error has occurred within the FB.
	wErrorID	SMC_ERROR	SMC_NO_ERROR	Error ID output
	poqDataOut	POINTER TO SMC_OUTQUEUE	-	Pointer to the SMC_GEOINFO object that performed arc correction

(Note 1) When using the bAppend function, set it to TRUE at the same time as bAppend of SMC_NCDecoder.

(Note 2) If you do not set the b input variable correctly, bBusy = TRUE and no error will occur.

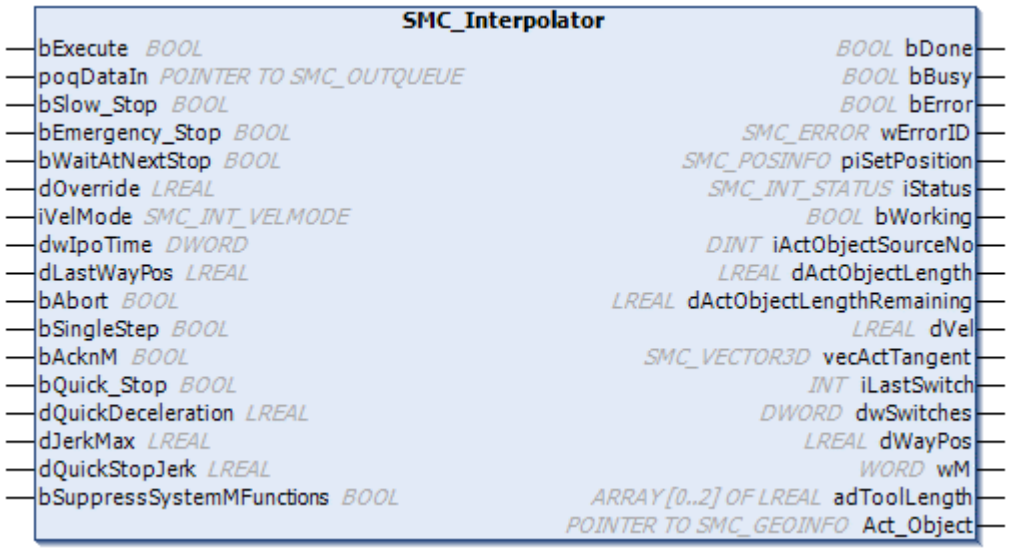
(Note 3) If the radius specification is 0, subsequent arc correction will not be performed.

8.3 Control calculation

8.3.1 SMC_Interpolator (CNC Control Operation)

This is a function block (FB) that converts a continuous path described by SMC_GEOINFO objects into discrete path position points taking into account a defined velocity profile and time pattern. Execute the function block by MotionTask.

■ **Icon**



■ **Parameter**

Scope	Name	Type	Default value	Description
Input	bExecute	BOOL	FALSE	Starts execution at the rising edge.
	poqDataIn	POINTER TO SMC_OUTQUEUE	-	Specifies a pointer to the CNC table.
	bSlow_Stop	BOOL	FALSE	TRUE: Executes deceleration stop according to the velocity profile (iVelMode). FALSE: The pause is canceled.
	bEmergency_Stop	BOOL	FALSE	TRUE: Causes an emergency stop, so that piSetPosition will be retained at the current value and the velocity will be set to 0.(Note 1) FALSE: The emergency stop is canceled.
	bWaitAtNextStop	BOOL	FALSE	TRUE: Executes a pause in the table where the velocity between paths becomes zero. FALSE: The pause is canceled.

Scope	Name	Type	Default value	Description
	dOverride	LREAL	1	This variable can be used to handle the override. The velocity gets scaled by dOverride. (0.01-)(Note 2) The modified override will only be applied if axis acceleration or deceleration is not in progress.
	iVelMode	SMC_INT_VELMODE	TRAPEZOID	Specifies a velocity profile.(Note 3)
	dwIpoTime	DWORD	0	MotionTask interval (μsec)
	dLastWayPos	LREAL	0	The total length of the path generated by the CNC control operation can be measured. To use this, dLastWayPos needs to be connected to dWayPos.
	bAbort	BOOL	FALSE	TRUE: Execution of the FB is stopped.
	bSingleStep	BOOL	FALSE	TRUE: All connections between paths are established through deceleration stop.
	bAcknM	BOOL	FALSE	TRUE：Output wM is cleared and processing resumes from the paused state.
	bQuick_Stop	BOOL	FALSE	TRUE: Reduces the velocity of the object to zero and stops it. The velocity is reduced according to the velocity profile specified in iVelMode and the deceleration given by the maximum of the values specified in dQuickDeceleration and programmed in the path. If a quadratic velocity profile is used, the jerk is limited by max(dJerkMax, dQuickStopJerk). FALSE: Cancels deceleration stop.
	dQuickDeceleration	LREAL	0	Specifies a deceleration value used for bQuick_Stop.
	dJerkMax	LREAL	0	Magnitude of the maximum allowed jerk used for quadratic velocity profiles Must be positive and cannot be changed while performing a CNC control operation
	dQuickStopJerk	LREAL	0	Specifies the jerk used for bQuick_Stop.
	bSuppressSystemMFunctions	BOOL	FALSE	TRUE: The output wM is not set for internal M-functions created by G75 or G4 commands. Even if this parameter is set, nothing is reflected in operation. Do not use.
Output	bDone	BOOL	FALSE	TRUE: Output is completed.

Scope	Name	Type	Default value	Description
	bBusy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	bError	BOOL	FALSE	TRUE: An error has occurred within the FB.
	wErrorID	SMC_ERROR	SMC_NO_ERROR	Error ID output during CNC control operation
	piSetPosition	SMC_POSINFO	-	The target coordinates of the next position set by CNC control operation (Cartesian coordinate system)
	iStatus	SMC_INT_STATUS	IPO_INIT	The current status of the FB
	bWorking	BOOL	FALSE	TRUE: The process is underway. We recommend that this output be connected to input bEnable of SMC_ControlAxisByPos.
	iActObjectSourceNo	DINT	-1	Outputs a value of iSourceLine_No of active SMC_GEOINFO object of poqDataIn-queue. When bWorking = FALSE, the value is set to -1.
	dActObjectLength	LREAL	0	The length of the current object. Valid if bWorking = TRUE. A correct value may not be output. Do not use.
	dActObjectLengthRemaining	LREAL	0	The remaning length of the current object. Valid if bWorking = TRUE. A correct value may not be output. Do not use.
	dVel	LREAL	0	Current path velocity
	vecActTangent	SMC_VECTOR3D	-	Current path tangent, a unit vector
	iLastSwitch	INT	0	The number of the last switch H passed A correct value may not be output. Do not use.
	dwSwitches	DWORD	0	The current switch status of H switches 1 to 32, in bit notation A correct value may not be output. Do not use.
	dWayPos	LREAL	0	Refer to dLastWayPos.
	wM	WORD	0	Number of M function where CNC control operation is paused.
	adToolLength	ARRAY [0..2] OF LREAL	-	Parameters for tool length compensation A correct value may not be output. Do not use.
	Act_Object	POINTER TO SMC_GEOINFO	0	Pointer to the path element currently in progress

Scope	Name	Type	Default value	Description
				A correct value may not be output. Do not use.

(Note 1) Make sure that bEmergency_Stop is connected to SMC_ControlAxisByPos.bStopIpo.

(Note 2) Set dOverride to a numerical value greater than 0.01.
If set to a smaller value, no axis movement starts and no error occurs.

(Note 3) Axis velocity ramp type settings do not apply to CNC control using SMC_Interpolator.
Specify the velocity profile in iVelMode.

SMC_INT_VELMODE (Enumeration type)

Name	Value	Description
TRAPEZOID	0	Trapezoid
SIGMOID	1	Sin2
SIGMOID_LIMIT	2	Sin2 (limit)
QUADRATIC	3	Quadratic
QUADRATIC_SMOOTH	4	Quadratic (smooth)

SMC_INT_STATUS (Enumeration type)

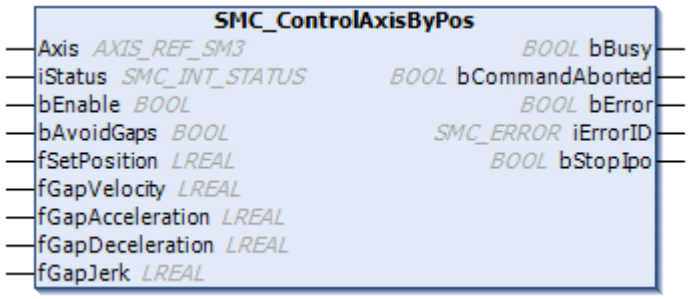
Name	Value	Description
IPO_UNKNOWN	0	Internal state that may not occur after a complete pass of the SMC_Interpolator
IPO_INIT	1	Initialization state, movement not started yet
IPO_ACCEL	2	Currently accelerating
IPO_CONSTANT	3	Movement ongoing with constant velocity
IPO_DECEL	4	Currently decelerating
IPO_FINISHED	5	CNC control is done. From then on, SMC_GEOINFO object input in poqDataIn is no processed.
IPO_WAIT	6	Currently waiting, for example, by a stop input
IPO_INCREASING_ACCEL	7	Currently increasing the acceleration
IPO_DECREASING_ACCEL	8	Currently decreasing the acceleration
IPO_INCREASING_DECEL	9	Currently increasing the deceleration
IPO_DECREASING_DECEL	10	Currently decreasing the deceleration

8.4 Control command & kinematics conversion

8.4.1 SMC_ControlAxisByPos (Axis Position Control)

This function block (FB) writes the input variable fSetPosition to the drive structure (AXIS_REF_SM3) and controls axis movement. Performs position control while monitoring whether the axis speed exceeds the dynamic limit setting. Execute the function block by MotionTask.

■ **Icon**



■ **Parameter**

The FB monitors if the axis velocity exceeds the dynamic limit setting. If the velocity exceeds the limit setting, the FB outputs bStopIpo = TRUE to decelerate it. The FB adapts the axis to the position at which the velocity exceeded the limit setting. When adaptation to the position is completed, the FB outputs bStopIpo = FALSE and then returns to its original control.

Scope	Name	Type	Default value	Description
I/O	Axis	AXIS_REF_SM3	-	Reference to the axis
Input	iStatus	SMC_INT_STATUS	IPO_INIT	Substitute iStatus of SMC_Interpolator
	bEnable	BOOL	FALSE	TRUE: The FB can be executed.
	bAvoidGaps	BOOL	TRUE	TRUE：Enable monitoring of dynamic limits The axis is moved to the position according to the values set in fGapVelocity, fGapAcceleration, and fGapDeceleration.
	fSetPosition	LREAL	-	Set position of the axis in (u).
	fGapVelocity	LREAL	1	Follow-up speed when the dynamic limit is exceeded (u/s) Set the value within the dynamic limit.(Note 1)
	fGapAcceleration	LREAL	1	Follow-up acceleration when the dynamic limit is exceeded (u/s^2) Set the value within the dynamic limit.(Note 1)

Scope	Name	Type	Default value	Description
	fGapDeceleration	LREAL	1E+15	Follow-up deceleration when the dynamic limit is exceeded (u/s^2) Set the value within the dynamic limit.(Note 1) Deceleration is also used if bAvoidGaps = FALSE, for stopping after the change of bEnable from TRUE to FALSE.
	fGapJerk	LREAL	1E+16	Follow-up jerk when dynamic limit is exceeded (u/s^3) Enabled when the axis velocity ramp type is set to a value other than “Trapezoid”
Output	bBusy	BOOL	FALSE	TRUE: FB operation is in progress.
	bCommandAborted	BOOL	FALSE	TRUE: An interruption is caused by another FB.
	bError	BOOL	FALSE	TRUE: An error has occurred within the FB.
	iErrorID	SMC_ERROR	SMC_NO_ERROR	Error ID output
	bStopIpo	BOOL	FALSE	TRUE：The speed of the axis exceeds the set value of the dynamic limit, and follow-up control is being executed.(Note 2)

(Note 1) Do not set fGapVelocity, fGapAcceleration, and fGapDeceleration to values that exceed the axis dynamic limit settings.

(Note 2) Make sure that bStopIpo is connected to SMC_Interpolator.bEmergency_Stop.

8.4.2 SMC_TRAFO_Polar (Conversion from Two-dimensional (X, Y) Coordinates to Polar Coordinates)

This function block (FB) converts two-dimensional (X, Y) coordinates into polar (R, φ) coordinates. The calculation of and conversion to R and φ are performed as follows.Please execute it as a motion task.

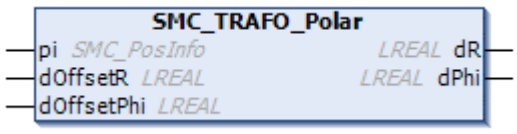
■ Conversion formula

R = (X^2 ＋ Y^2)^0.5 + offsetR　φ=Atan(Y / X) + offsetφ

When X = 0, φ = 90 deg (Y > 0) or -90 deg (Y≦ 0)

When Y =0, φ= 0 deg (X > 0) or -180 deg (X < 0)

■ Icon



■ Parameter

Scope	Name	Type	Default value	Description
Input	pi	SMC_PosInfo	-	Two-dimensional (X, Y) coordinates
	dOffsetR	LREAL	0	Offset for radial distance axis R
	dOffsetPhi	LREAL	0	Offset for angular direction axis φ in degree
Output	dR	LREAL	-	Position of radial distance axis R after conversion
	dPhi	LREAL	-	Position of angular direction axis φ in degree after conversion(Note 1)

(Note 1) An angle that forms φ = 180 deg is converted as an angle of φ = -180 deg.

8.4.3 SMC_TRAFOF_Polar (Conversion from Polar Coordinates to Two-dimensional (X, Y) Coordinates)

This function block (FB) converts polar (R, φ) coordinates into two-dimensional (X, Y) coordinates. The calculation of and conversion to X and Y are performed as follows.Please execute it as a motion task.

■ Conversion formula

X = (R + offsetR) * cos(φ + offsetφ) Y = (R + offsetR) * sin(φ + offsetφ)

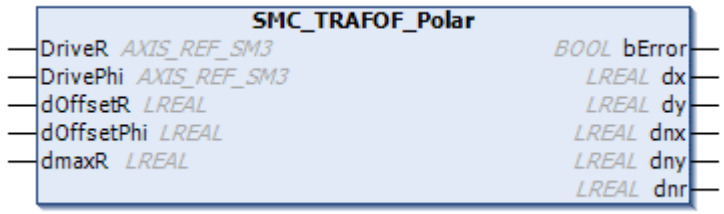
The norm after normalization with the maximum radius is calculated below.

nX = X / (dmaxR - offsetR)

nY = Y / (dmaxR - offsetR)

nR = (R - offsetR) / (dmaxR - offsetR)

■ Icon



■ Parameter

Scope	Name	Type	Default value	Description
I/O	DriveR	AXIS_REF_SM3	-	Reference to current position of radial distance axis R, fActPotion
	DrivePhi	AXIS_REF_SM3	-	Reference to current position of angular direction axis φ, fActPotion
Input	dOffsetR	LREAL	0	Offset for radial distance axis R

Scope	Name	Type	Default value	Description
	dOffsetPhi	LREAL	0	Offset for angular direction axis φ in degree
	dmaxR	LREAL	0	Maximum radius R (> 0) used in normalization(Note 1)
Output	bError	BOOL	FALSE	TRUE if conversion is not possible
	dx	LREAL	0	X-coordinate after conversion
	xy	LREAL	0	Y-coordinate after conversion
	dnx	LREAL	0	A position vector to the X-coordinate after conversion (after normalized by maximum radius)
	dny	LREAL	0	A position vector to the Y-coordinate after conversion (after normalized by maximum radius)
	dnr	LREAL	0	Norm after normalized by maximum radius

(Note 1) If the function block operates under the initial conditions of dmaxR = 0 and dOffsetR = 0, an exception error occurs.

8.4.4 SMC_TRAFO_Bipod_Arm (Bipod robot hand XY coordinates → conversion of each axis position)

It is a function block (FB) that converts the XY coordinates of the hand of the Bipod robot into the angle information of each axis motor. Please execute it as a motion task.

■ Conversion formula

dA ={-180-atan(Yo/Xa)} + acos {(L1^2 +Xa^2+Yo^2-L2^2)/(2*L1*(Xa^2+Yo^2)^1/2)} +dOffsetA

dB = atan(Yo/Xb) + acos {(L1^2 +Xb^2+Yo^2-L2^2)/(2*L1*(Xb^2+Yo^2)^1/2)} +dOffsetB

・When L2 > L1+D/2, Yo=Y-(L2^2-(L1+D/2)^2)^1/2 Other than that, Yo=Y

・Xa= X+1/2*D、Xb= X-1/2*D、L1=dArmLength1、L2=dArmLength2、D=dDistance

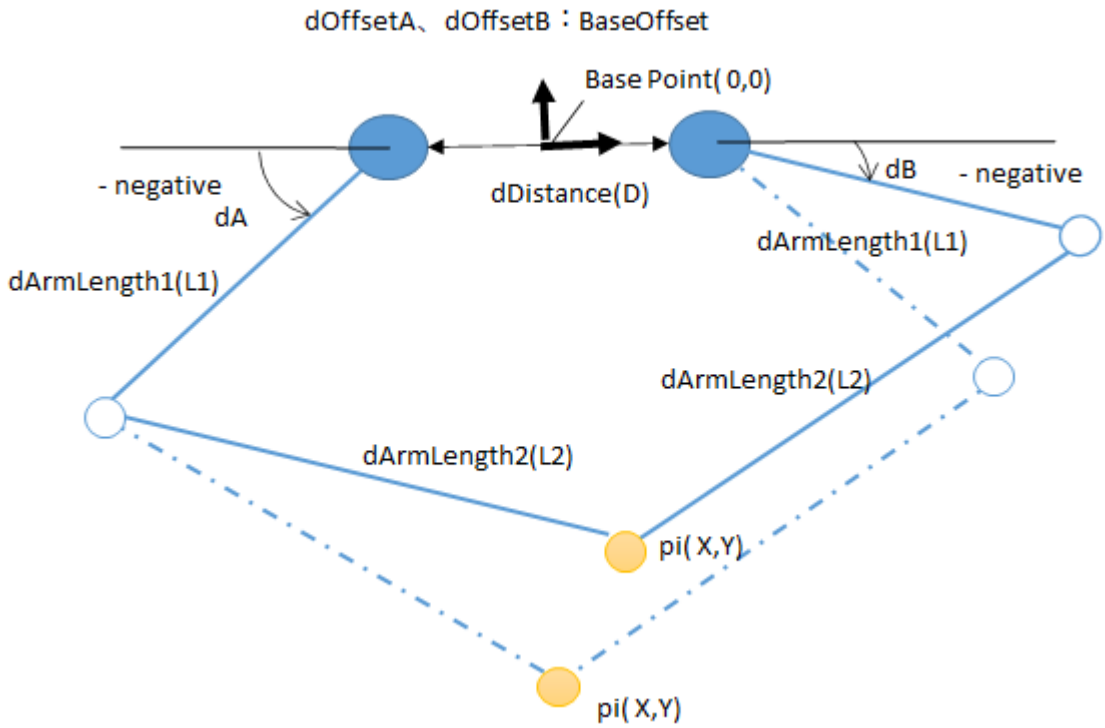
・When Xa = 0, atan (Yo / Xa) => -90, when Xb = 0, atan (Yo / Xb) => -90

・An error will occur if any of the following conditions are met:

1. Yo > 0
2. L1 ≦ 0 or L2 ≦ 0 or D < 0
3. (Xa^2+Yo^2)^1/2 > L1+L2 or (Xb^2+Yo^2)^1/2 > L1+L2
4. Posture that cannot be taken due to the mechanism

Note

- Please check the operating range according to the parameters in advance before use.
- Note that when L2> L1 + 1 / 2D, the origin position shifts in the Y direction by (L2 ^ 2- (L1 + D / 2) ^ 2) ^ 1/2 minutes.



■ Icon

■ Parameter

Scope	Name	Type	Default value	Description
入力	pi	SMC_PosInfo	-	XY2D coordinates of the hand
	dArmLength1(Note 1)	LREAL	0	Length from motor to joint(Note 2) dArmLength1>0
	dArmLength2(Note 1)	LREAL	0	Length from joint to hand(Note 2) dArmLength2>0
	dDistance(Note 1)	LREAL	0	Distance between two motors dDistance≧0
	dOffsetA(Note 1)	LREAL	0	Reference offset angle of left motor
	dOffsetB(Note 1)	LREAL	0	Reference offset angle of right motor

Scope	Name	Type	Default value	Description
出力	bError	BOOL	FALSE	TRUE: Argument or error in calculation process
	dA	LREAL	0	Angle of left motor to hand position
	dB	LREAL	0	Angle of right motor to hand position

(Note 1) Even if the values of dArmLength1, dArmLength2, dDistance, dOffsetA, and dOffsetB are changed after executing FB of the same instance, they are not reflected.

(Note 2) If the function block is operated with the initial value of dArmLength1 = 0 and dArmLength2 = 0, an exception error will occur.

(MEMO)

9 Motion Control Function Blocks (Motion Communication Control)

This section describes function blocks used to perform communication control.

9.1 RTEX

9.1.1 Types of Data To Be Handled by AMP Function Blocks

Item	Description	Related function blocks
AMP alarm	This is an AMP alarm that occurs in AMP operation.	RTEX_ClearAmpAlarm RTEX_ReadAmpAlarm RTEX_ReadAmpState
AMP warning	This is an AMP warning that occurs in AMP operation. This occurs before the AMP alarm. If the situation worsens, an AMP alarm occurs.	
Monitor data	This is monitor data (position deviation, load percentage, etc.) of the RTEX communication data.	RTEX_ReadAmpData
AMP parameter	This is configuration data of the AMP device itself.	RTEX_ReadAmpParameter RTEX_WriteAmpParameter
Multi-turn data	There are two types of data in the data read by the absolute encoder (23 bit/r): one type is single-turn data that indicates the position within one motor rotation and the other is multi-turn data that counts one for one turn.	RTEX_ClearAmpMultiTurnData
Deviation counter	This is a processing part in the AMP that receives move commands to the AMP. The motor moves according to the commands accumulated in the deviation counter. The commands used for the motor movement are deleted from the deviation counter. The amount of commands accumulated in the deviation counter is called the position deviation.	RTEX_ClearAmpPositionalDeviation
Limit switch	This data is collected to monitor the POT and NOT states of the AMP.	RTEX_ReadNot, RTEX_ReadPot

9.1.2 RTEX_ClearAmpAlarm (Clear Amplifier Alarm)

This is a function block (FB) that clears the AMP alarm. It deletes the alarm or warning that has occurred in the AMP.

■ Icon

■ Parameter

Scope	Name	Type	Initial	Description
Input / Output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	Execute	BOOL	FALSE	TRUE: Starts execution at the rising edge. FALSE: Stops processing.
Output	Done	BOOL	FALSE	TRUE：クリア完了
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	CommandAborted	BOOL	FALSE	TRUE：他の FB からの中断発生
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorId	SMC_ERROR	0	An error ID is output.
	Alarm	ALARM_CODE	-	A deleted alarm code is output.
	Warning	WARNING_CODE	-	A deleted warning code is output.

■ ALARM_CODE (Union)

Member	Type	Description
uiAlarmCode	UINT	Alarm code
tAlarmCodeMember	ALARM_WARNING_CODES	Main alarm code and sub alarm code

■ WARNING_CODE (Union)

Member	Type	Description
uiWarningCode	UINT	Warning code
tWarningCodeMember	ALARM_WARNING_CODES	Main warning code (warning number) and sub warning code (0)

■ ALARM_WARNING_CODES (Structure)

Member	Type	Description
byMainCode	BYTE	Main code

Member	Type	Description
bySubCode	BYTE	Sub code

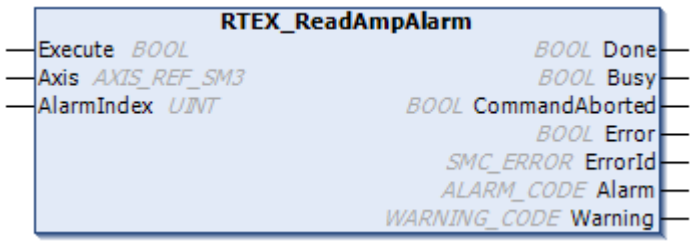
REFERENCE

13.2.1 RTEX Error ID
13.2.2 Alarm Codes
13.2.3 Warning Codes

9.1.3 RTEX_ReadAmpAlarm (Read Amplifier Alarm)

This is a function block (FB) that reads the AMP alarm. It reads the information of the alarm or warning that has occurred in the AMP.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	Execute	BOOL	FALSE	TRUE: Starts execution at the rising edge. FALSE: Stops processing.
	AlarmIndex	UINT	-	Specifies the history number (0 to 14). 0 is given for the latest history.
Output	Done	BOOL	FALSE	TRUE：読み出し完了
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	CommandAborted	BOOL	FALSE	TRUE：他の FB からの中断発生
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorId	SMC_ERROR	0	An error ID is output.
	Alarm	ALARM_CODE	-	A read alarm code is output.
	Warning	WARNING_CODE	-	A read warning code is output.

■ ALARM_CODE (Union)

Member	Type	Description
uiAlarmCode	UINT	Alarm code
tAlarmCodeMember	ALARM_WARNING_CODES	Main alarm code and sub alarm code

■ WARNING_CODE (Union)

Member	Type	Description
uiWarningCode	UINT	Warning code
tWarningCodeMember	ALARM_WARNING_CODES	Main warning code and sub warning code

■ ALARM_WARNING_CODES (Structure)

Member	Type	Description
byMainCode	BYTE	Main alarm number code
bySubCode	BYTE	Sub alarm number code

REFERENCE

13.2.1 RTEX Error ID
13.2.2 Alarm Codes
13.2.3 Warning Codes

9.1.4 RTEX_ReadAmpState (Amplifier Alarm Status)

This is a function block (FB) that reads the AMP alarm state. It outputs the information and state of the axis where the AMP alarm or warning has occurred.

■ Icon

■ Parameter

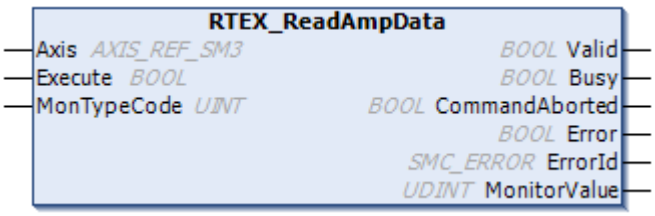
Scope	Name	Type	Initial	Description
Output	NumberOfSlaves	UDINT	-	The number of axes connected (1 to 16) is output.
	AlarmState	DWORD	-	The MAC-ID (0 to 15) where the AMP alarm has occurred is output.

Scope	Name	Type	Initial	Description
	WarningState	DWORD	-	The MAC-ID (0 to 15) where the AMP warning has occurred is output.

9.1.5 RTEX_ReadAmpData (Amplifier Monitor)

This is a function block (FB) that reads the monitor data of the AMP. It reads various monitor data of the AMP.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	Execute	BOOL	FALSE	TRUE: Starts execution at the rising edge. FALSE: Stops processing.
	MonTypeCode	UINT		Specifies the type code for the monitor command.
Output	Valid	BOOL	FALSE	TRUE：モニタ完了
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	CommandAborted	BOOL	FALSE	TRUE：他の FB からの中断発生
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorId	SMC_ERROR	0	An error ID is output.
	MonitorValue	UDINT	-	Read monitor command

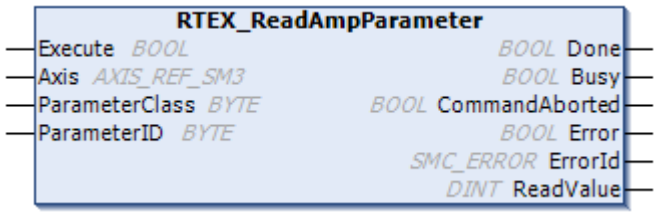
REFERENCE

13.2.1 RTEX Error ID
13.4 Monitor Commands

9.1.6 RTEX_ReadAmpParameter (Read Amplifier Parameter)

This is a function block (FB) that reads the AMP parameter.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	Execute	BOOL	FALSE	TRUE: Starts execution at the rising edge. FALSE: Stops processing.
	ParameterClass	BYTE	-	Specifies the AMP parameter classification.
	ParameterID	BYTE	-	Specifies the AMP parameter number.
Output	Done	BOOL	FALSE	TRUE：読み出し完了
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	CommandAborted	BOOL	FALSE	TRUE：他の FB からの中断発生
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorId	SMC_ERROR	0	An error ID is output.
	ReadValue	DINT	-	Read AMP parameter value

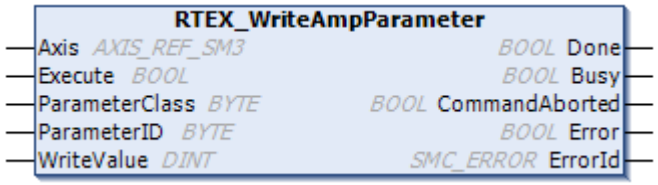
REFERENCE

13.2.1 RTEX Error ID
13.3 List of AMP Parameters

9.1.7 RTEX_WriteAmpParameter (Write Amplifier Parameter)

This is a function block (FB) that writes the AMP parameter.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	Execute	BOOL	FALSE	TRUE: Starts execution at the rising edge. FALSE: Stops processing.
	ParameterClass	BYTE	-	Specifies the AMP parameter classification.
	ParameterID	BYTE	-	Specifies the AMP parameter number.
	WriteValue	DINT	-	Value to be written in the AMP parameter
Output	Done	BOOL	FALSE	TRUE：書き込み完了
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	CommandAborted	BOOL	FALSE	TRUE：他の FB からの中断発生
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorId	SMC_ERROR	0	An error ID is output.

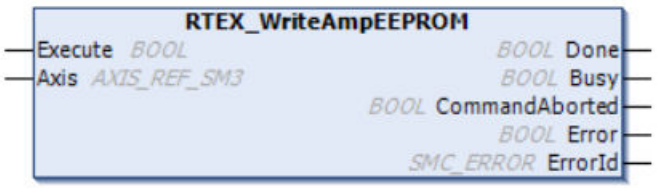
REFERENCE

13.2.1 RTEX Error ID
13.3 List of AMP Parameters

9.1.8 RTEX_WriteAmpEEPROM (Write Amplifier EEPROM)

This is a function block (FB) that writes the servo amplifier parameters to EEPROM.

■ **Icon**



■ **Parameter**

Scope	Name	Type	Default	Description
Input	Execute	BOOL	FALSE	TRUE: Starts execution at the rising edge. FALSE: Stops processing.
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Output	Done	BOOL	FALSE	TRUE: Writing is completed.
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	CommandAborted	BOOL	FALSE	TRUE: An interruption from other FB has occurred.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	Error ID output

9.1.9 RTEX_Reset (Reset RTEX)

Resets the entire RTEX network.

■ **Icon**

■ **Parameter**

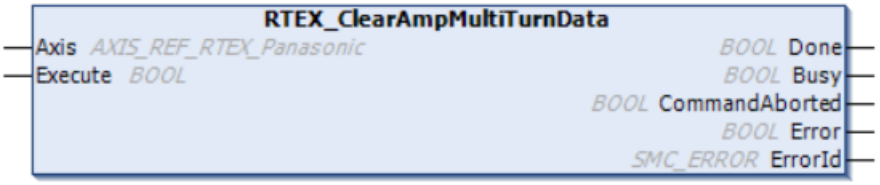
Scope	Name	Type	Default	Description
Input / output	Execute	BOOL	FALSE	TRUE: Starts execution at the rising edge.
Output	Done	BOOL	FALSE	TRUE: Reset done
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.

Scope	Name	Type	Default	Description
	CommandAborted	BOOL	FALSE	TRUE: An interruption from other FB has occurred.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	Error ID output

9.1.10 RTEX_ClearAmpMultiTurnData (Clear Amplifier Multi-turn Data)

This is a function block (FB) that clears the multi-turn data of the AMP.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_RTEX_Panasonic	-	Specifies the axis.
Input	Execute	BOOL	FALSE	TRUE: Starts execution at the rising edge. FALSE: Stops processing.
Output	CommandAborted	BOOL	FALSE	TRUE: An interruption from other FB has occurred.
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	Done	BOOL	FALSE	TRUE: Clearing is completed.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	An error ID is output.

The PMC_ClearAmpMultiTurnData function block outputs the following errors.

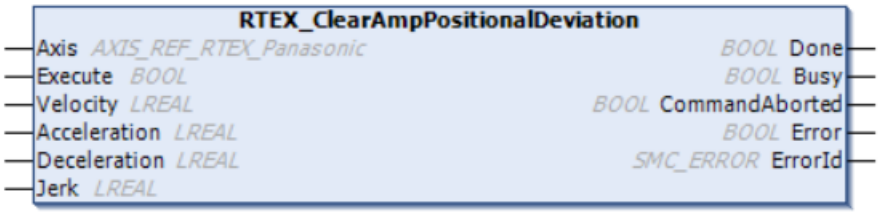
Error	Description
SMC_WRONG_CONTROLLER_MODE	Executed in a mode other than the position control mode. Change to SMC_position using SMC_SetControllerMode.
SMC_DI_HOMING_ERROR	The encoder used is an Incremental encoder.
SMC_AXIS_NOT_READY_FOR_MOTION	The axis is in a state where RTEX_ClearAmpMultiTurnData cannot be executed.

Error	Description
	It can be executed only when set to Disabled or Errorstop.
SMC_REGULATOR_OR_START_NOT_SET	The axis is in a servo ON state.
SMC_AXIS_REF_CHANGED_DURING_OPERATION	The Axis was changed during operation.

9.1.11 RTEX_ClearAmpPositionalDeviation (Clear Amplifier Deviation Counter)

This is a function block (FB) that clears the deviation counter of the AMP. It deletes the position deviation data in the deviation counter of the AMP.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_RTEX_Panasonic	-	Specifies the axis.
Input	Execute	BOOL	FALSE	Starts execution at the rising edge.
	Velocity(Note 1)	LREAL		Information required to execute MC_MoveAbsolute
	Acceleration(Note 1)	LREAL		Information required to execute MC_MoveAbsolute
	Deceleration(Note 1)	LREAL		Information required to execute MC_MoveAbsolute
	Jerk(Note 1)	LREAL		Information required to execute MC_MoveAbsolute
Output	CommandAborted	BOOL	FALSE	TRUE: An interruption from other FB has occurred.
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	Done	BOOL	FALSE	TRUE: Clearing is completed.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	An error ID is output.

(Note 1) This function block internally substitutes the command position with an actual position to call MC_MoveAbsolute and, therefore, requires parameters including Velocity, Acceleration, Deceleration, and Jerk.

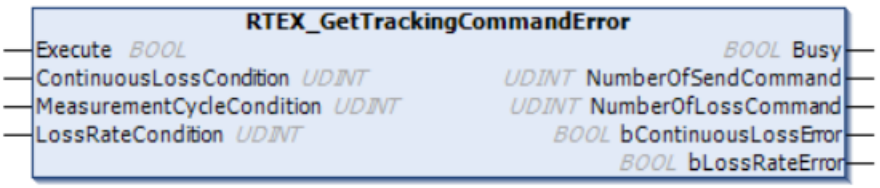
The RTEX_ClearAmpPositionalDeviation function block outputs the following errors.

Error	Description
SMC_WRONG_CONTROLLER_MODE	Executed in a mode other than the position control mode. Change to SMC_position using SMC_SetControllerMode.
SMC_AXIS_NOT_READY_FOR_MOTION	The axis is in a state where RTEX_ClearAmpPositionalDeviation cannot be executed. It can be executed only at the standstill state.
SMC_REGULATOR_OR_START_NOT_SET	The axis is in a servo ON state.
SMC_PP_WRONG_AXIS_TYPE	The axis is a virtual axis.

9.1.12 RTEX_GetTrackingCommandError (Read RTEX Command Send Statistics Information)

The RTEX periodically sends commands. With the GM1 specifications, when the MotionTask cycle time exceeds the control cycle, the command position for the servo amplifier is not updated for that cycle. (This is called a lost RTEX command.) This function block measures the number of sent RTEX commands and the number of lost RTEX commands. Using this function, you can check if the command position is updated normally for every cycle.

■ Icon



■ Parameter

Scope		Definition	Value	Description
Input	Execute	---		Execute = TRUE: Starts measurement when triggered. Execute = FALSE: Clears output.
	ContinuousLossCondtion	Continuous command loss condition	0: Disabled	If the command loss continuously occurs at ContinuousLossCondtion cycle, bContinuousLossError turns TRUE.
	MesurementCycleCondition	Command loss statistical measurement cycle condition	0: Disabled	If the command loss occurs as many times as specified in LossRateCondition during the MesurementCycleCondition period, bLossRateError turns TRUE.

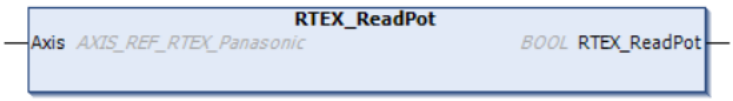
Scope		Definition	Value	Description
	LossRateCondition	Under measurement	0 to 100%	
Output	Busy	Under measurement		---
	NumberOfSendCommand	Total number of commands sent		Returns a value when Execute is TRUE. Clears when Execute is FALSE.
	NumberOfLossCommand	Total number of commands lost		Returns a value when Execute is TRUE. Clears when Execute is FALSE.
	bContinuousLossError	Occurrence of a continuous command loss error		Occurrence of a condition error of ContinuousLossCondtion
	bLossRateError	Occurrence of a command loss statistics error		Occurrence of a condition error of MesurementCycleCondition or LossRateCondition

(Note 1) If the number of frames exceeds 32 bits, normal value is not returned.

9.1.13 RTEX_ReadPot (Read POT of Amplifier)

This is a function that reads the POT state of the amplifier.

■ Icon



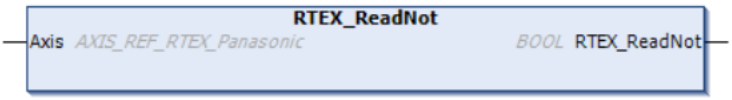
■ Parameter

Type	Parameter name	Type	Description
I/O	Axis	AXIS_REF_RTEX_Panasonic	Specifies the axis.
Output	RTEX_ReadPot	BOOL	TRUE: POT is ON.

9.1.14 RTEX_ReadNot (Read NOT of Amplifier)

This is a function that reads the NOT state of the amplifier.

■ Icon



■ Parameter

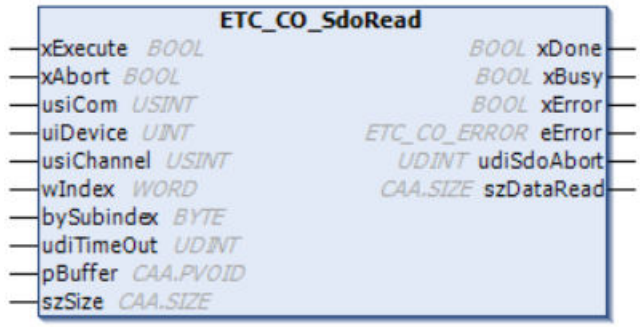
Type	Parameter name	Type	Description
I/O	Axis	AXIS_REF_RTEX_Panasonic	Specifies the axis.
Output	RTEX_ReadNot	BOOL	TRUE: NOT is ON.

9.2 EtherCAT

9.2.1 ETC_CO_SdoRead (Read Slave Parameter)

This is a function block (FB) that reads the EtherCAT slave parameters. Unlike ETC_CO_SdoRead4, this FB supports parameters longer than 4 bytes. Specify parameters to be read using the index and sub-index used for the object directory.

■ **Icon**



■ **Parameter**

Scope	Name	Type	Default	Description
Input	xExecute	BOOL	FALSE	TRUE: Starts execution at the rising edge. FALSE: Resets output after execution is completed.
	xAbort	BOOL	FALSE	TRUE: Interrupts processing and resets output.
	usiCom	USINT	1	1 (Fixed)
	uiDevice	UINT	0	Physical slave address
	usiChannel	USINT	1	Reserved
	wIndex	WORD	0	Parameter index in the object directory(Note 1)
	bySubIndex	BYTE	0	Parameter sub-index in the object directory(Note 1)
	udiTimeout	UDINT	0	Timeout (Unit: ms)
	pBuffer	CAA.PVOID	0	Pointer to the buffer that stores read data
	szSize	CAA.SIZE	0	Size of the buffer that stores data
Output	xDone	BOOL	FALSE	TRUE: FB processing is completed.
	xBusy	BOOL	FALSE	TRUE: FB is in progress.
	xError	BOOL	FALSE	TRUE: An error has occurred within the FB.

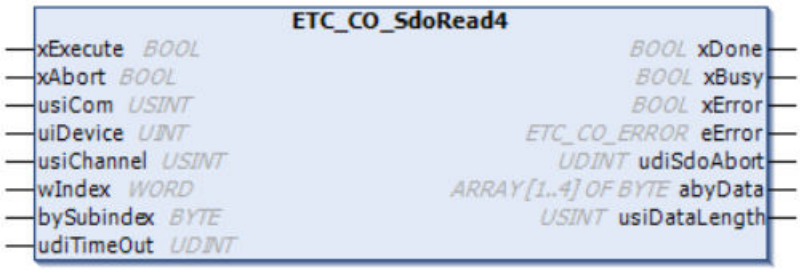
Scope	Name	Type	Default	Description
	eError	ETC_CO_ERROR	ETC_CO_NO_ERROR	Error ID output
	udiSdoAbort	UDINT	0	Abort code received from the slave device
	szDataRead	CAA.SIZE	0	Number of bytes read normally

(Note 1) The parameter content differs according to the slave. Refer to the manuals of corresponding slave devices.

9.2.2 ETC_CO_SdoRead4 (Read Four Bytes of Slave Parameter)

This is a function block (FB) that reads the EtherCAT slave parameters. Unlike ETC_CO_SdoRead, this FB supports only parameters with 4 bytes or less. Specify parameters to be read using the index and sub-index used for the object directory.

■ Icon



■ Parameter

Scope	Name	Type	Default	Description
Input	xExecute	BOOL	FALSE	TRUE: Starts execution at the rising edge. FALSE: Resets output after execution is completed.
	xAbort	BOOL	FALSE	TRUE: Interrupts processing and resets output.
	usiCom	USINT	1	1 (Fixed)
	uiDevice	UINT	1	Physical slave address
	usiChannel	USINT	1	Reserved
	wIndex	WORD	0	Parameter index in the object directory(Note 1)
	bySubIndex	BYTE	0	Parameter sub-index in the object directory(Note 1)
	udiTimeout	UDINT	0	Timeout (Unit: ms)
Output	xDone	BOOL	FALSE	TRUE: FB processing is completed.
	xBusy	BOOL	FALSE	TRUE: FB is in progress.

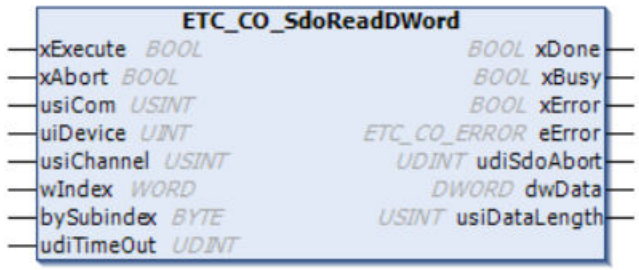
Scope	Name	Type	Default	Description
	xError	BOOL	FALSE	TRUE: An error has occurred within the FB.
	eError	ETC_CO_ERROR	ETC_CO_NO_ERROR	Error ID output
	udiSdoAbort	UDINT	0	Abort code received from the slave device
	abyData	ARRAY [1..4] OF BYTE	-	Read data storage location
	usiDataLength	USINT	0	Number of read bytes

(Note 1) The parameter content differs according to the slave. Refer to the manuals of corresponding slave devices.

9.2.3 ETC_CO_SdoReadDWord (Read Double Word of Slave Parameter)

Just like ETC_CO_SdoRead4, this is a function block (FB) that reads the EtherCAT slave parameters. The read data is stored in DWORD (dwData), not in an array. Since byte swapping is automatically executed, read data can be directly used.

■ Icon



■ Parameter

Scope	Name	Type	Default	Description
Input	xExecute	BOOL	FALSE	TRUE: Starts execution at the rising edge. FALSE: Resets output after execution is completed.
	xAbort	BOOL	FALSE	TRUE: Interrupts processing and resets output.
	usiCom	USINT	1	1 (Fixed)
	uiDevice	UINT	0	Physical slave address
	usiChannel	USINT	1	Reserved
	wIndex	WORD	0	Parameter index in the object directory(Note 1)

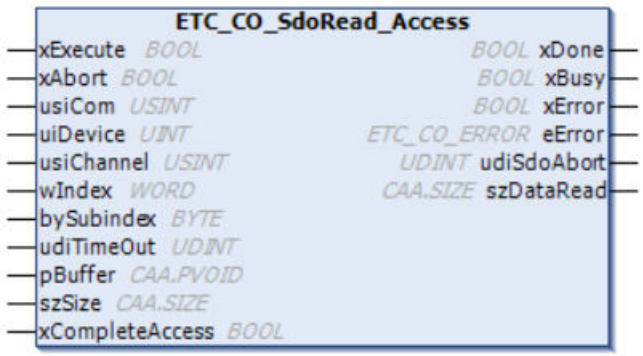
Scope	Name	Type	Default	Description
	bySubIndex	BYTE	0	Parameter sub-index in the object directory(Note 1)
	udiTimeout	UDINT	0	Timeout (Unit: ms)
Output	xDone	BOOL	FALSE	TRUE: FB processing is completed.
	xBusy	BOOL	FALSE	TRUE: FB is in progress.
	xError	BOOL	FALSE	TRUE: An error has occurred within the FB.
	eError	ETC_CO_ERROR	ETC_CO_NO_ERROR	Error ID output
	udiSdoAbort	UDINT	0	Abort code received from the slave device
	dwData	DWORD	0	Read data storage location
	usiDataLength	USINT	0	Number of read bytes

(Note 1) The parameter content differs according to the slave. Refer to the manuals of corresponding slave devices.

9.2.4 ETC_CO_SdoRead_Access (Read Slave Parameter Index)

Just like ETC_CO_SdoRead, this is a function block (FB) that reads the EtherCAT slave parameters. By setting the xCompleteAccess input to TRUE and the bySubIndex input to 0, you can read complete indexes including all entries.

■ Icon



■ Parameter

Scope	Name	Type	Default	Description
Input	xExecute	BOOL	FALSE	TRUE: Starts execution at the rising edge. FALSE: Resets output after execution is completed.
	xAbort	BOOL	FALSE	TRUE: Interrupts processing and resets output.

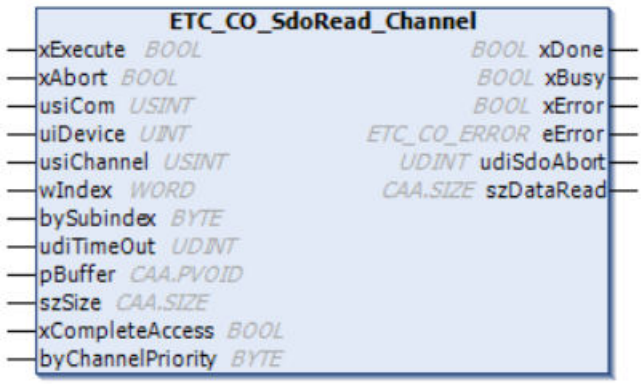
Scope	Name	Type	Default	Description
	usiCom	USINT	1	1 (Fixed)
	uiDevice	UINT	0	Physical slave address
	usiChannel	USINT	1	Reserved
	wIndex	WORD	0	Parameter index in the object directory(Note 1)
	bySubIndex	BYTE	0	Parameter sub-index in the object directory(Note 1)
	udiTimeout	UDINT	0	Timeout (Unit: ms)
	pBuffer	CAA.PVOID	0	Pointer to the buffer that stores read data
	szSize	CAA.SIZE	0	Size of the buffer that stores data
	xCompleteAccess	BOOL	FALSE	TRUE: Accesses all sub-indexes within the specified index.
Output	xDone	BOOL	FALSE	TRUE: FB processing is completed.
	xBusy	BOOL	FALSE	TRUE: FB is in progress.
	xError	BOOL	FALSE	TRUE: An error has occurred within the FB.
	eError	ETC_CO_ERROR	ETC_CO_NO_ERROR	Error ID output
	udiSdoAbort	UDINT	0	Abort code received from the slave device
	szDataRead	CAA.SIZE	0	Number of read bytes

(Note 1) The parameter content differs according to the slave. Refer to the manuals of corresponding slave devices.

9.2.5 ETC_CO_SdoRead_Channel (Read Priority Specification of Slave Parameter)

Just like ETC_CO_SdoRead_Access, this is a function block (FB) that reads the EtherCAT slave parameters. By using the byChannelPriority (BYTE) input, you can specify the channel and priority using a CoE mailbox message. Specify the channel with the first 6 bits (bit0 to bit5) and the priority with the last 2 bits (bit6 and bit7).

■ Icon



■ Parameter

Scope	Name	Type	Default	Description
Input	xExecute	BOOL	FALSE	TRUE: Starts execution at the rising edge. FALSE: Resets output after execution is completed.
	xAbort	BOOL	FALSE	TRUE: Interrupts processing and resets output.
	usiCom	USINT	1	1 (Fixed)
	uiDevice	UINT	0	Physical slave address
	usiChannel	USINT	1	Reserved
	wIndex	WORD	0	Parameter index in the object directory(Note 1)
	bySubIndex	BYTE	0	Parameter sub-index in the object directory(Note 1)
	udiTimeout	UDINT	0	Timeout (Unit: ms)
	pBuffer	CAA.PVOID	0	Pointer to the buffer that stores read data
	szSize	CAA.SIZE	0	Size of the buffer that stores data
	xCompleteAccess	BOOL	FALSE	TRUE: Accesses all sub-indexes within the specified index.
	byChannelPriority	BYTE	0	Specifies the channel and priority using a CoE mailbox message.
Output	xDone	BOOL	FALSE	TRUE: FB processing is completed.
	xBusy	BOOL	FALSE	TRUE: FB is in progress.
	xError	BOOL	FALSE	TRUE: An error has occurred within the FB.
	eError	ETC_CO_ERROR	ETC_CO_NO_ERROR	Error ID output
	udiSdoAbort	UDINT	0	Abort code received from the slave device

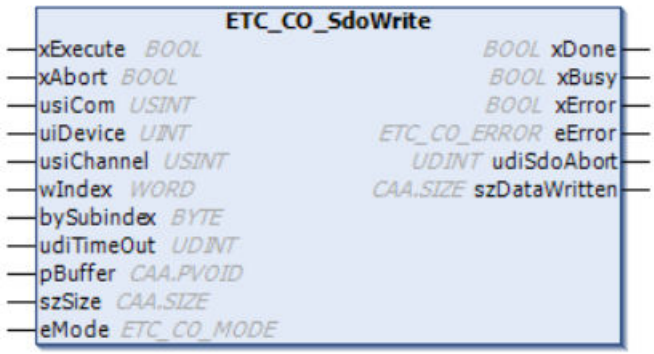
Scope	Name	Type	Default	Description
	szDataRead	CAA.SIZE	0	Number of read bytes

(Note 1) The parameter content differs according to the slave. Refer to the manuals of corresponding slave devices.

9.2.6 ETC_CO_SdoWrite (Write Slave Parameter)

This is a function block (FB) that writes the EtherCAT slave parameters. Unlike ETC_CO_SdoWrite4, this FB supports parameters longer than 4 bytes. Specify parameters to be written using the index and sub-index used for the object directory.

■ Icon



■ Parameter

Scope	Name	Type	Default	Description
Input	xExecute	BOOL	FALSE	TRUE: Starts execution at the rising edge. FALSE: Resets output after execution is completed.
	xAbort	BOOL	FALSE	TRUE: Interrupts processing and resets output.
	usiCom	USINT	1	1 (Fixed)
	uiDevice	UINT	0	Physical slave address
	usiChannel	USINT	1	Reserved
	wIndex	WORD	0	Parameter index in the object directory(Note 1)
	bySubIndex	BYTE	0	Parameter sub-index in the object directory(Note 1)
	udiTimeout	UDINT	0	Timeout (Unit: ms)
	pBuffer	CAA.PVOID	0	Pointer to the buffer storing write data
	szSize	CAA.SIZE	0	Number of written bytes

Scope	Name	Type	Default	Description
	eMode	ETC_CO_MODE	ETC_CO_AUTO	Transmission mode
Output	xDone	BOOL	FALSE	TRUE: FB processing is completed.
	xBusy	BOOL	FALSE	TRUE: FB is in progress.
	xError	BOOL	FALSE	TRUE: An error has occurred within the FB.
	eError	ETC_CO_ERROR	ETC_CO_NO_ERROR	Error ID output
	udiSdoAbort	UDINT	0	Abort code received from the slave device
	szDataWritten	CAA.SIZE	0	Number of bytes written normally

(Note 1) The parameter content differs according to the slave. Refer to the manuals of corresponding slave devices.

■ ETC_CO_ERROR (Union type)

Member	Type	Description
ETC_CO_NO_ERROR	WORD	No error
ETC_CO_FIRST_ERROR	WORD	Check udiSdoAbort for the cause of errors.
ETC_CO_OTHER_ERROR	WORD	The master is not found.
ETC_CO_DATA_OVERFLOW	WORD	ETC_CO_Expedited and size exceed 4.
ETC_CO_TIMEOUT	WORD	The time limit is exceeded.
ETC_CO_FIRST_MF	WORD	Not used
ETC_CO_LAST_ERROR	WORD	Not used

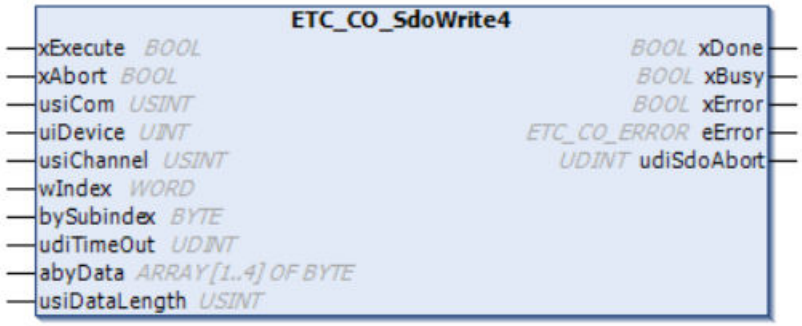
■ ETC_CO_MODE (Union type)

Member	Type	Description
ETC_CO_AUTO	WORD	Mode is selected automatically.
ETC_CO_EXPEDITED	WORD	Expedited transfer
ETC_CO_SEGMENTED	WORD	Segmented transfer

9.2.7 ETC_CO_SdoWrite4 (Write Four Bytes of Slave Parameter)

This is a function block (FB) that writes the EtherCAT slave parameters. Unlike ETC_CO_SdoWrite, this FB supports only parameters with 4 bytes or less. Specify parameters to be written using the index and sub-index used for the object directory.

■ Icon



■ Parameter

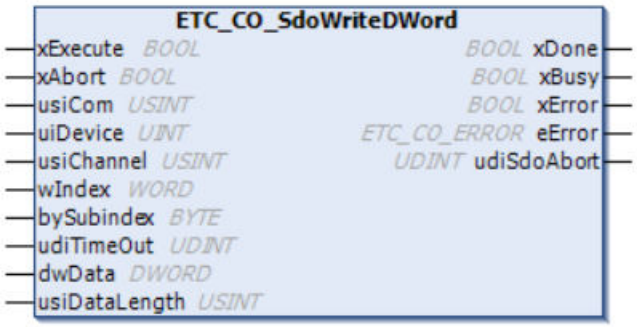
Scope	Name	Type	Default	Description
Input	xExecute	BOOL	FALSE	TRUE: Starts execution at the rising edge. FALSE: Resets output after execution is completed.
	xAbort	BOOL	FALSE	TRUE: Interrupts processing and resets output.
	usiCom	USINT	1	1 (Fixed)
	uiDevice	UINT	0	Physical slave address
	usiChannel	USINT	1	Reserved
	wIndex	WORD	0	Parameter index in the object directory(Note 1)
	bySubIndex	BYTE	0	Parameter sub-index in the object directory(Note 1)
	udiTimeout	UDINT	0	Timeout (Unit: ms)
	abyData	ARRAY [1..4] OF BYTE	-	Write data storage location
	usiDataLength	USINT	0	Number of written bytes
Output	xDone	BOOL	FALSE	TRUE: FB processing is completed.
	xBusy	BOOL	FALSE	TRUE: FB is in progress.
	xError	BOOL	FALSE	TRUE: An error has occurred within the FB.
	eError	ETC_CO_ERROR	ETC_CO_NO_ERROR	Error ID output
	udiSdoAbort	UDINT	0	Abort code received from the slave device

(Note 1) The parameter content differs according to the slave. Refer to the manuals of corresponding slave devices.

9.2.8 ETC_CO_SdoWriteDWord (Write Double Words of Slave Parameter)

Just like ETC_CO_SdoWrite4, this is a function block (FB) that writes the EtherCAT slave parameters. The write data is transferred in DWORD (dwData), not in an array. Since byte swapping is automatically executed, write data can be directly used.

■ Icon



■ Parameter

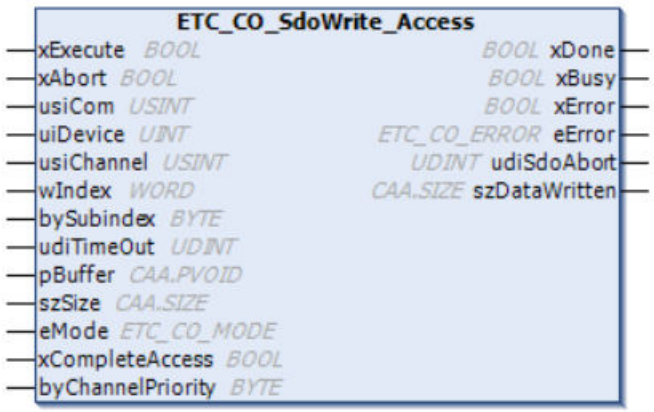
Scope	Name	Type	Default	Description
Input	xExecute	BOOL	FALSE	TRUE: Starts execution at the rising edge. FALSE: Resets output after execution is completed.
	xAbort	BOOL	FALSE	TRUE: Interrupts processing and resets output.
	usiCom	USINT	1	1 (Fixed)
	uiDevice	UINT	0	Physical slave address
	usiChannel	USINT	1	Reserved
	wIndex	WORD	0	Parameter index in the object directory(Note 1)
	bySubIndex	BYTE	0	Parameter sub-index in the object directory(Note 1)
	udiTimeout	UDINT	0	Timeout (Unit: ms)
	dwData	DWORD	0	Write data storage location
	usiDataLength	USINT	0	Number of written bytes
Output	xDone	BOOL	FALSE	TRUE: FB processing is completed.
	xBusy	BOOL	FALSE	TRUE: FB is in progress.
	xError	BOOL	FALSE	TRUE: An error has occurred within the FB.
	eError	ETC_CO_ERROR	ETC_CO_NO_ERROR	Error ID output
	udiSdoAbort	UDINT	0	Abort code received from the slave device

(Note 1) The parameter content differs according to the slave. Refer to the manuals of corresponding slave devices.

9.2.9 ETC_CO_SdoWrite_Access (Write Slave Parameter Index)

Just like ETC_CO_SdoWrite, this is a function block (FB) that writes the EtherCAT slave parameters. By setting the xCompleteAccess input to TRUE and the bySubIndex input to 0, you can write complete indexes including all entries. By using the byChannelPriority (BYTE) input, you can specify the channel and priority using a CoE mailbox message. Specify the channel with the first 6 bits (bit0 to bit5) and the priority with the last 2 bits (bit6 and bit7).

■ Icon



■ Parameter

Scope	Name	Type	Default	Description
Input	xExecute	BOOL	FALSE	TRUE: Starts execution at the rising edge. FALSE: Resets output after execution is completed.
	xAbort	BOOL	FALSE	TRUE: Interrupts processing and resets output.
	usiCom	USINT	1	1 (Fixed)
	uiDevice	UINT	0	Physical slave address
	usiChannel	USINT	1	Reserved
	wIndex	WORD	0	Parameter index in the object directory(Note 1)
	bySubIndex	BYTE	0	Parameter sub-index in the object directory(Note 1)
	udiTimeout	UDINT	0	Timeout (Unit: ms)
	pBuffer	CAA.PVOID	0	Pointer to the buffer storing write data
	szSize	CAA.SIZE	0	Number of written bytes

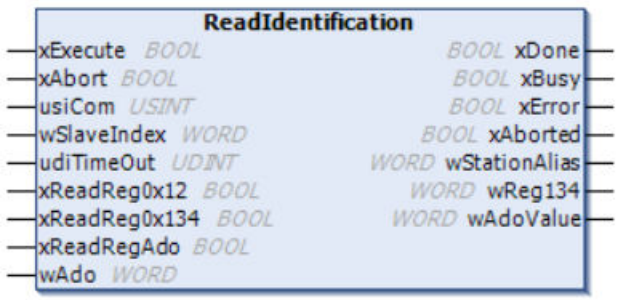
Scope	Name	Type	Default	Description
	eMode	ETC_CO_MODE	ETC_CO_AUTO	Transmission mode
	xCompleteAccess	BOOL	FALSE	TRUE: Accesses all sub-indexes within the specified index.
	byChannelPriority	BYTE		Specifies the channel and priority using a CoE mailbox message.
Output	xDone	BOOL	FALSE	TRUE: FB processing is completed.
	xBusy	BOOL	FALSE	TRUE: FB is in progress.
	xError	BOOL	FALSE	TRUE: An error has occurred within the FB.
	eError	ETC_CO_ERROR	ETC_CO_NO_ERROR	Error ID output
	udiSdoAbort	UDINT	0	Abort code received from the slave device
	szDataWritten	CAA.SIZE	0	Number of bytes written normally

(Note 1) The parameter content differs according to the slave. Refer to the manuals of corresponding slave devices.

9.2.10 ReadIdentification (Read Slave Identification Data)

Reads identification data from EtherCAT slaves.

■ Icon



■ Parameter

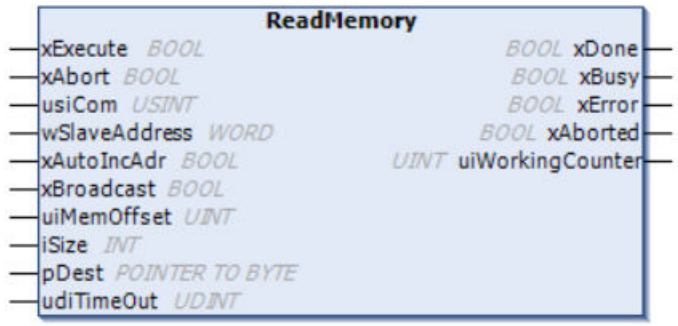
Scope	Name	Type	Default	Description
Input	xExecute	BOOL	FALSE	TRUE: Starts execution at the rising edge. FALSE: Resets output after execution is completed.
	xAbort	BOOL	FALSE	TRUE: Interrupts processing and resets output.
	usiCom	USINT	1	1 (Fixed)
	wSlaveIndex	WORD	0	Specifies EtherCAT slaves. EtherCAT slave numbers are

Scope	Name	Type	Default	Description
				allocated in ascending order from 0 to the one closest to the master.
	udiTimeout	UDINT	0	Timeout (Unit: ms) When executed with the default value 0, it will timeout immediately.
	xReadReg0x12	BOOL	FALSE	Register 16#12 (Station alias) read flag
	xReadReg0x134	BOOL	FALSE	Register 16#134 (Explicit Device ID) read flag
	xReadRegAdo	BOOL	FALSE	Register Ado read flag
	wAdo	WORD	0	Ado ID address
Output	xDone	BOOL	FALSE	TRUE: FB processing is completed.
	xBusy	BOOL	FALSE	TRUE: FB is in progress.
	xError	BOOL	FALSE	TRUE: An error has occurred within the FB.
	xAborted	BOOL	FALSE	TRUE: An interruption has occurred in FB processing.
	wStationAlias	WORD	0	Value of register 16#12
	wReg134	WORD	0	Value of register 16#0x134
	wAdoValue	WORD	0	Value of Ado ID

9.2.11 ReadMemory (Read Slave Memory)

This is a function block (FB) that reads the EtherCAT slave memory.

■ Icon



■ Parameter

Scope	Name	Type	Default	Description
Input	xExecute	BOOL	FALSE	TRUE: Starts execution at the rising edge.

Scope	Name	Type	Default	Description
				FALSE: Resets output after execution is completed.
	xAbort	BOOL	FALSE	TRUE: Interrupts processing and resets output.
	usiCom	USINT	1	Master device index (1 onwards)
	wSlaveAddress	WORD	0	Either the automatic incremental address or the device's physical address
	xAutoIncAdr	BOOL	FALSE	TRUE: Uses the automatic incremental address. When set to TRUE, specify the automatic incremental address for wSlaveAddress.
	xBroadcast	BOOL	FALSE	TRUE: Uses the broadcast read. If set to TRUE, wSlaveAddress and bAutoIncAdr are not used.
	uiMemOffset	UINT	0	Offset of the memory
	iSize	INT	0	Number of read bytes
	pDest	POINTER TO BYTE	0	Pointer to the buffer that stores read data
	udiTimeOut	UDINT	0	Timeout (Unit: ms) When executed with the default value 0, it will timeout immediately.
Output	xDone	BOOL	FALSE	TRUE: FB processing is completed.
	xBusy	BOOL	FALSE	TRUE: FB is in progress.
	xError	BOOL	FALSE	TRUE: An error has occurred within the FB.
	xAborted	BOOL	FALSE	TRUE: An interruption has occurred in FB processing.
	uiWorkingCounter	UINT	0	Working counter of received commands

9.2.12 ReadNbrSlaves (Read the Number of Connected Slaves)

Reads the number of slaves currently connected.

■ **Icon**

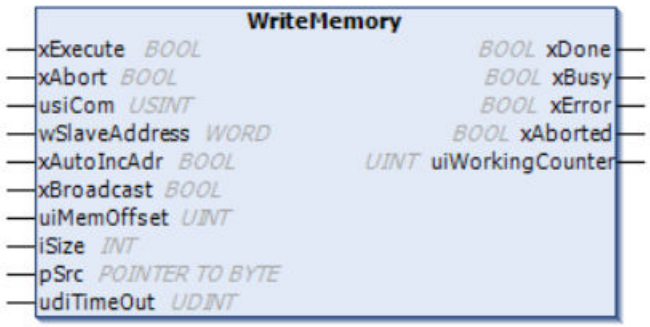
■ **Parameter**

Scope	Name	Type	Default	Description
Input	xExecute	BOOL	FALSE	TRUE: Starts execution at the rising edge. FALSE: Resets output after execution is completed.
	xAbort	BOOL	FALSE	TRUE: Interrupts processing and resets output.
	usiCom	USINT	1	1 (Fixed)
	udiTimeOut	UDINT	0	Timeout (Unit: ms) When executed with the default value 0, it will timeout immediately.
Output	xDone	BOOL	FALSE	TRUE: FB processing is completed.
	xBusy	BOOL	FALSE	TRUE: FB is in progress.
	xError	BOOL	FALSE	TRUE: An error has occurred within the FB.
	xAborted	BOOL	FALSE	TRUE: An interruption has occurred in FB processing.
	wNumberSlaves	WORD	0	Number of slaves connected

9.2.13 WriteMemory (Write Slave Memory)

This is a function block (FB) that writes the EtherCAT slave memory. Accesses "ESC address space".

■ **Icon**



■ **Parameter**

Scope	Name	Type	Default	Description
Input	xExecute	BOOL	FALSE	TRUE: Starts execution at the rising edge. FALSE: Resets output after execution is completed.

Scope	Name	Type	Default	Description
	xAbort	BOOL	FALSE	TRUE: Interrupts processing and resets output.
	usiCom	USINT	1	1 (Fixed)
	wSlaveAddress	WORD	0	When set to the physical slave address or to bAutoIncrementAddress, specify the automatic incremental address.
	xAutoIncAdr	BOOL	FALSE	TRUE: Uses the automatic incremental address. When set to TRUE, specify the automatic incremental address for wSlaveAddress.
	xBroadcast	BOOL	FALSE	TRUE: Uses the broadcast read. If set to TRUE, wSlaveAddress and bAutoIncAdr are not used.
	uiMemOffset	UINT	0	Offset of the memory
	iSize	INT	0	Number of bytes to be written
	pSrc	POINTER TO BYTE	0	Address of the data buffer to be written
	udiTimeOut	UDINT	0	Timeout (Unit: ms) When executed with the default value 0, it will timeout immediately.
Output	xDone	BOOL	FALSE	TRUE: FB processing is completed.
	xBusy	BOOL	FALSE	TRUE: FB is in progress.
	xError	BOOL	FALSE	TRUE: An error has occurred within the FB.
	xAborted	BOOL	FALSE	TRUE: An interruption has occurred in FB processing.
	uiWorkingCounter	UINT	0	Working counter of received commands

9.2.14 PETC_ClearAmpPositionalDeviation (Clear Amplifier Deviation Counter)

This is a function block (FB) that clears the deviation counter of the AMP. It deletes the position deviation data in the deviation counter of the AMP.

■ Icon

■ Parameter

Scope	Name	Type	Default	Description
Input / output	Axis	AXIS_REF_RTEX_Panasonic	-	Specifies the axis.
Input	Execute	BOOL	FALSE	Starts execution at the rising edge.
	Velocity(Note 1)	LREAL		Information required to execute MC_MoveAbsolute
	Acceleration(Note 1)	LREAL		Information required to execute MC_MoveAbsolute
	Deceleration(Note 1)	LREAL		Information required to execute MC_MoveAbsolute
	Jerk(Note 1)	LREAL		Information required to execute MC_MoveAbsolute
Output	Done	BOOL	FALSE	TRUE: Clearing is completed.
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	CommandAborted	BOOL	FALSE	TRUE: An interruption from other FB has occurred.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	Error ID output

(Note 1) This function block internally substitutes the command position with an actual position to call MC_MoveAbsolute and, therefore, requires parameters including Velocity, Acceleration, Deceleration, and Jerk.

The PETC_ClearAmpPositionalDeviation function block outputs the following errors.

Error	Description
SMC_WRONG_CONTROLLER_MODE	Executed in a mode other than the position control mode. Change to SMC_position using SMC_SetControllerMode.
SMC_AXIS_NOT_READY_FOR_MOTION	The axis is in a state where PMC_ClearAmpMultiTurnData cannot be executed. It can be executed only at the standstill state.
SMC_REGULATOR_OR_START_NOT_SET	The axis is in a servo OFF state.
SMC_PP_WRONG_AXIS_TYPE	The axis is a virtual axis.

9.2.15 IoDrvEtherCAT.GetStatistics (Get EtherCAT Frame Statistics Information)

Gets the EtherCAT frame statistics information.

Scope	Name	Type	Description
Output	udiSendFrameCount	UDINT	Number of EtherCAT send frames

Scope	Name	Type	Description
	udiFramesPerSecond	UDINT	Number of EtherCAT send frames per second
	udiLostFrameCount	UDINT	Number of missing EtherCAT frames
	udiTxErrorCount	UDINT	Number of send errors
	udiRxErrorCount	UDINT	Number of receive errors

10 Motion Control Function Blocks (Auxiliary Function)

10.1 Motion Auxiliary Function (Monitoring)

10.1.1 MC_ReadActualPosition (Read Current Position)

This is a function block (FB) that reads the actual position data of the axis.

■ Icon

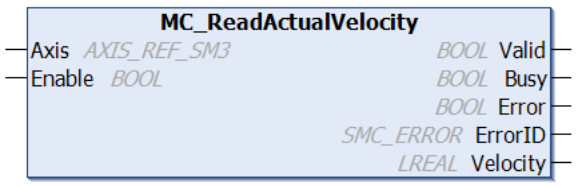
■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	Enable	BOOL	FALSE	Reads the actual position while Enable is set to TRUE.
Output	Valid	BOOL	FALSE	TRUE: Valid output
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	An error ID is output.
	Position	LREAL	0	Actual position (u) that is read out

10.1.2 MC_ReadActualVelocity (Read Current Velocity)

This is a function block (FB) that reads the actual velocity of the axis.

■ Icon



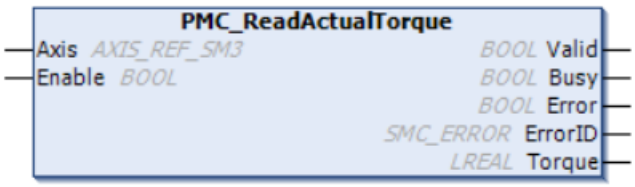
■ **Parameter**

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	Enable	BOOL	FALSE	Reads the actual velocity while Enable is set to TRUE.
Output	Valid	BOOL	FALSE	TRUE: Valid output
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	An error ID is output.
	Velocity	LREAL	0	Current actual velocity (u/s) that is read out

10.1.3 PMC_ReadActualTorque (Read Current Torque)

This is a function block (FB) that reads the actual torque value of the axis.

■ **Icon**



■ **Parameter**

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	Enable	BOOL	FALSE	Reads the actual torque value while Enable is set to TRUE.
Output	Valid	BOOL	FALSE	TRUE: Valid output
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	An error ID is output.
	Torque	LREAL	0	Current actual torque (%) that is read out

10.1.4 MC_ReadActualTorque (Read Current Torque)

This is a function block (FB) that reads the current torque value of the axis.

■ Icon

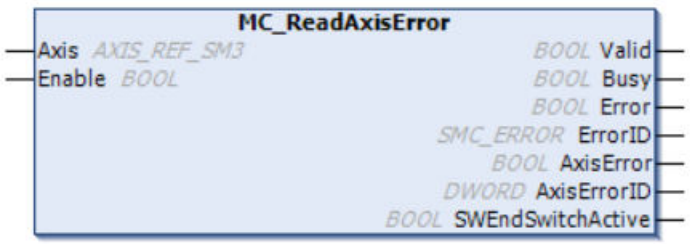
■ Parameter

Scope	Name	Type	Default	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	Enable	BOOL	FALSE	Reads the actual torque value while Enable is set to TRUE.
Output	Valid	BOOL	FALSE	TRUE: Valid output
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	Error ID output
	Torque	LREAL	0	Current actual torque that is read out (N·m, N)

10.1.5 MC_ReadAxisError (Read Axis Error)

This is a function block that gets general axis errors not related to function blocks.

■ Icon



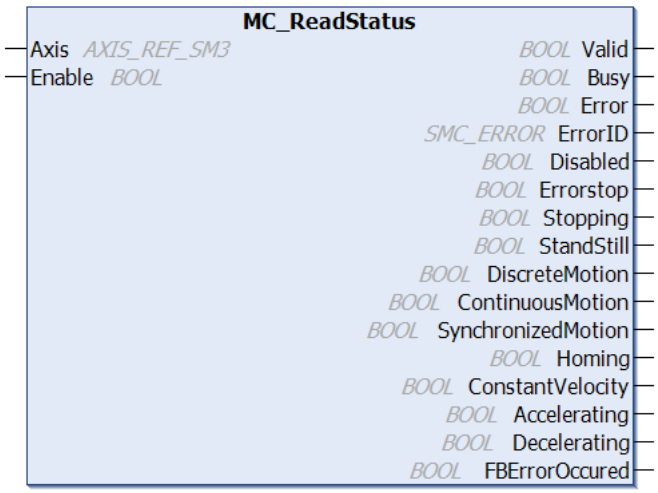
■ **Parameter**

Scope	Name	Type	Default	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	Enable	BOOL	FALSE	Reads the state while Enable is set to TRUE.
Output	Valid	BOOL	FALSE	TRUE: Valid output
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	Error ID output
	AxisError	BOOL	FALSE	TRUE: An axis error has occurred.
	AxisErrorID	DWORD	0	Axis error ID Servo amplifier alarms can be acquired from AxisErrorID. If an Err27.4 error occurs when connected to the MINAS, 27 (16#1B) is set and AxisErrorID becomes 16#0000FF1B.
	SWEndSwitchActive	BOOL	FALSE	TRUE: The software limit has been exceeded.

10.1.6 MC_ReadStatus (Read Status)

This is a function block (FB) that reads the status information of the axis. It reads detailed information about the axis state.

■ **Icon**



■ Parameter

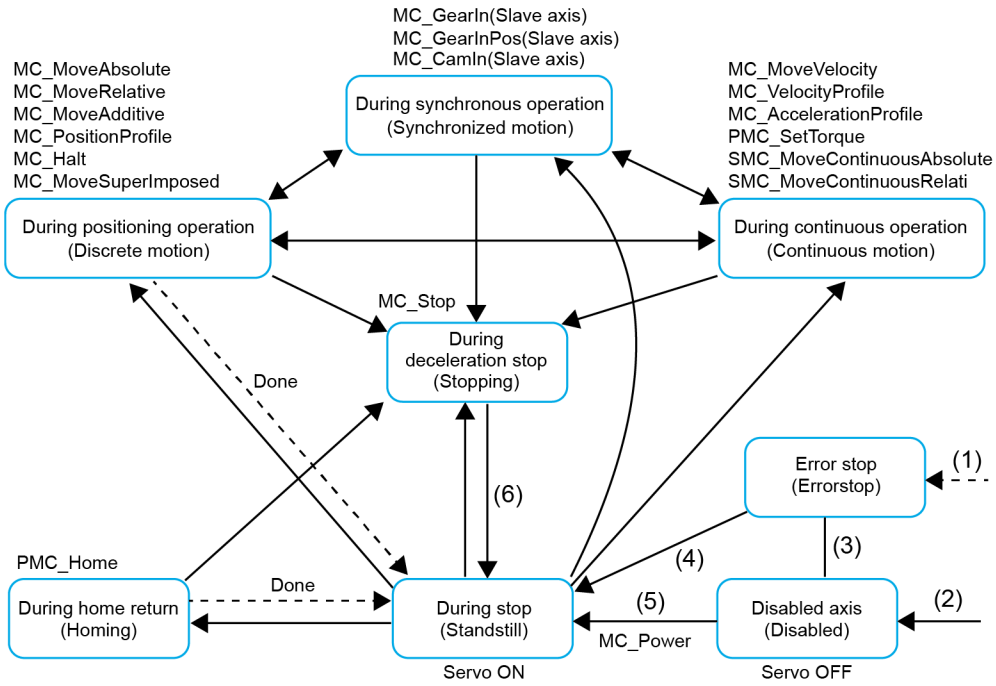
Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	Enable	BOOL	FALSE	Reads the status information while the input is TRUE.
Output	Valid	BOOL	FALSE	TRUE: Valid output
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	An error ID is output.
	Disabled	BOOL	FALSE	TRUE: The axis is in the Disabled state.
	ErrorStop	BOOL	FALSE	TRUE: The axis is in the ErrorStop state.
	Stopping	BOOL	FALSE	TRUE: The axis is in the Stopping state.
	StandStill	BOOL	FALSE	TRUE: The axis is in the StandStill state.
	DiscreteMotion	BOOL	FALSE	TRUE: The axis is in the DiscreteMotion state.
	ContinuousMotion	BOOL	FALSE	TRUE: The axis is in the ContinuousMotion state.
	SynchronizedMotion	BOOL	FALSE	TRUE: The axis is in the SynchronizedMotion state.
	Homing	BOOL	FALSE	TRUE: The axis is in the Homing state.
	ConstantVelocity	BOOL	FALSE	TRUE: The axis is moving at a constant velocity.
	Accelerating	BOOL	FALSE	TRUE: The axis is moving in acceleration.
	Decelerating	BOOL	FALSE	TRUE: The axis is moving in deceleration.
	FBErrorOccured	BOOL	FALSE	TRUE: An FB error has occurred.

■ Axis state

The following section describes state transition diagram of the axis when the motion function blocks are executed.

State transition diagram

- The blue frame indicates the state.
- When the function block indicated above the state is executed, the state transitions to the direction indicated by the solid-line arrow.
- When the execution is completed or when an error occurs, the state transitions to the state indicated at the tip of the broken-line arrow.
- The terms in parentheses are defined in PLCopen.

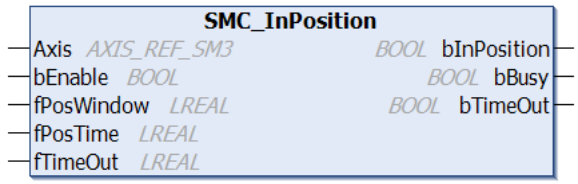


Number	Transition conditions
(1)	Regardless of the state, when an error occurs in the axis
(2)	Regardless of the state, when Enable of MC_Power is TRUE, bRegulator is FALSE, and there is no error in the axis
(3)	When Status of MC_Reset and Status of MC_Power are FALSE
(4)	When Enable of MC_Reset and Enable of MC_Power are TRUE, bRegulator is TRUE, and Status is TRUE
(5)	When Enable of MC_Power is TRUE, bRegulator is TRUE, and Status is TRUE
(6)	When Done of MC_Stop is TRUE and Execute of MC_Stop is FALSE

10.1.7 SMC_InPosition (In-position Judgment)

This is a function block (FB) that compares the actual position of the AMP with the command value and judges whether the position is within the specified range. The maximum difference between the actual position of the AMP and the command value as well as the dwell time are specified to judge (in-position judgment) whether the specified values are satisfied.

■ **Icon**



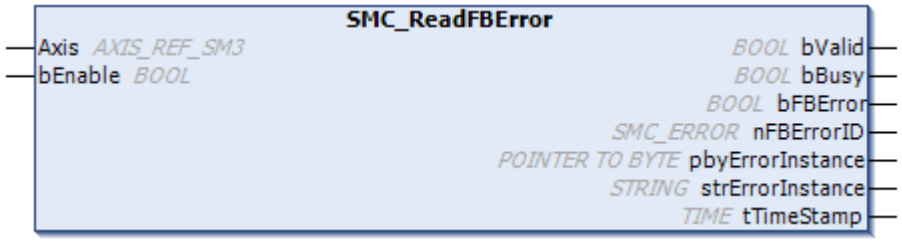
■ **Parameter**

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	bEnable	BOOL	FALSE	TRUE: Executes the FB.
	fPosWindow	LREAL	0	The maximum difference between the actual position and the command value to judge whether the target position has been reached.
	fPosTime	LREAL	0	The dwell time (s) to judge whether the axis has reached the position
	fTimeOut	LREAL	0	The time (s) from when the FB is enabled to when judgment is made that timeout has occurred When the value is “0”, the timeout judgment is not made yet.
Output	bInPosition	BOOL	FALSE	TRUE: The target position is reached. While the difference between the actual position and the command value is within the time specified in fPosTime, it is within the fPosWindow.
	bBusy	BOOL	FALSE	TRUE: The FB is in operation.
	bTimeOut	BOOL	FALSE	TRUE: Timeout has occurred.

10.1.8 SMC_ReadFBError (Read Oldest Error)

This is a function block (FB) that reads the oldest function block error information.

■ **Icon**



■ **Parameter**

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	bEnable	BOOL	FALSE	TRUE: The FB can be executed.
Output	bValid	BOOL	FALSE	TRUE: Error information is acquired.
	bBusy	BOOL	FALSE	TRUE: The FB is in operation.
	bFBError	BOOL	FALSE	TRUE: An error has occurred within the FB.
	nFBErrorID	SMC_ERROR	0	An error ID is output.
	pbyErrorInstance	POINTER TO BYTE	0	FB instance of the error acquisition source
	strErrorInstance	STRING	''	FB instance name of the error acquisition source
	tTimeStamp	TIME	TIME#0ms	Time stamp of the error information

Note

- The error information is cleared when SMC_ClearFBError is executed. When and error occurs again, SMC_ReadFBError reads the error.

REFERENCE

10.1.9 SMC_ClearFBError (Clear Oldest Error)

10.1.9 SMC_ClearFBError (Clear Oldest Error)

This function clears the oldest FB error information.

■ **Icon**

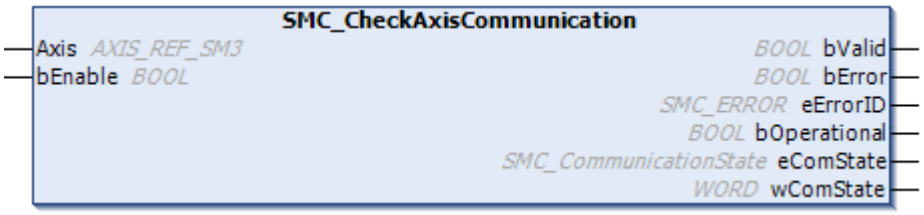
■ **Parameter**

Type	Parameter name	Type	Default	Description
Input	pDrive	POINTER TO AXIS_REF_SM3	-	Specifies the axis.
Return	SMC_ClearFBError	BOOL		This function always returns FALSE even for normal completion.

10.1.10 SMC_CheckAxisCommunication (Check Axis Communication Status)

This is a function block (FB) that checks the communication state of the axis.

■ **Icon**



■ **Parameter**

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	bEnable	BOOL	FALSE	TRUE: The FB can be executed.
Output	bValid	BOOL	FALSE	TRUE: The output value is valid.
	bError	BOOL	FALSE	TRUE: An error has occurred within the FB.
	eErrorID	SMC_ERROR	0	An error ID is output.
	bOperational	BOOL	FALSE	TRUE: Communication state is operational. (100)
	eComState	SMC_CommunicationState	SMC_COMSTATE_NOT_STARTED	Communication state
	wComState	WORD	0	Internal value of the communication state

■ **SMC_CommunicationState (Enumeration type)**

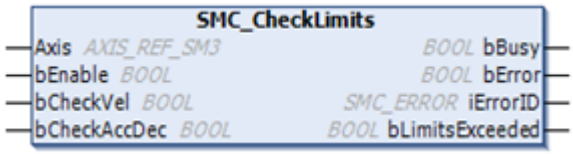
Name	Value	Description
SMC_COMSTATE_NOT_STARTED	0	Stop
SMC_COMSTATE_VARIABLE_INITIALIZATION	1	Initialization of variables

Name	Value	Description
SMC_COMSTATE_BASE_COM_INITIALIZATION	2	Initialization of base communication settings
SMC_COMSTATE_DRIVE_INITIALIZATION	3	Initialization of drive settings
SMC_COMSTATE_DRIVE_WAITING_FOR_SYNC	4	Waiting for drive synchronization
SMC_COMSTATE_INITIALIZATION_DONE	5	Initialization completed
SMC_COMSTATE_OPERATIONAL	6	Operational
SMC_COMSTATE_REINITIALIZATION	7	Re-initialization
SMC_COMSTATE_ERROR	8	Error
SMC_COMSTATE_UNKNOWN	9	Unknown

10.1.11 SMC_CheckLimits (Check Exceeding Limits)

This is a function block (FB) that checks whether the velocity, acceleration, or deceleration is in excess of the dynamic limit set value of the device.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	bEnable	BOOL	FALSE	TRUE: The FB can be executed.
	bCheckVel	BOOL	TRUE	TRUE: Checks the velocity setting.
	bCheckAccDec	BOOL	FALSE	TRUE: Checks the acceleration and deceleration settings.
Output	bBusy	BOOL	FALSE	TRUE: The FB is in operation.
	bError	BOOL	FALSE	TRUE: An error has occurred within the FB.
	iErrorID	SMC_ERROR	0	An error ID is output.
	bLimitsExceeded	BOOL	FALSE	TRUE: Limits are exceeded.

Info.

- Reference manual
  GM1 Controller RTEX User's Manual (Operation Edition)
  GM1 Controller EtherCAT User's Manual (Operation Edition)

10.1.12 SMC_GetMaxSetAccDec (Measure Maximum Acceleration / Deceleration)

This is a function block (FB) that measures the maximum value of the axis acceleration/ deceleration command.

■ Icon

■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	bEnable	BOOL	FALSE	TRUE: The FB can be executed.
	dwTimeStamp	DWORD	0	Time stamp
Output	bValid	BOOL	FALSE	TRUE: The output value is valid.
	bBusy	BOOL	FALSE	TRUE: The FB is in operation.
	fMaxAcceleration	LREAL	0	Maximum acceleration (u/s^2).
	dwTimeAtMax	DWORD	0	dwTimeStamp value at the maximum acceleration

Note

- It is possible to check when the maximum acceleration or deceleration has occurred by entering a call counter value in the input variable "dwTimeStamp".

10.1.13 SMC_GetMaxSetVelocity (Measure Maximum Velocity)

This is a function block (FB) that measures the maximum value of the axis velocity command.

■ **Icon**

■ **Parameter**

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	bEnable	BOOL	FALSE	TRUE: The FB can be executed.
	dwTimeStamp	DWORD	0	Time stamp
Output	bValid	BOOL	FALSE	TRUE: The output value is valid.
	bBusy	BOOL	FALSE	TRUE: The FB is in operation.
	fMaxVelocity	LREAL	0	Maximum velocity (u/s).
	dwTimeAtMax	DWORD	0	dwTimeStamp value at the maximum acceleration

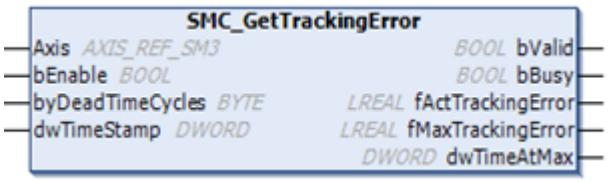
Note

- It is possible to check when the maximum velocity has occurred by entering a call counter value in the input variable "dwTimeStamp".

10.1.14 SMC_GetTrackingError (Measure Tracking Error)

This is a function block (FB) that measures the tracking error of the actual position for the axis command position.

■ **Icon**



■ **Parameter**

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	bEnable	BOOL	FALSE	TRUE: The FB can be executed.
	byDeadTimeCycles	BYTE	2	Number of dead time cycles

Scope	Name	Type	Initial	Description
				Compares the command position and actual position between the specified cycles.
	dwTimeStamp	DWORD	0	Time stamp
Output	bValid	BOOL	FALSE	TRUE: The output value is valid.
	bBusy	BOOL	FALSE	TRUE: The FB is in operation.
	fActTrackingError	LREAL	0	Actual tracking error
	fMaxTrackingError	LREAL	0	Maximum tracking error while the function block is being executed
	dwTimeAtMax	DWORD	0	dwTimeStamp value when the maximum tracking error is detected

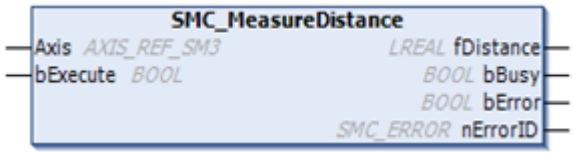
Note

- It is possible to check when the maximum tracking error has occurred by entering a call counter value in the input variable "dwTimeStamp".

10.1.15 SMC_MeasureDistance (Measure Turnaround Travel Distance)

This is a function block (FB) that measures the travel distance. For the modulo axis, the cover distance can be measured considering the laps.

■ Icon



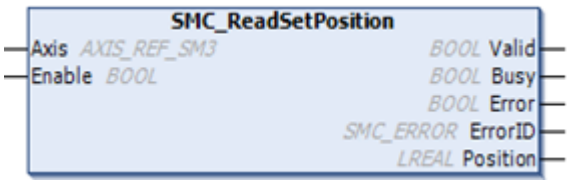
■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	bExecute	BOOL	FALSE	TRUE: Starts measurement at the rising edge. FALSE: Ends measurement.
Output	fDistance	LREAL	0	Distance traveled from the start of measurement
	bBusy	BOOL	FALSE	TRUE: The FB is in operation.
	bError	BOOL	0	TRUE: An error has occurred within the FB.
	nErrorID	SMC_ERROR	0	An error ID is output.

10.1.16 SMC_ReadSetPosition (Read Axis Set Position)

This is a function block (FB) that acquires the command position of the axis.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	Enable	BOOL	FALSE	TRUE: Executes the FB.
Output	Valid	BOOL	FALSE	TRUE: The output value is valid.
	Busy	BOOL	FALSE	TRUE: The FB is in operation.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	An error ID is output.
	Position	LREAL	0	Axis position

10.2 Motion Auxiliary Function (Change / Reset)

10.2.1 MC_Reset (Axis Error Reset)

This is a function block (FB) that resets the state transition error of the axis. It reset the axis error and transitions the state from the ErrorStop state to the StandStill state.

■ Icon

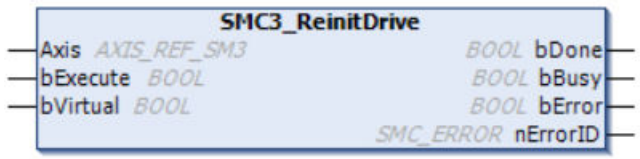
■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	Execute	BOOL	FALSE	Starts execution at the rising edge.
Output	Done	BOOL	FALSE	TRUE: Reset done
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	An error ID is output.

10.2.2 SMC3_ReinitDrive (Reinitialize Axis)

This is a function block that restarts the drive / axis. It means that the startup phase is executed again and the application cannot control the drive until bDone of the FB is set to TRUE.

■ Icon



■ Parameter

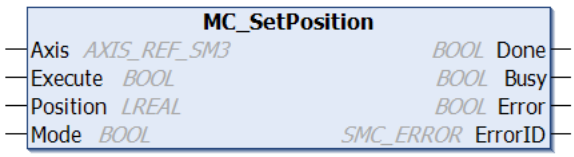
Scope	Name	Type	Default	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.

Scope	Name	Type	Default	Description
Input	bExecute	BOOL	FALSE	Starts execution at the rising edge.
	bVirtual	BOOL	FALSE	If bVirtual is set to TRUE, the axis is set to the virtual mode.
Output	bDone	BOOL	FALSE	TRUE: Reset is completed.
	bBusy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	bError	BOOL	FALSE	TRUE: An error has occurred within the FB.
	nErrorID	SMC_ERROR	0	Error ID output

10.2.3 MC_SetPosition (Change Current Position)

This is a function block (FB) that changes the current command position of the axis.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	Execute	BOOL	FALSE	Starts execution at the rising edge.
	Position	LREAL	0	Specifies the position when the mode is set to ABSOLUTE. Specifies the distance when the mode is set to RELATIVE.
	Mode	BOOL	FALSE	TRUE: RELATIVE (Relative position) FALSE: ABSOLUTE (Absolute position)
Output	Done	BOOL	FALSE	TRUE: Position change is completed.
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	Error	BOOL	FALSE	TRUE: An error has occurred.
	ErrorID	SMC_ERROR	0	An error ID is output.

10.2.4 SMC_ChangeDynamicLimits(Dynamic limit change)

This is a function block that changes the dynamic limits (velocity, acceleration, deceleration, jerk) of the real and virtual axes.When the axis state is power_off or standstill,this FB can be used.

■ Icon

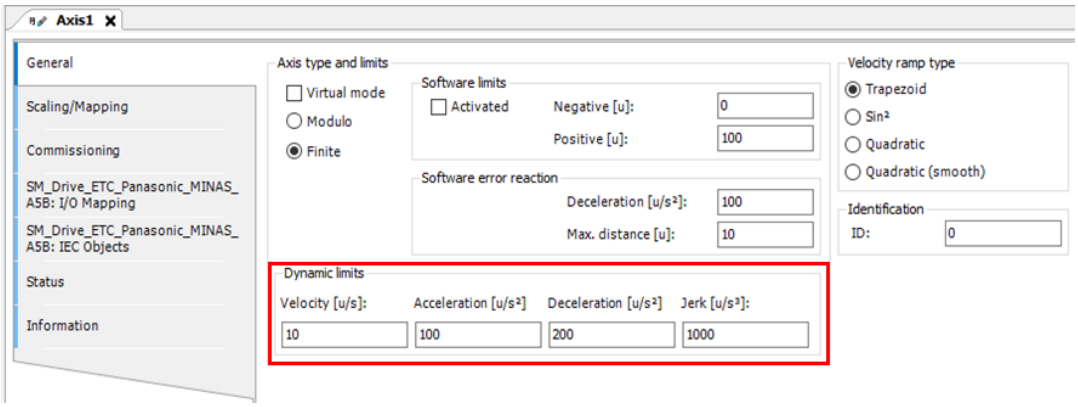
■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	bExecute	BOOL	FALSE	Starts execution at the rising edge.
	fMaxVelocity	LREAL	0	Dynamic limit velocity to set [u / s] (Note 1)
	fMaxAcceleration	LREAL	0	Dynamic limit acceleration to set [u / s^2] (Note 1)
	fMaxDeceleration	LREAL	0	Dynamic limit deceleration to set [u / s^2] (Note 1)
	fMaxJerk	LREAL	0	Dynamic limit Jerk to set[u / s^3] (Note 1)
Output	bDone	BOOL	FALSE	TRUE：FB execution completed
	bError	BOOL	FALSE	TRUE: An error has occurred.
	bBusy	BOOL	FALSE	TRUE：The FB is in operation.
	ErrorID	SMC_ERROR	SMC_NO_ERROR	An error ID is output.

(Note 1) An error occurs when 0 is set for each argument. Set a positive value.

■ Setting location

The dynamic limit in the figure below can be changed while GM1 is running.



Info.

- When the GM1 is powered off, cold reset, or warm reset, it returns to the previous setting value.
- For dynamic restrictions, refer to the GM1 Series Reference Manual (Operation).

10.2.5 SMC_ChangeGearingRatio(Gear ratio and axis type change)

This is a function block that changes the gear ratio and Axis type (Finite / Modulo) of the real and virtual axes.On change, the axis must be restarted by SMC3_ReinitDrived.When the axis state is power_off ,this FB can be used.

■ Icon

■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	bExecute	BOOL	FALSE	Starts execution at the rising edge.
	dwRatioTechUnitsDenom	DWORD	0	use the multiplication of three parameters in the figures (1) and (3) below. (Note 1)
	iRatioTechUnitsNum	DINT	0	use the multiplication of three parameters in the figures (2) and (4) below. (Note 2)

Scope	Name	Type	Initial	Description
	fPositionPeriod	LREAL	0	Effective for the modulo value and the modulo axis. (Note 3)
	iMovementType	SMC_MOVEMENTTYPE	0	Specify axis type
Output	bDone	BOOL	FALSE	TRUE：FB execution completed.
	bBusy	BOOL	FALSE	TRUE：The FB is in operation.
	bError	BOOL	FALSE	TRUE：An error has occurred.
	ErrorID	SMC_ERROR	SMC_NO_ERROR	An error ID is output.

(Note 1) Set other than 0.
When iMovementType is 0, do not set a value higher than 16 # 7FFFFF.

(Note 2) Set other than 0.When a negative value is set, the axis is reversed.
When iMovementType is 0, set a multiple of 360.

(Note 3) Set a positive value.
When iMovementType is 0, set a multiple of 360.

■ SMC_MOVEMENTTYPE（Enumeration type）

Name	Value	Description
rotary	0	the modulo axis
linear	1	the finite axis<

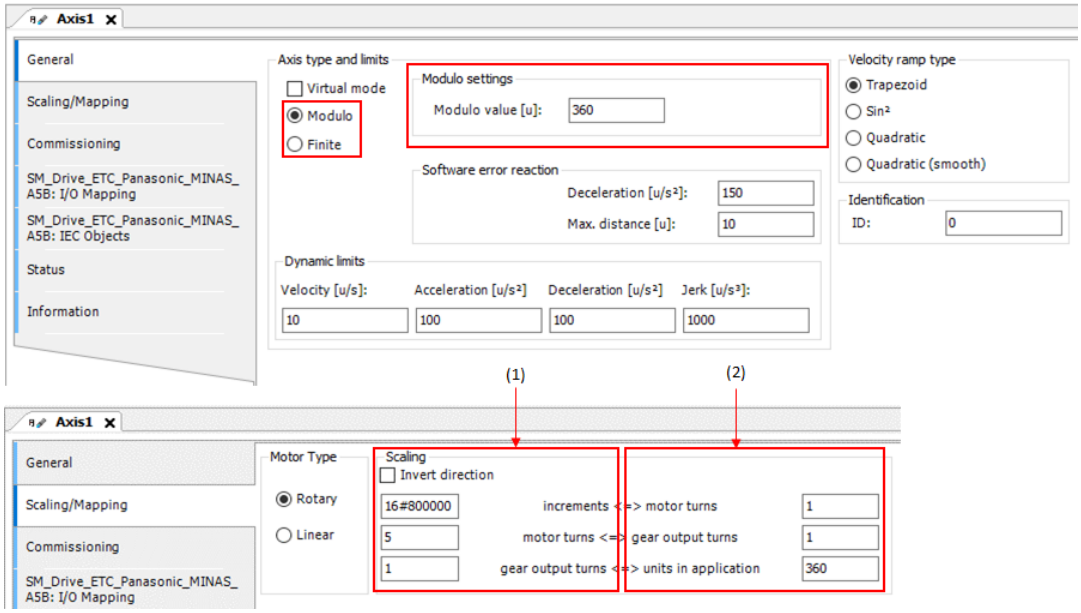
Info.

- When the execution is completed, the axis must be restarted by SMC3_ReinitDrived.

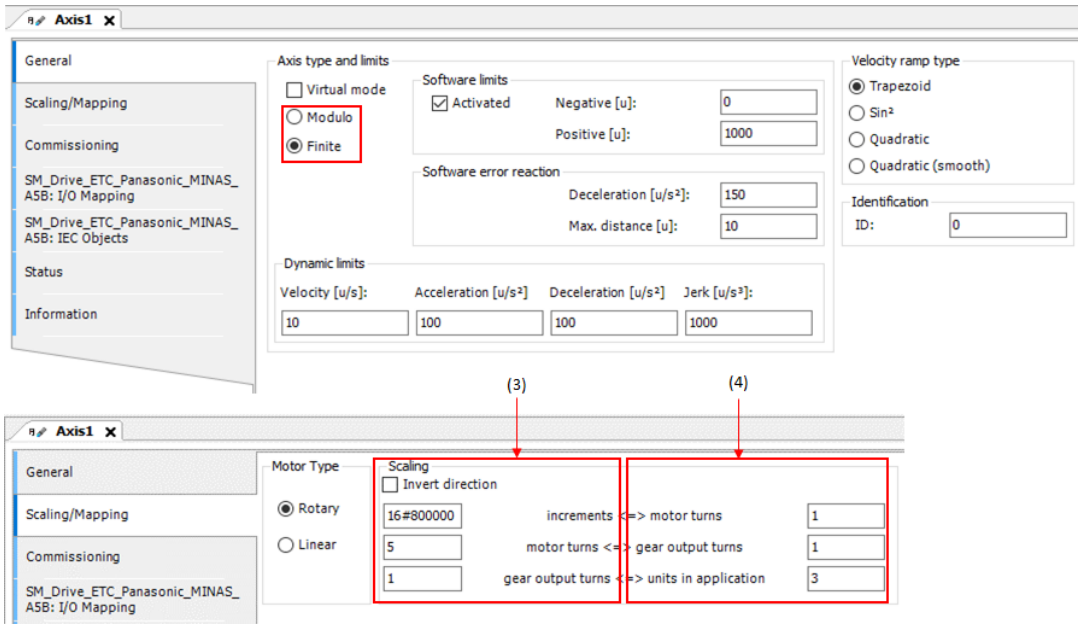
■ Setting location

The axis type, the modulo value and the scaling value in the figure below can be changed while GM1 is running.

- In the case of the modulo axis



- In the case of the finite axis



Info.

- When the GM1 is powered off, cold reset, or warm reset, it returns to the previous setting value.
- For scaling settings (gear ratio, encoder resolution), refer to the GM1 Series Reference Manual (Operation).
- When setting the modulo axis, set as follows.

```
dwRatioTechUnitsDenom    = "increments"×"motor turns"×"gear output turns" …in
the figure below(1)
                         = 16#800000 × 5 × 1 = 41943040
iRatioTechUnitsNum       = "motor turns"×"gear output turns"×"units in
application" …in the figure below(2)
                         = 1 × 1 × 360 = 360
fPositionPeriod          = the module value= 360
iMovementType            = 0(modulo)
```

- When setting the finite axis, set as follows.

```
dwRatioTechUnitsDenom    = "increments"×"motor turns"×"gear output turns" …in
the figure below(3)
                         = 16#800000 × 5 × 1 = 41943040
iRatioTechUnitsNum       = "motor turns"×"gear output turns"×"units in
application" …in the figure below(4)
                         = 1 × 1 × 3 = 3
iMovementType            = 1(finite)

fPositionPeriod is the same as the modulo value.
```

10.2.6 SMC_SetMovementType(Virtual axis type change)

This is a function block that changes the axis type and modulo value of a virtual axis.When the axis state is power_off or standstill,this FB can be used.

■ Icon

■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.(Note 1)
Input	bExecute	BOOL	FALSE	Starts execution at the rising edge.
	iMovementType	INT	0	Specify the axis type.0 is modulo.1 is finite.Other than error.
	fPositionPeriod	LREAL	1	the modululo value (Note 2)

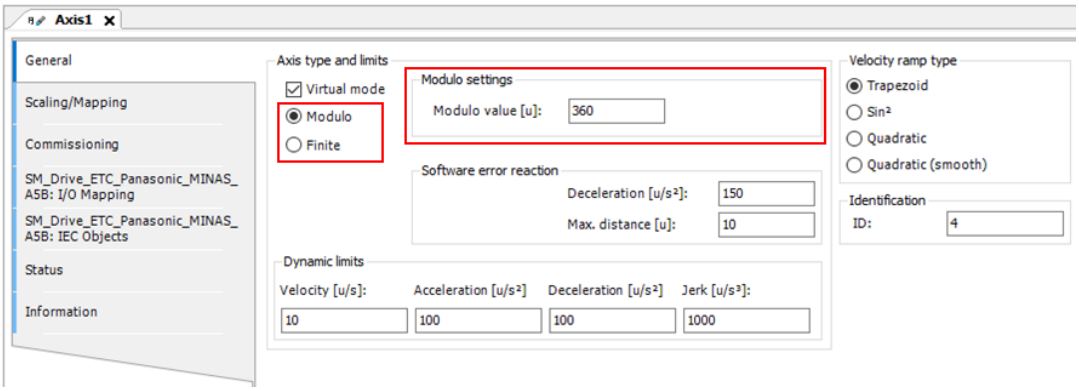
Scope	Name	Type	Initial	Description
Output	bDone	BOOL	FALSE	TRUE：The FB is in operation.
	bError	BOOL	FALSE	TRUE：An error has occurred.
	ErrorID	SMC_ERROR	SMC_NO_ERROR	An error ID is output.

(Note 1) Supports the virtual axes. If used on the real axis, an error will occur.
When using on real axis, enable virtual mode.

(Note 2) Set a positive value.

■ Setting location

The axis type, the modulo value and the scaling value in the figure below can be changed while GM1 is running.



Info.

- When the GM1 is powered off, cold reset, or warm reset, it returns to the previous setting value.
- For the axis type, refer to the GM1 Series Reference Manual (Operation).

10.2.7 SMC_SetRampType(Velocity ramp type change)

This is a function block that changes the velocity ramp type of the real and virtual axes.When the axis state is power_off or standstill,this FB can be used.

■ Icon

■ Parameter

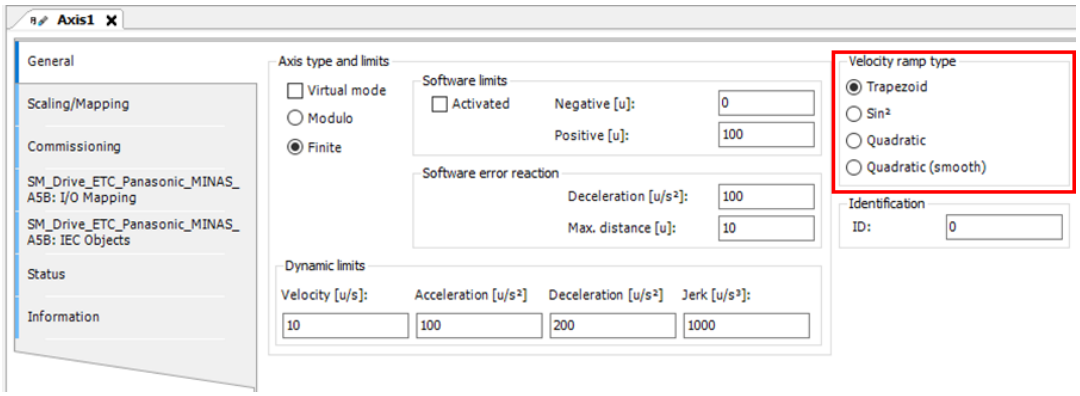
Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	bExecute	BOOL	FALSE	Starts execution at the rising edge.
	eRampType	SMC_RAMPTYPE	0	Specifies the velocity ramp type.
Output	bDone	BOOL	FALSE	TRUE：The FB is in operation.
	bError	BOOL	FALSE	TRUE：An error has occurred.
	ErrorID	SMC_ERROR	SMC_NO_ERROR	An error ID is output.

■ SMC_RAMPTYPE（Enumeration type）

Name	Value	Description
trapez	0	Trapezoid
sinsquare	1	sin^2
quadratic_ramp	2	Quadratic
quadratic_smooth_ramp	3	Quadratic (smooth)

■ Setting location

The velocity ramp type in the figure below can be changed while GM1 is running.



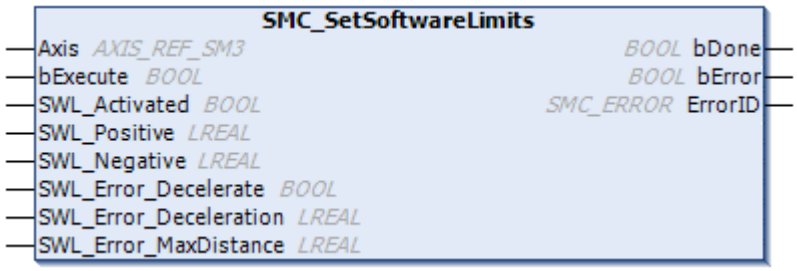
Info.

- When the GM1 is powered off, cold reset, or warm reset, it returns to the previous setting value.
- When CNC controlled by SMC_Interpolator, This FB has no effect.
- Check the GM1 Controller User's Manual (Operation Edition) for Axis operation specifications of the velocity ramp type.

10.2.8 SMC_SetSoftwareLimits(Soft limit change)

This is a function block that changes the enable / disable of soft limit of the real and virtual axes. It can be set without depending on the state of the axis.When the axis type is finite, the soft limit function is effective.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	bExecute	BOOL	FALSE	Starts execution at the rising edge.
	SWL_Activated	BOOL	FALSE	TRUE：Soft limit is valid. FALSE：Soft limit is invalid.
	SWL_Positive	LREAL	0	Soft limit for the positive direction[u] (Note 1)
	SWL_Negative	LREAL	0	Soft limit for negative direction[u] (Note 1)
	SWL_Error_Decelerate	BOOL	FALSE	Please do not use it.
	SWL_Error_Deceleration	LREAL	0	Deceleration when software error occurs[u /s²] (Note 2)
	SWL_Error_MaxDistance	LREAL	0	Maximum distance when a software error occurs[u] (Note 2)
Output	bDone	BOOL	FALSE	TRUE：The FB is in operation.
	bError	BOOL	FALSE	TRUE：An error has occurred.
	ErrorID	SMC_ERROR	SMC_NO_ERROR	An error ID is output.

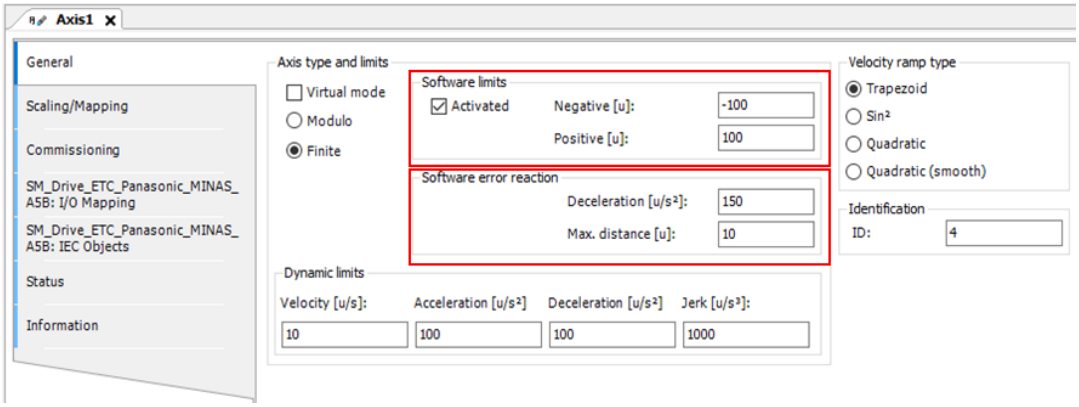
(Note 1) Set SWL_Positive to be larger than SWL_Negative.

(Note 2) Set a positive value.

When SWL_Error_Deceleration and WL_Error_MaxDistance are 0, Deceleration is fMaxDeceleration of SMC_ChangeDynamicLimits.

■ Setting location

The velocity ramp type in the figure below can be changed while GM1 is running.



Info.

- When the GM1 is powered off, cold reset, or warm reset, it returns to the previous setting value.
- For the soft limit, refer to the GM1 series reference manual (operation).

10.3 Motion Auxiliary Function (Other Functions)

10.3.1 PMC_ReadLatchPosition (Amplifier Latch Monitor)

This is a function block (FB) that monitor the AMP latch position. It reads the axis position when a trigger signal occurs.

■ **Icon**

■ **Parameter**

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	Execute	BOOL	FALSE	TRUE: Starts execution at the rising edge. FALSE: Stops processing.
	nStartCancelState	IoDRVRTEX.LATCH_STATE	MONITOR_LATCH	Specifies the start and cancellation of the latch mode.
	nLatchTrg1	IoDRVRTEX.LATCH_TRIGGER	Z_PHASE	Selects the trigger signal for latch position 1
	nLatchTrg2	IoDRVRTEX.LATCH_TRIGGER	-	Selects the trigger signal for latch position 2
	nMonitorSel	IoDRVRTEX.MONITOR_SELECT		Selects the latch position to be output as the output MonitorData.
Output	Done	BOOL	FALSE	TRUE: Output is completed.
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	CommandAborted	BOOL	FALSE	TRUE: An interruption from other FB has occurred.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	An error ID is output.
	MonitorData	LREAL		Axis position is output.
	bLatchComp1	BOOL	FALSE	TRUE : Latch completed at latch position 1 (CH1).

Scope	Name	Type	Initial	Description
	bLatchComp2	BOOL	FALSE	TRUE : Latch completed at latch position 2 (CH2).

■ IoDRVRTEX.LATCH_STATE (Enumeration type)

Name	Value	Description
MONITOR_LATCH	80	Monitors the position latch state. Monitors the state without newly starting or canceling.
START_LATCH1	81	Starts the position latch 1 (CH1).
START_LATCH2	82	Starts the position latch 2 (CH2).
START_LATCH1_AND2	83	Starts the position latch 1 (CH1) and position latch 2 (CH2).
CANCEL_LATCH1	84	Cancels the position latch 1 (CH1).
CANCEL_LATCH2	88	Cancels the position latch 2 (CH2).
CANCEL_LATCH1_AND2	92	Cancels the position latch 1 (CH1) and position latch 2 (CH2).

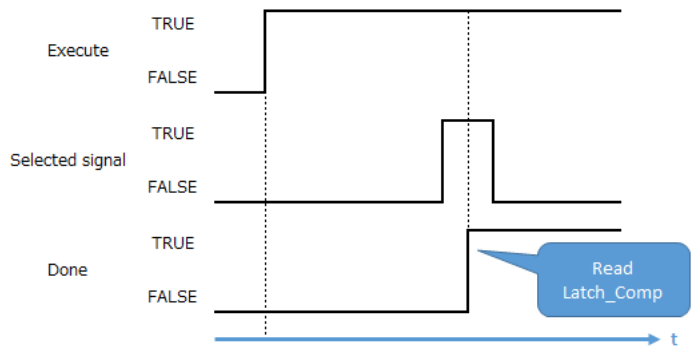
■ IoDRVRTEX.LATCH_TRIGGER (Enumeration type)

Name	Value	Description
Z_PHASE	0	Z phase
EXT1_RISING_EDGE	1	Rising edge of EXT1
EXT2_RISING_EDGE	2	Rising edge of EXT2
EXT3_RISING_EDGE	3	Rising edge of EXT3
PR7_111_RISING_EDGE	7	Not used for this FB.
EXT1_FALLING_EDGE	9	Falling edge of EXT1
EXT2_FALLING_EDGE	10	Falling edge of EXT2
EXT3_FALLING_EDGE	11	Falling edge of EXT3
PR7_111_FALLING_EDGE	15	Not used for this FB.

■ IoDRVRTEX.MONITOR_SELECT (Enumeration type)

Name	Value	Description
LPOS1	9	Latch position 1
LPOS2	10	Latch position 2

■ **Operations when the function block is executed**



The PMC_ReadLatchPosition function block outputs the following error.

Error	Description
SMC_WRONG_CONTROLLER_MODE	Executed in a mode other than the position control mode. Change to SMC_position using SMC_SetControllerMode.
SMC_RP_DRIVE_PARAMETER_NOT_MAPPED	Specified nLatchTrg1 and nLatchTrg2 to not use.
	Allocation of EXT1, EXT2, and EXT3 to the servo amplifier is faulty. Change the settings for Pr4.04 to Pr4.06.

As the PMC_ReadLatchPosition function block uses the RTEX home return command, it cannot be executed together with PMC_Home.

If PMC_ReadLatchPosition is executed while PMC_Home is being executed, the CommandAborted parameter of PMC_ReadLatchPosition becomes TRUE.

When using EXT1, EXT2, and EXT3 for nLatchTrg1 and nLatchTrg2, set amplifier parameters as shown in the following table.

Parameter	Parameter name	Settings
Pr4.04	SI5 input selection	EXT1
Pr4.05	SI6 input selection	EXT2
Pr4.06	SI7 input selection	EXT3

10.3.2 PMC_StopLatchPosition (Stop Amplifier Latch)

This is a function block (FB) that stops the axis at the AMP latch position. Stops the axis when a trigger event occurs.

■ Icon

■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
Input	Execute	BOOL	FALSE	TRUE: Starts execution at the rising edge. FALSE: Stops processing.
	nLatchTrg1	IoDRVRTEX.LATCH_TRIGGER	EXT1_RISING_EDGE	Selects the trigger signal for latch position
	Velocity	LREAL	0	Specifies the velocity (u/s).
	Acceleration	LREAL	0	Specifies the acceleration (u/s^2).
	Deceleration	LREAL	0	Specifies the deceleration (u/s^2).
	Jerk	LREAL	0	Specifies the jerk (u/s^3).
	Direction	MC_Direction	negative	Specifies the traveling direction of the axis.
Output	InVelocity	BOOL	FALSE	TRUE: The axis has reached the specified velocity for the first time.
	CommandAborted	BOOL	FALSE	TRUE: An interruption from other FB has occurred.
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	Done	BOOL	FALSE	TRUE: Stopping is completed.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	An error ID is output.
	MonitorData	LREAL		Axis position is output.

■ IoDRVRTEX.LATCH_TRIGGER (Enumeration type)

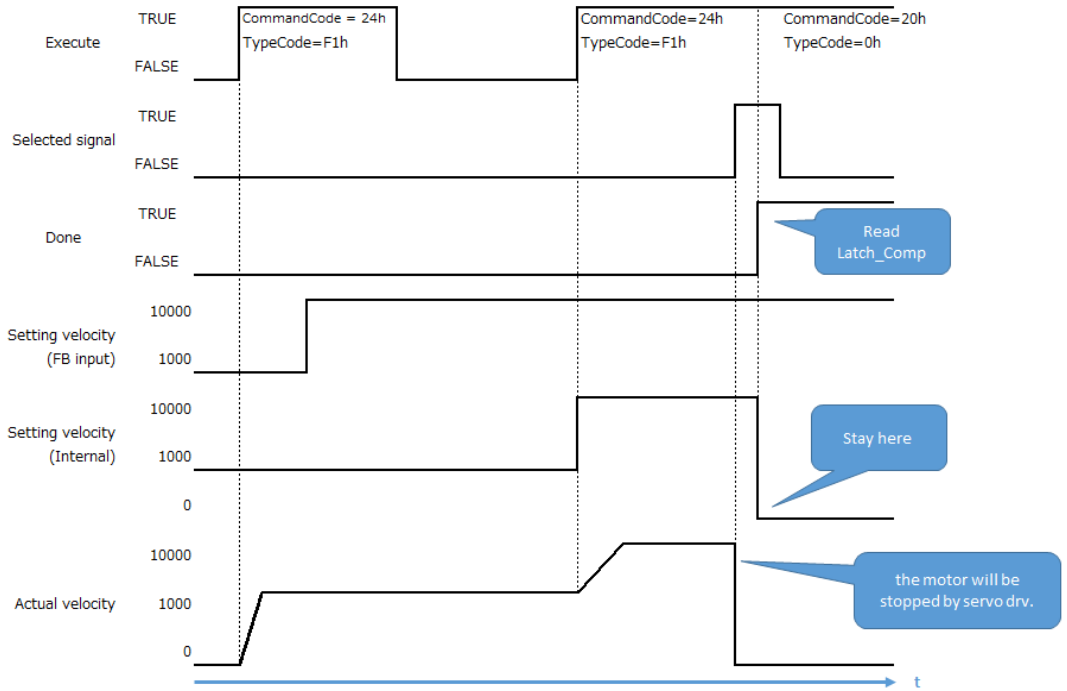
Name	Value	Description
Z_PHASE	0	Not used for this FB.
EXT1_RISING_EDGE	1	Rising edge of EXT1
EXT2_RISING_EDGE	2	Rising edge of EXT2

Name	Value	Description
EXT3_RISING_EDGE	3	Rising edge of EXT3
PR7_111_RISING_EDGE	7	Condition set by MINAS amplifier parameter Pr7.111
EXT1_FALLING_EDGE	9	Falling edge of EXT1
EXT2_FALLING_EDGE	10	Falling edge of EXT2
EXT3_FALLING_EDGE	11	Falling edge of EXT3
PR7_111_FALLING_EDGE	15	Condition set by MINAS amplifier parameter Pr7.111

■ MC_Direction (Enumeration type)

Name	Value	Description
positive	1	Travels in the positive direction.
negative	-1	Travels in the negative direction.
shortest	0	Not available. Do not specify this.
fastest	3	Not available. Do not specify this.
current	2	Travels to the current direction. Possible to use only for the modulo axis.

■ Operations when the function block is executed



- Execute = TRUE: Starts the latch mode. Execute = FALSE: Ends the latch, however, the axis operation continues as long as PMC_StopLatchPosition is called. Stop the axis using either MC_Stop or MC_Halt.

- When a trigger signal is input, the PMC_StopLatchPosition function block ignores the command value from the GM1 and stops at the latch position.

■ Execution errors

The PMC_StopLatchPosition function block outputs the following error.

Error	Description
SMC_WRONG_CONTROLLER_MODE	Executed in a mode other than the position control mode. Change to SMC_position using SMC_SetControllerMode.
SMC_RP_DRIVE_PARAMETER_NOT_MAPPED	Specified nLatchTrg1 to not use.
	Allocation of EXT1, EXT2, and EXT3 to the servo amplifier is faulty. Change the settings for Pr4.04 to Pr4.06.
SMC_DI_HOMING_ERROR	Servo amplifier version is lower than V1.24.
SMC_AXIS_NOT_READY_FOR_MOTION	The axis is in a state (Stopping, Disabled, or Errorstop) where PMC_StopLatchPosition cannot be executed.
SMC_REGULATOR_OR_START_NOT_SET	The servo OFF or brake is applied.
SMC_3SH_INVALID_VELACC_VALUES	The input (Velocity, Acceleration, or Deceleration) is faulty.
SMC_AXIS_REF_CHANGED_DURING_OPERATION	The Axis was changed during operation.

■ Execution conditions

- As the PMC_StopLatchPosition function block uses the RTEX home return command, it cannot be executed together with PMC_Home.
- To use the PMC_StopLatchPosition function block, the MINAS version must be V1.23 or higher.
- The function block supports only the control cycle of 1.0 ms and communication cycle of 0.5 ms.

■ Amplifier parameter conditions

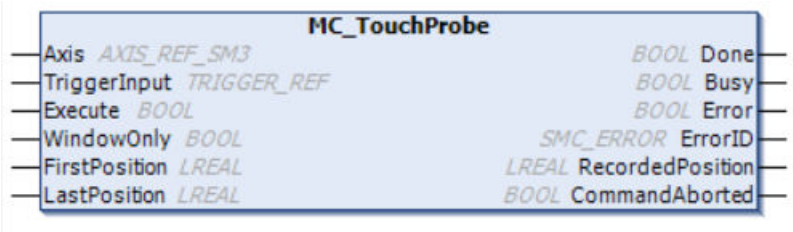
When using EXT1, EXT2, and EXT3 for nLatchTrg1, set amplifier parameters as shown in the following table.

Parameter	Parameter name	Settings
Pr4.04	SI5 input selection	EXT1
Pr4.05	SI6 input selection	EXT2
Pr4.06	SI7 input selection	EXT3

10.3.3 MC_TouchProbe (Enable AMP Latch Monitoring)

This a function block (FB) that reads the axis position when a trigger signal occurs.

■ Icon



■ Parameter

Scope	Name	Type	Default	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
	TriggerInput	TRIGGER_REF	0	Specifies the trigger signal.
Input	Execute	BOOL	FALSE	TRUE: Starts execution at the rising edge.
	WindowOnly	BOOL	FALSE	TRUE: The trigger event is accepted only in the specified window.
	FirstPosition	LREAL	0	The trigger event is accepted from the start position (in the positive direction). (Unit: [u])
	LastPosition	LREAL	0	The last position up to which the trigger event is accepted. (Unit: [u])
Output	Done	BOOL	FALSE	TRUE: Halt is completed.
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	Error ID output
	RecordedPosition	LREAL	0	The position where the trigger event has occurred. (Unit: [u])
	CommandAborted	BOOL	FALSE	TRUE: An interruption is caused by another FB.

■ TRIGGER_REF (Structure)

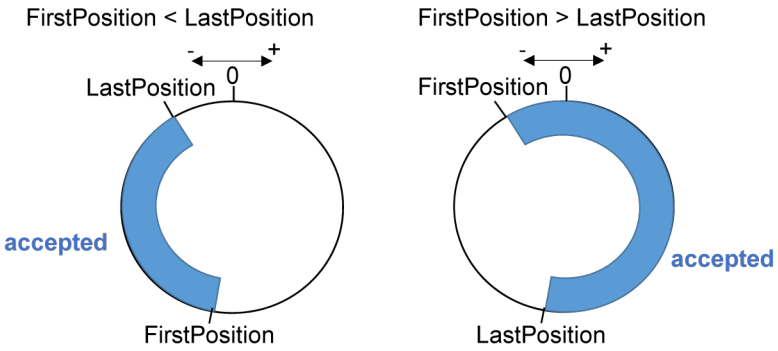
Member	Type	Default	Description
iTriggerNumber	INT	-1	Trigger channel: Defined by the driver. (Used only when bFastLatching is TRUE.)
bFastLatching	BOOL	TRUE	When bFastLatching is set to TRUE, latch is performed by the servo amplifier. When bFastLatching is set to FALSE, bInput is used as the trigger signal.
bInput	BOOL		When bFastLatching is set to FALSE, the trigger signal is input.
bActive	BOOL	FALSE	Internal variable. Do not set the value.

When using the MINAS, set iTriggerNumber as follows.

iTriggerNumber	Description
0	Rising edge of EXT1
1	Falling edge of EXT1
2	Rising edge of EXT2
3	Falling edge of EXT2

As for pin assignment, assign EXT1 to SI5 and EXT2 to SI6.

The range where the trigger event is accepted (WindowOnly) is as follows (in case of the modulo).



Note that hardware latch (EXT1 or EXT2 trigger) is not supported. Only software latch (bInput trigger) is supported.

10.3.4 MC_AbortTrigger (Disable AMP Latch Monitoring)

This is a function block (FB) that aborts the trigger event (MC_TouchProbe).

■ Icon

■ Parameter

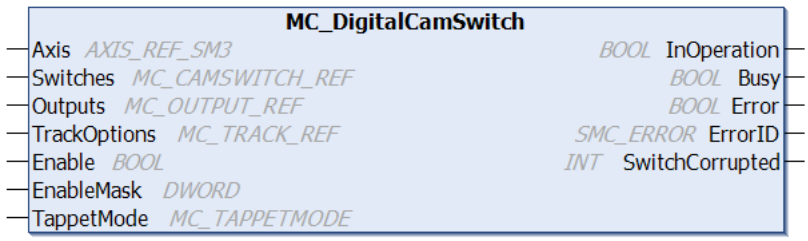
Scope	Name	Type	Default	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis.
	TriggerInput	TRIGGER_REF	0	Specifies the trigger signal.
Input	Execute	BOOL	FALSE	TRUE: Starts execution at the rising edge.
Output	Done	BOOL	FALSE	TRUE: Stopping is completed.
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.

Scope	Name	Type	Default	Description
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.
	ErrorID	SMC_ERROR	0	Error ID output

10.3.5 MC_DigitalCamSwitch (Enable Digital Cam Switch)

This is a function block (FB) that performs ON / OFF control on the digital output according to the axis position. It assigns digital cam switches to tracks (maximum of 32). Switching operations can be controlled by specifying the ON / OFF position for each digital camp switch.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Axis	AXIS_REF_SM3	-	Specifies the axis where the switch is connected.
	Switches	MC_CAMSWITCH_REF	-	Specifies the switching operation.
	Outputs	MC_OUTPUT_REF	-	ON or OFF of the switch is output. ARRAY [1..32] OF BOOL
	TrackOptions	MC_TRACK_REF	-	Specifies the property of the track. ARRAY [1..32] OF MC_TRACK_TR
Input	Enable	BOOL	FALSE	TRUE: The FB can be executed.
	EnableMask	DWORD	16#FFFFFFFF	Specifies the track to be enabled. 1: Enabled, 0: Disabled The least significant bit is the 1st track. The most significant bit is the 32nd track.
	TappetMode	MC_TAPPETMODE	tp_mode_auto	Specifies the tappet mode.
Output	InOperation	BOOL	FALSE	TRUE: The track is enabled.
	Busy	BOOL	FALSE	TRUE: Execution of the FB is not completed.
	Error	BOOL	FALSE	TRUE: An error has occurred within the FB.

Scope	Name	Type	Initial	Description
	ErrorID	SMC_ERROR	0	An error ID is output.
	SwitchCorrupted	INT	-1	Index output of a faulty switch -1: No problem 0 to 31: A problem has occurred in switches 1 to 32.

■ MC_CAMSWITCH_REF (Structure)

Member	Type	Description
NoOfSwitches	BYTE	Number of switches Specifies the number of switches to be enabled when the FB is executed in the MC_CAMSWITCH_TR type array (1 to 32).
CamSwitchPtr	POINTER TO MC_CAMSWITCH_TR	Pointer to the first element of the MC_CAMSWITCH_TR type array

■ MC_CAMSWITCH_TR (Structure)

Member	Type	Description
TrackNumber	INT	Switch track number (1 to 32)
FirstOnPosition	LREAL	Position where the switch turns ON when the axis is moving in the positive direction
LastOnPosition	LREAL	Position where the switch turns OFF when the axis is moving in the positive direction Not used when CamSwitchMode is set to 1.
AxisDirection	INT	Movement direction where the switch is enabled 0: Both positive and negative directions 1: Only positive direction 2: Only negative direction
CamSwitchMode	INT	Control method that performs switch ON / OFF control 0: ON and OFF are both controlled by the position. 1: ON is controlled by the position and OFF is controlled by the time.
Duration	TIME	Specifies the time during which the switch remains ON for when CamSwitchMode is set to 1.
bOn	BOOL	Used within the FB.
CounterOff	INT	Used within the FB.

■ MC_TRACK_REF (Structure)

Member	Type	Description
OnCompensation	LREAL	Specifies the switch ON delay time in seconds. When a positive value is specified, the switch turns ON later by the time specified. When a negative value is specified, the switch turns ON earlier by the time specified.
OffCompensation	LREAL	Specifies the switch OFF delay time in seconds.

Member	Type	Description
		When a positive value is specified, the switch turns OFF after a delay of the time specified. When a negative value is specified, the switch turns OFF earlier by the time specified.
Hysteresis	LREAL	Specifies the hysteresis value (position).

■ MC_TAPPETMODE (Enumeration type)

Name	Value	Description
tp_mode_auto	0	Automatically determined according to the state. Servo ON state: Command position (fSetPosition) of the master axis Servo OFF state: Actual position (fActPosition) of the master axis
tp_mode_demandposition	1	Command state (fSetPosition) of the master axis
tp_mode_actualposition	2	Actual state (fActPosition) of the master axis

Regarding the method for entering defaults for variables of the MC_CAMSWITCH_TR type structure, refer to “Default Setting for Variables of the MC_TP_REF Type Structure”.

■ Operations when the function block is executed

The following sections shows switching operations (Outputs) of each track when the function block is executed after setting the parameter as follows. The axis is set to the modulo (modulo value: 1000).

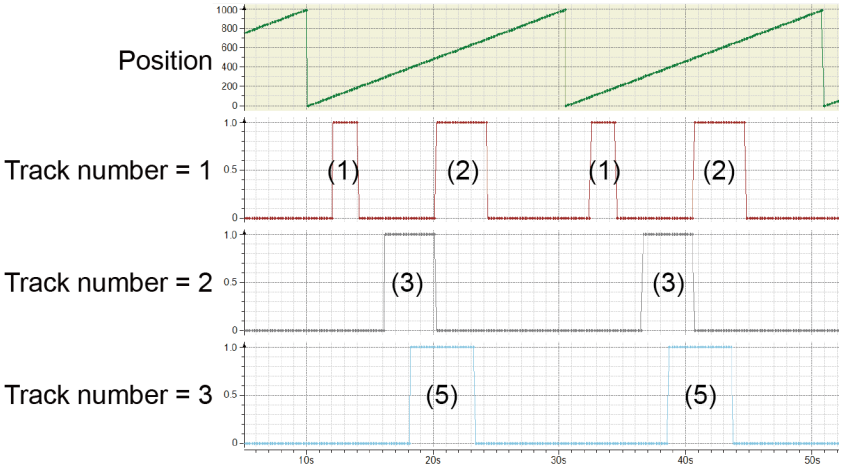
Function block input parameters

Five switches (CamSwitchPtr) are set.

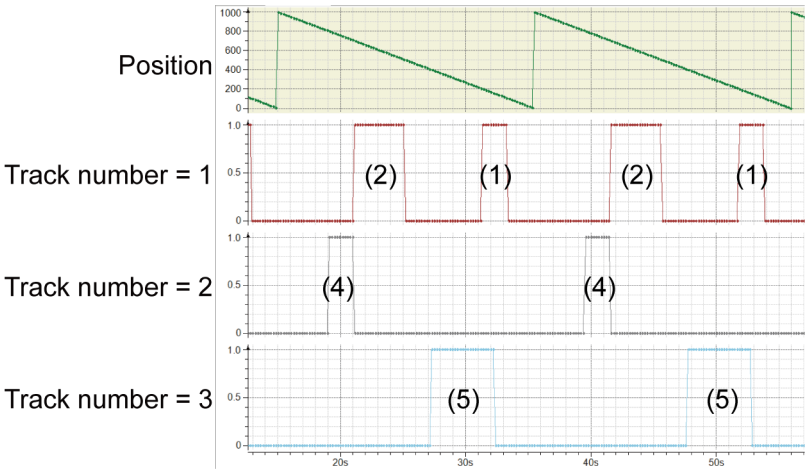
Switch	Index	Track Number	FirstOn Position	LastOn Position	Axis Direction	Cam SwitchMode	Duration
(1)	1	1	100	200	0 (Both)	0 (Position)	T#0ms
(2)	2	1	500	700	0 (Both)	0 (Position)	T#0ms
(3)	3	2	300	500	1 (Positive direction)	0 (Position)	T#0ms
(4)	4	2	700	800	2 (Negative direction)	0 (Position)	T#0ms
(5)	5	3	400	0	0 (Both)	1 (Time)	T#5s

Switching operations when the axis is moved in the positive direction

(1) to (5) are switch numbers.



Switching operations when the axis is moved in the negative direction



■ **Detection of faulty switch operation (SwitchCorrupted)**

SwitchCorrupted occurs when the switch does not turn ON/OFF as set.

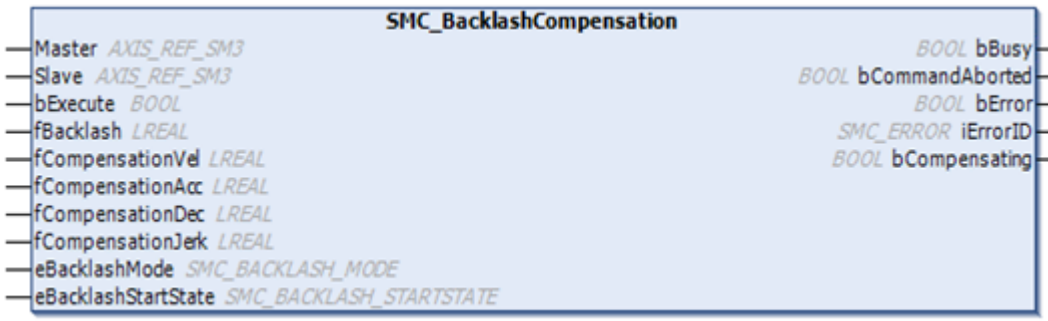
REFERENCE

5.6.6 Default Setting for Variables of the MC_TP_REF Type Structure

10.3.6 SMC_BacklashCompensation (Compensate Backlash)

This is a function block (FB) that compensates the backlash.

■ Icon



■ Parameter

Scope	Name	Type	Initial	Description
Input / output	Master	AXIS_REF_SM3	-	Specifies the master axis.
	Slave	AXIS_REF_SM3	-	Specifies the slave axis.
Input	bExecute	BOOL	FALSE	Starts execution at the rising edge. Remains enabled until the slave axis is interrupted by another operation or until an error occurs.
	fBacklash	LREAL	0	Distance to compensate (backlash)
	fCompensationVel	LREAL	0	Additional velocity used when compensation is performed (A value to be added to the master axis velocity)
	fCompensationAcc	LREAL	0	Additional acceleration used when compensation is performed (A value to be the maximum acceleration when compensation is performed)
	fCompensationDec	LREAL	0	Additional deceleration used when compensation is performed. (A value to be the maximum deceleration when compensation is performed)
	fCompensationJerk	LREAL	0	Additional jerk used when compensation is performed (Even if any value is set, the setting is disabled.)
	eBacklashMode	SMC_BACKLASH_MODE	SMC_BL_AUTO	Backlash compensation mode
	eBacklashStartState	SMC_BACKLASH_STARTSTATE	SMC_BL_START_NONE	Specifies the start conditions whether compensation is required or not when starting the backlash compensation.
Output	bBusy	BOOL	FALSE	TRUE: The FB is in operation.
	bCommandAborted	BOOL	FALSE	TRUE: An interruption from other FB has occurred.

Scope	Name	Type	Initial	Description
	bError	BOOL	FALSE	TRUE: An error has occurred within the FB.
	iErrorID	SMC_ERROR	0	An error ID is output.
	bCompensating	BOOL	FALSE	TRUE: Backlash compensation in operation

■ MC_BACKLASH_MODE (Enumeration type)

Name	Value	Description
SMC_BL_AUTO	2	Compensation in the traveling direction of the master axis
SMC_BL_POSITIVE	1	Compensation in the positive direction
SMC_BL_NEGATIVE	-1	Compensation in the negative direction
SMC_BL_OFF	0	No backlash compensation

■ SMC_BACKLASH_STARTSTATE (Enumeration type)

Name	Value	Description
SMC_BL_START_NEGATIVE	-1	If the slave axis is driven in the negative direction when compensation is started: ● To make the axis travel in the positive direction, compensation is required for the backlash distance (fBacklash). ● No compensation is required for the travels in the negative direction.
SMC_BL_START_NONE	0	If the slave axis is not driven in either direction when compensation is started: To make the axis travel in the positive or negative direction, compensation is required for half the amount of the backlash distance (fBacklash).
SMC_BL_START_POSITIVE	1	If the slave axis is driven in the positive direction when compensation is started: ● No compensation is required for the travels in the positive direction. ● To make the axis travel in the negative direction, compensation is required for the backlash distance (fBacklash).

- When starting operation, make sure that both the master axis and slave axis are in the same position. If they are not set at the same position, the slave axis travels to the master axis position at the moment when SMC_BacklashCompensation is executed.
- SMC_BacklashCompensation functions in the same way as the phase synchronous operation (MC_Phasing) and the phase depends on the master axis direction.

11 Other Function Blocks

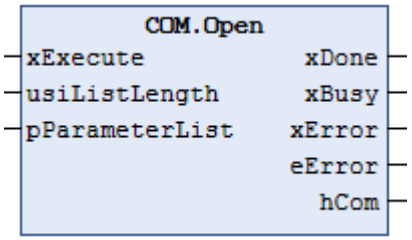
11.1 COM Port (General-purpose Communication)

This section describes function blocks that are used to perform general-purpose communication with the COM port.

11.1.1 COM.Open (Open COM port)

This is a function block that opens a COM port. It reads from and writes to the COM port using the output handle. Close the opened COM port using the COM.Close instruction.

■ Icon



■ Parameter

Scope	Name	Type	Description
Input	xExecute	BOOL	Starts execution at the rising edge.
	usiListLength	USINT	Number of pParameterList entries
	pParameterList	COM.CAA.PVOID	A pointer to the communication setting parameter list for the COM port. Specifies the pointer to the COM.PARAMETER structure array.
Output	xDone	BOOL	TRUE: Execution is completed.
	xBusy	BOOL	TRUE: Execution of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	eError	COM.ERROR	An error ID is output. Refer to "11.1.5 COM.ERROR (Error ID)".
	hCom	COM.CAA.HANDLE	Handle of the opened COM port.

■ COM.PARAMETER (Structure)

Member	Type	Description
udiParameterId	UDINT	Parameter ID to be set in the COM port. For a list of parameters, refer to "COM.CAA_Parameter_Constants (Constants)".
udiValue	UDINT	Value to be set in the COM port

■ COM.CAA_Parameter_Constants (Constants)

Name	Value	Support	Description
udiPort	16#1	Supported	Port number (Fixed to 1.)
udiStopBits	16#2	Supported	Stop bit Refer to "COM.STOPBIT (Enumeration type)".
udiParity	16#3	Supported	Parity Refer to "COM.PARITY (Enumeration type)".
udiBaudrate	16#4	Supported	Baud rate (Can be selected from 9600, 19200, 38400, 57600, and 115200)
udiTimeout	16#5	Not supported	Timeout
udiBufferSize	16#6	Not supported	Buffer size parameter Specifies a serial buffer size.
udiByteSize	16#7	Supported	Byte size parameter Sets the number of data bits to 4 to 8. (Specify 7 or 8 for the GM1 Controller.)
udiBinary	16#8	Not supported	Binary parameter Enables the binary mode. (With the GM1 Controller, it is fixed to 0 (binary mode).)
udiOutxCtsFlow (Note 1)	16#9	Not supported	CTS handshake for the output parameter
udiOutxDsrFlow (Note 1)	16#A	Not supported	DSR handshake for the output parameter
udiDtrControl (Note 1)	16#B	Not supported	DTR flow control parameter
udiDsrSensitivity (Note 1)	16#C	Not supported	DSR sensitivity parameter
udiRtsControl (Note 1)	16#D	Not supported	Rts flow control parameter
udiTXContinueOnXoff (Note 1)	16#E	Not supported	XOFF continues Tx parameter.
udiOutX (Note 1)	16#F	Not supported	XON / XOFF output flow control parameter
udiInX (Note 1)	16#10	Not supported	XON / XOFF of the flow control parameter
udiXonChar (Note 1)	16#11	Not supported	Tx AND Rx XON character parameter
udiXoffChar (Note 1)	16#12	Not supported	Tx AND Rx XOFF character parameter
udiXonLim (Note 1)	16#13	Not supported	Sends XON threshold parameter
udiXoffLim (Note 1)	16#14	Not supported	Sends XOFF threshold parameter

(Note 1) The GM1 Controller does not support the flow control.

■ COM.STOPBIT (Enumeration type)

Name	Value	Description
ONESTOPBIT	0	1 stop bit
ONE5STOPBITS	1	1.5 stop bit (Not available)
TWOSTOPBITS	2	2 stop bit

■ COM.PARITY (Enumeration type)

Name	Value	Description
EVEN	0	Even
ODD	1	Odd
NONE	2	None

■ Program example

Declaration section of ST program

```
Open : COM.Open;
OpenParam : ARRAY [1..7] OF COM.PARAMETER := [
    (udiParameterId := COM.CAA_Parameter_Constants.udiPort,     udiValue := 2),
    (udiParameterId := COM.CAA_Parameter_Constants.udiBaudrate, udiValue := 115200),
    (udiParameterId := COM.CAA_Parameter_Constants.udiParity,   udiValue := INT_TO_UDINT(COM.PARITY.ODD)),
    (udiParameterId := COM.CAA_Parameter_Constants.udiStopBits, udiValue := INT_TO_UDINT(COM.STOPBIT.ONESTOPBIT)),
    (udiParameterId := COM.CAA_Parameter_Constants.udiTimeout,  udiValue := 0),
    (udiParameterId := COM.CAA_Parameter_Constants.udiByteSize, udiValue := 8),
    (udiParameterId := COM.CAA_Parameter_Constants.udiBinary,   udiValue := 1)
];
```

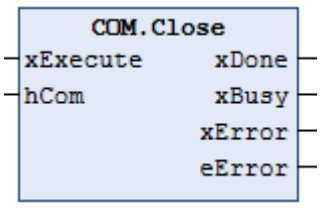
Implementation section of ST program

```
Open( xExecute:=TRUE , pParameterList:=ADR(OpenParam) , usiListLength:=SIZEOF(OpenParam)/SIZEOF(COM.PARAMETER) );
```

11.1.2 COM.Close (Close COM Port)

This is a function block that closes the COM port.

■ Icon



■ Parameter

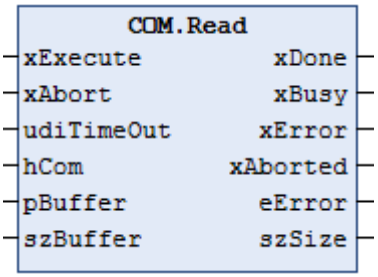
Scope	Name	Type	Description
Input	xExecute	BOOL	Starts execution at the rising edge.

Scope	Name	Type	Description
	hCom	COM.CAA.HANDLE	Handle of the COM port to be closed
Output	xDone	BOOL	TRUE: Execution is completed.
	xBusy	BOOL	TRUE: Execution of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	eError	COM.ERROR	An error ID is output. Refer to "11.1.5 COM.ERROR (Error ID)".

11.1.3 COM.Read (Read COM Port)

This is a function block that reads data from the COM port.

■ Icon



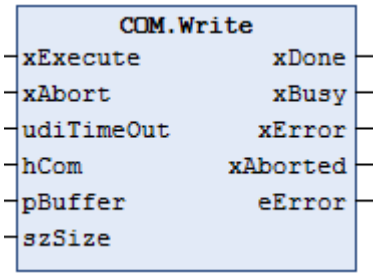
■ Parameter

Scope	Name	Type	Description
Input	xExecute	BOOL	Starts execution at the rising edge.
	xAbort	BOOL	TRUE: Stops execution and resets all outputs.
	udiTimeOut	UDINT	Timeout time until the execution is stopped (μs)
	hCom	COM.CAA.HANDLE	Handle of the COM port
	pBuffer	CAA.PVOID	Pointer to the buffer that acquires data read from the COM port
	szBuffer	CAA.SIZE	Maximum byte of pBuffer
Output	xDone	BOOL	TRUE: Execution is completed.
	xBusy	BOOL	TRUE: Execution of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	xAborted	BOOL	TRUE: Execution is stopped by the user.
	eError	COM.ERROR	An error ID is output. Refer to "11.1.5 COM.ERROR (Error ID)".
	szSize	COM.CAA.SIZE	Data size (bytes) acquired by the pBuffer

11.1.4 COM.Write (Write COM Port)

This is a function block that writes data to the COM port.

■ Icon



■ Parameter

Scope	Name	Type	Description
Input	xExecute	BOOL	Starts execution at the rising edge.
	xAbort	BOOL	TRUE: Stops execution and resets all outputs.
	udiTimeOut	UDINT	Timeout time until the execution is stopped (μs)
	hCom	COM.CAA.HANDLE	Handle of the COM port
	pBuffer	CAA.PVOID	Pointer to the buffer of the data written to the COM port
	szSize	COM.CAA.SIZE	Data size (bytes) of the pBuffer to be written to the COM port
Output	xDone	BOOL	TRUE: Execution is completed.
	xBusy	BOOL	TRUE: Execution of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	xAborted	BOOL	TRUE: Execution is stopped by the user.
	eError	COM.ERROR	An error ID is output. Refer to "11.1.5 COM.ERROR (Error ID)".

11.1.5 COM.ERROR (Error ID)

This is an enumeration type error ID that is output when the COM port (general-purpose communication) function block is executed.

■ COM .ERROR (Enumeration type)

Name	Value	Description
NO_ERROR	0	No error
TIME_OUT	5001	Timeout error
ABORT	5002	xAbort input enabled

Name	Value	Description
HANDLE_INVALID	5003	Invalid handle
ERROR_UNKNOWN	5004	Unknown error
WRONG_PARAMETER	5005	Wrong parameter
WRITE_INCOMPLETE	5006	Incomplete write

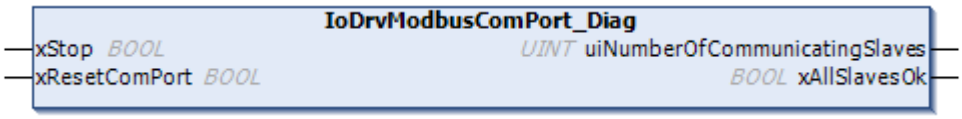
11.2 COM port (Modbus COM)

This section describes the instructions that are used to perform ModbusRTU communication with the COM port.

11.2.1 IoDrvModbusComPort

This is a function block that controls the Modbus_Master_COM_Port device.

■ **Icon**



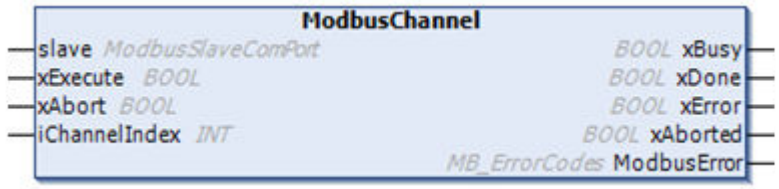
■ **Parameter**

Scope	Name	Type	Description
Input	xStop	BOOL	TRUE: Stops sending a new request to the slave. FALSE: Continues the current request.
	xResetComPort	BOOL	Closes the COM port at a rising edge.
Output	uiNumberOfCommunicatingSlaves	UINT	Number of remote slaves under communication.
	xAllSlavesOk	BOOL	TRUE: All slaves are communicating normally. FALSE: An error has occurred in one of the slaves.

11.2.2 IoDrvModbus.ModbusChannel(Start Sending Modbus Command)

This is a function block that sends the commands set in the Modbus Slave channel of the ModbusSlaveCOM_Port device.

■ **Icon**



■ **Parameter**

Scope	Name	Type	Description
Input	xExecute	BOOL	Starts sending commands at the rising edge.
	xAbort	BOOL	TRUE: Stops execution and resets all outputs.

Scope	Name	Type	Description
	iChannelIndex	INT	Channel number where commands to be sent are set
I/O	slave	ModbusSlaveComPort	Handle of the ModbusSlaveComPort device
Output	xBusy	BOOL	TRUE: Processing of the FB is not completed.
	xDone	BOOL	TRUE: Processing is completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	xAborted	BOOL	TRUE: Execution is stopped by the user's xAbort input.
	ModbusError	MB_ErrorCodes	An error code is output. Refer to "11.5.5 IoDrvModbus.MB_ErrorCodes (Error Codes)".

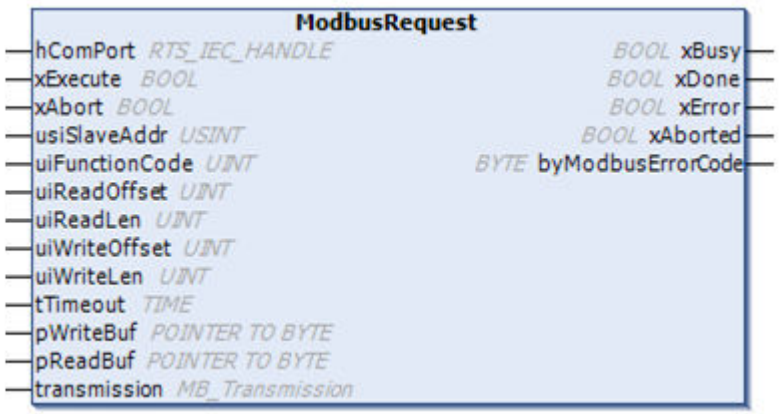
11.2.3 IoDrvModbus.ModbusRequest (Modbus Request)

This is a function block that processes the Modbus command specified by I/O without using the ModbusMasterComPort device.

■ Supported commands

- Command 1 (Read multi-point coil state)
- Command 2 (Read multi-point input state)
- Command 3 (Read multi-point holding register)
- Command 4 (Read multi-point input register)
- Command 5 (Write single-point coil)
- Command 6 (Write single-point holding register)
- Command 15 (Write multi-point coil)
- Command 16 (Write multi-point holding register)
- Command 23 (Read / write multi-point holding register)

■ Icon



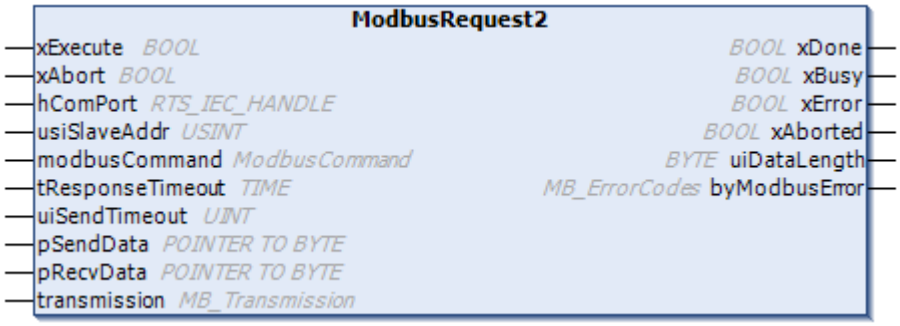
■ Parameter

Scope	Name	Type	Description
Input	hComPort	RTS_IEC_HANDLE	COM port handle acquired by COM.Open
	xExecute	BOOL	Starts sending commands at the rising edge.
	xAbort	BOOL	TRUE: Stops execution and resets all outputs.
	usiSlaveAddr	USINT	Slave address 1 to 247
	uiFunctionCode	UINT	Modbus function code
	uiReadOffset	UINT	Read address offset (0 to 65535)
	uiReadLen	UINT	Read length (1 to 125)
	uiWriteOffset	UINT	Write address offset (0 to 65535)
	uiWriteLen	UINT	Write length (1 to 121)
	tTimeout	UINT	Timeout value (in ms units)
	pWriteBuf	POINTER TO BYTE	Pointer to the send buffer.
	pReadBuf	POINTER TO BYTE	Pointer to the receive buffer
	transmission	MB_Transmission	Transmission type (RTU / ASCII) * Supports only RTU.
Output	xBusy	BOOL	TRUE: Processing of the FB is not completed.
	xDone	BOOL	TRUE: Processing is completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	xAborted	BOOL	TRUE: Execution is stopped by the user's xAbort input.
	byModbusErrorCode	BYTE	An error code is output. Refer to "11.2.6 IoDrvModbus.MB_ErrorCodes (Error Codes)".

11.2.4 IoDrvModbus.ModbusRequest 2 (Modbus Request 2)

This is a function block that processes, like the ModbusRequest, the Modbus command specified by I/O without using the ModbusMasterComPort device. It is different from ModbusRequest in that the structure type is used to specify the Modbus command.

■ Icon



■ Parameter

Scope	Name	Type	Description
Input	xExecute	BOOL	Starts sending commands at the rising edge.
	xAbort	BOOL	TRUE: Stops execution and resets all outputs.
	hComPort	RTS_IEC_HANDLE	COM port handle acquired by COM.Open
	usiSlaveAddr	USINT	Slave address 1 to 247
	modbusCommand	ModbusCommnad	Modbus command
	tResponseTimeout	TIME	Timeout (in ms units) of the response for a request
	uiSendTimeout	UINT	Transmission timeout
	pSendData	UINT	Pointer to the send data
	pRecvData	UINT	Pointer to the receive data
	transmission	MB_Transmission	Transmission type (RTU / ASCII) * Supports only RTU.
Output	xDone	BOOL	TRUE: Processing is completed.
	xBusy	BOOL	TRUE: Processing of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	xAborted	BOOL	TRUE: Execution is stopped by the user's xAbort input.
	uiDataLength	BYTE	Received data length (byte)
	byModbusError	MB_ErrorCodes	An error code is output. Refer to "11.2.6 IoDrvModbus.MB_ErrorCodes (Error Codes)".

■ ModbusCommand (Structure)

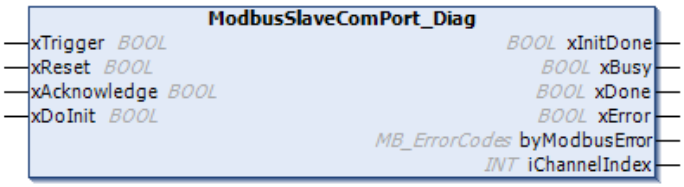
Name	Type	Description
uiFunctionCode	UINT	Modbus command code
uiReadOffset	UINT	Read address 0 to 65535

Name	Type	Description
uiReadLen	UINT	Range in the number of read instances varies depending on commands.
uiWriteOffset	UINT	Write address 0 to 65535
uiWriteLen	UINT	Range in the number of write instances varies depending on commands.

11.2.5 IoDrvModbus.ModbusSlaveComPort

This is a function block that controls the Modbus_Slave_COM_Port device.

■ Icon



■ Parameter

Scope	Name	Type	Description
Input	xTrigger	BOOL	Sends all the commands of the Modbus channel at the rising edge.
	xReset	BOOL	Resets xError and byModbusError and resumes communication.
	xAcknowledge	BOOL	Resumes communication without resetting xError and byModbusError.
	xDoInit	BOOL	TRUE: Sends a slave initialization command when communication is resumed.
Output	xInitDone	BOOL	TRUE: Modbus slave initialization command is fully completed.
	xBusy	BOOL	TRUE: Processing of the FB is not completed.
	xDone	BOOL	TRUE: Processing is completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	byModbusError	MB_ErrorCodes	An error code is output. Refer to "11.2.6 IoDrvModbus.MB_ErrorCodes (Error Codes)".
	iChannelIndex	INT	Channel index

11.2.6 IoDrvModbus.MB_ErrorCodes (Error Codes)

This is an enumeration type error code that is output when the function block for Modbus communication instruction that uses the COM port is executed.

■ IoDrvModbus.MB_ErrorCodes (Enumeration type)

Name	Value	Description
RESPONSE_SUCCESS	16#0	Succeeded
ILLEGAL_FUNCTION	16#1	Function code not supported by the slave
ILLEGAL_DATA_ADDRESS	16#2	Register offset not supported by the slave
ILLEGAL_DATA_VALUE	16#3	Illegal data writing
SLAVE_DEVICE_FAILURE	16#4	Non-recoverable error
ACKNOWLEDGE	16#5	Start operation
SLAVE_DEVICE_BUSY	16#6	During operation
MEMORY_PARITY_ERROR	16#8	Memory parity error
GATEWAY_PATH_UNAVAILABLE	16#A	Gateway path unavailable
GATEWAY_DEVICE_FAILED_TO_RESPOND	16#B	Gateway device failed to respond
RESPONSE_TIMEOUT	16#A1	Timeout
RESPONSE_CRC_FAIL	16#A2	CRC error
RESPONSE_WRONG_SLAVE	16#A3	Wrong response
RESPONSE_WRONG_FUNCTIONCODE	16#A4	Wrong function code in the response
REQUEST_FAILED_TO_SEND	16#A5	Request not sent
RESPONSE_INVALID_DATA	16#A6	Invalid response data
RESPONSE_INVALID_PROTOCOL	16#A7	Invalid response protocol
RESPONSE_INVALID_HEADER	16#A8	Invalid response header
UNDEFINED	16#FF	Undefined

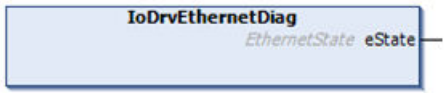
11.3 LAN port (IoDrvEthernet)

This section describes the library functions that are used for the network interface to perform communication with the LAN port.

11.3.1 IoDrvEthernet

This is a function block that acquires the status of the LANPort device.

■ Icon



■ Parameter

Scope	Name	Type	Description
Output	eState	EthernetState	Ethernet state Refer to "EthernetState (Enumeration type)".

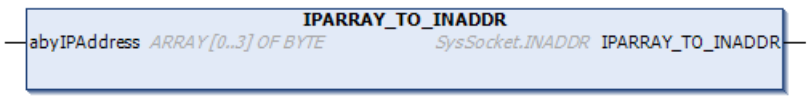
■ EthernetState (Enumeration type)

Name	Value	Description
NOT_CONFIGURED	0	Before configuration
CONFIGURED	1	After configuration
DISCONNECTED	2	Disconnected
RUNNING	3	Being executed
ERROR	4	An error has occurred.
SET_IP_ERROR	5	An IP error has occurred.

11.3.2 IoDrvEthernet.IPARRAY_TO_INADDR (Array Type to Union Type)

This is a function that converts an array type IP address to an INADDR (union type).

■ Icon



■ Parameter

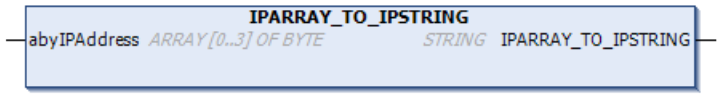
Scope	Name	Type	Description
Input	abyIPAddress	ARRY[0..3] OF BYTE	IP address array

Scope	Name	Type	Description
Output	IPARRAY_TO_INADDR	SysSocket.INADDR	Union type IP address

11.3.3 IoDrvEthernet.IPARRAY_TO_IPSTRING (Array Type to Character String Type)

This is a function that converts an array type IP address to a character string type.

■ Icon



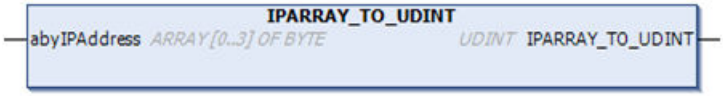
■ Parameter

Scope	Name	Type	Description
Input	abyIPAddress	ARRY[0..3] OF BYTE	IP address array
Output	IPARRAY_TO_STRING	STRING	Character string type IP address

11.3.4 IoDrvEthernet.IPARRAY_TO_UDINT (Array Type to UDINT Type)

This is a function that converts an array type IP address to a UDINT type.

■ Icon



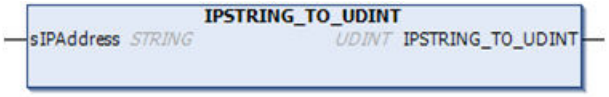
■ Parameter

Scope	Name	Type	Description
Input	abyIPAddress	ARRY[0..3] OF BYTE	IP address array
Output	IPARRAY_TO_UDINT	UDINT	UDINT type IP address

11.3.5 IoDrvEthernet.IPSTRING_TO_UDINT (Character String Type to UDINT Type)

This is a function that converts a character string type IP address to a UDINT type.

■ Icon



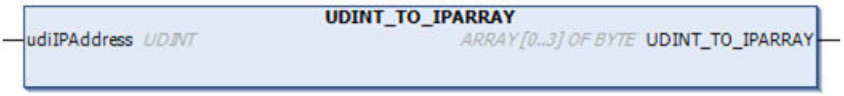
■ Parameter

Scope	Name	Type	Description
Input	abyIPAddress	STRING	Character string type IP address
Output	IPARRAY_TO_UDINT	UDINT	UDINT type IP address

11.3.6 IoDrvEthernet.UDINT_TO_IPARRAY (UDINT Type to Array Type)

This is a function that converts a UDINT type IP address to an array type.

■ Icon



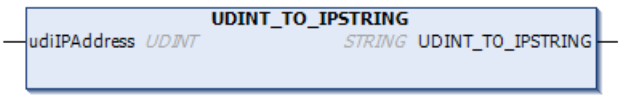
■ Parameter

Scope	Name	Type	Description
Input	abyIPAddress	UDINT	UDINT type IP address
Output	UDINT_TO_IPARRAY	ARRY[0..3] OF BYTE	IP address array

11.3.7 IoDrvEthernet.UDINT_TO_IPSTRING (UDINT Type to Character String Type)

This is a function that converts a UDINT type IP address to an array type.

■ Icon



■ Parameter

Scope	Name	Type	Description
Input	abyIPAddress	UDINT	UDINT type IP address
Output	UDINT_TO_STRING	STRING	Character string type IP address

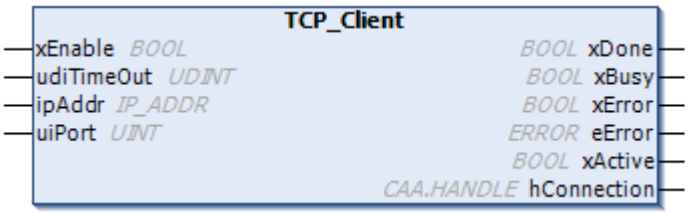
11.4 LAN Port (General-purpose Communication)

This section describes the library functions that are used to perform general-purpose communication with the LAN port using the TCP or UDP protocol.

11.4.1 NBS.TCP_Client (Connect to TCP Client)

This is a function block that connects to the TCP/IP client.

■ **Icon**



■ **Parameter**

Scope	Name	Type	Description
Input	xEnable	BOOL	TRUE: Active
	udiTimeOut	UDINT	Connection timeout (us) No timeout when set to 0.
	ipAddr	NBS.IP ADDR	Server IP address (character string type)
	uiPort	UINT	Server port No.
Output	xDone	BOOL	TRUE: Processing is completed.
	xBusy	BOOL	TRUE: Processing of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	eError	NBS.ERROR	Connection result Refer to "11.4.8 NBS.ERROR (Error Code)".
	xActive	BOOL	TRUE: Connection is established.
	hConnection	CAA.HANDLE	Connection handle (Valid when xActive = TRUE)

11.4.2 NBS.TCP_Connection (Connect TCP)

This is a function block that establishes the connection of the client connecting to the connection port opened by TCP_Server.

■ **Icon**

■ **Parameter**

Scope	Name	Type	Description
Input	xEnable	BOOL	TRUE: Active
	hServer	CAA.HANDLE	Connection port handle acquired by TCP_Server
Output	xDone	BOOL	TRUE: Processing is completed.
	xBusy	BOOL	TRUE: Processing of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	eError	NBS.ERROR	Connection result Refer to "11.4.8 NBS.ERROR (Error Code)".
	xActive	BOOL	TRUE: Connection is established.(Note 1)
	hConnection	CAA.HANDLE	Connection handle (Valid when xActive = TRUE)

(Note 1) To detect a disconnection from the client after the line is connected, it is necessary to periodically call TCP_Read.

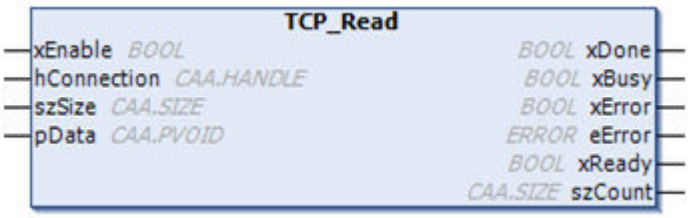
Info.

- When multiple clients are connected simultaneously to the same port, multiple TCP_Connection instances are created.
- The hServer handle acquired by one TCP_Server is set to the multiple TCP_Connection instances.

11.4.3 NBS.TCP_Read (Receive TCP Data)

This is a function block that acquires data received by the connection port that is established by TCP_Connection.

■ **Icon**



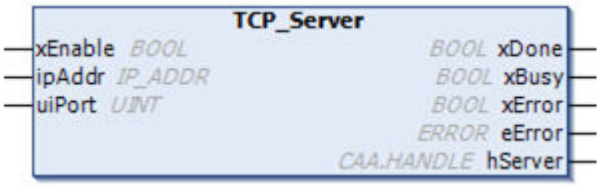
■ **Parameter**

Scope	Name	Type	Description
Input	xEnable	BOOL	TRUE: Active
	hConnection	CAA.HANDLE	Connection port handle acquired by TCP_Connection
	szSize	CAA.SIZE	Received buffer size (byte)
	pData	CAA.PVOID	Pointer to the receive buffer
Output	xDone	BOOL	Always FALSE
	xBusy	BOOL	TRUE: Processing of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	eError	NBS.ERROR	Connection result Refer to "11.4.8 NBS.ERROR (Error Code)".
	xReady	BOOL	TRUE: Data is received.
	szCount	CAA.SIZE	Received data size (byte)

11.4.4 NBS.TCP_Server (Connect TCP Server)

This is a function block that opens the specified port as a TCP/IP connection port.

■ **Icon**



■ **Parameter**

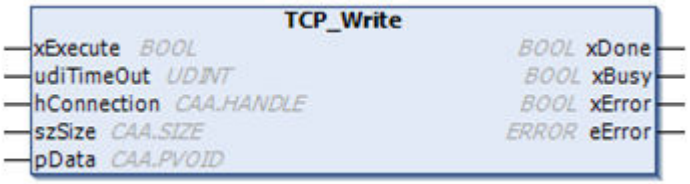
Scope	Name	Type	Description
Input	xEnable	BOOL	TRUE: Active
	ipAddr	NBS.IP_ADDR	Home IP address (character string), LANPort1 or LANPort2 IP address
	uiPort	UINT	Home waiting port number

Scope	Name	Type	Description
Output	xDone	BOOL	TRUE: Processing is completed.
	xBusy	BOOL	TRUE: Processing of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	eError	NBS.ERROR	Connection result Refer to "11.4.8 NBS.ERROR (Error Code)".
	hServer	CAA.HANDLE	Connection handle used by TCP_Connection

11.4.5 NBS.TCP_Write (Send TCP Data)

This is a function block that sends data to the connection port that is established by TCP_Connection.

■ Icon



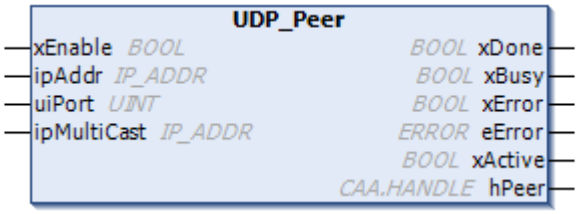
■ Parameter

Scope	Name	Type	Description
Input	xExecute	BOOL	TRUE: Send started (edge) FALSE: Processing ended (edge)
	udiTimeOut	UDINT	Timeout (us)
	hConnection	CAA.HANDLE	Connection port handle acquired by TCP_Connection
	szSize	CAA.SIZE	Send data size (byte)
	pData	CAA.PVOID	Pointer to the send data buffer.
Output	xDone	BOOL	TRUE: Processing is completed.
	xBusy	BOOL	TRUE: Processing of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	eError	NBS.ERROR	Connection result Refer to "11.4.8 NBS.ERROR (Error Code)".

11.4.6 NBS.UDP_Peer (Open UDP Port)

This is a function block that opens the UDP/IP port.

■ **Icon**



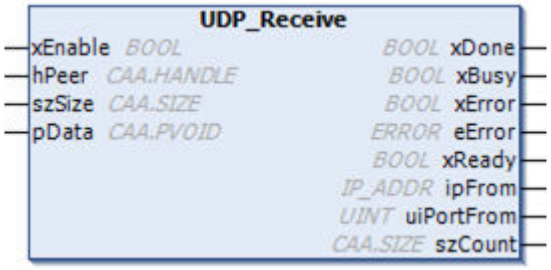
■ **Parameter**

Scope	Name	Type	Description
Input	xEnable	BOOL	TRUE: Active FALSE: Stop (xDone, xBusy, and xError are reset.)
	ipAddr	NBS.IP_ADDR	Home IP address (character string), LANPort1 or LANPort2 IP address
	uiPort	UINT	Home port number; Not possible to set to 0
	ipMultiCast	NBS.IP_ADDR	Multicast address ("255.255.255.255"=> INADDR_NONE)
Output	xDone	BOOL	TRUE: Processing is completed.
	xBusy	BOOL	TRUE: Processing of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	eError	NBS.ERROR	Connection result Refer to "11.4.8 NBS.ERROR (Error Code)".
	xActive	BOOL	TRUE: Connection is established.
	hPeer	CAA.HANDLE	Connection handle (Valid when xActive = TRUE)

11.4.7 NBS.UDP_Receive (Receive UDP Data)

This is a function block that receives data to the connection handle acquired by UDP_Peer.

■ **Icon**



■ Parameter

Scope	Name	Type	Description
Input	xEnable	BOOL	TRUE: Active FALSE: Stop (xDone, xBusy, and xError are reset.)
	hPeer	CAA.HANDLE	Connection handle acquired by UDP_Peer
	szSize	CAA.SIZE	Receive data buffer size (byte)
	pData	CAA.PVOID	Pointer to the receive data buffer
Output	xDone	BOOL	Always FALSE
	xBusy	BOOL	TRUE: Processing of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	eError	NBS.ERROR	Connection result Refer to "11.4.8 NBS.ERROR (Error Code)".
	xReady	BOOL	TRUE: Data acquired, FALSE: No received data
	ipFrom	NBS.IP_ADDR	Data sending source IP address
	uiPortFrom	UINT	Data sending source port No.
	szCount	CAA.SIZE	Received data size (byte)

(Note 1) If the szSize (receive data buffer size) is smaller than the received data size, only the data equivalent to the size specified by szSize is stored in pData and the data exceeding the size specified by szSize is discarded.

11.4.8 NBS.ERROR (Error Code)

This is an enumeration type error code that is output when the function block for communication instruction that uses the LAN port is executed.

■ NBS.ERROR (Enumeration type)

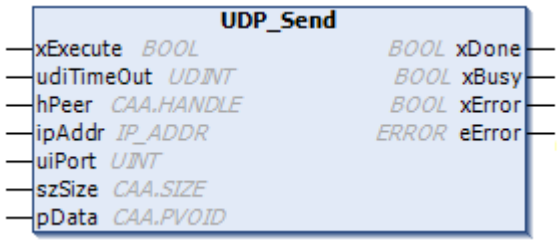
Name	Value	Description
NO_ERROR	0	No error is occurring.
FIRST_ERROR	6000	Reserved
TIME_OUT	6001	Reserved
INVALID_ADDR	6002	IP address is invalid.
INVALID_HANDLE	6003	Handle is invalid.
INVALID_DATAPOINTER	6004	Data pointer is invalid.
INVALID_DATASIZE	6005	Data size is invalid.
UDP_RECEIVE_ERROR	6006	UDP datagram cannot be received.
UDP_SEND_ERROR	6007	UDP datagram cannot be sent.
UDP_SEND_NOT_COMPLETE	6008	Reserved
UDP_OPEN_ERROR	6009	Port cannot be opened.

Name	Value	Description
UDP_CLOSE_ERROR	6010	Port cannot be released.
TCP_SEND_ERROR	6011	TCP message cannot be sent.
TCP_RECEIVE_ERROR	6012	TCP message cannot be received.
TCP_OPEN_ERROR	6013	TCP port cannot be created.
TCP_CONNECT_ERROR	6014	TCP connection cannot be established.
TCP_CLOSE_ERROR	6015	TCP port cannot be released.
TCP_SERVER_ERROR	6016	Reserved
WRONG_PARAMETER	6017	The parameter contains an invalid value.
ERROR_UNKNOWN	6018	Reserved
TCP_NO_CONNECTION	6019	There is no TCP connection.
IOCTL_ERROR	6020	Internal error (IOCTL is not supported.)
FIRST_MF	6050	Reserved
LAST_ERROR	6099	Reserved

11.4.9 NBS.UDP_Send (Send UDP Data)

This is a function block that sends data to the connection handle acquired by UDP_Peer.

■ Icon



■ Parameter

Scope	Name	Type	Description
Input	xExecute	BOOL	TRUE: Send started (edge) FALSE: Processing ended (edge)
	udiTimeOut	UDINT	Timeout (us)
	hPeer	CAA.HANDLE	Connection port handle acquired by UDP_Peer
	ipAddr	NBS.IP_ADDR	Destination IP address
	uiPort	UINT	Destination port No.
	szSize	CAA.SIZE	Send data size (byte)
	pData	CAA.PVOID	Pointer to the send data buffer.

Scope	Name	Type	Description
Output	xDone	BOOL	TRUE: Processing is completed.
	xBusy	BOOL	TRUE: Processing of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	eError	NBS.ERROR	Connection result Refer to "11.4.8 NBS.ERROR (Error Code)".

11.5 LAN Port (Modbus TCP)

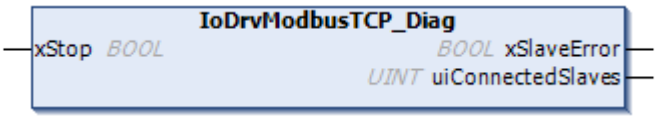
This section describes the library functions that are used to perform ModbusTCP communication with the LAN port.

It is created from Modbus master TCP available in the device tree.

11.5.1 IoDrvModbusTCP

This is a function block that controls the Modbus_TCP_Master device.

■ **Icon**



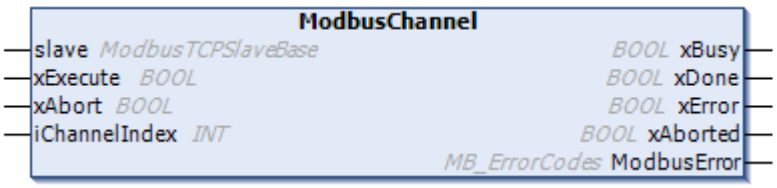
■ **Parameter**

Scope	Name	Type	Description
I/O	xStop	BOOL	TRUE: Stops sending commands to the slave.
Output	xSlaveError	BOOL	There is an error in the slave function
	uiConnectesSlaves	UINT	Number of slaves connected via TCP/IP

11.5.2 IoDrvModbusTCP.ModbusChannel (Start Sending Modbus Command)

This is a function block that sends the commands set in the Modbus Slave channel of the ModbusTCP_Slave device.

■ **Icon**



■ **Parameter**

Scope	Name	Type	Description
Input	xExecute	BOOL	Starts sending commands at the rising edge.
	xAbort	BOOL	TRUE: Stops execution and resets all outputs.
	iChannelIndex	INT	Channel number where commands to be sent are set
I/O	slave	ModbusTCPSlaveBase	Handle of the Modbus_TCP_Slave device Output

Scope	Name	Type	Description
Output	xBusy	BOOL	TRUE: Processing of the FB is not completed.
	xDone	BOOL	TRUE: Processing is completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	xAborted	BOOL	TRUE: Execution is stopped by the user's xAbort input
	ModbusError	MB_ErrorCodes	An error code is output. Refer to "11.5.5 IoDrvModbus.MB_ErrorCodes (Error Codes)".

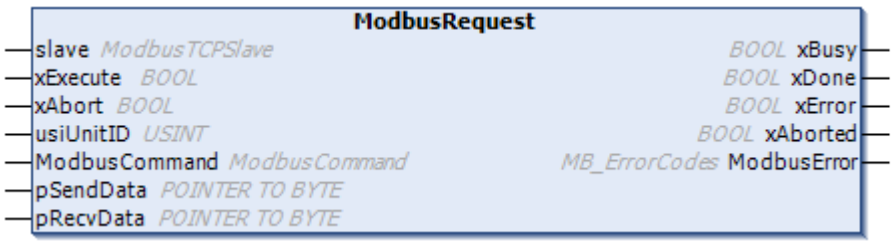
11.5.3 IoDrvModbusTCP.ModbusRequest (Modbus Request)

This is a function block that processes the Modbus command specified by I/O without using the Modbus_TCP_Slave device.

■ Supported commands

- Command 1 (Read multi-point coil state)
- Command 2 (Read multi-point input state)
- Command 3 (Read multi-point holding register)
- Command 4 (Read multi-point input register)
- Command 5 (Write single-point coil)
- Command 6 (Write single-point holding register)
- Command 15 (Write multi-point coil)
- Command 16 (Write multi-point holding register)
- Command 23 (Read / write multi-point holding register)

■ Icon



■ Parameter

Scope	Name	Type	Description
Input	slave	ModbusTCP Slave	Handle of the Modbus_TCP_Slave device
	xExecute	BOOL	Starts sending commands at the rising edge.
	xAbort	BOOL	TRUE: Stops execution and resets all outputs.
	usiUnitID	USINT	Slave address 1 to 247
	ModbusCommand	ModbusCommnand	Structure that stores parameters of the commands issued.

Scope	Name	Type	Description
	pSendData	POINTER TO BYTE	Pointer to the send data buffer.
	pRecvData	POINTER TO BYTE	Pointer to the receive data buffer
Output	xBusy	BOOL	TRUE: Processing of the FB is not completed.
	xDone	BOOL	TRUE: Processing is completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	xAborted	BOOL	TRUE: Execution is stopped by the user's xAbort input
	ModbusError	BYTE	An error code is output. Refer to "11.5.5 IoDrvModbus.MB_ErrorCodes (Error Codes)". Also possible to convert the type and use as enumeration type MB_ErrorCodes.

■ ModbusCommand (Structure)

Name	Type	Description
uiFunctionCode	UINT	Modbus command code
uiReadOffset	UINT	Read address 0 to 65535
uiReadLen	UINT	Range in the number of read instances varies depending on commands.
uiWriteOffset	UINT	Write address 0 to 65535
uiWriteLen	UINT	Range in the number of write instances varies depending on commands.

11.5.4 IoDrvModbusTCPSlave

This is a function block that controls the Modbus_TCP_Slave device.

■ Icon

■ Parameter

Scope	Name	Type	Description
Input	xConfirmError	BOOL	Resets xError and byModbusError and resumes communication.

Scope	Name	Type	Description
	xDoInit	BOOL	TRUE: Sends a slave initialization command when communication is resumed.
Output	xInitDone	UINT	TRUE: Modbus slave initialization command is fully completed.
	xDone	BOOL	TRUE: Processing is completed.
	xBusy	BOOL	TRUE: Processing of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	byModbusError	MB_ErrorCodes	An error code is output. Refer to "11.5.5 IoDrvModbus.MB_ErrorCodes (Error Codes)".
	ComSettings	ModbusTCPComSettings	IP address and port number registered in the Modbus_TCP_Slave device.
	ComState	ModbusTCPComState	Communication status
	iChannelIndex	INT	Channel number

11.5.5 IoDrvModbus.MB_ErrorCodes (Error Codes)

This is an enumeration type error code that is output when the function block for Modbus communication instruction that uses the COM port is executed.

■ IoDrvModbus.MB_ErrorCodes (Enumeration type)

Name	Value	Description
RESPONSE_SUCCESS	16#0	Succeeded
ILLEGAL_FUNCTION	16#1	Function code not supported by the slave
ILLEGAL_DATA_ADDRESS	16#2	Register offset not supported by the slave
ILLEGAL_DATA_VALUE	16#3	Illegal data writing
SLAVE_DEVICE_FAILURE	16#4	Non-recoverable error
ACKNOWLEDGE	16#5	Start operation
SLAVE_DEVICE_BUSY	16#6	During operation
MEMORY_PARITY_ERROR	16#8	Memory parity error
GATEWAY_PATH_UNAVAILABLE	16#A	Gateway path unavailable
GATEWAY_DEVICE_FAILED_TO_RESPOND	16#B	Gateway device failed to respond
RESPONSE_TIMEOUT	16#A1	Timeout
RESPONSE_CRC_FAIL	16#A2	CRC error
RESPONSE_WRONG_SLAVE	16#A3	Wrong response
RESPONSE_WRONG_FUNCTIONCODE	16#A4	Wrong function code in the response
REQUEST_FAILED_TO_SEND	16#A5	Request not sent

Name	Value	Description
RESPONSE_INVALID_DATA	16#A6	Invalid response data
RESPONSE_INVALID_PROTOCOL	16#A7	Invalid response protocol
RESPONSE_INVALID_HEADER	16#A8	Invalid response header
UNDEFINED	16#FF	Undefined

11.6 LAN Port (EtherNet/IP)

This section describes the instructions that are used to control EtherNet/IP scanner and adapter functions using the SMC.

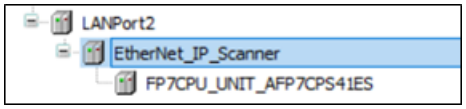
11.6.1 IoDrvEtherNetIP (EtherNet/IP Scanner Device)

This is a function block (FB) that controls the EtherNet/IP scanner device.

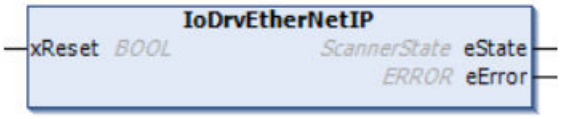
This function block is automatically generated by adding an EtherNet/IP scanner device and the name of the device that is added is used as the instance name.

Example

Adding an EtherNet/IP scanner device named "EtherNet_IP_Scanner" to LANPort2



■ **Icon**



■ **Parameter**

Scope	Name	Type	Description
Input	xReset	BOOL	Resets the scanner function at the rising edge
Output	eState	ScannerState	EtherNet/IP scanner device state
	eError	ERROR	Error state code of EtherNet/IP scanner

■ **ScannerState (EtherNet/IP scanner device state)**

Name	Description
INITIALIZING	The device is setting up a CIP object. It is continuing IP_CONFIG.
IP_CONFIG	The device creates an IP configuration for Ethernet interface and waits until it enters a RUNNING state.
UDP_CONFIG	The device opens the socket for UDP default port 2222.
ENCAPSULATION_CONFIG	The encapsulation server for the scanner is started via the default TCP port (44818).
ADAPTER_CONFIG	The device is in an empty state. It is continuing OPEN_CONNECTIONS.
OPEN_CONNECTIONS	The CIP ID status is set to "configured" and the RUNNING state continues.
RUNNING	The device opens a connection to the adapter and processes explicit messages with I/O communication.

Name	Description
DIAGNOSTIC_AVAILABLE	There are diagnostic messages from the configurator or editor.
BUS_ERROR	The UDP or TCP port failed to open.
RESET	xReset for the CIP ID object was received.
ERROR	When the network interface is in a continued state, the scanner enters the INITIALIZING state.

■ ERROR (Error state code of EtherNet/IP scanner)

Name	Description
NO_ERROR	No error is occurring.
INVALID_COMMAND	The command is invalid.
OUT_OF_MEMORY	A memory shortage occurred.
INVALID_DATA	The data is invalid.
INVALID_SESSION_HANDLE	The session handle is invalid.
INVALID_LENGTH	The data length is invalid.
UNSUPPORTED_PROTOCOL_VERSION	The protocol version is unsupported.
NBS_ERROR	An NBS error occurred.
NBS_RCV_ERROR	Data cannot be received via NBS.
NBS_SND_ERROR	Data cannot be sent via NBS.
ENCAPSULATION_ERROR	An encapsulation error occurred.
TCPIP_CONFIG_ERROR	TCP IP settings are incorrect.
UDP_CONFIG_ERROR	UDP settings are incorrect.
UDP_RECV_ERROR	UDP datagrams cannot be received.
UDP_SEND_ERROR	UDP datagrams cannot be sent.
UDP_CLOSE_ERROR	UDP ports cannot be released.
NULL_POINTER	This is a null pointer.
DEVICE_STATE_ERROR	An error is occurring on the device.
RECONFIGURATION_FAILED	Reconfiguration failed.
PERFORMANCE_MONITOR_DISABLED	The performance monitor is disabled.
INVALID_MEASURING_POINT	Measuring points are invalid.
IP_CONFIG_ERROR	IP settings are faulty.

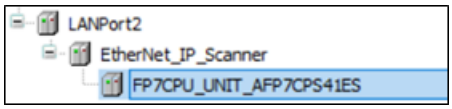
11.6.2 RemoteAdapter (Remote Adapter Device)

This is a function block (FB) for the remote adapter device linked to the EtherNet/IP scanner device.

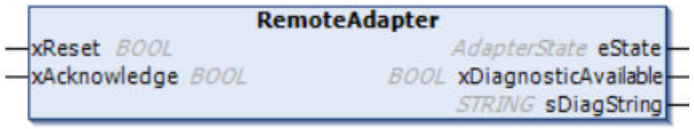
This function block is automatically generated by adding an EtherNet/IP remote adapter device and the name of the device that is added is used as the instance name.

Example

Adding a remote adapter device named "FP7CPU_UNIT_AFP7CPS41ES" to EtherNet_IP_Scanner



■ Icon



■ Parameter

Scope	Name	Type	Description
Input	xReset	BOOL	Resets the remote adapter function at the rising edge
	xAcknowledge	BOOL	Acknowledges the diagnostic information at the rising edge
Output	eState	Adapter State	Remote adapter state
	xDiagnosticAvailable	BOOL	The output remains TRUE when there is diagnostic information
	sDiagString	STRING	Diagnosis string

■ AdapterState (Adapter device state)

Name	Description
DISABLED	The device is disabled in device tree
NOT_CONFIGURED	Parameters are being loaded
IP_CONFIG	The device has configured a TCP object and is waiting for an Ethernet node
ENCAPSULATION_CONFIG	Encapsulation is being configured
LIST_SERVICES	List services are being executed
REGISTER_SESSION	Register session is in progress
PARAMETER_CONFIG	Parameters are being configured
CONFIGURED	The device is in configuration completion state
RUNNING	The device is in running state
IDLE	The device is in idle state
RESET	UDP and TCP connection is closing
RESET_SERVICE	Reset service is being executed
CONNECTIVITY_CHECK	Connectivity check is in progress
BUS_ERROR	Bus error is occurring

Name	Description
ERROR	Error is occurring

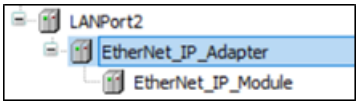
11.6.3 IoDrvEtherNetIPAdapter (EtherNet/IP adapter device)

This is a function block (FB) that controls the EtherNet/IP adapter device.

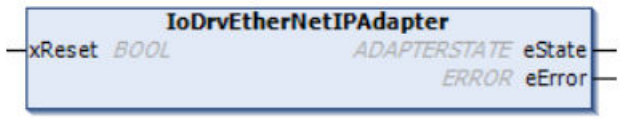
This function block is automatically generated by adding an EtherNet/IP adapter device and the name of the device that is added is used as the instance name.

Example

Adding an EtherNet/IP adapter device named "EtherNet_IP_Adapter" to LANPort2



■ Icon



■ Parameter

Scope	Name	Type	Description
Input	xReset	BOOL	Resets the adapter function at the rising edge
Output	eState	ADAPTERSTATE	EtherNet/IP adapter device state
	eError	ERROR	Error state code of EtherNet/IP adapter

■ ADAPTERSTATE (EtherNet/IP adapter device state)

Name	Description
UPDATE_CONFIGURATION	Startup phase
NOT_CONFIGURED	Parameters are being loaded
DISABLED	The device is disabled in device tree
CONFIGURED	A CIP object has been created
IP_CONFIG	The device has configured a TCP object and is waiting for an Ethernet node
IMPLICITMESSAGING_CONFIG	UDP port has been opened
EXPLICITMESSAGING_CONFIG	TCP port has been opened
NO_CONNECTION	The protocol stack has been started, but the scanner is unconnected.
RUNNING	The protocol stack is running, and the scanner is connected.

Name	Description
STOPPED	The Ethernet node is inactive, and the device is waiting for the Ethernet node to return.
RESET	UDP and TCP connection is closing.
SCANNER_EXTENSION	If the scanner registered this adapter as an I/O extension, the adapter is active in this state.
ERROR	Critical error
BUS_ERROR	Ethernet is not ready yet or is unavailable.

■ ERROR (EtherNet/IP adapter error state)

Name	Description
NO_ERROR	No error
TIME_OUT	Timeout
CONFIGURATION_FAILED	Failed to initialize resources, load connector parameters, or communicate with sub-connectors (modules)
IP_CONFIG_FAILED	The Ethernet node issued an error
IMPLICITMESSAGING_CONFIG_FAILED	Failed to create UDP port "CIP_ENC.ParameterList.gc_uiUDPPort" (default: 2222)
EXPLICITMESSAGING_CONFIG_FAILED	Failed to create TCP / UDP port "IP_ENC.ParameterList.gc_uiTCPPort" (default: 44818)
EXPLICITMESSAGE_RECEIVE_FAILED	Problem related to TCP or UDP port socket | CIP_ENC.ParameterList.gc_uiTCPPort | (default: 44818)
EXPLICITMESSAGE_SEND_FAILED	Problem related to TCP or UDP port socket | CIP_ENC.ParameterList.gc_uiTCPPort | (default: 44818)
LICENSE_MISSING	No license

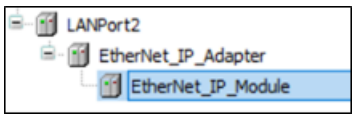
11.6.4 Module (EtherNet/IP Module Device)

This is a function block (FB) that controls the EtherNet/IP module device.

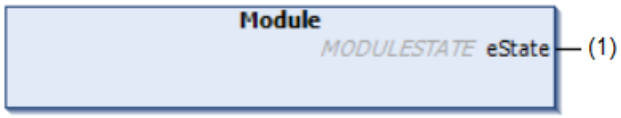
This function block is automatically generated by adding an EtherNet/IP module device and the name of the device that is added is used as the instance name.

Example

Adding an EtherNet/IP module device named "EtherNet_IP_Module" to EtherNet/IP adapter device



■ **Icon**



■ **Parameter**

Scope	Name	Type	Description
Output	eState	MODULESTATE	Module device state

■ **MODULESTATE (EtherNet/IP module device state)**

Name	Description
NOT_CONFIGURED	Parameters are being loaded.
CONFIGURED	A CIP object has been created.
NO_CONNECTION	The protocol stack has been started, but the scanner is unconnected.
RUNNING	The protocol stack is running, and the scanner is connected.
STOPPED	The Ethernet node is inactive, and the device is waiting for the Ethernet node to return.
DISABLED	The device is disabled in device tree.
ERROR	Critical error

11.6.5 Apply_Attributes (Apply_Attributes Service)

This is a function block (FB) that calls the "Apply_Attributes" service of the CIP object instance.

The attribute set in "Get_Attribut_Single" or "Get_Attribut_All" is adopted and saved in the adapter.

■ **Icon**

■ **Parameter**

Scope	Name	Type	Description
Input	xExecute	BOOL	Execution flag
	itfEtherNetIPDevice	IEtherNetIPService	EtherNet/IP device that implements the EtherNet/IP service

Scope	Name	Type	Description
	eClass	ENIP.CIPClass	Class that executes the service
	dwInstance	DWORD	Instance that executes the service (0: Class level, 1-x: Instance level)
Output	xDone	BOOL	Completion flag
	xBusy	BOOL	Busy flag
	xError	BOOL	Error flag
	eError	ENIP.ERROR	Error (0-255: CIP error, 256-x: Library error)

11.6.6 Generic_Service (Generic Service Execution)

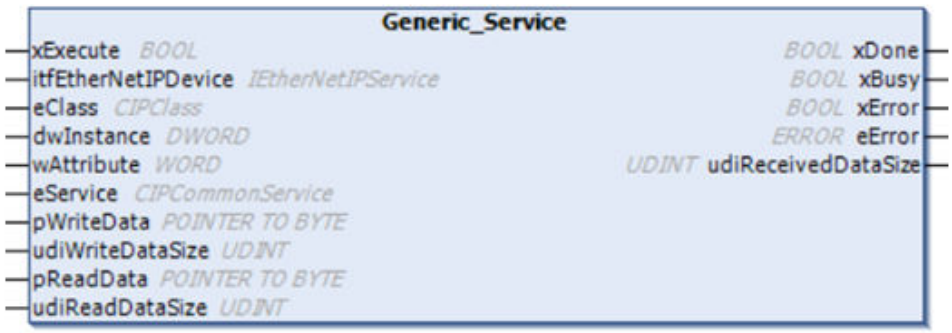
This is a function block (FB) that executes generic services with the EtherNet/IP adapter.

Messages are sent as unconnected explicit message requests.

Note

- The endianness of data to be sent or received must be exchanged by devices.

■ Icon



■ Parameter

Scope	Name	Type	Description
Input	xExecute	BOOL	Execution flag
	itfEtherNetIPDevice	IEtherNetIPService	EtherNet/IP device that implements the EtherNet/IP service
	eClass	ENIP.CIPClass	Class that executes the service
	dwInstance	DWORD	Instance that executes the service (0: Class level, 1-x: Instance level)
	wAttribute	WORD	Attribute corresponding to the service

Scope	Name	Type	Description
	eService	ENIP.CIPCommonService	CIPCommonService member service code or vendor-specific service code
	pWriteData	POINTER TO BYTE	Pointer to data to be written to the EtherNet/IP adapter. The parameter is set to 0 when no data is sent.
	udiWriteDataSize	UDINT	Size of data to be written to the EtherNet/IP adapter. The parameter is set to 0 when no data is sent.
	pReadData	POINTER TO BYTE	Storage pointer to data received from the EtherNet/IP adapter. The parameter is set to 0 when no data is received.
	udiReadDataSize	UDINT	Size of storage buffer for data received from the EtherNet/IP adapter. The parameter is set to 0 when no data is received.
Output	xDone	BOOL	Completion flag
	xBusy	BOOL	Busy flag
	xError	BOOL	Error flag
	eError	ENIP.ERROR	Error (0-255: CIP error, 256-x: Library error)
	udiReceivedDataSize	UDINT	Size of received data

■ ENIP.CIPCommonService (CIPCommonService member service code)

Name	Value
None	16#0
GET_ATTRIBUTES_ALL	16#1
SET_ATTRIBUTES_ALL	16#2
RESET	16#5
START	16#6
STOP	16#7
APPLY_ATTRIBUTES	16#D
GET_ATTRIBUTE_SINGLE	16#E
SET_ATTRIBUTE_SINGLE	16#10
NO_OPERATION	16#17

11.6.7 Get_Attribute_Single (Inquire Specific Attributes of a Specific Instance)

This is a function block (FB) that inquires specific attributes of a specific instance of the CIP object.

■ **Icon**

■ **Parameter**

Scope	Name	Type	Description
Input	xExecute	BOOL	Execution flag
	itfEtherNetIPDevice	IEtherNetIPService	EtherNet/IP device that implements the EtherNet/IP service
	eClass	ENIP.CIPClass	Class that executes the service
	dwInstance	DWORD	Instance that executes the service (0: Class level, 1-x: Instance level)
	wAttribute	WORD	Attribute corresponding to the service
	pData	POINTER TO BYTE	Storage pointer to data received from the EtherNet/IP adapter
	udiDataSize	UDINT	Size of storage buffer for data received from the EtherNet/IP adapter
Output	xDone	BOOL	Completion flag
	xBusy	BOOL	Busy flag
	xError	BOOL	Error flag
	eError	ENIP.ERROR	Error (0-255: CIP error, 256-x: Library error)
	udiReceivedDataSize	UDINT	Size of received data

11.6.8 Get_Attributes_All (Inquire All Attributes of a Specific Instance)

This is a function block (FB) that inquires all attributes of a specific instance of the CIP object.

■ **Icon**

■ **Parameter**

Scope	Name	Type	Description
Input	xExecute	BOOL	Execution flag
	itfEtherNetIPDevice	IEtherNetIPService	EtherNet/IP device that implements the EtherNet/IP service
	eClass	ENIP.CIPClass	Class that executes the service
	dwInstance	DWORD	Instance that executes the service (0: Class level, 1-x: Instance level)
	pData	POINTER TO BYTE	Storage pointer to data received from the EtherNet/IP adapter
	udiDataSize	UDINT	Size of storage buffer for data received from the EtherNet/IP adapter
Output	xDone	BOOL	Completion flag
	xBusy	BOOL	Busy flag
	xError	BOOL	Error flag
	eError	ENIP.ERROR	Error (0-255: CIP error, 256-x: Library error)
	udiReceivedDataSize	UDINT	Size of received data

11.6.9 Set_Attribute_Single (Set Specific Attributes of a Specific Instance)

This is a function block (FB) that sets specific attributes of a specific instance of the CIP object

■ **Icon**

■ **Parameter**

Scope	Name	Type	Description
Input	xExecute	BOOL	Execution flag
	itfEtherNetIPDevice	IEtherNetIPService	EtherNet/IP device that implements the EtherNet/IP service
	eClass	ENIP.CIPClass	Class that executes the service
	dwInstance	DWORD	Instance that executes the service (0: Class level, 1-x: Instance level)
	wAttribute	WORD	Attribute corresponding to the service
	pData	POINTER TO BYTE	Pointer to data to be written
	udiDataSize	UDINT	Size of data to be written
Output	xDone	BOOL	Completion flag
	xBusy	BOOL	Busy flag
	xError	BOOL	Error flag
	eError	ENIP.ERROR	Error (0-255: CIP error, 256-x: Library error)

11.6.10 Set_Attributes_All (Set All Attributes of a Specific Instance)

This is a function block (FB) that sets all attributes of a specific instance of the CIP object.

■ **Icon**

■ **Parameter**

Scope	Name	Type	Description
Input	xExecute	BOOL	Execution flag
	itfEtherNetIPDevice	IEtherNetIPService	EtherNet/IP device that implements the EtherNet/IP service
	eClass	ENIP.CIPClass	Class that executes the service
	dwInstance	DWORD	Instance that executes the service (0: Class level, 1-x: Instance level)

Scope	Name	Type	Description
	pData	POINTER TO BYTE	Pointer to data to be written
	udiDataSize	UDINT	Size of data to be written
Output	xDone	BOOL	Completion flag
	xBusy	BOOL	Busy flag
	xError	BOOL	Error flag
	eError	ENIP.ERROR	Error (0-255: CIP error, 256-x: Library error)

11.6.11 NOP (NOP Service)

This is a function block (FB) that executes the NOP service of a specific instance of the CIP object.

Normally, this service is used to check whether the adapter can still be used in the network.

■ Icon

■ Parameter

Scope	Name	Type	Description
Input	xExecute	BOOL	Execution flag
	itfEtherNetIPDevice	IEtherNetIPService	EtherNet/IP device that implements the EtherNet/IP service
	eClass	ENIP.CIPClass	Class that executes the service
	dwInstance	DWORD	Instance that executes the service (0: Class level, 1-x: Instance level)
Output	xDone	BOOL	Completion flag
	xBusy	BOOL	Busy flag
	xError	BOOL	Error flag
	eError	ENIP.ERROR	Error (0-255: CIP error, 256-x: Library error)

11.6.12 Reset (Reset Service)

This is a function block (FB) that executes the Reset service of a specific instance of the CIP object.

The effects of this service differ according to the CIP object.

■ Icon

■ Parameter

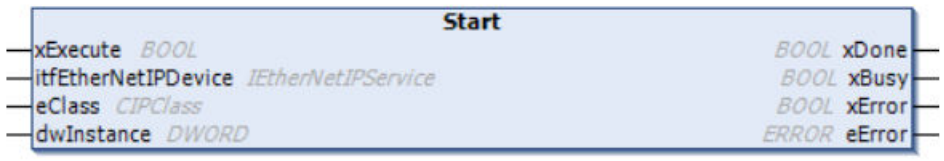
Scope	Name	Type	Description
Input	xExecute	BOOL	Execution flag
	itfEtherNetIPDevice	IEtherNetIPService	EtherNet/IP device that implements the EtherNet/IP service
	eClass	ENIP.CIPClass	Class that executes the service
	dwInstance	DWORD	Instance that executes the service (0: Class level, 1-x: Instance level)
Output	xDone	BOOL	Completion flag
	xBusy	BOOL	Busy flag
	xError	BOOL	Error flag
	eError	ENIP.ERROR	Error (0-255: CIP error, 256-x: Library error)

11.6.13 Start (Start Service)

This is a function block (FB) that executes the Start service of a specific instance of the CIP object.

The effects of this service differ according to the CIP object.

■ Icon



■ **Parameter**

Scope	Name	Type	Description
Input	xExecute	BOOL	Execution flag
	itfEtherNetIPDevice	IEtherNetIPService	EtherNet/IP device that implements the EtherNet/IP service
	eClass	ENIP.CIPClass	Class that executes the service
	dwInstance	DWORD	Instance that executes the service (0: Class level, 1-x: Instance level)
Output	xDone	BOOL	Completion flag
	xBusy	BOOL	Busy flag
	xError	BOOL	Error flag
	eError	ENIP.ERROR	Error (0-255: CIP error, 256-x: Library error)

11.6.14 Stop (Stop Service)

This is a function block (FB) that executes the Stop service of a specific instance of the CIP object.

The effects of this service differ according to the CIP object.

■ **Icon**

■ **Parameter**

Scope	Name	Type	Description
Input	xExecute	BOOL	Execution flag
	itfEtherNetIPDevice	IEtherNetIPService	EtherNet/IP device that implements the EtherNet/IP service
	eClass	ENIP.CIPClass	Class that executes the service
	dwInstance	DWORD	Instance that executes the service (0: Class level, 1-x: Instance level)
Output	xDone	BOOL	Completion flag
	xBusy	BOOL	Busy flag
	xError	BOOL	Error flag

Scope	Name	Type	Description
	eError	ENIP.ERROR	Error (0-255: CIP error, 256-x: Library error)

11.6.15 ENIP.ERROR (Message Service Instruction Error Code)

Name	Value	Description
NO_ERROR	0	The service was executed normally by the specified object.
CONNECTION_FAILURE	16#1	The connection-related service failed due to the connection path.
RESOURCE_UNAVAILABLE	16#2	The object was unable to use the resources that it required to execute the requested service.
INVALID_PARAM_VALUE	16#3	Refer to status code 16#20 that is an appropriate value to be used in this situation.
PATH_SEGMENT_ERROR	16#4	The path segment identifier or segment syntax was not recognized by the processing node. Path processing stops when a path segment error occurs.
PATH_DESTINATION_UNKNOWN	16#5	The path refers to an object class, instance, or structure element that is unknown or not included in the processing node. If an unknown path destination error occurs, path processing will stop.
PARTIAL_TRANSFER	16#6	Only part of the expected data was transferred.
CONNECTION_LOST	16#7	The messaging connection was lost.
SERVICE_NOT_SUPPORTED	16#8	The requested service is not implemented or defined for this object class or instance.
INVALID_ATTRIBUTE_VALUE	16#9	Invalid attribute data was detected.
ATTRIBUTE_LIST_ERROR	16#A	The status of the attribute of Get_Attribute_List or Set_Attribute_List response is other than zero.
ALREADY_IN_REQUEST_STATE	16#B	The object is already in the mode or state requested by the service.
OBJECT_STATE_ERROR	16#C	The object cannot execute the requested service in the current mode or state.
OBJECT_ALREADY_EXISTS	16#D	An instance requested for the object to be created already exists.
ATTRIBUTE_NOT_SETTABLE	16#E	A request to change a read-only attribute was received.
PRIVILEGE_VIOLATION	16#F	An authority / privilege check failed.
DEVICE_STATE_ERROR	16#10	The current mode or state of the device prohibits the requested service from being executed.
REPLY_DATA_TOO_LARGE	16#11	The size of data to be sent via a response buffer is larger than the capacity of the allocated response buffer.
FRAGMENTATION_OF_VALUE	16#12	The service specifies an operation that fragmentates half of primitive data values which are a REAL data type.
NOT_ENOUGH_DATA	16#13	The service did not provide enough data to execute the specified operation.

Name	Value	Description
ATTRIBUTE_NOT_SUPPORTED	16#14	The attribute specified in the request is not supported.
TOO_MUCH_DATA	16#15	The service provided more data than expected.
OBJECT_DOES_NOT_EXIST	16#16	The specified object does not exist in the device.
SERVICE_FRAGMENTATION_ SEQUENCE_NOT_IN_PROGRESS	16#17	The fragmentation sequence of this service is currently not active for this data.
NO_STORED_ATTRIBUTE_DATA	16#18	The attribute data of this object has not been saved before the requested service is executed.
STORE_OPERATION_FAILURE	16#19	The attribute data of this object has not been saved because an error occurred during the attempt to save the data.
ROUTING_FAILURE_REQUEST_ PACKET_TOO_LARGE	16#1A	The service request packet was too large to send through the network existing in the path to the destination. The routing device forcibly canceled the service.
ROUTING_FAILURE_RESPONSE_ PACKET_TOO_LARGE	16#1B	The service response packet was too large to send through the network existing in the path from the destination. The routing device forcibly canceled the service.
MISSING_ATTRIBUTE_LIST_ ENTRY_DATA	16#1C	The service did not provide attributes in the list of attributes that it requires to execute the requested operation.
INVALID_ATTRIBUTE_VALUE_LIST	16#1D	The service returned a list of provided attributes together with status information of invalid attributes.
EMBEDDED_SERVICE_ERROR	16#1E	An error occurred in the embedded service.
VENDOR_SPECIFIC_ERROR	16#1F	A vendor-specific error occurred. The additional code field for error response is used to define a specific error that occurred. Use this field only if the error in question does not apply to any of the error codes shown in these tables or those shown in the object class definition.
INVALID_PARAMETER	16#20	The parameter associated with the request is invalid. This code is used when the parameter does not meet the requirements of this specification or the requirements defined in the application object specification.
WRITE_ONEC_VALUE_OR_ MEDIUM_ALREADY_WRITTEN	16#21	An attempt was made to write to a write-once medium (such as WORM drive or PROM) to which data has already been written or to change a value that cannot be changed once set.
INVALID_REPLY_RECEIVED	16#22	An invalid response was received (for example, the response service code does not match the request service code or the response message is shorter than the expected minimum response size). This status code is useful to investigate other causes of invalid responses.
BUFFER_OVERFLOW	16#23	The size of the received message exceeds the maximum size of messages that can be handled by the receiver buffer. The entire message was discarded.
MESSAGE_FORMAT_ERROR	16#24	The format of the received message is not supported by the server.
KEY_FAILURE_IN_PATH	16#25	The key segment included as the first segment of the path does not match the destination module. The object-specific status indicates which part of the key check has failed.

Name	Value	Description
PATH_SIZE_INVALID	16#26	The size of the path sent with the service request is not large enough to route the request to the object or routing data included in the path is too much.
UNEXPECTED_ATTRIBUTE_IN_LIST	16#27	An attempt was made to set an attribute that cannot currently be set.
INVALID_MEMBER_ID	16#28	The member ID specified in the request does not exist in the specified class, instance, or attribute.
MEMBER_NOT_SETTABLE	16#29	A request to change an unchangeable member was received.
GROUP_2_ONLY_SERVER_ GENERAL_FAILURE	16#2A	This error code is issued only by DeviceNet Group 2 Only servers with 4K or less code space and is supported only instead of the server. Attributes are not supported and cannot be set.
UNKNOWN_MODBUS_ERROR	16#2B	The program for conversion from CIP to Modbus received an unknown Modbus exception code.
ATTRIBUTE_NOT_GETTABLE	16#2C	A request to read an unreadable attribute was received.
INSTANCE_NOT_DELETABLE	16#2D	The requested object instance cannot be deleted.
SERVICE_NOT_SUPPORTED_ FOR_SPECIFIED_PATH	16#2E	The object supports the service but does not support the specified application path (such as attributes). Note: Do not use this code for the set service. (Instead, use general status code 16#0E or 16#29.)
TIME_OUT	16#100	The request has timed out.
INTERFACE_MISSING		IEtherNetIPService is not implemented.
REMOTE_CALL_FAILED		There is no physical connection.
NULL_POINTER		A null value was entered by mistake.
INVALID_DATA_SIZE		The data size is invalid.
WRONG_INTERFACE_VERSION		The versions do not match. The device is not equipped with the same version of interface as the called method.
NO_MEMORY		There is not enough memory.
UNKNOWN_ERROR		An unknown error occurred.
ABORTED		The service was aborted.

11.6.16 ENIP.CIPClass (Service Class Code)

Name	Value
IdentityObject	16#1
MessageRouterObject	16#2
DeviceNetObject	16#3
AssemblyObject	16#4
ConnectionObject	16#5
ConnectionManagerObject	16#6
RegisterObject	16#7

Name	Value
DiscreteInputPointObject	16#8
DiscreteOutputPointObject	16#9
AnalogInputPointObject	16#A
AnalogOutputPointObject	16#B
PresenceSensingObject	16#E
ParameterObject	16#F
ParameterGroupObject	16#10
GroupObject	16#12
DiscreteInputGroupObject	16#1D
DiscreteOutputGroupObject	16#1E
DiscreteGroupObject	16#1F
AnalogInputGroupObject	16#20
AnalogOutputGroupObject	16#21
AnalogGroupObject	16#22
PositionSensorObject	16#23
PositionControllerSupervisorObject	16#24
PositionControllerObject	16#25
BlockSequencerObject	16#26
CommandBlockObject	16#27
MotorDataObject	16#28
ControlSupervisorObject	16#29
ACDCDriveObject	16#2A
AcknowledgeHandlerObject	16#2B
OverloadObject	16#2C
SoftstartObject	16#2D
SelectionObject	16#2E
S_DeviceSupervisorObject	16#30
S_AnalogSensorObject	16#31
S_AnalogActuatorObject	16#32
S_SingleStageControllerObject	16#33
S_GasCalibrationObject	16#34
TripPointObject	16#35
FileObject	16#37
S_PartialPressureObject	16#38
SafetySupervisorObject	16#39
SafetyValidatorObject	16#3A
SafetyDiscreteOutputPointObject	16#3B

Name	Value
SafetyDiscreteOutputGroupObject	16#3C
SafetyDiscreteInputPointObject	16#3D
SafetyDiscreteInputGroupObject	16#3E
SafetyDualChannelOutputObject	16#3F
S_SensorCalibrationObject	16#40
EventLogObject	16#41
MotionDeviceAxisObject	16#42
TimeSyncObject	16#43
ModbusObject	16#44
OriginatorConnectionListObject	16#45
ModbusSerialLinkObject	16#46
DeviceLevelRingObject	16#47
QoSObject	16#48
SafetyAnalogInputPointObject	16#49
SafetyAnalogInputGroupObject	16#4A
SafetyDualChannelAnalogInputObject	16#4B
SERCOSIIILinkObject	16#4C
TargetConnectionListObject	16#4D
EnergyObject	16#4E
ElectricalEnergyObject	16#4F
Non_ElectricalEnergyObject	16#50
BaseSwitchObject	16#51
SNMPObject	16#52
PowerManagementObject	16#53
ControlNetObject	16#F0
ControlNetKeeperObject	16#F1
ControlNetSchedulingObject	16#F2
ConnectionConfigurationObject	16#F3
PortObject	16#F4
TCPIPInterfaceObject	16#F5
EthernetLinkObject	16#F6
CompoNetLink	16#F7
CompoNetRepeater	16#F8

11.7 SD Card Operation (File Operation)

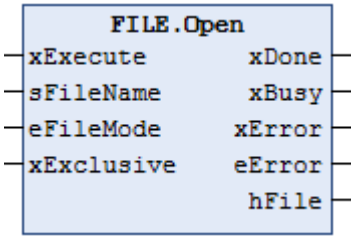
Files in the SD card inserted in the SD memory card slot can be operated.

In file operation using the GM1 Controller, WSTRING (kanji) cannot be used in the file name and directory name.

11.7.1 FILE.Open (Open File)

This is a function block that opens a file or creates a new file.

■ Icon



■ Parameter

Scope	Name	Type	Description
Input	xExecute	BOOL	Starts execution at the rising edge.
	sFileName	FILE.CAA.FILENAME	Specifies the file name with an absolute path or relative path.
	eFileMode	FILE.MODE	File mode
	xExclusive	BOOL	TRUE: Exclusive access mode FALSE: Multiple access mode xExclusive is not supported.
Output	xDone	BOOL	TRUE: Execution is completed.
	xBusy	BOOL	TRUE: Execution of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	eError	FILE.ERROR	An error ID is output. Refer to "11.7.15 FILE.ERROR (Error ID)".
	hFile	FILE.CAA.HANDLE	Handle of a file

■ FILE.MODE (Enumeration type)

Name	Value	Description
MWRITE	0	Overwrite mode (When the specified file does not exist, a new file is created.)
MREAD	1	Read mode
MRDWR	2	Read / write mode (When the specified file does not exist, a new file is created.)

Name	Value	Description
MAPPD	3	Append write mode

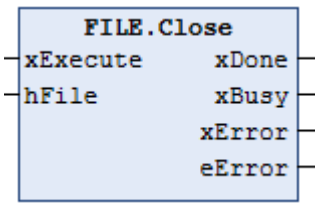
Info.

- You cannot use full size characters and the following symbols in a file name: [\], [/], [:], [*], [?], [”], [<], [>], [|].

11.7.2 FILE.Close (Close File)

This is a function block that closes a file.

■ Icon



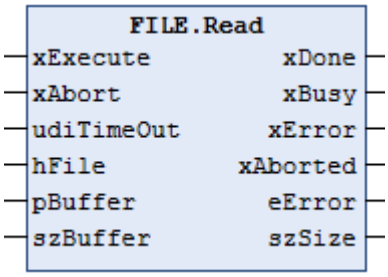
■ Parameter

Scope	Name	Type	Description
Input	xExecute	BOOL	Starts execution at the rising edge.
	hFile	FILE.CAA.HANDLE	Handle of a file to be closed Specifies the handle output by FILE.Open.
Output	xDone	BOOL	TRUE: Execution is completed.
	xBusy	BOOL	TRUE: Execution of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	eError	FILE.ERROR	An error ID is output. Refer to "11.7.15 FILE.ERROR (Error ID)".

11.7.3 FILE.Read (Read File)

This is a function block that reads data from the file opened.

■ **Icon**



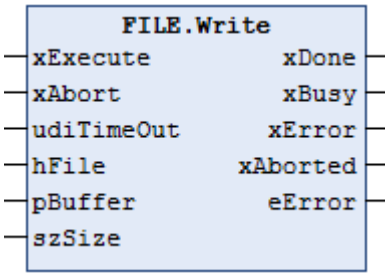
■ **Parameter**

Scope	Name	Type	Description
Input	xExecute	BOOL	Starts execution at the rising edge.
	xAbort	BOOL	TRUE: Stops execution and resets all outputs.
	udiTimeOut	UDINT	Timeout time until the execution is stopped (μs)
	hFile	FILE.CAA.HANDLE	Handle of a file Specifies the handle output by FILE.Open.
	pBuffer	FILE.CAA.PVOID	Pointer to the data buffer to be read Gets a pointer by the ADR operator.
	szBuffer	FILE.CAA.SIZE	Size of the data buffer to be read Gets a pointer by the SIZEOF operator.
Output	xDone	BOOL	TRUE: Execution is completed.
	xBusy	BOOL	TRUE: Execution of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	xAborted	BOOL	TRUE: Execution is stopped by the user.
	eError	FILE.ERROR	An error ID is output. Refer to "11.7.15 FILE.ERROR (Error ID)".
	szSize	FILE.CAA.SIZE	Size of the read data buffer

11.7.4 FILE.Write (Write File)

This is a function block that writes data to the file opened.

■ **Icon**



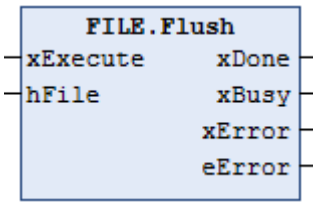
■ **Parameter**

Scope	Name	Type	Description
Input	xExecute	BOOL	Starts execution at the rising edge.
	xAbort	BOOL	TRUE: Stops execution and resets all outputs.
	udiTimeOut	UDINT	Timeout time until the execution is stopped (μs)
	hFile	FILE.CAA.HANDLE	Handle of a file Specifies the handle output by FILE.Open.
	pBuffer	FILE.CAA.PVOID	Pointer to the data buffer to be written Gets a pointer by the ADR operator.
	szSize	FILE.CAA.SIZE	Size of the data buffer to be written Gets a pointer by the SIZEOF operator.
Output	xDone	BOOL	TRUE: Execution is completed.
	xBusy	BOOL	TRUE: Execution of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	xAborted	BOOL	TRUE: Execution is stopped by the user.
	eError	FILE.ERROR	An error ID is output. Refer to "11.7.15 FILE.ERROR (Error ID)".

11.7.5 FILE.Flush (Flush File)

This is a function block that flushes buffer contents to a file.

■ **Icon**



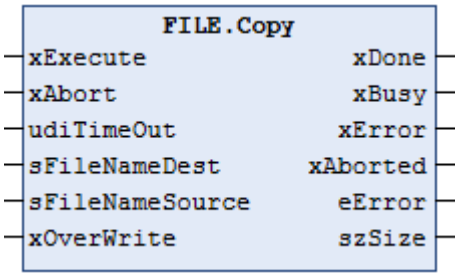
■ **Parameter**

Scope	Name	Type	Description
Input	xExecute	BOOL	Starts execution at the rising edge.
	hFile	FILE.CAA.HANDLE	Handle of a file Specifies the handle output by FILE.Open.
Output	xDone	BOOL	TRUE: Execution is completed.
	xBusy	BOOL	TRUE: Execution of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	eError	FILE.ERROR	An error ID is output. Refer to "11.7.15 FILE.ERROR (Error ID)".

11.7.6 FILE.Copy (Copy File)

This is a function block that copies a file.

■ **Icon**



■ **Parameter**

Scope	Name	Type	Description
Input	xExecute	BOOL	Starts execution at the rising edge.
	xAbort	BOOL	TRUE: Stops execution and resets all outputs.
	udiTimeOut	UDINT	Timeout time until the execution is stopped (μs)
	sFileNameDest	FILE.CAA.FILENAME	Copy destination file name
	sFileNameSource	FILE.CAA.FILENAME	Copy source file name
	xOverWrite	BOOL	TRUE: Copies to overwrite an existing file. FALSE: Outputs an error without copying to overwrite. If FALSE is specified in a case where there is an existing file, copy is not executed. No error is output.
Output	xDone	BOOL	TRUE: Execution is completed.
	xBusy	BOOL	TRUE: Execution of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.

Scope	Name	Type	Description
	xAborted	BOOL	TRUE: Execution is stopped by the user.
	eError	FILE.ERROR	An error ID is output. Refer to "11.7.15 FILE.ERROR (Error ID)".
	szSize	FILE.CAA.SIZE	Size of the copied file

Info.

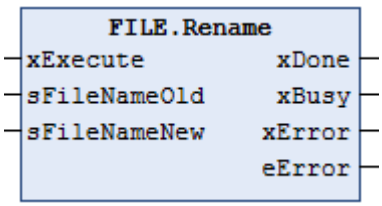
- You cannot use full size characters and the following symbols in a file name: [\], [/], [:], [*], [?], [”], [<], [>], [|].

11.7.7 FILE.Rename (Rename File)

This is a function block that changes a file name.

It is not possible to change the directory name of a directory that is currently open. Close it using the DirClose function block.

■ Icon



■ Parameter

Scope	Name	Type	Description
Input	xExecute	BOOL	Starts execution at the rising edge.
	sFileNameOld	FILE.CAA.FILENAME	File name before change
	sFileNameOld	FILE.CAA.FILENAME	File name after change
Output	xDone	BOOL	TRUE: Execution is completed.
	xBusy	BOOL	TRUE: Execution of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	eError	FILE.ERROR	An error ID is output. Refer to "11.7.15 FILE.ERROR (Error ID)".

Info.

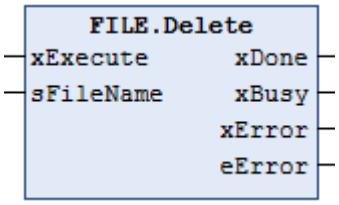
- You cannot use full size characters and the following symbols in a file name: [\], [/], [:], [*], [?], [”], [<], [>], [|].

11.7.8 FILE.Delete (Delete File)

This is a function block that deletes a file.

It is not possible to delete a file that is currently open. Close it using the Close function block.

■ Icon



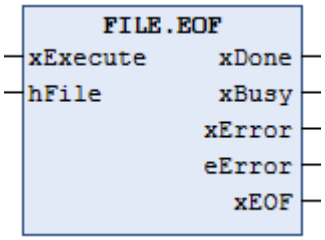
■ Parameter

Scope	Name	Type	Description
Input	xExecute	BOOL	Starts execution at the rising edge.
	sFileName	FILE.CAA.FILENAME	File to be deleted
Output	xDone	BOOL	TRUE: Execution is completed.
	xBusy	BOOL	TRUE: Execution of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	eError	FILE.ERROR	An error ID is output. Refer to "11.7.15 FILE.ERROR (Error ID)".

11.7.9 FILE.EOF (End of File)

This is a function block that determines whether the current offset of a file is EOF (End Of File) or not. It can be used only when the OPEN mode is set to MREAD/MREADPLUS.

■ Icon



■ Parameter

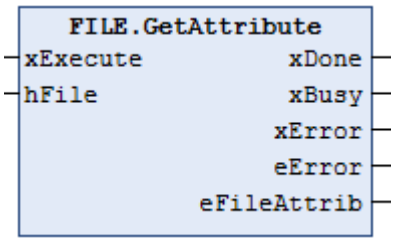
Scope	Name	Type	Description
Input	xExecute	BOOL	Starts execution at the rising edge.

Scope	Name	Type	Description
	hFile	FILE.CAA.FILENAME	Handle of a file Specifies the handle output by FILE.Open.
Output	xDone	BOOL	TRUE: Execution is completed.
	xBusy	BOOL	TRUE: Execution of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	eError	FILE.ERROR	An error ID is output. Refer to "11.7.15 FILE.ERROR (Error ID)".
	xEOF	BOOL	File: The current offset is EOF.

11.7.10 FILE.GetAttribute (Get File Attribute)

This is a function block that gets file attributes.

■ Icon



■ Parameter

Scope	Name	Type	Description
Input	xExecute	BOOL	Starts execution at the rising edge.
	hFile	FILE.CAA.FILENAME	Handle of a file Specifies the handle output by FILE.Open.
Output	xDone	BOOL	TRUE: Execution is completed.
	xBusy	BOOL	TRUE: Execution of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	eError	FILE.ERROR	An error ID is output. Refer to "11.7.15 FILE.ERROR (Error ID)".
	eFileAttrib	FILE.ATTRIB	TRUE: The current offset is EOF. FALSE: The current offset is not EOF.

■ FILE.ATTRIB (Enumeration type)

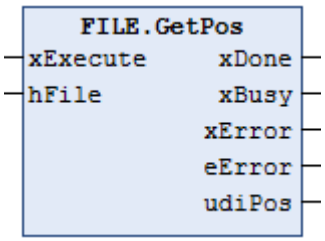
Name	Value	Description
ARCHIVE	0	Archive file
HIDDEN	1	Hidden file

Name	Value	Description
NORMAL	2	File without any other attributes
READONLY	3	Read only

11.7.11 FILE.GetPos (Get File Offset)

This is a function block that gets the current offset of a file.

■ Icon



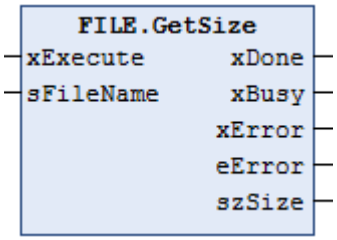
■ Parameter

Scope	Name	Type	Description
Input	xExecute	BOOL	Starts execution at the rising edge.
	hFile	FILE.CAA.FILENAME	Handle of a file Specifies the handle output by FILE.Open.
Output	xDone	BOOL	TRUE: Execution is completed.
	xBusy	BOOL	TRUE: Execution of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	eError	FILE.ERROR	An error ID is output. Refer to "11.7.15 FILE.ERROR (Error ID)".
	udiPos	__UXINT	The current offset (byte) is output.

11.7.12 FILE.GetSize (Get File Size)

This is a function block that gets the file size.

■ **Icon**



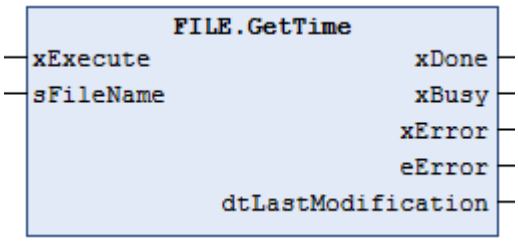
■ **Parameter**

Scope	Name	Type	Description
Input	xExecute	BOOL	Starts execution at the rising edge.
	sFileName	FILE.CAA.FILENAME	File from which to get the file size
Output	xDone	BOOL	TRUE: Execution is completed.
	xBusy	BOOL	TRUE: Execution of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	eError	FILE.ERROR	An error ID is output. Refer to "11.7.15 FILE.ERROR (Error ID)".
	szSize	FILE.CAA.SIZE	The file size (byte) is output.

11.7.13 FILE.GetTime (Get File Update Time)

This is a function block that gets the update time of a file.

■ **Icon**



■ **Parameter**

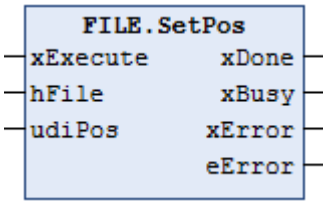
Scope	Name	Type	Description
Input	xExecute	BOOL	Starts execution at the rising edge.
	sFileName	FILE.CAA.FILENAME	File from which to get the file update time
Output	xDone	BOOL	TRUE: Execution is completed.

Scope	Name	Type	Description
	xBusy	BOOL	TRUE: Execution of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	eError	FILE.ERROR	An error ID is output. Refer to "11.7.15 FILE.ERROR (Error ID)".
	dtLastModification	DATE_AND_TIME	The last update date and time is output. Example: DATE_AND_TIME#2020-01-11-15:12:30

11.7.14 FILE.SetPos (Set File Offset)

This is a function block that sets the offset of a file.

■ Icon



■ Parameter

Scope	Name	Type	Description
Input	xExecute	BOOL	Starts execution at the rising edge.
	hFile	FILE.CAA.HANDLE	Handle of a file Specifies the handle output by FILE.Open.
	udiPos	__UXINT	Offset to be set (byte)
Output	xDone	BOOL	TRUE: Execution is completed.
	xBusy	BOOL	TRUE: Execution of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	eError	FILE.ERROR	An error ID is output. Refer to "11.7.15 FILE.ERROR (Error ID)".

11.7.15 FILE.ERROR (Error ID)

This is an enumeration type error ID that is output when a function block for file operation is executed. It is used to output an error in a file operation or directory operation of the SD card.

■ FILE.ERROR (Enumeration type)

Name	Value	Description
NO_ERROR	0	Normal end

Name	Value	Description
FIRST_ERROR	5100	First library specific error
TIME_OUT	5101	Timeout
ABORT	5102	Aborts processing by xAbort.
HANDLE_INVALID	5103	Invalid handle
NOT_EXIST	5104	No file or directory exists.
EXIST	5105	A file or directory already exists
NO_MORE_ENTRIES	5106	There are no other entries.
NOT_EMPTY	5107	The file or directory is not empty.
READ_ONLY_CAA	5108	The file or directory is write protected.
WRONG_PARAMETER	5109	Wrong parameter
ERROR_UNKNOWN	5110	Unknown error
WRITE_INCOMPLETE	5111	Not all the data is written.
FILE_NOT_IMPLEMENTED	5112	The function is not implemented.
ASM_CREATEJOB_FAILED	5113	Failed to create an AsyncManager job.
FILE_OPERATION_DENIED	5114	No access due to ForceFilePath / ForceIecFilePath
FIRST_MF	5150	First error unique to the manufacturer
LAST_ERROR	5199	insert manuf. specific errors here Last library specific error

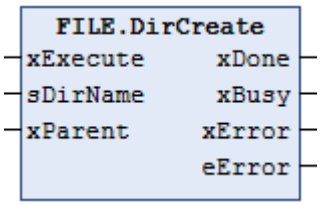
11.8 SD Card Operation (Directory Operation)

Directories in the SD card inserted in the SD memory card slot can be operated.

11.8.1 FILE.DirCreate (Create Directory)

This is a function block that creates a directory. An error occurs if there already exists a sub-directory.

■ **Icon**



■ **Parameter**

Scope	Name	Type	Description
Input	xExecute	BOOL	Starts execution at the rising edge.
	sDirName	FILE.CAA.FILENAME	Specifies a new directory name with an absolute path or relative path.
	xParent	BOOL	TRUE: Automatically creates a non-existing sub-directory. FALSE: An error occurs if there already exists a sub-directory.
Output	xDone	BOOL	TRUE: Execution is completed.
	xBusy	BOOL	TRUE: Execution of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	eError	FILE.ERROR	An error ID is output. Refer to "11.7.15 FILE.ERROR (Error ID)".

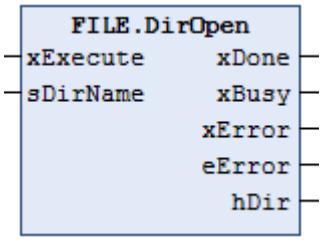
Info.

- You cannot use full size characters and the following symbols in a directory name: [\], [/], [:], [*], [?], [”], [<], [>], [|].

11.8.2 FILE.DirOpen (Open Directory)

This is a function block that opens a directory.

■ **Icon**



■ **Parameter**

Scope	Name	Type	Description
Input	xExecute	BOOL	Starts execution at the rising edge.
	sDirName	FILE.CAA.FILENAME	Specifies a directory name with an absolute path or relative path.
Output	xDone	BOOL	TRUE: Execution is completed.
	xBusy	BOOL	TRUE: Execution of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	eError	FILE.ERROR	An error ID is output. Refer to "11.7.15 FILE.ERROR (Error ID)".
	hDir	FILE.CAA.HANDLE	Handle of the FILE.CAA.HANDLE directory

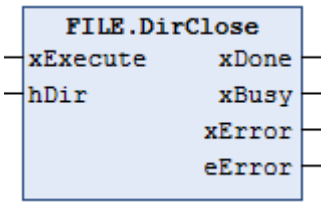
Info.

- You cannot use full size characters and the following symbols in a directory name: [\], [/], [:], [*], [?], [”], [<], [>], [|].

11.8.3 FILE.DirClose (Close Directory)

This is a function block that closes a directory.

■ **Icon**



■ **Parameter**

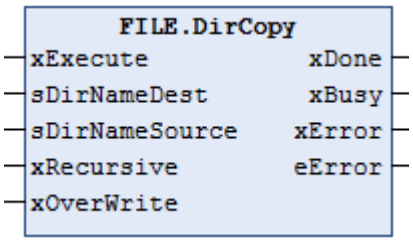
Scope	Name	Type	Description
Input	xExecute	BOOL	Starts execution at the rising edge.

Scope	Name	Type	Description
	hFile	FILE.CAA.FILENAME	Handle of the directory to be closed Specifies the handle output by FILE.Open.
Output	xDone	BOOL	TRUE: Execution is completed.
	xBusy	BOOL	TRUE: Execution of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	eError	FILE.ERROR	An error ID is output. Refer to "11.7.15 FILE.ERROR (Error ID)".

11.8.4 FILE.DirCopy (Copy Directory)

This is a function block that copies a directory.

■ Icon



■ Parameter

Scope	Name	Type	Description
Input	xExecute	BOOL	Starts execution at the rising edge.
	sDirNameDest	FILE.CAA.FILENAME	Directory name of the copy destination
	sDirNameSource	FILE.CAA.FILENAME	Directory of the copy source
	xRecursive	BOOL	TRUE: Copies the sub-directory and files.
	xOverWrite	BOOL	TRUE: Copies to overwrite an existing file.
Output	xDone	BOOL	TRUE: Execution is completed.
	xBusy	BOOL	TRUE: Execution of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	eError	FILE.ERROR	An error ID is output. Refer to "11.7.15 FILE.ERROR (Error ID)".

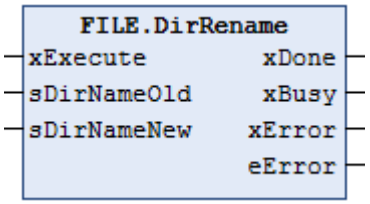
Info.

- You cannot use full size characters and the following symbols in a directory name: [\], [/], [:], [*], [?], ["], [<], [>], [|].

11.8.5 FILE.DirRename (Rename Directory)

This is a function block that renames a directory name. It is not possible to change the directory name of a directory that is currently open. Close it using the DirClose function block.

■ Icon



■ Parameter

Scope	Name	Type	Description
Input	xExecute	BOOL	Starts execution at the rising edge.
	sDirNameOld	FILE.CAA.FILENAME	Directory name before change
	sDirNameNew	FILE.CAA.FILENAME	Directory name after change
Output	xDone	BOOL	TRUE: Execution is completed.
	xBusy	BOOL	TRUE: Execution of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	eError	FILE.ERROR	An error ID is output. Refer to "11.7.15 FILE.ERROR (Error ID)".

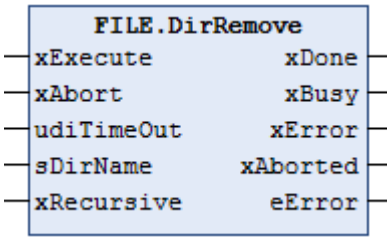
Info.

- You cannot use full size characters and the following symbols in a directory name: [\], [/], [:], [*], [?], [”], [<], [>], [|].

11.8.6 FILE.DirRemove (Delete Directory)

This is a function block that deletes a directory. It is not possible to delete a directory that is currently open. Close it using the DirClose function block.

■ **Icon**



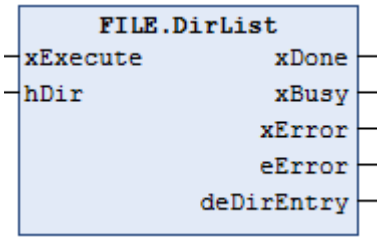
■ **Parameter**

Scope	Name	Type	Description
Input	xExecute	BOOL	Starts execution at the rising edge.
	xAbort	BOOL	TRUE: Stops execution and resets all outputs.
	udiTimeOut	UDINT	Timeout time until the execution is stopped (μs)
	sDirName	FILE.CAA.FILENAME	Specifies a directory name with an absolute path or relative path.
	xRecursive	BOOL	TRUE: Deletes the sub-directory and all files. FALSE: Deletes only when the directory is empty. An error occurs if the directory is not empty.
Output	xDone	BOOL	TRUE: Execution is completed.
	xBusy	BOOL	TRUE: Execution of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	xAborted	BOOL	TRUE: Execution is stopped by the user.
	eError	FILE.ERROR	An error ID is output. Refer to "11.7.15 FILE.ERROR (Error ID)".

11.8.7 FILE.DirList (Directory List)

This is a function block that outputs a list of directories and files inside the directory.

■ **Icon**



■ Parameter

Scope	Name	Type	Description
Input	xExecute	BOOL	Starts execution at the rising edge.
	hDir	FILE.CAA.HANDLE	Directory from which to output a list Specifies the handle output by FILE.Open.
Output	xDone	BOOL	TRUE: Execution is completed.
	xBusy	BOOL	TRUE: Execution of the FB is not completed.
	xError	BOOL	TRUE: An error has occurred within the FB.
	eError	FILE.ERROR	An error ID is output. Refer to "11.7.15 FILE.ERROR (Error ID)".
	deDirEntry	FILE.FILE_DIR_ENTRY	Files and directories are output.

■ FILE_DIR_ENTRY (Structure)

Member	Type	Description
sEntry	FILE.CAA.FILENAME	Directory or file name
szSize	FILE.CAA.SIZE	File size
xDirectory	BOOL	TRUE: Directory FALSE: File
xExclusive	BOOL	TRUE: Exclusive access mode FALSE: Multiple access mode
dtLastModification	DATE_AND_TIME	Last update date and time.

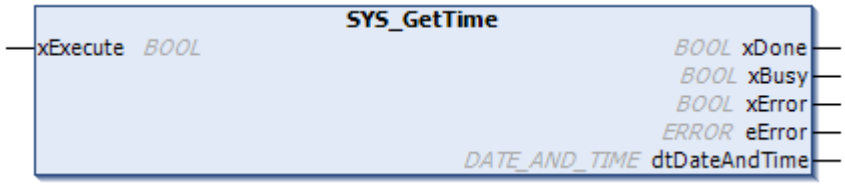
11.9 Clock Setting

This section describes function blocks that are used to set the clock of the GM1 Controller.
Enter a function block name by using the RTCLK (namespace).

11.9.1 SYS_GetTime (Get Time)

This is a function block (FB) that gets the current local time

■ **Icon**



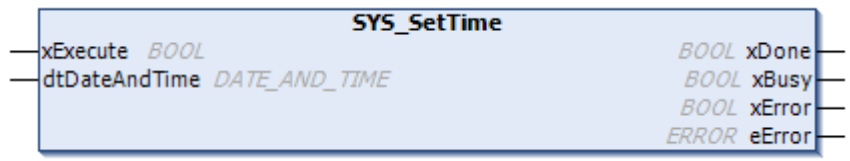
■ **Parameter**

Scope	Name	Type	Description
Input	xExecute	BOOL	TRUE: Active FALSE: Stop
Output	xDone	BOOL	TRUE: The function block is normally ended.
	xBusy	BOOL	TRUE: The function block is active.
	xError	BOOL	TRUE: An error has occurred.
	eError	ERROR	Details of error contents
	dtDateAndTime	DT	Current local time

11.9.2 SYS_SetTime (Set Time)

This is a function block (FB) that sets the current local time.

■ **Icon**



■ **Parameter**

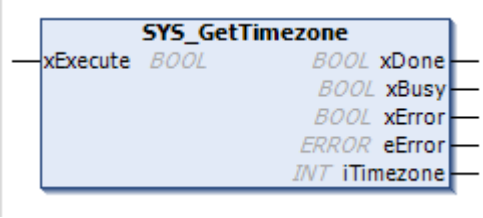
Scope	Name	Type	Description
Input	xExecute	BOOL	TRUE: Active

Scope	Name	Type	Description
			FALSE: Stop
	dtDateAndTime	DT	Current time to be set
Output	xDone	BOOL	TRUE: The function block is normally ended.
	xBusy	BOOL	TRUE: The function block is active.
	xError	BOOL	TRUE: An error has occurred.
	eError	ERROR	Details of error contents

11.9.3 SYS_GetTimezone (Get Time Zone Information)

This is a function block (FB) that gets the time zone information.

■ Icon



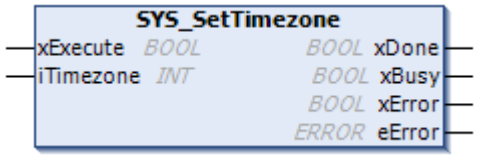
■ Parameter

Scope	Name	Type	Description
Input	xExecute	BOOL	TRUE: Active FALSE: Stop
Output	xDone	BOOL	TRUE: The function block is normally ended.
	xBusy	BOOL	TRUE: The function block is active.
	xError	BOOL	TRUE: An error has occurred.
	eError	ERROR	Details of error contents
	iTimezone	INT	Time zone information (Offset from UTC)

11.9.4 SYS_SetTimezone (Set Time Zone Information)

This is a function block (FB) that sets the time zone information.

■ **Icon**



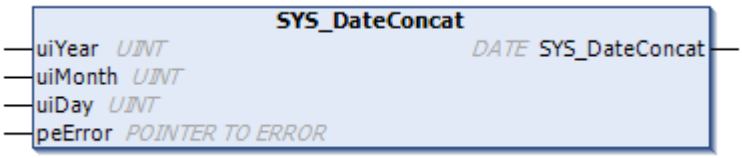
■ **Parameter**

Scope	Name	Type	Description
Input	xExecute	BOOL	TRUE: Active FALSE: Stop
	iTimezone	INT	Time zone information (Offset from UTC)
Output	xDone	BOOL	TRUE: The function block is normally ended.
	xBusy	BOOL	TRUE: The function block is active.
	xError	BOOL	TRUE: An error has occurred.
	eError	ERROR	Details of error contents

11.9.5 SYS_DateConcat (Convert from UINT Type to DATE Type)

This is a function (FUN) that converts a UINT type date to a DATE type.

■ **Icon**



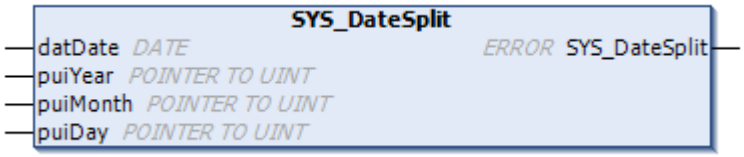
■ **Parameter**

Scope	Name	Type	Description
Input	uiYear	UINT	Year: 1970 to 2099
	uiMonth	UINT	Month: 1 to 12
	uiDay	UINT	Day: 1 to 31
	peError	POINTER TO ERROR	Pointer to the error information storage location
Output	SYS_DateConcat	DATE	Return value: Returns DT#1970-01-01 if the input value is invalid.

11.9.6 SYS_DateSplit (Convert from DATE Type to UINT Type)

This is a function (FUN) that converts a DATE type date to a UINT type.

■ Icon



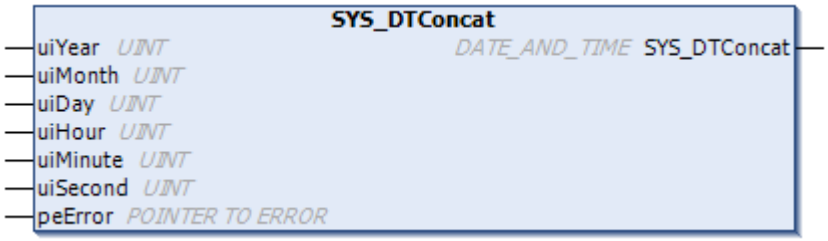
■ Parameter

Scope	Name	Type	Description
Input	datDate	DATE	Date data
	puiYear	POINTER TO UINT	Pointer to the year data storage location: 1970 to 2099
	puiMonth	POINTER TO UINT	Pointer to the month data storage location: 1 to 12
	puiDay	POINTER TO UINT	Pointer to the day data storage location: 1 to 31
Output	SYS_DateSplit	ERROR	Return value: Error information

11.9.7 SYS_DTConcat (Convert from UINT Type to DT Type)

This is a function (FUN) that converts a UINT type date and time to a DT type.

■ Icon



■ Parameter

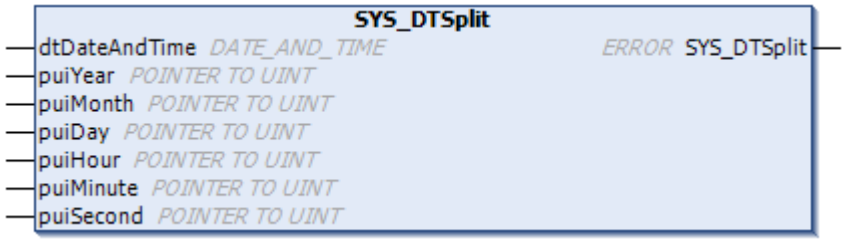
Type	Name	Type	Description
Input	uiYear	UINT	Year: 1970 to 2099
	uiMonth	UINT	Month: 1 to 12
	uiDay	UINT	Day: 1 to 31

Type	Name	Type	Description
	uiHour	UINT	Hour: 0 to 23
	uiMinute	UINT	Minute: 0 to 59
	uiSecond	UINT	Second: 0 to 59
	peError	POINTER TO ERROR	Pointer to the error information
Output	SYS_DTConcat	DT	Return value: Returns DT#1970-01-01-00:00:00 if the input value is invalid.

11.9.8 SYS_DTSplit (Convert from UINT Type to DT Type)

This is a function (FUN) that converts a UINT type date and time to a DT type.

■ Icon



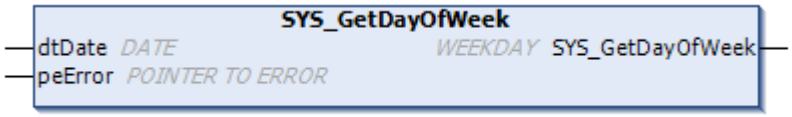
■ Parameter

Scope	Name	Type	Description
Input	dtDateAndTime	DT	Date and time data
	uiYear	POINTER TO UINT	Pointer to the year data storage location: 1970 to 2099
	uiMonth	POINTER TO UINT	Pointer to the month data storage location: 1 to 12
	uiDay	POINTER TO UINT	Pointer to the day data storage location: 1 to 31
	uiHour	POINTER TO UINT	Pointer to the hour data storage location: 0 to 23
	uiMinute	POINTER TO UINT	Pointer to the minute data storage location: 0 to 59
	uiSecond	POINTER TO UINT	Pointer to the second data storage location: 0 to 59
Output	SYS_DTSplit	ERROR	Return value: Error information

11.9.9 SYS_GetDayOfWeek (Get Day of the Week)

This is a function (FUN) that gets the day of the week from the DATE type date.

■ Icon



■ Parameter

Scope	Name	Type	Description
Input	dtDate	DATE	Date data
	peError	POINTER TO ERROR	Pointer to the error information
Output	SYS_GetDayOfWeek	RTCLK.WEEKDAY	Return value: Day of the week

RTCLK.WEEKDAY (Day of the week)

Name	Value	Description
SUNDAY	16#00	Sunday
MONDAY	16#01	Monday
TUESDAY	16#02	Tuesday
WEDNESDAY	16#03	Wednesday
THURSDAY	16#04	Thursday
FRIDAY	16#05	Friday
SATURDAY	16#06	Saturday

11.9.10 SYS_TODConcat (Convert from UINT Type to TOD Type)

This is a function (FUN) that converts a UINT type time with milliseconds to a TOD type.

■ Icon

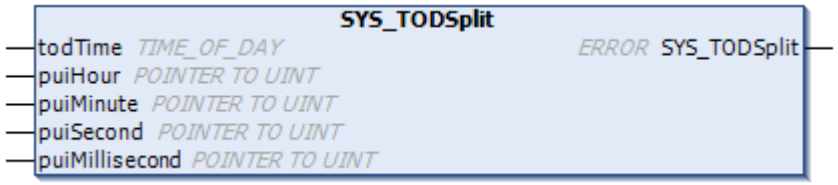
■ **Parameter**

Scope	Name	Type	Description
Input	uiHour	UINT	Hour: 0 to 23
	uiMinute	UINT	Minute: 0 to 59
	uiSecond	UINT	Second: 0 to 59
	uiMillisecond	UINT	Millisecond: 0 to 999
	peError	POINTER TO ERROR	Pointer to the error information
Output	SYS_TODConcat	TOD	Return value Returns TOD#00:00:00 if the input value is invalid.

11.9.11 SYS_TODSplit (Convert from TOD Type to UINT Type)

This is a function (FUN) that converts a TOD type time with milliseconds to a UINT type.

■ **Icon**



■ **Parameter**

Scope	Name	Type	Description
Input	todTime	TOD	Time (hour, minute, second) with millisecond data
	puiHour	POINTER TO UINT	Pointer to the hour data storage location: 0 to 23
	puiMinute	POINTER TO UINT	Pointer to the minute data storage location: 0 to 59
	puiSecond	POINTER TO UINT	Pointer to the second data storage location: 0 to 59
	puiMillisecond	POINTER TO UINT	Pointer to the millisecond data storage location: 0 to 999
Output	SYS_TODSplit	ERROR	Return value: Error information

11.9.12 ERROR (Clock Instruction Error Code)

Name	Value	Description
NO_ERROR	0	No error
FIRST_ERROR	5700	First error unique to the library
TIME_OUT	5751	The time limit is exceeded.
NOT_AVAILABLE	5752	Not available.
INPUT_VALID	5753	Invalid input value
DTU_ERROR_UNKNOWN	5754	Unknown error
DTU_WRONG_PARAMETER	5755	Wrong parameter
DTU_TZI_NOT_SET	5756	The time zone information has not been initialized.
FIRST_MF	5770	First error unique to the manufacturer
LAST_ERROR	5799	Last error unique to the library

11.10 System Data

11.10.1 SYS_GetSystemError (Get System Error)

This is a function block that gets the information of a system error that has occurred in the GM1 Controller.

■ **Icon**

■ **Parameter**

Scope	Name	Type	Description
Input	None		
Output	Error	BOOL	TRUE: An error has occurred.
	ErrorId	UDINT	Error ID of the error that has occurred

Info.

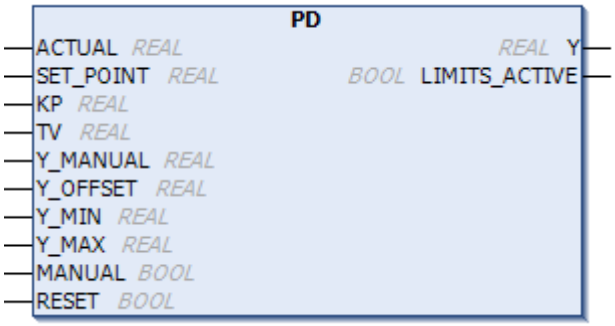
- For the error IDs, refer to the *GM1 Controller RTEX User's Manual (Operation Edition)* or *GM1 Controller EtherCAT User's Manual (Operation Edition)*.

11.11 PID Control

11.11.1 PD (PD Control)

This is a function block (FB) that performs PD control. P control can be performed when TV is set to 0.

■ Icon



■ Parameter

Scope	Name	Type	Default	Description
Input	ACTUAL	REAL	-	Current value
	SET POINT	REAL	-	Target value
	KP	REAL	-	Proportionality constant P
	TV	REAL	-	Derivative time D (unit: s)
	Y_MANUAL	REAL	-	A value output to output value (Y) when MANUAL = TRUE is set.
	Y_OFFSET	REAL	-	An offset value of output value (Y)
	Y_MIN	REAL	-	A lower limit value of output value (Y)
	Y_MAX	REAL	-	An upper limit value of output value (Y)
	MANUAL	BOOL	-	TRUE: The value set in Y_MANUAL is output to output value (Y).
	RESET	BOOL	-	TRUE: Output value (Y) is reset. Output value (Y) is set to the offset value (Y_OFFSET) to reset the integral portion.
Output	Y	REAL	-	Output value
	LIMITS_ACTIVE	BOOL	FALSE	TRUE: Output value (Y) is outside the range defined by Y_MIN/Y_MAX.

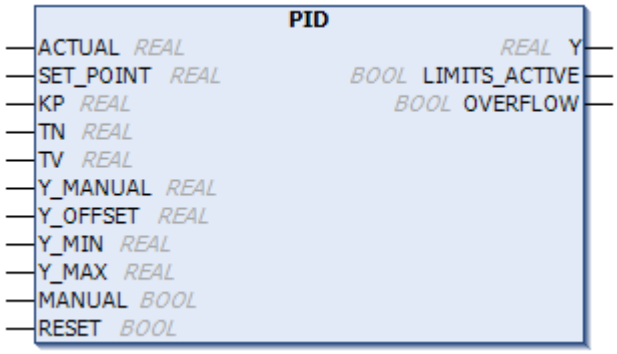
11.11.2 PID (PID Control)

This is a function block (FB) that performs PID control. Cycle time is automatically measured and PID operation is executed. PI control can be performed when TV is set to 0.

Info.

- Cycle time is a time passed while the FB is called twice.

■ Icon



■ Parameter

Scope	Name	Type	Default	Description
Input	ACTUAL	REAL	-	Current value
	SET POINT	REAL	-	Target value
	KP	REAL	-	Proportionality constant P
	TN	REAL	-	Integral time I (unit: s)
	TV	REAL	-	Derivative time D (unit: s)
	Y_MANUAL	REAL	-	A value output to output value (Y) when MANUAL = TRUE is set.
	Y_OFFSET	REAL	-	An offset value of output value (Y)
	Y_MIN	REAL	-	A lower limit value of output value (Y)
	Y_MAX	REAL	-	An upper limit value of output value (Y)
	MANUAL	BOOL	-	TRUE: The value set in Y_MANUAL is output to output value (Y).
	RESET	BOOL	-	TRUE: Output value (Y) is reset. Output value (Y) is set to the offset value (Y_OFFSET) to reset the integral portion.
Output	Y	REAL	-	Output value
	LIMITS_ACTIVE	BOOL	FALSE	TRUE: Output value (Y) is outside the range defined by Y_MIN/Y_MAX.

Scope	Name	Type	Default	Description
	OVERFLOW	BOOL	-	TRUE: Integer portion overflow

Info.

- The maximum preciseness is 1 ms and thus the precision decreases when the operation is executed in a short cycle time.

  For instance, if the cycle time is 1 ms, PID control measures the cycle time as 2 ms or 0 ms in some cases.

  If the operation is executed in a short cycle time, use of PID_FIXCYCLE is recommended.

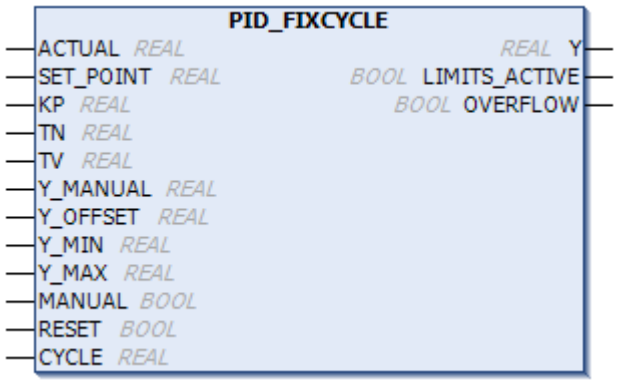
11.11.3 PID_FIXCYCLE [PID Control (Any Cycle Time)]

This is a function block (FB) that performs PID control. Cycle time can be manually set. PID operation is executed over the set cycle time. Except for the manually set cycle time, this FB operates in the same way as PID function blocks.

Info.

- Cycle time is a time passed while the FB is called twice.

■ Icon



■ Parameter

Scope	Name	Type	Default	Description
Input	ACTUAL	REAL	-	Current value
	SET POINT	REAL	-	Target value
	KP	REAL	-	Proportionality constant P
	TN	REAL	-	Integral time I (unit: s)
	TV	REAL	-	Derivative time D (unit: s)

Scope	Name	Type	Default	Description
	Y_MANUAL	REAL	-	A value output to output value (Y) when MANUAL = TRUE is set.
	Y_OFFSET	REAL	-	An offset value of output value (Y)
	Y_MIN	REAL	-	A lower limit value of output value (Y)
	Y_MAX	REAL	-	An upper limit value of output value (Y)
	MANUAL	BOOL	-	TRUE: The value set in Y_MANUAL is output to output value (Y).
	RESET	BOOL	-	TRUE: Output value (Y) is reset. Output value (Y) is set to the offset value (Y_OFFSET) to reset the integral portion.
	CYCLE	REAL	-	A time passed while the FB is called twice
Output	Y	REAL	-	Output value
	LIMITS_ACTIVE	BOOL	FALSE	TRUE: Output value (Y) is outside the range defined by Y_MIN/Y_MAX.
	OVERFLOW	BOOL	-	TRUE: Integer portion overflow

11.12 Recipe function

Variables in each recipe definition added to the Recipe Manager can be manipulated as recipes with the Recipe Method command.

The recipe method is affected by the “Storage” and “General” tabs.

If these are not configured correctly, the function will not behave properly.

■ Recipe method command description method

Each recipe method command is a method belonging to the function block RecipeManCommands.

Therefore, an item that sets RecipeManCommands as an instance must be written at the beginning of the method.

- Declaration section

```
RecipeManCommands_0   : RecipeManCommands;
```

- Implement section

```
output := RecipeManCommands_0.CreateRecipe(input1 , input2);
```

■ Terminology

- Recipe (GM Programmer tools)

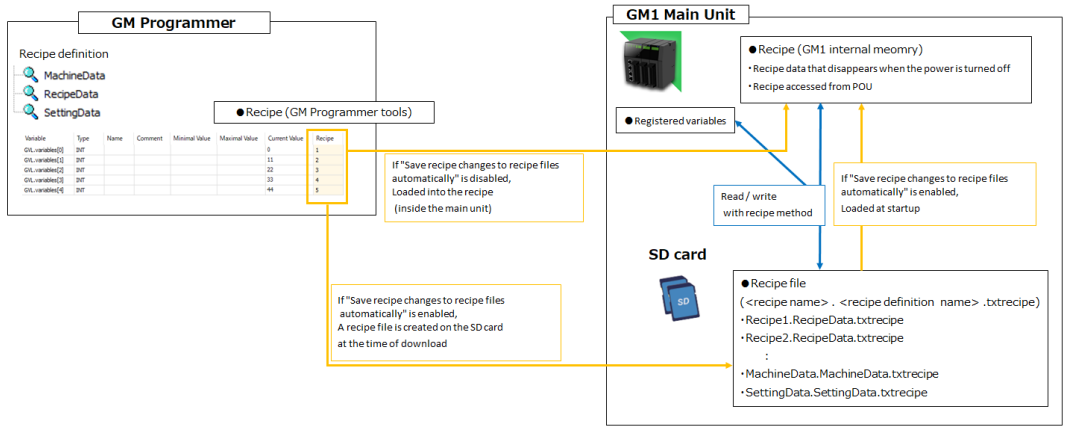
  A common file format for recipes handled by the GM Programmer recipe feature. The format of the standard recipe file is <Recipe Name>.<Recipe Definition Name>.<Recipe Extension>. The format for automatic saving in Recipe Manager is also the recipe file format.The contents follow the settings of the “Storage” tab.

- Recipe File

  The file format of recipes handled by the GM Programmer recipe function.The recipe file format is <Recipe Name<.<Recipe Extension<. By setting <Recipe Name< to <Recipe Name<.<Recipe Definition Name<, you can also handle recipe files.

- Recipe (GM1 internal meomry)

  Refers to a recipe that was created at runtime in the GM1 Main Unit.This recipe is not automatically saved and is deleted by resetting or turning on and off. If “Automatically save changes” in the Recipe Manager is enabled, a recipe file is created base on the Recipe(in SD card) when the user logs in to the GM1 Main Unit or when ReloadRecipes is run. In recipe commands, “recipe” refers mainly to the Recipe (GM1 internal meomry). The format of the recipe file is <recipe name>.<Recipe definition name>.<Recipe extension>.



Info.

- Recipes created in the tool cannot be rewritten directly from POU.
- Basically runtime recipes are manipulated by POU.
- If you want to handle the contents of a recipe created in the tool in POU, it is done via a recipe file or a runtime recipe saved automatically in the SD card.

■ Recipe file character limit

In GM Programmer tool, Recipe files has a limited number of characters. If set outside of these ranges, it may not work correctly.

The character limit for each item is as follows:

Setting item	Limit value
Recipe Name	Specify up to 35 characters.
Recipe definition name	Specify up to 35 characters.
Extension	Please enter no more than 10 characters, including periods.
File path name	Please enter no more than 176 characters, including the “\” symbol.

11.12.1 CreateRecipe (Create Recipe)

A method that creates a new recipe in the specified recipe definition from the "current value" and saves it as a recipe file. In addition, it is not possible to overwrite an existing recipe or specify the same name. An error occurs.

■ Icon

■ Parameter

Scope	Name	Type	Description
Input	RecipeDefinitionName	STRING	Recipe definition name for the recipe to be created (Note 1)
	RecipeName	STRING	The name of the recipe to be created (Note 2)
Output	CreateRecipe	DWORD	Output runtime ReturnValues

(Note 1) Be sure to specify the target recipe definition before operating.

(Note 2) Be sure to specify the recipe name to be created before operating.

■ Program example

A program that reads the “current value” of the recipe definition corresponding to the input variable input1 and creates a recipe with the recipe name of the input variable input2 and a recipe file.

- Declaration section

```
RecipeManCommands_0    : RecipeManCommands;
input1                 : STRING := 'RecipeDefName';
input2                 : STRING := 'RecipeName';
```

- Implement section

```
output := RecipeManCommands_0.CreateRecipe(input1 , input2);
```

■ Recipe Manager settings

The settings on the General tab of the Recipe Manager affect the following.

Setting item		Overview
Recipe management within PLC	-	If enabled, the RecipeManCommands method will be executable.
Save Recipe	Save recipe changes to recipe files automatically	If enabled, recipes created by the CreateRecipe method will automatically be saved to the SD card as recipe files.If this setting is disabled, recipes are created at runtime only and are not stored in recipe files.
Load Recipe	Load only exact matches in the variable list	The CreateRecipe method is not affected.
	Load matching variables by the variable name	The CreateRecipe method is not affected.
Write Recipe	Limit variables to minimum/maximum when the recipe value is out of range	The CreateRecipe method is not affected.
	Do not write to a variable if the recipe value is outside the minimum/maximum range	The CreateRecipe method is not affected.
Read Recipe	Check the recipe changes	The CreateRecipe method is not affected.

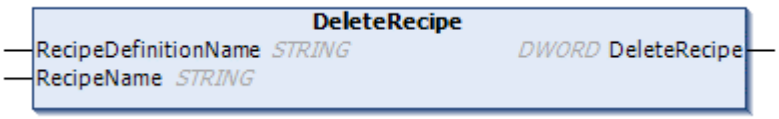
Info.

- If the SD card is not inserted into the GM1 Main Unit, a recipe file is not created, but no error occurs.
- When SD card write protection is enabled, a recipe file is not created, but no error occurs.
- If there is no free space on the SD card, no error occurs, but an empty recipe file is created.
- Since no error occurs in the above three cases, if you want to create a recipe file, check the size of the recipe file (FILE.GetSize) after it is generated.

11.12.2 DeleteRecipe (Delete Recipe)

This is a method to delete a recipe for the specified recipe definition. If a corresponding recipe file exists, the file is also deleted.

■ Icon



■ Parameter

Scope	Name	Type	Description
Input	RecipeDefinitionName	STRING	Recipe definition name for the recipe to be deleted (Note 1)
	RecipeName	STRING	The name of the recipe to be deleted (Note 2)
Output	DeleteRecipe	DWORD	Output runtime ReturnValues

(Note 1) Be sure to specify the target recipe definition before operating.

(Note 2) Be sure to specify the recipe name to be deleted before operating.

■ Program example

This is a program that deletes the recipes present in the recipe definition that correspond to the input variables input1 and input2.

- Declaration section

```
RecipeManCommands_0   : RecipeManCommands;
input1                : STRING := 'RecipeDefName';
input2                : STRING := 'RecipeName';
```

- Implement section

```
output := RecipeManCommands_0.DeleteRecipe(input1 , input2);
```

■ Recipe Manager settings

The settings on the General tab of the Recipe Manager affect the following.

Setting item		Overview
Recipe management within PLC	-	If enabled, the RecipeManCommands method will be executable.
Save Recipe	Save recipe changes to recipe files automatically	If this setting is disabled, DeleteRecipe will not delete recipe files.
Load Recipe	Load only exact matches in the variable list	The DeleteRecipe method is not affected.
	Load matching variables by the variable name	The DeleteRecipe method is not affected.
Write Recipe	Limit variables to minimum/maximum when the recipe value is out of range	The DeleteRecipe method is not affected.
	Do not write to a variable if the recipe value is outside the minimum/maximum range	The DeleteRecipe method is not affected.
Read Recipe	Check the recipe changes	The DeleteRecipe method is not affected.

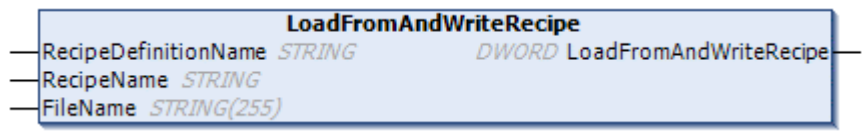
Info.

- If an SD card is not inserted into the GM1 Main Unit, the recipe file is not deleted and an error occurs.
- When SD card write protection is enabled, the recipe file is not deleted and an error occurs.

11.12.3 LoadFromAndWriteRecipe (Load and Write Recipe File)

Load the recipe from the specified recipe file. This is a method that then writes the recipe value to the corresponding recipe and “current value”. If the item “Variable” is not listed in the recipe file to be loaded, it will not be loaded correctly.

■ Icon



■ Parameter

Scope	Name	Type	Description
Input	RecipeDefinitionName	STRING	Recipe definition name for the recipe to be loaded (Note 1)
	RecipeName	STRING	The name of the recipe to be loaded (Note 2)
	FileName	STRING(255)	Name of the recipe file to be loaded (Note 3)
Output	LoadFromAndWriteRecipe	DWORD	Output runtime ReturnValues

(Note 1) Be sure to specify the target recipe definition before operating.

(Note 2) Be sure to specify the recipe name to be loaded before operating.

(Note 3) Be sure to specify the target recipe file before operating.

■ Program example

Load the recipe file corresponding to the input variable input3 for the recipe present in the recipe definition corresponding to the input variables input1 and input2. This is a program that writes that value to the “current value” of the recipe and recipe definition.

- Declaration section

```
RecipeManCommands_0   : RecipeManCommands;
input1                : STRING := 'RecipeDefName';
input2                : STRING := 'RecipeName';
input3                : STRING := 'RecipeName.RecipeDefName.txtrecipe';
```

- Implement section

```
output := RecipeManCommands_0.LoadFromAndWriteRecipe(input1 , input2 , inpu
t3);
```

■ Recipe Manager settings

The settings on the General tab of the Recipe Manager affect the following.

Setting item		Overview
Recipe management within PLC	-	If enabled, the RecipeManCommands method will be executable.
Save Recipe	Save recipe changes to recipe files automatically	If this setting is disabled, the recipe must be executed after it has been created in runtime.
Load Recipe	Load only exact matches in the variable list	The LoadFromAndWriteRecipe method can only be executed if the contents of the recipe file to be loaded match all of the settings on the Storage tab and the variable names in the recipe definition.
	Load matching variables by the variable name	When the LoadFromAndWriteRecipe method is executed, only variables that match the variable names in the recipe file to be loaded and the variable names in the recipe definition are written.
Write Recipe (Note 1)	Limit variables to minimum/maximum when the recipe value is out of range	When writing a recipe value to the “current value” in the recipe definition, if the value is outside the minimum/ maximum value range specified in the recipe definition, the minimum or maximum value set in the recipe definition will be written instead.
	Do not write to a variable if the recipe value is outside the minimum/maximum range	When writing a recipe value to the “current value” in the recipe definition, if the value is outside the minimum/ maximum value range specified in the recipe definition, the value is not written and the “current value” is preserved.
Read Recipe	Check the recipe changes	The LoadFromAndWriteRecipe method is not affected.

(Note 1) When writing recipe values to “current value” using this function, be sure to set the maximum and minimum values for all variables that can be set. It will not work correctly if you only set it partially. If you want to write all variables regardless of the maximum and minimum values, do not set the maximum and minimum values for all variables.

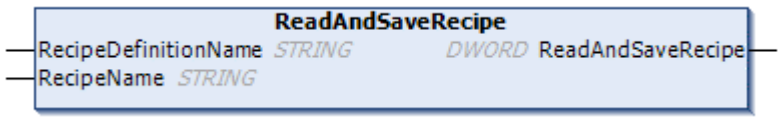
Info.

- If an SD card is not inserted into the GM1 Main Unit, an error occurs.

11.12.4 ReadAndSaveRecipe (Recipe File Overwrite Save)

This is a method that loads the “current value” into the recipe and then saves the recipe to a recipe file. If a recipe file with the same name already exists, it is overwritten and saved. To execute the ReadAndSaveRecipe method, a recipe must be created, for example, by CreateRecipe.

■ Icon



■ Parameter

Scope	Name	Type	Description
Input	RecipeDefinitionName	STRING	Recipe definition name for the recipe to be read (Note 1)
	RecipeName	STRING	The name of the recipe to be saved (Note 2)
Output	ReadAndSaveRecipe	DWORD	Output runtime ReturnValues

(Note 1) If there is no free space on the SD card, no error occurs, but an empty recipe file is created.

(Note 2) Be sure to specify the target recipe name before operating.

■ Program example

This program reads the current value of the recipe definition corresponding to the input variable input1 into the recipe of input variable input2 and saves it in a recipe file.

- Declaration section

```
RecipeManCommands_0   : RecipeManCommands;
input1                : STRING := 'RecipeDefName';
input2                : STRING := 'RecipeName';
```

- Implement section

```
output := RecipeManCommands_0.ReadAndSaveRecipe(input1 , input2);
```

■ Recipe Manager settings

The settings on the General tab of the Recipe Manager affect the following.

Setting item		Overview
Recipe management within PLC	-	If enabled, the RecipeManCommands method will be executable.
Save Recipe	Save recipe changes to recipe files automatically	If this setting is disabled, the behavior will vary depending on the “Check for recipe changes” setting. ● “Check for recipe changes” is disabled: A recipe file is created on the SD card. ● “Check for recipe changes” is enabled: A recipe file is created on the SD card only when it is determined that the overwrite save has occurred due to a change in the recipe value. If this setting is enabled, follow the “Check for recipe changes” setting.
Load Recipe	Load only exact matches in the variable list	The ReadAndSaveRecipe method is not affected.
	Load matching variables by the variable name	The ReadAndSaveRecipe method is not affected.
Write Recipe	Limit variables to minimum/maximum when the recipe value is out of range	The ReadAndSaveRecipe method is not affected.
	Do not write to a variable if the recipe value is outside the minimum/maximum range	The ReadAndSaveRecipe method is not affected.
Read Recipe	Check the recipe changes	If you enable this setting, the ReadAndSaveRecipe method first loads the “current value” into the recipe when it runs.

Info.

- If an SD card is not inserted into the GM1 Main Unit, the recipe file is not overwritten and saved and an error occurs.
- When SD card write protection is enabled, the recipe file is not overwritten and saved and an error occurs.
- If there is no free space on the SD card, no error occurs, but an empty recipe file is created.

11.12.5 prvCompareRecipe (Compare Recipes)

This is a method to compare the “current value” of the recipe definition with the specified recipe (GM1 internal meomry). It can be executed when the “Check for recipe changes” setting in the Recipe Manager is enabled. Since the prvCompareRecipe method belongs to Private, an error occurs when the function block RecipeManCommands is executed before operation.

■ Icon

■ Parameter

Scope	Name	Type	Description
Input	RecipeDefinitionName	STRING	Recipe definition name to be compared (Note 1)
	RecipeName	STRING	The name of the recipe to be compared (Note 2)
Output	prvCompareRecipe	BOOL	If the recipe matches the “current value” of the recipe definition, TRUE is returned

(Note 1) Be sure to specify the target recipe definition before operating.

(Note 2) Be sure to specify the recipe name to be compared before operating.

■ Program example

This is a program that compares the value of the recipe of input variable input2 present in the recipe definition corresponding to input variable input1 with the “current value”.

- Declaration section

```
RecipeManCommands_0   : RecipeManCommands;
input1                : STRING := 'RecipeDefName';
input2                : STRING := 'RecipeName';
```

- Implement section

```
output := RecipeManCommands_0.ReloadRecipes(input1);
output := RecipeManCommands_0.prvCompareRecipe(input1 , input2);
```

■ Recipe Manager settings

The settings on the General tab of the Recipe Manager affect the following.

Setting item		Overview
Recipe management within PLC	-	If enabled, the RecipeManCommands method will be executable.
Save Recipe	Save recipe changes to recipe files automatically	If this setting is disabled, the recipe must be executed after it has been created in runtime.
Load Recipe	Load only exact matches in the variable list	Verify that the settings in the Storage tab and the variable names in the recipe definition exactly match the contents of the recipe file to be compared.
	Load matching variables by the variable name	
Write Recipe	Limit variables to minimum/maximum when the recipe value is out of range	The prvCompareRecipe method is not affected.

Setting item		Overview
	Do not write to a variable if the recipe value is outside the minimum/maximum range	The prvCompareRecipe method is not affected.
Read Recipe	Check the recipe changes	If enabled, the prvCompareRecipe method becomes executable.

Info.

- Please execute ReloadRecipes (reload of the recipe file in the SD card) and reflect it in the Recipe(GM1 internal meomry).If another method is executed before this method, an error will occur.

11.12.6 ReloadRecipes (Load Recipe File in SD Card)

This is a method to load the recipe file saved in the SD card for the specified recipe definition. The loaded recipe file is saved as recipe(GM1 internal meomry). This is necessary to access the recipe file in the card correctly, such as when the SD card is inserted and disconnected during GM1 operation.

■ Icon

■ Parameter

Scope	Name	Type	Description
Input	RecipeDefinitionName	STRING	Recipe definition name to be reloaded (Note 1)
Output	ReloadRecipes	DWORD	Output runtime ReturnValues

(Note 1) Be sure to specify the target recipe definition before operating.

■ Program example

This is a program that reads the recipe file in the SD card that belongs to the recipe definition corresponding to the input variable input1.

- Declaration section

```
RecipeManCommands_0   : RecipeManCommands;
input1                : STRING := 'RecipeDefName';
```

- Implement section

```
output := RecipeManCommands_0.ReloadRecipes(input1);
```

■ Recipe Manager settings

The settings on the General tab of the Recipe Manager affect the following.

Setting item		Overview
Recipe management within PLC	-	If enabled, the RecipeManCommands method will be executable.
Save Recipe	Save recipe changes to recipe files automatically	If this setting is disabled, ReloadRecipes will not load the recipe file.
Load Recipe	Load only exact matches in the variable list	The ReloadRecipes method is not affected.
	Load matching variables by the variable name	The ReloadRecipes method is not affected.
Write Recipe	Limit variables to minimum/maximum when the recipe value is out of range	The ReloadRecipes method is not affected.
	Do not write to a variable if the recipe value is outside the minimum/maximum range	The ReloadRecipes method is not affected.
Read Recipe	Check the recipe changes	The ReloadRecipes method is not affected.

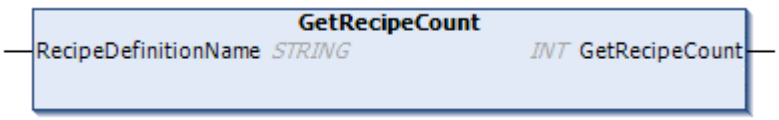
Info.

- If there is no data on the SD card inserted into GM1, or if the SD card is not inserted, no error occurs. Run GetRecipeCount, for example, to verify that the recipe file in the card was loaded correctly.

11.12.7 GetRecipeCount (Count Recipes)

This is a method to count the number of recipes belonging to the specified recipe definition.The count target is the recipes (GM1 internal meomry).

■ Icon



■ Parameter

Scope	Name	Type	Description
Input	RecipeDefinitionName	STRING	Recipe definition name that counts the number of recipes (Note 1)
Output	GetRecipeCount	INT	Output the number of recipes present in the recipe definition

(Note 1) Be sure to specify the target recipe definition before operating.

■ Program example

This is a program that counts the number of recipes present in the recipe definition corresponding to the input variable input1.

- Declaration section

```
RecipeManCommands_0    : RecipeManCommands;
input1                 : STRING := 'RecipeDefName';
```

- Implement section

```
output := RecipeManCommands_0.GetRecipeCount(input1);
```

■ Recipe Manager settings

The settings on the General tab of the Recipe Manager affect the following.

Setting item		Overview
Recipe management within PLC	-	If enabled, the RecipeManCommands method will be executable.
Save Recipe	Save recipe changes to recipe files automatically	The GetRecipeCount method is not affected.
Load Recipe	Load only exact matches in the variable list	The GetRecipeCount method is not affected.
	Load matching variables by the variable name	The GetRecipeCount method is not affected.
Write Recipe	Limit variables to minimum/maximum when the recipe value is out of range	The GetRecipeCount method is not affected.
	Do not write to a variable if the recipe value is outside the minimum/maximum range	The GetRecipeCount method is not affected.
Read Recipe	Check the recipe changes	The GetRecipeCount method is not affected.

11.12.8 GetRecipeNames (Get Recipe Names)

This is a method to acquire the name of the recipe that belongs to the specified recipe definition. The count target is the recipes (GM1 internal meomry).

■ Icon

■ Parameter

Scope	Name	Type	Description
Input	RecipeDefinitionName	STRING	Recipe definition name that acquires the recipe name (Note 1)
	pStrings	POINTER TO ARRAY OF STRING	Pointer to store the acquired recipe name (Note 2)
	iSize	INT	Number of recipes to acquire (Note 3) (Note 4)
	iStartIndex	INT	Index value of the recipe to acquire. (Note 4) Index value for recipe starting with 0.
Output	GetRecipeNames	DWORD	Output runtime ReturnValues

(Note 1) Be sure to specify the target recipe definition before operating.

(Note 2) The pointer array must be STRING (80) type. If you specify another type, the correct recipe name will not be acquired.

(Note 3) Do not set a value greater than the number of arrays set for pStrings. Failure to set the appropriate value may result in an unexpected error.

(Note 4) Be sure to set a value greater than or equal to 0 before operating. If you set a negative number, it will not work.

■ Program example

This is a program that acquires the recipe name from the input variable input4 +1th recipe for the number of input variables input3 for the recipe present in the recipe definition corresponding to the input variable input1.

- Declaration section

```
RecipeManCommands_0   : RecipeManCommands;
input1                : STRING := 'RecipeDefName';
input2                : ARRAY [0..9] OF STRING;
input3                : INT:=10;
input4                : INT:=0;
```

- Implement section

```
output := RecipeManCommands_0.GetRecipeNames(input1,ADR(input2),input3,inpu
t4);
```

■ Recipe Manager settings

The settings on the General tab of the Recipe Manager affect the following.

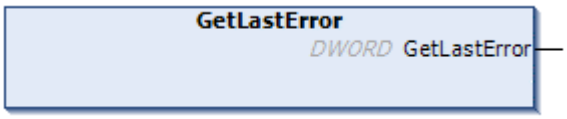
Setting item		Overview
Recipe management within PLC	-	If enabled, the RecipeManCommands method will be executable.
Save Recipe	Save recipe changes to recipe files automatically	The GetRecipeNames method is not affected.
Load Recipe	Load only exact matches in the variable list	The GetRecipeNames method is not affected.

Setting item		Overview
	Load matching variables by the variable name	The GetRecipeNames method is not affected.
Write Recipe	Limit variables to minimum/maximum when the recipe value is out of range	The GetRecipeNames method is not affected.
	Do not write to a variable if the recipe value is outside the minimum/maximum range	The GetRecipeNames method is not affected.
Read Recipe	Check the recipe changes	The GetRecipeNames method is not affected.

11.12.9 GetLastError (Get Last ReturnValues)

This is a method that acquires the ReturnValues value from the last action when using the recipe method.

■ Icon



■ Parameter

Scope	Name	Type	Description
Output	GetLastError	DWORD	Output the last output ReturnValues

■ ReturnValues (GVL)

Name	Number (DWORD)	Description
ERR_OK	16#0	Operation successful
ERR_RECIPE_FILE_NOT_FOUND	16#4000	Recipe file not found
ERR_RECIPE_MISMATCH	16#4001	The recipe file contents do not match the current recipe. Occurs only if the storage type is a string and the variable name in the file does not match the variable name in the recipe definition. The recipe file will not be loaded at this time.
ERR_RECIPE_SAVE_ERR	16#4002	The save operation failed due to the following reasons: ● The file cannot be created or opened because the SD card is full ● The configured file path does not exist ● The configured file extension is not allowed at runtime
ERR_RECIPE_NOT_FOUND	16#4003	The specified recipe does not exist
ERR_RECIPE_DEFINITION_NOT_FOUND	16#4004	The specified recipe definition does not exist

Name	Number (DWORD)	Description
ERR_RECIPE_ALREADY_EXIST	16#4005	The specified recipe already exists in the recipe definition
ERR_NO_RECIPE_MANAGER_SET	16#4006	“Recipe Management in PLC” is not enabled in the Recipe Manager
ERR_RECIPE_NOMEMORY	16#4008	The recipe definition does not have enough free memory to create a new recipe: ● More than 50 recipes were created in the recipe definition

■ Program example

This is a program that acquires the ReturnValues value obtained from the last recipe method operation executed.

● Declaration section

```
RecipeManCommands_0   : RecipeManCommands;
```

● Implement section

```
output := RecipeManCommands_0.GetLastError();
```

■ ■Recipe Manager settings

The settings on the General tab of the Recipe Manager affect the following.

Setting item		Overview
Recipe management within PLC	-	If enabled, the RecipeManCommands method will be executable.
Save Recipe	Save recipe changes to recipe files automatically	The GetLastError method is not affected.
Load Recipe	Load only exact matches in the variable list	The GetLastError method is not affected.
	Load matching variables by the variable name	The GetLastError method is not affected.
Write Recipe	Limit variables to minimum/maximum when the recipe value is out of range	The GetLastError method is not affected.
	Do not write to a variable if the recipe value is outside the minimum/maximum range	The GetLastError method is not affected.
Read Recipe	Check the recipe changes	The GetLastError method is not affected.

11.12.10 GetLastInfo (Get Last InfoValues)

This is a method that acquires the InfoValues value obtained from the last action when using the recipe method. It may be acquired automatically at the start of GM1 operation.

■ Icon

■ Parameter

Scope	Name	Type	Description
Output	GetLastInfo	InfoValues	Outputs the final output InfoValues (Note 1)

(Note 1) If multiple InfoValues occur simultaneously, the sum of the UDINT values for each Info is output.

■ InfoValues (Enumeration type)

Occurs when the Recipe Manager load recipe setting is “Load variables that match the variable name”

Name	Number (UDINT)	Description
NO_INFO	16#0	No Info occurrence
INFO_RECIPE_MANAGER_NOT_ALL_VARIABLES_FOUND	16#1	When loading a recipe file, some variables in “Variables” in the recipe definition are not in the recipe file
INFO_RECIPE_MANAGER_OTHER_VARIABLES_FOUND	16#2	When loading a recipe file, some variables in the recipe file are not in “variables” in the recipe definition
INFO_RECIPE_MANAGER_ONE_OR_MORE_VARIABLES_FOUND	16#3	When the recipe file was loaded, one or more values were written to “current value” in the recipe definition
INFO_RECIPE_MANAGER_ALL_VARIABLES_FOUND	16#4	When the recipe file was loaded, all values were written to “current value” in the recipe definition

■ Program example

This is a program that acquires the InfoValues value obtained from the last recipe method operation executed.

- Declaration section

```
RecipeManCommands_0   : RecipeManCommands;
```

- Implement section

```
output := RecipeManCommands_0.GetLastInfo();
```

■ Recipe Manager settings

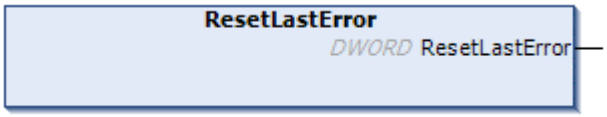
The settings on the General tab of the Recipe Manager affect the following.

Setting item		Overview
Recipe management within PLC	-	If enabled, the RecipeManCommands method will be executable.
Save Recipe	Save recipe changes to recipe files automatically	The GetLastInfo method is not affected.
Load Recipe	Load only exact matches in the variable list	The GetLastInfo method is not affected.
	Load matching variables by the variable name	The GetLastInfo method is not affected.
Write Recipe	Limit variables to minimum/maximum when the recipe value is out of range	The GetLastInfo method is not affected.
	Do not write to a variable if the recipe value is outside the minimum/maximum range	The GetLastInfo method is not affected.
Read Recipe	Check the recipe changes	The GetLastInfo method is not affected.

11.12.11 ResetLastError (GetLastError Reset)

This is a method to reset the value of GetLastError. It is used in conjunction with GetLastError.

■ Icon



■ Parameter

Scope	Name	Type	Description
Output	ResetLastError	DWORD	Output runtime ReturnValues

■ Program example

This is a program that resets the GetLastError value.

- Declaration section

```
RecipeManCommands_0   : RecipeManCommands;
```

- Implement section

```
output := RecipeManCommands_0.ResetLastError();
```

■ Recipe Manager settings

The settings on the General tab of the Recipe Manager affect the following.

Setting item		Overview
Recipe management within PLC	-	If enabled, the RecipeManCommands method will be executable.
Save Recipe	Save recipe changes to recipe files automatically	The ResetLastError method is not affected.
Load Recipe	Load only exact matches in the variable list	The ResetLastError method is not affected.
	Load matching variables by the variable name	The ResetLastError method is not affected.
Write Recipe	Limit variables to minimum/maximum when the recipe value is out of range	The ResetLastError method is not affected.
	Do not write to a variable if the recipe value is outside the minimum/maximum range	The ResetLastError method is not affected.
Read Recipe	Check the recipe changes	The ResetLastError method is not affected.

11.12.12 ResetLastInfo (GetLastInfo Reset)

This is a method to reset the value of GetLastInfo. It is used in conjunction with GetLastInfo.

■ Icon

■ Parameter

Scope	Name	Type	Description
Output	ResetLastInfo	DWORD	Output runtime ReturnValues

■ Program example

This is a program that resets the GetLastInfo value obtained when using the Recipe Method command.

- Declaration section

```
RecipeManCommands_0   : RecipeManCommands;
```

- Implement section

```
output := RecipeManCommands_0.ResetLastInfo();
```

■ Recipe Manager settings

The settings on the General tab of the Recipe Manager affect the following.

Setting item		Overview
Recipe management within PLC	-	If enabled, the RecipeManCommands method will be executable.
Save Recipe	Save recipe changes to recipe files automatically	The ResetLastInfo method is not affected.
Load Recipe	Load only exact matches in the variable list	The ResetLastInfo method is not affected.
	Load matching variables by the variable name	The ResetLastInfo method is not affected.
Write Recipe	Limit variables to minimum/maximum when the recipe value is out of range	The ResetLastInfo method is not affected.
	Do not write to a variable if the recipe value is outside the minimum/maximum range	The ResetLastInfo method is not affected.
Read Recipe	Check the recipe changes	The ResetLastInfo method is not affected.

(MEMO)

12 Function Blocks for the Pulse Output Unit

12.1 Basic Configuration of Function Blocks for the Pulse Output Unit

This section describes the basic configuration of the function block.

12.1.1 Specifications of the Function Block

■ Common parameters

The following table lists common parameters used by the function blocks for the GM1 Positioning Unit.

Scope	Parameter	Description
Input	UnitID	Specify the connection order for positioning units where function blocks are applied in a range from 1 to 15.
	AxisNo	Specify the No. of the axis where function blocks are applied in a range from 1 to 4.
	Execute	This is a trigger that executes the function block. ● It executes the function block at the rising edge when the Execute input becomes TRUE. ● When the Execute input is set to FALSE after a function block is processed, one of the following parameters is cleared: CommandAborted, Done, or Error. ● Even when the Execute input is set to FALSE while a function block is executed (Busy = TRUE), the function block keeps operating. At this time, when operation is finished, CommandAborted, Done or Error is held for one cycle.
	Enable	This enables an execution of a function block. ● It executes the function block at the rising edge when the Enable input becomes TRUE. ● There are some parameter that can be changed even when the Enable input is set to TRUE. Check this for every function block. ● When the Enable input is set to FALSE, the function block processing is stopped.
Output	Busy	The Busy output is set to TRUE while the function block is being executed.(Note 1)
	Done	The Done output is set to TRUE when the function block processing is completed.(Note 1)
	CommandAborted	When the function block processing is interrupted, the CommandAborted output is set to TRUE.(Note 1)
	Error	When an error occurs while the function block is executed, the Error output is set to TRUE.
	ErrorID	Error information can be checked when an error occurs.
	Valid	When an output becomes valid, this output becomes TRUE.

(Note 1) After a function block is processed, one of the following parameters is set to TRUE: CommandAborted, Done, or Error.

■ Tasks

Either MotionTask or UserTask can be executed.

Use the same task for performing a process for the same axis.

■ Simulation mode

The function block is not supported in the simulation mode. If a function block is executed in the simulation mode, the PG_NOT_SUPPORTED error is issued.

12.1.2 Notes for Executing the Function Block

■ Busy state

If the function block is busy (Busy = TRUE), call the function block at every cycle when executing a task.

Info.

- Exceptions, if any, will be described in the specifications for each function block.

■ Interruption of function block processing

- The following function blocks cannot be executed at the same time on the same axis: PG_Jog, PG_MoveRelative, PG_MoveAbsolute, PG_LatchPosition, PG_Home, and PG_Pulser

  The function block executed first takes precedence over the others.

  As for the function block that is executed later, CommandAborted is set to TRUE and processing is not started.
- PG_Stop takes precedence over any other control.

  If operation is stopped by executing PG_Stop, CommandAborted is set to TRUE for all other functions blocks and their controls are stopped
- If a stop operation (RUN → STOP) is applied in the middle of executing the function block, the pulse output unit stops outputting pulse signals.

  When operation is resumed, CommandAborted is set to TRUE for the function block being operated and its processing is interrupted.
- If an error occurs in a pulse output unit, the pulse output unit stops outputting pulse signals.

  For the function block that is being executed, the PG_AXIS_UNIT_ERROR is issued and its processing is interrupted.

■ Continuous execution of a function block

In a cycle where the Done output is set to TRUE, re-execute the function block (Execute = TRUE) to continue execution of the function block.

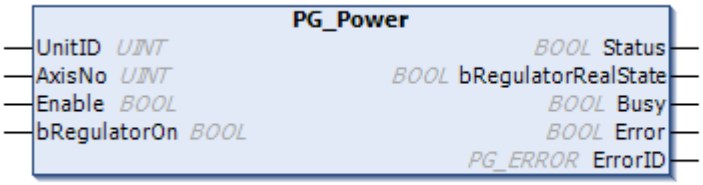
12.2 Function Blocks for the Pulse Output Unit

Various motion operations can be realized by using the function blocks for the pulse output unit. This section describes motion control function blocks for the single axis.

12.2.1 PG_Power

This is a function block (FB) that performs servo ON/OFF control on the axis of the pulse output unit.

■ Icon



■ Parameter

Scope	Parameter name	Type	Default	Description
Input	UnitID	UINT	-	Specifies the unit ID.
	AxisNo	UINT	-	Specifies the axis No.
	Enable	BOOL	FALSE	TRUE: Execution of the FB is enabled.
	bRegulatorOn	BOOL	FALSE	TRUE: Servo ON FALSE: Servo OFF
Output	Status	BOOL	FALSE	TRUE: The axis can be executed. FALSE: The axis cannot be executed.
	bRegulatorRealState	BOOL	FALSE	TRUE: Servo ON state FALSE: Servo OFF state
	Busy	BOOL	FALSE	TRUE: FB is in progress.
	Error	BOOL	FALSE	TRUE: FB is abnormally completed.
	ErrorID	PG_ERROR	NO_ERROR	An error ID is output.

■ Notes for Executing the Function Block

- After executing PG_Power, check the servo ON state using the bRegulatorRealState output parameter.
- It is not necessary to call PG_Power at every cycle.

12.2.2 PG_Jog

This is a function block (FB) that causes the axis of the pulse output unit to keep traveling in a forward or backward direction.

■ Icon

■ Parameter

Scope	Parameter name	Type	Default	Description
Input	UnitID	UINT	-	Specifies the unit ID.
	AxisNo	UINT	-	Specifies the axis No.
	JogForward	BOOL	FALSE	TRUE: Travels in a forward direction.
	JogBackward	BOOL	FALSE	TRUE: Travels in a backward direction.
	Velocity	LREAL	0	Specifies the target speed (u/s).
	AccDec	LREAL	0	Specifies the acceleration / deceleration (u/s^2).
Output	Done	BOOL	FALSE	TRUE: FB operation is completed.
	Busy	BOOL	FALSE	TRUE: FB operation is in progress.
	CommandAborted	BOOL	FALSE	TRUE: FB operation is interrupted.
	Error	BOOL	FALSE	TRUE: FB is abnormally completed.
	ErrorID	PG_ERROR	NO_ERROR	An error ID is output.

■ Operations when the function block is executed

- Set either JogForward or JogBackward to TRUE to execute the function block.
- Depending on the direction in which the axis is to be moved, set either JogForward or JogBackward to TRUE
- If JogForward and JogBackward are simultaneously executed, the PG_JOG_INVALID_REQUEST error is issued.

> - When executing again after the occurrence of the error, set both JogForward and JogBackward to FALSE once.

- When JogForward is switched to JogBackward (or JogBackward is switched to JogForward), the operation is switched only after the pulse output unit completes outputting pulse output currently controlled.

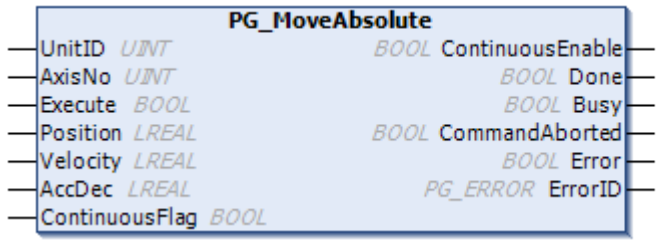
- The Velocity input can be changed in the middle of operation (in a situation where either JogForward or JogBackward is set to TRUE).
- The AccDec input cannot be changed during operation.

- When executing the function block, use the PG_Power function block in advance to set to the servo ON state.
- If the function block is executed in the servo OFF state, the PG_AXIS_NOT_READY_FOR_MOTION error is issued.

12.2.3 PG_MoveAbsolute

This is a function block (FB) that causes the axis of the pulse output unit to travel to a position specified as an absolute position.

■ Icon



■ Parameter

Scope	Parameter name	Type	Default	Description
Input	UnitID	UINT	-	Specifies the unit ID.
	AxisNo	UINT	-	Specifies the axis No.
	Execute	BOOL	FALSE	Starts execution at the rising edge.
	Position	LREAL	0	Specifies the target position (u).
	Velocity	LREAL	0	Specifies the maximum velocity (u/s).
	AccDec	LREAL	0	Specifies the acceleration / deceleration (u/s^2).
	ContinuousFlag	BOOL	FALSE	FALSE: E-point control TRUE: P-point control
Output	ContinuousEnable	BOOL	FALSE	TRUE: Position, Velocity, and AccDec can be changed.
	Done	BOOL	FALSE	TRUE: FB operation is completed.
	Busy	BOOL	FALSE	TRUE: FB operation is in progress.
	CommandAborted	BOOL	FALSE	TRUE: FB operation is interrupted.
	Error	BOOL	FALSE	TRUE: FB is abnormally completed.
	ErrorID	PG_ERROR	NO_ERROR	An error ID is output.

■ Operations when the function block is executed

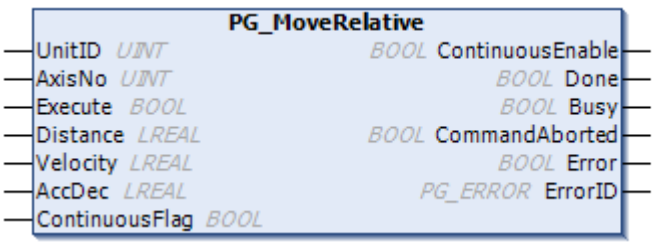
- Absolute value control is performed according to the position specified in the Position input. (Coordinates: -2147483648 to 2147483647u)
- To use the E-point control, specify as follows.
  - Specifies ContinuousFlag = FALSE and execute.
- To use the P-point control, specify as follows.
  - Specifies ContinuousFlag = TRUE and execute.
  - When the ContinuousEnable output becomes TRUE after PG_MoveAbsolute is started, the Position, Velocity, and AccDec inputs can be rewritten. Change the inputs using the rising edge when the Execute input becomes TRUE as a trigger.
  - If rewriting is performed when the ContinuousEnable output is FALSE, the PG_FB_IN_BUSY error is issued.
  - If rewriting is not executed, the execution of PG_MoveAbsolute is completed (Done=TRUE) when the pulse output is completed.

- When executing the function block, use the PG_Power function block in advance to set to the servo ON state.
- If the function block is executed in the servo OFF state, the PG_AXIS_NOT_READY_FOR_MOTION error is issued.

12.2.4 PG_MoveRelative

This is a function block (FB) that causes the axis of the pulse output unit to travel to a position specified as a relative position.

■ Icon



■ Parameter

Scope	Parameter name	Type	Default	Description
Input	UnitID	UINT	-	Specifies the unit ID.
	AxisNo	UINT	-	Specifies the axis No.
	Execute	BOOL	FALSE	Starts execution at the rising edge.
	Distance	LREAL	0	Specifies the relative distance (u).
	Velocity	LREAL	0	Specifies the maximum velocity (u/s).

Scope	Parameter name	Type	Default	Description
	AccDec	LREAL	0	Specifies the acceleration / deceleration (u/s^2).
	ContinuousFlag	BOOL	FALSE	FALSE: E-point control TRUE: P-point control
Output	ContinuousEnable	BOOL	FALSE	TRUE: Distance, Velocity, and AccDec can be changed.
	Done	BOOL	FALSE	TRUE: FB operation is completed.
	Busy	BOOL	FALSE	TRUE: FB operation is in progress.
	CommandAborted	BOOL	FALSE	TRUE: FB operation is interrupted.
	Error	BOOL	FALSE	TRUE: FB is abnormally completed.
	ErrorID	PG_ERROR	NO_ERROR	An error ID is output.

■ Operations when the function block is executed

- Relative value control is performed according to the distance specified in the Distance input. (Coordinates: -2147483648 to 2147483647u)
- To use the E-point control, specify as follows.
  - Specifies ContinuousFlag = FALSE and execute.
- To use the P-point control, specify as follows.
  - Specifies ContinuousFlag = TRUE and execute.
  - When the ContinuousEnable output becomes TRUE after PG_MoveRelative is started, the Position, Velocity, and AccDec inputs can be rewritten. Change the inputs using the rising edge when the Execute input becomes TRUE as a trigger.
  - If rewriting is performed when the ContinuousEnable output is FALSE, the PG_FB_IN_BUSY error is issued.
  - If rewriting is not executed, the execution of PG_MoveRelative is completed (Done=TRUE) when the pulse output is completed.

- When executing the function block, use the PG_Power function block in advance to set to the servo ON state. If the function block is executed in the servo OFF state, the PG_AXIS_NOT_READY_FOR_MOTION error is issued.

12.2.5 PG_LatchPosition

This is a function block (FB) that causes the axis of the pulse output unit to travel to a position specified by an external signal input as a relative position.

■ Icon

■ Parameter

Scope	Parameter name	Type	Default	Description
Input	UnitID	UINT	-	Specifies the unit ID.
	AxisNo	UINT	-	Specifies the axis No.
	Execute	BOOL	FALSE	Starts execution at the rising edge.
	Distance	LREAL	0	Specifies the relative distance (u).
	Velocity	LREAL	0	Specifies the maximum velocity (u/s).
	AccDec	LREAL	0	Specifies the acceleration / deceleration (u/s^2).
	PositioningStart	BOOL	FALSE	Positioning start input (for debugging)
Output	ContinuousEnable	BOOL	FALSE	TRUE: Start input
	Done	BOOL	FALSE	TRUE: FB operation is completed.
	Busy	BOOL	FALSE	TRUE: FB operation is in progress.
	CommandAborted	BOOL	FALSE	TRUE: FB operation is interrupted.
	Error	BOOL	FALSE	TRUE: FB is abnormally completed.
	ErrorID	PG_ERROR	NO_ERROR	An error ID is output.

■ Operations when the function block is executed

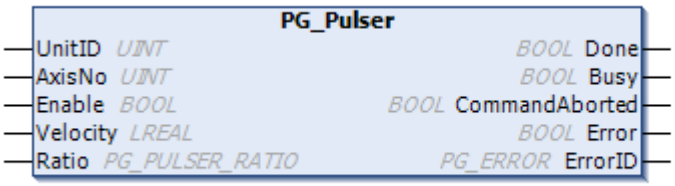
- Latch operation is started by using the timing when the Execute input becomes TRUE as a trigger. Makes a relative movement staring from the input of the external signal for a distance specified in the Distance input.
- PositioningStart works as an input for the position control start input of the external signal. It can be used when performing a debug.

- When executing the function block, use the PG_Power function block in advance to set to the servo ON state. If the function block is executed in the servo OFF state, the PG_AXIS_NOT_READY_FOR_MOTION error is issued.
- The input parameter cannot be changed while this function block is executed (Busy = TRUE). Using the timing when the Execute input becomes TRUE as a trigger, the PG_FB_IN_BUSY error is issued.

12.2.6 PG_Pulser

This is a function block (FB) that enables constant speed operation for the axes of the pulse output unit using external pulse input.

■ Icon



■ Parameter

Scope	Parameter name	Type	Default	Description
Input	UnitID	UINT	-	Specifies the unit ID.
	AxisNo	UINT	-	Specifies the axis No.
	Enable	BOOL	FALSE	TRUE: Enables pulser operation. FALSE: Disables pulser operation.
	Velocity	LREAL	0	Specifies the maximum velocity (u/s).
	Ratio	PG_PULSER_RATIO	RATIO_x1	Specifies the multiplication ratio between the pulser input and pulser output.
Output	Done	BOOL	FALSE	TRUE: FB operation is completed.
	Busy	BOOL	FALSE	TRUE: FB operation is in progress.
	CommandAborted	BOOL	FALSE	TRUE: FB operation is interrupted.
	Error	BOOL	FALSE	TRUE: FB is abnormally completed.
	ErrorID	PG_ERROR	NO_ERROR	An error ID is output.

■ Operations when the function block is executed

- When executing the function block, use the PG_Power function block in advance to set to the servo ON state. If the function block is executed in the servo OFF state, the PG_AXIS_NOT_READY_FOR_MOTION error is issued.

■ PG_PULSER_RATIO (Enumeration type)

For specifying the multiplication ratio between the pulser input and pulser output using PG_PULSER_RATIO, refer to the following table.

Definition	Description
RATIO_x1	x1
RATIO_x2	x2

Definition	Description
RATIO_x5	x5
RATIO_x10	x10
RATIO_x50	x50
RATIO_x100	x100
RATIO_x500	x500
RATIO_x1000	x1000

12.2.7 PG_Stop

This is a function block (FB) that causes the axis of the pulse output unit to make a forced stop or deceleration stop.

■ Icon

■ Parameter

Scope	Parameter name	Type	Default	Description
Input	UnitID	UINT	-	Specifies the unit ID.
	AxisNo	UINT	-	Specifies the axis No.
	Execute	BOOL	FALSE	Starts execution at the rising edge.
	StopMethod	BOOL	FALSE	TRUE: Forced stop FALSE: Deceleration stop
Output	Done	BOOL	FALSE	TRUE: FB operation is completed.
	Busy	BOOL	FALSE	TRUE: FB operation is in progress.
	CommandAborted	BOOL	FALSE	TRUE: FB operation is interrupted.
	Error	BOOL	FALSE	TRUE: FB is abnormally completed.
	ErrorID	PG_ERROR	NO_ERROR	An error ID is output.

■ Operations when the function block is executed

- If PG_Stop is executed while the following function block is being executed, execution of the function block that is being executed is interrupted (CommandAborted = TRUE).
  - PG_Jog
  - PG_MoveRelative

- PG_MoveAbsolute
- PG_LatchPosition
- PG_Pulser
- PG_Home

When executing one of these function blocks again, set the Execute input of PG_Stop to FALSE and then execute the function block.

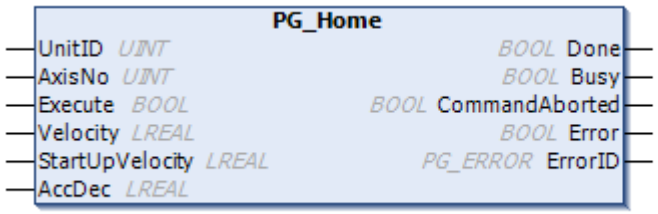
- The stop method can be specified using the StopMethod input.
  TRUE: Forced stop
  FALSE: Deceleration stop
- When the deceleration stop is selected, deceleration takes place at the acceleration / deceleration specified in function block.

- When executing the function block, use the PG_Power function block in advance to set to the servo ON state. If the function block is executed in the servo OFF state, the PG_AXIS_NOT_READY_FOR_MOTION error is issued.
- The input cannot be changed while this function block is executed (Busy = TRUE). Using the timing when the Execute input becomes TRUE as a trigger, the PG_FB_IN_BUSY error is issued.

12.2.8 PG_Home

This is a function block (FB) that performs home return of the pulse output unit.

■ Icon



■ Parameter

Scope	Parameter name	Type	Default	Description
Input	UnitID	UINT	-	Specifies the unit ID.
	AxisNo	UINT	-	Specifies the axis No.
	Execute	BOOL	FALSE	Starts execution at the rising edge.
	Velocity	LREAL	0	Specifies the maximum velocity (u/s).
	StartUpVelocity	LREAL	0	Specifies the startup velocity (u/s).
	AccDec	LREAL	0	Specifies the acceleration / deceleration (u/s^2).
Output	Done	BOOL	FALSE	TRUE: FB operation is completed.

Scope	Parameter name	Type	Default	Description
	Busy	BOOL	FALSE	TRUE: FB operation is in progress.
	CommandAborted	BOOL	FALSE	TRUE: FB operation is interrupted.
	Error	BOOL	FALSE	TRUE: FB is abnormally completed.
	ErrorID	PG_ERROR	NO_ERROR	An error ID is output.

■ Operations when the function block is executed

- When returning to the origin, the value of StartUpVelocity is used for the startup velocity, not the parameter setting.
- Use PG_ReadStatus to read the external signal for the home input or the near home input.
- The StartUpVelocity should be 1 u/s or higher.The value set in the StartUpVelocity is the creep speed setting.
- The maximum velocity should be 1 u/s or more.If set to 0, an error will occur at runtime.

- When executing the function block, use the PG_Power function block in advance to set to the servo ON state. If the function block is executed in the servo OFF state, the PG_AXIS_NOT_READY_FOR_MOTION error is issued.
- The input cannot be changed while this function block is executed (Busy = TRUE). Using the timing when the Execute input becomes TRUE as a trigger, the PG_FB_IN_BUSY error is issued.

12.2.9 PG_SetPosition

This is a function block (FB) that sets the elapsed value and the feedback counter of the pulse output unit to desired values.

■ Icon

■ Parameter

Scope	Parameter name	Type	Default	Description
Input	UnitID	UINT	-	Specifies the unit ID.
	AxisNo	UINT	-	Specifies the axis No.
	Execute	BOOL	FALSE	Starts execution at the rising edge.
	PositionType	PG_POSITION_TYPE	ABSOLUTE	Specify a setting target. ABSOLUTE: Elapsed value

Scope	Parameter name	Type	Default	Description
				FEEDBACK: Feedback counter BOTH: Elapsed value and feedback counter
	AbsoluteCounter	LREAL	0	Set value (Elapsed value)
	FeedbackCounter	LREAL	0	Set value (Feedback counter)
Output	Done	BOOL	FALSE	TRUE: FB operation is completed.
	Busy	BOOL	FALSE	TRUE: FB operation is in progress.
	CommandAborted	BOOL	FALSE	TRUE: FB operation is interrupted.
	Error	BOOL	FALSE	TRUE: FB is abnormally completed.
	ErrorID	PG_ERROR	NO_ERROR	An error ID is output.

■ Operations when the function block is executed

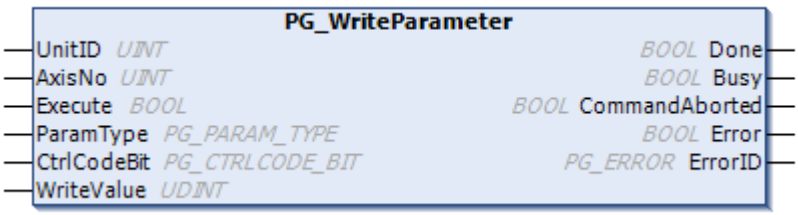
- When the PositionType input is set to BOTH, both the elapsed value and feedback counter can be set simultaneously.

- Always execute the function while all axes are stopped. If the function block is executed while axes are moving, the PG_AXIS_IN_DRIVEN error is issued.

12.2.10 PG_WriteParameter

This is a function block (FB) that writes the parameters to the pulse output unit.

■ Icon



■ Parameter

Scope	Parameter name	Type	Default	Description
Input	UnitID	UINT	-	Specifies the unit ID.
	AxisNo	UINT	-	Specifies the axis No.
	Execute	BOOL	FALSE	Starts execution at the rising edge.
	ParamType	PG_PARAM_TYPE	0	Specify target parameters using PG_PARAM_TYPE and CtrlCodeBit in combination.
	CtrlCodeBit	PG_CTRLCODE_BIT	0	

Scope	Parameter name	Type	Default	Description
	WriteValue	UDINT	0	Write value
Output	Done	BOOL	FALSE	TRUE: FB operation is completed.
	Busy	BOOL	FALSE	TRUE: FB operation is in progress.
	CommandAborted	BOOL	FALSE	TRUE: FB operation is interrupted.
	Error	BOOL	FALSE	TRUE: FB is abnormally completed.
	ErrorID	PG_ERROR	NO_ERROR	An error ID is output.

■ Operations when the function block is executed

- If the GM1 is started, the values set in the "Pulse_4Axes Parameters" of the GM Programmer are set.
- The parameters changed by PG_WriteParameter are not saved in the pulse output unit.
  When saving the parameter data in the pulse output unit, use the held data. After starting the GM1, write parameters using PG_WriteParameter.

■ How to Specify "Pulse_4Axes Parameters"

Specify "Pulse_4Axes Parameters" by combining the ParamType parameter and the CtrlCodeBit parameter as shown in the following table.

When ParamType = CONTROL_CODE

- Specify a target parameter by combining the PARAM_TYPE parameter and the CtrlCodeBit parameter.
- Specify a value to be written using the WriteValue parameter. Specify using PG_CTRLCODE_VALUE (enumeration type).

Pulse_4Axes Parameter	CtrlCodeBit	WriteValue	Description
DirectionOfRotationInput	PULSE_INPUT_ROTATION_DIRECTION	PULSEIN_DIRECTION_FORWARD	Forward
		PULSEIN_DIRECTION_REVERSE	Reverse
Count	PULSE_INPUT_COUNT	PULSEIN_COUNT_ENABLE	Enable
		PULSEIN_COUNT_DISABLE	Disable
PulseInputMode	PULSE_INPUT_MODE	PULSEIN_MODE_2PHASE	2-phase input
		PULSEIN_MODE_DIRECTION_DISTINCTION	Direction distinction input
		PULSEIN_MODE_INDIVIDUAL	Individual input
PulseInputCountMultiplication	PULSE_INPUT_MULTIPLICATION	PULSEIN_MULTIPLICATION_x1	x1
		PULSEIN_MULTIPLICATION_x2	x2
		PULSEIN_MULTIPLICATION_x4	x4
DirectionOfRotationOutput	PULSE_OUTPUT_ROTATION_DIRECTION	PULSEOUT_DIRECTION_FORWARD	Forward
		PULSEOUT_DIRECTION_REVERSE	Reverse
PulseOutputMode	PULSE_OUTPUT_MODE	PULSEOUT_MODE_PULSESIGN	Pulse/Sign

Pulse_4Axes Parameter	CtrlCodeBit	WriteValue	Description
		PULSEOUT_MODE_CWCCW	CW/CCW
PulseOutputFrequencyDivisionMode	PULSE_OUTPUT_DIVIDED_MODE	PULSEOUT_DIVIDED_BY1	Divided by 1
		PULSEOUT_DIVIDED_BY2	Divided by 2
		PULSEOUT_DIVIDED_BY4	Divided by 4
		PULSEOUT_DIVIDED_BY8	Divided by 8
		PULSEOUT_DIVIDED_BY16	Divided by 16
		PULSEOUT_DIVIDED_BY32	Divided by 32
		PULSEOUT_DIVIDED_BY64	Divided by 64
		PULSEOUT_DIVIDED_BY128	Divided by 128
DeviationCounterClearTime	DEVIATION_COUNTER_CLEAR_TIME	DEVIATION_COUNTER_CLEAR_TIME_1ms	1ms
		DEVIATION_COUNTER_CLEAR_TIME_10ms	10 ms
AccelerationDecelerationMethod	PULSE_OUT_ACC_DEC	PULSEOUT_ACC_DEC_LINEAR	Linear Acceleration / Deceleration
		PULSEOUT_ACC_DEC_SSHAPED	S Acceleration / Deceleration
OriginReturnDirection	HOMING_DIRECTION	HOME_DIRECTION_NEGATIVE	(-) Direction of the elapsed value
		HOME_DIRECTION_POSITIVE	(+) Direction of the elapsed value
StartUpTime	STARTUP_TIME	STARTUP_TIME_20us	0.02ms
		STARTUP_TIME_5us	0.005ms
		STARTUP_TIME_1us	0.001ms
OriginInputLogic	HOME_INPUT_LOGIC	HOME_INPUT_NORMAL_CLOSE	Normal Close
		HOME_INPUT_NORMAL_OPEN	Normal Open
OriginNeighborhoodLogic	NEARHOME_INPUT_LOGIC	NEARHOME_INPUT_NORMAL_OPEN	Normal Open
		NEARHOME_INPUT_NORMAL_CLOSE	Normal Close
OriginSearch	HOME_SEARCH	HOME_SEARCH_DISABLE	Disable
		HOME_SEARCH_ENABLE	Enable
LimitInputLogic	LIMIT_INPUT_LOGIC	LIMIT_INPUT_NORMAL_CLOSE	Normal Close
		LIMIT_INPUT_NORMAL_OPEN	Normal Open
SShapedPattern	S_ACC_DEC	S_ACC_DEC_SIN_CURVE	Sin curve
		S_ACC_DEC_SECONDRY_CURVE	Quadratic curve
		S_ACC_DEC_CYCLOID_CURVE	Cycloid curve
		S_ACC_DEC_THIRD_CURVE	Third curve

When other than ParamType = CONTROL_CODE

- Specify a target parameter by using only the PARAM_TYPE parameter.
- Specify a value to be written using the WriteValue parameter. Specify in the UDINT format.

Pulse_4Axes Parameter	ParamType	CtrlCodeBit	WriteValue
PulseInputAorB SignalInCnst	PULSE_INPUT_SIGNAL_INCNST	-	0: Not InCnst (No input time constant) 1: 0.1 us 2: 0.5 us 3: 1.0 us 4: 2.0 us 5: 10.0 us
OriginInputInCnst	HOME_INPUT_INCNST	-	0: Not InCnst (No input time constant) 1: 10 us 2: 100 us

12.2.11 PG_ReadParameter

This is a function block (FB) that reads the parameters of the pulse output unit.

■ Icon

■ Parameter

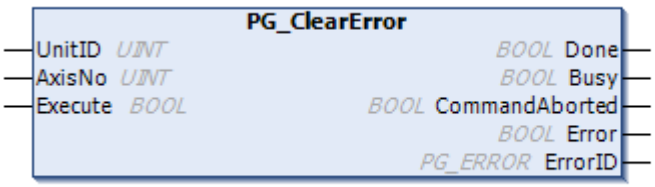
Scope	Parameter name	Type	Default	Description
Input	UnitID	UINT	-	Specifies the unit ID.
	AxisNo	UINT	-	Specifies the axis No.
	Execute	BOOL	FALSE	Starts execution at the rising edge.
	ParamType	PG_PARAM_TYPE	0	Specify target parameters using PG_PARAM_TYPE and CtrlCodeBit in combination.
	CtrlCodeBit	PG_CTRLCODE_BIT	0	
Output	ReadValue	UDINT	0	Read value
	Done	BOOL	FALSE	TRUE: FB operation is completed.
	Busy	BOOL	FALSE	TRUE: FB operation is in progress.

Scope	Parameter name	Type	Default	Description
	Error	BOOL	FALSE	TRUE: FB is abnormally completed.
	ErrorID	PG_ERROR	NO_ERROR	An error ID is output.

12.2.12 PG_ClearError

This is a function block (FB) that clears the limit error or the set value error of the pulse output unit.

■ Icon



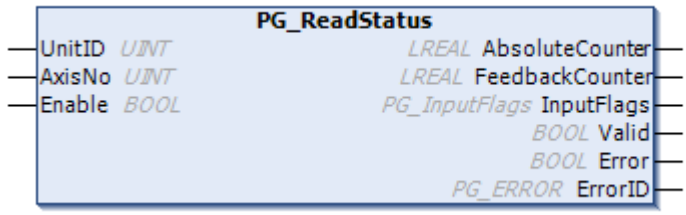
■ Parameter

Scope	Parameter name	Type	Default	Description
Input	UnitID	UINT	-	Specifies the unit ID.
	AxisNo	UINT	-	Specifies the axis No.
	Execute	BOOL	FALSE	Starts execution at the rising edge.
Output	Done	BOOL	FALSE	TRUE: FB operation is completed.
	Busy	BOOL	FALSE	TRUE: FB operation is in progress.
	CommandAborted	BOOL	FALSE	TRUE: FB operation is interrupted.
	Error	BOOL	FALSE	TRUE: FB is abnormally completed.
	ErrorID	PG_ERROR	0	An error ID is output.

12.2.13 PG_ReadStatus

This is a function block (FB) that reads the status of the pulse output unit.

■ Icon



■ Parameter

Scope	Parameter name	Type	Default	Description
Input	UnitID	UINT	-	Specifies the unit ID.
	AxisNo	UINT	-	Specifies the axis No.
	Enable	BOOL	FALSE	TRUE: Execution of the FB is enabled.
Output	AbsoluteCounter	LREAL	0	Read value (Elapsed value)
	FeedbackCounter	LREAL	0	Read value (Feedback counter)
	InputFlags	PG_InputFlags	0	Read value content (Input flag)
	Valid	BOOL	FALSE	TRUE:
	Error	BOOL	FALSE	TRUE: FB is abnormally completed.
	ErrorID	PG_ERROR	NO_ERROR	An error ID is output.

■ PG_InputFlags

For the contents of PG_InputFlags, refer to the following table.

Parameter	Name	Description
PulseOutputBusy	Pulse output busy	TRUE when the pulse output is being output
PulseOutputDone	Pulse output done	TRUE when the pulse output is completed
AccelerationZone	Acceleration zone	TRUE when in the acceleration zone
ConstantSpeedZone	Constant speed zone	TRUE when in the constant speed zone
DecelerationZone	Deceleration zone	TRUE when in the deceleration zone
RotationDirection	Rotation direction	Monitors the rotation direction. TRUE when the elapsed value is incrementing
HomeInput	Home input	Monitors the home Input signal. TRUE when the home input is enabled
NearHomeInput	Near home input	Monitors the near home input signal. TRUE when the near home input is enabled
HomingDone	Home return done	TRUE when the home return is completed
OutputStopError	Output stop error	TRUE when an error occurs in the pulse output unit and when output is stopped
SetValueChangeConfirmation	Set value change	Used to confirm rewriting of the set value when P-point control is performed.

Parameter	Name	Description
OverPositiveLimitInput	Over limit input (+)	Monitor contact of the over limit input (+) TRUE when over limit input (+) is enabled
OverNegativeLimitInput	Over limit input (-)	Monitor contact of the over limit input (-) TRUE when over limit input (-) is enabled
TimingInputMonitor	Timing input monitor	Monitor contact of the positioning control start input (timing input) TRUE when the positioning control start input is enabled
SetValueError	Set value error	TRUE when a set value error occurs
LimitError	Limit error	TRUE when a limit input is input during operation or when started
ServoOnOutputState	Servo ON output status	TRUE when servo ON

12.3 Error Codes

This section describes errors that are output in function blocks for pulse output unit and their contents. These errors are defined in PG_ERROR.

12.3.1 Error Check Method

With a function block that has the Error and ErrorID output parameters, it is possible to check whether an error has occurred.

When the Error output becomes TRUE, its error content is output to the ErrorID.

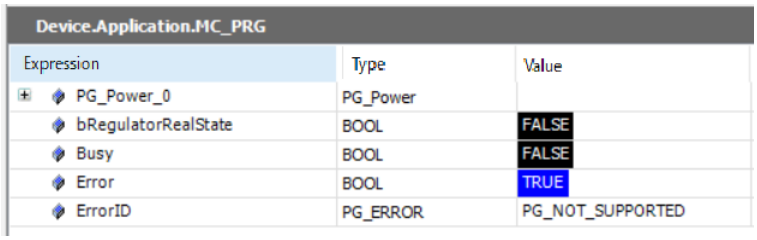
■ Error occurrence example

In the following example, PG_NOT_SUPPORTED has occurred in the PG_Power function block.

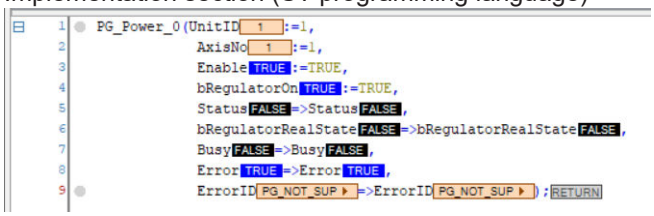
Error = TRUE (An error has occurred.)

ErrorID = PG_NOT_SUPPORTED (The function block not supported in the simulation mode was executed.)

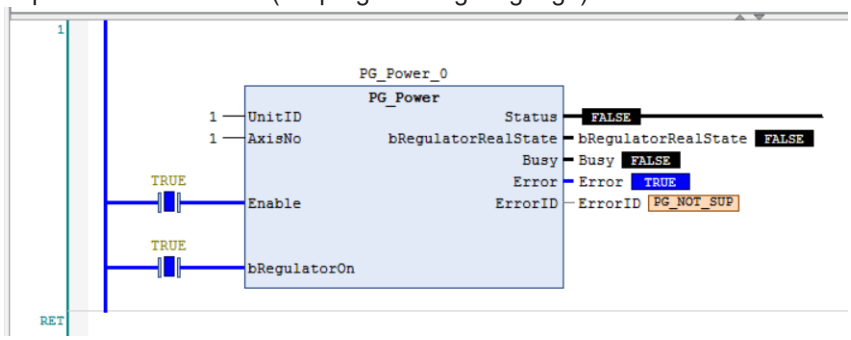
Declaration section



Implementation section (ST programming language)



Implementation section (LD programming language)



12.3.2 PG_ERROR

For the content of PG_ERROR output in each function block, refer to the following table.

Error name	Description
PG_NO_ERROR	Normal (no error)
PG_INVALID_UNIT	The specified unit ID or axis No. is invalid.
PG_SYSTEM_ERROR	This is an internal error in the GM1 Controller.
PG_NOT_SUPPORTED	The function block not supported in the simulation mode was executed.
PG_AXIS_NOT_READY_FOR_MOTION	The axis is in the servo OFF state.
PG_AXIS_UNIT_ERROR	The output of the pulse output unit has made an error stop.
PG_AXIS_SET_VALUE_ERROR	A set value error has occurred in the pulse output unit.
PG_AXIS_LIMIT_ERROR	A limit error has occurred in the pulse output unit.
PG_AXIS_IN_STOP	The function block could not be executed because stop processing was being executed by PG_Stop.
PG_AXIS_IN_RESET	The function block could not be executed because axis information was being changed by PG_ClearError and PG_SetPosition.
PG_AXIS_IN_DRIVEN	The function block could not be executed because the axis was moving.
PG_CHANGED_DURING_OPERATION	The unit ID or axis No. was changed during operation.
PG_INVALID_TARGET_VALUE	Abnormal Information (position, speed, acceleration, or deceleration) was Input.
PG_JOG_INVALID_REQUEST	Both JogForward and JogBackward of JOG were simultaneously set to TRUE.
PG_SET_VALUE_CHANGE_FAILED	Failed to change command information.
PG_FB_IN_BUSY	Using the timing when Execute is set to TRUE as the trigger is invalid for the function block being executed (Busy = TRUE).
PG_ERROR_CLEAR_FAILED	Failed to clear the error in the pulse output unit.
PG_PARAMETER_WRITE_FAILED	Failed to write the parameter.

13 Reference Information

13.1 Motion Errors (SMC_ERROR Type)

This section describes errors that are output in motion control instructions and their contents. Motion control errors are defined in SMC_ERROR.

13.1.1 Error Check Method

This section describes errors that are output in motion control instructions and their contents.

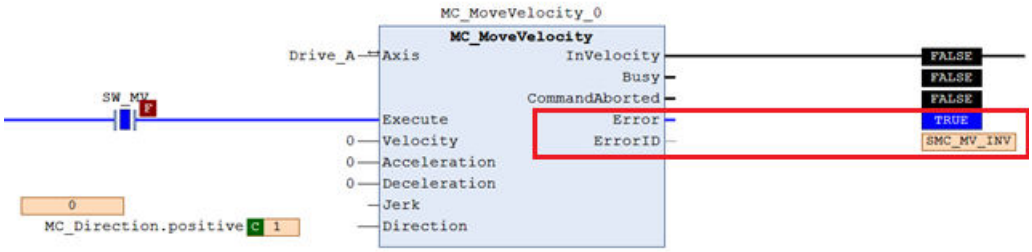
Motion control errors are defined in SMC_ERROR.

■ Error check method

With a function block that has an output Error and output ErrorID, it is possible to check that an error has occurred.

The following shows an example of an error that has occurred when the MC_MoveVelocity function block is executed.

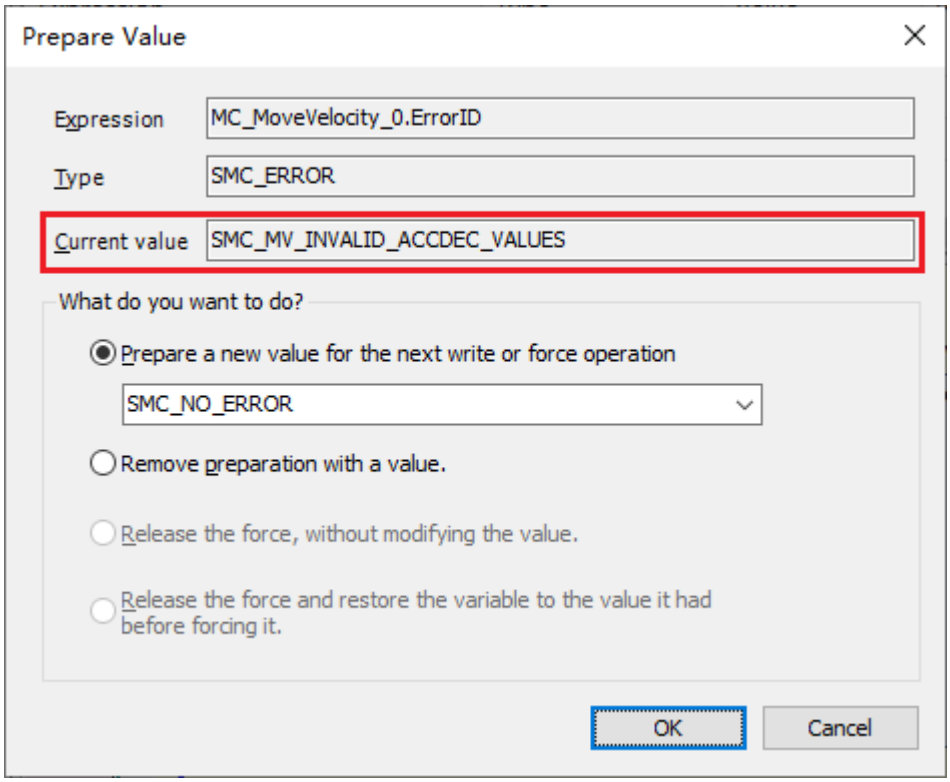
“TRUE” is output to the output Error and an error is output to the output ErrorID.



The error name can be checked by double-clicking the output ErrorID.

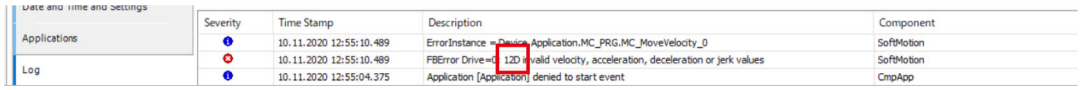
An error name defined in the enumeration type SMC_ERROR is displayed in the "Current value" field in the "Presetting values" dialog box.

Double-clicking SMC_MV_INV in the above execution example displays the following dialog box where error name "SMC_MV_INVALID_ACCDEC_VALUES" can be checked.



When an error occurs, the value of the error that has occurred (SMC_ERROR) is also recorded in hexadecimal number on the "Log" screen of the device editor.

The following example shows a record when an error ("SMC_MV_INVALID_ACCDEC_VALUES") with an error value of 12D (301 in decimal) has occurred.



13.1.2 SMC_ERROR

Error name	Value	Description
SMC_NO_ERROR	0	No error
SMC_DI_GENERAL_COMMUNICATION_ERROR	1	Communication error Communication disconnection or another communication problem occurred.
SMC_DI_AXIS_ERROR	2	Axis error Amplifier alarm or another axis problem occurred.
SMC_DI_FIELDBUS_LOST_SYNCRONICITY	3	The fieldbus lost synchronicity.
SMC_DI_SWLIMITS_EXCEEDED	10	The software limit has been exceeded.
SMC_DI_HWLIMITS_EXCEEDED	11	The hardware end switch is active.

Error name	Value	Description
SMC_DI_LINEAR_AXIS_OUTOFRANGE	12	An overflow occurred in the linear axis.
SMC_DI_HALT_OR_QUICKSTOP_NOT_SUPPORTED	13	The drive state is set to Halt or the Quickstop is unsupported.
SMC_DI_VOLTAGE_DISABLED	14	No power is supplied to the drive.
SMC_DI_IRREGULAR_ACTPOSITION	15	This error is not used.
SMC_DI_POSITIONLAGERROR	16	Position lag error The difference between the commanded position and actual position has exceeded the specified limit when position lag monitoring is active.
SMC_DI_HOMING_ERROR	17	Home return error
SMC_REGULATOR_OR_START_NOT_SET	20	Either the controller is disabled or the brake has been applied. Servo OFF or another similar problem occurred during axis movement.
SMC_WRONG_CONTROLLER_MODE	21	The executed function block is set to unsupported controller mode (SMC_SetControllerMode).
SMC_INVALID_ACTION_FOR_LOGICAL	25	Invalid operation was performed on the logical axis.
SMC_FB_WASNT_CALLED_DURING_MOTION	30	The function block was not called on the POU while the motion instruction was being executed ("Busy"). The operation was stopped while the motion instruction was being executed ("Busy").
SMC_AXIS_IS_NO_AXIS_REF	31	The type of the AXIS_REF type variable is different.
SMC_AXIS_REF_CHANGED_DURING_OPERATION	32	The AXIS_REF variable was changed during operation.
SMC_FB_ACTIVE_AXIS_DISABLED	33	The axis became disabled (MC_Power.bRegulatorOn) during movement.
SMC_AXIS_NOT_READY_FOR_MOTION	34	The axis cannot execute a motion instruction (an attempt was made to execute a motion instruction with MC_Stop enabled, for example).
SMC_AXIS_ERROR_DURING_MOTION	35	An error (such as amplifier alarm) occurred during motion operation.
SMC_VD_MAX_VELOCITY_EXCEEDED	40	The maximum velocity (fMaxVelocity) was exceeded.
SMC_VD_MAX_ACCELERATION_EXCEEDED	41	The maximum acceleration (fMaxAccleration) was exceeded.
SMC_VD_MAX_DECELERATION_EXCEEDED	42	The maximum deceleration (fMaxDeceleration) was exceeded.
SMC_3SH_INVALID_VELACC_VALUES	50	Either an invalid velocity or acceleration was specified.
SMC_3SH_MODE_NEEDS_HWLIMIT	51	For safety reasons, a request is made to invoke the mode in which the end switch is used.
SMC_FRC_NO_FREE_HANDLE	60	There is no file open handle.

Error name	Value	Description
SMC_MAC_INITIALIZATION_FAILED	65	SMC_MultiAcyclicCommunicator initialization failed.
SMC_MAC_INVALID_TASK_HANDLE	66	There is an invalid handle for the axis.
SMC_MAC_TOO_MANY_TASKS	67	There are too many tasks that use an axis generating SDO.
SMC_MAC_ATOMIC_ADD_FAILED	68	An attempt to add Atomic failed.
SMC_SDO_INVALID_DATALENGTH	69	An invalid data length (> 4) occurred due to SDO reading.
SMC_SCM_NOT_SUPPORTED	70	An invalid controller mode was set for SMC_SetControllerMode.
SMC_SCM_AXIS_IN_WRONG_STATE	71	The controller mode cannot be changed in the current axis state (an attempt was made to execute SMC_SetControllerMode with MC_Stop enabled, for example).
SMC_SCM_INTERRUPTED	72	SMC_SetControllerMode was interrupted by MC_Stop or ErrorStop.
SMC_ST_WRONG_CONTROLLER_MODE	75	The motion instruction was executed in an incorrect controller mode.
SMC_RAG_ERROR_DURING_STARTUP	80	An error occurred when the axis group was started up.
SMC_RAG_ERROR_AXIS_NOT_INITIALIZED	81	The axis is not in the specified state.
SMC_PP_WRONG_AXIS_TYPE	85	The function block does not support virtual axes or logical axes.
SMC_PP_NUMBER_OF_ABSOLUTE_BITS_INVALID	86	The number of bits is invalid (between 8 and 32 bits).
SMC_CGR_ZERO_VALUES	90	An invalid value was specified.
SMC_CGR_DRIVE_POWERED	91	A gear parameter was changed while the drive was in operation.
SMC_CGR_INVALID_POSPERIOD	92	An invalid position (0 or less, or half or more than the bus bandwidth) was specified.
SMC_CGR_POSPERIOD_NOT_INTEGRAL	93	The modulo period is not an integer.
SMC_P_FTASKCYCLE_EMPTY	110	There is no information in one cycle time. (fTaskCycle = 0)
SMC_R_NO_ERROR_TO_RESET	120	There is no error to be reset (MC_Reset was executed when there was no function block error, for example).
SMC_R_DRIVE_DOESNT_ANSWER	121	There is no response to an error reset.
SMC_R_ERROR_NOT_RESETTABLE	122	An error reset cannot be executed.
SMC_R_DRIVE_DOESNT_ANSWER_IN_TIME	123	Communication with the axis is not working.
SMC_R_CANNOT_RESET_COMMUNICATION_ERROR	124	A reset cannot be executed due to a communication error.
SMC_RP_PARAM_UNKNOWN	130	The parameter number is undefined.
SMC_RP_REQUESTING_ERROR	131	An error occurred in communication with the drive.
SMC_RP_DRIVE_PARAMETER_NOT_MAPPED	132	Parameters are not assigned to the drive.

Error name	Value	Description
SMC_RP_PARAM_CONVERSION_ERROR	133	Conversion of drive parameter values failed. Soft motion parameters are undefined.
SMC_WP_PARAM_INVALID	140	The parameter number is undefined or write operations are inhibited.
SMC_WP_SENDING_ERROR	141	Refer to the error number for WriteDriveParameter.
SMC_WP_DRIVE_PARAMETER_NOT_MAPPED	142	Parameters are undefined for the drive.
SMC_WP_PARAM_CONVERSION_ERROR	143	Conversion of drive parameter values failed. Soft motion parameters are undefined.
SMC_H_AXIS_WASNT_STANDSTILL	170	The axis is not in a standstill state.
SMC_H_AXIS_DIDNT_START_HOMING	171	An error occurred when home return was started.
SMC_H_AXIS_DIDNT_ANSWER	172	An error occurred when home return was started.
SMC_H_ERROR_WHEN_STOPPING	173	An error occurred after the axis stopped in home return mode. It is possible that deceleration was not set.
SMC_H_AXIS_IN_ERRORSTOP	174	The drive is in the ErrorStop state. Home return cannot be executed.
SMC_MS_UNKNOWN_STOPPING_ERROR	180	Undefined error
SMC_MS_INVALID_ACCDEC_VALUES	181	Either an invalid velocity or acceleration was specified.
SMC_MS_DIRECTION_NOT_APPLICABLE	182	"shortest" cannot be applied to the direction.
SMC_MS_AXIS_IN_ERRORSTOP	183	Because the drive is in the ErrorStop state, stop operation cannot be executed with MC_Stop.
SMC_BLOCKING_MC_STOP_WASNT_CALLED	184	MC_Stop (Execute=TRUE) blocks the axis. MC_Stop (Execute=FALSE) must be executed.
SMC_UNKNOWN_TASK_INTERVAL	200	The task interval of the bus task is undefined.
SMC_MA_INVALID_VELACC_VALUES	201	Either an invalid velocity or acceleration was specified.
SMC_MA_INVALID_DIRECTION	202	There is an error in the specified direction ("Direction").
SMC_MR_INVALID_VELACC_VALUES	226	Either an invalid velocity or acceleration was specified.
SMC_MR_INVALID_DIRECTION	227	There is an error in the specified direction ("Direction").
SMC_MAD_INVALID_VELACC_VALUES	251	Either an invalid velocity or acceleration was specified.
SMC_MAD_INVALID_DIRECTION	252	There is an error in the specified direction ("Direction").
SMC_MSI_INVALID_VELACC_VALUES	276	Either an invalid velocity or acceleration was specified.
SMC_MSI_INVALID_DIRECTION	277	There is an error in the specified direction ("Direction").
SMC_MSI_INVALID_EXECUTION_ORDER	278	The same instance of MC_MoveSuperImposed was called more than once in a single cycle.

Error name	Value	Description
SMC_LOGICAL_NO_REAL_AXIS	300	Unused error
SMC_MV_INVALID_ACCDEC_VALUES	301	Either an invalid velocity or acceleration was specified.
SMC_MV_DIRECTION_NOT_APPLICABLE	302	"shortest" or "fastest" cannot be applied to the direction ("Direction").
SMC_PP_ARRAYSIZE	325	There is an error in the specified array size.
SMC_PP_STEP0MS	326	The step time is 0s.
SMC_VP_ARRAYSIZE	350	There is an error in the specified array size.
SMC_VP_STEP0MS	351	The step time is 0s.
SMC_AP_ARRAYSIZE	375	There is an error in the specified array size.
SMC_AP_STEP0MS	376	The step time is 0s.
SMC_TP_TRIGGEROCCUPIED	400	The trigger is already enabled.
SMC_TP_COULDNT_SET_WINDOW	401	The drive does not support the window function.
SMC_TP_COMM_ERROR	402	Communication error
SMC_AT_TRIGGERNOTOCCUPIED	410	The trigger is already disabled.
SMC_MCR_INVALID_VELACC_VALUES	426	Either an invalid velocity or acceleration was specified.
SMC_MCR_INVALID_DIRECTION	427	An invalid direction was specified.
SMC_MCA_INVALID_VELACC_VALUES	451	Either an invalid velocity or acceleration was specified.
SMC_MCA_INVALID_DIRECTION	452	An invalid direction was specified.
SMC_MCA_DIRECTION_NOT_APPLICABLE	453	"fastest" cannot be applied to the direction ("Direction").
SMC_SDL_INVALID_AXIS_STATE	475	Function block "SMC_ChangeDynamicLimits" was executed in a state other than "Standstill" or "Power_off".
SMC_SDL_INVALID_VELACC_VALUES	476	An invalid velocity, acceleration, deceleration, or jerk was specified.
SMC_CR_NO_TAPPETS_IN_CAM	600	The cam is not equipped with a tappet.
SMC_CR_TOO_MANY_TAPPETS	601	The tappet group ID exceeds MAX_NUM_TAPPETS.
SMC_CR_MORE_THAN_32_ACCESSES	602	There are 32 or more accesses to one cam.
SMC_CI_NO_CAM_SELECTED	625	No cam is selected. It is possible that the correct cam table is not set in the CamTableID parameter of MC_CamIn.
SMC_CI_MASTER_OUT_OF_SCALE	626	The current commanded position on the master axis is outside the range of the cam table.
SMC_CI_RAMPIN_NEEDS_VELACC_VALUES	627	A velocity and acceleration must be specified when StartMode is set to ramp_in.
SMC_CI_SCALING_INCORRECT	628	The scaling variables (fEditor, TableMasterMin, and Max) are incorrect.
SMC_CI_TOO_MANY_TAPPETS_PER_CYCLE	629	The number of tappet outputs is too many to be enabled in one cycle.

Error name	Value	Description
SMC_CB_NOT_IMPLEMENTED	640	The function block for cam format is not implemented.
SMC_GI_RATIO_DENOM	675	RatioDenominator (denominator of the gear ratio) is set to 0.
SMC_GI_INVALID_ACC	676	The value specified in "Acceleration" is invalid.
SMC_GI_INVALID_DEC	677	The value specified in "Deceleration" is invalid.
SMC_GI_MASTER_REGULATOR_CHANGED	678	The Enable / Disable state of the master axis was changed without permission.
SMC_GI_INVALID_JERK	679	The value specified in "Jerk" is invalid.
SMC_PH_INVALID_VELACCDEC	725	The specified velocity, acceleration, or deceleration were invalid.
SMC_PH_ROTARYAXIS_PERIOD0	726	fPositionPeriod for the rotation axis is set to 0.
SMC_NO_CAM_REF_TYPE	750	The cam type is not an MC_CAM_REF structure.
SMC_CAM_TABLE_DOES_NOT_COVER_MASTER_SCALE	751	The master axis area (xStart and xEnd) of the cam table is outside the curve data range.
SMC_CAM_TABLE_EMPTY_MASTER_RANGE	752	The cam data table is empty.
SMC_CAM_TABLE_INVALID_MASTER_MINMAX	753	The maximum value and minimum value of the master axis in the cam data are invalid.
SMC_CAM_TABLE_INVALID_SLAVE_MINMAX	754	The maximum value and minimum value of the slave axis in the cam data are invalid.
SMC_GIP_MASTER_DIRECTION_CHANGE	775	The rotation direction of the master axis was changed while the salve axis was connected.
SMC_GIP_SLAVE_REVERSAL_CANNOT_BE_AVOIDED	776	The AvoidReversal input is set, but reverse rotation of the slave axis cannot be avoided.
SMC_GIP_AVOID_REVERSAL_FOR_FINITE_AXIS	777	The AvoidReversal input cannot be set for the finite slave axis.
SMC_BC_BL_TOO_BIG	800	The fBacklash gear backlash is too large (larger than position period/2).
SMC_QPROF_DIVERGES	825	Internal error: Failed in calculating the secondary path
SMC_QPROF_INVALID_PARAMETER	826	Internal error: Failed in calculating the secondary path
SMC_QPROF_NO_RESULT	827	Internal error: Failed in calculating the secondary path
SMC_QPROF_INVALID_NEW_LBD	828	Internal error: Failed in calculating the secondary path
SMC_QPROF_BAD_NEGOTIATION	829	Internal error: Failed in calculating the secondary path
SMC_QPROF_INVALID_INTERVAL	830	Internal error: Failed in calculating the secondary path
SMC_QPROF_NOT_ENOUGH_PHASES	831	Internal error: Failed in calculating the secondary path
SMC_TG_INTERNAL_ERROR	832	Internal error: Failed in calculating the secondary path

Error name	Value	Description
SMC_SRT_NOT_STANDSTILL_OR_POWEROFF	850	Execution is possible only in the standstill or power_off state.
SMC_SRT_INVALID_RAMPTYPE	851	The value specified in RampType is invalid.
SMC_SMT_NOT_STANDSTILL_OR_POWEROFF	852	Execution is possible only in the standstill or power_off state.
SMC_SMT_INVALID_MOVEMENTTYPE_OR_POSITIONPERIOD	853	The value specified in MovementType or PositionPeriod is invalid.
SMC_SMT_AXIS_NOT_VIRTUAL	854	The function block is valid only for the virtual axis.
SMC_NO_LICENSE	1000	License error
SMC_INT_VEL_ZERO	1001	Because Velocity is set to 0, path processing cannot be performed.
SMC_INT_NO_STOP_AT_END	1002	The final velocity of the path is other than 0.
SMC_INT_DATA_UNDERRUN	1003	GEOINFO-List was processed by DataIn, but the end of the list has not been reached.
SMC_INT_VEL_NONZERO_AT_STOP	1004	The velocity at the time of stoppage is greater than 0.
SMC_INT_TOO_MANY_RECURSIONS	1005	There are too many recursions of SMC_Interpolator.
SMC_INT_NO_CHECKVELOCITIES	1006	SMC_CheckVelocities is not called by SMC_OUTQUEUE.
SMC_INT_PATH_EXCEEDED	1007	Internal error or calculation error
SMC_INT_VEL_ACC_DEC_ZERO	1008	The specified velocity and acceleration / deceleration are 0 or less.
SMC_INT_DWIPOTIME_ZERO	1009	The motion task was called when dwIpoTime = 0.
SMC_INT_JERK_NONPOSITIVE	1010	A negative value was set for "Jerk".
SMC_INT_QPROF_DIVERGES	1011	Internal error The calculation algorithm is incorrect.
SMC_INT_INVLALID_VELOCITY_MODE	1012	The specified velocity mode is invalid.
SMC_INT_TOO_MANY_AXES_INTERPOLATED	1013	The maximum allowable number of axes for interpolation has been exceeded.
SMC_INT_DEGENERATE_SEGMENT	1014	
SMC_INT2DIR_BUFFER_TOO_SMALL	1050	
SMC_INT2DIR_PATH_FITS_NOT_IN_QUEUE	1051	
SMC_XINT_INVALID_DIRECTION	1070	
SMC_XINT_NOINTERSECTION	1071	
SMC_WAR_INT_OUTQUEUE_TOO_SMALL	1080	
SMC_WAR_END_VELOCITIES_INCORRECT	1081	The specified final velocity is incorrect.
SMC_CV_ACC_DEC_VEL_NONPOSITIVE	1100	Negative values are specified for the velocity and acceleration/deceleration.
SMC_CA_INVALID_ACCDEC_VALUES	1120	Negative values are specified for fGapVelocity, fGapAcceleration, and fGapDeceleration.
SMC_DEC_ACC_TOO_LITTLE	1200	The specified acceleration is unacceptable.

Error name	Value	Description
SMC_DEC_RET_TOO_LITTLE	1201	The specified deceleration is unacceptable.
SMC_DEC_OUTQUEUE_RAN_EMPTY	1202	Data underrun The queue was read, but it was empty.
SMC_DEC_JUMP_TO_UNKNOWN_LINE	1203	Because the line number is unknown, the cursor cannot jump to the line.
SMC_DEC_INVALID_SYNTAX	1204	The syntax is invalid.
SMC_DEC_3DMODE_OBJECT_NOT_SUPPORTED	1205	The object is not supported in 3D mode.
SMC_DEC_NEGATIVE_PERIOD	1206	A negative value is specified for the period during which an additional axis is disabled.
SMC_DEC_DIMENSIONS_EXCLUSIVE_AU	1207	Both axis A and axis U are not always interpolated. PA and PU are mutually exclusive.
SMC_DEC_DIMENSIONS_EXCLUSIVE_BV	1208	Both axis B and axis V are not always interpolated. PB and PV are mutually exclusive.
SMC_DEC_DIMENSIONS_EXCLUSIVE_CW	1209	Both axis C and axis W are not always interpolated. PC and PW are mutually exclusive.
SMC_IPR_TOO_SMALL__BUFFER	1259	The buffer size specified for OutOueue is too small.
SMC_GCV_BUFFER_TOO_SMALL	1300	
SMC_GCV_BUFFER_WRONG_TYPE	1301	
SMC_GCV_UNKNOWN_IPO_LINE	1302	
SMC_NO_CNC_REF_TYPE	1500	
SMC_NO_OUTQUEUE_TYPE	1501	The specified pointer is not SMC_OUTQUEUE.
SMC_GEOINFO_BUFFER_MISALIGNED	1502	The buffer segments aligned by four-byte boundaries are not used by pbyBuffer.
SMC_3D_MODE_NOT_SUPPORTED	1600	The FB functions only with 2D paths.

13.2 RTEX communication error

13.2.1 RTEX Error ID

■ WARNING_CODE (Union type)

Member	Type	Description
uiWarningCode	UINT	Warning code
tWarningCodeMember	ALARM_WARNING_CODES	Main code (warning number) and sub-code (0) of the warning code

■ List of RTEX Error IDs

Category	Error_Code / Sub_Error_Code	Cause
Command header related	0011h	● Mismatched node address (MAC-ID)
	0012h	● C/R bit is 1 despite of command. ● Sub_Chk is 0 in 32-byte mode.
Command code, control mode related	0021h	● Cyclic command is not defined.
	0022h	● Non-cyclic command is not defined (when cyclic command is normal). ● Combination error of control mode and non-cyclic command. ● Subcommand is undefined in 32-byte mode.
	002Eh	● Combination of communication cycle, semi-closed/full-closed, 16 / 32 byte mode, and control mode is not correct. ● Control mode has been changed in less than 2 ms. ● Control mode has been changed during profile position latch positioning / profile home return (Type_Code = 12h, 13h, 31h, 32h, 33h, 34h, 36h) operation. ● Control mode has been changed during execution of non-cyclic command (Busy = 1). ● Run the home return command (□4h) Type_Code = 1□h / 2□h during the velocity control (CV) / torque control (CT). ● Control mode has been changed to the velocity control during the 2 degrees of freedom control (synchronous) mode. ● Control mode has been changed to the torque control during the 2 degrees of freedom control (standard / synchronous) mode. ● Control mode has been changed during the retracting operation.
Argument related	0031h	● Type_Code / Sub_Type_Code is not defined.
	0032h	● Non-cyclic data / sub-command data other than Type_Code / Sub_Type_Code is out of setup range.
	0033h	● Cyclic data (Command_Data1) is out of setup range.
	0034h	● Feed forward data (Command_Data3, Sub_Command_Data2 / 3) is out of setup range.
Not executable 1 (general)	0041h	● Write access is attempted to read only media.
	0042h	● Alarm clear command is executed while an alarm that cannot be cleared has occurred and no warning was issued.

Category	Error_Code / Sub_Error_Code	Cause
	0043h	● External scale error clear command is executed when not in full-closed control mode or when no external scale error is detected.
	0045h	● In servo on state, reset command is executed in attribute C parameter validation mode.
	0046h	● After deceleration and stop according to the drive inhibit input (POT / NOT), direction command POT / NOT is applied. ● During deceleration according to the drive inhibit input (POT / NOT), a profile operation (except Type_Code = 31h, 32h, 33h, 34h, and 36h) is started.
Not executable 2 (related to home return)	0051h	● Multi-turn clearing of the home return command was executed while the encoder was in the incremental mode. ● Multi-turn clearing of the home return command was executed even when the single-turn absolute function was effective.
	0052h	● During cyclic position control (CP) (* including full-closed control) in absolute mode, Type_Code = 1□h of the home return command (□4h) has been executed. ● During profile position control (PP) (* including full-closed control) in absolute mode, profile home return has been executed.
	0053h	● During cyclic position control (CP) (* including full-closed control) in absolute mode, actual position set / command position set (Type_Code = 21h, 22h) of the home return command (□4h) have been executed.
	0055h	● Multi-turn clearing of the home return command is executed while in the full-closed control mode.
	0056h	● Multi-turn clearing of the home return command is executed while in the servo-on condition.
	0057h	● Type_Code = 1□h of the home return command is executed while in the servo-off state.
	0058h	● While the external input is not assigned to the latch correction terminal, Type_Code is executed by using the external input as a trigger. ● Started the latch mode with a stop function operated by the amplifier output signal as the trigger signal when Pr7.111 “Trigger signal assignment setting for the latch mode with a stop function” was set to 0 “Disabled”.
	0059h	● Executed the home return command (□4h) while the profile position latch positioning / profile home return (Type_Code = 12h, 13g, 31h, 32h, 33h, 34h, 36h) was operated. ● During profile positioning / profile continuous revolution (Type_Code = 10h, 11h, 20h), initialization mode (Type_Code = 1□h, 31h) of home return command (□4h) has been executed.
	005Ah	● Z phase is set to latch trigger signal despite absolute external scale.
	005Bh	● Received the following commands in the virtual full-closed control mode. • Home return command (□4h) • Profile position latch absolute positioning (12h) of the profile command (17h) • Profile position latch absolute positioning (13h) of the profile command (17h) • Profile home return (31h to 34h, 36h) of the profile command (17h) • Config command

Category	Error_Code / Sub_Error_Code	Cause
		● Received a command to change to the virtual full-closed control mode under the following conditions. • While initialization mode of home return command (□4h) was operated, latch mode was operated, or latch mode with stop function was operated • Changed to a command other than command code (□4h) after starting home return command (Type_Code: 51h to 53h) • During a period from starting the latch to detecting the latch after starting home return command (Type_Code: 51h to 53h) • While profile position latch absolute positioning (12h) of the profile command (17h) was operated • While profile position latch absolute positioning (13h) of the profile command (17h) was operated • While profile home return (31h to 34h, 36h) of the profile command (17h) was operated • After starting profile command (12h, 13h, 31h to 34h, 36h), during the period from when a change was made to a command other than command code (17h) until the latch or home was detected • While Config command was executed
	005Fh	● Latch mode with stop function (Type_Code = F1h) was used in a setting other than the cyclic position control (CP). ● Latch mode with stop function (Type_Code = F1h) was used in a setting other than the communication cycle of 0.5 ms/command update cycle of 1.0 ms ● Latch mode with stop function (Type_Code = F1h) was used in a setting other than the electronic gear ratio of less than 1.
Not executable 3 (Related to hardware factor)	0061h	● EEPROM writing is not permitted because of under voltage of the control power.
Not executable 4 (in process)	0101h	● Not permitted to be accepted because the previous command is in process.
	0102h	● Command is not permitted to be accepted because the servo driver is accessing to the encoder now.
	0103h	● Command is not permitted to be accepted because the servo driver is accessing to the external scale now.
	0104h	● Type_Code has been changed while operating under profile position control (PP).
	0105h	● During execution of the PANATERM command (test run operation, FFT, Z phase search, pin assignment setting, or fit gain), received the RTEX command (reset command, home return command, or parameter command).
Not executable 5 (access inhibited)	0201h	● Command is not permitted to be accepted because parameter writing or writing to EEPROM is inhibited now. ● Write parameter command or write EEPROM command is issued while bit 0 of Pr7.23 RTEX function expansion setup 2 is set at 1.

13.2.2 Alarm Codes

■ ALARM_CODE (Union type)

Member	Type	Description
uiAlarmCode	UINT	Alarm code
tAlarmCodeMember	ALARM_WARNING_CODES	Main code and sub-code of the alarm code

■ List of alarm codes

Error No.		Alarm name	Attribute		
Main	Sub		History	Can be cleared	Emergency stop(Note 6)
11	0	Control power supply undervoltage protection		○	
12	0	Over-voltage protection	○	○	
13	0	Main power supply undervoltage protection (Insufficient voltage between P and N)		○	○
	1	Main power supply undervoltage protection (AC interception detection)		○	○
14	0	Over-current protection	○		
	1	IPM error protection	○		
15	0	Overheat protection	○		○
	1	Encoder overheat error protection	○		○
16	0	Overload protection	○	○(Note 1)	
	1	Torque saturation error protection	○	○	
18	0	Regenerative overload protection	○		○
	1	Regenerative transistor error protection	○		
21	0	Encoder communication line breakage fault protection	○		
	1	Encoder communication error protection	○		
23	0	Encoder communication data error protection	○		
24	0	Position deviation excess protection	○	○	○
	1	Speed deviation excess protection	○	○	○
25	0	Hybrid deviation excess protection	○		○
26	0	Overspeed protection	○	○	○
	1	2nd overspeed protection	○	○	
27	1	Absolute clearing protection	○		
	4	Command error protection	○		○
	5	Command generation error protection	○		○
	6	Operation command contention protection	○	○	
	7	Position information initialization error protection	○		

Error No.		Alarm name	Attribute		
Main	Sub		History	Can be cleared	Emergency stop(Note 6)
28	0	Pulse regeneration limit protection	○	○	○
29	1	Counter overflow protection 1	○		
	2	Counter overflow protection 2	○		
31	0	Safety function error protection 1	○		
	2	Safety function error protection 2	○		
33	0	Input duplicated allocation error-1 protection	○		
	1	Input duplicated allocation error-2 protection	○		
	2	Input function number error-1 protection	○		
	3	Input function number error-2 protection	○		
	4	Output function number error-1 protection	○		
	5	Output function number error-2 protection	○		
	8	Latch input allocation error protection	○		
34	0	Motor operable range setting error protection	○	○	
	1	One revolution absolute operable range error protection	○	○	
36	0 to 1	EEPROM parameter error protection			
37	0 to 2	EEPROM check code error protection			
38	0	Over-travel inhibit input protection 1		○	
	1	Over-travel inhibit input protection 2		○	
	2	Over-travel inhibit input protection 3	○		
40	0	Absolute system failure protection	○	○(Note 2)	
41	0	Absolute counter limit excess protection	○		
42	0	Absolute overspeed protection	○	○(Note 2)	
44	0	Single-turn counter error protection	○		
45	0	Multi-turn counter error protection	○		
47	0	Absolute status error protection	○		
50	0	External scale wiring error protection	○		
	1	External scale communication error protection	○		
	2	External scale communication data error protection	○		
51	0	External scale ST error protection 0	○		
	1	External scale ST error protection 1	○		
	2	External scale ST error protection 2	○		
	3	External scale ST error protection 3	○		
	4	External scale ST error protection 4	○		
	5	External scale ST error protection 5	○		
55	0	Phase-A wiring error protection	○		

Error No.		Alarm name	Attribute		
Main	Sub		History	Can be cleared	Emergency stop(Note 6)
	1	Phase-B wiring error protection	○		
	2	Phase-Z wiring error protection	○		
70	0	Phase U current detector error protection	○		
	1	Phase W current detector error protection	○		
72	0	Thermal relay error protection	○		
80	3	PLL incomplete error protection	○	○	
82	0	RTEX node addressing error protection	○		
83	0	RTEX continuous communication error protection 1	○	○	○
	1	RTEX continuous communication error protection 2	○	○	○
84	0	RTEX Communication timeout error protection	○	○	○
	3	RTEX communication synchronization error protection	○		
	5	RTEX communication cycle error protection	○	○	○
85	0	Retracting operation completion (I/O)(Note 7)	○	(Note 8)	○
	2	Retracting operation error(Note 7)	○	(Note 8)	○
86	0	RTEX cyclic data error protection 1	○	○	○
	1	RTEX cyclic data error protection 2	○	○	○
	2	RTEX update counter error protection	○		○
87	0	Forced alarm input protection		○	○
	1	Retracting operation completion (I/O)(Note 7)	○	(Note 8)	○
	3	Retracting operation error(Note 7)	○	(Note 8)	○
90	2	RTEX multi-axis synchronization establishment error protection	○		
91	1	RTEX command error protection	○	○	
	3	RTEX command error protection 2	○	○	
92	0	Encoder data restoration error protection	○		
	1	External scale data restoration error protection	○		
	3	Multi-turn data upper-limit value mismatch error protection	○		
93	0	Parameter setting error protection 1	○		
	2	Parameter setting error protection 2	○		
	3	External scale connection error protection	○		
	5	Parameter setting error protection 4	○		
	8	Parameter setting error protection 6	○		
94	2	Home return error protection	○	○	
	3	Home return error protection 2	○	○	

Error No.		Alarm name	Attribute		
Main	Sub		History	Can be cleared	Emergency stop(Note 6)
95	0 to 4	Motor automatic recognition error protection			
96	2	Control unit error protection 1	○		
	3	Control unit error protection 2	○		
	4	Control unit error protection 3	○		
	5	Control unit error protection 4	○		
	6	Control unit error protection 5	○		
	7	Control unit error protection 6	○		
98	1	RTEX hardware error protection 1	○		
	2	RTEX hardware error protection 2	○		
	3	RTEX hardware error protection 3	○		
Other numbers		Other error protections	-	-	-

(Note 1) When Err 16.0 “Over-load protection” occurs, it can be cleared approx. 10 seconds after it occurs. The alarm clear command is received as is and clearing process takes place after it is ready to be cleared.

(Note 2) When Err 40.0 “Absolute system failure protection” or Err 42.0 ”Absolute overspeed protection” occurs, the error cannot be cleared until absolute clear is performed.

(Note 3) When an alarm that cannot be cleared occurs, cycle the control power supply after removing the cause of the error or use RTEX software reset command to clear the alarm.

(Note 4) When an alarm that can be cleared occurs, use RTEX communication or USB communication (setup support software) to clear the alarm. Always clear the alarm while all axes are stopped and after securing safety.

(Note 5) If the internal control circuit of the servo amplifier malfunctions due to excessive noise etc., the display will be as shown below.

In such a case, immediately turn OFF the power.

(Note 6) Emergency stop refers to an alarm that is triggered if Pr 5.10 “Sequence at alarm” is set to 4 to 7 and that causes an immediate stop. For details, refer to the instruction manual and other technical references for the servo amplifier.

(Note 7) The alarm generated during retracting operation is switched by Pr 6.86 “Retreat operation alarm setup” bit 15.

Example: When bit 15 = 0, Err 85.0 and Err 85.2 will occur (A5N compatible specification).

When bit 15 = 1, Err 87.1 and Err 87.3 will occur (A6B compatible specification).

(Note 8) Whether alarm can be cleared or not is determined by the setting (bit 0 or 2) of Pr 6.86.

Bit 0: Err 85.0 / Err 87.1 (Retracting operation completion (I/O)) alarm clear attribute

Bit 2: Err 85.2 / Err 87.3 (Retracting operation error) alarm clear attribute; For either case, 0: Alarm clear invalid, 1: Alarm clear valid

13.2.3 Warning Codes

■ ALARM_WARNING_CODES (Structure)

Member	Type	Description
byMainCode	BYTE	Main code
bySubCode	BYTE	Sub-code

■ General warnings

Warning No. (hexadecimal)	Warning name	Description	Warning latch	Output setting	Warning mask
			Pr6.27 (Note 1)	Pr4.40 / Pr4.41 (Note 2)	Pr6.38 / Pr6.39 Corresponding bit (Note 3)
A0	Overload warning Warning	Load factor is 85% or more of the protection level.	○	1	Pr6.38 bit7
A1	Over-regeneration warning	Regenerative load factor has exceeded 85% of the protection level.	○	2	Pr6.38 bit5
A2	Battery warning (Note 4)	Battery voltage is 3.2 V or less.	Latch fixed	3	Pr6.38 bit0
A3	Fan warning	Fan has stopped for 1 second.	○	4	Pr6.38 bit6
A4	Encoder communication warning	The number of successive encoder communication errors has exceeded the specified value.	○	5	Pr6.38 bit4
A5	Encoder overheat warning	The encoder temperature exceeds the specified value.	○	6	Pr6.38 bit3
A6	Oscillation detection warning	Oscillation state was detected.	○	7	Pr6.38 bit13
A7	Lifetime detection warning	The remaining life expectancy of a capacitor or a fan dropped below the specified value.	Latch fixed	8	Pr6.38 bit2
A8	External scale error warning	The external scale detected a warning.	○	9	Pr6.38 bit8
A9	External scale communication warning	The number of successive external scale communication errors has exceeded the specified value.	○	10	Pr6.38 bit14
AC	Deterioration diagnosis warning (Note 6)	Load characteristic estimated value or torque command value at a constant velocity has exceeded the set range.	○	22	Pr6.39 bit7

■ Extended warning

Warning No. (hexadecimal)	Warning name	Description	Warning latch	Output setting	Warning mask
			Pr6.27 (Note 1)	Pr4.40 / Pr4.41 (Note 2)	Pr6.38 / Pr6.39 Corresponding bit (Note 3)
C0	RTEX continuous communication error warning	The number of successive errors (CRC error) detected when reading the received data sent to the local node The number of successive errors (CRC error) has exceeded the value set by Pr 7.26 “RTEX continuous communication error warning setup”.	○	11	Pr6.38 bit9
C1	RTEX accumulated communication error warning	The number of successive errors (CRC error) detected when reading the received data sent to the local node has exceeded the value set by Pr 7.27 “RTEX accumulated communication error warning setup”.	Latch fixed	12	Pr6.38 bit10
C2	RTEX_ Update_Counter error warning	The Update_Counter was not updated properly because the data accumulated exceeded the count value set by Pr 7.28 “RTEX_Update_Counter error warning setup”.	Latch fixed	13	Pr6.38 bit11
C3	Main power OFF warning	When Pr 7.14 “Main power OFF warning detection time” was set to 10 to 1999, instantaneous power failure that occurred between L1 and L3 exceeded the time set by Pr 7.14.	○	14	Pr6.38 bit12
D2	PANATERM command execution warning	While bit 0 of Pr 7.99 “RTEX function enhancement setting 6” was set to 1 and RTEX communication was established, an operation command (test run, FFT, etc.) was executed by the setup support software “PANATERM”.	○	30	Pr6.39 bit8

(Note 1) The symbol “○” marked in the “Warning latch” column indicates that it is possible to switch the mode between non-latch mode (latch for 1 second) and latch mode by using Pr 6.27 ”Warning latch state setup”. Only latch mode is available for the battery warning and the lifetime detection warning.

(Note 2) Select the warning output signal 1 (WARN 1) or warning output signal 2 (WARN 2) through Pr 4.40 “Warning output select 1” or Pr 4.41 “Warning output select 2”. When the set value is 0, all warnings are Ored before being output. Do not use any settings other than the settings shown in the above table.

(Note 3) Each warning detection can be disabled by Pr 6.38 “Warning mask setting” or Pr 6.39 “Warning mask setting 2”. The corresponding bits are shown in the table. Set the bit to 1 to disable the warning detection. For extended warning, warning detection can be disabled by parameter settings.

Also note that bit arrangements of these masks are different from those of general-purpose type MINAS-A6 series.

(Note 4) When the single-turn absolute function is enabled, a battery alarm is not detected.

(Note 5) Warning can be cleared by alarm clear. If warning cause is not resolved yet, the warning is cleared once, but a warning is issued again.

(Note 6) If bit 1 of Pr 6.97 “Function enhancement setup 3” is set to 0, it is disabled.

13.3 List of AMP Parameters

13.3.1 Class 0: Basic Setting

Class	No.	Parameter name	Unit	Setting range
0	00	Rotational direction setup	-	0 to 1
	01	Control mode setup	-	0 to 6
	02	Real-time auto-gain tuning setup	-	0 to 6
	03	Selection of machine stiffness at real-time auto-gain tuning	-	0 to 31
	04	Inertia ratio	%	0 to 10000
	08	Command pulse counts per one motor revolution	pulse	0 to 2^{23}
	09	Numerator of electronic gear	-	0 to 2^{30}
	10	Denominator of electronic gear	-	1 to 2^{30}
	11	Number of output pulses per motor rotation	pulse/r	1 to 2097152
	12	Reversal of pulse output logic / output source selection	-	0 to 3
	13	1st torque limit	%	0 to 500
	14	Position deviation excess setup	Command unit	0 to 2^{30}
	15	Absolute encoder setup	-	0 to 4
	16	External regenerative resistor setup	-	0 to 3
	17	Load factor of external regenerative resistor selection	-	0 to 4

13.3.2 Class 1: Gain Adjustment

Class	No.	Parameter name	Unit	Setting range
1	00	1st gain of position loop	0.1/s	0 to 30000
	01	1st gain of velocity loop	0.1 Hz	1 to 32767
	02	1st time constant of velocity loop integration	0.1 ms	1 to 10000
	03	1st filter of speed detection	-	0 to 5
	04	1st time constant of torque filter	0.01 ms	0 to 2500
	05	2nd gain of position loop	0.1/s	0 to 30000
	06	2nd gain of velocity loop	0.1 Hz	1 to 32767
	07	2nd time constant of velocity loop integration	0.1 ms	1 to 10000
	08	2nd filter of speed detection	-	0 to 5

Class	No.	Parameter name	Unit	Setting range
	09	2nd time constant of torque filter	0.01 ms	0 to 2500
	10	Velocity feed forward gain	0.1%	0 to 4000
	11	Velocity feed forward gain	0.1%	0 to 4000
	12	Velocity feed forward gain	0.01 ms	0 to 6400
	13	Torque feed forward filter	0.01 ms	0 to 6400
	14	2nd gain setup	-	0 to 1
	15	Mode of position control switching	-	0 to 10
	16	Delay time of position control switching	0.1 ms	0 to 10000
	17	Level of position control switching	-	0 to 20000
	18	Hysteresis at position control switching	-	0 to 20000
	19	Position gain switching time	0.1 ms	0 to 10000
	20	Mode of velocity control switching	-	0 to 5
	21	Delay time of velocity control switching	0.1 ms	0 to 10000
	22	Level of velocity control switching	-	0 to 20000
	23	Hysteresis at velocity control switching	-	0 to 20000
	24	Mode of torque control switching	-	0 to 3
	25	Delay time of torque control switching	0.1 ms	0 to 10000
	26	Level of torque control switching	-	0 to 20000
	27	Hysteresis at torque control switching	-	0 to 20000

13.3.3 Class 2: Vibration Suppression Function

Class	No.	Parameter name	Unit	Setting range
2	00	Adaptive filter mode setup	-	0 to 6
	01	1st notch frequency	Hz	50 to 5000
	02	1st notch width selection	-	0 to 20
	03	1st notch depth selection	-	0 to 99
	04	2nd notch frequency	Hz	50 to 5000
	05	2nd notch width selection	-	0 to 20
	06	2nd notch depth selection	-	0 to 99
	07	3rd notch frequency	Hz	50 to 5000
	08	3rd notch width selection	-	0 to 20
	09	3rd notch depth selection	-	0 to 99

Class	No.	Parameter name	Unit	Setting range
	10	4th notch frequency	Hz	50 to 5000
	11	4th notch width selection	-	0 to 20
	12	4th notch depth selection	-	0 to 99
	13	Selection of damping filter switching	-	0 to 6
	14	1st damping frequency	0.1 Hz	0 to 3000
	15	1st damping filter setup	0.1 Hz	0 to 1500
	16	2nd damping frequency	0.1 Hz	0 to 3000
	17	2nd damping filter setup	0.1 Hz	0 to 1500
	18	3rd damping frequency	0.1 Hz	0 to 3000
	19	3rd damping filter setup	0.1 Hz	0 to 1500
	20	4th damping frequency	0.1 Hz	0 to 3000
	21	4th damping filter setup	0.1 Hz	0 to 1500
	22	Command smoothing filter	0.1 ms	0 to 10000
	23	Command FIR filter	0.1 ms	0 to 10000
	24	5th notch frequency	Hz	50 to 5000
	25	5th notch width selection	-	0 to 20
	26	5th notch depth selection	-	0 to 99
	27	1st vibration control width setting	-	0 to 1000
	28	2nd vibration control width setting	-	0 to 1000
	29	3rd vibration control width setting	-	0 to 1000
	30	4th vibration control width setting	-	0 to 1000

13.3.4 Class 3: Speed, Torque Control, Full-closed Control

Class	No.	Parameter name	Unit	Setting range
3	12	Acceleration time setting	ms/(1000 r/min)	0 to 10000
	13	Deceleration time setting	ms/(1000 r/min)	0 to 10000
	14	Sigmoid acceleration/ deceleration time setup	ms	0 to 10000
	17	Speed limit selection	-	0 to 1
	21	Speed limit value 1	r/min	0 to 20000
	22	Speed limit value 2	r/min	0 to 20000
	23	External scale selection	-	0 to 6
	24	External scale numerator of division	-	0 to 2^{23}
	25	External scale denominator of division	-	1 to 2^{23}

Class	No.	Parameter name	Unit	Setting range
	26	External scale reversal of direction	-	0 to 3
	27	External scale Z phase disconnection detection disable	-	0 to 1
	28	Hybrid deviation excess protection	Command unit	1 to 2^{27}
	29	Hybrid deviation clear setting	Rotation	0 to 100
	32	External scale movement judgment threshold at virtual full-closed control mode	External scale unit	0 to 65534

13.3.5 Class 4: I/O Monitor Setting

Class	No.	Parameter name	Unit	Setting range
4	00	SI1 input selection	-	0 to 00FFFFFFh
	01	SI2 input selection	-	0 to 00FFFFFFh
	02	SI3 input selection	-	0 to 00FFFFFFh
	03	SI4 input selection	-	0 to 00FFFFFFh
	04	SI5 input selection	-	0 to 00FFFFFFh
	05	SI6 input selection	-	0 to 00FFFFFFh
	06	SI7 input selection	-	0 to 00FFFFFFh
	07	SI8 input selection	-	0 to 00FFFFFFh
	10	SO1 output selection	-	0 to 00FFFFFFh
	11	SO2 output selection	-	0 to 00FFFFFFh
	12	SO3 output selection	-	0 to 00FFFFFFh
	16	Type of analog monitor 1	-	0 to 28
	17	Analog monitor 1 output gain	-	0 to 214748364
	18	Type of analog monitor 2	-	0 to 28
	19	Analog monitor 2 output gain	-	0 to 214748364
	21	Analog monitor output setup	-	0 to 2
	31	Positioning complete range	Command unit	0 to 2097152
	32	Positioning complete (In-position) output setup	-	0 to 10
	33	INP hold time	ms	0 to 30000
	34	Zero-speed	r/min	10 to 20000
	35	Speed coincidence range	r/min	10 to 20000
	36	At-speed (Speed arrival)	r/min	10 to 20000

Class	No.	Parameter name	Unit	Setting range
	37	Mechanical brake action at stalling setup	ms	0 to 10000
	38	Mechanical brake action at running setup	ms	0 to 32000
	39	Brake release speed setup	r/min	30 to 3000
	40	Selection of alarm output 1	-	0 to 40
	41	Selection of alarm output 2	-	0 to 40
	42	2nd Positioning complete (In-position) range	Command unit	0 to 2097152
	44	Position compare output pulse width setting	0.1 ms	0 to 32767
	45	Position compare output polarity select	-	0 to 7
	47	Pulse output select	-	0 to 1
	48	Position compare value 1	Command unit	-2147483648 to 2147483647
	49	Position compare value 2	Command unit	-2147483648 to 2147483647
	50	Position compare value 3	Command unit	-2147483648 to 2147483647
	51	Position compare value 4	Command unit	-2147483648 to 2147483647
	52	Position compare value 5	Command unit	-2147483648 to 2147483647
	53	Position compare value 6	Command unit	-2147483648 to 2147483647
	54	Position compare value 7	Command unit	-2147483648 to 2147483647
	55	Position compare value 8	Command unit	-2147483648 to 2147483647
	56	Position compare output delay compensation amount	0.1 us	-32768 to 32767
	57	Position compare output assignment setting	-	-2147483648 to 2147483647

13.3.6 Class 5: Enhancing Setting

Class	No.	Parameter name	Unit	Setting range
5	03	Denominator of pulse output division	-	0 to 8388608
	04	Over-travel inhibit input setup	-	0 to 2
	05	Sequence at over-travel inhibit	-	0 to 2

Class	No.	Parameter name	Unit	Setting range
	06	Sequence at Servo-Off	-	0 to 9
	07	Sequence at main power OFF	-	0 to 9
	08	LV trip selection at main power OFF	-	0 to 3
	09	Detection time of main power off	ms	20 to 2000"13.3.9 Class 8: Special Setting 3"
	10	Sequence at alarm	-	0 to 7
	11	Torque setup for emergency stop	%	0 to 500
	12	Over-load level setup	%	0 to 500
	13	Over-speed level setup	r/min	0 to 20000
	14	Motor working range setup	0.1 revolution	0 to 1000
	15	Control input signal read setting	-	0 to 3
	20	Position setup unit select	-	0 to 1
	21	Selection of torque limit	-	0 to 4
	22	2nd torque limit	%	0 to 500
	23	Torque limit switching setup 1	ms/100 %	0 to 4000
	24	Torque limit switching setup 2	ms/100 %	0 to 4000
	25	Positive direction torque limit	%	0 to 500
	26	Negative direction torque limit	%	0 to 500
	31	USB axis address	-	0 to 127
	33	Pulse regenerative output limit setup	-	0 to 1
	45	Quadrant projection positive direction compensation value	0.1%	-1000 to 1000
	46	Quadrant projection negative direction compensation value	0.1%	-1000 to 1000
	47	Quadrant projection compensation delay time	ms	0 to 1000
	48	Quadrant projection compensation filter setting L	0.01 ms	0 to 6400
	49	Quadrant projection compensation filter setting H	0.1 ms	0 to 10000
	56	Slow stop deceleration time setting	ms/(1000 r/min)	0 to 10000

Class	No.	Parameter name	Unit	Setting range
	57	Slow stop S-shape acceleration and deceleration setting	ms	0 to 1000
	66	Deterioration diagnosis convergence judgment time	0.1 s	0 to 10000
	67	Deterioration diagnosis inertia ratio upper limit	%	0 to 10000
	68	Deterioration diagnosis inertia ratio lower limit	%	0 to 10000
	69	Deterioration diagnosis unbalanced load upper limit	0.1%	-1000 to 1000
	70	Deterioration diagnosis unbalanced load lower limit	0.1%	-1000 to 1000
	71	Deterioration diagnosis dynamic friction upper limit	0.1%	-1000 to 1000
	72	Deterioration diagnosis dynamic friction lower limit	0.1%	-1000 to 1000
	73	Deterioration diagnosis viscous friction upper limit	0.1%/ (10000 r/min)	0 to 10000
	74	Deterioration diagnosis viscous friction lower limit	0.1%/ (10000 r/min)	0 to 10000
	75	Deterioration diagnosis velocity setting	r/min	-20000 to 20000
	76	Deterioration diagnosis torque average time	ms	0 to 10000
	77	Deterioration diagnosis torque upper limit	0.1%	-1000 to 1000
	78	Deterioration diagnosis torque lower limit	0.1%	-1000 to 1000

(Note 1) When using this setup value at a value smaller than the default value, confirm that it matches the user's power supply environment.

13.3.7 Class 6: Special Setting 1

Class	No.	Parameter name	Unit	Setting range
6	02	Speed deviation excess setup	r/min	0 to 20000
	05	Position control 3rd gain effective time	0.1 ms	0 to 10000
	06	Position control 3rd gain scale factor	%	50 to 1000
	07	Additional value to torque command	%	-100 to 100

Class	No.	Parameter name	Unit	Setting range
	08	Torque compensation value in positive direction	%	-100 to 100
	09	Torque compensation value in negative direction	%	-100 to 100
	10	Function expansion setup	-	-32768 to 32767
	11	Current response setup	%	10 to 100
	14	Immediate stop time at the time of alarming	ms	1000
	15	2nd over-speed level setup	r/min	0 to 20000
	18	Power turn-on wait time	0.1 s	0 to 100
	22	A, B phase external scale pulse output method selection	-	0 to 1
	23	Load fluctuation correction gain	%	-100 to 100
	24	Load fluctuation correction filter	0.01 ms	10 to 2500
	27	Alarm latch time selection	-	0 to 3
	31	Real time auto tuning estimation speed	-	0 to 3
	32	Real time auto tuning custom setup	-	-32768 to 32767
	34	Hybrid vibration suppression gain	-	0 to 30000
	35	Hybrid vibration suppression filter	0.1/s	0 to 32000
	36	Dynamic brake operation input setup	0.01 ms	0 to 1
	37	Oscillation detecting level	-	0 to 1000
	38	Alarm mask setup	0.1%	-32768 to 32767
	39	Alarm mask setup 2	-	-32768 to 32767
	41	1st damping depth	-	0 to 1000
	42	Two-stage torque filter time constant	-	0 to 2500
	43	Two-stage torque filter damping term	0.01 ms	0 to 1000
	47	Function expansion settings 2	-32768 to 32767	-32768 to 32767
	48	Adjustment filter	0 to 2000	0 to 2000
	49	Command response filter / adjustment filter damping term setting	0 to 99	0 to 99
	50	Viscous friction compensation gain	0.1%/ (10000 r/min)	0 to 10000
	51	Immediate stop completion wait time	ms	0 to 10000
	57	Torque saturation error protection detection time	ms	0 to 5000
	60	2nd damping depth	-	0 to 1000
	61	1st resonance frequency	0.1 Hz	0 to 3000

Class	No.	Parameter name	Unit	Setting range
	62	1st resonance damping ratio	-	0 to 1000
	63	1st anti-resonance frequency	0.1 Hz	0 to 3000
	64	1st anti-resonance damping ratio	-	0 to 1000
	65	1st response frequency	0.1 Hz	0 to 3000
	66	2nd resonance frequency	0.1 Hz	0 to 3000
	67	2nd resonance damping ratio	-	0 to 1000
	68	2nd anti-resonance frequency	0.1 Hz	0 to 3000
	69	2nd anti-resonance damping ratio	-	0 to 1000
	70	2nd response frequency	0.1 Hz	0 to 3000
	71	3rd damping depth	-	0 to 1000
	72	4th damping depth	-	0 to 1000
	73	Load estimation filter	0.01 ms	0 to 2500
	74	Torque compensation frequency 1	0.1 Hz	0 to 5000
	75	Torque compensation frequency 2	0.1 Hz	0 to 5000
	76	Load estimation count	-	0 to 8
	85	Retracting operation condition setting	-	-32768 to 32767
	86	Retracting operation alarm setting	-	-32768 to 32767
	88	Absolute multi-rotation data upper limit	-	0 to 65534
	97	Function expansion setting 3	-	-2147483648 to 2147483647
	98	Function expansion setting 4	-	-2147483648 to 2147483647

13.3.8 Class 7: Special Setting 2

Class	No.	Parameter name	Unit	Setting range
7	00	Display on LED	-	0 to 32767
	01	Display time setup upon power-up	100 ms	-1 to 1000
	03	Output setup during torque limit	-	0 to 1
	09	Correction time of latch delay 1	25 ns	-2000 to 2000
	10	Soft limit function	-	0 to 3
	11	Positive side software limit value	Command unit	-1073741823 to 1073741823
	12	Negative side software limit value	Command unit	-1073741823 to 1073741823
	13	Absolute home position offset	Command unit	-1073741823 to 1073741823

Class	No.	Parameter name	Unit	Setting range
	14	Main power OFF warning detection time	ms	0 to 2000
	15	Positioning adjacent range	Command unit	0 to 1073741823
	16	Torque saturation error protection frequency	No. of times	0 to 30000
	20	RTEX communication cycle setup	-	-1 to 12
	21	RTEX command updating cycle ratio setting	-	1 to 2
	22	RTEX function extended setup 1	-	-32768 to 32767
	23	RTEX function extended setup 2	-	-32768 to 32767
	24	RTEX function extended setup 3	-	-32768 to 32767
	25	RTEX speed unit setup	-	0 to 1
	26	RTEX continuous error warning setup	No. of times	0 to 32767
	27	RTEX accumulated error warning setup	No. of times	0 to 32767
	28	RTEX_Update_Counter error warning setup	No. of times	0 to 32767
	29	RTEX monitor select 1	-	0 to 32767
	30	RTEX monitor select 2	-	0 to 32767
	31	RTEX monitor select 3	-	0 to 32767
	32	RTEX monitor select 4	-	0 to 32767
	33	RTEX monitor select 5	-	0 to 32767
	34	RTEX monitor select 6	-	0 to 32767
	35	RTEX command setting 1	-	0 to 2
	36	RTEX command setting 2	-	0 to 2
	37	RTEX command setting 3	-	0 to 2
	38	RTEX_Update_Counter error protection setup	No. of times	0 to 32767
	41	RTEX function extended setup 5	-	-32768 to 32767
	78	Signal reading setting for latch trigger with stop function	-	0 to 3
	91	RTEX communication cycle extended setup	ns	0 to 2000000
	92	Correction time of latch delay 2	25 ns	-2000 to 2000
	93	Home return limit speed	r/min	0 to 20000
	95	Number of RTEX continuous communication error protection 1 detections	No. of times	0 to 17
	96	Number of RTEX continuous communication error protection 2 detections	No. of times	0 to 17

Class	No.	Parameter name	Unit	Setting range
	97	Number of RTEX communication timeout error protection detections	No. of times	0 to 17
	98	Number of RTEX cyclic data error protection 1 / 2 detections	No. of times	0 to 17
	99	RTEX function extended setup 6	-	-32768 to 32767
	108	RTEX communication synchronization setup	-	0 to 7
	110	RTEX function extended setup 7	-	-2147483648 to 2147483647
	111	Trigger signal allocation setting of latch mode with stop function	-	0 to 64
	112	Selection of RTEX communication status flag	-	0 to 1

13.3.9 Class 8: Special Setting 3

Class	No.	Parameter name	Unit	Setting range
8	01	Profile linear acceleration constant	10000 Command unit/s 2	1 to 429496
	04	Profile linear deceleration constant	10000 Command unit/s 2	1 to 429496
	10	Amount of travel after profile position latch detection	Command unit	-1073741823 to 1073741823
	12	Profile home return position mode setup	-	0 to 1
	13	Profile home return velocity 1	Command unit/s or r/min	0 to 2147483647
	14	Profile home return velocity 2	Command unit/s or r/min	0 to 2147483647
	17	Relative movement of retracting operation	Command unit	-2147483648 to 2147483647
	18	Retracting operation speed	Command unit/s or r/min	0 to 2147483647

13.4 Monitor Commands

These commands are specified with RTEX_ReadAmpData (amplifier monitor).

<table>
<tr><th colspan="2">Type_Code
(Note 1) (Note 3)</th><th colspan="2" rowspan="2">Name</th><th rowspan="2">Index
(Note 2)</th><th rowspan="2">Unit</th><th rowspan="2">Description</th></tr>
<tr><th>A4N compatible</th><th>Standard</th></tr>
<tr><td>101h</td><td>01h</td><td>Position deviation</td><td>PERR</td><td>0
(1,2)</td><td>Command unit</td><td><In position control mode>
Position deviation
<In full-closed control mode>
External scale deviation
* The computation method (reference) of position deviation and external scale deviation is set in bit 14 of Pr 7.23 “Command position deviation output switching”.
<table><tr><th>r7.23 bit14</th><th>Computation method of positional deviation</th></tr><tr><td>0</td><td>Deviation from the command after filtering</td></tr><tr><td>1</td><td>Deviation from the command before filtering</td></tr></table><In speed / torque control mode>
Undefined
Note: Although the same data is returned whether Index is 1 or 2, use Index = 0.</td></tr>
<tr><td>102h</td><td>02h</td><td>Encoder resolution</td><td>-</td><td>0</td><td>pulse/r</td><td>Encoder resolution of the motor connected</td></tr>
<tr><td>104h</td><td>04h</td><td>Command position
(after filtering)</td><td>MPOS</td><td>0</td><td>Command unit</td><td>Command position (after filtering)</td></tr>
<tr><td>105h</td><td>05h</td><td>Actual speed</td><td>ASPD</td><td>0</td><td>Set the unit through Pr 7.25.</td><td>Motor actual speed
* Set the unit through Pr 7.25 “RTEX speed unit setup”.
<table><tr><th>Pr7.25</th><th>Unit</th></tr><tr><td>0</td><td>[r/min]</td></tr><tr><td>1</td><td>[Command unit/s]</td></tr></table></td></tr>
<tr><td>106h</td><td>06h</td><td>Internal command torque</td><td>TRQ</td><td>0</td><td>0.1%</td><td>Command torque to motor</td></tr>
<tr><td>-</td><td>07h</td><td>Actual position</td><td>APOS</td><td>0</td><td>Command unit</td><td>Motor actual position
* Position of the external scale in full-closed mode</td></tr>
<tr><td>-</td><td>08h</td><td>Internal command position (before filtering)</td><td>IPOS</td><td>0</td><td>Command unit</td><td>Internal command position before filtering</td></tr>
</table>

Type_Code (Note 1) (Note 3)		Name		Index (Note 2)	Unit	Description
A4N compatible	Standard					
-	09h	Latch position 1	LPOS1	0	Command unit	Motor actual position latched in CH1
-	0Ah	Latch position 2	LPOS2	0	Command unit	Motor actual position latched in CH2
-	0Ch	Command velocity (after filtering)	MSPD	0	Set the unit through Pr 7.25.	Command velocity after filtering * Set the unit through Pr 7.25 “RTEX speed unit setup”. Pr7.25 / Unit: 0 / [r/min]; 1 / [Command unit/s] * The value is undefined in torque control mode.
-	0Dh	External scale position (Note 4)	EXPOS	0	Pulse (External scale)	External scale position

(Note 1) When a Type_Code error occurs, command error (0031h) will be returned.

Manufacturer will use a Type_Code not listed above.

When a Type_Code used by the manufacturer is set, undefined value will be returned in place of command error (0031h).

(Note 2) When an Index error occurs, command error (0032h) will be returned.

(Note 3) A4N compatible: Type_Code compatible with A4N series can be used, but only with main commands.

Standard: Type_Code newly created for A5N and A5N series and can be used with both main commands and subcommands. When using with main commands, set leftmost 4 bits to 0.

* Although the product supports the A4N compatible Typer_Code to maintain compatibility, basically use the standard Type_Code.

(Note 4) The version before the function extended version 1 is not supported.

Type_Code		Name		Index	Unit	Description
A4N compatible	Standard					
111h	11h	Regenerative load ratio	-	0	% (Note 2)	Ratio of the regenerative overload protection to the alarm occurrence level
112h	12h	Overload ratio	-	0	0.1%	Ratio of the actual load to the rated motor load
-	21h	Logical input signal	-	0	-	Logic level of input signal
-	22h	Logical output signal	-	0	-	Logic level of output signal
-	23h	Logical input signal	-	0	-	Logic level of input signal (expansion portion)

Type_Code		Name		Index	Unit	Description
A4N compatible	Standard					
		(expansion portion)				
-	24h	Logical output signal (expansion portion)	-	0	-	Logic level of output signal (expansion portion)
-	25h	Physical input signal	-	0	-	Physical level of input signal
-	26h	Physical output signal	-	0	-	Physical level of output signal
131h	31h	Inertia ratio	-	0	%	The ratio of load inertia to the motor's rotor inertia (equivalent of value in Pr 0.04) Inertia ratio = (load inertia / rotor inertia) × 100
132h	32h	Automatic motor recognition	-	0	-	0: Invalid 1: Valid
133h	33h	Cause of no revolution	-	0	-	The number which shows the cause that the motor is not running.
134h	34h	Warning flags	-	0	-	The flag that shows the state of the waring currently occurring. * The corresponding bit is set to 1 to activate the flag (showing warning status).
-	37h	Multiple alarm occurrences /Warning information(Note 1)	-	Refer to Section 6-9-6.	-	Information of all the alarms or warnings currently occurring
201h	41h	Mechanical angle (Single turn data)	-	0	pulse	The mechanical angle (one revolution data of an absolute encoder) of the motor * The polarity is fixed and data increases at CCW rotation. One revolution data = 0 to (Encoder resolution - 1) .
202h	42h	Electrical angle	-	0	0.7031°	Motor electrical angle * The polarity is fixed and data increases at CCW rotation. Electrical angle = 0 to 1FF [Hex]
-	43h	Multi-turn data	-	0	Turn	Multi-turn data of the absolute encoder * In the incremental mode (Pr 0.15 = 1), multi-turn data becomes an indefinite value.
-	44h	Encoder status(Note 1)	-	0	-	The status of the encoder
-	47h	Encoder pulse	-	0	pulse	The sum of encoder feedback pulses

Type_Code		Name		Index	Unit	Description
A4N compatible	Standard					
		sum(Note 1)				
-	48h	External scale pulse sum(Note 1)	-	0	Pulse (External scale)	The sum of external scale feedback pulses
-	49h	External scale absolute position(Note 1)	-	0	Pulse (External scale)	The absolute position of the external scale
-	61h	Power on cumulative time	-	-	30 min	Cumulative on-time of control power to the servo amplifier * Because the power ON time is recored in unit of 30 minutes, a turn-on period shorter than 30 minutes is not recorded in the cumulative on-time. not recorded in the cumulative on-time.

(Note 1) The version before the function extended version 1 is not supported.

(Note 2) Be careful that the unit is different from that used for A4N and A5N. (A4N, A5N: [0.1%], A6N: [%])

* With the function extended version 3 or higher, the unit can be changed through bit 7 of Pr 7.99.

Pr7.99 bit7 0: [%], 1: [0.1%]

Type_Code		Name		Index	Unit	Description
A4N compatible	Standard					
-	62h	Servo amplifier temperature	-	-	°C	Temperature inside the servo amplifier
-	63h	Encoder temperature	-	-	°C	Temperature inside the encoder * Applicable only to 23-bit encoder. 0 for unsupported encoder.
-	64h	Number of inrush resistance relay operations	-	-	Cycle	Operating cycles of inrush current suppression resistor relay * Saturation will occur at maximum value of 40000000h. * Because the power ON time is recored in unit of 30 minutes, a turn-on period shorter than 30 minutes is not recorded in the cumulative cycles.
-	65h	No. of dynamic brake operations	-	-	Cycle	Number of operations of dynamic brake relay * Saturation will occur at maximum value of 40000000h. * Because the power ON time is recored in unit of 30 minutes, a turn-on period shorter than 30 minutes is not recorded in the cumulative time.
-	66h	Fan operating time	-	-	30 min	Operating time of cooling fan * Because the power ON time is recored in unit of 30 minutes, a turn-on period

Type_Code		Name		Index	Unit	Description
A4N compatible	Standard					
						shorter than 30 minutes is not recorded in the cumulative time. * 0 when no fan is installed.
-	67h	Fan life expectancy	-	-	0.1%	Percent of fan life expectancy * Because the power ON time is recored in unit of 30 minutes, a turn-on period shorter than 30 minutes is not recorded in the cumulative time. * 0 when no fan is installed.
-	68h	Capacitor life expectancy	-	-	0.1%	Percent of life expectancy of main power source capacitor * Because the power ON time is recored in unit of 30 minutes, a turn-on period shorter than 30 minutes is not recorded in the cumulative time.
-	69h	Voltage across PN	-	-	V	Main power source PN voltage
-	6Ch	Consumed power of motor (Note 1)	-	-	W	Momentary power consumption of the motor
-	6Dh	Motor power consumption (Note 1)	-	-	Wh	Power consumption of the motor
-	6Eh	Cumulative motor power consumption (Note 1)	-	-	Wh	Cumulative value of motor power consumption
401h	71h	RTEX Cumulative communication errors	-	0	Cycle	Cumulative number of RTEX communication errors * Saturation will occur at maximum value of FFFFh. The count will be cleared upon restarting of servo amplifier or resetting of control power source.
-	77h	RTEX UpdateCounter cumulative error count(Note 1)	-	0	Cycle	Cumulative number of communication errors of RTEX UpdateCounter * Saturation will occur at maximum value of 7FFFh. The count will be cleared upon restarting of servo amplifier or resetting of control power source.
-	78h	RTEX communication Cumulative RTEX communication timeout errors(Note 1)	-	0	Cycle	Cumulative number of RTEX communication data reception interruption errors * Saturation will occur at maximum value of FFFFh. The count will be cleared upon restarting of servo amplifier or resetting of control power source.

Type_Code		Name		Index	Unit	Description
A4N compatible	Standard					
411h	81h	Encoder cumulative communication errors	-	0	Cycle	Cumulative number of communication errors between encoders * Saturation will occur at maximum value of FFFFh. The count will be cleared upon restarting of servo amplifier or resetting of control power source.

(Note 1) The version before the function extended version 1 is not supported.

Type_Code		Name		Index	Unit	Description
A4N compatible	Standard					
413h	83h	External scale cumulative communication errors(Note 1)	-	0	Cycle	Cumulative number of communication errors between external scales * Saturation will occur at maximum value of FFFFh. The count will be cleared upon restarting of servo amplifier or resetting of control power source.
-	84h	External scale abnormal communication data errors (Note 1)	-	0	Cycle	Cumulative number of communication data errors in communication between external scales * Saturation will occur at maximum value of FFFFh. The count will be cleared upon restarting of servo amplifier or resetting of control power source.
-	85h	For manufacturer's use	-	-	-	-
-	86h	Hybrid position deviation (Note 1)	-	-	Command unit	Tolerance between encoder position and external scale position
-	87h	External scale data(Note 1) (Leftmost 24 bits)	-	0	Pulse (External scale)	Rightmost 24 bits of external scale data
-	88h	External scale data(Note 1) (Rightmost 24 bits)	-	0	Pulse (External scale)	<Virtual full-close control mode function disabled> Leftmost 24 bits of external scale data is output. <Virtual full-close control mode function enabled> ● When an AB-phase output type scale is connected, position data (16 bits) is output that is set to 0 when the power is turned ON. Note that it is not affected by Pr 3.26 Reversal of direction.

Type_Code		Name		Index	Unit	Description
A4N compatible	Standard					
						● When a serial incremental scale is connected, position data (24 bits) of the serial incremental scale is output. Note that the data output is position data affected by Pr 3.26 Reversal of direction.
-	89h	External scale status(Note 1)	-	0	-	Status of external scale
-	A1h	Velocity control command (Note 1)	-	0	r/min	Velocity control command
-	A5h	Internal position command speed(Note 1)	-	0	r/min	Internal position command speed
-	A6h	Speed deviation(Note 3)	-	0	r/min	Speed deviation
-	A8h	Positive direction torque limit value(Note 1)	-	0	0.05%	Positive direction torque limit value
-	A9h	Negative direction torque limit (Note 1)	-	0	0.05%	Negative direction torque limit value
-	AAh	Speed limit value (Note 1)	-	0	r/min	Speed limit value
-	ABh	Gain switching flag (Note 1)	-	0	-	Gain switching flag
-	B1h	Deterioration diagnosis state (Note 1)	-	0	-	Deterioration diagnosis state
-	B2h	Deterioration diagnosis torque average time (Note 1)	-	0	0.1% (Note 2)	Deterioration diagnosis torque command average time
-	B3h	Deterioration diagnosis torque command standard value (Note 3)	-	0	0.1%	Deterioration diagnosis torque command standard value
-	B4h	Deterioration diagnosis inertia ratio estimate(Note 1)	-	0	%	Deterioration diagnosis inertia ratio estimate

(Note 1) The version before the function extended version 1 is not supported.

(Note 2) Be careful that the unit is different from the one of the data displayed on the setup support software (PANATERM).

(Note 3) The version before the function extended version 2 is not supported.

Type_Code		Name		Index	Unit	Description
A4N compatible	Standard					
-	B5h	Deterioration diagnosis unbalanced load estimate (Note 1)	-	0	0.1% *2)	Deterioration diagnosis unbalanced load estimate
-	B6h	Deterioration diagnosis unbalanced load estimate (Note 1)	-	0	0.1% *2)	Deterioration diagnosis unbalanced load estimate
-	B7h	Deterioration diagnosis unbalanced load estimate (Note 1)	-	0	0.1%/(10000 r/min) *2)	Deterioration diagnosis unbalanced load estimate
-	FAh	Monitor flag (Note 1)	-	0	-	Various flag information of the servo amplifier The contents of Monitor_Data, the response data, are as follows.

Byte	bit	Description
12, 20	7 to 0	For manufacturer's use
13, 21	7 to 0	For manufacturer's use
14, 22	7 to 6	For manufacturer's use
	5	Semi-closed / full-closed selection state 0: Semi-closed 1: Full-closed
	4	Incremental / absolute selection state 0: Incremental mode 1: Absolute mode
	3 to 0	For manufacturer's use
15, 23	7 to 0	For manufacturer's use

(Note 1) The version before the function extended version 2 is not supported.

(MEMO)

Revision History

The manual code is shown at the bottom of the cover page.

Date of issue	Manual No.	Revision details
February 2021	WUME-GM1PGR-01	1st edition
August 2021	WUME-GM1PGR-02	2nd Edition Added instructions related to EtherCAT. Changed PMC_ReadLatchPosition parameters. Added instructions related to PID control. Added instructions related to the GM1 Pulse Output Unit.
March 2022	WUME-GM1PGR-03	3rd Edition Clerical corrections Added function block argument for pulse output unit ● Added switching of P-point / E-point control for positioning control ● Added specification of creep speed to home return
April 2022	WUME-GM1PGR-04	4th Edition ● Changed the company name.
June 2022	WUME-GM1PGR-05	5th Edition ● Added instructions related to CNC control. ● Added instructions related to motor setting. ● Added instructions related to recipe manager. ● Added description of ladder instruction execution box. ● Clerical corrections

(MEMO)

(MEMO)

Please contact .....

Industrial Device Business Division,
Panasonic Industry Co., Ltd.
7-1-1 Morofuku, Daito City, Osaka, 574-0044, Japan
industrial.panasonic.com/ac/e/

WUME-GM1PGR-05